

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

БОЙКО УСТИМ ВІКТОРОВИЧ

Допускається до захисту:

завідувач кафедри

інформаційних технологій,

доктор технічних наук, доцент

_____ Т. В. Нескородева

« ____ » _____ 20__ р.

**РОЗРОБКА ГРИ ДЛЯ ВИВЧЕННЯ МОВИ С# У ВИГЛЯДІ ВЕБ-
ДОДАТКУ**

Спеціальність 122 «Комп'ютерні науки»

Кваліфікаційна (бакалаврська) робота

Керівник:

Антонов Ю. С., доцент кафедри

інформаційних технологій,

к.ф.-м.н., доцент

Оцінка: _____ / _____ / _____

(бали за шкалою ЄКТС/за національною шкалою)

Голова ЕК: _____

(підпис)

АНОТАЦІЯ

Бойко У. В. Розробка гри для вивчення мови C# у вигляді веб-додатку. Спеціальність 122 «Комп'ютерні науки», освітня програма «Сучасні інформаційні технології та програмування». Донецький національний університет імені Василя Стуса, Вінниця, 2022.

У кваліфікаційній (бакалаврській) роботі досліджено поняття веб-додатку, принципи та актуальність їх створення. Описано основні технології для розробки під веб та їх популярні надбудови. За допомогою технологій HTML та CSS, препроцесора SASS, мови програмування JavaScript, бібліотеки React було розроблено веб-додаток, який допомагатиме його користувачам вивчати мову програмування C#.

Ключові слова: веб-додаток, веб-розробка, JavaScript, React.

51 с., 21 рис., 42 джерела.

ABSTRACT

Boyko U.V. Development of a game is for learning the language of C# as a web-addition. Specialty is 122 "Computer sciences", educational program "Modern information technologies and programming". Vasyl' Stus Donetsk National University, Vinnitsa, 2022.

In qualifying (to the bachelor) work a concept is investigational to web-addition, principles, and actuality of their creation. Basic technologies are described for development under a web and them popular building on. By means of technologies of HTML and CSS, preprocessor of SASS, programming of JavaScript, library of React languages were worked out web-addition that helps users to study a programming of C# language.

Keywords: web-addition, web-development, JavaScript, React.

51 p., 21 pict., 42 source.

ЗМІСТ

ВСТУП	5
РОЗДІЛ 1	7
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	7
1.1 Поняття веб-сайту та веб-додатку	7
1.2 Основи відмінності веб-додатку від веб сайту	7
1.3 Переваги та недоліки веб-додатків	9
1.3.1 Переваги створення веб-додатків	9
1.3.2 Недоліки створення веб-додатків	10
1.4 Принцип роботи веб-додатків	11
1.5 Види веб-додатків	13
1.5.1 SPA.....	14
1.5.2 MPA	15
1.5.3 PWA	16
1.6 Розгляд існуючих аналогів	17
Висновок до першого розділу	20
РОЗДІЛ 2	21
ВИКОРИСТАНІ ТЕХНОЛОГІЇ	21
2.1 HTML	21
2.2 CSS	23
2.3 Sass	26
2.4 JavaScript	29
2.5 React	32
2.6 Visual Studio Code	34
2.7 GitHub Pages	36
Висновок до другого розділу	37
РОЗДІЛ 3	38
РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКУ ДЛЯ ВИВЧЕННЯ C#	38
3.1 Постановка задачі	38
3.2 Реалізація веб-додатку	38
Висновок до третього розділу	44

ВИСНОВКИ	45
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ.....	46



ВСТУП

На сьогоднішній день уявити наше життя без комп'ютерних технологій неможливо. Використовуючи складну електронну операційну техніку, багато людей не замислюється, що в її основі лежить програмування. Воно відіграє важливу роль в сучасному світі.

Програмування означає створення конкретних програм або інструкцій на певній мові програмування [1]. На сьогодні в світі існує близько 10 тисяч мов програмування. Сфера їх застосування дуже широка, починаючи з веб-програмування, розробки ігор та різних додатків на андроїд чи iOS, закінчуючи розробкою програмного забезпечення, нейронних мереж та аналізом даних [2].

Однією з найпопулярніших і затребуваних мов програмування є C#. Це об'єктно-орієнтована мова програмування, створена для роботи у середовищі Microsoft .NET Framework [3].

Згідно з рейтингом мов програмування, за версією DOU, C# у 2022-му році, обійшовши мову програмування Java, посіла друге місце за популярністю в Україні [4].

Актуальність вивчення C# зумовлена постійним її розвитком. C# є гарним вибором для розробників-початківців, адже вона легка у вивченні і має перспективу на майбутнє завдяки масштабним оновленням з додаванням корисних функцій, які періодично робить компанія Microsoft. Спеціалістів, які нею володіють можна з впевненістю назвати затребуваними та високооплачуваними, адже, коли мова йде про C#, то майже завжди мають на увазі коштовні серйозні проекти.

Мова програмування C# широко застосовується. Зокрема для створення корпоративного програмного забезпечення, мобільних додатків, розробці ігор

на Unity, проведення складних експериментальних розрахунків, розпізнавання образів тощо [5].

Завдяки розвитку інформаційних технологій кожен хто має бажання чи потребу може вивчити мову програмування C# просто вдома. Для цього існують різноманітні способи, такі як, додатки на мобільні телефони, сайти, книги та інші. Але зазвичай кожен з них має свої недоліки, що вповільняє процес вивчення або робить його нудним та непродуктивним.

Існує чи мало курсів та тренінгів, за допомогою яких можна опанувати C#. Але зазвичай вони коштують грошей, що і відлякує зацікавлених у вивченні людей.

Мета бакалаврської роботи полягає: в розгляді комплексного підходу створення веб-додатків і власне в розробці гри для вивчення мови C# у вигляді веб-додатку.

Задачами дослідження є:

1. розгляд основних принципів та технологій для побудови веб-додатків;
2. розгляд комплексного підходу для створення веб-додатків;
3. обґрунтування процесу створення власного веб-додатку;
4. створення веб-додатку для вивчення C# у вигляді гри.

Об'єктом дослідження є веб-додаток для вивчення мови C#.

Предмет дослідження – веб розробка.

Зв'язок роботи з науковими програмами, планами, темами. Наведені у бакалаврській роботі дослідження пов'язані з фундаментальною науково-дослідною роботою «Дослідження та комп'ютерно-математичне моделювання складних систем та процесів у науці, освіті та інформаційно-комунікаційній діяльності підприємств» (№ держреєстрації 0116U002394, 2018-2022 рр.).

Структура бакалаврської роботи – робота складається зі вступу, трьох розділів зі своїми підрозділами, висновків, списку використаних посилань.

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Поняття веб-сайту та веб-додатку

Веб-сайт – це сукупність глобально доступних взаємопов’язаних веб-сторінок, в яких єдине доменне ім’я [6]. Зазвичай складаються з простої архітектури на основі HTML, здебільшого з розміщеним на собі набором інформації у вигляді тексту, зображень, фото чи відео. Вони несуть інформативний характер і не передбачають взаємодію з користувачем. Наприклад, звичайний веб-сайт може містити інформацію про певний товар, але здійснити замовлення чи його оплату користувач не зможе. Яскравими прикладами є різноманітні блоги, сайти новин, кулінарії та погоди [7].

Веб-додаток – це комп’ютерна програма, яка виконує певну функцію і яку можна відкрити за допомогою будь-якого браузера, використовуючи його як свого клієнта. Веб-додаток відображається на екранах користувачів як веб-сторінка, на якій не лише розміщується певна інформація, але й існує передбачений функціонал, що дозволяє взаємодіяти з користувачем, виконуючи конкретні дії, для отримання необхідного результату. Dodatok може мати різну складність, починаючи від контактної форми на веб-сайті, закінчуючи текстовим процесором [8].

1.2 Основі відмінності веб-додатку від веб сайту

Веб-додаток та веб-сайт здаються дуже схожими поняттями, проте варто зауважити, що вони мають різний функціонал та компоненти розробки.

Відмінності веб-додатку від веб-сайту:

- Інтерактивність. В той час як веб-сайти містять візуальний контент, взаємодіяти з яким користувач не може, веб-додатки дають користувачу можливість не лише переглядати, але й змінювати, додавати, маніпулювати інформацією на сторінці;
- Автентифікація. Більшість веб-додатків вимагають автентифікації, щоб користувач міг скористатися ним, в той час як для сайтів обов'язкова автентифікація не потрібна;
- Функції, користувацький інтерфейс, завдання веб-додатків на порядок складніші;
- Інтегрованість. Розробники можуть інтегрувати веб-додатки і веб-сайти з програмами, включаючи ERM, CRM. Однак в більшості випадків процес інтеграції проходить саме з веб-додатками, оскільки їхнім складним функціям зазвичай потрібна додаткова інформація від сторонніх систем [9];
- Веб-додатки є більш ресурсномісткими, тому що можуть взаємодіяти з користувачем і виконувати різноманітні дії;
- Веб-додаток може бути складовою частиною веб-сайту або окремим ресурсом;
- Веб-додаток здебільшого має підключення до бази даних [10];
- Деплоймент. Щоб внести прості зміни в сайт, не потрібна повна компіляція – тільки оновлення HTML-коду. А для змін у веб-додатку потрібно робити рев'ю всього коду і після вписувати, те що потрібно змінити [11].

В сучасному світі майже не залишилось веб-сайтів які не включають в себе елементи інтерактивності. Натомість багато веб-додатків включають в себе пошук інформації. Так чи інакше, веб-сайти залишаються інформаційними джерелами, в той час як веб-додатки користувацькими інструментами [12].

В розробці сайту немає нічого складного. Але для того, щоб створити веб-додаток потрібні більш глибокі знання, навички і досвід.

1.3 Переваги та недоліки веб-додатків

1.3.1 Переваги створення веб-додатків

Веб-додатки не потребують попереднього встановлення на пристрій користувача. Все, що потрібно для його використання – це веб-браузер і доступ в інтернет. Це означає економію пам'яті на пристрої і непотрібність платити кошти за встановлення додатку, оскільки вони працюють через просту URL-адресу.

На відміну від нативних додатків, веб-додаток може використовуватися з будь-якою операційною системою на усіх пристроях.

Веб-додатки не вимогливі і не пред'являють ніяких вимог до апаратної платформи.

Встановлюючи додаток на свій пристрій, користувач бере на себе обов'язки адміністратора, оскільки йому потрібно його встановити, запустити, налаштувати під себе, крім того можуть виникнути не зрозумілі помилки. У випадку з веб-додатком, який фактично лежить на сервері, з цим проблем не буде [13].

Немає ніяких проблем з підтримкою нових версій програм, оскільки веб-додаток оновлюється централізовано. Це суттєво оптимізує процес обслуговування, модернізації інтерфейсу. При завантаженні сторінки всі оновлення автоматично активуються[14]. Існує лише одна версія додатку з якою працюють всі користувачі і у випадку з виходом нової всі без винятку переходять на неї, інколи навіть не помічаючи цього.

Вся особиста інформація користувачів зберігається в хмарному сховищі, що забезпечує захист їхніх даних. Втратити дані, як це буває в разі пошкодження жорстких дисків, не є проблемою.

На сьогоднішній день, розвиток і надійність мережеских з'єднань і web-технологій знаходяться на високому рівні.

Високий рівень безпеки є характерним для веб-додатків, оскільки у системи єдина точка входу. Налаштувати і забезпечити захист не складно – це все робиться централізовано.

Веб-додатки дають можливість працювати в мережі та зберігати результати своєї роботи на сервері. Це робить користувачів веб-додатків мобільними, оскільки вони влюбий час злюбого місця, де тільки є вихід до інтернету, мають доступ до своїх даних.

