

ПВНЗ «ЄВРОПЕЙСЬКИЙ УНІВЕРСИТЕТ»

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА МАТЕМАТИЧНИХ НАУК
СПЕЦІАЛЬНІСТЬ 121 ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА 121 ІНЖЕНЕРІЯ ПРОГРАМНОГО
ЗАБЕЗПЕЧЕННЯ

**ВИПУСКНА КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА
РОБОТА**

на тему:

Розробка навчальної гри для вивчення концепцій програмування

Студентка _____ Шеремет Анна Романівна

Науковий

керівник _____ Яровий Р. О.

Допущено до захисту “ ” 2025 р.

Завідувач кафедри _____ Сушинський О. С.

Декан факультету _____ Яровий Р. О.

Завідувач

навчальної частини _____ Козинець А. О.

Київ – 2025

АНОТАЦІЯ

Дипломна робота присвячена розробці інтерактивної навчальної гри для вивчення основ програмування, що поєднує гейміфіковані підходи з освітнім контентом. Актуальність теми зумовлена зростаючим попитом на фахівців у сфері інформаційних технологій та необхідністю удосконалення методів навчання програмуванню, особливо для початківців. Традиційні форми подачі матеріалу часто не враховують потреби візуального, практичного та інтерактивного навчання, що ускладнює засвоєння абстрактних понять, таких як алгоритми, змінні, умовні оператори чи об'єктно-орієнтоване програмування.

Метою дослідження є створення програмного продукту — навчальної гри, яка дозволяє користувачам різного рівня підготовки вивчати базові концепції програмування у доступній, захопливій та адаптивній формі. У межах роботи проаналізовано сучасні освітні ігрові платформи (зокрема CodeCombat, Tynker, Scratch, CodinGame), вивчено досвід їх застосування та виділено основні вимоги до ефективної системи інтерактивного навчання. На основі цього аналізу сформульовано вимоги до власної гри: зручність інтерфейсу, підтримка кількох мов програмування, наявність системи прогресу, миттєвий зворотний зв'язок, гнучка архітектура.

У роботі описано процес проектування архітектури платформи, реалізовано основні функціональні модулі (реєстрація, вибір мови, система рівнів, перевірка завдань), спроектовано інтерфейс користувача з урахуванням принципів UX/UI-дизайну. Для розробки використано сучасний стек технологій: React і Tailwind CSS — для фронтенду, Node.js з Express — для серверної частини, PostgreSQL — для зберігання даних, а також Vite як інструмент збірки.

Особливу увагу приділено гейміфікації: користувачі отримують нагороди, досягають нових рівнів і мають змогу відстежувати власний прогрес, що сприяє збереженню мотивації. Проведено функціональне тестування

системи, зокрема юзабіліті-тестування, яке підтвердило ефективність та зручність розробленого продукту.

ABSTRACT

This thesis is devoted to the development of an interactive educational game for learning the fundamental concepts of programming by combining gamified approaches with educational content. The relevance of the topic is driven by the growing demand for IT specialists and the need to improve methods of teaching programming, especially for beginners. Traditional teaching formats often fail to meet the needs of visual, practical, and interactive learners, making it difficult to grasp abstract concepts such as algorithms, variables, conditional statements, and object-oriented programming.

The aim of this research is to create a software product—an educational game—that allows users of varying levels of knowledge to study programming concepts in an accessible, engaging, and adaptive manner. Within the scope of the work, current gamified learning platforms (such as CodeCombat, Tynker, Scratch, CodinGame) were analyzed, and their practical use was evaluated to determine the key requirements for an effective interactive learning system. Based on this analysis, the requirements for the proposed game were defined: intuitive interface, support for multiple programming languages, level progression system, instant feedback, and a flexible modular architecture.

The thesis outlines the design process of the game's architecture, the implementation of core functional modules (user registration, language selection, level system, task validation), and the user interface design based on UX/UI best practices. A modern technology stack was used for development: React and Tailwind CSS for the front-end, Node.js with Express for the back-end, PostgreSQL for data storage, and Vite as the build tool.

Special attention was paid to gamification: users receive rewards, level up, and track their learning progress, which helps maintain motivation. Functional and

usability testing confirmed the effectiveness and user-friendliness of the developed product.

ЗАВДАННЯ

на випускню кваліфікаційну бакалаврську роботу студентці:

Шеремет Анні Романівні

1. Тема роботи “Розробка навчальної гри для вивчення концепцій програмування”

2. Затверджена наказом по університету від “ ” 202 р. № .

3. Цільова установка на вихідні дані до роботи

Метою розробки є створення інтерактивної навчальної гри, яка дозволяє ефективно вивчати базові концепції програмування за допомогою гейміфікованого підходу. Вихідні дані повинні забезпечити реалізацію адаптивного освітнього середовища з підтримкою кількох мов програмування (зокрема Python, JavaScript), системи прогресу користувача, інтерактивних завдань із миттєвим зворотним зв'язком, а також інтерфейсу, орієнтованого на UX/UI-принципи.

До вихідних вимог належать:

- забезпечення реєстрації та авторизації користувачів;
- збереження й відображення прогресу;
- підтримка навчальних курсів з різних мов;
- реалізація гейміфікаційних елементів (нагороди, рівні, бали);
- можливість тестування знань у форматі вправ або вікторин;
- інтеграція з базою даних для зберігання статистики користувача;
- кросплатформеність (працездатність у браузері на ПК та мобільних пристроях).

Ці вихідні дані лягають в основу розробки веб-платформи навчальної гри та визначають її функціональну архітектуру.

4. Перелік графічного матеріалу (з точним визначенням обов'язкових креслень)

Рис 1.1. – Платформа CodeCombat – ст. 9

Рис 1.2. – Платформа CodinGame – ст. 11

Рис 1.3. – Платформа Tynker - ст. 13

Рис 1.4. – Платформа Kodable – ст. 13

Рис 1.5. – Платформа Scratch – ст. 15

Рис 1.6. – Платформа Robocode – ст. 17

Рис 1.7. – Платформа Lightbot – ст. 17

Рис 1.8. – Платформа CodeAcademy – ст. 20

Рис 1.9. – Платформа freeCodeCamp – ст. 22

Рис 1.10. – Платформа Coursera – ст. 24

Рис 1.11. – Платформа Udemy – ст. 24

Рис 1.12. – Платформа Pluralsight – ст. 25

Рис 1.13. – Платформа Khan Academy – ст. 26

Рис 1.14. – Платформа LeetCode – ст. 26

Рис 1.15. – Платформа edX – ст. 27

Рис 1.16. – Платформа HackerRank – ст. 28

Рис 2.1. – React – ст. 35

Рис 2.2. – Angular – ст. 35

Рис 2.3. – Vue.js – ст. 36

Рис 2.4. – Tailwindcss – ст. 37

Рис 2.5. – Bootstrap – ст. 37

Рис 2.6. – Node.js – ст. 38

Рис 2.7. – Express.js – ст. 38

Рис 2.8. – PostgreSQL – ст. 39

Рис 2.9. – MongoDB – ст. 40

Рис 2.10. – Amazon Web Services – ст. 41

Рис 2.11. – Google Cloud – ст. 41

Рис 2.12. – Google Analytics – ст. 42

Рис 2.13. – Mixpanel – ст. 43

Рис 2.14. – Vite – ст. 44

Рис 2.15. – Прототип дизайну – ст. 50

Рис 2.16. – Прототип дизайну 2 – ст. 50

Рис 2.17. – Прототип дизайну 3 – ст. 50

Рис 2.18. – Вибір мови програмування – ст. 53

Рис 2.19. – CodeInput – ст. 53

Рис.2.20 – Робот – ст. 54

Рис 2.21. – Система зворотного зв'язку – ст. 54

Рис 2.22. – Система балів – ст. 57

Рис 2.23. – Схвальне повідомлення – ст. 57

Рис 3.1. – Тестове завдання – ст. 60

Рис 3.2. – Модальне вікно реєстрації – ст. 61

Рис 3.3. – Вибір мови програмування – ст. 75

Рис 3.4. – Вибір режиму – ст. 75

Рис 3.5. – Завдання виконано не правильно – ст. 75

Рис 3.6. – Завдання виконано правильно – ст. 76

5. Консультанти по роботі, із зазначенням розділів, що їх стосуються:

Розділ	Науковий керівник	Підпис, дата	
		Завдання видав	Завдання отримав
1	Яровий Р. О.	03.10.24	20.10.24
2	Яровий Р. О.	10.10.24	25.10.24
3	Яровий Р. О.	25.10.24	30.10.24

6. Зміст випускної кваліфікаційної бакалаврської роботи (перелік

питань)

Аналіз предметної області та постановка задачі

Огляд навчальних ігор для вивчення програмування

Аналіз популярних платформ для вивчення мов програмування

Постановка задачі: створення платформи для вивчення мов програмування

Розробка архітектури та інтерфейсу навчальної гри

Вибір інструментів та технологій для реалізації платформи

Проектування інтерфейсу користувача

Реалізація та тестування навчальної гри

Реалізація базових функцій платформи

Інтеграція системи навчальних курсів для різних мов програмування

Тестування функціональності та юзабіліті гри

7. Календарний план виконання випускної кваліфікаційної бакалаврської роботи

№ з/п	Назва етапів випускної кваліфікаційної бакалаврської роботи	Термін виконання етапів випускної кваліфікаційної бакалаврської роботи	
		За планом	Фактично
1	Обрання теми випускної кваліфікаційної магістерської роботи (ВКБР)	05.08.24 – 13.09.24	30.09.24
2	Опрацювання літератури та інших джерел для написання ВКБР	16.09.24 – 01.11.24	03.10.24
3	Виконання 1 розділу ВКБР	04.11.24 – 28.02.25	28.02.25
4	Виконання 2 розділу ВКБР	03.03.25 – 21.03.25	21.03.25
5	Виконання 3 розділу ВКБР, висновки	24.03.25 – 18.04.25	14.04.25
6	Передзахист та проведення перевірки на плагіат	21.04.24 – 02.05.25	02.05.25
7	Відгук керівника, рецензія	01.05.25 – 12.05.25	12.05.25

8. Дата видачі завдання “ ” 202 р.

Керівник _____

Завдання прийняв до виконання _____

Зміст

ВСТУП.....	3
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА	
ЗАДАЧІ.....	8
1.1.Огляд навчальних ігор для вивчення програмування.....	8
1.2.Аналіз популярних платформ для вивчення мов програмування.....	18
1.3.Цільова аудиторія та вимоги до системи.....	28
1.4.Постановка задачі: створення платформи для вивчення мов програмування.....	32
РОЗДІЛ 2 . РОЗРОБКА АРХІТЕКТУРИ ТА ІНТЕРФЕЙСУ НАВЧАЛЬНОЇ	
ГРИ.....	34
2.1. Вибір інструментів та технологій для реалізації платформи.....	34
2.2. Опис функціональних модулів системи.....	44
2.3. Проектування інтерфейсу користувача	47
2.4. Розробка системи прогресу та мотивації.....	55
РОЗДІЛ 3 . РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ НАВЧАЛЬНОЇ	
ГРИ.....	59
3.1. Реалізація базових функцій платформи	59
3.2. Інтеграція системи навчальних курсів для різних мов програмування.....	65
3.3. Тестування функціональності та юзабіліті гри.....	72
3.4. Оцінка ефективності навчання та плани на подальший розвиток платформи.....	79
ВИСНОВКИ.....	84

ПОСИЛАННЯ І ЛІТЕРАТУРА.....	87
ДОДАТКИ.....	90

ВСТУП

Актуальність теми. У сучасному світі програмування стає невід'ємною складовою багатьох професійних галузей, зокрема в ІТ, науці, інженерії та бізнесі. З розвитком цифрових технологій зростає попит на фахівців, які володіють програмними навичками. Тому проблема навчання програмування набула особливої актуальності, що стимулює створення нових методів та підходів до засвоєння основ програмування як серед новачків, так і серед більш досвідчених спеціалістів.

Незважаючи на наявність численних онлайн-курсів та платформ для вивчення мов програмування, таких як Codecademy, Coursera та Duolingo (який також почав впроваджувати програмування у свій навчальний контент), більшість із них орієнтовані на певні рівні користувачів або конкретні методи навчання. Проте, як свідчать сучасні дослідження, ефективність навчання значно зростає, коли використовується інтерактивний підхід, що поєднує ігрові елементи та мотиваційні механізми.

Розробка навчальної гри для вивчення концепцій програмування, яка буде працювати за принципами, схожими на Duolingo, є актуальним кроком у розвитку сучасних технологій навчання. Така платформа дозволить інтегрувати гейміфікацію, покрокове навчання, мотиваційні компоненти, а також гнучке адаптування до різних рівнів знань користувача. Гра буде спрямована на поглиблене розуміння програмування, інтегруючи теорію та практику в єдину платформу. Це дозволить не лише опанувати синтаксис мов програмування, але й засвоїти фундаментальні концепції, такі як алгоритми, структури даних, об'єктно-орієнтоване програмування та інші основи.

У контексті розвитку галузі освіти, створення подібної гри дозволить розширити спектр навчальних можливостей для студентів, самостійних учнів та всіх, хто бажає опанувати програмування. Проєкт буде виконано у

відповідності з програмами кафедри інформаційних систем та математичних дисциплін та спрямований на вирішення завдань, пов'язаних із вдосконаленням методів цифрового навчання.

Мета і завдання дослідження. Метою даної дипломної роботи є розробка інтерактивної навчальної гри, яка дозволить користувачам ефективно вивчати основні концепції програмування через гейміфіковані методи навчання. Створена платформа буде спрямована на поступове опанування користувачами таких концепцій, як алгоритми, структури даних, об'єктно-орієнтоване програмування, шляхом інтеграції ігрових елементів, подібних до платформи Duolingo. Основною метою є створення веб-додатка, який допоможе користувачам різних рівнів володіння програмуванням у цікавій формі здобувати знання.

Для досягнення цієї мети необхідно провести аналіз предметної області, дослідити існуючі платформи для вивчення програмування, зокрема з елементами гейміфікації, для визначення сильних і слабких сторін цих підходів. Важливим етапом є визначення вимог до системи та потреб цільової аудиторії, що сприятиме ефективному проектуванню архітектури та інтерфейсу платформи.

Ключовим завданням також є проектування архітектури веб-додатка, вибір відповідних технологій для реалізації клієнт-серверної взаємодії та створення інтуїтивно зрозумілого інтерфейсу користувача. Окрім того, необхідно впровадити ігрову механіку з рівнями складності, завданнями та системою мотивації (нагороди, досягнення), яка стимулюватиме користувачів до проходження навчального курсу. Одним із важливих аспектів є створення навчальних матеріалів, що охоплюватимуть різні мови програмування та програмні концепції.

Реалізація гри передбачає розробку функціональних модулів, таких як реєстрація користувачів, вибір мов програмування, виконання завдань і перевірка правильності рішень. Після цього необхідно провести тестування гри

для виявлення недоліків та покращення користувацького досвіду, а також оптимізувати платформу на основі результатів тестування.

Завершальним завданням буде оцінка ефективності навчання користувачів за допомогою гри, порівняно з традиційними методами вивчення програмування, що дозволить оцінити доцільність та перспективи подальшого розвитку платформи.

Об'єктом дослідження є процес навчання програмування, зокрема засвоєння базових концепцій та принципів програмування початківцями. Цей процес є фундаментально важливим для формування навичок у галузі інформаційних технологій, оскільки володіння базовими знаннями програмування є основою для подальшого розвитку професійних компетенцій. Процес навчання програмування починається з вивчення основних понять, таких як змінні, цикли, умовні оператори, функції, а також принципів алгоритмічного мислення, що допомагають зрозуміти логіку програмних рішень.

Однак навчання програмуванню початківців супроводжується низкою труднощів, серед яких можна виділити складність сприйняття абстрактних концепцій, високу кількість нової інформації та вимогу швидкого засвоєння нових інструментів і мов програмування. У багатьох випадках традиційні методи навчання, такі як лекції та теоретичні вправи, не в змозі достатньо залучити студентів до активного процесу засвоєння знань. Це призводить до того, що багато учнів відчують низьку мотивацію, важкість у розумінні матеріалу та відчуття фрустрації через невдачі на перших етапах навчання.

Останні тенденції в освітньому процесі вказують на те, що використання ігрових елементів (гейміфікація) у навчанні може стати потужним інструментом для подолання цих труднощів. Ігрові механіки, такі як досягнення, рівні, змагання, а також система нагород, стимулюють інтерес і підтримують мотивацію учнів на всіх етапах навчання. Ігрові середовища створюють контекст, у якому учні можуть експериментувати, пробувати різні підходи до розв'язання проблем та вчитися на власних помилках без страху

перед невдачею. Це знижує психологічні бар'єри та полегшує процес освоєння програмування.

Окрім того, інтеграція інтерактивних вправ і миттєвого зворотного зв'язку дає змогу учням бачити результати своїх дій у реальному часі, що значно покращує якість засвоєння матеріалу. Такий підхід дає можливість краще розуміти помилки та отримувати рекомендації щодо їх виправлення. Таким чином, об'єкт дослідження охоплює не лише сам процес навчання програмуванню, але й ті фактори, які впливають на його ефективність, включаючи мотиваційні аспекти, способи подання матеріалу та використання інноваційних підходів для підвищення залученості учнів.

Тому дослідження цього процесу є надзвичайно важливим, оскільки воно дозволяє виявити нові шляхи підвищення ефективності навчання програмуванню та розробити нові методології, що можуть бути застосовані у сучасних освітніх практиках.

Предмет дослідження. Предметом дослідження є розробка інтерактивної навчальної гри для вивчення концепцій програмування, яка базується на методах гейміфікації. Основна увага зосереджується на створенні ефективних навчальних механік, що поєднують ігрові елементи з освітнім контентом для полегшення засвоєння базових програмних концепцій. Це включає в себе розробку архітектури платформи, ігрових механік, інтерфейсу та інтеграцію навчальних матеріалів, які сприяють підвищенню мотивації та результативності вивчення програмування.

Практичне значення одержаних результатів. Практичне значення одержаних результатів полягає у створенні інтерактивної навчальної платформи для вивчення основних концепцій програмування, що може бути використана в різних галузях освіти. Розроблена гра сприяє підвищенню мотивації та зацікавленості у навчанні програмування завдяки використанню гейміфікованих методик, які поєднують ігрові елементи з навчальним контентом. Така платформа може бути інтегрована у навчальні курси для

студентів ІТ-напрямів, а також використовуватися у школах, коледжах та університетах для самостійного чи додаткового навчання.

Отримані результати можуть бути впроваджені у навчальні програми різних освітніх установ, зокрема у рамках курсів з основ програмування. Платформа також може стати корисною для організацій, які займаються підготовкою ІТ-спеціалістів, шляхом інтеграції її у тренінги чи корпоративні програми навчання. Рекомендації щодо використання розробленої гри можуть бути надані викладачам, що працюють у сфері ІТ-освіти, для підвищення рівня засвоєння знань студентами.

У разі подальшого розвитку та вдосконалення платформи можливе її впровадження в широкому масштабі у навчальних закладах, які спеціалізуються на підготовці програмістів.

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1.Огляд навчальних ігор для вивчення програмування

У сучасному світі навчальні ігри набувають дедалі більшої популярності, особливо у сфері освіти, зокрема програмування. Вони дозволяють поєднувати процес навчання з елементами гри, що сприяє підвищенню мотивації та ефективності засвоєння матеріалу. Навчальні ігри для програмування допомагають не тільки вивчати синтаксис мов програмування, але й розвивати логічне мислення, вирішувати складні алгоритмічні завдання та навчатися через експерименти.

Одним із найвідоміших прикладів навчальних ігор для вивчення програмування є **CodeCombat** (рис. 1.1.), яка дозволяє користувачам опановувати програмування через інтерактивний ігровий процес. Гравцям пропонується писати код для управління персонажами, вирішуючи різні завдання, що пов'язані з основами програмування. CodeCombat підтримує популярні мови програмування, такі як Python і JavaScript, що дає змогу користувачам не лише вчитися, але й обирати ту мову, яка найбільше відповідає їхнім інтересам або потребам.

Однією з головних переваг CodeCombat є його поступова складність. На перших етапах гравці виконують прості завдання, що дозволяє їм ознайомитися з базовими концепціями програмування, такими як змінні, цикли, умовні оператори. З кожним новим рівнем завдання стають складнішими, поступово вводяться нові елементи синтаксису та алгоритми, що сприяє розвитку

логічного мислення і глибшого розуміння програмування. Такий підхід забезпечує плавний перехід від початкового рівня до більш складного, що дозволяє уникнути відчуття перевантаженості інформацією.

Ігровий процес побудований таким чином, що учні вчаться на своїх помилках і отримують негайний зворотний зв'язок. Це сприяє швидшому засвоєнню матеріалу і робить навчання менш стресовим. Крім того, використання персонажів і сюжетних ліній створює додаткову мотивацію для учнів, оскільки вони не лише розв'язують абстрактні завдання, але й керують подіями у грі, що робить процес навчання більш захопливим і цікавим.

Важливо відзначити, що CodeCombat також застосовує елементи гейміфікації, такі як досягнення, нагороди та змагання з іншими гравцями. Це створює конкурентне середовище, що мотивує учнів виконувати завдання на вищому рівні та продовжувати навчання. Крім того, система нагород сприяє формуванню відчуття прогресу та задоволення від виконання складних завдань, що є важливим аспектом у довгостроковому навчанні.

CodeCombat є прикладом того, як ігрові елементи можуть інтегруватися в навчальний процес, роблячи його більш інтерактивним і ефективним. Завдяки своїй гнучкості та широкій аудиторії, ця гра стала важливим інструментом у навчанні програмуванню як для дітей, так і для дорослих, що прагнуть покращити свої навички.

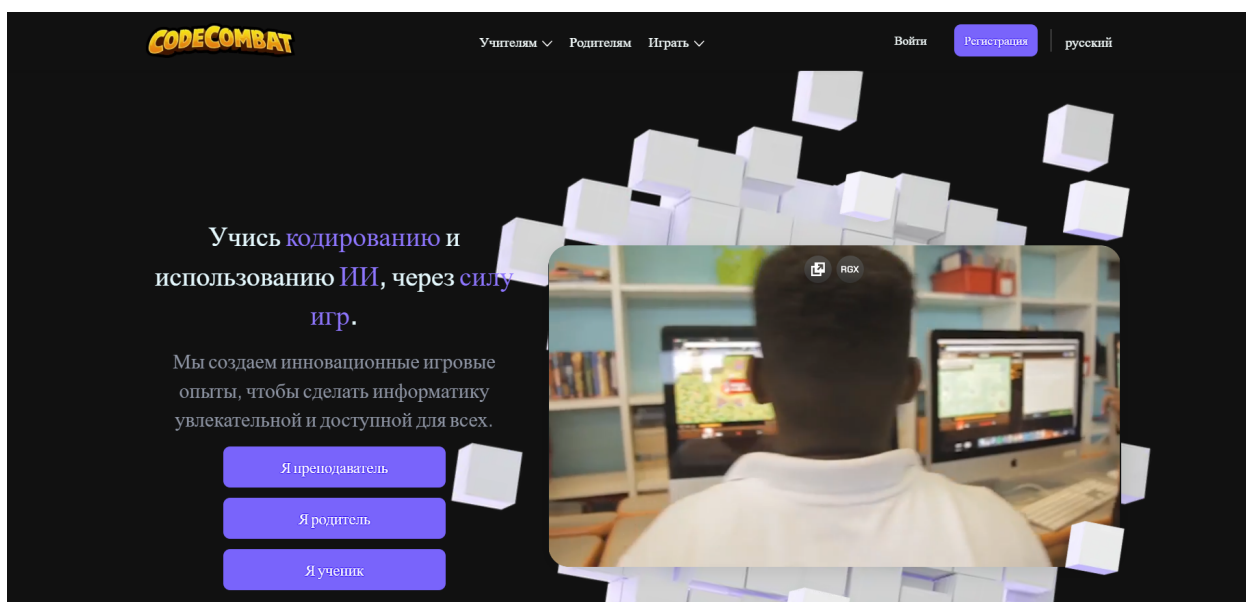


Рис 1.1. – Платформа CodeCombat

Інший відомий приклад – **CodinGame**, (рис 1.2.) платформа, яка поєднує програмування з ігровими елементами. CodinGame — це інтерактивна платформа для навчання програмуванню, яка поєднує елементи гейміфікації та змагання. Вона орієнтована як на новачків, так і на досвідчених розробників, пропонуючи широкий спектр завдань, які можна вирішувати в більш ніж 25 мовах програмування, серед яких Python, Java, C++, JavaScript та багато інших.

Основна особливість CodinGame полягає у форматі навчання, що базується на вирішенні алгоритмічних завдань у вигляді ігрових сценаріїв. Користувачі отримують виклики, які потрібно виконати шляхом написання коду, а результати їхніх рішень відображаються у вигляді інтерактивної візуалізації. Це дозволяє не лише бачити правильність виконаних дій, а й аналізувати різні підходи до вирішення проблем.

Окрім стандартних алгоритмічних завдань, платформа підтримує змагання між користувачами, що стимулює розвиток їхніх навичок. Регулярно проводяться онлайн-турніри, де програмісти можуть змагатися один з одним у режимі реального часу. Це додає елемент суперництва і мотивує до вдосконалення знань.

CodinGame також має режим мультиплеєра, де користувачі можуть розробляти власних ботів, які змагаються між собою в програмованих ігрових симуляціях. Це чудова можливість для вдосконалення навичок алгоритмічного програмування, оптимізації коду та стратегічного мислення.

Платформа є не лише навчальним середовищем, а й місцем для рекрутингу. Багато компаній використовують CodinGame для пошуку талановитих розробників, оскільки платформа дозволяє оцінити реальні навички кандидатів через вирішення практичних завдань. Це робить CodinGame корисною як для тих, хто хоче покращити свої навички, так і для тих, хто шукає можливості працевлаштування в ІТ-сфері.

Таким чином, CodinGame є унікальним прикладом інтеграції навчання та ігрових механік, що робить процес вивчення програмування більш цікавим, ефективним і мотивуючим для користувачів різного рівня.

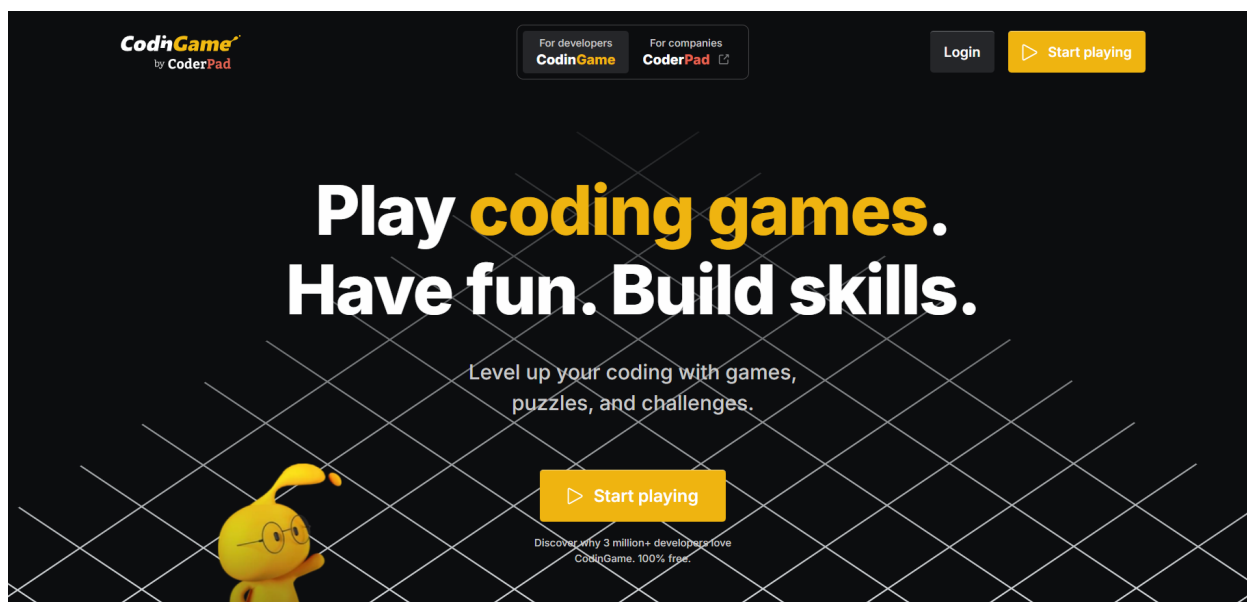


Рис 1.2. – Платформа CodinGame

Ще один цікавий приклад — це **Tynker** (рис. 1.3.), платформа, орієнтована на молодшу аудиторію, яка використовує візуальне програмування для навчання дітей основ програмування. Інтерфейс Tynker базується на блоках, які діти можуть перетягувати та комбінувати для створення програм.

Такий підхід дозволяє спрощено представити основи програмування, роблячи навчання цікавим та доступним для дітей.

Завдяки блоковому підходу Tynker дозволяє користувачам опанувати основні концепції програмування, такі як цикли, умови, функції та змінні, без необхідності одразу вивчати складний синтаксис мов програмування. Це сприяє кращому розумінню логіки програмування, оскільки діти можуть наочно бачити, як їхні дії впливають на результат програми.

Що робить Tynker особливо привабливою для дітей, так це її гейміфікований підхід до навчання. Завдання поділяються на інтерактивні ігрові рівні, де діти вирішують різноманітні головоломки, створюють власні ігри або анімації. Це не тільки робить процес навчання захоплюючим, але й стимулює розвиток креативного мислення та навичок вирішення проблем.

Крім того, платформа підтримує можливість переходу від візуального програмування до написання реального коду. Після того як діти засвоїли основи логіки через блоки, вони можуть перейти до таких мов програмування, як JavaScript або Python, зокрема, в рамках просунутих курсів Tynker. Це забезпечує плавний перехід від візуальних концепцій до текстового коду, дозволяючи учням розширювати свої знання.

Платформа також пропонує навчальні плани для вчителів та батьків, що робить її корисною в освітньому середовищі. Вчителі можуть використовувати Tynker як інструмент для інтеграції програмування у шкільну програму, навіть якщо самі не мають глибоких знань у цій сфері. Створені спеціальні плани уроків, тести та інтерактивні завдання, які можна використовувати для оцінки знань та прогресу учнів.

Tynker також пропонує дітям можливість брати участь у конкурсах та викликах, де вони можуть демонструвати свої проекти і досягнення, що додає елемент змагання та мотивації для подальшого навчання. Ця платформа сприяє розвитку як технічних, так і креативних навичок, забезпечуючи широкий спектр можливостей для дітей, які цікавляться програмуванням.

Загалом, Tynker — це потужний інструмент для навчання програмуванню дітей, який поєднує в собі легкість візуального програмування та інтерактивність ігрових завдань. Вона допомагає молодій аудиторії легко та весело опановувати основи програмування, забезпечуючи необхідні базові знання для подальшого розвитку в цій галузі.