1.3.2 Недоліки створення веб-додатків

Як вже було зазначено вище, для користування веб-додатком необхідне постійне підключення до інтернету. Це є проблемою, оскільки в нашій країні ще не всюди він є і не завжди його швидкість задовольняє умови комфортного його використання. Наприклад, під час подорожі, через нестабільне інтернет з'єднання ускладнюється робота з веб-додатками.

Крім того не всі додатки можна реалізувати в Web. Наприклад, складні трьох вимірні моделі в браузері створювати неможливо.

Багатьох користувачів бентежить той факт, що їхні дані будуть зберігатися та оброблятися десь на чужому сервері. Це може призвести до витоку інформації, через це не кожен ризикне викласти особисту інформацію до мережі.

Працюючи з веб-додатками, при їх перезавантаженні може виникнути затримка, викликана необхідністю встановити HTTP-з'єднання, отримати оброблені дані з сервера і перезавантажити сторінку браузером. Це все займає час і призводить до переривчастої роботи додатку [15].

1.4 Принцип роботи веб-додатків

Веб-додатки працюють за принципом «клієнт-сервер». Клієнтом є браузер, який зв'язується з веб-сервером, на якому розміщується веб-додаток, через інтернет мережу. За командою користувача HTTP запит відправляється на сервер, де його обробляє серверний код і повертає відповідь клієнту. В залежності від типу формування сторінки, у відповіді може зберігатися як готова HTML-сторінка, так і її шаблон або дані, наприклад в форматі JSON.

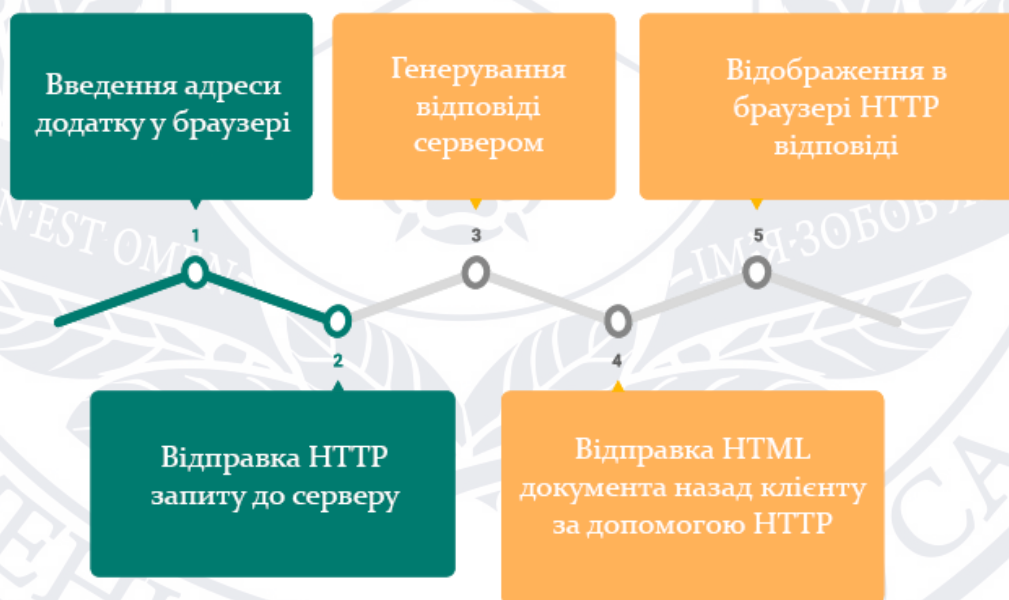


Рисунок 1.1 – Етапи взаємодії користувача з веб-додатком [16]

За клієнтську частину роботи веб-додатку відповідає frontend-розробка. Вона відповідає за створення візуальної частини додатку. Все, що користувач бачить та з чим може на пряму взаємодіяти, починаючи від дизайну,

закінчуючи елементами на сторінці, з якими він може проводити певні маніпуляції. Frontend включає в себе:

- Визначення структури веб-додатку. За це відповідає HTML, мова гіпертекстової розмітки, яка дозволяє описати структуру за допомогою HTML-тегів;
- Налаштування зовнішнього вигляду веб-додатку. За це відповідає CSS, каскадні таблиці стилів, які дозволяють визначити стиль тих елементів, які визначенні в HTML;
- Реалізація механізму взаємодії на стороні користувача. За це відповідає мова програмування JavaScript.

За серверну частину роботи веб-додатку відповідає backend-розробка. Вона відповідає за те, до чого користувач не має доступу та ніяк візуально оцінити не може. В ній описується логіка роботи веб-додатку на віддаленому сервері.

Коли користувач відкриває сторінку веб-додатку, то від нього надходить запит на сервер. На сервері цей запит обробляється і користувачу надходить згенерована веб-сторінка. Взаємодії зі сторінкою також ведуть до формування запитів на сервер, наприклад, пошуковий запит чи заповнення форми.

Серверна частина додатків розробляється на різних мовах програмування (Java, Python, C# та інші), або за допомогою backend-фреймворків і веб-сервісів.

Важливою частину backend є бази даних. База даних – це місце де зберігається дані, з якими можна легко взаємодіяти, керувати та обновляти, до яких можна звертатись. При запитах користувача ці дані можуть діставатися і відображатися у веб-додатку.

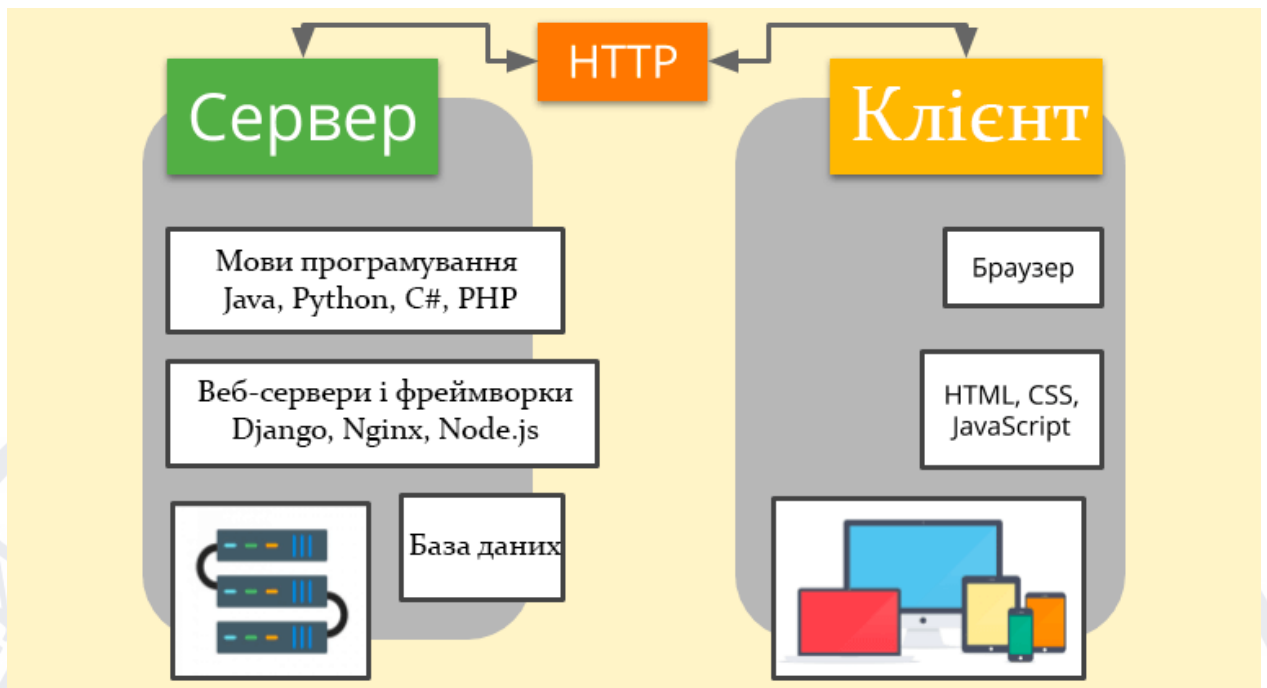


Рисунок 1.2 – Взаємодія серверної і клієнтської частини веб-додатку [16]

Щоб клієнт і сервер могли взаємодіяти одне з одним, застосовується HTTP-протокол.

HTTP – це протокол, який лежить в основі передачі і обміну даними у мережі інтернет. На сьогоднішній день він відповідає за передачу будь-яких даних у клієнт-серверному додатку [17].

1.5 Види веб-додатків

Веб-додатки можна класифікувати по різним критеріям: в залежності від їх функціоналу і призначення.

Існує три основних підходи в розробці веб-додатків:

- SPA (Single Page Application)
- MPA (Multi Page Application)
- PWA (Progressive Web Application)

1.5.1 SPA

SPA – це односторінкові веб-додатки, які містять в собі HTML-сторінку, що динамічно оновлюється в залежності від дій користувача [18]. Відвідувач одразу бачить в браузері весь основний контент, а нові дані з’являються при взаємодії з ним, по мірі необхідності, наприклад, при прогортюванні чи натиску на іконку. Сторінка повністю не перезавантажується, а лише підвантажує потрібні користувачеві елементи.

Односторінкові додатки схожі на десктопні, оскільки при переході на нову сторінку підвантажується лише частина контенту, що дозволяє не завантажувати одні і ті ж елементи багато разів. Це забезпечує швидку реакцію на дії користувача, без підвисань та затримок.

Переваги SPA:

- Висока швидкість – всі дані завантажуються при першому потраплянні на веб-додаток, а під час дій користувача SPA швидко оновлює потрібну частину контенту, що значно економить час;
- Гнучкість користувацького інтерфейсу – за рахунок того, що сторінка одна, її набагато простіше налаштовувати, зберігати дані про сеанс, розробити цікавий та інтерактивний дизайн інтерфейсу;
- Легкість створення – для розробки SPA додатку вже існують різні бібліотеки та фреймворки, може вестись паралельна робота frontend та backend частин додатку;
- Кешування даних – додаток робить запит, після чого дані підвантажуються і користувач може ним користуватися в офлайн режимі [19].

Недоліки SPA:

- Необхідна підтримка JavaScript – без JS немає можливості повноцінно користуватися функціоналом веб-додатку. У тих

користувачі які відключають скрипти, для збільшення швидкості завантаження додатку та позбавлення від небажаної реклами, додаток працювати не буде;

- Навантаження на веб-браузер – через те, що клієнтські фреймворки ресурсномісткі, воно доволі довго завантажуються;
- Погано піддаються SEO оптимізації – через те, що URL сторінок практично не міняється, а дані підвантажуються динамічно, а для оптимізації важлива стійкість і унікальні URL для кожної сторінки.
- Проблема витоку пам'яті JavaScript – яка зменшує швидкість завантаження і відкриває дані користувачів для дій злоумисників.

1.5.2 МРА

МРА – це багатосторінкові веб-програми, які зазвичай використовуються для створення складних систем. Вони працюють за традиційною схемою, як звичайні сайти. МРА відправляють запит на сервер і повністю перезавантажують сторінку, коли на ній відбуваються якісь зміни. Оскільки велика частина даних повторно завантажується на сторінку, обмін даними здійснюється повільніше, ніж у SPA.

Переваги МРА:

- Проста SEO оптимізація. Пошукові двигуни пристосовані для індексації багатосторінкових програм. Можна оптимізувати кожен сторінку додатку під потрібні запити користувачів;
- Звичність у використанні для користувачів, за рахунок зрозумілого інтерфейсу і класичної навігації;
- Масштабованість. У МРА можна вкласти велику кількість інформації, без обмежень за кількістю сторінок та функцій [20].

Недоліки МРА:

- Складність розробки. Frontend і backend тісно пов'язані, тому розробляти їх окремо не вийде. Фреймворки використовуються як на стороні серверу так і на стороні клієнта. Це все ускладнює роботу розробника і збільшує терміни розробки додатку;
- Швидкість взаємодії. МРА перезавантажують контент, коли користувач з ним взаємодіє.

1.5.3 PWA

PWA – це прогресивні веб-додатки близькі за своєю функціональністю, можливостями та якістю користувацького досвіду до нативних мобільних чи комп'ютерних додатків. PWA можуть встановлюватися на робочий стіл девайсу, відправляти push-сповіщення та працювати офлайн.

Треба розуміти, що такі додатки не є повноцінно нативними, вони мають все ті ж обмеження у веб-технологіях, але це чудова можливість перетворити свій сайт у встановлюваний додаток [21].

Переваги PWA:

- Кросплатформеність – можуть взаємодіяти з різними операційними системами;
- Висока швидкість встановлення;
- Відображення даних офлайн з миттєвим їх завантаженням;
- Дозволяють відправляти push-сповіщення;
- Легко конвертуються із вже існуючого сайти, потрібно лише трохи його змінити;
- Вони не потребують зайвої пам'яті на пристрої (адже мають розмір менше 1 мегабайту).

Недоліки PWA:

- Не всі браузеры підтримують основні функції таких додатків (наприклад, Firefox і Edge)
- На відміну від мобільних додатків, вони не розміщені в магазинах, наприклад, як Google Play чи AppStore.
- Вони споживають більше енергії ніж звичайні веб-сайти [22].

1.6 Розгляд існуючих аналогів

На сьогоднішній день існує безліч курсів та різних програм для вивчення мов програмування. В кожній є свої переваги та недоліки. Існує два добре обладнанні веб-сайти, що пропонують своїм користувачам курси з різних мов програмування, а саме SoloLearn та Codecademy.

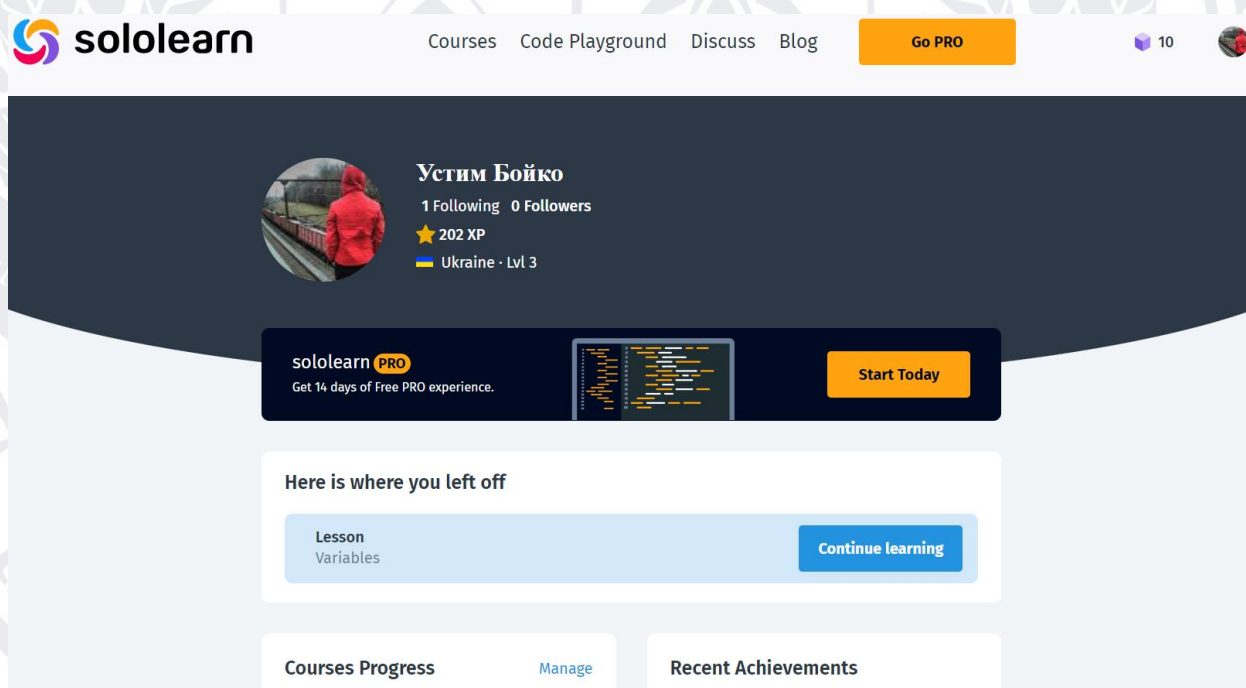


Рисунок 1.3 – Головний екран SoloLearn

SoloLearn – це платформа, що допомагає вивчити різні мови програмування. Вона є безкоштовною, проте існує також її професійна версія, яка надає додаткові функції за які треба платити. SoloLearn найбільше підходить для тих, хто має на меті вивчити саме основи програмування.

Платформа має зручний інтерфейс, яким легко користуватися. Всі запропоновані курси діляться на модулі, а ті в свою чергу на уроки[23].

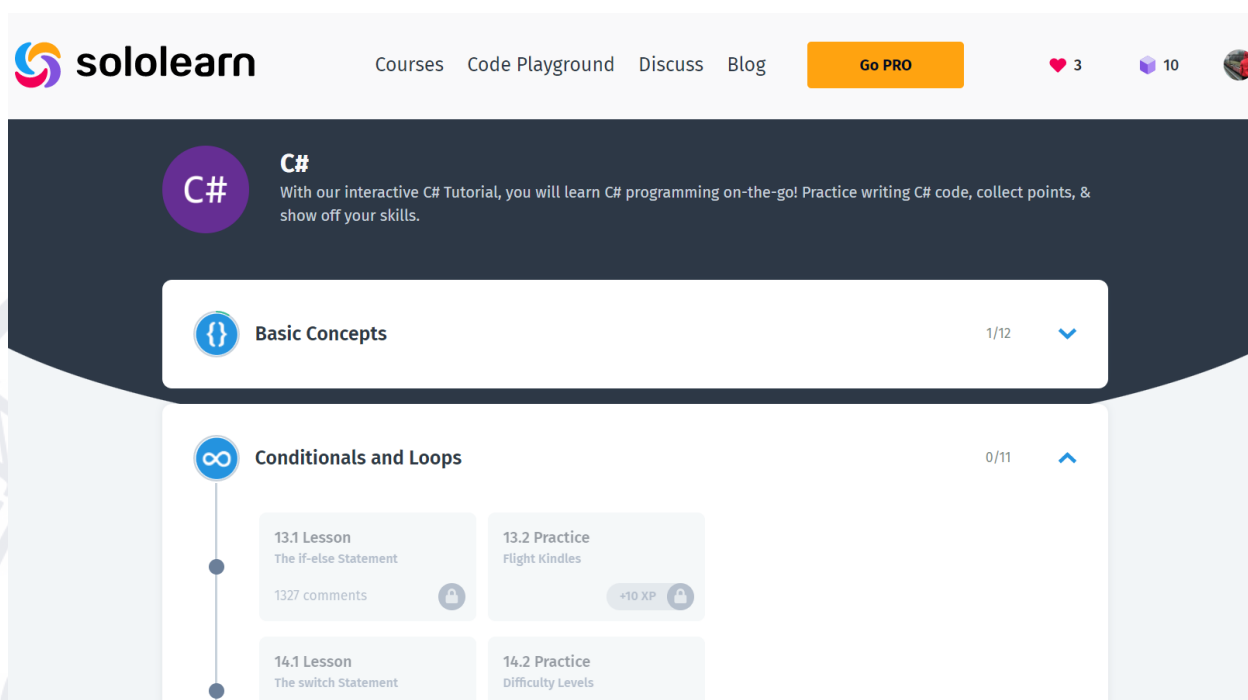


Рисунок 1.4 – Структура проходження курсу

Codecademy – це веб-сайт, що надає можливість навчитись програмуванню та кодуванню.

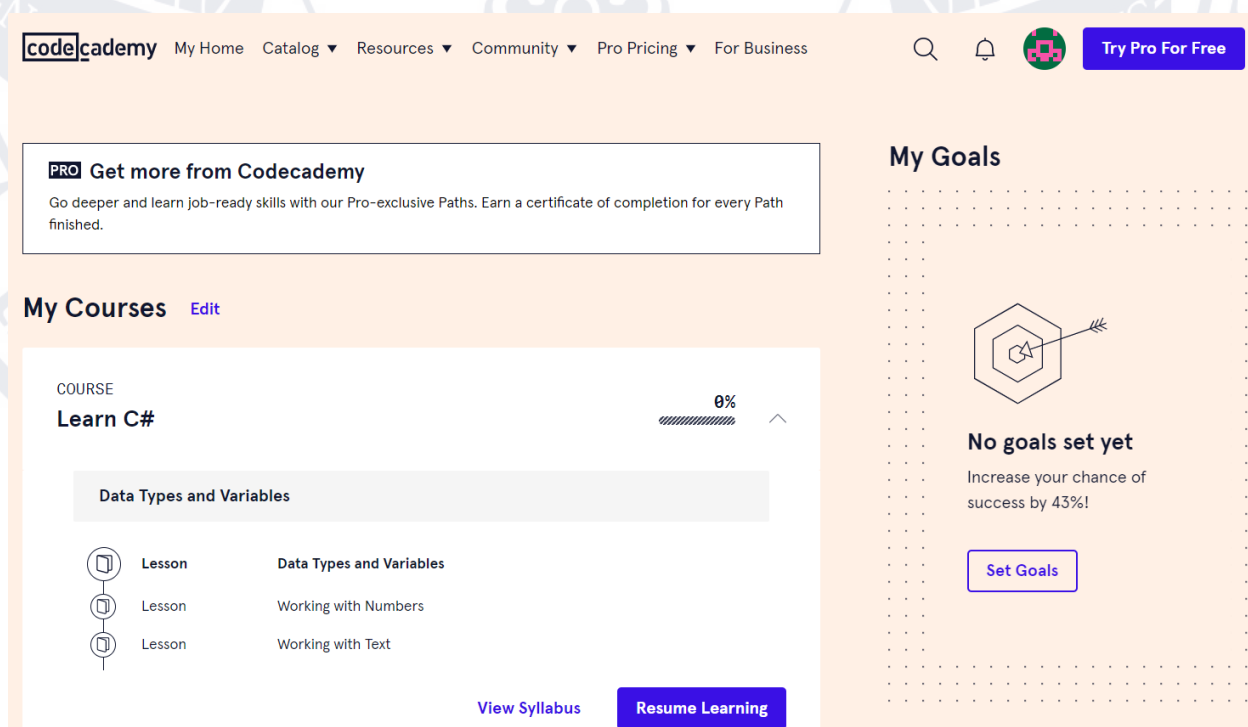


Рисунок 1.5 – Головний екран Codecademy

На відмінну від SoloLearn, де доступ до всіх курсів є безкоштовний, в Codecademy потрібно для цього потрібно підписатися на Pro версію.

Платформа надає доступ до курсів по веб-розробці, розробці ігор, дизайну, програмуванню і партнерству. Вона найкраще підходить для тих, хто хоче стати професіоналом в програмуванні, і для яких не проблема заплатити певну суму грошей, для більш якісного навчання.

Так само як і в SoloLearn курси поділені на модулі, а ті в свою чергу на теоретичні та практичні уроки[24].