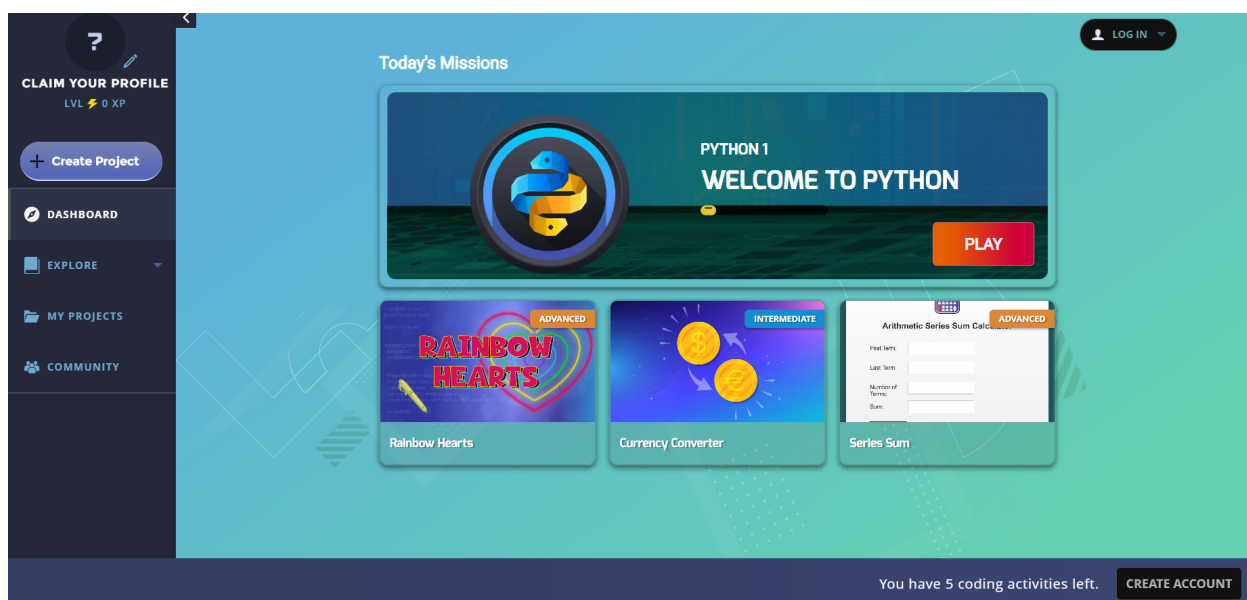


Рис 1.3. – Платформа Tynker

Також варто згадати **Kodable**, (рис 1.4.) яка створена для дітей молодшого шкільного віку і навчає їх основам програмування через гру, використовуючи аналогічні блокові методи кодування. Поступово гра вводить дітей у світ програмування, допомагаючи розвивати критичне мислення.

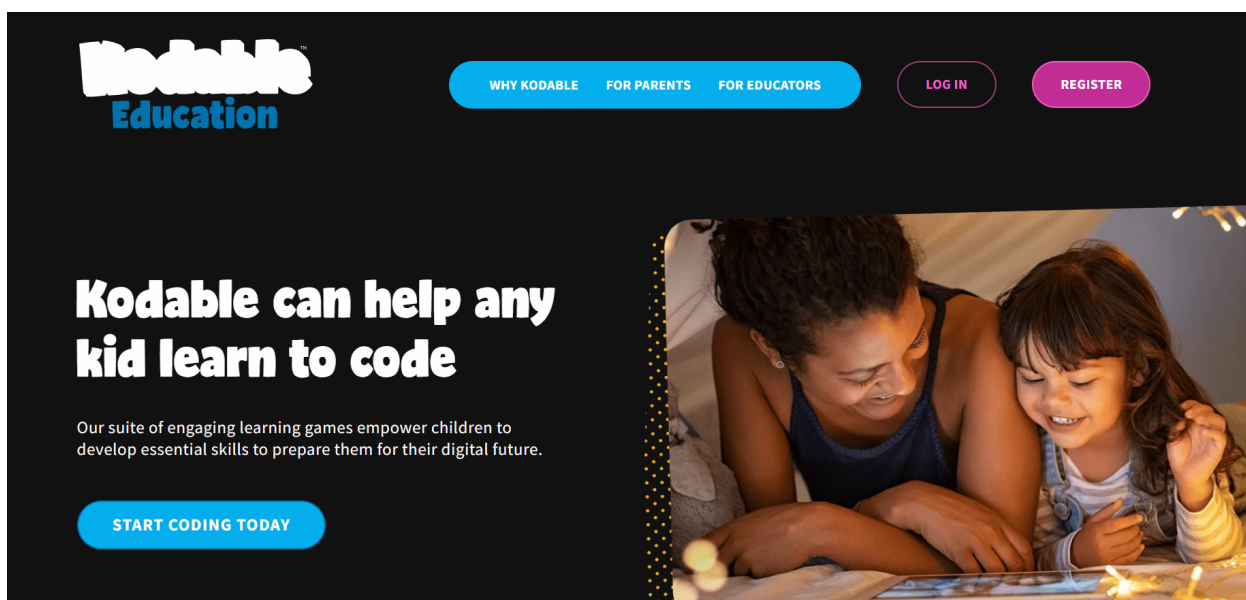


Рис 1.4. – Платформа Kodable

Нарешті, не можна оминати увагою платформу **Scratch** (рис. 1.5.), яка є однією з найбільш відомих і популярних для навчання дітей та підлітків основам програмування. Scratch використовує блокове програмування, що дозволяє створювати програми, перетягуючи графічні елементи — блоки коду — в інтерфейсі платформи. Цей підхід допомагає дітям зрозуміти логіку програмування без необхідності вивчати складний синтаксис традиційних мов програмування.

Основною метою Scratch є сприяння розвитку креативного мислення у молодих користувачів, надаючи їм можливість створювати власні ігри, анімації, історії та інтерактивні проекти. Оскільки платформа орієнтована на дітей віком від 8 років, вона надає їм простий інтуїтивно зрозумілий інтерфейс, що полегшує процес навчання, роблячи його веселим і захоплюючим.

Важливим аспектом Scratch є те, що він дає можливість вивчати основи алгоритмів, логіки програмування та розробки ігор, не потребуючи глибоких знань у програмуванні. Створення анімацій або інтерактивних проектів дозволяє дітям зрозуміти принципи циклів, умовних операторів, змінних і багато інших важливих понять, що лежать в основі програмування. За

допомогою блоків, які можна з'єднувати між собою, користувачі формують програму, яка працює як звичайний код, але без складності написання тексту.

Особливістю Scratch є те, що вона підтримує спільну роботу та обмін проектами серед користувачів. Це дає можливість дітям вчитися разом, обговорювати свої ідеї та навіть брати участь у конкурсах або творчих проектах, що сприяє розвитку їхніх навичок і покращує мотивацію до навчання. Користувачі можуть також публікувати свої роботи на платформі, отримувати відгуки від інших учасників і удосконалювати свої проекти за допомогою коментарів.

Крім того, Scratch має величезну спільноту вчителів і батьків, які використовують платформу для підтримки навчання своїх дітей. Це дозволяє забезпечити додаткові ресурси, матеріали для навчання та інтерактивні уроки, що допомагають молодим програмістам краще освоїти платформу та поглибити свої знання в області програмування.

Завдяки своїй доступності, простоті використання та можливості створювати творчі проекти, Scratch стала основним інструментом для багатьох шкіл і освітніх установ по всьому світу. Вона є чудовим стартом для дітей, які хочуть вивчати програмування і розвивати свої креативні навички в цій сфері, а також для батьків, які хочуть допомогти своїм дітям освоїти базові принципи програмування через гру і творчість.

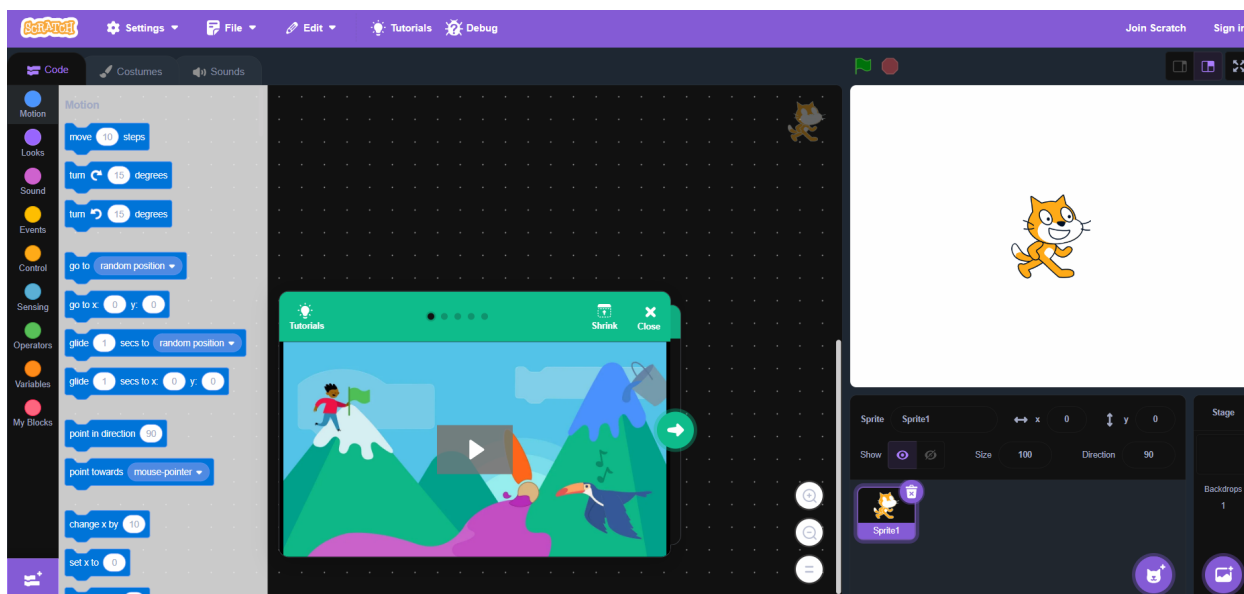


Рис 1.5. – Платформа Scratch

Robocode(рис 1.6.) — це навчальна гра, яка дозволяє користувачам програмувати бойові танки для участі у віртуальних битвах. Головна мета гравця — розробити ефективний алгоритм управління танком, щоб перемогти суперників у симуляційному бою. Гра побудована на мові Java, але також підтримує програмування на інших JVM-сумісних мовах, таких як Kotlin та Scala.

Однією з ключових особливостей Robocode є можливість застосовувати складні алгоритмічні концепції на практиці. Гравці можуть використовувати структури даних, машинне навчання, стратегічне планування та оптимізацію руху танка в реальному часі. Завдяки цьому Robocode є не лише розважальним, а й потужним освітнім інструментом для навчання алгоритмічного мислення, обробки подій і створення адаптивних систем.

Геймплей побудований на взаємодії між танками, кожен з яких діє на основі написаного користувачем коду. Програмісту потрібно визначити, як танк буде реагувати на рухи суперників, як ухилятися від атак і коли відкривати вогонь. Гра включає систему обробки подій, де танки можуть виявляти один одного за допомогою радіолокаційних сенсорів, аналізувати швидкість і напрямок ворога та приймати рішення відповідно до зібраної інформації.

Для підтримки спільноти Robocode має активний форум і базу знань, де користувачі можуть ділитися своїми алгоритмами, стратегіями та вдосконалювати свої навички через вивчення коду інших учасників. Крім того, існує багато відкритих змагань, де програмісти можуть тестувати свої навички проти ботів інших користувачів, що додає грі елемент конкуренції та мотивації.

Robocode використовується в навчальних закладах як платформа для вивчення основ штучного інтелекту, розробки автономних систем і візуалізації алгоритмічних рішень. Завдяки своїй інтерактивності та можливості реалізації складних стратегій гра стала популярною серед студентів та ентузіастів програмування, які хочуть застосувати свої знання в практичному, динамічному середовищі.

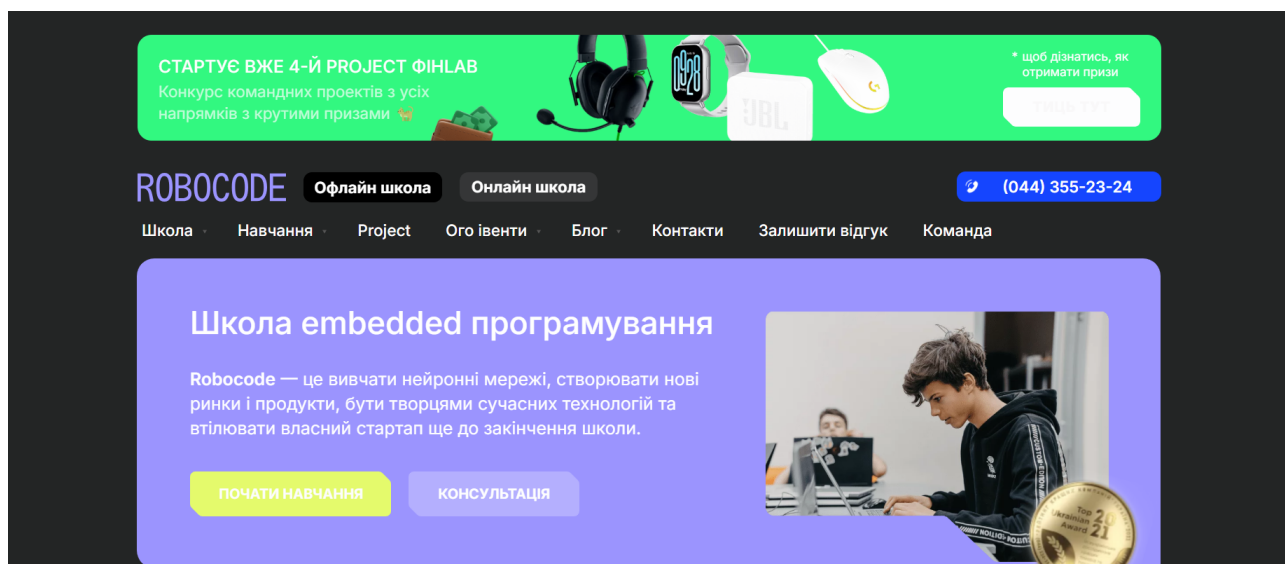


Рис 1.6. – Платформа Robocode

Інноваційний підхід використовує гра **Lightbot**(рис 1.7.), що дозволяє вивчати базові принципи алгоритмів і логічного мислення навіть дітям. Користувачі керують роботом, виконуючи серії завдань, що навчають таких концепцій, як цикли, умовні оператори та функції. Це робить Lightbot

ідеальним інструментом для наймолодших учнів, оскільки він починає з основ програмування

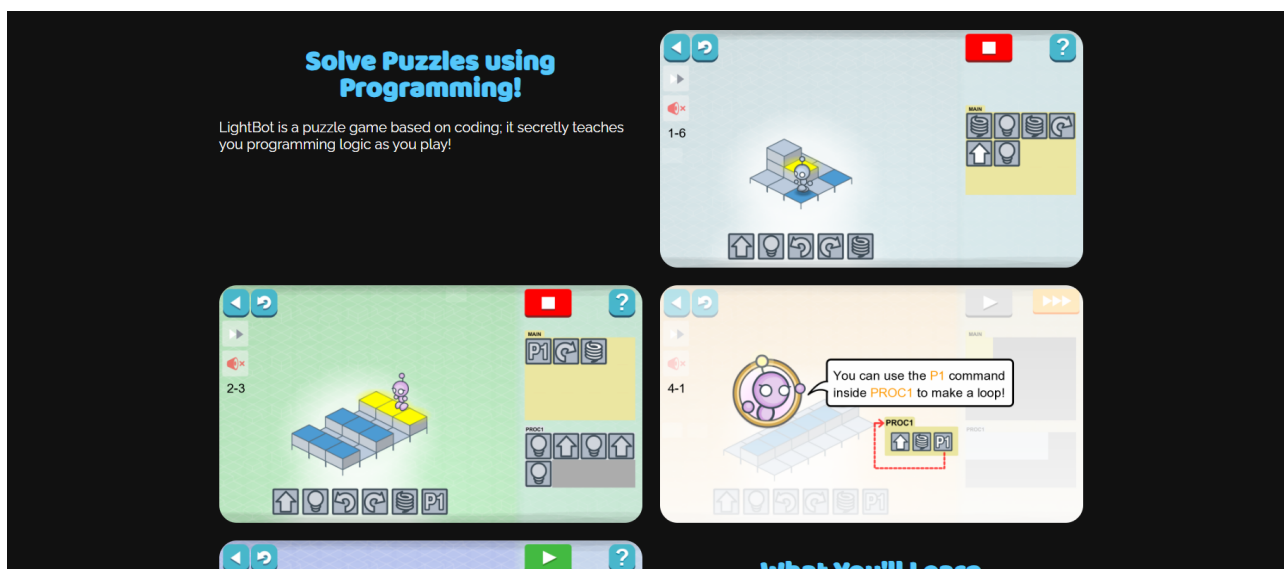


Рис 1.7. – Платформа Lightbot

Таким чином, існує велика кількість навчальних ігор та платформ, які допомагають вивчати програмування через інтерактивний ігровий процес. Ці платформи відрізняються як за цільовою аудиторією, так і за підходами до навчання, але всі вони спрямовані на те, щоб зробити процес вивчення програмування більш захопливим і доступним. Кожна з цих ігор має свої унікальні особливості та підходи до навчання програмуванню, проте всі вони об'єднуються тим, що роблять процес навчання більш цікавим та доступним. У цьому контексті важливо відзначити, що навчальні ігри дозволяють вивчати складні концепції поступово і в інтерактивний спосіб, що підвищує залучення і мотивацію користувачів.

1.2. Аналіз популярних платформ для вивчення мов програмування

Останні роки спостерігається зростання кількості онлайн-платформ, які пропонують користувачам можливість вивчати мови програмування в інтерактивному режимі. Ці платформи дають змогу опановувати програмування в зручний для користувачів час та з використанням інноваційних методик навчання. Кожна з них пропонує унікальний набір інструментів і методів, які роблять процес навчання ефективнішим. Сучасні платформи для вивчення мов програмування пропонують широкий спектр можливостей для навчання, поєднуючи теоретичні матеріали з практичними завданнями. Їхня популярність обумовлена не тільки зручністю доступу, але й можливістю навчатися у власному темпі, що робить їх ідеальними для самостійного навчання.

Однією з найбільш популярних і визнаних платформ для вивчення програмування є **Codecademy** (рис. 1.8.). Вона пропонує інтерактивні курси з різних мов програмування, включаючи Python, JavaScript, HTML, CSS, SQL та інші, що робить її універсальним інструментом для навчання як для новачків, так і для більш досвідчених користувачів. Одна з головних переваг Codecademy полягає в інтерактивному підході до навчання — учні можуть одразу застосовувати нові знання на практиці, виконуючи завдання прямо в браузері. Це допомагає краще закріпити матеріал і зменшує розрив між теоретичними знаннями та практичними навичками.

Codecademy пропонує поступове навчання, починаючи з основних понять і поступово переходячи до більш складних тем. Завдання, які надаються під час курсів, супроводжуються чіткими інструкціями та роз'ясненнями, що робить платформу особливо зручною для новачків. Крім того, користувачі можуть скористатися інтерактивними підказками та автоматичною перевіркою коду, що дозволяє їм швидко виправляти помилки та одразу розуміти, де саме вони зробили помилку. Такий підхід сприяє ефективному та самостійному навчанню.

Важливо також відзначити, що Codecademy використовує адаптивну систему навчання, яка дозволяє кожному користувачу рухатися у власному темпі. Учні можуть вибирати конкретні теми або мови програмування, які їх цікавлять, і будувати індивідуальні траєкторії навчання. Це створює гнучкість у

навчальному процесі, що особливо важливо для тих, хто поєднує навчання з роботою або іншими обов'язками.

Крім того, Codecademy надає користувачам можливість створювати власні проекти на основі знань, отриманих під час навчання. Це дає змогу застосовувати на практиці навички, які були засвоєні в теоретичній частині, та вирішувати реальні завдання. Створення власних проектів допомагає учням підвищити свою впевненість у використанні мов програмування та надає відчуття прогресу.

Платформа також підтримує соціальне навчання, пропонуючи учасникам можливість спілкуватися через форуми та групи, де вони можуть обмінюватися досвідом, отримувати допомогу або обговорювати складні питання. Це створює відчуття спільноти, що є додатковим джерелом мотивації для продовження навчання.

Codecademy має як безкоштовний, так і платний контент. Преміум-підписка надає доступ до додаткових функцій, таких як персональні кар'єрні поради, глибші курси та реальні проекти для виконання. Це дозволяє користувачам з різними потребами отримати те, що їм потрібно, незалежно від рівня підготовки чи професійних амбіцій.

У цілому, Codecademy є потужним інструментом для навчання програмуванню, оскільки поєднує теоретичні знання з практикою, забезпечуючи учням доступ до високоякісного контенту, який допомагає освоїти основні мови програмування та розвивати практичні навички, необхідні в сучасній IT-індустрії.

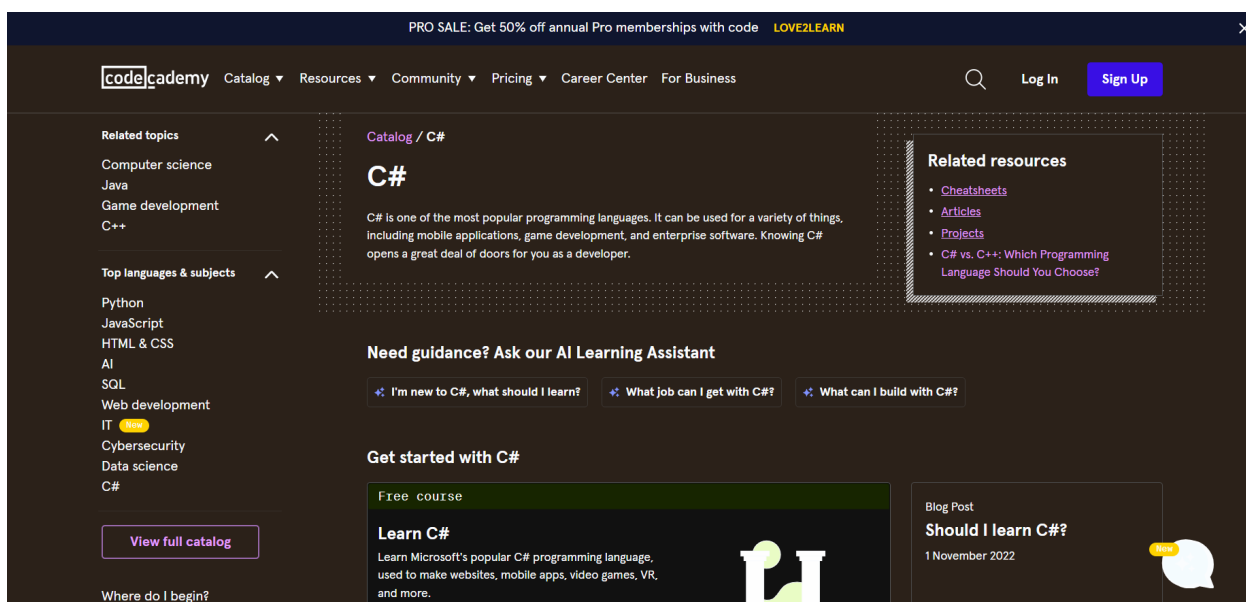


Рис 1.8. – Платформа CodeAcademy

Іншою відомою платформою є **freeCodeCamp**(рис 1.9.), яка безкоштовно надає доступ до великої кількості навчальних матеріалів з програмування. freeCodeCamp — це одна з найпопулярніших безкоштовних платформ для навчання програмуванню, яка пропонує широкий спектр навчальних матеріалів, інтерактивних завдань та реальних проєктів для практики. Головною метою платформи є надання доступної освіти всім охочим, незалежно від рівня підготовки та фінансових можливостей.

Навчальна програма freeCodeCamp структурована у вигляді сертифікаційних курсів, які охоплюють різні напрямки програмування, такі як веб-розробка, алгоритмічне програмування, аналіз даних, машинне навчання, безпека та DevOps. Кожен курс містить серію інтерактивних вправ, які дозволяють користувачам поступово освоювати нові концепції, а також практичні завдання, які допомагають закріпити знання через безпосереднє написання коду.

Однією з ключових переваг freeCodeCamp є можливість працювати над реальними проєктами, які імітують завдання, що зустрічаються в комерційній розробці. Наприклад, користувачі можуть створювати веб-сайти, API, інтерактивні веб-застосунки та навіть брати участь у благодійних проєктах,

допомагаючи неприбутковим організаціям автоматизувати їхні процеси. Це дає не лише практичний досвід, але й можливість наповнити власне портфоліо реальними кейсами, що суттєво підвищує шанси на отримання роботи у сфері ІТ.

Крім самостійного навчання, freeCodeCamp пропонує потужну спільноту, де користувачі можуть взаємодіяти, обговорювати проблеми, обмінюватися досвідом та отримувати допомогу від менторів та досвідчених розробників. Спільнота активно функціонує через форум, Discord, а також локальні зустрічі та хакатони, що допомагає новачкам швидше адаптуватися у сфері програмування.

Платформа також має YouTube-канал, на якому регулярно публікуються відеоуроки, курси та лекції від експертів, а також власний блог, де можна знайти докладні статті про технології, кар'єрний розвиток і новітні тренди у програмуванні.

Завдяки своїй структурі, безкоштовному доступу, інтерактивним завданням, практичним проєктам та сильній спільноті, freeCodeCamp є чудовим інструментом як для початківців, які тільки починають свій шлях у програмуванні, так і для досвідчених розробників, які хочуть поглибити свої знання або змінити напрямок роботи.

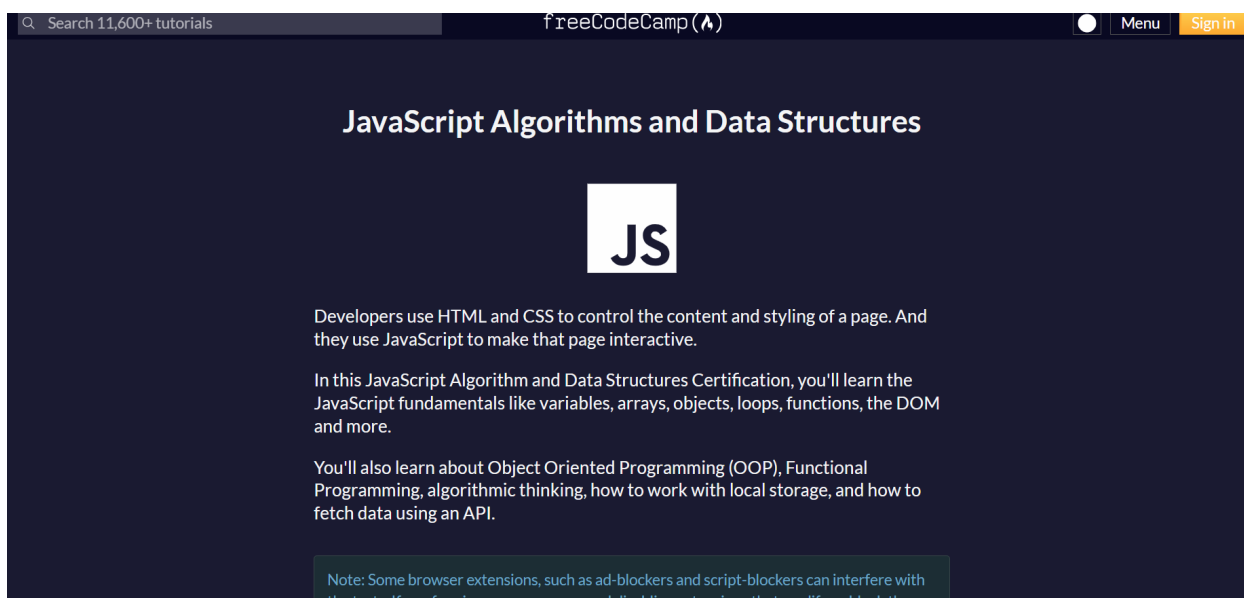


Рис 1.9. – Платформа freeCodeCamp

Coursera (рис. 1.10.) є ще однією популярною платформою для навчання програмуванню, яка пропонує широкий вибір курсів, розроблених провідними університетами та навчальними закладами з усього світу. Завдяки співпраці з такими відомими університетами, як Стенфорд, Мічиганський університет, а також багатьма іншими інституціями, Coursera надає доступ до освітнього контенту високої якості, що робить платформу надзвичайно привабливою для людей, які прагнуть отримати знання з програмування або інших суміжних дисциплін.

Однією з основних переваг Coursera є те, що курси, представлені на платформі, охоплюють не лише основи програмування, але й спеціалізовані теми, такі як машинне навчання, штучний інтелект, розробка мобільних додатків та веб-розробка. Це дозволяє користувачам поступово розвивати свої навички, починаючи з базових концепцій і досягати поглибленого розуміння складних тем, що мають практичне застосування в сучасній індустрії ІТ.

Платформа пропонує курси з різних мов програмування, таких як Python, Java, C++, JavaScript та багато інших. Завдяки цьому користувачі можуть обирати навчальні програми відповідно до своїх інтересів або професійних потреб. Крім того, Coursera забезпечує високий рівень підтримки з боку інструкторів та спільноти учасників курсу, що робить процес навчання більш інтерактивним та ефективним.

Ще однією важливою особливістю Coursera є можливість отримання сертифікатів після завершення курсів. Це дозволяє користувачам підтвердити свої знання і навички, що може бути корисним при пошуку роботи або для подальшого професійного розвитку. Сертифікати від провідних університетів та компаній є вагомим додатком до резюме, і їх визнають багато роботодавців по всьому світу.

Coursera також відзначається своїм гнучким підходом до навчання. Курси на платформі можна проходити у зручному темпі, що дозволяє поєднувати

навчання з роботою чи іншими обов'язками. Більшість курсів мають структуровані завдання, які допомагають закріпити отримані знання через практику, а також проміжні тести для оцінки прогресу.

Крім того, Coursera пропонує спеціалізації та професійні сертифікати, які є наборами курсів, об'єднаних спільною тематикою або напрямом. Це дозволяє користувачам отримати глибші знання у певній галузі та стати більш конкурентоспроможними на ринку праці. Така структура допомагає чітко визначити навчальні цілі та досягати їх через поступовий прогрес.

Загалом, Coursera є платформою, яка відкриває можливості для якісної освіти в галузі програмування та інформаційних технологій, пропонуючи не тільки окремі курси, але й цілі навчальні програми, що підходять для різних рівнів підготовки — від новачків до професіоналів. Завдяки своїй репутації, інтеграції з провідними університетами та можливості отримати сертифікат, Coursera залишається одним з найкращих виборів для тих, хто хоче розвиватися в IT-сфері.

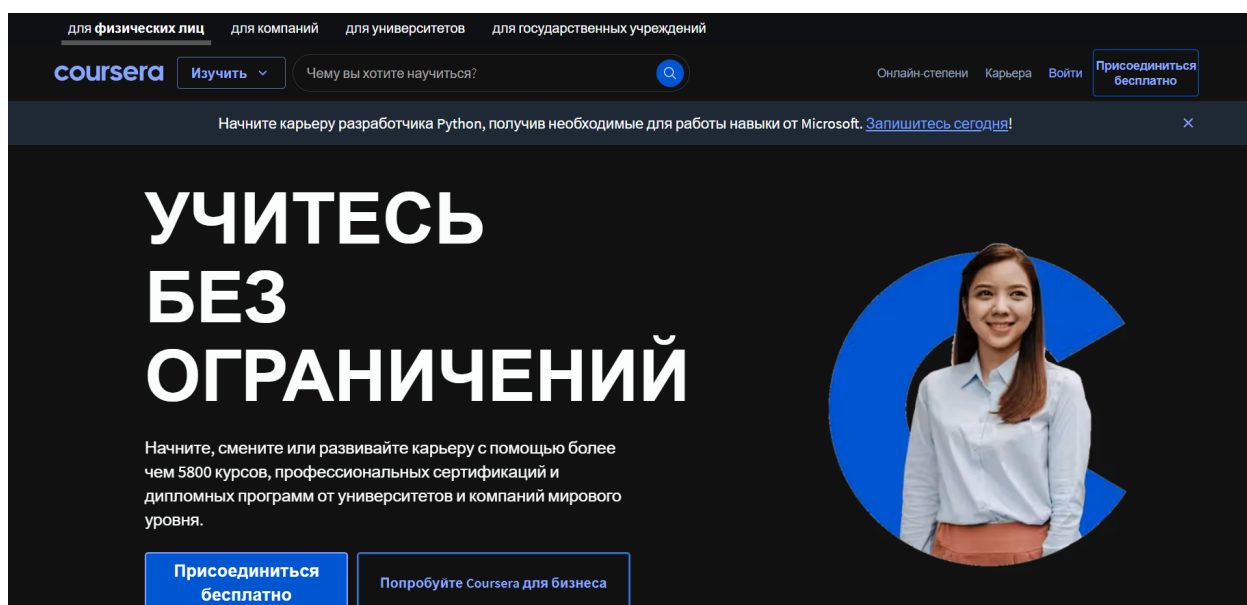


Рис 1.10. – Платформа Coursera

Платформа **Udemy**(рис 1.11.) також надає широкий вибір курсів з програмування, як для початківців, так і для досвідчених розробників. Udemy

відома своїм доступом до курсів з багатьох мов програмування, і особливістю платформи є те, що кожен викладач може створювати і продавати свої курси. Це дозволяє користувачам отримувати доступ до різноманітних підходів і стилів навчання.

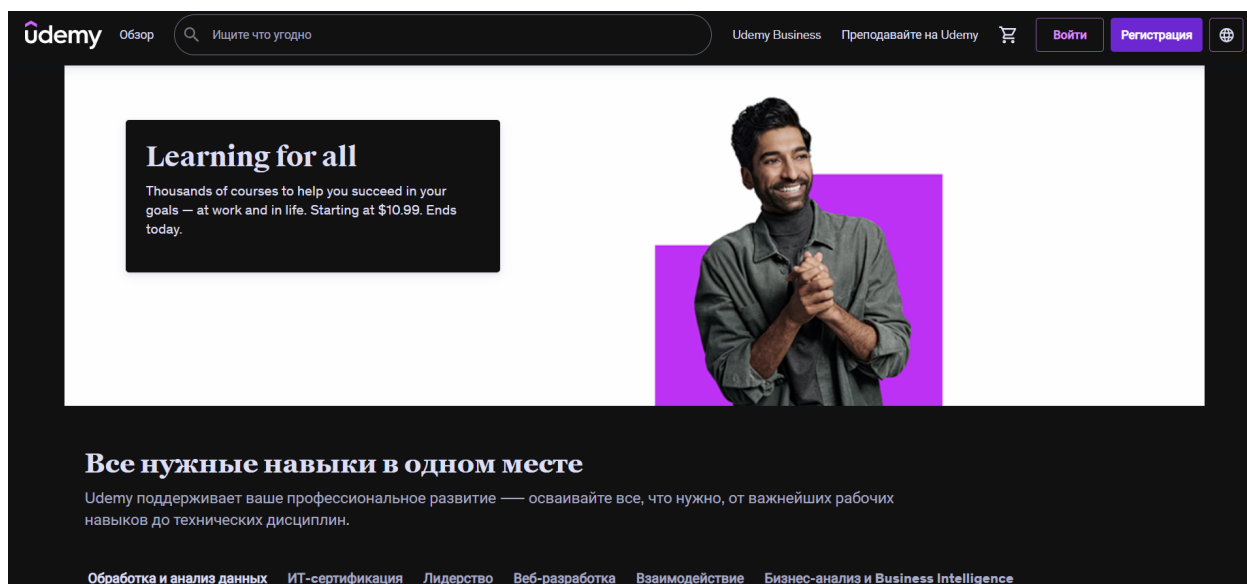


Рис 1.11. – Платформа Udemy

Pluralsight(рис 1.12.) є ще однією платформою, яка пропонує курси для розробників з різним рівнем підготовки. Вона фокусується на професійному розвитку програмістів і надає доступ до курсів, що охоплюють не лише мови програмування, а й такі аспекти, як кібербезпека, робота з базами даних та хмарні технології. Однією з ключових переваг Pluralsight є можливість проходження оцінок навичок, які допомагають користувачам визначити свої слабкі сторони і працювати над їх поліпшенням.

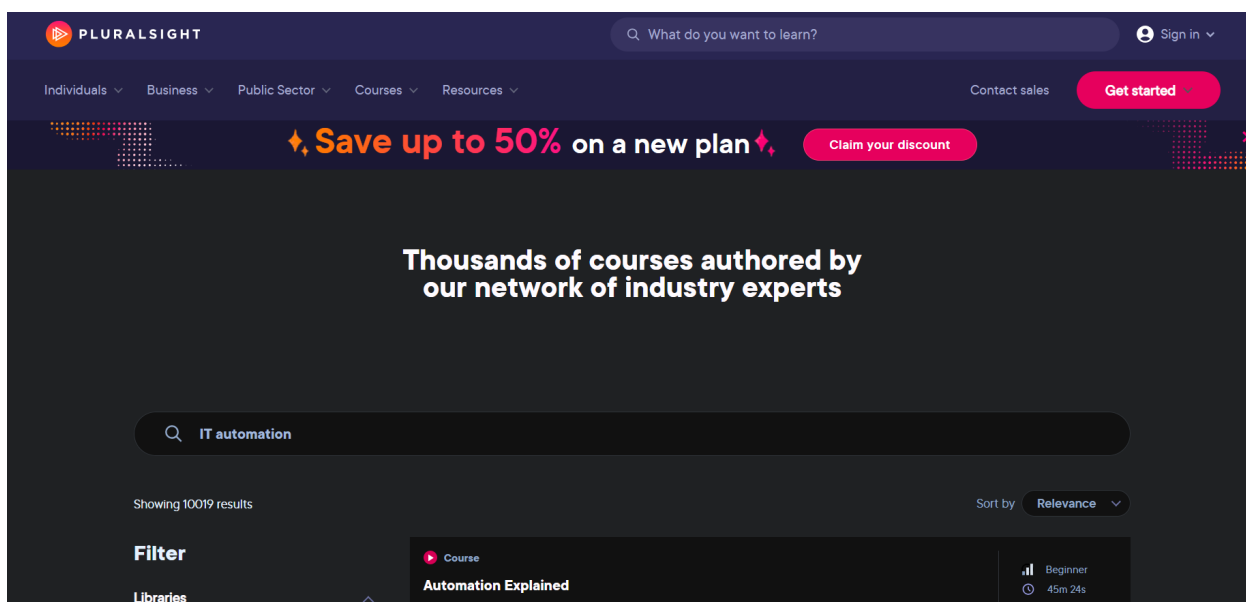


Рис 1.12. – Платформа Pluralsight

Не можна оминуту увагою і платформу **Khan Academy**(рис 1.13.), яка пропонує курси з основ програмування, а також надає пояснення алгоритмів та структур даних. Вона відома своїм доступним підходом до навчання, де складні концепції пояснюються простою мовою. Платформа підходить для тих, хто тільки починає свій шлях у програмуванні.

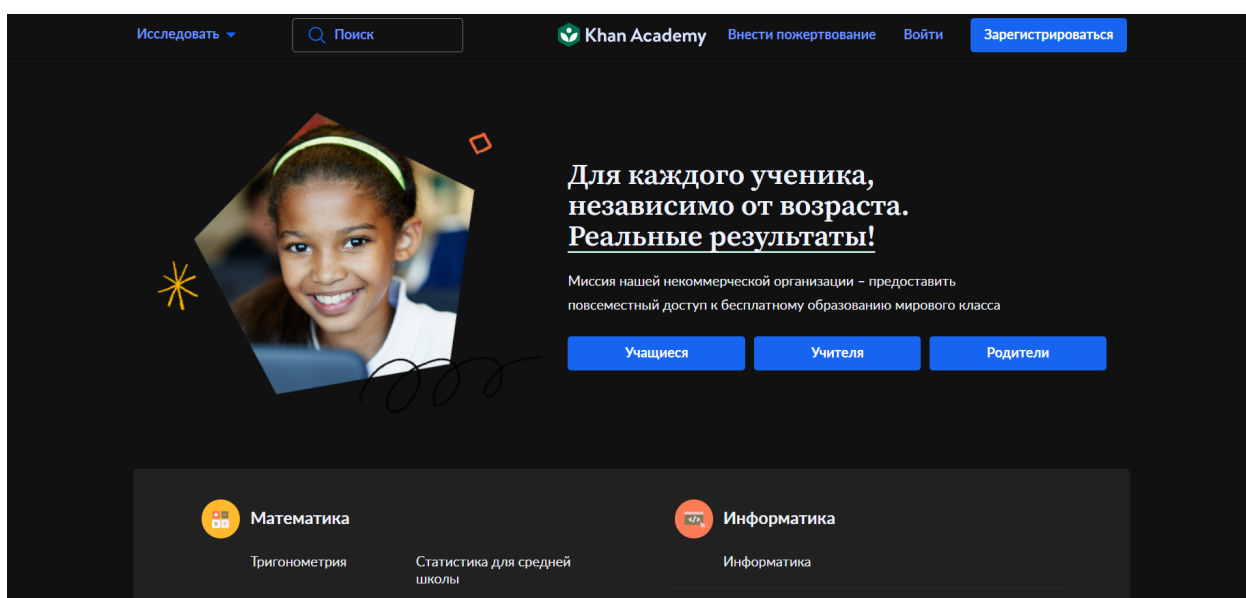


Рис 1.13. – Платформа Khan Academy

Останньою, але не менш важливою, є платформа **LeetCode**(рис 1.14.), яка спеціалізується на завданнях для підготовки до технічних інтерв'ю. LeetCode пропонує широкий спектр задач з різних мов програмування, фокусуючись на розв'язанні алгоритмічних проблем. Ця платформа є однією з найкращих для підготовки до співбесід у великих ІТ-компаніях, таких як Google, Facebook, Amazon.

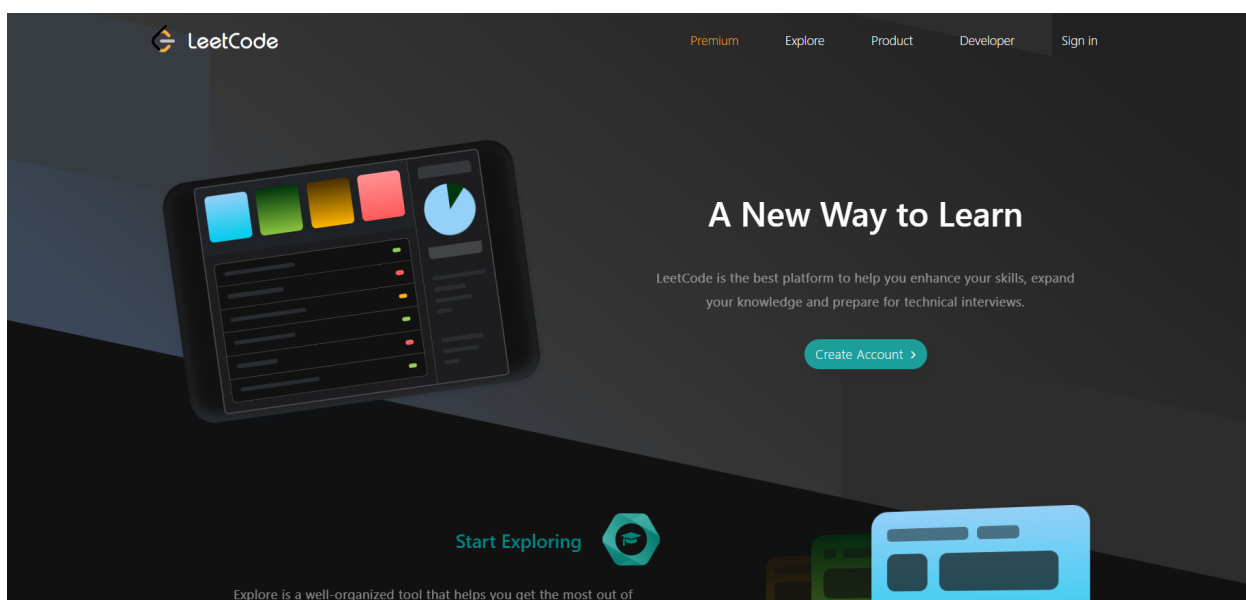


Рис 1.14. – Платформа LeetCode

edX(рис 1.15.) — це ще одна відома платформа, яка, як і Coursera, працює у співпраці з університетами. Вона пропонує курси з програмування на таких мовах, як Python, C, Java та інші. edX надає можливість не тільки отримати теоретичні знання, але й пройти сертифікаційні програми, що дозволяють закріпити кваліфікацію.

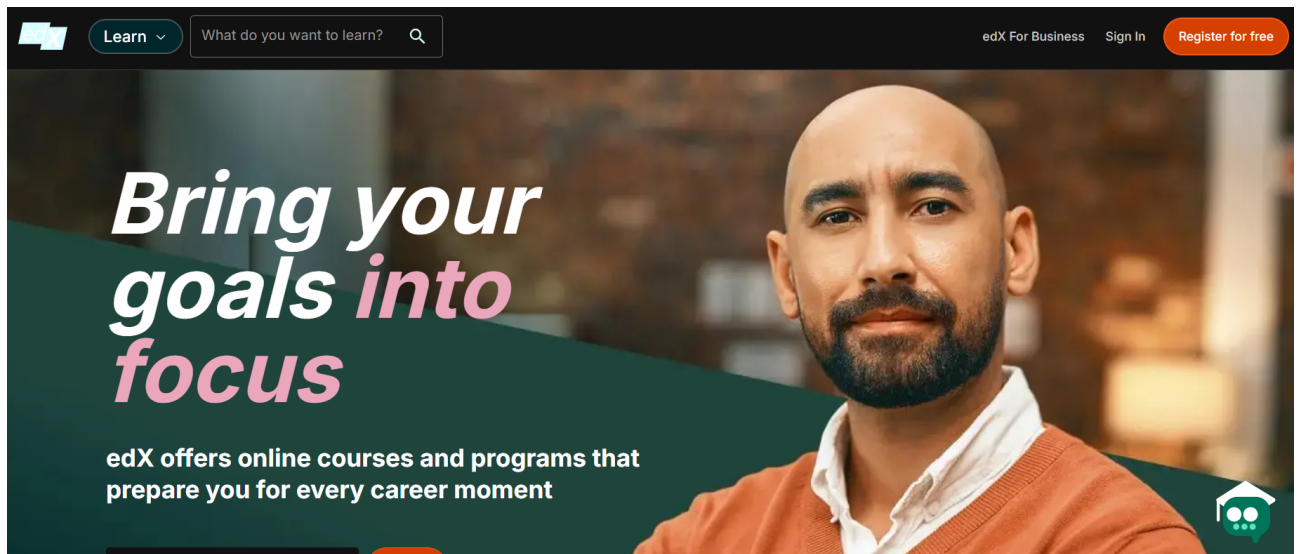


Рис 1.15. – Платформа edX

Також варто відзначити платформу **HackerRank**(рис 1.16.), яка орієнтована на практичні завдання з програмування та проведення конкурсів. Користувачі можуть розв'язувати задачі різної складності на багатьох мовах програмування, змагаючись із іншими користувачами. HackerRank дозволяє не лише навчатися, а й перевіряти свої навички, беручи участь у челенджах і конкурсах.

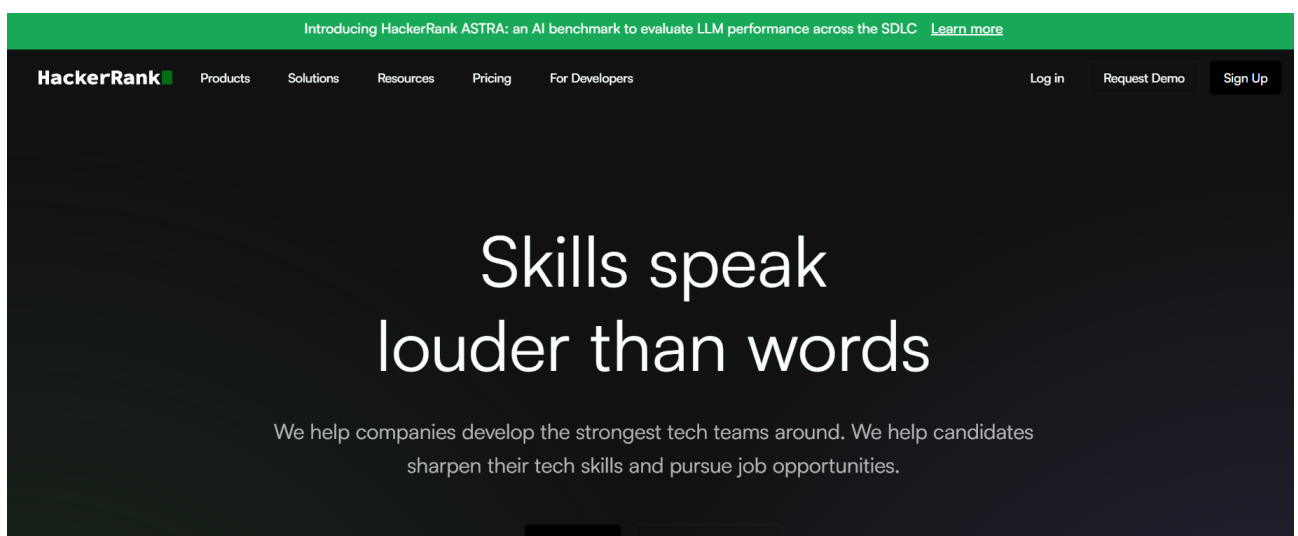


Рис 1.16. – Платформа HackerRank

Кожна з цих платформ має свої сильні сторони та специфіку, але всі вони сприяють навчанню програмуванню через практику і теоретичні знання. Завдяки інтерактивному підходу, користувачі мають можливість швидше опановувати матеріал і застосовувати його на практиці, що робить процес навчання більш ефективним та цікавим. Усі ці платформи пропонують різноманітні формати навчання, що включають відеоуроки, інтерактивні вправи, реальні проекти та змагання. Вони дозволяють адаптувати навчання до індивідуальних потреб користувачів, будь то новачки або досвідчені програмісти.

1.3.Цільова аудиторія та вимоги до системи

Цільова аудиторія навчальної гри для вивчення програмування охоплює широкий спектр користувачів, включаючи як початківців, так і тих, хто вже має певний досвід у програмуванні та бажає вдосконалити свої навички. Однією з ключових категорій користувачів є школярі та студенти, які тільки починають знайомитися з програмуванням. Для них гра може стати важливим інструментом для самостійного або додаткового навчання поза межами формальної освітньої програми. Вони можуть використовувати гру як засіб для закріплення матеріалу, який вивчається у школі чи університеті, а також для практики, яка допомагає краще зрозуміти складні концепції.

Другою важливою групою є дорослі люди, які вирішили змінити професію та прагнуть увійти до ІТ-сфери. Ця категорія користувачів, як правило, шукає практичні рішення, що дозволять швидко та якісно освоїти програмування з мінімальними витратами часу. Для них навчальна гра може стати ідеальним рішенням, оскільки поєднує інтерактивні методи навчання та елементи гри, які полегшують процес засвоєння нових знань.

Крім того, гра може зацікавити користувачів, які вивчають програмування у якості хобі. Це люди, які прагнуть розширити свій кругозір або просто отримують задоволення від вивчення нових технологій, не ставлячи

перед собою професійних цілей у цій галузі. Для них гра може бути засобом задоволення інтелектуальної цікавості та можливістю отримати нові знання у зручній та доступній формі.

Однією з таких груп є **роботодавці та компанії**, які можуть використовувати платформу для внутрішнього навчання своїх працівників або підвищення їхньої кваліфікації. Для компаній, що працюють у сфері ІТ, така платформа може стати важливим інструментом для оцінки та розвитку навичок співробітників. Додатково, ця аудиторія може використовувати платформу для тренінгів та індивідуальних планів розвитку для нових працівників.

Ще однією важливою аудиторією є освітні установи, зокрема викладачі та школи, які можуть використовувати гру як інструмент для залучення студентів до програмування. Гра здатна інтегруватися у навчальний процес та стати частиною шкільної чи університетської програми. Це сприятиме більш інтерактивному і цікавому підходу до вивчення програмування, що, у свою чергу, полегшить засвоєння складних тем і мотивуватиме учнів та студентів до активної участі в навчальному процесі.

Також варто розглянути **непрофесіоналів у галузі ІТ**, які можуть використовувати програмування для вирішення специфічних завдань у своїй професії. Це можуть бути маркетологи, фінансисти, дизайнери, які прагнуть автоматизувати свої робочі процеси або створювати додатки для власних потреб. Для них важливо, щоб платформа пропонувала прикладні курси з наголосом на реальні задачі, які вони можуть використати у своїй професійній діяльності.

Щодо вимог до системи, однією з ключових вимог є зручність користувацького інтерфейсу. Інтерфейс повинен бути інтуїтивно зрозумілим, простим та адаптованим до новачків у програмуванні. Навчальний процес повинен починатися з простих завдань і поступово ускладнюватися, забезпечуючи комфортне навчання навіть для тих, хто не має попереднього досвіду в програмуванні.

Платформа повинна підтримувати різні мови програмування, такі як Python, JavaScript, C++, Java та інші. Це дасть можливість користувачам обирати ті мови, які найбільше відповідають їхнім цілям та інтересам. Така гнучкість зробить платформу доступною для ширшого кола користувачів з різними професійними та навчальними потребами.

Ще однією важливою вимогою є кросплатформенність системи. Навчальна гра повинна бути доступною як на стаціонарних комп'ютерах, так і на мобільних пристроях. Це дозволить користувачам вивчати програмування у будь-який час і в будь-якому місці, що забезпечить гнучкість у процесі навчання. Завдяки цьому більше користувачів зможуть користуватися платформою, незалежно від того, де і на якому пристрої вони воліють вчитися.

Модульність системи також є важливою вимогою. Система повинна мати гнучку архітектуру, що дозволить легко додавати нові навчальні матеріали, вправи та ігрові елементи. Це дасть змогу платформі постійно розвиватися та відповідати сучасним вимогам ринку освітніх технологій і програмування.

Гейміфікація повинна бути органічно інтегрована у навчальний процес. Ігрові елементи, такі як рівні, досягнення, нагороди, допоможуть мотивувати користувачів продовжувати навчання. Важливо, щоб ці елементи не відволікали від навчання, а навпаки, підтримували прогрес у засвоєнні програмування. Це сприятиме підвищенню зацікавленості користувачів у навчанні та покращенню результатів.

Важливою є **доступність контенту**, що передбачає створення адаптивної системи навчання для людей із різними рівнями здібностей та знань. Наприклад, користувачам із вадами слуху можуть бути запропоновані субтитри до відеоматеріалів, а для людей з вадами зору – можливість збільшення тексту або використання озвучування.

Соціальні інтеграції також є важливим аспектом сучасних навчальних платформ. Впровадження можливостей для спільного навчання, форумів для обговорень або інтеграція з соціальними мережами може підвищити залученість користувачів. Соціальний компонент допоможе створити відчуття

спільноти, що буде мотивувати користувачів продовжувати навчання, ділитися успіхами, запитувати поради та допомогу.

Індивідуалізація навчання – це ще один критичний аспект. Платформа повинна вміти адаптуватися до різних типів учнів, їхнього темпу навчання та потреб. Можливість аналізувати прогрес і пропонувати рекомендації щодо покращення або додаткові матеріали стане важливою складовою для підтримки мотивації та забезпечення результативного навчання.

Інтерактивні вправи та зворотний зв'язок є ще однією ключовою вимогою до системи. Користувачі повинні мати змогу отримувати миттєвий зворотний зв'язок після виконання вправ, що допоможе краще зрозуміти свої помилки та швидко їх виправляти. Система повинна автоматично аналізувати виконані завдання і надавати рекомендації щодо їх покращення.

Підтримка прогресу користувачів також є важливим аспектом. Система повинна відстежувати та зберігати результати виконаних завдань, надавати можливість переглядати досягнуті успіхи. Це дозволить користувачам бачити свій прогрес та стимулюватиме до подальшого навчання. Загалом, платформа повинна бути не лише технічно вдосконаленою, але й орієнтованою на зручність і ефективність навчання для різних аудиторій, незалежно від рівня їхніх знань чи мети навчання.

1.4.Постановка задачі: створення платформи для вивчення мов програмування

Постановка задачі передбачає створення платформи для вивчення мов програмування, яка надаватиме користувачам можливість інтерактивного та поступового засвоєння базових і просунутих знань у сфері програмування. Основним завданням платформи є забезпечення доступного і зручного навчального процесу, який буде цікавим для широкого кола користувачів – від новачків до людей з певним досвідом.

Однією з головних задач є створення інтерактивного, гнучкого та доступного середовища для навчання. Платформа повинна враховувати різні рівні підготовки користувачів, починаючи від новачків, які тільки знайомляться з основами програмування, до досвідчених фахівців, що прагнуть вдосконалити свої знання. Важливо, щоб платформа надавала можливість як для послідовного навчання з чітко визначеними рівнями складності, так і для вивчення конкретних тем або мов програмування, що цікавлять користувачів.

Наступним завданням є розробка адаптивної системи навчання, яка дозволяє користувачам вивчати матеріали у своєму темпі. Це особливо важливо для людей з різним рівнем попередньої підготовки або обмеженими ресурсами часу. Платформа має забезпечувати автоматичну адаптацію складності завдань на основі прогресу користувача, а також пропонувати додаткові матеріали та пояснення у випадку виникнення труднощів.

Платформа повинна мати інтуїтивно зрозумілий інтерфейс, який дозволить користувачам легко орієнтуватися в матеріалах і вправлятися у написанні коду в різних мовах програмування, таких як Python, JavaScript, C++ та інші. Основною особливістю платформи стане інтерактивність, що полягає у виконанні реальних завдань і отриманні миттєвого зворотного зв'язку.

Платформа повинна містити елементи гейміфікації, які допоможуть підтримувати мотивацію користувачів на високому рівні. Це можуть бути рівні складності, нагороди за успішне виконання завдань, лідерборди, а також інші елементи, що зроблять процес навчання більш захопливим і стимулюючим.

Крім того, важливою задачею є забезпечення інтерактивності платформи. Вона повинна містити різні типи завдань, такі як тестові завдання, практичні завдання на написання коду, а також інтерактивні вправи з миттєвим зворотним зв'язком. Така система дозволить користувачам швидко оцінити свої результати, проаналізувати помилки та отримати рекомендації щодо покращення. Миттєвий зворотний зв'язок є критично важливим для підтримки мотивації користувачів.

Однією з додаткових задач є забезпечення можливості інтеграції з іншими освітніми платформами або додатками. Це може включати інтеграцію з системами управління навчанням (LMS), платформами для зберігання сертифікатів або іншими навчальними ресурсами. Така інтеграція дозволить створити більш комплексну освітню екосистему для користувачів.

Важливою вимогою до системи є її кроссплатформенність, що дозволить користувачам навчатися на різних пристроях – як на стаціонарних комп'ютерах, так і на мобільних телефонах чи планшетах. Платформа також повинна бути адаптована до різних рівнів знань користувачів і надавати їм можливість вибирати індивідуальний темп навчання.

Система повинна забезпечувати моніторинг прогресу користувачів, зберігання результатів виконаних завдань і надавати можливість повернутися до будь-якого етапу навчання для повторення чи вдосконалення навичок.

Таким чином, основна задача полягає в створенні гнучкої, доступної та ефективної платформи для вивчення мов програмування, що задовольнить потреби різних категорій користувачів, дозволяючи їм швидко освоювати нові знання та застосовувати їх на практиці.

Загалом, платформа повинна вирішувати задачу створення доступного, інтерактивного та ефективного середовища для навчання програмуванню, яке зможе адаптуватися до індивідуальних потреб користувачів, стимулювати їхній розвиток і сприяти досягненню поставлених навчальних цілей.

РОЗДІЛ 2

РОЗРОБКА АРХІТЕКТУРИ ТА ІНТЕРФЕЙСУ НАВЧАЛЬНОЇ ГРИ

2.1. Вибір інструментів та технологій для реалізації платформи

Для створення ефективної та функціональної платформи для вивчення мов програмування, необхідно ретельно підійти до вибору інструментів та

технологій, які будуть використовуватися на всіх етапах розробки. Відповідний вибір залежить від таких факторів, як зручність для користувачів, масштабованість системи, підтримка мультимовності, інтерактивність та можливість впровадження гейміфікації. Також важливо враховувати доступність платформи на різних пристроях, зокрема на мобільних і десктопних платформах. Технологічні рішення повинні забезпечити високий рівень продуктивності та стабільності роботи системи.

Основним елементом будь-якої сучасної веб-платформи є її фронтенд-частина, яка відповідає за взаємодію користувача з системою. Для розробки фронтенду часто використовують JavaScript-фреймворки, такі як React(рис 2.1.), Angular(рис 2.2.) або Vue.js(рис 2.3.). Серед цих фреймворків, React може бути найкращим вибором завдяки своїй гнучкості, швидкості та активній спільноті. React дозволяє легко створювати динамічні інтерфейси користувача, що є важливим для інтерактивних елементів платформи, таких як вправи та завдання для вивчення програмування. Крім того, React має компонентну архітектуру, що дозволяє розробляти платформу як набір незалежних модулів, що полегшує підтримку та розширення функціоналу в майбутньому.