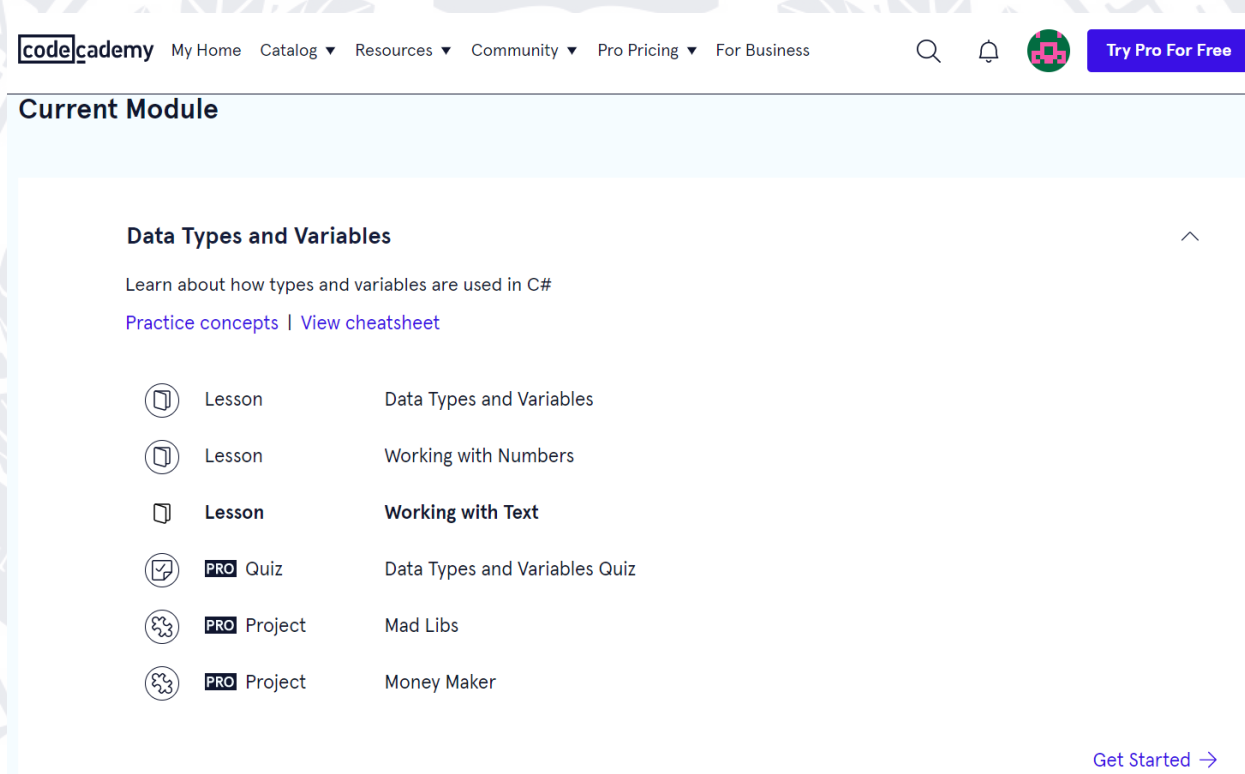


Рисунок 1.6 – Структура модуля одного з курсів Codecademy

Великим мінусом є те, що практичні завдання зазвичай не є безкоштовними.

SoloLearn як і Codecademy прекрасні платформи на яких можна вивчити не одну мову програмування. Проте в них є свої недоліки:

- Не має можливості навчатись українською мовою. Платформи мають англomовний інтерфейс, що не завжди є зручним, оскільки

не всі користувачі мають навички для комфортного вивчення програмування англійською мовою.

- Існування платного контенту. В SoloLearn, на відмінну від Codecademy, всі курси є безкоштовними, але щоб користуватись всім функціоналом сповна, потрібно заплатити за Pro версію. Це є недоліком, оскільки, зазвичай це відлякує користувачів і не кожен має можливість оплатити платний контент[25].

Висновок до першого розділу

В цьому розділі було розкрито поняття веб-додатку та веб-сайту, розглянуто їхні основні відмінності. Описано особливості веб-додатків, їхні переваги та недоліки. Розглянуто принцип роботи веб-додатків, їхні види та основні підходи при їх створенні. Проведено огляд існуючих аналогів, їхні особливості та недоліки. Наступний розділ буде присвячено технологіям, які будуть використовуватись для реалізації власного веб-додатку.

РОЗДІЛ 2

ВИКОРИСТАНІ ТЕХНОЛОГІЇ

2.1 HTML

HTML (Hypertext Markup Language) – це мова гіпертекстової розмітки, яка повідомляє браузеру, про відображення в ньому тексту, зображень та інших мультимедіа на веб-сторінці. Це текстовий підхід до опису структури вмісту, що зберігається у файлі HTML.

HTML – це офіційна рекомендація World Wide Web Consortium (W3C). W3C відповідає за підтримку та розробку сертифікацій HTML. HTML5 – остання версія сертифікації.



Рисунок 2.1 – Логотип HTML5

Документи HTML – це текстові файли, що містять певний синтаксис та правила найменування. Здійснити їх перегляд можна за допомогою будь-якого веб-браузера, який в свою чергу зчитує файл та відтворює його вміст.

Однією з умов HTML є оголошення типу документу на початку текстового файлу. Заголовок виглядає наступним чином: `<!DOCTYPE html>`. Він завжди має такий вигляд, без будь-якого ніякого вмісту всередині нього чи розривів. Вміст, як розміщується до заголовку з оголошенням типу не буде розпізнаватися веб-браузером як HTML.

Іншою важливою вимогою для створення HTML файлу є його збереження з розширенням .html. У той час як оголошення doctype сигналізує з середини про приналежність документу до HTML, то про приналежність зовні сигналізує його розширення. Це стає особливо важливим під час завантаження файлів в мережу інтернет, оскільки веб-сервер повинен знати, як взаємодіяти з файлами, перш ніж він зможе відправити їх, для прочитання внутрішнього вмісту, на клієнтський комп'ютер.

Кожна сторінка HTML складається з тегів, які створюють ієрархію та працюють як будівельні блоки. Теги можуть містити пари ім'я-значення, відомі як атрибути, а частини вмісту, що розміщуються в тегах, називається елементом HTML. Атрибути можуть надавати додаткову інформацію про елемент. Вони завжди повинні мати:

- Пробіл між собою і назвою елемента або попереднім атрибутом;
- Ім'я атрибуту після якого йде знак дорівнює;
- Значення атрибута взяте в лапки [26].

Більшість елементів HTML мають відкриваючі та закриваючі теги. Відкриваючий тег, на прикладі тегу абзацу, виглядає так: <p>. Так само, за виключенням зворотної косої риски, що вказує на кінець елемента, виглядає закриваючий тег: </p>.

В останній версії HTML5 до мови було додано багато нових елементів для підвищення інтерактивності, розширення мультимедійних можливостей і семантичної ефективності. Однією з найбільш очікуваних особливостей які було додано стала підтримка вбудовування аудіо та відео на веб-сторінки за допомогою тегів <audio> </audio> та <video> </video>. Також було додано кілька семантичних елементів, таких як: <article> , <header> та <footer> [27].

Плюси використання HTML:

- Широко використовується з великою кількістю доступних ресурсів та має величезну спільноту;

- Має відкритий вихідний код і абсолютно безкоштовна;
- Відносно легка у вивченні;
- Запускається в кожному веб-браузері;
- Має чистий і послідовний вихідний код;
- Офіційні веб-стандарти підтримуються W3C;
- Можна просто інтегрувати з серверними мовами та платформами, такими як PHP та Node.js.

Також існує кілька мінусів, які слід враховувати:

- Здебільшого використовується для статичних веб-сторінок і не має, як такої, динамічної функціональності;
- Окреме створення веб-сторінок, навіть за умови, що використовуються подібні елементи;
- Непередбачувана поведінка веб-браузера. Старі браузери можуть бути не сумісними з новішими функціями [28].

HTML використовується для створення веб-сторінок, але має обмеження, коли мова йде про взаємодію з користувачем. Тому HTML слід використовувати лише для додавання текстових елементів і структурування їх на сторінці. Для більш складних функцій HTML можна комбінувати з каскадними таблицями стилів (CSS) та мовою програмування JavaScript (JS). Разом вони здатні розширити функціонал. В той час як CSS відповідатиме за дизайн та візуальне оформлення сторінки, JavaScript відповідатиме за взаємодію з користувачем та динаміку.

2.2 CSS

CSS (Cascading Style Sheets) – це каскадні таблиці стилів, які використовуються для візуального оформлення веб-сторінок.



Рисунок 2.2 – Логотип CSS

Таблиці стилів можуть містити стільки інформації, стільки потрібно. Зазвичай вони містять правила, які керують макетом і форматуванням сторінки, а також кольори, шрифти та іншу інформацію про стилі для окремих елементів HTML [29].

Існує кілька методів застосування коду CSS для стилістичних змін у HTML.

Через зовнішні таблиці стилів, коли код CSS розміщується в повністю окремому файлі. Файл має розширення .css і посилається через елемент `<link>` в коді HTML. Більшість програмістів віддають перевагу саме цьому методу через його універсальність. Оскільки можна використовувати одну таблицю стилів для встановлення стилю кількох різних документів.

Застосування CSS до HTML через внутрішні таблиці стилів. У цьому випадку замість того, щоб розміщувати CSS у зовнішньому файлі, стилі розміщуються у документі HTML за допомогою тегу `<style>`. Даний метод використовується в конкретних ситуаціях і не є таким ефективним як використання зовнішніх таблиць. У даному випадку стилі впливають лише на одну сторінку, на якій знаходяться. Якщо потрібно використовувати ті самі стилі для декількох сторінок, доведеться повторювати CSS на кожній з них.

Також можна використання вбудовані стилі. Вбудовані стилі – це конкретні оголошення CSS, які впливають лише на один елемент HTML. На відміну від внутрішніх таблиць стилів, вбудовані стилі розміщуються в атрибуті `style` замість елемента `<style>`. Досвідчені розробники намагаються

уникати цього конкретного методу, коли це можливо, оскільки він вимагає від них оновлювати ідентичну інформацію кілька разів в одному документі і робить кодування CSS більш складним для розуміння та читання. Замість цього краще відокремити різні типи коду, які стануть у нагоді під час розробки і налагодження. Крім того, кожному хто буде читати чи використовувати такий код, буде набагато легше його зрозуміти [30].

Переваги використання CSS:

- Економія часу – один CSS файл може бути застосований для відображення багатьох HTML-сторінок. Це дозволяє описувати зовнішній вигляд елементів на сторінці лише один раз;
- Допомогає вносити спонтанні та послідовні зміни – коли виникає необхідність змінити стилістичне оформлення HTML-сторінок, можна всього лише відкоригувати CSS-файл замість того, щоб коригувати кожен сторінку окремо. За допомогою єдиної таблиці стилів можна бути впевненим, що зміни виглядатимуть узгоджено на всіх сторінках;
- Більш широкий вибір дизайну – CSS набагато потужніший ніж чистий HTML. Він надає додаткові можливості форматування, які покращують стилістичне оформлення веб-сторінок;
- Покращення швидкості завантаження веб-сторінки – реалізується за рахунок перенесення в окремий CSS-файл правил представлення даних. Сторінки швидше починають завантажуватись, коли веб-браузер під час першого відвідування сторінки кешує таблицю стилів, а далі, при наступних відвідуваннях завантажуються тільки структура документа і дані, які зберігаються на сторінці, що є вигідним для оптимізації її швидкості;
- Сумісність пристроїв – оскільки люди використовують широкий спектр пристроїв для доступу до веб-сайтів чи веб-додатків через мережу інтернет, існує потреба у адаптивному веб-дизайні. CSS

дозволяє зробити веб-сторінки більш адаптивними, для їх комфортного відображення на пристроях користувачів;

- Розміщення елементів – можна змінювати положення елементів HTML на сторінці;
- Покращення пошукової системи сканування веб-сторінки – завдяки тому, що атрибути дизайну знаходяться в окремому CSS-файлі, в HTML-документі міститься менше коду, що робить його зручнішим для SEO оптимізації [31].

CSS – це потужний інструмент і основа практично для будь-якого веб-сайту чи веб-додатку. Сьогодні це загальновизнаний стандарт розробки, який приймають усі без винятку компанії-розробники, що явно демонструє його важливість і необхідність.