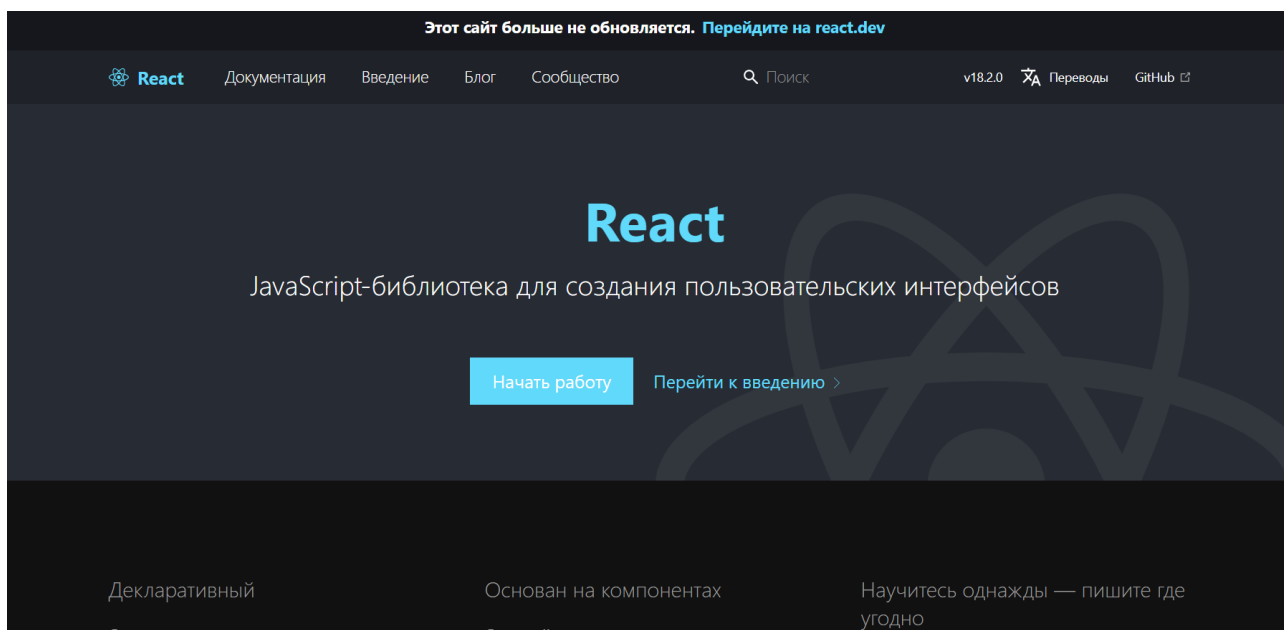


Рис 2.1. – React

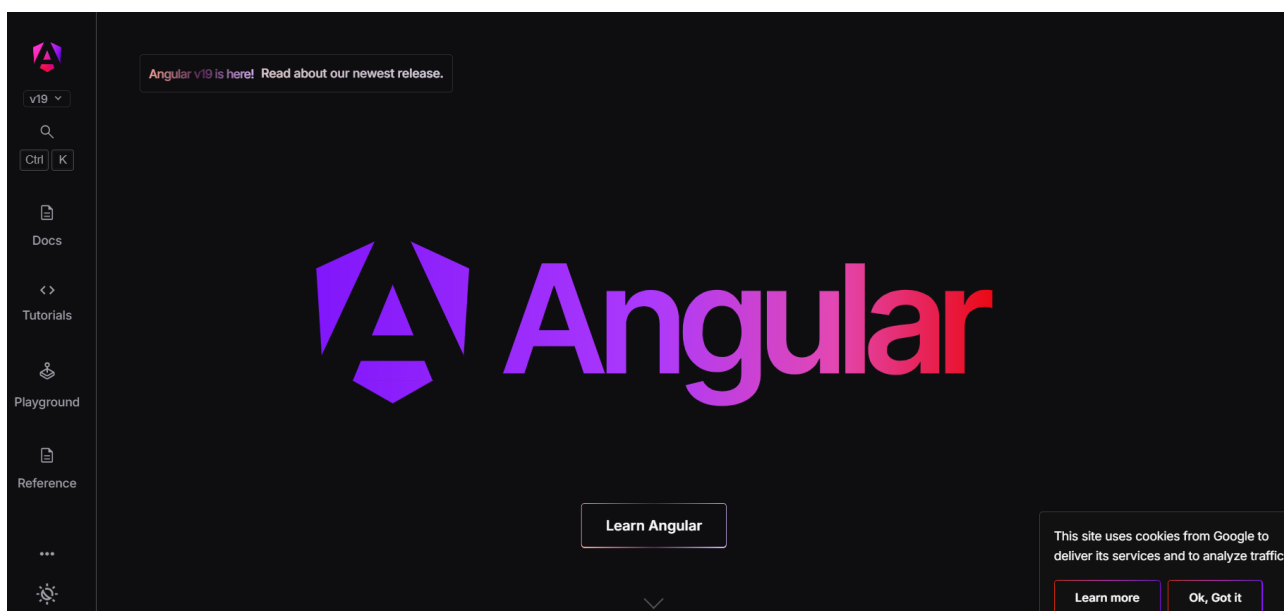


Рис 2.2. – Angular

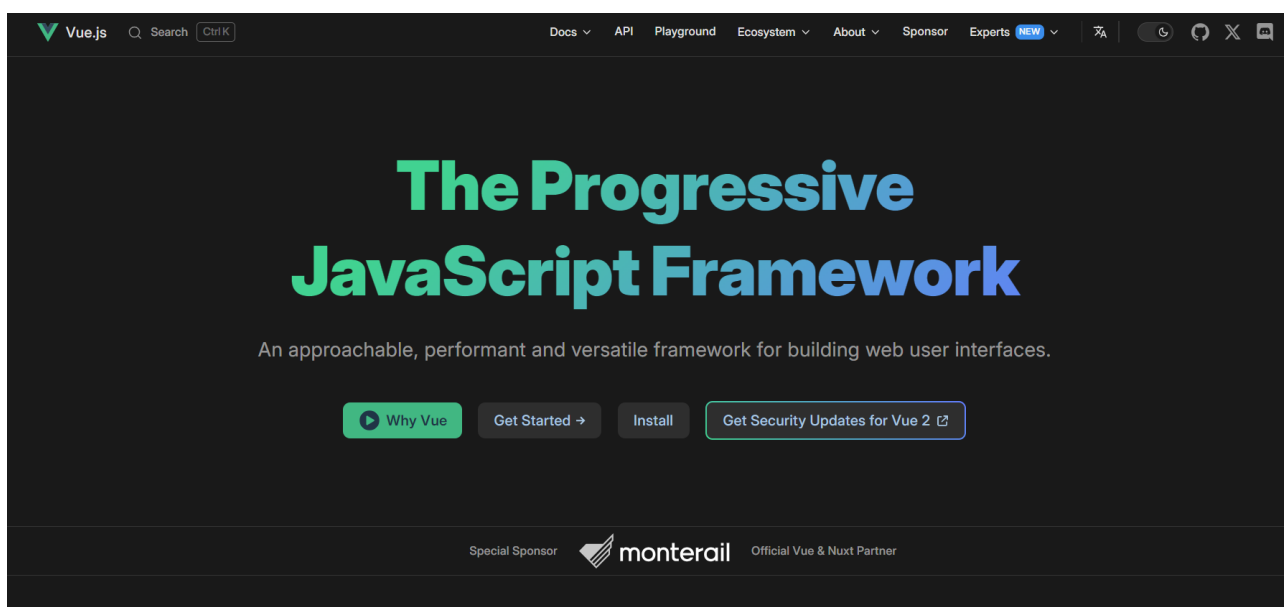


Рис 2.3. – Vue.js

Для забезпечення стильового оформлення платформи можна використовувати CSS-фреймворки, такі як Tailwind CSS(рис 2.4.) або Bootstrap(рис 2.5.). Tailwind CSS є сучасним інструментом, який дозволяє швидко створювати адаптивні інтерфейси, використовуючи утилітарні класи. Це надає можливість зменшити кількість кастомного CSS-коду і швидше розробляти сучасні дизайни. Адаптивність є важливим фактором, оскільки

платформа повинна коректно працювати на різних екранах — від мобільних телефонів до великих моніторів.

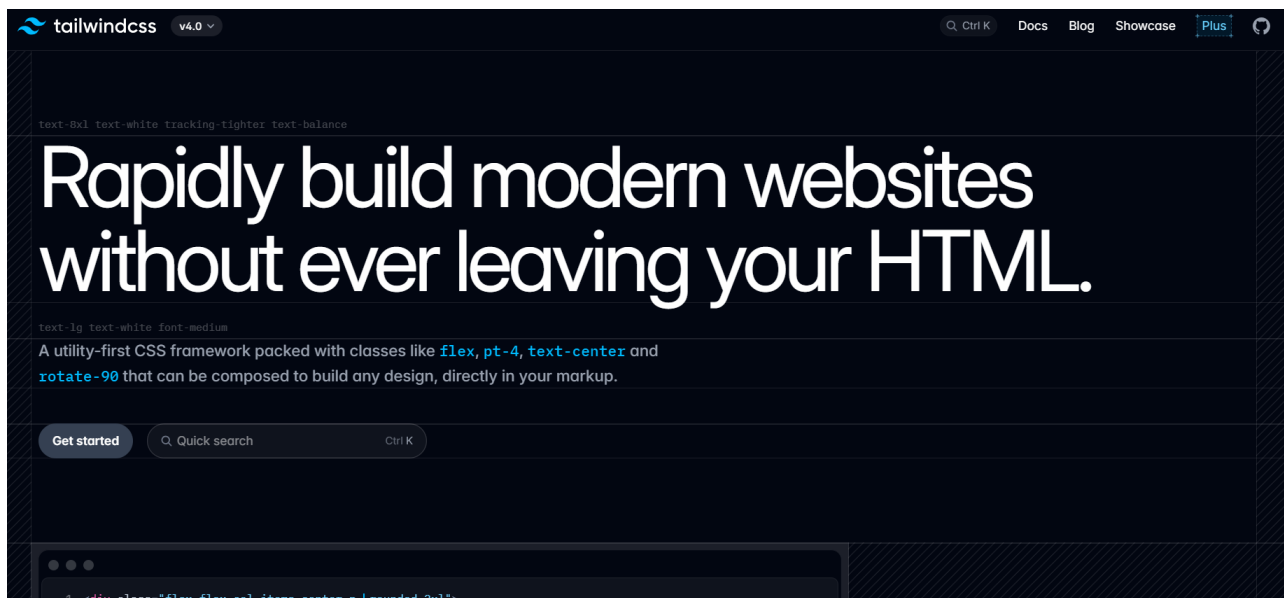


Рис 2.4. – Tailwindcss

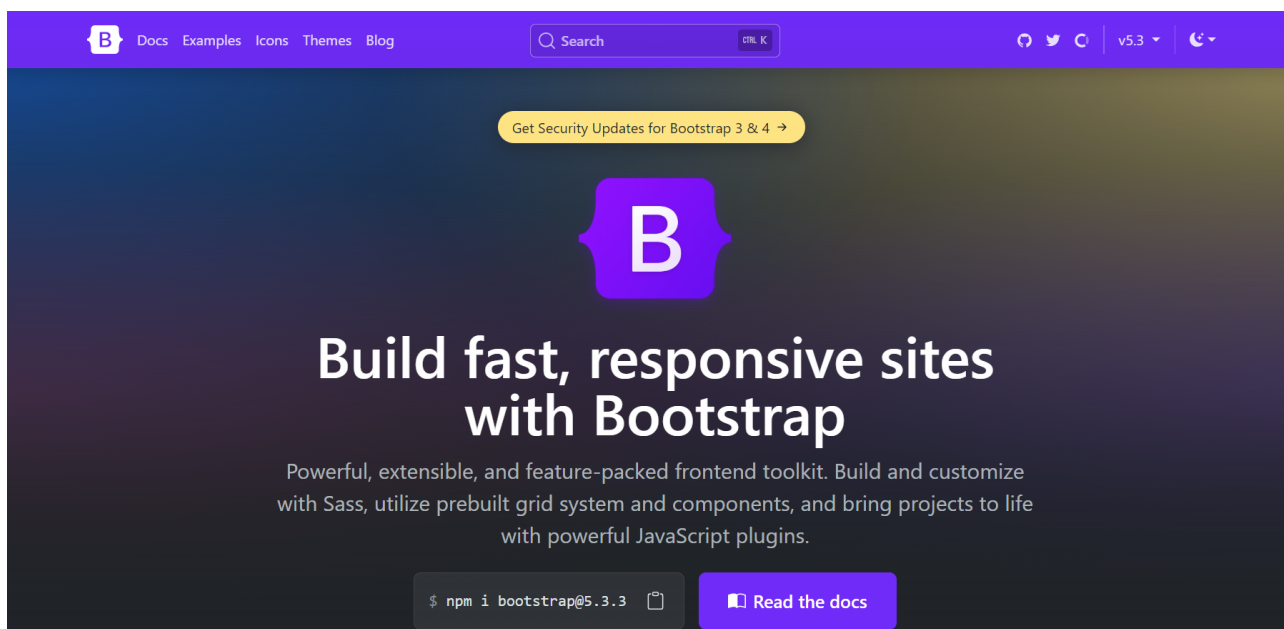


Рис 2.5. – Bootstrap

Для бекенду, що відповідає за логіку роботи системи та обробку даних, можна використовувати Node.js(рис 2.6.) разом з фреймворком Express.js(рис

2.7.). Node.js є одним з найбільш популярних середовищ для серверного програмування, оскільки дозволяє використовувати одну мову — JavaScript — як для фронтенду, так і для бекенду. Це знижує складність розробки та полегшує підтримку коду. Express.js надає мінімалістичний і гнучкий фреймворк для побудови серверних додатків, що дозволяє швидко реалізовувати різноманітні функції, такі як аутентифікація користувачів, збереження прогресу, робота з базами даних та інтеграція з іншими сервісами.

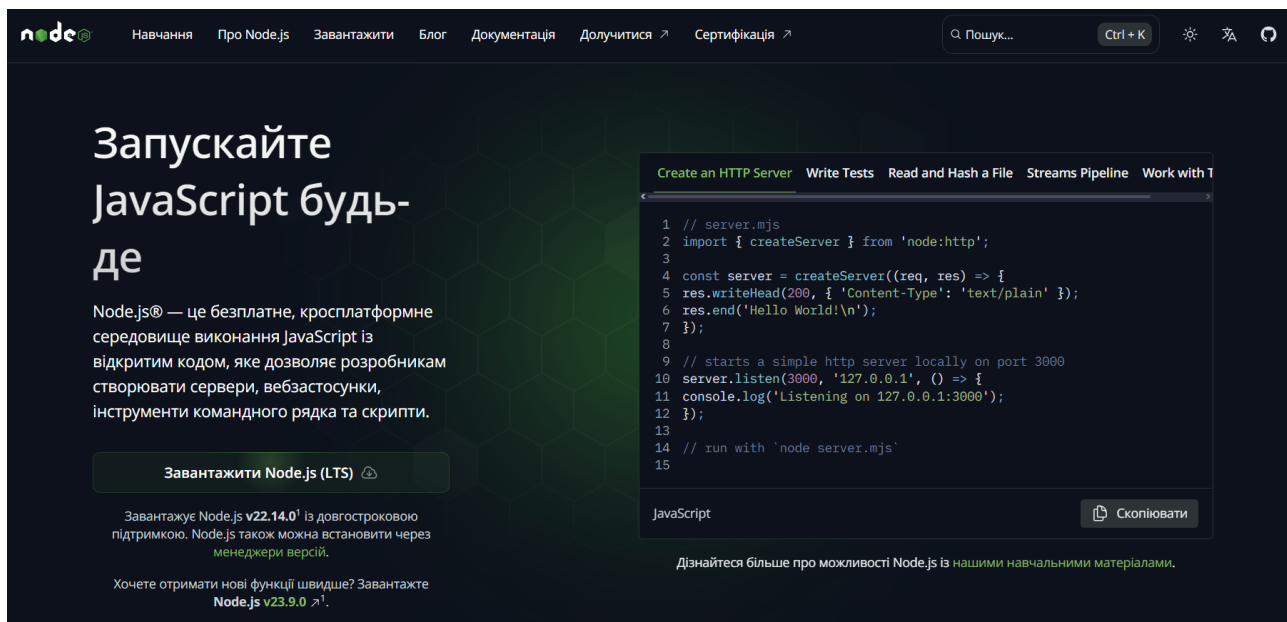


Рис 2.6. – Node.js

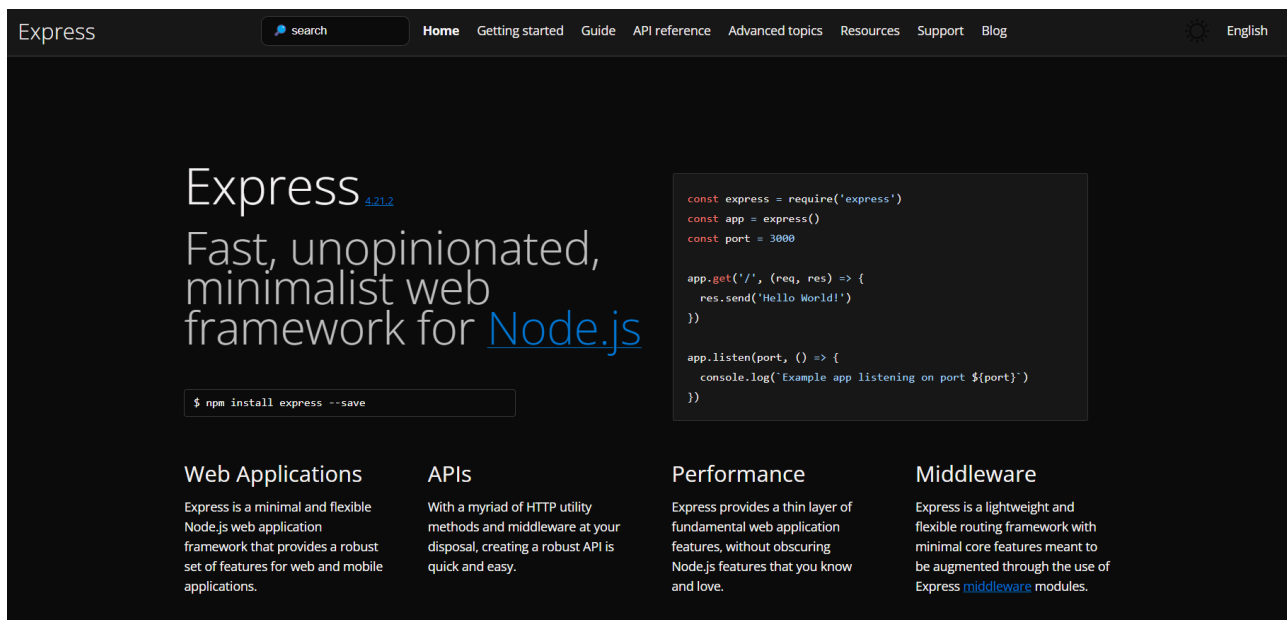


Рис 2.7. – Express.js

Щодо бази даних, оптимальним вибором може бути реляційна база даних, така як PostgreSQL(рис 2.8.), або нереляційна база даних, наприклад, MongoDB(рис 2.9.). PostgreSQL є потужною, надійною і високо масштабованою базою даних, яка підтримує складні запити та транзакції. Це важливо, якщо платформа має обробляти велику кількість користувачів і зберігати їхні індивідуальні дані, такі як результати виконаних завдань або прогрес навчання. MongoDB, у свою чергу, підходить для зберігання даних у гнучкому форматі JSON, що може бути корисним для систем із динамічними та нестандартними даними. Обидві бази даних добре масштабуються і підтримують різні інструменти для роботи з великими обсягами даних.



Рис 2.8. – PostgreSQL

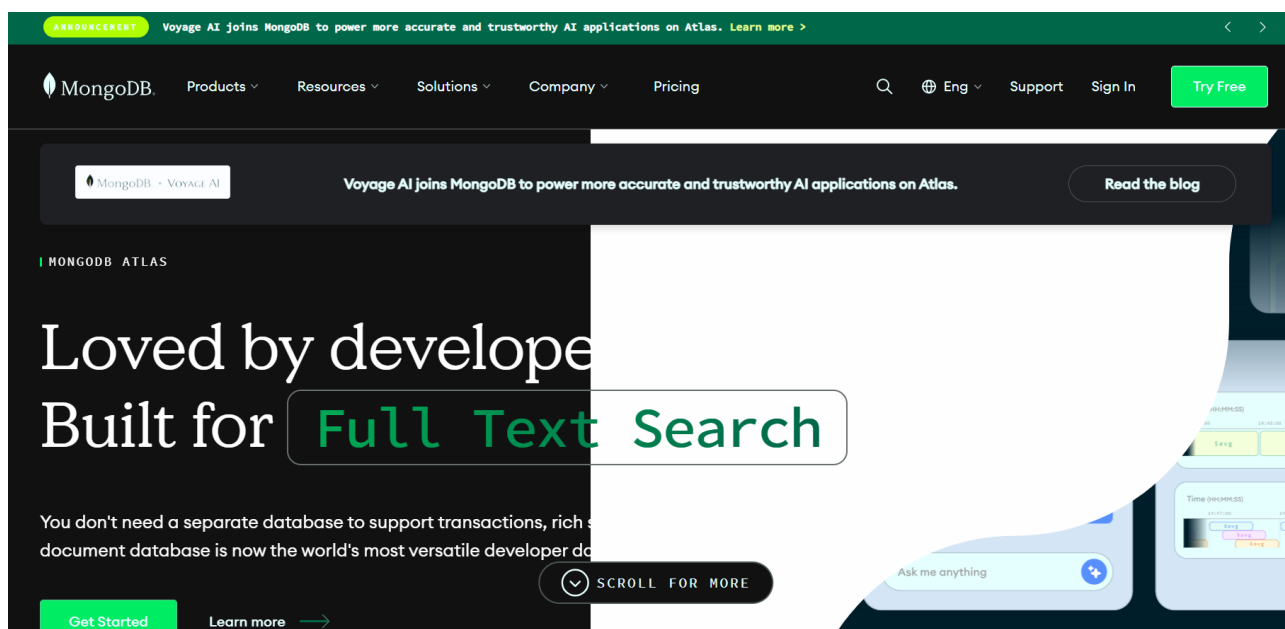


Рис 2.9. – MongoDB

Іншою важливою частиною розробки платформи є інтеграція з хмарними сервісами, що дозволяє забезпечити зберігання даних, резервні копії та масштабованість платформи без необхідності обслуговування власних серверів. Такі сервіси, як Amazon Web Services (AWS)(рис 2.10.) або Google Cloud(рис 2.11.), надають можливість хостингу, автоматичного масштабування та забезпечення безпеки даних. Наприклад, Amazon S3 може бути використаний для зберігання медіа-контенту, а Amazon RDS — для управління базами даних. Також хмарні сервіси дозволяють легко додавати нові функції, такі як машинне навчання або аналітика, які можуть бути корисними для вдосконалення системи у майбутньому.

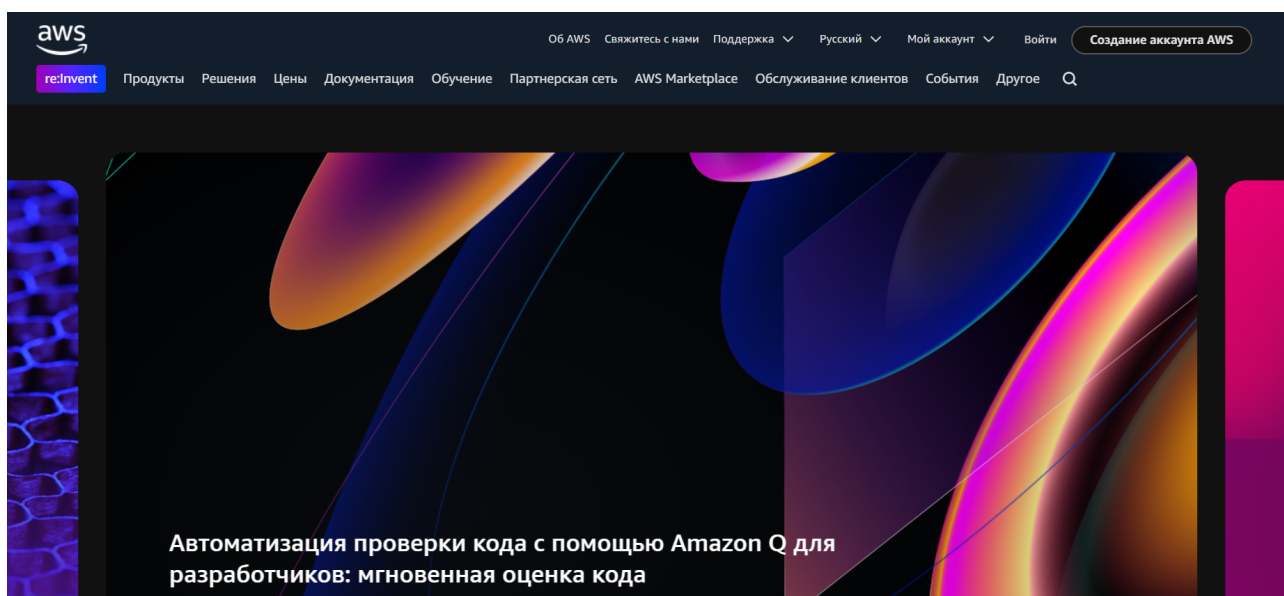


Рис 2.10. – Amazon Web Services

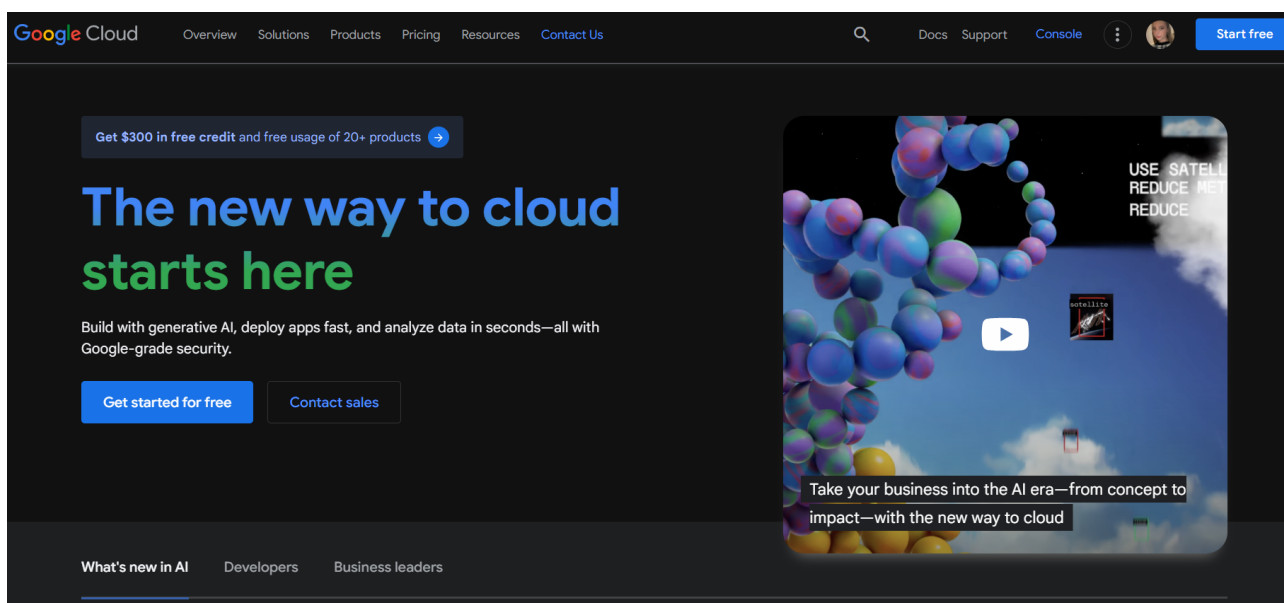


Рис 2.11. – Google Cloud

Для забезпечення високого рівня безпеки платформи варто використовувати протоколи шифрування даних, такі як HTTPS і SSL/TLS. Аутентифікація користувачів може бути реалізована за допомогою OAuth або JWT (JSON Web Tokens), що забезпечує безпечний доступ до платформи і захист персональних даних. Безпека є критичним аспектом, оскільки платформа працюватиме з даними користувачів, зокрема їхньою персональною інформацією та прогресом у навчанні.

Також важливо забезпечити підтримку багатомовності, оскільки платформа буде використовуватися користувачами з різних країн. Для цього можна використовувати бібліотеки для локалізації, такі як i18next, що дозволяє легко додавати переклади та підтримувати кілька мов у рамках одного інтерфейсу.

Окрім технічної сторони, вибір інструментів для забезпечення якісного користувацького досвіду також включає інструменти для збирання та аналізу даних про поведінку користувачів. Використання аналітичних інструментів, таких як Google Analytics(рис 2.12.) або Mixpanel(рис 2.13.), дозволяє відстежувати, як користувачі взаємодіють із платформою, які завдання викликають у них труднощі і як можна покращити інтерфейс для підвищення ефективності навчання.