Вносячи зміни в CSS-файл веб-сторінки, можна отримати майже повний контроль над її макетом і зовнішнім виглядом. Більше того, каскадні таблиці стилів дозволяють змінити зовнішній вигляд сторінки, не впливаючи на її вміст.

Дизайн і кодування можуть здатися двома абсолютно різними наборами навичок. Однак, коли справа доходить до веб-розробки, вони нерозривно пов'язані між собою.

2.3 Sass

Sass (Syntactically Awesome Style Sheets) – це препроцесор CSS, який надає більш широкий функціонал, що спрощує та пришвидшує процес написання стилів.



Рисунок 2.3 – Логотип Sass

CSS використовуються у всій мережі інтернет, але це не робить її найзручнішою для кодування стилів. Вона була створена для того, щоб допомогти веб-розробникам написати інструкції для представлення візуального вигляду веб-сторінок, а не для роботи зі змінними чи виконання складні завдання щодо прийняття рішень. Sass усуває деякі з цих серйозних недоліків CSS, заощаджуючи час і зусилля розробників на проектах.

Хоча Sass і CSS теоретично мають однакові можливості, Sass може виконувати ту ж роботу, використовуючи менше коду. Це робить код Sass набагато легшим для читання та розуміння, особливо в великих веб-проектах із залученням декількох розробників.

Синтаксис Sass багато в чому схожий на синтаксис CSS. Хоча між ними є кілька ключових відмінностей, більшість розробників які вже знають CSS, можуть з легкістю опанувати Sass, почитавши трохи теорії.

Sass – це мова препроцесора, яка інтерпретується в CSS. Вона приймає вхідні дані та перетворює їх у вихідні, які використовуються в якості вхідних даних іншою програмою. Іншими словами, Sass скомпілює свій код і згенерує його в CSS, зрозумілий браузеру.

Процес інтерпретації потрібен, оскільки браузер не розуміє Sass і може читати тільки код CSS-код.

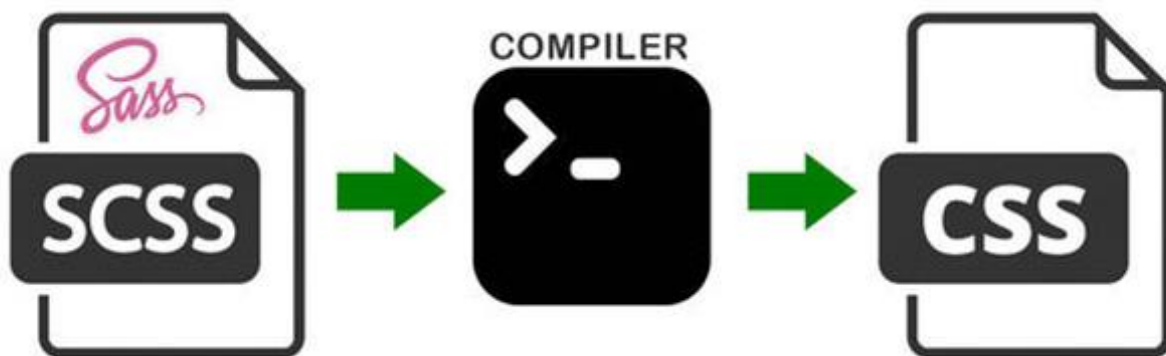


Рисунок 2.4 – Процес інтерпретації Sass у CSS [32]

Sass дозволяє використовувати змінні, які можуть зберігати інформацію, що буде використовуватись в таблиці стилів. Наприклад, можна зберегти значення кольору у змінній на початку файлу, а потім використовувати цю змінну під час встановлення кольору елементів на сторінці.

У чистому CSS, якщо для веб-сторінки потрібна інша колірна схема, то доведеться в ручну редагувати її в кожному місці де вона застосовується. Це забирає багато зусиль та часу. Але завдяки змінним у Sass, можна швидко змінювати кольори (чи інші стилі), не змінюючи кожен рядок окремо.

За допомогою Sass можна не тільки визначати змінні, але й пакувати групи змінних, відомі як Mixins. Наприклад, якщо більшість сторінок веб-сайту використовують однакові шрифти, кольори і схеми меж, то можна визначити Mixin, який містить в собі всі ці значення. І на майбутніх веб-сторінках все, що потрібно буде зробити, це викликати Mixin, а не повторювати код [33].

Sass надає можливість виконувати обчислення, оскільки він дозволяє використовувати математичні вирази. Він надає доступ до стандартних математичних функцій, таких як додавання, віднімання, множення та ділення. Їх можна використовувати для складних обчислень.

У Sass існують вбудовані функції, які легко викликаються, щоб оптимізувати процес написання коду.

Також в препроцесорі Sass є можливість використовувати вкладений синтаксис, який являє собою код, що міститься в середині іншого коду, який виконує більш широкий функціонал. Іншими словами, можна вкладати елементи HTML за допомогою селекторів CSS. Це дозволяє отримати більш природній і зручний для читання синтаксис та запобігає необхідності багаторазового переписування селекторів [34].

Отже, переваги використання Sass:

- Скорочує час на створення і підтримку CSS;
- Sass дозволяє мати модульну організацію стилів, що є життєво важливим для великих проєктів;
- Він забезпечує розширені структури, типові для мов програмування, такі як змінні, списки, функції та структури курування;
- Сумісний з усіма версіями CSS. Таким чином можна використовувати всі доступні бібліотеки CSS
- Дозволяє слідкувати за файлами таким чином, що, якщо відбулися зміни в таблиці стилів, вона автоматично регенерується (режим перегляду).
- Існує велика спільнота користувачів.

2.4 JavaScript

JavaScript – це мова програмування, яку використовують переважно веб-браузери для створення динамічного та інтерактивного досвіду користувачів.

Коли JavaScript було вперше розроблено в 1995 році, вона задумувалась як легка мова, яка використовується для підвищення динамічності веб-сторінок. Вона додавала прості ефекти, такі як показ і приховування елементів

одним натисканням кнопки, але незабаром розробники зрозуміли, що JavaScript може набагато більше.



Рисунок 2.5 – Логотип JavaScript

З часом JavaScript стала однією з небагатьох мов, які використовувалися практично в кожному веб-браузері. Сьогодні більшість інших мов відійшли на другий план, залишивши JavaScript першою мовою програмування, яку можна використовувати в кожному популярному браузері.

Наступний великий прорив для JavaScript настав, коли розробники почали експериментувати з нею за межами веб-браузерів. Вони почали застосовувати її до бекенду веб-серверів, де вона обробляла дані, які використовує фронтенд. Після появи смартфонів вона знайшла застосування в багатьох інструментах розробки мобільних пристроїв, що дозволяють створювати програми, які працюють як на платформах Android, так і на iOS

JavaScript є мовою сценаріїв, яку можна використовувати як для роботи на клієнтській так і на серверній стороні для розробки динамічних веб-сторінок. Тим часом як HTML і CSS створюють структуру і стилістику веб-сторінки, JavaScript робить її функціональною, для забезпечення взаємодії з кінцевим користувачем [35].

Завдяки великій колекції фреймворків JavaScript став ще більш універсальним. Фреймворки – це бібліотеки попередньо написаного коду JavaScript, який розробники використовують для стандартних функцій.

Найбільш популярними фреймворками на JavaScript є:

- React: бібліотека JavaScript, створена для побудови користувацьких інтерфейсів веб-додатків;
- React Native: фреймворк, який дозволяє розробникам створювати мобільні додатки на JavaScript;
- Node.js: платформа, яка забезпечує двосторонній зв'язок із серверами для обміну даними [36].

Front-end розробники використовують JavaScript, разом із HTML і CSS, для створення частин веб-сторінки, які користувачі бачать та взаємодіють із ними в своїх браузерах.

До появи JavaScript на веб-сторінках відображалось лише статичний вміст. JavaScript дав можливість розробникам надавати своїм користувачам динамічний досвід роботи з сторінкою, додаючи анімації та інші інтерактивні елементи.

JavaScript також популярний серед розробників back-end. Внутрішня розробка (також відома як розробка на стороні сервера) передбачає створення коду, який виконується на веб-сервері. Коли браузер завантажує веб-сторінку, він звертається до віддаленого серверу. Після цього код на стороні серверу аналізує URL сторінки, щоб визначити, що запитує користувач, а після витягує і перетворює необхідні дані для повернення назад браузеру. Для розробки back-end зазвичай використовується фреймворк Node.js.

Протягом багатьох років розробникам мобільних додатків доводилося обирати між Android та iOS, оскільки для кожної платформи використовувалися різні мови програмування. Розробники додатків на Android вивчали Java або Kotlin, а розробники на iOS віддавали перевагу Swift або Objective-C.

Тепер, з появою таких фреймворків для мобільної розробки, як React Native, можна використовувати JavaScript для створення додатків, працюючих на обох платформах.

Також JavaScript можна використовувати для створення 2D і 3D відеоігор, які запускаються у веб-браузерах. Оскільки браузери стають потужнішими, веб-відеоігри виходять за рамки простих платформерів. Тепер такі фреймворки JavaScript, як Phaser.js, дозволяють швидко і легко створювати більш складні ігри прямо з браузера.

В JavaScript є фреймворки для підтримки розробки проектів, пов'язаних з штучним інтелектом. Наприклад, Tensorflow.js – це бібліотека JavaScript, за допомогою якої можна створювати та навчати моделі машинного навчання [37].

JavaScript – одна із найпопулярніших мов програмування на сьогоднішній день. Її універсальність і широкий спектр застосування роблять її цінним доповненням до технологічного стеку будь-якого розробника.

2.5 React

React – це бібліотека на розробки користувацького інтерфейсу на основі JavaScript. Вона розроблена компанією Facebook і спільнотою розробників з відкритим кодом. Бібліотека вперше з'явилась в травні 2013 року і зараз є однією з найбільш використовуваних інтерфейсних бібліотек для веб-розробки [38].



Рисунок 2.6 – React

Сьогодні популярність React перевершила популярність усіх інших фреймворків для розробки front-end. Основні причини її популярності:

- Легке створення динамічних додатків: React вимагає менше кодування та пропонує більше функціональних можливостей, тим самим полегшуючи створення динамічних веб-додатків, на відміну від JS, де кодування часто стає складним;
- Покращена продуктивність: React використовує Virtual DOM, тим самим забезпечуючи швидкість створення веб-додатків. На відмінну від звичайних веб-додатків, яку оновлюють всі компоненти на сторінці не залежно від того чи змінювалися вони, Virtual DOM порівнює попередні стани компонентів і оновлює лише ті елементи в Real DOM, які зазнали змін;
- Багаторазово використовуванні компоненти: компоненти є будівельними блоками будь-якого додатку на React, і один додаток зазвичай складається з декількох компонентів. Вони можуть повторно використовуватися, мають свою логіку та елементи керування, що, у свою чергу, значно скорочує час розробки додатку;
- Односпрямований потік даних: це означає, що при проектуванні додатку на React розробники часто вкладають дочірні компоненти в батьківські. Набагато легше налагоджувати помилки та знати про проблеми, виникаючі в додатку, коли дані надходять в одному напрямку;
- Не складна у вивченні: React проста у освоєнні, оскільки вона переважно поєднує в собі базові концепції HTML і JavaScript з деякими корисними доповненнями. Проте, як це часто буває у випадку з іншими бібліотеками та фреймворками, потрібен час для її більшого розуміння;

- Використовується для розробки не лише для розробки веб-додатків, а й для розробки мобільних: хоча React зазвичай використовується для розробки веб-додатків, існує фреймворк React Native, що має походження від самої React. Він є дуже популярним і з його допомогою можна створювати мобільні додатки;
- Спеціальні інструменти для легкого налагодження: Facebook випустив розширення для Chrome, яке можна використовувати для налагодження програм React. Це робить процес налагодження веб-додатків на React швидшим і простішим.

Компоненти React пишуться на JSX – синтаксичне розширення JavaScript. Це термін, який використовується в React для опису того, як має виглядати інтерфейс користувача. Можна описувати структуру HTML у той самий файл, що й код JavaScript, використовуючи JSX.

2.6 Visual Studio Code

Visual Studio Code (VS Code) – це текстовий редактор з відкритим вихідним кодом, що є повністю безкоштовний, від компанії Microsoft. Він доступний для macOS, Linux і Windows, тому з ним можна працювати незалежно від платформи. Хоча редактор відносно легкий, він містить в собі деякі потужні функції, за допомогою яких VS Code займає ключові позиції серед конкурентів та є одним з найбільш використовуваних інструментів середовища розробки [39].



Рисунок 2.7 – Visual Studio Code

В основі VS Code лежить блискавичний редактор вихідного коду, ідеальний для повсякденного використання. Підтримуючи сотні мов VS Code дає змогу миттєво підвищити продуктивність завдяки підсвічуванням синтаксису, підбору дужок, автоматичним відступам, виділенням поля тощо. Інтуїтивно зрозумілі комбінації клавіш і просте налаштування дозволяють легко орієнтуватися в коді.

VS Code включає в собі вбудовану підтримку завершення коду IntelliSense, розширене розуміння семантичного коду, навігацію та рефакторинг коду. Також він містить в собі інтерактивний налагоджувач, що дозволяє продивлятися вихідний код, перевіряти змінні, виконувати команди в консолі і переглядати стеки викликів.

Інтерфейс VS Code дуже зручний і забезпечує велику кількість користувацької взаємодії порівняно з іншими текстовими редакторами.

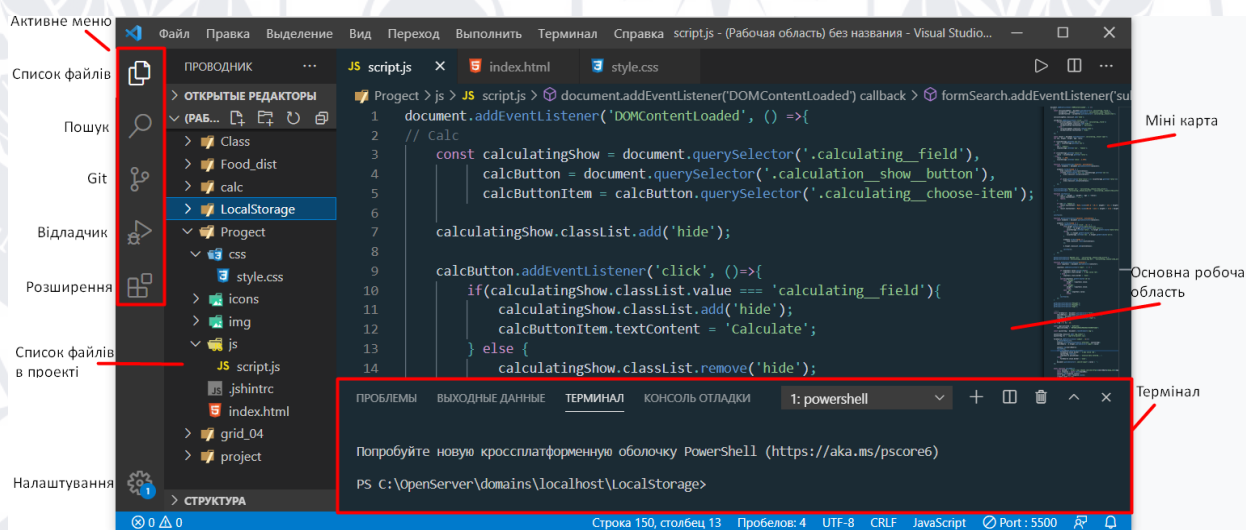


Рисунок 2.8 – Інтерфейс VS Code на прикладі одного з проєктів

Основну частину екрану займає вікно з текстовим вмістом файлу. У лівій частині екрана містяться вкладки активного меню, в якому знаходяться головні функції редактора, та вкладка провідника, що відкривається за замовчуванням при запуску програми. У неї виводяться список відкритих файлів і каталог відкритої папки. В правій частині знаходиться міні карта, яка допомагає орієнтуватися в коді. В низу знаходиться панель, яка містить в собі

вкладка з інтегрованим терміналом, консоллю налагодження, вихідними даними та вкладка з проблемами, про які редактор сповіщатиме під час компіляції коду.

VS Code підтримує роботу з системою контролю версій Git. Можна працювати з контролем вихідного коду, не виходячи з редактору, включно з переглядом змін, що відбулися під час останнього коміту.

Редактор включає розширену вбудовану підтримку розробки на Node.js з використанням JavaScript і TypeScript на основі тих самих базових технологій, які лежать в основі Visual Studio. Він також містить відмінний інструментарій для веб-технологій, таких як JSX/React, HTML, CSS, SCSS, Less і JSON [40].

VS Code дозволяє налаштувати кожен функцію так, як потрібно користувачеві, і встановити велику кількість сторонніх розширень, що забезпечують функціональність і комфортність роботи.

2.7 GitHub Pages

Після створення веб-додатку його необхідно дець опублікувати, щоб користувачі могли ним скористатися. Існує велика кількість хостингів для розміщення веб-додатків. Одним з таких є GitHub Pages.

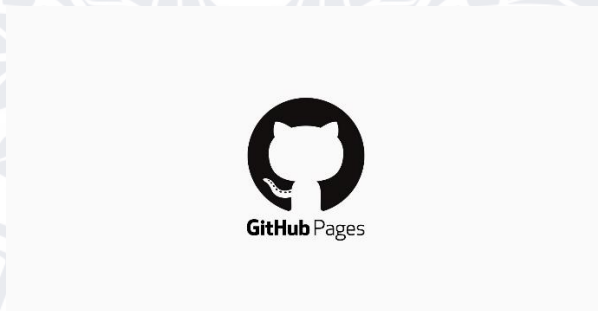


Рисунок 2.9 – Логотип GitHub Pages.

GitHub Pages – це безкоштовний веб-хостинг який пропонує GitHub для розміщення веб-сторінок для своїх користувачів. Він дуже зручний у використанні, все що потрібно це завантажити свій проект на GitHub і після

короткої інструкції з опублікування, яка міститься на сайті, можна користуватись додатком[41].

Висновок до другого розділу

Цей розділ демонструє технології, які будуть використовуватись для реалізації веб-додатку. Розглянуто такі технології як HTML, CSS, препроцесор SASS, мову програмування JavaScript, бібліотека React та середовище розробки Visual Studio Code. Обрано засіб для хостингу веб-додатку GitHub Pages. Наступний розділ буде присвячено безпосередньо реалізації веб-додатку для вивчення мови програмування C#.

РОЗДІЛ 3

РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКУ ДЛЯ ВИВЧЕННЯ C#

3.1 Постановка задачі

Згідно поставленої задачі теми бакалаврської роботи, необхідно створити веб-додаток для вивчення мови програмування C#. Потрібно реалізувати зручний та зрозумілий інтерфейс для користувачів. Можливість обирати тему необхідну для вивчення та закріпити її шляхом проходження гри у вигляді вікторини.

3.2 Реалізація веб-додатку

Головний екран веб-додатку містить шапку, основний контент підвал сторінки.

В шапці розміщується логотип додатку, який в свою чергу є посиланням на його стартову сторінку. Логотип максимально простий і зрозумілий, його легко запам'ятати, що неодмінно є плюсом.

В підвалі додатку знаходиться інформація про його автора, дату створення і призначення.

Основний контент головної сторінки містить в собі короткий опис актуальності вивчення мови програмування C#. Це допомагає зацікавити користувачів використовувати даний додаток.

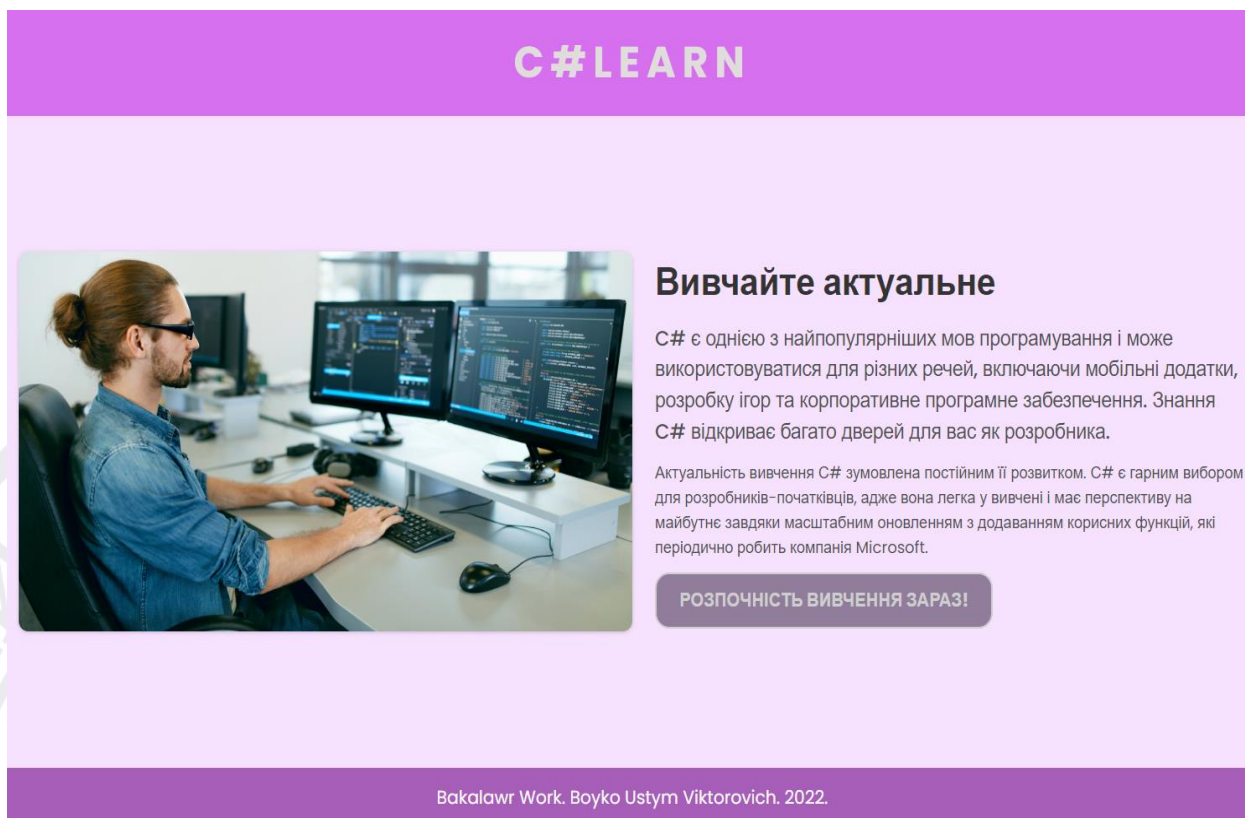


Рисунок 3.1 – Головна сторінка веб-додатку

Після короткого опису актуальності вивчення C#, в користувачів є можливість розпочати вивчення мови програмування клацнувши по відповідній кнопці.

Після перед користувачами з'явиться каталог з темами для вивчення мови програмування C#.

Після обрання однієї з доступних в додатку тем, користувач може ознайомитись з теоретичним матеріалом та вивчити теоретичні положення, безпосередньо по обраній ним темі.

Оскільки, додаток написаний за допомогою бібліотеки React, то перезавантаження сторінки не відбувається. Оновлюється лише основний контент, що значно пришвидшує роботу в ньому. Це дуже зручно, оскільки при його створенні не потрібно повторювати вміст, який зустрічається на декількох сторінках веб-додатку.