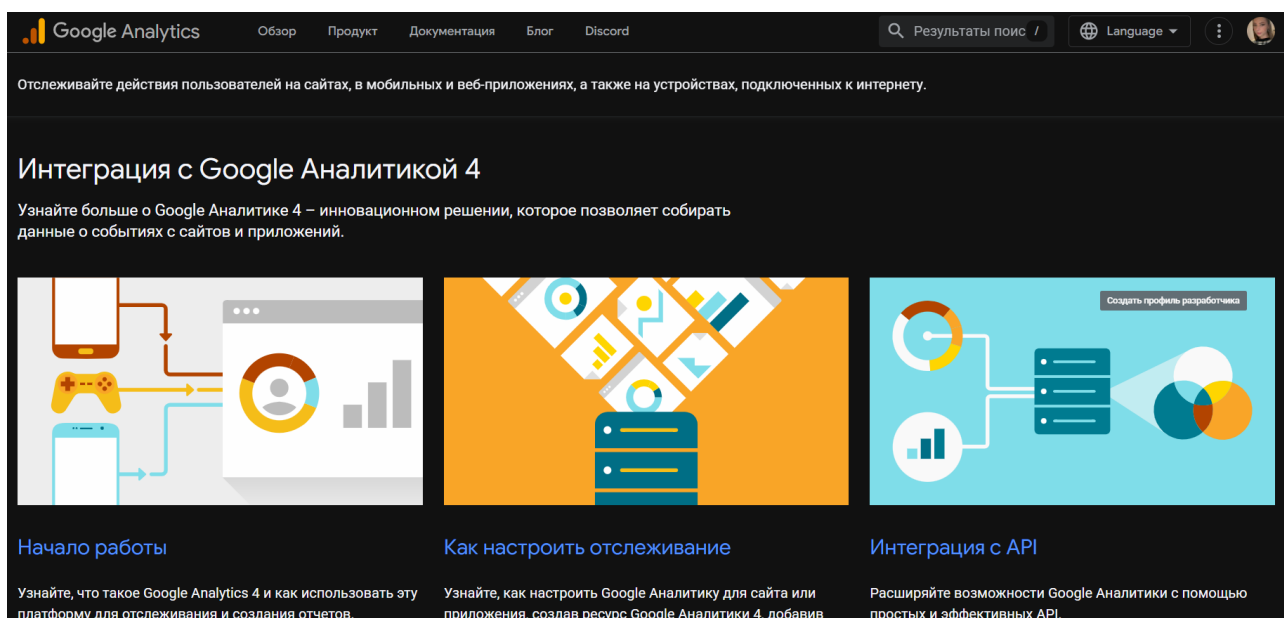


Рис 2.12. – Google Analytics

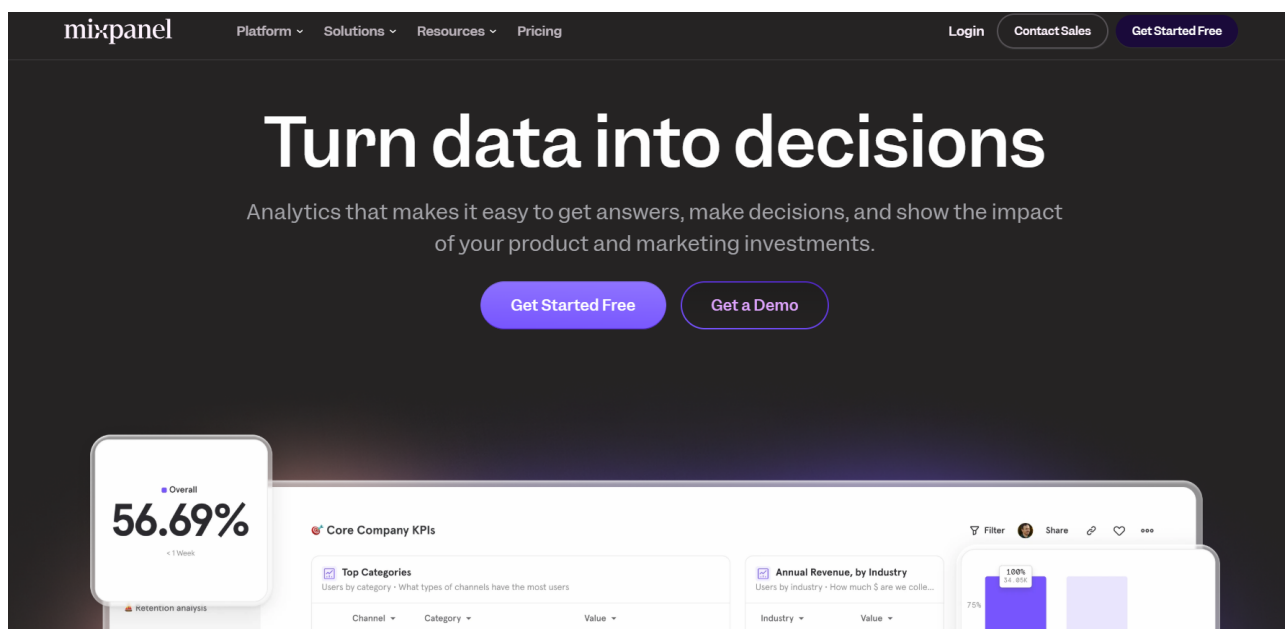


Рис 2.13. – Mixpanel

Таким чином, вибір інструментів та технологій для розробки платформи для вивчення мов програмування є багатогранним процесом, що потребує врахування різних аспектів, таких як інтерактивність, масштабованість, безпека та підтримка користувачів на різних пристроях.

Для реалізації платформи для вивчення програмування ми обрали використання сучасних технологій, які забезпечать швидкість розробки, ефективність та гнучкість у створенні інтерфейсу і функціоналу. Одним з основних інструментів, який буде використовуватися для реалізації, є сайт Vite(рис 2.14.), що надає зручне середовище для роботи з JavaScript-проектами. Vite відомий своєю швидкістю завдяки інноваційним підходам до бандлінгу й оптимізації коду, що дозволить створити продуктивну і швидку веб-платформу.

Основним фреймворком для розробки користувацького інтерфейсу буде React. Це популярний JavaScript-фреймворк, який дозволяє створювати динамічні веб-застосунки, забезпечуючи високий рівень взаємодії з користувачем. React дозволяє розробляти компоненти інтерфейсу як окремі модулі, що робить код більш структурованим і легким для підтримки. Це є важливою перевагою в контексті розвитку платформи, де можливість легко додавати нові функції та оновлювати контент є ключовою вимогою.

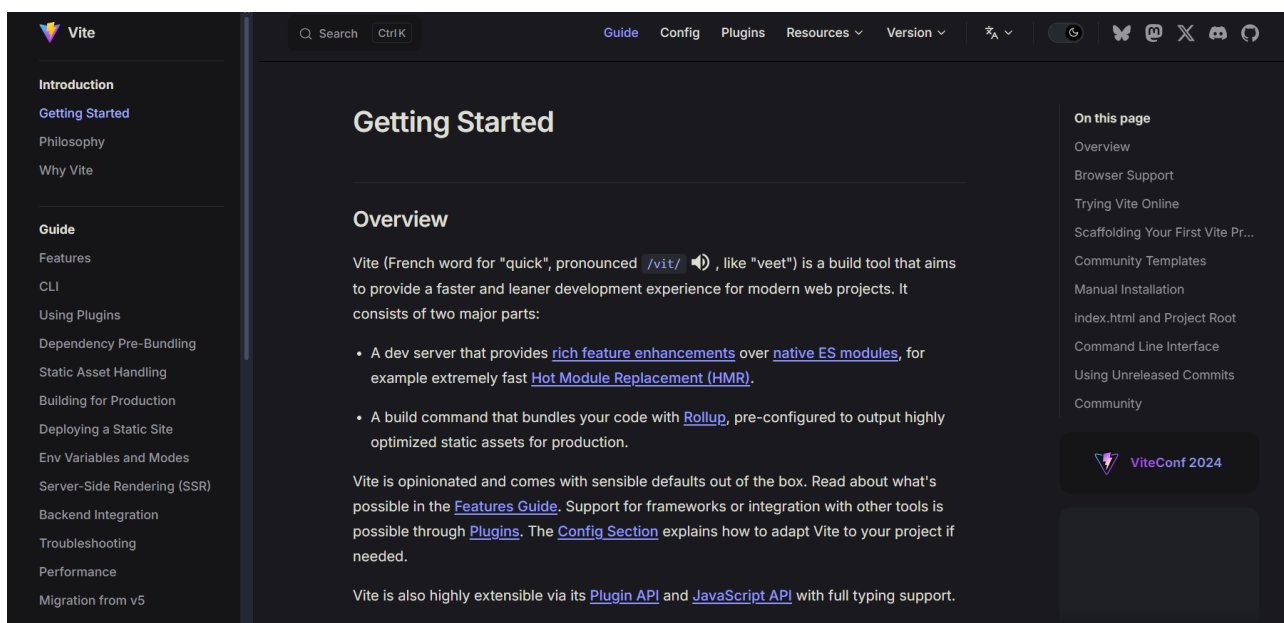


Рис 2.14. – Vite

Окрім цього, на цьому ж сайті ми будемо розробляти стилі за допомогою CSS. Це забезпечить відповідність сучасним стандартам веб-дизайну та дозволить створити інтуїтивно зрозумілий і привабливий інтерфейс для користувачів різного віку та рівня підготовки.

2.2. Опис функціональних модулів системи

Основним елементом системи є модуль користувацького інтерфейсу, який відповідає за зручну та інтуїтивно зрозумілу взаємодію користувача з платформою. Інтерфейс системи спроектований таким чином, щоб користувач міг легко знаходити потрібні функції, перемикатися між режимами навчання і гри, обирати мови програмування, виконувати завдання та отримувати зворотний зв'язок. Модуль інтерфейсу підтримує інтерактивність, забезпечуючи миттєву реакцію на дії користувача, що дозволяє створити динамічне навчальне середовище. Важливим аспектом цього модуля є адаптивність інтерфейсу, яка забезпечує коректне відображення елементів на різних пристроях, таких як комп'ютери, планшети та смартфони.

Одним із ключових функціональних модулів платформи є модуль вибору та виконання завдань. Він відповідає за надання користувачам завдань з програмування, а також обробку їхніх відповідей. Цей модуль інтегрується з базою даних завдань, яка містить завдання різного рівня складності та різних типів — від простих задач на синтаксис до складних алгоритмічних викликів. Після вибору завдання користувачем, система динамічно підвантажує відповідне середовище для написання коду. Цей модуль підтримує кілька мов програмування, таких як JavaScript, Python, Java, C++, що дозволяє користувачам обирати мову, з якою вони бажають працювати. Платформа також пропонує можливість тестування програмного коду в реальному часі, дозволяючи одразу отримати зворотний зв'язок щодо його коректності. Кожне завдання має чіткі інструкції, а користувачі можуть вводити свій код або вибирати правильні відповіді прямо на платформі. Після завершення завдання модуль автоматично перевіряє результати та надає миттєвий зворотний зв'язок.

Модуль перевірки відповідає за аналіз введених користувачем відповідей та їхнє порівняння з правильними варіантами. Для тестових завдань це може бути автоматична перевірка правильності вибраних відповідей, а для програмних завдань — аналіз написаного коду та його відповідності умовам завдання. Результати перевірки відображаються в окремому полі з можливими коментарями щодо помилок чи підказками для подальшого виконання.

Модуль зворотного зв'язку є важливою частиною платформи, оскільки забезпечує комунікацію між користувачем і системою щодо виконаних завдань. Він аналізує правильність виконаних завдань, перевіряючи як синтаксис, так і логіку роботи коду. Після завершення завдання або модуля користувачі можуть залишати відгуки та отримувати поради, що підвищує залученість у процес навчання. Зворотний зв'язок може бути також наданий у вигляді автоматичних повідомлень після виконання завдання. Користувач отримує детальні повідомлення про помилки або успішне виконання завдання, а також рекомендації для подальшого вдосконалення. Завдяки цьому модулю процес навчання стає більш продуктивним і зосередженим на самостійній роботі, адже

користувач отримує миттєвий зворотний зв'язок, що дозволяє одразу виправляти помилки та поглиблювати свої знання.

Ще одним важливим компонентом платформи є модуль управління профілем користувача. Він дозволяє кожному користувачу створити власний акаунт, де зберігається інформація про виконані завдання, досягнутий прогрес і отримані бали. Цей модуль також відповідає за інтеграцію соціальних елементів у систему, таких як можливість порівняння своїх результатів з результатами інших користувачів, отримання нагород за виконання певних рівнів або завдань, а також підтримку взаємодії між користувачами у вигляді коментарів або запитань. Він відстежує кількість виконаних завдань, збережені результати та досягнуті бали. Це дозволяє користувачам легко переглядати свої досягнення, а також отримувати загальну оцінку їх успіхів у навчанні. Профіль включає також можливість перегляду детальної статистики, що підвищує мотивацію та допомагає користувачам планувати свій навчальний процес. Важливою особливістю цього модуля є його безпека та конфіденційність, що забезпечує захист особистих даних користувачів і їх прогресу в навчанні.

Модуль адміністративної панелі дозволяє адміністраторам платформи керувати контентом, зокрема додаванням нових завдань, оновленням матеріалів, а також налаштуванням певних параметрів системи. Адміністратори можуть стежити за активністю користувачів, модифікувати базу завдань, а також отримувати аналітичні дані про використання платформи. Цей модуль спрямований на забезпечення безперервної підтримки та оновлення системи, що дозволяє їй адаптуватися до потреб користувачів і відповідати сучасним вимогам у сфері освіти.

На платформі також реалізовано модуль аналітики, який збирає та аналізує дані про взаємодію користувачів із системою. Завдяки цьому модулю можна оцінювати, які завдання викликають найбільше труднощів, які мови програмування користуються найбільшою популярністю, а також як швидко користувачі опановують нові теми. Така аналітика дозволяє розробникам

платформи вдосконалювати її роботу, додавати нові функції та завдання, а також підлаштовувати контент під потреби користувачів.

Модуль оцінювання прогресу користувача включає індикатори рівня досягнень, які відображають поточний рівень успішності. Наприклад, після завершення певної кількості завдань користувачі можуть отримувати бали чи досягнення, які відображаються у їхньому профілі. Це допомагає відслідковувати успіхи в навчанні та мотивує до подальшого виконання завдань.

Також важливим є модуль інтерактивної гри, який надає користувачам можливість вибору між навчанням у режимі гри та стандартного вивчення матеріалу. Цей модуль дозволяє додати гейміфікацію у процес навчання, перетворюючи навчальні завдання на захоплюючі челенджі з оцінюванням результатів у балах чи нагородах.

Таким чином, система навчання програмування побудована на основі кількох функціональних модулів, які тісно взаємодіють між собою для забезпечення ефективного процесу навчання. Кожен модуль виконує свої специфічні функції, що разом створює комплексне рішення для користувачів різного рівня знань та підготовки.

2.3. Проектування інтерфейсу користувача

Розділ "Проектування інтерфейсу користувача" є важливим етапом у створенні платформи для вивчення мов програмування, оскільки саме через інтерфейс користувачі взаємодіють із системою. Основною метою цього процесу є розробка інтуїтивно зрозумілого, зручного та ефективного інтерфейсу, який допоможе користувачам легко освоювати нові матеріали, виконувати завдання та отримувати зворотний зв'язок.

На першому етапі проектування важливо визначити структуру та загальну навігацію інтерфейсу. Інтерфейс платформи повинен бути організований так, щоб користувачі могли швидко знайти потрібні функції та

виконати необхідні дії. Головне меню платформи включатиме основні розділи, такі як вибір мови програмування, доступ до завдань, перегляд прогресу, а також можливість налаштування профілю користувача. При цьому навігація повинна бути простою та логічною, що допоможе новим користувачам швидко зрозуміти, як користуватися платформою.

Інтерфейс для вибору мови програмування є важливим елементом системи, оскільки користувачі можуть вивчати різні мови, такі як JavaScript, Python, Java чи C++. На екрані вибору мови слід відобразити іконки чи назви мов програмування, щоб користувачі могли зручно вибрати потрібну мову. Важливо також забезпечити зворотний зв'язок після вибору мови, щоб користувачі бачили, що їх вибір успішно збережений.

Основна частина інтерфейсу платформи – це робоча зона для виконання завдань. Ця зона повинна бути розроблена таким чином, щоб користувачі мали доступ до тексту завдання та могли безпосередньо писати код у зручному редакторі. Інтерфейс повинен підтримувати можливість миттєвого виконання та перевірки коду, щоб користувачі могли швидко побачити результат своєї роботи. Окрім того, для полегшення роботи з кодом редактор повинен мати функції підсвічування синтаксису, автоматичного завершення команд та форматування коду.

Важливою складовою є система зворотного зв'язку, яка повинна бути інтегрована в інтерфейс таким чином, щоб користувачі миттєво отримували інформацію про результати виконання завдань. Зворотний зв'язок повинен бути чітким і зрозумілим, а повідомлення про помилки мають містити підказки щодо того, як виправити код або що було зроблено неправильно. Це допоможе користувачам навчатися через виправлення своїх помилок і покращувати свої навички програмування.

Інтерфейс платформи також передбачає реалізацію профілю користувача, де будуть відображатися персональні досягнення, виконані завдання, рівень прогресу та набрані бали. Ця інформація допоможе мотивувати користувачів продовжувати навчання і досягати нових рівнів. Профіль користувача також

включатиме налаштування, де можна буде змінювати персональні дані, обирати рівень складності завдань або налаштовувати інтерфейс відповідно до власних потреб.

Окрім робочої зони та профілю користувача, інтерфейс платформи включатиме модуль відгуків та рейтингу, де користувачі зможуть залишати відгуки про завдання або модулі, а також спілкуватися з іншими учасниками. Це сприятиме розвитку спільноти навколо платформи, що є важливим аспектом для підтримки мотивації користувачів.

Дизайн інтерфейсу повинен бути приємним на вигляд та естетично привабливим. Використання сучасних елементів дизайну, таких як м'які кольорові схеми, чіткі шрифти та зручні кнопки, допоможе створити позитивний користувацький досвід. При цьому важливо дотримуватися принципів мінімалізму, щоб уникнути перевантаження інтерфейсу зайвими елементами. Особливу увагу потрібно приділити адаптивності інтерфейсу, щоб платформа однаково добре працювала на різних пристроях — від настільних комп'ютерів до смартфонів.

Таким чином, проектування інтерфейсу користувача базується на забезпеченні зручності, інтерактивності та естетичної привабливості. Кожен елемент інтерфейсу, від вибору мови програмування до отримання зворотного зв'язку, повинен бути логічно спроектований та зручно реалізований, щоб користувачі могли ефективно навчатися програмуванню та розвивати свої навички.

Дизайн інтерфейсу навчальної платформи(рис 2.15.), (рис 2.16.), (рис 2.17.), представлений у цьому проекті, є прикладом сучасного підходу до UI/UX дизайну, що фокусується на простоті, функціональності та зручності для користувача. Основна концепція дизайну базується на мінімалістичному підході, де основний акцент робиться на зміст та простоту взаємодії. Користувачеві надається можливість зосередитися на виконанні завдань без зайвих елементів, що могли б відволікати його від основної мети.

ЗАНУРЮЙСЯ У ПРОГРАМУВАННЯ З CODEFLOW!

Про нас

CodeFlow — це онлайн-платформа, яка допомагає вам вивчати мови програмування через інтерактивні завдання та тести. Наше завдання — допомогти вам освоїти основи програмування, незалежно від вашого рівня підготовки. Кожен урок розроблений таким чином, щоб ви могли не лише вивчати синтаксис мови, але й розв'язувати реальні проблеми.

Ми віримо, що навчання програмуванню повинно бути цікавим, доступним та ефективним. Тому ми створили CodeFlow, щоб кожен міг покращити свої навички та стати впевненим програмістом.



Урок 1: Вступ до JavaScript

Рис 2.15. – Прототип дизайну

Оберіть мову програмування:

JavaScript

Python

Java

Що про нас кажуть наші користувачі



Олександр, початківець

"CodeFlow — це чудовий ресурс для навчання програмуванню! Завдяки інтерактивним завданням я швидко освоїв нові мови."



Ірина, студентка

"Завдання дуже цікаві, і кожен урок допомагає покращити мої навички. Платформа дуже зручна!"



Андрій, програміст

"CodeFlow допоміг мені вивчити JavaScript за рекордно короткий час. Чудова платформа для самостійного навчання!"



Марія, розробниця

"Мені дуже подобається, що тут можна не тільки вивчати синтаксис, але й розв'язувати реальні задачі!"

Рис 2.16. – Прототип дизайну

Модуль 1



Instagram Facebook TikTok

Адреса: вул. Програмістів 15, Київ, Україна
Телефон: +38 050 123 45 67

Рис 2.17. – Прототип дизайну

Фон інтерфейсу виконаний у світлих, переважно білих тонах, що створює відчуття простору та чистоти. Білий фон є нейтральним і дозволяє основним елементам, таким як кнопки, поля для введення тексту та інші інтерфейсні компоненти, чітко виділятися. Це сприяє легкому сприйняттю інформації та забезпечує зрозумілу візуальну ієрархію, де користувач без зусиль може орієнтуватися між розділами та блоками контенту.

Навігаційна панель розташована у верхній частині екрана, що є звичним для більшості користувачів і відповідає принципам зручної навігації. Панель дозволяє легко перемикатися між ключовими розділами платформи, такими як "Про нас", "Завдання", "Рецензії" та інші. Такий лаконічний підхід до структури навігації забезпечує швидкий доступ до основної інформації та дозволяє користувачам виконувати переходи без зайвих зусиль. Важливі елементи, такі як кнопки вибору мови програмування або режиму навчання (гра чи вивчення), добре помітні та розташовані у верхній частині, що робить їх легкодоступними.

Центральне місце в інтерфейсі відведено безпосередньо навчальним завданням. Завдання можуть бути представлені у вигляді поля для введення коду або вибору відповідей на тестові запитання. Візуальна організація цих блоків спрямована на забезпечення користувачам максимальної зручності при виконанні завдань. Завдання подаються у структурованій формі, з чіткими інструкціями, які розташовані поруч або над полем для відповіді. Таке розташування дозволяє користувачам швидко ознайомитися з вимогами та негайно приступити до їх виконання.

Особливу увагу приділено інтуїтивній структурі сторінок та взаємодії з користувачем. Наприклад, після вибору певної мови програмування чи режиму гри, користувач автоматично перенаправляється до відповідного розділу з завданнями. Ця функція автоматичного прокручування або перемикання забезпечує плавність користувацького досвіду та мінімізує потребу в додаткових діях з боку користувача. Сторінки також мають чітку візуальну роздільність між текстовими блоками, кнопками та іншими інтерактивними елементами, що дозволяє уникнути перевантаження інформацією.

Дизайн враховує різні сценарії використання платформи. Наприклад, поля для введення тексту або коду достатньо великі, що сприяє зручності введення даних, особливо під час виконання завдань з програмування. Всі елементи взаємодії, такі як кнопки підтвердження, скасування або вибору відповіді, розташовані логічно та мають відповідні розміри для зручної взаємодії.

Дизайн прогресу та результатів користувачів реалізований через використання графічних індикаторів та візуалізації досягнень. Наприклад, у вкладці "Прогрес" користувач може побачити свої досягнення, представлені у вигляді балів або графічних шкал, що дозволяють швидко оцінити результати без потреби заглиблюватися в додаткові деталі. Такий підхід полегшує користувачеві стеження за своїм навчанням і мотивує до подальшого виконання завдань.

Ще однією важливою рисою цього дизайну є його адаптивність. Інтерфейс виглядає однаково добре як на настільних комп'ютерах, так і на мобільних пристроях. Це досягається завдяки гнучкому підходу до розміщення елементів, використанню адаптивних шрифтів та кнопок, які змінюють свої розміри та положення в залежності від розміру екрана. Таким чином, користувач може зручно працювати з платформою на різних пристроях, що підвищує її доступність.

Загалом, цей UI/UX дизайн забезпечує високий рівень зручності та функціональності, зосереджуючи увагу користувачів на основних завданнях і мінімізуючи всі можливі відволікання. Це створює позитивний користувацький досвід та сприяє ефективному засвоєнню навчального матеріалу, що є ключовою метою навчальної платформи.

Інтерфейс користувача для вашого проекту побудований таким чином, що надає зручний та інтерактивний спосіб взаємодії з платформою навчання програмуванню. Користувач починає з вибору мови програмування, що відображається через компонент `LanguageSelection`. Тут є три кнопки для вибору мови: JavaScript, Python і Java(рис 2.18.). Після того, як користувач

вибирає мову, на екрані з'являється завдання, яке відповідає обраній мові. Всі ці дії керуються станом за допомогою хука `useState`, що дозволяє змінювати вміст на екрані відповідно до вибору користувача.

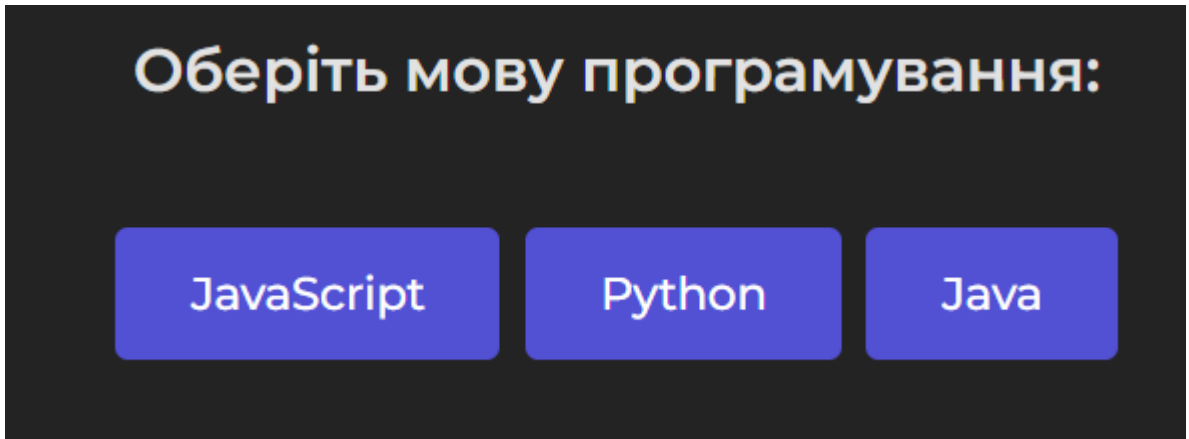


Рис 2.18. – Вибір мови програмування

Після вибору мови користувач бачить завдання, яке складається з інструкцій для програмування. Для цього використовується компонент `CodeInput`(рис 2.19.), де користувач може ввести свій код. Залежно від мови програмування, завдання буде змінюватися. Наприклад, для JavaScript завданням може бути команда для виведення повідомлень у консоль, для Python — підняття руки робота, а для Java — виконання певних дій з роботом. Компонент `CodeInput` містить текстове поле для введення коду, кнопку для його виконання, а також відображає зворотний зв'язок після виконання, що повідомляє про успішне виконання або помилку.

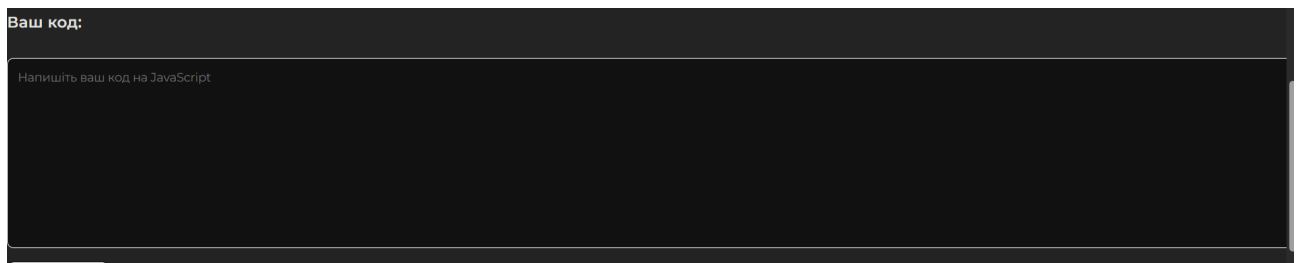


Рис 2.19. – `CodeInput`

Крім того, є інтерактивний робот(рис 2.20.), який відображається через компонент Robot. Цей робот змінює свою позу в залежності від дій користувача в кодї. Кожна дія робота, така як рух вперед, підняття руки чи присідання, прив'язана до певних команд у кодї користувача. Замість того, щоб просто виводити текст, робот у візуальному вигляді демонструє результат виконаного коду. Це додає елемент гейміфікації та робить процес навчання більш цікавим.



Рис.2.20 – Робот

Також є підсистема зворотного зв'язку(рис 2.21.), яка надає підказки у випадку помилок та допомагає користувачам виправити їх, що робить процес навчання більш ефективним. Окремо є кнопка для виходу з завдання, що дозволяє користувачам повернутися до початкового екрану.

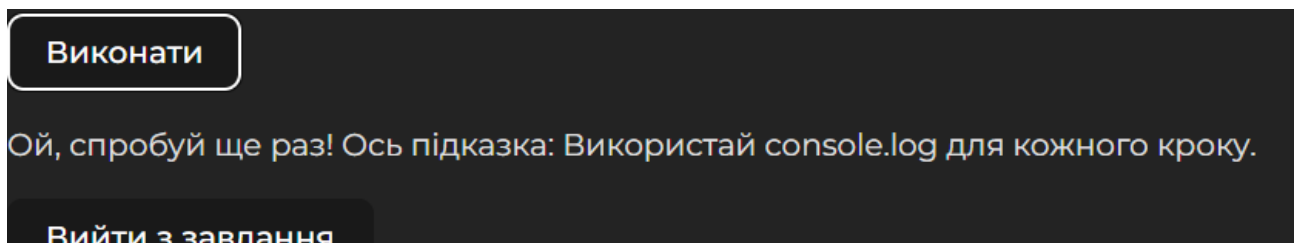


Рис 2.21. – Система зворотного зв'язку

У загальному, інтерфейс користувача побудований навколо інтерактивного навчального процесу, де кожна дія відображається на екрані у вигляді відгуку робота, а також є можливість виправляти помилки та отримувати зворотний зв'язок, що підвищує зацікавленість користувача та покращує навчальний досвід.

2.4. Розробка системи прогресу та мотивації

Розробка системи прогресу та мотивації є однією з ключових складових освітньої платформи, оскільки саме вона впливає на залученість користувачів та сприяє підтриманню їхнього інтересу до навчання. Основна мета цієї системи – створити позитивну динаміку навчання, де користувачі бачать свої досягнення, отримують миттєвий зворотний зв'язок та мають чітке розуміння своїх успіхів, що мотивує їх продовжувати навчання.

Система прогресу ґрунтується на кількох важливих аспектах. Перш за все, кожен користувач має власний профіль, у якому зберігаються всі його досягнення: кількість виконаних завдань, результати тестів, отримані бали та інші показники успіху. Це забезпечує персоналізований підхід, оскільки користувачі можуть легко відстежувати свій індивідуальний прогрес. Візуалізація досягнень представлена у вигляді графіків або індикаторів рівня, що підвищує прозорість системи та робить навчальний процес більш структурованим. Користувачі можуть бачити свій прогрес не тільки у кількісних показниках, але й у вигляді спеціальних значків або нагород за досягнення певних етапів.

Механізми мотивації включають гейміфікацію, що додає навчальному процесу елемент гри. Платформа передбачає можливість проходження завдань у різних режимах, зокрема в ігровій формі, де користувачі можуть змагатися за бали, проходити рівні або виконувати завдання на час. Такий підхід дозволяє додати елемент виклику та змагання, що підвищує рівень залученості та інтерес

до завдань. Система також передбачає бонуси або нагороди за виконання завдань на високому рівні, що слугує додатковим стимулом для користувачів.

Зворотний зв'язок також відіграє важливу роль у мотиваційній складовій платформи. Після кожного завдання користувач отримує детальні результати своєї роботи з аналізом помилок або успіхів. Це дозволяє не тільки отримувати миттєве підтвердження своїх досягнень, але й розуміти, у яких аспектах необхідно покращитися. Додатково система може надавати рекомендації щодо покращення результатів або пропонувати додаткові завдання для вдосконалення навичок.

Ще одним важливим компонентом є система рівнів або ступенів складності, яка поступово збільшує складність завдань у міру прогресу користувача. Це допомагає уникнути перенасичення простими завданнями та забезпечує відчуття зростання викликів. Користувачі починають із базових завдань і поступово рухаються до більш складних, що сприяє їх постійному розвитку та відчуттю прогресу.

Загалом, система прогресу та мотивації на платформі побудована таким чином, щоб користувачі постійно бачили свої досягнення, відчували підтримку і заохочення до подальшого навчання. Візуалізація результатів, нагороди за успіхи, гейміфікація та персоналізовані рекомендації створюють ефективну середу для навчання, де мотивація користувачів підвищується на кожному етапі. Це не лише сприяє залученню, але й формує у користувачів звичку до постійного самовдосконалення та розвитку своїх навичок.

Почнемо з того, що прогрес користувача відстежується через систему балів. У компоненті Game ви використовуєте змінну стану `score`(рис 2.22.), яка відображає поточні бали користувача. Кожен раз, коли користувач успішно виконує завдання, йому нараховуються бали. Це стимулює досягати результатів, оскільки користувач може бачити свій прогрес через зміни в рахунку, що є важливою частиною мотивації. Крім того, система нарахування балів реалізована таким чином, що кожен успішний результат, наприклад,

виконання правильного коду, збільшує бали, що забезпечує позитивний зворотний зв'язок та підвищує залученість користувача.

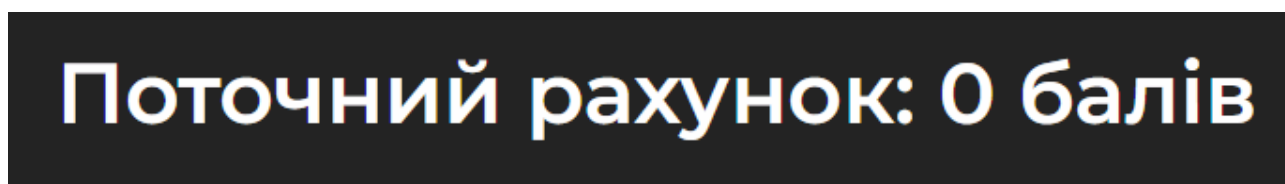


Рис 2.22. – Система балів

Мотивація також підтримується через зворотний зв'язок після виконання коду. Коли користувач правильно вирішує завдання, відображається повідомлення з похвалою: "Правильно! Молодець!"(рис 2.23.) Це є важливим моментом для підтримки мотивації, оскільки позитивні відгуки допомагають користувачу відчувати досягнення та бажання продовжувати вчитися. Водночас, коли відповідь неправильна, надається підказка, що дає користувачу шанс спробувати ще раз. Такий підхід допомагає створити атмосферу навчання через помилки, де неуспіх сприймається як частина процесу вдосконалення, а не як невдача.

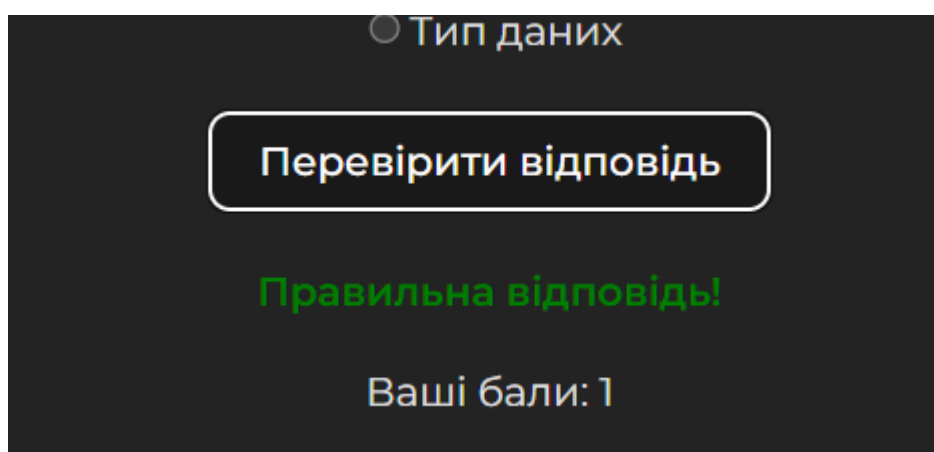


Рис 2.23. – Схвальне повідомлення

Також важливою частиною системи прогресу є роль завдань, які відрізняються за складністю в залежності від мови програмування. Кожне

завдання має свою специфіку і надає користувачу можливість застосовувати знання на практиці. У разі успішного виконання завдання, робот на екрані виконує певну дію, що є візуальним відображенням прогресу користувача. Наприклад, робот може зробити крок вперед, підняти руку чи пригнутися — кожна з цих дій є результатом правильно виконаного коду, що додає гейміфікаційний елемент до навчання, перетворюючи його на гру, де користувач має змогу побачити реальний результат своїх зусиль у вигляді анімацій робота.

Інтерактивність завдань, зворотний зв'язок і система балів працюють разом, щоб створити ефективну систему прогресу, яка надає користувачам відчуття досягнень і підтримує їх мотивацію в процесі навчання. Завдяки цим елементам користувачі не тільки отримують знання, але й бачать, як їхні зусилля відображаються на реальних результатах, що робить процес навчання більш захоплюючим і ефективним.

РОЗДІЛ 3

РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ НАВЧАЛЬНОЇ ГРИ

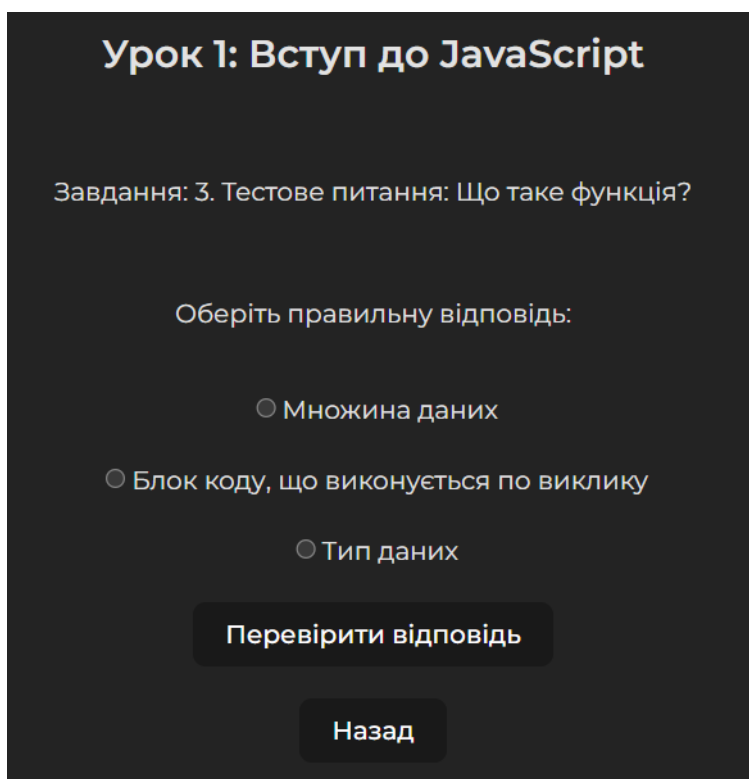
3.1. Реалізація базових функцій платформи

Реалізація базових функцій платформи відіграє ключову роль у забезпеченні інтерактивного досвіду навчання на платформі для вивчення програмування. Ці функції забезпечують можливість користувачам вибрати мову програмування, виконувати завдання різного типу, отримувати зворотній зв'язок щодо виконаних завдань та відслідковувати свій прогрес. Розглянемо кожен з основних функціональних елементів платформи.

Перше, що зустрічає користувача після входу на платформу, — це можливість вибору мови програмування. В даній реалізації користувачам пропонується обрати одну з трьох мов: JavaScript, Python або Java. Для цього на платформі створено окремий модуль вибору мови, який розроблений з використанням бібліотеки React та її хуків, зокрема `useState`. Коли користувач обирає мову, відповідний стан зберігається і використовується для генерації завдань та підбору специфічного середовища для написання коду. Вибір мови є інтерактивним і негайно відображається на інтерфейсі платформи, що дозволяє швидко перейти до виконання завдань на обраній мові.

Другий ключовий елемент платформи — це інтерактивні завдання. Кожне завдання може бути або кодовим, або у вигляді тесту(рис 3.1.) з вибором варіантів відповідей. Завдання розроблені так, щоб навчити користувача основам програмування на обраній мові. Для кожного завдання на платформі реалізовано механізм перевірки правильності виконання. У випадку кодових завдань, платформа приймає введений користувачем код і порівнює його з еталонним рішенням. Результати перевірки коду відображаються на екрані в реальному часі, що забезпечує інтерактивність навчання. Для тестових завдань

реалізована функція автоматичної перевірки вибору варіанту відповідей, яка також надає миттєвий зворотній зв'язок.



The image shows a dark-themed user interface for a JavaScript lesson. At the top, the title 'Урок 1: Вступ до JavaScript' is displayed in a light green font. Below it, the question 'Завдання: 3. Тестове питання: Що таке функція?' is shown in a light blue font. The instruction 'Оберіть правильну відповідь:' is in a light green font. There are three radio button options: 'Множина даних', 'Блок коду, що виконується по виклику', and 'Тип даних'. The second option is selected. At the bottom, there are two buttons: 'Перевірити відповідь' and 'Назад'.

Рис 3.1. – Тестове завдання

Зворотній зв'язок на платформі має вирішальне значення для покращення навчального процесу. Після кожного виконаного завдання користувач отримує повідомлення про успішність або необхідність коригування свого рішення. Цей механізм зворотного зв'язку також реалізований за допомогою React-хуків і забезпечує оновлення стану компонента, що відображає результати. Візуальні ефекти, такі як зміна кольору або поява спливаючих вікон з повідомленнями, додатково підсилюють ефективність взаємодії користувача з платформою.

Ще однією важливою функцією є відслідковування прогресу користувача. Платформа має модуль, який зберігає інформацію про виконані завдання, отримані оцінки та загальний бал користувача. Це дозволяє користувачам бачити свій прогрес у реальному часі і мотивує до подальшого проходження навчальних модулів. Прогрес зберігається локально під час поточної сесії і

відображається в інтерфейсі у вигляді графіків або лічильників виконаних завдань.

Для управління модальними вікнами(рис 3.2.), що відповідають за реєстрацію, завершення завдання чи перегляд отриманих результатів, на платформі реалізовано відповідні модулі. Ці модальні вікна створені для забезпечення плавної взаємодії користувача з платформою та управління ключовими подіями, такими як реєстрація нового користувача або отримання підтвердження після завершення завдання. Реєстрація користувача виконується через окрему форму, яка відображається у модальному вікні і включає поля для введення імені, електронної пошти та пароля. Після успішної реєстрації користувач може відразу приступити до навчального процесу.

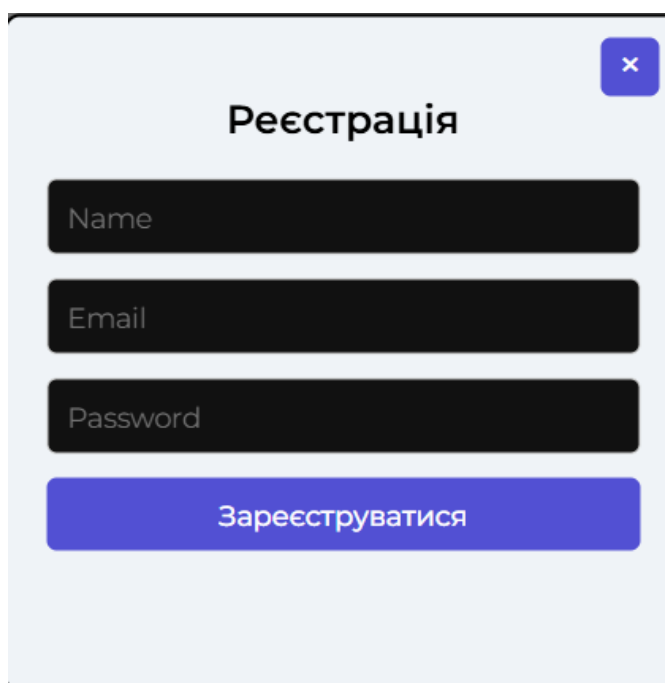
A screenshot of a modal registration window titled "Реєстрація" (Registration). The window has a light blue background and a dark blue close button in the top right corner. It contains three input fields: "Name", "Email", and "Password", each with a dark blue border and placeholder text. Below the input fields is a large blue button with the text "Зареєструватися" (Register).

Рис 3.2. – Модальне вікно реєстрації

Модальні вікна також використовуються для підтвердження завершення завдань і надання підсумкових результатів. Після успішного виконання завдання користувач отримує повідомлення про кількість набраних балів, час виконання завдання та пропозиції щодо подальших дій. Це дає можливість

швидко переглянути успіхи і перейти до наступного завдання або повторити попереднє у разі помилок.

Окрім базових функцій, таких як вибір мови, виконання завдань та зворотний зв'язок, платформа реалізує функції управління рівнем складності. Користувачі можуть самостійно обирати рівень завдань — від простих до складніших. Ця можливість надає гнучкість у навчальному процесі та дозволяє адаптуватися до індивідуального рівня підготовки користувача. Завдання з різних рівнів мають свої вимоги до написання коду або вибору правильних відповідей у тестах, що стимулює поступове ускладнення навчального матеріалу.

Також, на платформі реалізовано зручний інтерфейс для взаємодії з користувачем. Усі елементи інтерфейсу, включно з кнопками, полями введення, списками завдань і повідомленнями, створені з урахуванням принципів сучасного веб-дизайну. Для забезпечення адаптивності платформи і зручного користування нею на різних пристроях, був розроблений відповідний дизайн з використанням CSS та Tailwind, що дозволяє коректно відображати елементи на екранах різних розмірів, включно з мобільними пристроями.

Інтерфейс містить заголовки, списки завдань, панелі вибору мови програмування, кнопки для взаємодії з завданнями, а також області для відображення результатів. Всі ці елементи пов'язані між собою і використовують внутрішню логіку платформи для динамічного оновлення вмісту на основі дій користувача. Наприклад, після вибору мови програмування відбувається негайне завантаження відповідного списку завдань без необхідності перезавантаження сторінки.

Особливу увагу приділено реалізації функції автоматичного скролінгу до потрібного розділу сторінки після певних дій користувача. Зокрема, після вибору мови програмування або завершення завдання, інтерфейс автоматично прокручується до відповідної секції, щоб користувач міг без зайвих дій продовжувати виконувати наступні завдання або переглядати результати. Це робить взаємодію з платформою максимально інтуїтивною та зручною.

Таким чином, реалізація базових функцій платформи включає не лише стандартні компоненти вибору мови, виконання завдань та отримання зворотного зв'язку, але й забезпечує інтерактивний користувацький досвід завдяки використанню сучасних веб-технологій та інтерфейсних рішень. Усі ці функції покликані створити комфортне середовище для навчання програмуванню, де користувач може поступово розвивати свої навички і отримувати миттєвий зворотній зв'язок щодо своїх дій.

Крім основних функцій, що забезпечують вибір мови програмування, виконання завдань, отримання зворотного зв'язку та відслідковування прогресу, важливим аспектом роботи платформи є інтеграція механізмів персоналізації навчального процесу. Завдяки використанню технологій збереження даних, таких як локальне сховище браузера або серверна база даних, система може адаптувати навчальні модулі до індивідуальних потреб кожного користувача. Це означає, що платформа здатна аналізувати виконані завдання, оцінювати рівень підготовки і пропонувати наступні вправи відповідно до отриманих результатів. Таким чином, користувачі можуть розвивати свої навички в оптимальному темпі, без необхідності проходити занадто легкі або надто складні завдання.

Одним із ключових елементів персоналізації є система рекомендацій, яка допомагає користувачам орієнтуватися у навчальному процесі. Після завершення кожного блоку завдань платформа аналізує виконані вправи і пропонує наступні кроки, враховуючи сильні та слабкі сторони користувача. Наприклад, якщо система виявляє, що певні теми викликають труднощі, вона може запропонувати додаткові матеріали для вивчення або простіші завдання для закріплення базових навичок. Якщо ж навпаки, користувач впевнено виконує завдання певного рівня, платформа рекомендує складніші вправи, щоб стимулювати подальший розвиток.

Важливу роль у підвищенні ефективності навчання відіграє ігрова механіка, яка мотивує користувачів проходити більше завдань та вдосконалювати свої знання. Для цього на платформі реалізована система

нарахування балів за правильно виконані завдання, а також досягнення, які користувач отримує за певні успіхи, наприклад, виконання серії вправ без помилок або проходження всіх завдань у певному розділі. Візуалізація прогресу через графічні елементи, такі як шкала рівня, значки досягнень або рейтингові таблиці, сприяє підвищенню мотивації користувачів і заохочує їх до постійного вдосконалення навичок.

Окрім цього, інтеграція соціальних функцій дозволяє користувачам взаємодіяти між собою, що робить процес навчання більш захопливим та ефективним. Реалізація чатів, форумів або групових завдань сприяє створенню спільноти, де учасники можуть обговорювати проблеми, ділитися досвідом та допомагати один одному у вирішенні складних питань. Завдяки такій взаємодії навчання стає не лише індивідуальним, а й колективним процесом, що дає змогу глибше засвоювати матеріал та отримувати нові знання від інших учасників платформи.

Ще одним важливим аспектом є підтримка багатоплатформності, що дозволяє користувачам отримувати доступ до навчальних матеріалів з різних пристроїв. Завдяки адаптивному дизайну та використанню сучасних веб-технологій, платформа коректно відображається як на великих екранах, так і на мобільних пристроях. Це забезпечує гнучкість у навчальному процесі та дозволяє користувачам продовжувати навчання у зручній для них час та в будь-якому місці.

Окрім адаптивності, важливим елементом є підтримка офлайн-режиму, що дає змогу користувачам завантажувати навчальні матеріали та виконувати завдання без підключення до інтернету. Це особливо корисно для тих, хто має обмежений доступ до мережі або хоче використовувати платформу в умовах з нестабільним інтернет-з'єднанням. Усі результати виконаних завдань синхронізуються з сервером після відновлення підключення, що забезпечує безперервність навчального процесу.

Загалом, комплексний підхід до розробки платформи дозволяє створити ефективне навчальне середовище, яке поєднує індивідуальний підхід,

інтерактивність, ігрову мотивацію та соціальну взаємодію. Завдяки цьому користувачі не лише отримують теоретичні знання, але й активно застосовують їх на практиці, поступово підвищуючи свій рівень компетентності у програмуванні. Реалізація всіх цих функцій сприяє створенню інтуїтивної, доступної та ефективної платформи для навчання, яка відповідає сучасним вимогам освітніх технологій та допомагає користувачам розвивати свої навички у програмуванні максимально комфортним і продуктивним способом.

3.2. Інтеграція системи навчальних курсів для різних мов програмування

Інтеграція системи навчальних курсів для різних мов програмування є ключовою складовою нашої платформи, яка забезпечує можливість навчання широкого спектра користувачів. Одним із основних завдань під час реалізації цієї функції було створення універсальної системи, яка дозволяла б легко адаптувати курси для різних мов програмування та забезпечувала відповідний досвід для кожного користувача незалежно від обраної мови.

Наша платформа підтримує кілька популярних мов програмування, зокрема JavaScript, Python, Java та C++. Це дозволяє охопити як новачків, так і досвідченіших користувачів, надаючи їм можливість обрати ту мову, яка відповідає їхнім потребам або інтересам. Для кожної з цих мов було розроблено окремі набори завдань та навчальних матеріалів, які враховують особливості синтаксису та логіки програмування відповідної мови.

На початковому етапі реалізації інтеграції системи навчальних курсів ми сфокусувалися на забезпеченні правильного вибору користувачем мови програмування. Після вибору мови, інтерфейс платформи автоматично підлаштовується під вибір користувача, відображаючи завдання, які відповідають саме цій мові. Для цього було розроблено механізм динамічної зміни контенту на основі мови, що була обрана. Цей механізм включає

завантаження різних наборів завдань, кодових шаблонів та тестів залежно від мови програмування, яка була вибрана користувачем.

Кожен курс, незалежно від обраної мови, розроблений за єдиною структурою, яка дозволяє користувачеві поступово проходити завдання, від простих до складних. Спочатку це можуть бути елементарні вправи для знайомства із синтаксисом мови, які поступово переходять до більш складних задач, що потребують глибшого розуміння алгоритмів та структур даних. Такий підхід забезпечує плавне занурення у процес навчання, дозволяючи користувачеві не лише отримувати теоретичні знання, але й одразу ж застосовувати їх на практиці.

```
const tasks = {
  JavaScript: [
    {
      question: "1. Напишіть програму, яка виводить 'Привіт, світ!' на екран:",
      correctCode: "console.log('Привіт, світ!');",
      options: null,
      correctAnswer: null,
    },
    {
      question: "2. Напишіть програму, яка додає два числа:",
      correctCode: `let a = 5;
let b = 10;
console.log(a + b);`,
      options: null,
      correctAnswer: null,
    },
    {
      question: "3. Тестове питання: Що таке функція?",
      options: [
        "Множина даних",
        "Блок коду, що виконується по виклику",
        "Тип даних",
      ],
      correctAnswer: "Блок коду, що виконується по виклику",
    },
  ],
  Python: [
    {
      question: "1. Напишіть програму, яка виводить 'Привіт, світ!' на екран:",
      correctCode: "print('Привіт, світ!')",
      options: null,
      correctAnswer: null,
    },
  ],
}
```

```

{
  question: "2. Напишіть програму, яка додає два числа:",
  correctCode: `a = 5
b = 10
print(a + b)`,
  options: null,
  correctAnswer: null,
},
{
  question: "3. Тестове питання: Що таке функція?",
  options: [
    "Множина даних",
    "Блок коду, що виконується по виклику",
    "Тип даних",
  ],
  correctAnswer: "Блок коду, що виконується по виклику",
},
],
Java: [
  {
    question: "1. Напишіть програму, яка виводить 'Привіт, світ!' на екран:",
    correctCode: 'System.out.println("Привіт, світ!");',
    options: null,
    correctAnswer: null,
  },
  {
    question: "2. Напишіть програму, яка додає два числа:",
    correctCode: `int a = 5;
int b = 10;
System.out.println(a + b);`,
    options: null,
    correctAnswer: null,
  },
  {
    question: "3. Тестове питання: Що таке функція?",
    options: [
      "Множина даних",
      "Блок коду, що виконується по виклику",
      "Тип даних",
    ],
    correctAnswer: "Блок коду, що виконується по виклику",
  },
],
];

```

Ще одним важливим аспектом інтеграції навчальних курсів стало забезпечення механізмів перевірки правильності виконання завдань. Для цього було розроблено універсальний модуль перевірки, який працює з кількома мовами програмування. Цей модуль може обробляти код, написаний

користувачем, перевіряючи його на наявність синтаксичних помилок, правильність вирішення завдання та відповідність вимогам. Для кожної мови програмування були впроваджені спеціалізовані механізми перевірки, що враховують особливості кожної мови, наприклад, специфічні помилки синтаксису або специфіку використання певних конструкцій.

```
const handleCodeSubmit = () => {
  const correctCode = tasks[selectedLanguage][selectedTask].codeExample;
  if (userCode === correctCode) {
    setFeedback("Правильно!");
    setScore(score + 1);
  } else {
    setFeedback("Неправильно, спробуйте ще раз!");
  }
  setTaskCompleted(true);
};

const handleTestSubmit = () => {
  if (selectedAnswer === tasks[selectedLanguage][selectedTask].correctAnswer) {
    setFeedback("Правильно!");
    setScore(score + 1);
  } else {
    setFeedback("Неправильно, спробуйте ще раз!");
  }
  setTaskCompleted(true);
};
```

Щоб зробити процес навчання максимально інтерактивним, ми також впровадили систему зворотного зв'язку, яка надає користувачеві миттєву відповідь на виконане завдання. Це не просто перевірка правильності або неправильності відповіді, а й розширений аналіз помилок з порадами щодо їх виправлення. Якщо користувач допустив синтаксичну помилку, система вказує на неї та пояснює, як можна її виправити. Це дозволяє користувачеві не лише зрозуміти свої помилки, але й отримати негайні підказки щодо їхнього вирішення.

Крім того, було інтегровано функціонал збереження прогресу користувача. Це дозволяє платформі автоматично відстежувати виконані завдання, і зберігати стан курсу для кожного користувача. Така функція забезпечує зручність навчального процесу, оскільки користувач може

зупинитися на певному етапі, а потім продовжити навчання з того ж місця без втрати попередніх результатів.

Особливий акцент був зроблений на адаптації системи під потреби користувача на різних рівнях підготовки. Інтерфейс дозволяє обирати рівень складності завдань для кожної мови програмування, що сприяє поступовому розвитку навичок. Для новачків доступні завдання початкового рівня, які акцентують увагу на базових конструкціях мови, таких як змінні, цикли та умовні оператори. Більш досвідчені користувачі можуть обирати завдання середнього та високого рівнів складності, де вже потрібно застосовувати алгоритмічні підходи та працювати з більш складними структурами даних, такими як дерева або графи.

```
<section className="difficulty-levels">
  <h2>Модуль 1</h2>
  <div className="levels-container">
    <div className="level-item">
      <div className="level-circle"></div>
    </div>
    <div className="dot-separator"></div>
    <div className="level-item">
      <div className="level-circle"></div>
    </div>
    <div className="dot-separator"></div>
    <div className="level-item">
      <div className="level-circle"></div>
    </div>
  </div>
</section>
```

Також було розроблено механізм адаптації курсів під користувача, який на основі результатів попередніх завдань може пропонувати користувачеві більш складні або прості завдання. Це дозволяє системі навчання бути більш гнучкою та підлаштовуватися під індивідуальні потреби кожного користувача, забезпечуючи тим самим ефективніший процес навчання.

Таким чином, інтеграція системи навчальних курсів для різних мов програмування у нашої платформи дозволяє забезпечити гнучке, інтерактивне та індивідуальне навчання. Завдяки універсальній системі перевірки завдань,

зворотному зв'язку та адаптивним можливостям, платформа створює умови для ефективного навчання програмуванню, надаючи користувачам не тільки можливість засвоювати нові знання, але й відразу ж застосовувати їх на практиці, що є ключовим елементом у розвитку навичок програмування.

Інтеграція системи навчальних курсів для різних мов програмування також вимагала особливої уваги до архітектурних рішень, що дозволяли забезпечити масштабованість платформи. Однією з важливих вимог було створення модульної системи, яка дозволяла б легко додавати нові мови програмування та відповідні навчальні курси, без потреби суттєво змінювати основний код платформи. Для цього було використано концепцію компонентно-орієнтованого підходу до розробки, що дозволило створити окремі модулі для кожної мови, які взаємодіють із загальними механізмами платформи, такими як перевірка завдань та надання зворотного зв'язку.

Кожен курс, незалежно від обраної мови програмування, має чітку структуру, що складається з кількох розділів. Ці розділи включають як теоретичний матеріал, так і практичні завдання, що охоплюють різні аспекти програмування. Теоретичні блоки подаються у вигляді інтерактивних лекцій, які можна переглядати онлайн, а після цього переходити до виконання практичних завдань. Завдяки використанню сучасних технологій, таких як React та Vite, платформа забезпечує швидке завантаження та плавне перемикання між розділами курсу, що створює комфортний користувацький досвід.

Забезпечення гнучкості при реалізації різних типів завдань також стало важливим аспектом інтеграції системи навчальних курсів. Для кожної мови програмування були розроблені завдання, що включають як практичні вправи з написання коду, так і тестові запитання з вибором правильних відповідей. Завдяки цьому підходу користувачі можуть не лише розвивати свої навички програмування, але й перевіряти своє розуміння теоретичних концепцій. Крім того, для кожної мови програмування були додані спеціалізовані завдання, які враховують особливості цієї мови. Наприклад, для JavaScript були створені

завдання, що дозволяють користувачеві попрактикуватися у роботі з веб-API, тоді як для Python надаються завдання, орієнтовані на роботу з бібліотеками для аналізу даних.

Щоб забезпечити персоналізацію навчального процесу, було розроблено систему оцінювання та відстеження прогресу, яка враховує результати виконання завдань кожним користувачем. Ця система не лише фіксує, які завдання були виконані, але й аналізує, скільки часу користувач витратив на їх вирішення, скільки спроб було здійснено, та на які саме аспекти він звертав увагу під час виконання завдання. Зібрані дані використовуються для надання рекомендацій щодо подальших кроків у навчанні, а також для адаптації навчального процесу. Якщо користувач успішно виконав кілька завдань поспіль, система може запропонувати йому складніше завдання, тим самим стимулюючи його розвиток. Якщо ж користувач зазнає труднощів з виконанням завдань, платформа може порадити йому переглянути додатковий теоретичний матеріал або виконати простіші завдання для закріплення базових знань.

Особливої уваги потребувала розробка інтерфейсу користувача для системи навчальних курсів. Оскільки платформа повинна була підтримувати кілька мов програмування, інтерфейс мав бути інтуїтивно зрозумілим та зручним для навігації між різними розділами курсу. Для цього було реалізовано динамічне меню, яке дозволяє легко перемикатися між теоретичними матеріалами, завданнями та результатами виконання. Крім того, ми інтегрували функцію автоматичного збереження прогресу, яка дозволяє користувачам зупиняти навчання на будь-якому етапі та повертатися до нього пізніше, не втрачаючи виконаних завдань. Це було особливо важливо для тих користувачів, які не можуть виділяти багато часу на навчання за один сеанс.

Також ми подбали про інтеграцію із зовнішніми інструментами та сервісами для додаткових можливостей навчання. Наприклад, для курсів з Python ми передбачили можливість інтеграції з бібліотеками для машинного навчання, що дозволяє користувачам експериментувати з реальними наборами

даних та проводити аналітичні обчислення. Для JavaScript ми інтегрували можливість взаємодії з популярними бібліотеками для розробки інтерфейсів користувача, що дозволяє користувачам не лише вивчати мову, але й відразу ж застосовувати отримані знання на практиці, створюючи власні веб-додатки.

Важливою частиною інтеграції системи курсів стало впровадження механізмів обміну досвідом між користувачами платформи. У рамках платформи користувачі можуть ділитися своїми рішеннями, обговорювати проблеми, які виникли під час виконання завдань, а також отримувати поради від інших учасників спільноти. Це створює ефект спільного навчання, що значно підвищує мотивацію та результативність користувачів.

Таким чином, система навчальних курсів для різних мов програмування є комплексною та гнучкою частиною нашої платформи, яка забезпечує користувачам можливість навчатися в індивідуальному темпі та відповідно до власних потреб. Завдяки використанню сучасних технологій та підходів до розробки, наша платформа дозволяє ефективно вивчати різні мови програмування, надаючи користувачам необхідні інструменти для успішного засвоєння матеріалу та розвитку професійних навичок.

3.3. Тестування функціональності та юзабіліті гри

Перед тим як перейти до тестування нашої гри, проаналізуємо код який у нас вийшов. Аналізуючи код нашої платформи, можна виділити декілька важливих аспектів, які визначають її структуру та функціональність. Наша освітня платформа побудована з використанням фреймворку React, що дозволяє створювати динамічні інтерфейси для користувачів та ефективно працювати з управлінням станом програми. Основною метою платформи є надання можливості користувачам вибрати різні мови програмування для навчання, виконувати завдання різного типу (включаючи як написання коду, так і тестові запитання), а також отримувати зворотний зв'язок в режимі реального часу.

Ключовими елементами реалізації стали використання React-хуків, кастомних CSS-стилів, а також інтеграція функцій для обробки користувацьких даних.