C# LEARN

Мова C# та платформа .NET

Змінні та константи

Типи даних

Умовні вирази

Цикли

Масиви

Методи

На сьогоднішній момент мова програмування C# одна з найпотужніших мов, що швидко розвиваються і затребуваних в IT-галузі. Зараз на ній пишуть різні програми: від невеликих десктопних програм до великих веб-порталів і веб-сервісів, що обслуговують щодня мільйони користувачів.

C# вже не молода мова і, як і вся платформа .NET, вже пройшов великий шлях. Перша версія мови вийшла разом із релізом Microsoft Visual Studio .NET у лютому 2002 року. Поточною версією мови є версія C# 10.0, яка вийшла 8 листопада 2021 разом із релізом .NET 6. C# є мовою із Сі-подібним синтаксисом і близький у цьому відношенні до C++ та Java. Тому якщо ви знайомі з однією з цих мов, то оволодіти C# буде легше.

C# є об'єктно-орієнтованим і в цьому плані багато перейняв у Java та C++. Наприклад, C# підтримує поліморфізм, успадкування, навантаження операторів, статичну типізацію. Об'єктно-орієнтований підхід дозволяє вирішити завдання з побудови великих, але в той же час гнучких, масштабованих і додатків, що розширюються. І C# продовжує активно розвиватися і з кожною новою версією з'являється все більше цікавих функціональностей.

Роль платформи

Коли говорять C#, часто мають на увазі технології платформи .NET (Windows Forms, WPF, ASP.NET, Xamarin). І навпаки, коли говорять .NET, нерідко мають на увазі C#. Проте, хоча ці поняття пов'язані, ототожнювати їх не так. Мова C# була створена спеціально для роботи з фреймворком .NET, проте саме поняття .NET дещо ширше.

Якось Білл Гейтс сказав, що платформа .NET – це найкраще, що створила компанія Microsoft. Можливо, він мав рацію. Фреймворк .NET є потужною платформою для створення додатків. Можна виділити такі основні риси:

- Підтримка кількох мов. Основою платформи є загальномовне середовище виконання Common Language Runtime (CLR), завдяки чому .NET підтримує кілька мов: поряд з C# це також VB.NET, C++, F#, а також різні діалекти інших мов, прив'язані до .NET, наприклад, Delphi. NET. При компіляції код будь-якою з цих мов компілюється у складання загальною мовою CIL (Common Intermediate Language) – свого роду асемблер платформи .NET. Тому за певних умов ми можемо зробити окремі модулі однієї програми окремими мовами.
- Кросплатформність .NET є платформою, що переноситься (з деякими обмеженнями). Наприклад, остання версія платформи на даний момент – .NET 6 підтримується на більшості сучасних ОС Windows, MacOS, Linux. Використовуючи різні технології на платформі .NET, можна розробляти програми на мові C# для різних платформ – Windows, MacOS, Linux, Android, iOS, Tizen.
- Потужна бібліотека класів .NET представляє єдину всім підтримуваних мов бібліотеку класів. І яку б програму ми не збиралися писати на C# – текстовий редактор, чат-бот, онлайн веб-сайт – так чи інакше ми використаємо бібліотеку класів .NET

Рисунок 3.2 – Представлення теоретичного матеріалу

Після проходження теоретичного матеріалу, в користувачів є змога перевірити засвоєну інформацію шляхом проходження вікторини. Достатньо лише натиснути на відповідну кнопку в кінці теорії.

Керований та некерований код

Нерідко програму, створену на C#, називають керованим кодом (managed code). Що це означає? А це означає, що ця програма створена на основі платформи .NET і тому управляється загальномовним середовищем CLR, яке завантажує програму і при необхідності очищає пам'ять. Але є також програми, наприклад, створені мовою C++, які компілюються над загальну мову CIL, як C#, VB.NET чи F#, а звичайний машинний код. У цьому випадку .NET не керує програмою. У той же час платформа .NET надає можливості для взаємодії з некерованим кодом.

JIT-компіляція

Як вище писалося, код C# компілюється в додатки або збірки з розширеннями exe або dll мовою CIL. Далі при запуску на виконання подібного додатка відбувається JIT-компіляція (Just-In-Time) в машинний код, який потім виконується. При цьому, оскільки наша програма може бути великою і містити купу інструкцій, в даний момент компілюватиметься лише та частина програми, до якої безпосередньо йде звернення. Якщо ми звернемося до іншої частини коду, вона буде скомпільована з CIL в машинний код. При цьому вже скомпільована частина програми зберігається до завершення роботи програми. У результаті це підвищує продуктивність.

[Перейти до уроку!](#)

Bakalawr Work. Boyko Ustym Viktorovich. 2022.

Рисунок 3.3 – Кнопка переходу до проходження вікторини

Вікторина складається з певної кількості питань. Їхня кількість залежить від обраної теми. Під час проходження вікторини є змога повернутися до теорії натиснувши на відповідну кнопку.

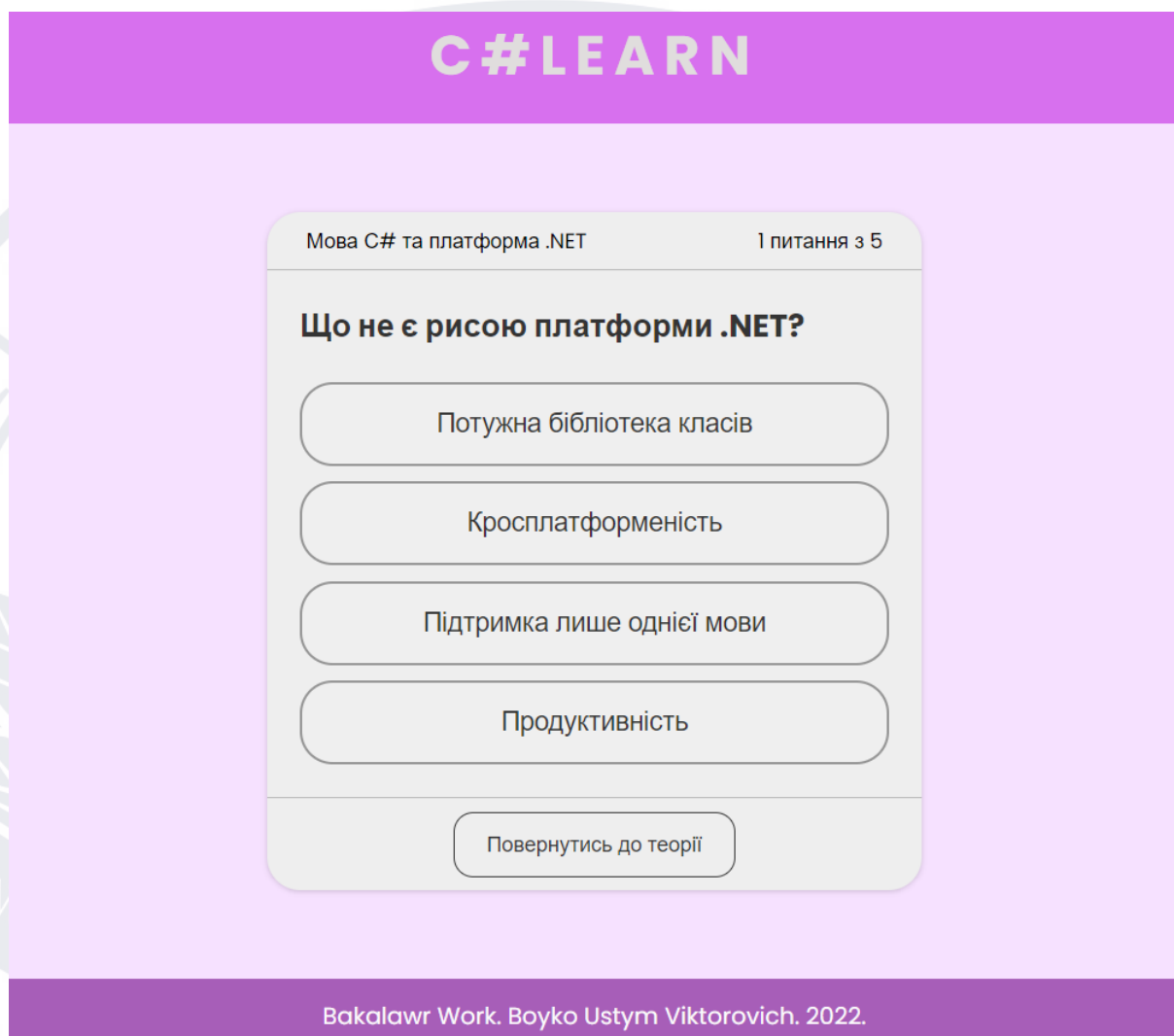


Рисунок 3.4 – Вікторина для засвоєння матеріалу

Вікторина – це вид гри, під час якого потрібно відповідати на запитання з варіантами відповідей. Вона чудово підходить для подібних додатків, щоб засвоїти теоретичний матеріал і додати інтерактивності та взаємодії з користувачами.

Правила вікторини прості, відповісти вірно на якомога більшу кількість питань. Всі питання наведені в вікторині, в даному веб-додатку, суворо відповідають теоретичному матеріалу.

Після її проходження користувачам показується, на сторінці додатку, кількість правильних відповідей на запитання.



Рисунок 3.5 – Результат проходження вікторини

В додатку використовується мінімалістичний дизайн, оскільки він має низку переваг:

- Важливе видно одразу. Мінімальна кількість елементів на сторінці дозволяє не плутатись в контенті та максимально просто і зручно користуватися додатком;
- Проста і зрозуміла навігація. Користувачі веб-додатку комфортно почувають себе, коли знають, що і де знаходиться. Процес вивчення інтерфейсу в таких додатках займає мінімум часу;

- Швидкість завантаження. Оскільки додаток містить мінімум ресурсоємних елементів, таких як різні фотографії, що особливо не впливають на зміст, чи недоречні анімації, він не перенавантажується та його завантаження займає малу кількість часу.

Важливо розуміти, що мінімалістичний дизайн актуальний до поки в змозі підтримувати функціональні та естетичні потреби веб-додатку.

Під час реалізації веб-додатку було створено багато файлів в редакторі Visual Studio Code, за допомогою яких він функціонує.

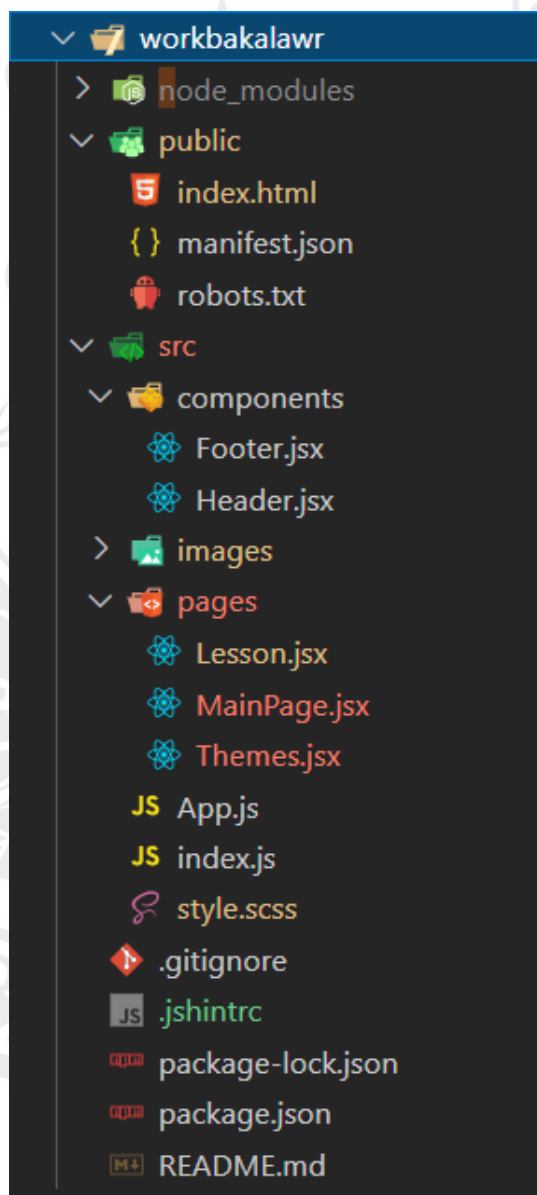


Рисунок 3.6 – Створені файли для веб-додатку у VS Code

Скористатись додатком і поглянути на його функціональність можна за посиланням [42], на хостингу де він розміщений.