Одна з базових функцій платформи – вибір мови програмування. Ця частина реалізована через використання React-хуків, зокрема хуків стану (useState), які дозволяють зберігати інформацію про вибір мови. Коли користувач вибирає одну з доступних мов (наприклад, JavaScript, Python або Java), відповідні завдання динамічно завантажуються на екран. Це дозволяє інтерактивно змінювати контент залежно від обраної мови програмування, що підвищує зручність використання платформи та робить її адаптивною до різних потреб користувачів.

Ще одним важливим елементом є система завдань, яка надає користувачам можливість виконувати різноманітні типи завдань. У коді платформи ми реалізували два основних види завдань: програмування та тестові питання з вибором відповідей. Завдання з програмування передбачають, що користувач вводить код у спеціально відведене поле, після чого його рішення перевіряється на наявність помилок та коректність. Для цього використовуються спеціальні функції в React, які обробляють введені дані та порівнюють їх з еталонними рішеннями. Тестові завдання, у свою чергу, дають можливість користувачам вибрати одну або кілька відповідей з наведених варіантів. Реалізована логіка перевірки дозволяє одразу визначити правильність вибору та надати відповідний зворотний зв'язок.

Важливою складовою є функція надання зворотного зв'язку користувачам. Після того, як користувач виконує завдання (незалежно від його типу), платформа надає зворотний зв'язок, інформуючи про те, чи було завдання виконане правильно. Для цього використовується логіка, яка аналізує введені користувачем дані та формує повідомлення на основі результатів. Цей механізм не лише допомагає користувачам краще зрозуміти свої помилки, але й мотивує їх до подальшого навчання та вдосконалення своїх навичок програмування.

Дизайн платформи також заслуговує на окрему увагу. Для надання користувачам сучасного та інтуїтивно зрозумілого інтерфейсу ми створили кастомні CSS-стилі, що відповідають загальній естетиці та функціональності платформи. Використовуючи сучасні практики веб-дизайну, ми забезпечили адаптивність інтерфейсу для різних розмірів екранів, що дозволяє користувачам комфортно працювати як на великих моніторах, так і на мобільних пристроях. Кольорові схеми, шрифти та розміщення елементів були ретельно продумані для полегшення навігації та покращення взаємодії з платформою. Крім того, ми додали інтерактивні елементи, такі як анімації при наведенні на кнопки та інші елементи керування, що робить взаємодію більш живою та привабливою для користувачів.

Що стосується архітектури коду, ми намагалися зробити його максимально модульним та легко підтримуваним. Кожен компонент платформи був винесений у окремий файл, що дозволяє легко оновлювати або розширювати функціонал у майбутньому без необхідності вносити зміни в інші частини коду. Такий підхід забезпечує гнучкість у розробці та підтримці проєкту, оскільки нові функції можуть бути додані без ризику порушити роботу існуючих компонентів.

Загалом, наш код(Додаток А) реалізує всі необхідні функції для створення інтерактивної платформи для навчання програмуванню. Він підтримує динамічне завантаження контенту, обробку користувацьких даних, перевірку завдань та надання зворотного зв'язку, що дозволяє користувачам активно навчатися, перевіряти свої знання та отримувати миттєвий зворотний зв'язок.

Тестування функціональності та юзабіліті гри є важливою частиною процесу розробки, оскільки воно дозволяє перевірити, наскільки ефективно працюють основні функції платформи, а також оцінити зручність взаємодії користувачів з інтерфейсом. У нашому випадку тестування охоплює дві основні складові: функціональність (перевірка коректності роботи всіх компонентів

платформи) та юзабіліті (оцінка зручності використання, інтуїтивності інтерфейсу, і комфортності роботи користувача).

Перший етап тестування передбачав перевірку основних функцій платформи, таких як вибір мови програмування, завдання для кожної мови, правильність обробки введених даних та надання зворотного зв'язку. Для цього ми створили кілька тестових сценаріїв, які включали перевірку на правильність завантаження завдань в залежності від вибраної мови програмування, перевірку роботи функціоналу для введення коду, а також перевірку правильності оцінки результатів виконання завдань. Усі тестові сценарії були спрямовані на перевірку кожного етапу взаємодії користувача з платформою, починаючи з вибору мови і закінчуючи отриманням результатів виконання завдання.

Для тестування нашої гри, спочатку оберемо мову програмування(рис 3.3.)

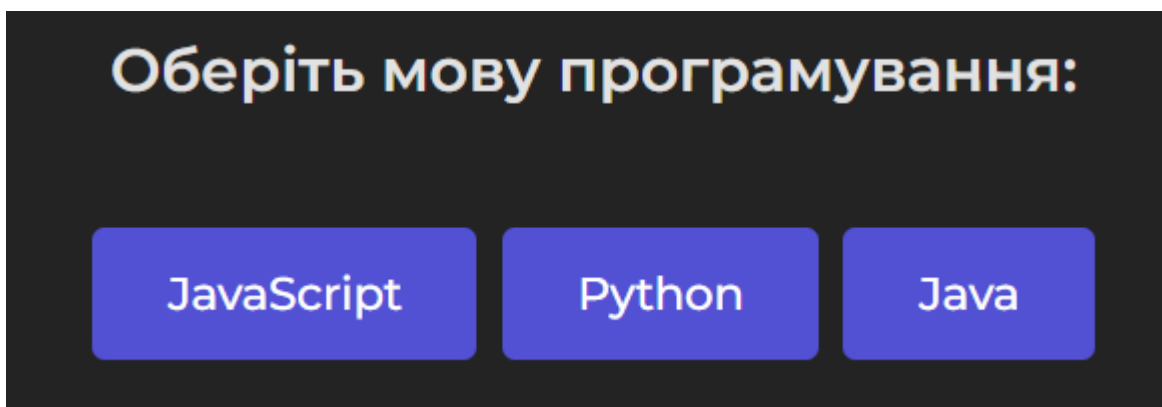


Рис 3.3. – Вибір мови програмування

після цього обраємо режим(рис 3.4.). Якщо завдання було виконано не правильно, бачимо помилку(рис 3.5.).

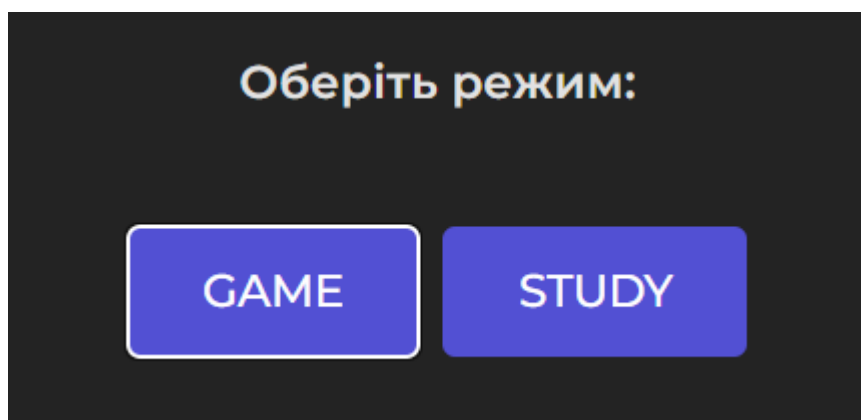


Рис 3.4. – Вибір режиму

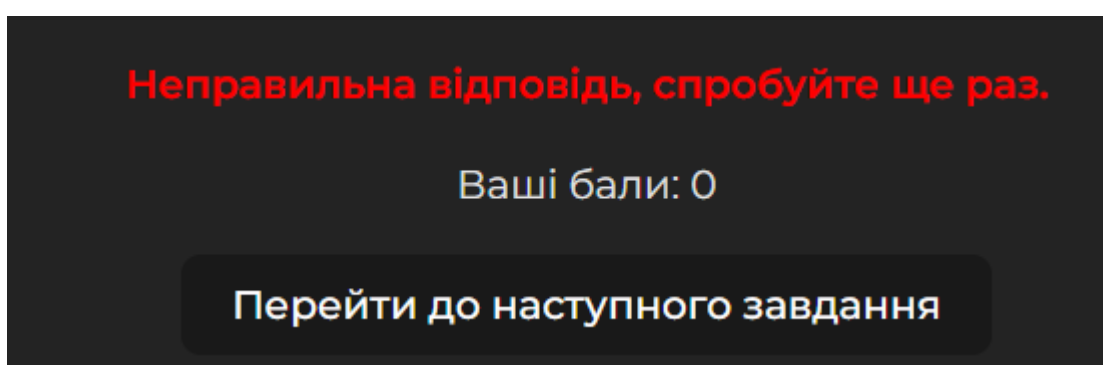


Рис 3.5. – Завдання виконано не правильно

Якщо завдання виконано правильно, спостерігаємо відповідний напис(рис 3.6.).

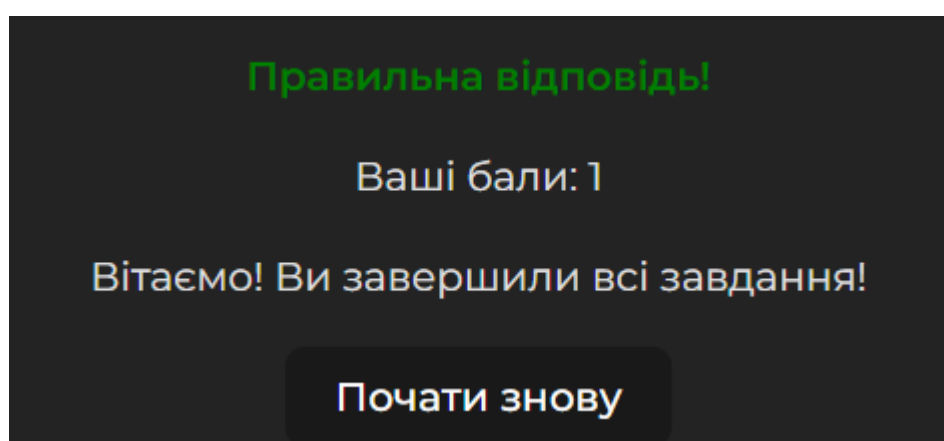


Рис 3.6. – Завдання виконано правильно

Кожен компонент, пов'язаний із виконанням завдань, був ретельно перевірений для виявлення помилок. Ми здійснили перевірку правильності

виконання завдань за кожною мовою програмування (JavaScript, Python, Java), а також протестували механізм зворотного зв'язку, щоб переконатися, що платформа правильно обробляє всі типи відповідей, незалежно від того, чи це тестове питання, чи програмний код. Для цього використовували як ручне тестування, так і автоматичні тести, які перевіряли логіку завдань та правильність результатів.

Другий етап тестування був орієнтований на юзабіліті платформи. Оскільки ми ставили перед собою завдання створити зручний і інтуїтивно зрозумілий інтерфейс, важливо було оцінити, як користувачі взаємодіють з елементами інтерфейсу. Для цього ми залучили декілька реальних користувачів, щоб вони пройшли через основні етапи взаємодії з платформою – вибір мови, виконання завдання, перегляд результатів. Спостерігаючи за їхньою взаємодією з платформою, ми змогли виявити проблеми, пов'язані з незручністю інтерфейсу, непорозумінням в навігації або складнощами при виконанні завдань. Були проведені сесії, під час яких користувачі могли ознайомитися з функціоналом платформи та дати зворотний зв'язок щодо її зручності.

Одним із ключових аспектів юзабіліті було забезпечення доступності платформи для різних типів користувачів, у тому числі з обмеженими можливостями. Ми тестували можливість використання клавіатурних скорочень, перевіряли контрастність тексту, адаптивність для мобільних пристроїв і переконувалися, що всі функції доступні без використання миші (наприклад, через навігацію за допомогою клавіатури).

За результатами тестування ми отримали низку рекомендацій для покращення інтерфейсу. Зокрема, було виявлено, що деякі елементи інтерфейсу мають бути більш помітними, а також, що в певних випадках навігація могла бути більш інтуїтивною. Також було виявлено, що деякі інтерфейсні елементи потребують додаткових підказок або пояснень для користувачів, які не знайомі з програмуванням.

Після внесення змін до інтерфейсу та вдосконалення функціональності, ми провели повторне тестування, яке показало значне покращення взаємодії користувачів з платформою. Інтерфейс став більш інтуїтивним, а завдання виконувались без затримок. Користувачі змогли без проблем орієнтуватися на платформі, а процес навчання став більш приємним і ефективним.

Загалом, тестування функціональності та юзабіліті дозволило не лише переконатися у правильності роботи всіх основних функцій платформи, але й виявити можливі проблеми з інтерфейсом та взаємодією користувача з платформою. Після внесення змін платформа стала більш зручною та доступною для різних типів користувачів, що сприяє підвищенню її ефективності як інструменту для навчання програмуванню.

Після завершення тестування функціональності та юзабіліті платформи наступним етапом стала оптимізація продуктивності системи. Важливою частиною тестування стало виявлення проблем, які могли б вплинути на швидкість і стабільність роботи платформи, особливо при збільшенні кількості користувачів або завдань. Під час тестування ми також звертали увагу на можливі проблеми із затримками в обробці даних, які могли б виникнути при виконанні складних завдань або роботі з великими обсягами даних.

Основну увагу було приділено таким аспектам, як швидкість завантаження сторінок, час виконання коду користувача та швидкість надання результатів. Для цього ми використовували різноманітні інструменти для моніторингу продуктивності, щоб виміряти час, необхідний для завантаження компонентів інтерфейсу, а також час, що витрачається на обробку запитів і виконання завдань користувачів. У результаті тестувань було виявлено кілька місць, де система могла працювати повільніше, ніж очікувалося. Це стосувалося, зокрема, великих обсягів коду, що передавався для перевірки, а також певних компонентів інтерфейсу, які завантажувалися не так швидко, як це було необхідно для комфортного використання.

Після того як були виявлені потенційні проблеми з продуктивністю, ми провели оптимізацію відповідних компонентів. Для покращення швидкості

завантаження сторінок було застосовано різні техніки, такі як відкладене завантаження деяких ресурсів (lazy loading), що дозволило зменшити навантаження на сервер і пришвидшити інтерфейс. Крім того, були оптимізовані алгоритми перевірки коду, щоб зменшити час, необхідний для обробки запитів користувачів, що виконували завдання. Оптимізація запитів до бази даних також дозволила значно прискорити взаємодію з платформою та зменшити час очікування результатів.

Особливу увагу було приділено тестуванню платформи на мобільних пристроях. Оскільки платформа має бути доступною не тільки на десктопах, але й на смартфонах і планшетах, тестування на різних розмірах екранів стало важливою частиною процесу. Платформа була адаптована до різних типів екранів, що дозволило забезпечити комфортне використання користувачами на мобільних пристроях. Було проведено тестування на різних версіях браузерів, а також на пристроях з різними розмірами екранів, що допомогло виявити потенційні проблеми з адаптивністю інтерфейсу.

Завершальним етапом тестування була перевірка безпеки платформи. Безпека даних користувачів є важливою складовою платформи, тому ми провели серію тестів для виявлення можливих вразливостей. Зокрема, перевіряли захист від SQL-ін'єкцій, захист даних користувачів під час реєстрації та авторизації, а також захист від атак типу XSS. Крім того, була проведена перевірка правильності зберігання паролів користувачів у зашифрованому вигляді та забезпечення безпеки під час передачі даних між клієнтом і сервером через використання HTTPS протоколу.

Після всіх тестувань і оптимізацій ми отримали платформу, яка працює стабільно, швидко і безпечно. Тестування функціональності та юзабіліті допомогло не тільки знайти і виправити помилки в коді та інтерфейсі, але й значно покращити продуктивність і взаємодію користувачів з платформою. Платформа стала більш зручною для використання, а її швидкодія дозволяє ефективно використовувати її в умовах різних пристроїв і платформ, що є важливим аспектом для інструмента для навчання програмуванню.

Однак, навіть після завершення основних етапів тестування, ми не припиняємо процес вдосконалення системи. Збір зворотного зв'язку від користувачів і подальше тестування нових функцій дозволяють постійно вдосконалювати платформу і підвищувати її якість. Це забезпечить не тільки стабільну роботу на довгострокову перспективу, а й підвищить задоволення користувачів від взаємодії з платформою, що є одним із основних критеріїв успішності в сфері розробки освітніх технологій.

3.4. Оцінка ефективності навчання та плани на подальший розвиток платформи

Оцінка ефективності навчального процесу є ключовим аспектом розробки освітньої платформи, оскільки саме результати навчання визначають її успішність та затребуваність серед користувачів. Для аналізу ефективності ми провели серію тестувань, які включали як технічні аспекти роботи системи, так і аналіз взаємодії користувачів із платформою. Було важливо зрозуміти, наскільки добре користувачі освоюють матеріал, чи сприяє гейміфікація підвищенню мотивації до навчання, а також як зручний і зрозумілий інтерфейс впливає на ефективність проходження завдань.

Під час тестування ми відстежували різні параметри, зокрема рівень завершення курсів, час, витрачений на виконання завдань, частоту повернення користувачів до навчального процесу, а також кількість помилок у виконаних завданнях. Дані свідчать про те, що інтеграція ігрових механік дійсно підвищує зацікавленість користувачів. Елементи гейміфікації, такі як система балів, рівнів і нагород, сприяли тому, що користувачі прагнули досягти кращих результатів і поверталися до навчання частіше. Також важливим аспектом стало те, що можливість практичного виконання коду безпосередньо на платформі значно спрощує процес навчання та робить його більш ефективним.

Аналізуючи результати тестування, можна зробити висновок, що найбільш ефективним є поєднання теоретичних матеріалів із практичними

завданнями. Завдання різного рівня складності дозволяють користувачам поступово нарощувати свої знання та навички, а наявність тестових запитань після кожного блоку допомагає закріпити отриманий матеріал. Крім того, можливість одразу отримувати зворотний зв'язок щодо правильності розв'язку мотивує користувачів виправляти помилки та рухатися вперед, не зупиняючись на невдачах.

Незважаючи на позитивні результати, у процесі тестування ми також виявили аспекти, які можна покращити. Деякі користувачі відзначали, що певні завдання потребують більш детального пояснення або прикладів, що свідчить про необхідність розширення теоретичної частини курсу. Також важливим є удосконалення алгоритмів перевірки коду, щоб ще точніше аналізувати введені користувачем рішення та надавати більш детальні підказки у випадку помилок.

Зважаючи на отримані результати, ми сформували плани щодо подальшого розвитку платформи. Одним із ключових напрямків є розширення бази навчальних курсів, зокрема додавання нових мов програмування, таких як C#, Swift, та інших популярних технологій. Це дозволить охопити ширшу аудиторію користувачів і зробити платформу корисною для різних категорій студентів – від початківців до досвідчених програмістів.

Ще одним важливим напрямком є розширення гейміфікаційних елементів. Планується додати інтерактивні змагання між користувачами, а також спільні командні проекти, що сприятиме розвитку навичок командної роботи. Крім того, ми розглядаємо можливість інтеграції штучного інтелекту для аналізу стилю коду користувачів і надання персоналізованих рекомендацій щодо покращення їхніх рішень.

З метою покращення комунікації між користувачами буде додано можливість залишати коментарі до завдань, обговорювати різні способи розв'язання та ставити запитання в межах платформи. Це допоможе створити спільноту навколо платформи, де користувачі зможуть ділитися досвідом і допомагати один одному у навчанні.

Окрім технічних удосконалень, ми також плануємо співпрацю з освітніми установами та ІТ-компаніями, що дозволить створювати сертифіковані курси та надавати можливості для стажування успішним студентам. Це значно підвищить мотивацію користувачів, оскільки вони отримають реальні перспективи працевлаштування після проходження курсів.

Для покращення ефективності навчання планується реалізація додаткових механізмів підтримки користувачів. Одним із таких елементів є персоналізовані рекомендації на основі попереднього досвіду користувача. Використання алгоритмів машинного навчання дозволить адаптивно підбирати завдання відповідно до рівня підготовки студента. Це забезпечить більш ефективне засвоєння матеріалу, адже кожен користувач отримуватиме оптимальний рівень складності вправ.

Ще одним важливим аспектом є вдосконалення зворотного зв'язку. Крім автоматичних перевірок завдань, передбачається можливість отримання розгорнутих пояснень до правильних відповідей та розширених підказок. Це допоможе користувачам не лише бачити свої помилки, а й розуміти причини їх виникнення та вчитися на них.

Додатково розглядається можливість інтеграції платформи з іншими системами навчання та використання її в освітніх закладах. Планується розробка API, який дозволить викладачам створювати власні курси та інтегрувати їх у платформу. Це дозволить розширити функціонал та зробити платформу більш універсальним інструментом для навчання.

У перспективі передбачається розширення списку підтримуваних мов програмування, додавання нових типів завдань та інтеграція із середовищами розробки. Також розглядається можливість створення мобільного додатка, який дозволить користувачам навчатися у зручному форматі на будь-якому пристрої.

Таким чином, наша платформа вже показала ефективність у навчанні програмуванню, проте ми продовжуватимемо її вдосконалювати, додаючи нові функції та можливості. Подальший розвиток включатиме не лише розширення навчальної бази, а й поліпшення взаємодії користувачів із системою,

персоналізацію навчального процесу та інтеграцію новітніх технологій для підвищення ефективності навчання.

Крім зазначених покращень, ми також розглядаємо можливість інтеграції адаптивного навчання, яке базуватиметься на результатах попередніх тестувань користувача. Це означає, що система зможе автоматично підлаштовувати рівень складності завдань, враховуючи індивідуальні потреби та прогрес кожного студента. Завдяки цьому навчання стане ще більш ефективним і персоналізованим.

Ще одним напрямком розвитку є розширення системи мотивації. Планується введення щоденних і щотижневих челенджів, які спонукатимуть користувачів регулярно виконувати завдання та підтримувати високий рівень зацікавленості. Крім того, ми розглядаємо можливість додавання спеціальних бонусів за активність на платформі, таких як доступ до ексклюзивних матеріалів чи додаткові навчальні ресурси.

У сфері технічних удосконалень ми також працюємо над оптимізацією продуктивності платформи, що дозволить прискорити завантаження сторінок, поліпшити роботу редактора коду та забезпечити стабільність роботи навіть при високому навантаженні. Ми плануємо впровадити сучасні технології кешування та розподілених обчислень для ще більш плавного користувацького досвіду.

Окрему увагу буде приділено безпеці даних користувачів. Ми працюємо над посиленням механізмів автентифікації, зокрема впровадженням двофакторної аутентифікації (2FA) та покращенням системи захисту від несанкціонованого доступу. Це дозволить забезпечити безпечне середовище для навчання та захист персональних даних користувачів.

Довгостроковими планами є створення глобальної екосистеми навчання, яка дозволить не лише опановувати програмування, а й взаємодіяти з потенційними роботодавцями. Ми працюємо над партнерствами з компаніями, що надаватимуть можливість користувачам проходити сертифікаційні курси,

отримувати рекомендації від провідних спеціалістів галузі та навіть проходити стажування безпосередньо через платформу.

Завдяки цим покращенням наша освітня платформа стане ще більш ефективною, доступною та привабливою для широкого кола користувачів. Ми продовжуватимемо вдосконалювати систему, впроваджуючи найновітніші технології та підходи до навчання, щоб забезпечити якісний освітній процес і сприяти професійному розвитку користувачів.

ВИСНОВКИ

У ході виконання дипломної роботи було розроблено інтерактивну освітню платформу для вивчення основ програмування, що поєднує теоретичний матеріал із практичними завданнями. Основною метою проекту було створення ефективного та зручного інструменту для навчання, який би дозволяв користувачам освоювати основи програмування за допомогою ігрових механік та миттєвого зворотного зв'язку.

У результаті проведених досліджень та практичної реалізації платформи було підтверджено, що гейміфікація є ефективним підходом до підвищення мотивації користувачів. Використання системи балів, рівнів складності та

нагород стимулювало студентів до проходження завдань та повернення до навчального процесу. Крім того, інтеграція тестових завдань після кожного блоку теоретичних матеріалів сприяла кращому засвоєнню знань та їх практичному застосуванню.

Важливим досягненням є адаптивність платформи до різних рівнів підготовки користувачів. Розроблені завдання різного рівня складності дозволяють як новачкам, так і більш досвідченим програмістам розвивати свої навички. Впровадження механізму миттєвого зворотного зв'язку у перевірці завдань сприяло швидкому виявленню помилок та їх виправленню, що значно покращило ефективність навчального процесу.

Разом з тим, аналіз тестування системи виявив деякі аспекти, які можуть бути покращені у майбутньому. Зокрема, користувачі відзначили потребу у розширенні теоретичного матеріалу, наявності детальніших пояснень до деяких завдань та вдосконаленні алгоритмів перевірки коду, щоб забезпечити більш точні підказки при виправленні помилок.

Також було проведено аналіз існуючих освітніх платформ та методик викладання програмування. Це дозволило визначити найкращі практики, які можуть бути впроваджені для підвищення ефективності навчання. Зокрема, враховано досвід популярних онлайн-курсів та платформ, що використовують адаптивне навчання, персоналізовані завдання та штучний інтелект для оцінки знань.

Дипломна робота також розглядає технічні аспекти реалізації платформи, включаючи вибір технологічного стеку, архітектуру системи та методи забезпечення безпеки користувацьких даних. Реалізація проєкту на основі сучасних веб-технологій дозволяє забезпечити високу продуктивність, масштабованість та зручність використання для широкого кола користувачів.

Окрім цього, було здійснено тестування роботи платформи на групі користувачів із різним рівнем підготовки. Аналіз отриманих результатів показав, що більшість учасників позитивно оцінили інтерфейс, функціональність та зручність використання. При цьому було виявлено ряд

потенційних покращень, зокрема додаткові інтерактивні елементи, можливість персоналізованих рекомендацій та розширення набору підтримуваних мов програмування.

Значна увага приділена безпеці користувачів, що включає захист персональних даних, безпечне збереження результатів навчання та реалізацію механізмів автентифікації. Подальші дослідження у цьому напрямку дозволять удосконалити захист платформи від можливих кіберзагроз.

Окрім технічних і методичних аспектів, розроблена платформа має значний потенціал для подальшого розвитку. Одним із перспективних напрямків є інтеграція елементів штучного інтелекту для автоматизованого аналізу помилок у коді та надання персоналізованих рекомендацій користувачам. Це дозволить покращити якість навчання та зробити процес освоєння програмування більш інтуїтивним і доступним.

Також можливим напрямком удосконалення є розширення платформи за рахунок мобільної версії, що забезпечить зручність доступу до навчальних матеріалів у будь-який час та з будь-якого пристрою. Окрім цього, інтеграція платформи з іншими освітніми сервісами та можливість спільного навчання у групах можуть сприяти підвищенню взаємодії між користувачами та створенню спільноти для обміну знаннями.

Розроблена система оцінювання прогресу користувачів також може бути доповнена розширеною аналітикою та звітами про успішність, що дозволить як студентам, так і викладачам більш ефективно відстежувати рівень засвоєння матеріалу. Враховуючи отримані результати тестування та зворотний зв'язок від користувачів, впровадження цих функцій сприятиме підвищенню ефективності навчання та адаптації платформи до індивідуальних потреб кожного студента.

Таким чином, виконана дипломна робота не лише реалізувала поставлену мету, але й відкрила перспективи для подальшого розвитку платформи. Запропоновані покращення та інтеграція сучасних технологій сприятимуть створенню ще більш ефективного та інноваційного інструменту для навчання

програмуванню, який може бути корисним як для індивідуального навчання, так і для використання у закладах освіти.

Загалом, створена платформа продемонструвала свою ефективність як інструмент для навчання програмуванню. Вона може бути використана як самостійний освітній ресурс або як допоміжний інструмент для викладачів у навчальних закладах. Подальше вдосконалення платформи, розширення її функціональності та вдосконалення алгоритмів оцінювання можуть зробити її ще більш корисною для користувачів і сприяти підвищенню якості освітнього процесу.

ПОСИЛАННЯ І ЛІТЕРАТУРА

1. CodeCombat - <https://codecombat.com>
2. CodinGame - <https://www.codingame.com/start/>
3. Tynker - <https://www.tynker.com>
4. Kodable - <https://www.kodable.com>
5. Scratch - <https://scratch.mit.edu>
6. Robocode - <https://robocode.ua>
7. Lightbot - <https://lightbot.com>

8. Codecademy

https://www.codecademy.com/?g_network=g&g_productchannel=&g_adid=624951457618&g_locinterest=&g_keyword=codecademy&g_acctid=243-039-7011&g_adtype=&g_keywordid=kwd-41065460761&g_ifcreative=&g_campaign=account&g_locphysical=9061017&g_adgroupid=70492864474&g_productid=&g_source={sourceid}&g_merchantid=&g_placement=&g_partition=&g_campaignid=1726903838&g_ifproduct=&utm_id=t_kwd-41065460761:ag_70492864474:cp_1726903838:n_g:d_c&utm_source=google&utm_medium=paid-search&utm_term=codecademy&utm_campaign=INTL_Brand_Exact&utm_content=624951457618&g_adtype=search&g_acctid=243-039-7011&gad_source=1&gclid=CjwKCAjw-qi_BhBxEiwAkxvbkOUswP2E8GCJshlsDW6Efp8ONDj3tqfVqWpQGoFySDJozjGJe7WcRBoCqfYQAvD_BwE

9. freeCodeCamp - <https://www.freecodecamp.org>10. Coursera - <https://www.coursera.org>

11. Udemy

<https://www.udemy.com/ru/?srsltid=AfmBOor6TOun7KhKH1Cp07MkSQW1V1eMNEXX5aFRWzsT9-0L6KMjAk3g>

12. Pluralsight - <https://www.pluralsight.com>13. Khan Academy - <https://ru.khanacademy.org>14. LeetCode - <https://leetcode.com>15. edX - <https://www.edx.org>16. HackerRank - <https://www.hackerrank.com>17. React - <https://ru.legacy.reactjs.org>18. Angular - <https://angular.dev>19. Vue.js - <https://vuejs.org>

20. Навіщо розуміти backend-частину та чому варто обрати Node.js - <https://robotdreams.cc/uk/blog/241-zachem-ponimat-backend-chast-i-pochemu-stoit-vybrat-node-js>
21. Реляційні бази даних: Що це таке і як вони працюють - <https://alexhost.com/uk/faq/relyaczijni-bazy-danyh-shho-cze-take-i-yak-vony-praczyuyut/>
22. Amazon Web Services - <https://aws.amazon.com/ru/>
23. Google Cloud - <https://cloud.google.com>
24. Міньков Костянтин Павлович Система захисту мікросервісних систем - Київ – 2019 року - 24-27 ст.
25. Google Analytics - <https://developers.google.com/analytics?hl=ru>
26. Mixpanel - <https://mixpanel.com>
27. Vite - <https://vitejsvitetygarply-u45k--5173--495c5120.local-corp.webcontainer.io>
28. Суханова Марія Юріївна Веб-застосунок для продажу аксесуарів із застосуванням фреймворку React - Київ – 2023 р - 14-15 ст.
29. Навчально-науковий Інститут прикладного системного аналізу КПІ ім. Ігоря Сікорського Системні науки та інформатика - 25–29 листопада 2024 року, Київ, Україна - 164-170 ст.
30. Іван ДЕМИДОВ 82-га студентська науково-технічна конференція - Жовтень 2024 року, Львів - 5-7 ст.
31. Букатич А.В Дослідження методів гейміфікації процесу онлайн навчання на прикладі розробки веб-застосунку для опанування навичок сліпого набору - 2024 р., Харків - 49-50 ст.