Висновок до третього розділу

В даному розділі було описано постановку задачі. Розглянуто реалізацію та функціональність веб-додатку. Наведено переваги використання мінімалістичного дизайну.



ВИСНОВКИ

В результаті виконання бакалаврської кваліфікаційної роботи було створено веб-додаток для вивчення мови програмування C#.

Розкрито поняття веб-додатку та його основні відмінності від веб-сайту. Розглянуто особливості створення веб-додатків, їхні переваги та недоліки. Досліджено принцип їхньої роботи, їхні види та основні підходи при їх створенні. Приведено огляд існуючих аналогів.

Також було розглянуто технології, що використовувались під час створення веб-додатку. Обрано середовище розробки Visual Studio Code та засіб для хостингу GitHub Pages. Показано реалізацію веб-додатку, його функціонал та дизайн.

Таким чином, можна зробити висновок, що даний веб-додаток повністю реалізовано з усім необхідним функціоналом, згідно з постановкою задачі

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Програмування – основа сучасного світу [Електронний ресурс] / replace.org.ua – Режим доступу до ресурсу: <https://replace.org.ua/topic/194/>
2. Мови програмування і сфери їх застосування [Електронний ресурс] / blog.jungo.dev – Режим доступу до ресурсу: <https://blog.jungo.dev/uk/2020/12/yazyki-programmirovaniya-i-sfery-ih-primeneniya-2/>
3. C Sharp [Електронний ресурс] / uk.wikibooks.org – Режим доступу до ресурсу: https://uk.wikibooks.org/wiki/C_Sharp
4. Рейтинг мов програмування 2022 [Електронний ресурс] / dou.ua – Режим доступу до ресурсу: <https://dou.ua/lenta/articles/language-rating-2022/>
5. Чому варто вивчати мову програмування C#? [Електронний ресурс] / www.quality-assurance-group.com – Режим доступу до ресурсу: <https://www.quality-assurance-group.com/chomu-varto-vyvchaty-movu-programuvannya-c-c-sharp/>
6. Веб-сайт – Вікіпедія [Електронний ресурс] / uk.wikipedia.org – Режим доступу до ресурсу: <https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1%D1%81%D0%B0%D0%B9%D1%82>
7. Різниця між веб-додатком та веб-сайтом [Електронний ресурс] / ua.sawakinome.com – Режим доступу до ресурсу: <https://ua.sawakinome.com/articles/technology/difference-between-web-application-and-website-2.html>
8. Що таке веб додаток? [Електронний ресурс] / tebapit.com – Режим доступу до ресурсу: <https://tebapit.com/%D1%89%D0%BE-%D1%82%D0%B0%D0%BA%D0%B5-%D0%B2%D0%B5%D0%B1-%D0%B4%D0%BE%D0%B4%D0%B0%D1%82%D0%BE%D0%BA/>

9. Website и Web Application: в чем разница [Электронный ресурс] / dinarys.com – Режим доступа до ресурсу: <https://dinarys.com/ru/blog/website-vs-webapplication>
10. Різниця між веб-додатком та веб-сайтом [Электронный ресурс] / uk.strephonsays.com – Режим доступа до ресурсу: <https://uk.strephonsays.com/web-application-and-vs-website-1009>
11. Чем веб-приложение отличается от веб-сайта [Электронный ресурс] / itproger.com – Режим доступа до ресурсу: <https://itproger.com/news/chem-veb-prilozhenie-otlichaetsya-ot-sayta>
12. Основные различия между веб-сайтом и веб-приложением [Электронный ресурс] / artismedia.by – Режим доступа до ресурсу: <http://artismedia.by/blog/osnovnye-razlichiya-mezhdu-veb-sajtom-i-veb-prilozheniem/>
13. Web-приложения – преимущества и недостатки [Электронный ресурс] / mydiv.net – Режим доступа до ресурсу: https://mydiv.net/arts/view-web-prilozhenija_preimuxhestva_i_nedostatki.html
14. Веб-додаток і його характеристики | Centum-D [Электронный ресурс] / www.centum-d.com – Режим доступа до ресурсу: <https://www.centum-d.com/veb-dodatok-yogo-harakteristiki/>
15. Переваги та недоліки Web-додатків [Электронный ресурс] / stud.com.ua – Режим доступа до ресурсу: https://stud.com.ua/97611/informatika/perevagi_nedoliki_dodatktiv
16. Web-приложение: понятие, компоненты и принципы работы [Электронный ресурс] / smartiqa.ru – Режим доступа до ресурсу: <https://smartiqa.ru/courses/web/lesson-1>
17. Обзор протокола HTTP [Электронный ресурс] / developer.mozilla.org – Режим доступа до ресурсу: <https://developer.mozilla.org/ru/docs/Web/HTTP/Overview>

18. Что такое веб-приложения, виды и их преимущества [Электронный ресурс] / azoft.ru – Режим доступа до ресурсу: <https://www.azoft.ru/blog/web-apps/>
19. Одностраничные (spa) и многостраничные (pwa) веб-приложения [Электронный ресурс] / vc.ru – Режим доступа до ресурсу: <https://vc.ru/seo/108149-odnostranichnye-spa-i-mnogostranichnye-pwa-veb-prilozheniya?comments>
20. SPA, MPA и PWA. Плюсы и минусы популярных подходов к построению сайтов [Электронный ресурс] / azoft.ru – Режим доступа до ресурсу: <https://www.azoft.ru/blog/spa-mpa-pwa/>
21. Виды веб-приложений [Электронный ресурс] / doka.guide – Режим доступа до ресурсу: <https://doka.guide/js/web-app-types/>
22. Що таке веб-додаток? Види веб-додатків [Электронный ресурс] / webcase.com.ua – Режим доступа до ресурсу: <https://webcase.com.ua/uk/blog/cho-takoe-web-prilozhenie-vse-vidy/>
23. Sololearn: Learn to code [Электронный ресурс] / sololearn.com – Режим доступа до ресурсу: <https://www.sololearn.com>
24. My Home | Codecademi [Электронный ресурс] / codecademy.com – Режим доступа до ресурсу: <https://www.codecademy.com/learn>
25. SoloLearn проти Codecademy 2022: остаточне порівняння [Электронный ресурс] / bloggersideas.com – Режим доступа до ресурсу: <https://www.bloggersideas.com/uk/sololearn-vs-codecademy/>
26. Основы HTML – Изучение веб-разработки [Электронный ресурс] / developer.mozilla.org – Режим доступа до ресурсу: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics
27. Що таке HTML? Пояснюємо основи [Электронный ресурс] / futurenow.com.ua – Режим доступа до ресурсу: <https://futurenow.com.ua/shho-take-html/>

28. What is HTML and How Does Hypertext Markup Language Work? [Електронний ресурс] / theserverside.com – Режим доступу до ресурсу: <https://www.theserverside.com/definition/HTML-Hypertext-Markup-Language>
29. What is CSS (And How Does It Work)? [Електронний ресурс] / designbombs.com – Режим доступу до ресурсу: <https://www.designbombs.com/glossary/what-is-css/>
30. How Does CSS Work? [Електронний ресурс] / bootcamp.berkeley.edu – Режим доступу до ресурсу: <https://bootcamp.berkeley.edu/resources/coding/learn-css/how-does-css-work/>
31. Advantages and Disadvantages of CSS Everyone Should Know [Електронний ресурс] / motocms.com – Режим доступу до ресурсу: <https://www.motocms.com/blog/en/advantages-and-disadvantages-of-css/>
32. What is Sass CSS? Advantages & disadvantages of using Sass CSS [Електронний ресурс] / mobiosolutions.medium.com – Режим доступу до ресурсу: <https://mobiosolutions.medium.com/what-is-sass-css-advantages-disadvantages-of-using-sass-css-61a753d63a01>
33. 7 benefits of using SASS over conventional CSS [Електронний ресурс] / mugo.ca – Режим доступу до ресурсу: <https://www.mugo.ca/Blog/7-benefits-of-using-SASS-over-conventional-CSS>
34. A Beginner's Guide to Sass [Електронний ресурс] / codecademy.com – Режим доступу до ресурсу: <https://www.codecademy.com/resources/blog/what-is-sass/>
35. Importance of JavaScript: What is JavaScript and What are the Advantages of JavaScript? [Електронний ресурс] / status200.net – Режим доступу до ресурсу: <https://status200.net/importance-of-javascript/>
36. What is JavaScript used for? [Електронний ресурс] / lighthouselabs.ca – Режим доступу до ресурсу: <https://www.lighthouselabs.ca/en/blog/what-is-javascript-used->

<for#:~:text=Javascript%20is%20used%20by%20programmers,by%2097.0%25%20of%20all%20websites.>

37. What is JavaScript used for? [Електронний ресурс] / codecademy.com – Режим доступу до ресурсу: <https://www.codecademy.com/resources/blog/what-is-javascript-used-for/>
38. The Best Guide to Know What Is React [Електронний ресурс] / simplilearn.com – Режим доступу до ресурсу: <https://www.simplilearn.com/tutorials/reactjs-tutorial/what-is-reactjs>
39. What is Visual Studio Code? [Електронний ресурс] / educative.io – Режим доступу до ресурсу: <https://www.educative.io/edpresso/what-is-visual-studio-code>
40. Why Visual Studio Code? [Електронний ресурс] / code.visualstudio.com – Режим доступу до ресурсу: <https://code.visualstudio.com/docs/editor/whyvscode>
41. GitHub Pages [Електронний ресурс] / pages.github.com – Режим доступу до ресурсу: <https://pages.github.com/>
42. Відкритий проект на сервері GitHub [Електронний ресурс] / github.com – Режим доступу до ресурсу: <https://github.com/Ustym-b/Bakalavr-work-web-app>

Декларація щодо унікальності текстів роботи
та невикористання матеріалів інших авторів без посилань

Бойко Устим Вікторович

Прізвище, ім'я, по батькові

Факультет інформаційних і прикладних технологій

Факультет

122 Комп'ютерні науки

Шифр і назва спеціальності

Сучасні інформаційні технології та програмування

Освітня програма

ДЕКЛАРАЦІЯ

Усвідомлюючи свою відповідальність за надання неправдивої інформації, стверджую, що подана кваліфікаційна (бакалаврська) робота на тему: «РОЗРОБКА ГРИ ДЛЯ ВИВЧЕННЯ МОВИ C# У ВИГЛЯДІ ВЕБ-ДОДАТКУ» є написаною мною особисто.

Одночасно заявляю, що ця робота:

- не передавалась іншим особам і подається до захисту вперше;
- не порушує авторських та суміжних прав, закріплених статтями 21-25 Закону України «Про авторське право та суміжні права»;
- не отримувались іншими особами, а також дані та інформація не отримувались у недозволений спосіб.

Я усвідомлюю, що у разі порушення цього порядку моя кваліфікаційна (бакалаврська) робота буде відхилена без права її захисту, або під час захисту за неї буде поставлена оцінка «незадовільно».

дата

підпис