ДОДАТКИ

Додаток А

App.jsx

```
import React, { useState } from "react";
import Game from "../Game";
import "../App.css";

function Feedback({ isCorrect }) {
  return (
    <div className={isCorrect ? 'correct-answer' : 'incorrect-answer'}>
```

```

    {isCorrect ? 'Правильна відповідь!' : 'Неправильна відповідь, спробуйте ще раз.'}
  </div>
);
}

const App = () => {
  const [selectedLanguage, setSelectedLanguage] = useState(null);
  const [selectedMode, setSelectedMode] = useState(null);
  const [selectedTask, setSelectedTask] = useState(0);
  const [score, setScore] = useState(0);
  const [taskCompleted, setTaskCompleted] = useState(false);
  const [feedback, setFeedback] = useState("");
  const [userCode, setUserCode] = useState("");
  const [selectedAnswer, setSelectedAnswer] = useState("");
  const [isModalOpen, setIsModalOpen] = useState(false);
  const [name, setName] = useState("");
  const [isRegistered, setIsRegistered] = useState(false);
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [error, setError] = useState("");
  const [user, setUser] = useState(null);
  const [progress, setProgress] = useState(50);
  const [showContacts, setShowContacts] = useState(false);
  const [highlightAboutUs, setHighlightAboutUs] = useState(false);

  const tasks = {
    JavaScript: [
      {
        question: "1. Напишіть програму, яка виводить 'Привіт, світ!' на екран:",
        correctCode: "console.log('Привіт, світ!');",
        options: null,
        correctAnswer: null,
      },
      {
        question: "2. Напишіть програму, яка додає два числа:",
        correctCode: `let a = 5;
let b = 10;
console.log(a + b);`,
        options: null,
        correctAnswer: null,
      },
      {
        question: "3. Тестове питання: Що таке функція?",
        options: [
          "Множина даних",
          "Блок коду, що виконується по виклику",
          "Тип даних",
        ],
        correctAnswer: "Блок коду, що виконується по виклику",
      },
    ],
  };

```

```

    },
  ],
  Python: [
    {
      question: "1. Напишіть програму, яка виводить 'Привіт, світ!' на екран:",
      correctCode: "print('Привіт, світ!')",
      options: null,
      correctAnswer: null,
    },
    {
      question: "2. Напишіть програму, яка додає два числа:",
      correctCode: `a = 5
b = 10
print(a + b)`,
      options: null,
      correctAnswer: null,
    },
    {
      question: "3. Тестове питання: Що таке функція?",
      options: [
        "Множина даних",
        "Блок коду, що виконується по виклику",
        "Тип даних",
      ],
      correctAnswer: "Блок коду, що виконується по виклику",
    },
  ],
  Java: [
    {
      question: "1. Напишіть програму, яка виводить 'Привіт, світ!' на екран:",
      correctCode: 'System.out.println("Привіт, світ!");',
      options: null,
      correctAnswer: null,
    },
    {
      question: "2. Напишіть програму, яка додає два числа:",
      correctCode: `int a = 5;
int b = 10;
System.out.println(a + b);`,
      options: null,
      correctAnswer: null,
    },
    {
      question: "3. Тестове питання: Що таке функція?",
      options: [
        "Множина даних",
        "Блок коду, що виконується по виклику",
        "Тип даних",
      ],
      correctAnswer: "Блок коду, що виконується по виклику",
    },
  ],

```

```

    },
  ],
};

const handleLanguageSelect = (language) => {
  setSelectedLanguage(language);
  setSelectedMode(null);
  setSelectedTask(0);
  setScore(0);
  setTaskCompleted(false);
  setFeedback("");
  setUserCode("");
  setSelectedAnswer("");
};

const handleAboutUsClick = () => {
  setHighlightAboutUs(true);
  // Після 3 секунд вимкнути підсвічування
  setTimeout(() => {
    setHighlightAboutUs(false);
  }, 3000); // 3 секунди
};

const handleMouseEnter = () => {
  setShowContacts(true);
};

const handleMouseLeave = () => {
  setShowContacts(false);
};

const handleCodeChange = (e) => {
  setUserCode(e.target.value);
};

const goBack = () => {
  setSelectedMode(null); // Повернення до вибору режиму
  setSelectedTask(0);    // Скидання поточного завдання
  setTaskCompleted(false); // Скидання завершення завдання
  setFeedback("");      // Скидання зворотного зв'язку
};

const handleCodeSubmit = () => {
  const correctCode = tasks[selectedLanguage][selectedTask].codeExample;
  if (userCode === correctCode) {
    setFeedback("Правильно!");
    setScore(score + 1);
  } else {
    setFeedback("Неправильно, спробуйте ще раз!");
  }
  setTaskCompleted(true);
};

```

```

};

const handleTestSubmit = () => {
  if (selectedAnswer === tasks[selectedLanguage][selectedTask].correctAnswer) {
    setFeedback("Правильно!");
    setScore(score + 1);
  } else {
    setFeedback("Неправильно, спробуйте ще раз!");
  }
  setTaskCompleted(true);
};

const handleModeSelect = (mode) => {
  setSelectedMode(mode);
  setTaskCompleted(false);
  setSelectedTask(0);
};

const openModal = () => {
  setIsModalOpen(true);
};

const closeModal = () => {
  setIsModalOpen(false);
};

const handleUserClick = () => {
  setIsModalOpen(true);
};

const handleRegistrationSubmit = (e) => {
  e.preventDefault();
  if (!name || !email || !password) {
    setError("Будь ласка, заповніть всі поля.");
  } else {
    setError("");
    console.log("Реєстрація успішна:", { name, email, password });
    setIsRegistered(true);
    closeModal();
  }
};

const renderLesson = () => {
  if (selectedLanguage && selectedMode === "STUDY" && selectedTask <
tasks[selectedLanguage].length) {
    const task = tasks[selectedLanguage][selectedTask];

    return (
      <>
      <h2>Урок 1: Вступ до {selectedLanguage}</h2>

```

```

<p>Завдання: {task.question}</p>

{task.options ? (
  <>
    <p>Оберіть правильну відповідь:</p>
    {task.options.map((option, index) => (
      <div key={index}>
        <input
          type="radio"
          id={option}
          name="answer"
          value={option}
          onChange={(e) => setSelectedAnswer(e.target.value)}
        />
        <label htmlFor={option}>{option}</label>
      </div>
    ))}
    <button onClick={handleTestSubmit}>Перевірити відповідь</button>
  </>
) : (
  <>
    <p>Введіть свій код:</p>
    <textarea
      value={userCode}
      onChange={handleCodeChange}
      placeholder="Введіть свій код тут..."
    ></textarea>
    <button onClick={handleCodeSubmit}>Перевірити</button>
  </>
)}

{taskCompleted && (
  <div>
    <Feedback isCorrect={feedback === "Правильно!"} />
    <p>Ваші бали: {score}</p>
    {selectedTask + 1 < tasks[selectedLanguage].length ? (
      <button
        onClick={() => {
          const nextTask = selectedTask + 1;
          setSelectedTask(nextTask);
          setTaskCompleted(false);
          setFeedback("");
          setSelectedAnswer("");
        }}
      >
        Перейти до наступного завдання
      </button>
    ) : (
      <div>
        <p>Вітаємо! Ви завершили всі завдання!</p>

```

```

        <button onClick={() => setSelectedTask(0)}>
            Почати знову
        </button>
    </div>
    )}
</div>
)}
</>
);
} else {
    return <p>Виберіть завдання, щоб почати!</p>;
}
};

return (
    <div className="app-container">
        {/* Хедер */}
        <header className="header">
            <div className="logo">CodeFlow</div>
            <div className="header-menu">
                <a href="#menu">Меню</a>
                <a href="#contacts"
                    onMouseEnter={handleMouseEnter}
                    onMouseLeave={handleMouseLeave}
                >
                    Контакти
                </a>
                <a
                    href="#about-us"
                    onClick={handleAboutUsClick}
                    className={highlightAboutUs ? 'highlight' : ''}
                >
                    Про нас
                </a>

                {isRegistered ? (
                    <a href="#" className="user-name">
                        {name}
                    </a>
                ) : (
                    <a href="#" onClick={openModal}>
                        Реєстрація
                    </a>
                )}
            </div>
        </header>

        {showContacts && (
            <div className={`contact-dropdown ${showContacts ? "show" : ""}`}>

```

```

    <p>Адреса: вул. Програмістів 15, Київ, Україна</p>
    <p>Телефон: +38 050 123 45 67</p>
  </div>
)}

{isModalOpen && (
  <div className="modal-overlay" onClick={closeModal}>
    <div className="modal" onClick={(e) => e.stopPropagation()}>
      <button className="modal-close" onClick={closeModal}>
      </button>
      <h2>Реєстрація</h2>
      <form onSubmit={handleRegistrationSubmit}>
        <input
          type="text"
          placeholder="Name"
          value={name}
          onChange={(e) => setName(e.target.value)}
        />
        <input
          type="email"
          placeholder="Email"
          value={email}
          onChange={(e) => setEmail(e.target.value)}
        />
        <input
          type="password"
          placeholder="Password"
          value={password}
          onChange={(e) => setPassword(e.target.value)}
        />
        <button type="submit">Зареєструватися</button>
        {error && <p className="error">{error}</p>}
      </form>
    </div>
  </div>
)}

{/* Основний заголовок */}
<h1 className="main-heading">Занурюйся у програмування з CodeFlow!</h1>

{/* Про нас */}
<section id="about-us" className={`about-us ${highlightAboutUs ? 'highlight'
: ''}`>
  <div className="about-us-content">
    <div className="about-text">
      <h2>Про нас</h2>
      <p>
        CodeFlow — це онлайн-платформа, яка допомагає вам вивчати мови
        програмування через інтерактивні завдання та тести.

```

Наше завдання – допомогти вам освоїти основи програмування, незалежно від вашого рівня підготовки.

Кожен урок розроблений таким чином, щоб ви могли не лише вивчати синтаксис мови, але й розв'язувати реальні проблеми.

</p>

<p>

Ми віримо, що навчання програмуванню повинно бути цікавим, доступним та ефективним. Тому ми створили CodeFlow, щоб кожен міг покращити свої навички та стати впевненим програмістом.

</p>

</div>

<div className="about-image">

</div>

</div>

</section>

{/* Вибір мови програмування */}

{selectedLanguage === null ? (

<div className="container">

<h2>Оберіть мову програмування:</h2>

<div className="buttons-container">

<button onClick={() =>

handleLanguageSelect("JavaScript")}>JavaScript</button>

<button onClick={() => handleLanguageSelect("Python")}>Python</button>

<button onClick={() => handleLanguageSelect("Java")}>Java</button>

</div>

</div>

) : selectedMode === null ? (

<div className="container">

<h3>Оберіть режим:</h3>

<div className="buttons-container">

<button onClick={() => handleModeSelect("GAME")}>GAME</button>

<button onClick={() => handleModeSelect("STUDY")}>STUDY</button>

</div>

</div>

) : selectedMode === "GAME" ? (

<div>

<Game />

<button className="back-button" onClick={goBack}>Назад</button>

</div>

) : (

<div className="container">

{/* Тут буде відображатися урок після вибору мови та режиму */}

{renderLesson()}

<button className="back-button" onClick={goBack}>Назад</button>

</div>

)}

```

    { /* Відгуки */ }
    <section className="reviews">
    <h2>Що про нас кажуть наші користувачі</h2>
    <div className="reviews-list">
      <div className="review-item">
        <div className="reviewer-info">
          
          <p className="reviewer-name">Олександр, початківець</p>
        </div>
        <p>"CodeFlow – це чудовий ресурс для навчання програмуванню! Завдяки
і інтерактивним завданням я швидко освоїв нові мови."</p>
      </div>
      <div className="review-item">
        <div className="reviewer-info">
          
          <p className="reviewer-name">Ірина, студентка</p>
        </div>
        <p>"Завдання дуже цікаві, і кожен урок допомагає покращити мої навички.
Платформа дуже зручна!"</p>
      </div>
      <div className="review-item">
        <div className="reviewer-info">
          
          <p className="reviewer-name">Андрій, програміст</p>
        </div>
        <p>"CodeFlow допоміг мені вивчити JavaScript за рекордно короткий час. Чудова
платформа для самостійного навчання!"</p>
      </div>
      <div className="review-item">
        <div className="reviewer-info">
          
          <p className="reviewer-name">Марія, розробниця</p>
        </div>

```

```

        <p>"Мені дуже подобається, що тут можна не тільки вивчати синтаксис, але й
розв'язувати реальні задачі!"</p>
    </div>
</div>
</section>

{/*Завдання*/}
<section className="difficulty-levels">
    <h2>Модуль 1</h2>
    <div className="levels-container">
        <div className="level-item">
            <div className="level-circle"></div>
        </div>
        <div className="dot-separator"></div>
        <div className="level-item">
            <div className="level-circle"></div>
        </div>
        <div className="dot-separator"></div>
        <div className="level-item">
            <div className="level-circle"></div>
        </div>
    </div>
</section>

    {/* Футер */}
    <footer>
        <div className="footer-content">
            <div className="social-media">
                <a href="https://instagram.com" target="_blank" rel="noopener
norereferrer">Instagram</a>
                <a href="https://facebook.com" target="_blank" rel="noopener
norereferrer">Facebook</a>
                <a href="https://tiktok.com" target="_blank" rel="noopener
norereferrer">TikTok</a>
            </div>
            <div className="contact-info">
                <p>Адреса: вул. Програмістів 15, Київ, Україна</p>
                <p>Телефон: +38 050 123 45 67</p>
            </div>
        </div>
    </footer>

</div>
);
};

export default App;

```

Game.jsx

```

import React, { useState } from 'react';
import './App.css';

const LanguageSelection = ({ onLanguageSelect }) => {
  const handleLanguageSelect = (language) => {
    onLanguageSelect(language);
  };

  return (
    <div className="language-selection">
      <h2>Виберіть мову програмування:</h2>
      <div className="button-container">
        <button className="language-button" onClick={() =>
handleLanguageSelect('JavaScript')}>JavaScript</button>
        <button className="language-button" onClick={() =>
handleLanguageSelect('Python')}>Python</button>
        <button className="language-button" onClick={() =>
handleLanguageSelect('Java')}>Java</button>
      </div>
    </div>
  );
};

const CodeInput = ({ language, onActionCompleted, task, onCheckAnswer,
onCodeExecution, onExit }) => {
  const [code, setCode] = useState('');
  const [feedback, setFeedback] = useState('');

  const handleCodeChange = (event) => {
    setCode(event.target.value);
  };

  const handleSubmit = (event) => {
    event.preventDefault();
    if (onCheckAnswer(code)) {
      setFeedback('Правильно! Молодець!');
      onActionCompleted();
      onCodeExecution(code); // Виконання коду
    } else {
      setFeedback('Ой, спробуй ще раз! Ось підказка: ' + task.hint);
    }
  };

  return (
    <div className="code-input">
      <h3>Ваш код:</h3>
      <textarea
        value={code}
        onChange={handleCodeChange}

```

```

        placeholder={`Напишіть ваш код на ${language}`}
        rows="6"
        cols="50"
      />
      <br />
      <button onClick={handleSubmit}>Виконати</button>
      <p>{feedback}</p>
      <button onClick={onExit}>Вийти з завдання</button>
    </div>
  );
};

const Robot = ({ action }) => {
  return (
    <div className={`robot ${action}`}>
      <div className="robot-head">
        <div className="robot-eyes">
          <div className="robot-eye"></div>
          <div className="robot-eye"></div>
        </div>
        <div className="robot-mouth"></div>
      </div>
      <div className="robot-body"></div>
      <div className="robot-arms">
        <div className="robot-arm left"></div>
        <div className="robot-arm right"></div>
      </div>
      <div className="robot-legs">
        <div className="robot-leg left"></div>
        <div className="robot-leg right"></div>
      </div>
    </div>
  );
};

const Game = () => {
  const [selectedLanguage, setSelectedLanguage] = useState(null);
  const [task, setTask] = useState('');
  const [action, setAction] = useState('');
  const [checkAnswer, setCheckAnswer] = useState(() => () => true);
  const [robotAction, setRobotAction] = useState('');
  const [score, setScore] = useState(0);

  const handleLanguageSelect = (language) => {
    setSelectedLanguage(language);
    loadTask(language);
  };

  const loadTask = (language) => {

```

```

let taskContent = '';
let hint = '';
let checkAnswerFn = () => true;
let correctAnswer = '';

switch (language) {
  case 'JavaScript':
    taskContent = 'Напиши код, щоб робот зробив три кроки вперед.';
    hint = 'Використай console.log для кожного кроку.';
    checkAnswerFn = (code) => {
      return code.trim().includes('console.log("Робот робить перший крок вперед!")') &&
        code.trim().includes('console.log("Робот робить другий крок вперед!")') &&
        code.trim().includes('console.log("Робот робить третій крок вперед!")');
    };
    break;
  case 'Python':
    taskContent = 'Допоможи роботу підняти руку!';
    hint = 'Використай метод raise_hand для роботи.';
    checkAnswerFn = (code) => code.trim().includes('robot.raise_hand()');
    break;
  case 'Java':
    taskContent = 'Допоможи роботу пригнути!';
    hint = 'Використай метод crouch для роботи.';
    checkAnswerFn = (code) => code.trim().includes('robot.crouch()');
    break;
  default:
    taskContent = 'Виберіть мову програмування!';
    hint = '';
    checkAnswerFn = () => false;
}

setTask({ content: taskContent, hint });
setCheckAnswer(() => checkAnswerFn);
};

const handleActionCompleted = () => {
  setRobotAction('jump');
  setScore(score + 10); // Додавання балів
  setTimeout(() => setRobotAction(''), 1000);
};

const executeCode = (code) => {
  if (code.includes('console.log("Робот робить перший крок вперед!")')) {
    setRobotAction('move-forward');
  } else if (code.includes('robot.crouch()')) {
    setRobotAction('jump');
  } else if (code.includes('robot.raise_hand()')) {

```

```

    setRobotAction('raise-hand');
  } else if (code.includes('robot.crouch();')) {
    setRobotAction('crouch');
  }
};

const handleExit = () => {
  setSelectedLanguage(null);
  setTask('');
  setAction('');
  setRobotAction('');
};

return (
  <div className="game">
    <h1>Поточний рахунок: {score} балів</h1>
    {selectedLanguage ? (
      <>
        <h1>Вибрано мову: {selectedLanguage}</h1>
        <p>{task.content}</p>
        <Robot action={robotAction} />
        <CodeInput
          language={selectedLanguage}
          onActionCompleted={handleActionCompleted}
          task={task}
          onCheckAnswer={checkAnswer}
          onCodeExecution={executeCode} // Виконання коду
          onExit={handleExit} // Кнопка для виходу
        />
      </>
    ) : (
      <LanguageSelection onLanguageSelect={handleLanguageSelect} />
    )}
  </div>
);
};

export default Game;

```

Додаток В

App.css

```

/* Підключення шрифту Montserrat */
@import
url('https://fonts.googleapis.com/css2?family=Montserrat:wght@400;500;600&display=swap');

/* Додаємо шрифт Montserrat до всіх елементів */

```

```

* {
  font-family: 'Montserrat', sans-serif;
}

/* Стили для секції "Про нас" */
.about-us {
  background-color: #fbec67;
  padding: 40px 20px;
  margin-top: 30px;
  border-radius: 10px;
}

.about-us-content {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.about-text {
  width: 60%; /* Ліва частина займає 60% */
  font-size: 18px;
  color: rgb(0, 0, 0);
  line-height: 1.6;
}

.about-text h2 {
  font-size: 28px;
  color: #000;
  margin-bottom: 20px;
}

.about-text p {
  margin-bottom: 15px;
}

.about-image {
  width: 35%; /* Права частина займає 35% */
}

.about-image img {
  width: 100%;
  height: auto;
  border-radius: 10px;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
}

/* Стили для тексту і картинки, щоб бути адаптивними */
@media (max-width: 768px) {
  .about-us-content {
    flex-direction: column; /* На маленьких екранах блоки йдуть один під одним */
  }
}

```

```

}

.about-text,
.about-image {
  width: 100%; /* Кожен блок займає 100% ширини */
  margin-bottom: 20px; /* Відступ між блоками */
}
}

/* Стили для хедера */
.header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 20px;
  background-color: #fff; /* Білий фон для хедера */
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1); /* Тінь під хедером */
}

.header .logo {
  font-size: 36px;
  font-weight: bold;
  color: #5351d4; /* Той самий синій колір для логотипу */
}

.header-menu a {
  color: #000000; /* Колір тексту */
  font-size: 20px; /* Розмір шрифту */
  text-decoration: none; /* Без підкреслення */
  font-weight: 300; /* Більша товщина тексту */
  letter-spacing: 1.5px; /* Відстань між літерами */
  margin-right: 20px; /* Відстань між елементами меню */
}

.header-menu a:hover {
  color: #130ff5; /* Колір при наведенні */
}

.header-menu .user-name {
  color: #000000 !important; /* Додаємо !important для переваги цього стилю */
  font-size: 24px;
  text-decoration: none;
  letter-spacing: 1.5px;
  font-weight: bold;
}

/* Контейнер для заголовка та кнопок */
.container {
  display: flex;

```

```

flex-direction: column; /* Вирівнювання елементів по вертикалі */
justify-content: center;
align-items: center;
text-align: center;
gap: 20px; /* Відстань між заголовком та кнопками */
}

/* Стиль для контейнера з кнопками */
.buttons-container {
  display: flex;
  justify-content: center; /* Центрує кнопки по горизонталі */
  gap: 10px; /* Відстань між кнопками */
  margin-top: 5px; /* Зменшено відстань між заголовком та кнопками */
}

/* Стиль для кнопок */
.buttons-container button {
  padding: 15px 30px; /* Збільшено відступи для більших кнопок */
  font-size: 18px; /* Збільшено розмір шрифту */
  cursor: pointer;
  background-color: #5351d4;
  color: #fff;
  border: none;
  border-radius: 5px;
  transition: background-color 0.3s ease;
}

/* Ефект при наведенні на кнопку */
.buttons-container button:hover {
  background-color: #130ff5; /* Колір при наведенні */
}

/* Заголовок */
.title {
  font-size: 24px; /* Розмір шрифту */
  font-weight: bold; /* Товстий шрифт */
  color: rgb(29, 28, 28); /* Колір тексту заголовка */
  margin-bottom: 10px; /* Зменшено відстань між заголовком та кнопками */
}

/* Стили для текстових полів та кнопок */
textarea {
  width: 100%; /* Повна ширина контейнера */
  height: 200px; /* Зменшено висоту для текстового поля */
  padding: 12px; /* Зменшено відступи */
  margin: 10px 0;
  border-radius: 8px; /* Збільшено радіус для більш округлених кутів */
  border: 1px solid rgb(255, 255, 255);
  font-size: 14px; /* Зменшено розмір шрифту */
  line-height: 1.5; /* Покращено читабельність тексту */
}

```

```

resize: vertical; /* Дозволено змінювати висоту вручну */
}

/* Стили для футера */
footer {
  background-color: #fff;
  padding: 20px;
  border-top: 1px solid #e0e0e0;
}

.footer-content {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.social-media a {
  margin-right: 15px;
  font-size: 16px;
  color: #5351d4;
  text-decoration: none;
}

.social-media a:hover {
  color: #130ff5;
}

.contact-info p {
  font-size: 16px;
  color: #777;
  margin: 0;
}

/* Стили для основного заголовка */
h1 {
  text-align: center;
  font-size: 36px;
  color: rgb(255, 255, 255);
  margin-top: 20px;
  margin-bottom: 30px;
}

/* Стили для основного заголовка */
.main-heading {
  font-size: 48px; /* Зменшено розмір шрифту */
  color: #5351d4;
  text-align: center;
  margin-top: 30px;
  font-weight: 400; /* Зменшено товщину шрифту */
  padding: 10px;

```

```

background-color: #f1f1f1;
border-radius: 8px;
}
.lesson p {
  text-align: center; /* Центруємо текст */
  font-size: 18px; /* За бажанням можна змінити розмір шрифту */
  color: rgb(255, 255, 255); /* Колір тексту */
}
/* Стили для секції відгуків */
.reviews {
  text-align: center;
  padding: 40px 20px;
  background-color: #f8f8f8;
  margin-top: 40px;
  border-radius: 10px;
}

.reviews h2 {
  font-size: 28px;
  color: #000;
  margin-bottom: 20px;
}

.reviews-list {
  display: flex;
  padding-bottom: 10px;
}

.review-item {
  background-color: #fff;
  padding: 20px;
  margin-right: 20px;
  border-radius: 8px;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
  min-width: 250px;
  text-align: left;
  font-size: 16px;
  color: rgba(51, 51, 51, 0.7); /* Напівпрозорий колір тексту */
}

.review-item p:first-child {
  font-style: italic;
  opacity: 0.7; /* Напівпрозорий текст у першому абзаці */
}

reviewer-info {
  display: flex;
  align-items: center;
  margin-bottom: 10px;
}

```

```

.reviewer-photo {
  width: 40px;
  height: 40px;
  border-radius: 50%;
  margin-right: 10px;
}

.reviewer-name {
  font-weight: bold;
  color: rgba(51, 51, 51, 0.7); /* Напівпрозорий колір імені рецензента */
}

/* Стили для островочків з рівнями складності */
.difficulty-levels {
  text-align: center;
  padding: 40px 20px;
  background-color: #f8f8f8;
  margin-top: 40px;
  border-radius: 10px;
}

.difficulty-levels h2 {
  font-size: 28px;
  color: #000;
  margin-bottom: 20px;
}

.levels-container {
  display: flex;
  justify-content: center;
  align-items: center;
}

.level-item {
  display: flex;
  align-items: center;
  position: relative;
  margin: 0 20px;
}

.level-circle {
  width: 100px;
  height: 100px;
  border-radius: 50%;
  background-color: #5351d4;
  display: flex;
  justify-content: center;
  align-items: center;
  font-weight: bold;

```

```

color: #fff;
font-size: 36px; /* Збільшено розмір для зірочок */
font-family: 'Montserrat', sans-serif;
box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.1);
transition: background-color 0.3s ease, transform 0.3s ease; /* Додаємо плавний
ефект зміни кольору */
}

/* Зірочка, яка додається перед кожним островком */
.level-circle::before {
  content: '★'; /* Зірочка для кожного островка */
}

/* Зміна кольору островочка при наведенні */
.level-circle:hover {
  background-color: #130ff5; /* Змінюємо колір фону на синій при наведенні */
  transform: scale(1.1); /* Збільшуємо розмір островка при наведенні */
}

.level-item span {
  margin-left: 10px;
  font-size: 16px;
  color: #333;
}

.dot-separator {
  width: 8px;
  height: 8px;
  border-radius: 50%;
  background-color: #b0b0b0;
  margin: 0 10px;
}

/* Стили для модального вікна */
.modal {
  position: fixed;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  background-color: #f0f4f8; /* Світло-блакитний фон для вікна */
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  z-index: 1000;
  width: 300px; /* Ширина квадратного вікна */
  height: 300px; /* Висота квадратного вікна */
  text-align: center;
  font-family: 'Arial', sans-serif;
  color: #333;
  border: 2px solid #000; /* Чорна рамка */
  transition: all 0.3s ease;

```

```

}

/* Фон для затемнення */
.modal-overlay {
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background-color: rgba(0, 0, 0, 0.5); /* Темний фон */
  z-index: 999;
}

/* Стили для форм і тексту всередині модалки */
.modal h2 {
  font-size: 20px;
  margin-bottom: 15px;
  font-weight: bold;
  color: #000; /* Чорний колір для заголовку */
  text-align: center;
}

.modal form {
  display: flex;
  flex-direction: column;
  gap: 12px;
}

/* Поля вводу */
.modal input {
  padding: 10px;
  font-size: 14px;
  border: 1px solid #ccc;
  border-radius: 5px;
  transition: border-color 0.3s ease;
}

.modal input:focus {
  border-color: #5351d4; /* Синя рамка при фокусуванні */
  outline: none;
}

/* Кнопка */
.modal button {
  padding: 10px;
  font-size: 14px;
  background-color: #5351d4; /* Синій фон для кнопки */
  color: white;
  border: none;
}

```

```

border-radius: 5px;
cursor: pointer;
transition: background-color 0.3s ease;
}
.modal button:hover {
  background-color: #130ff5; /* Кольор при наведенні */
}

/* Кнопка закриття */
.modal-close {
  position: absolute;
  top: 10px;
  right: 10px;
  background: none;
  border: none;
  font-size: 20px;
  color: #5351d4;
  cursor: pointer;
  transition: transform 0.3s ease, color 0.3s ease;
}

/* Кнопка закриття при наведенні */
.modal-close:hover {
  color: #130ff5;
  transform: rotate(90deg); /* Додаємо ефект обертання */
}

/* Кнопка закриття при активному натисканні */
.modal-close:active {
  transform: scale(0.9); /* Зменшуємо розмір при натисканні */
}

/* Кнопка закриття */
.modal-close {
  position: absolute;
  top: 10px;
  right: 10px;
  background: none;
  border: none;
  font-size: 20px; /* Зменшено розмір шрифту */
  color: #5351d4;
  cursor: pointer;
  padding: 5px; /* Додано відступи для зменшення розміру кнопки */
  width: 30px; /* Встановлено ширину для пропорційності */
  height: 30px; /* Встановлено висоту для пропорційності */
  display: flex;
  justify-content: center;
  align-items: center;
  border-radius: 50%; /* Округлі кути для створення кнопки у формі кола */
}

```

```

.modal-close:hover {
  color: #130ff5;
}

/* Додаємо символ × для кнопки закриття */
.modal-close::before {
  content: "x"; /* Символ хрестика */
  font-size: 18px; /* Зменшений розмір шрифту для хрестика */
  color: inherit; /* Наслідує колір від modal-close */
}

/* Для мобільних пристроїв */
@media (max-width: 480px) {
  .modal {
    width: 85%; /* Ще менша ширина для мобільних пристроїв */
    height: 85%; /* Висота також адаптована */
    padding: 15px;
  }
}

.correct-answer {
  color: green;
  font-weight: bold;
}

.incorrect-answer {
  color: red;
  font-weight: bold;
}

.robot {
  display: inline-block;
  position: relative;
  width: 100px;
  height: 150px;
  background-color: #ccc;
  border-radius: 10px;
  transition: all 0.5s ease-in-out;
}

.robot-head {
  width: 100%;
  height: 30px;
  background-color: #444;
  border-radius: 50%;
}

.robot-eyes {
  display: flex;

```

```
    justify-content: space-evenly;
    margin-top: 10px;
}

.robot-eye {
    width: 10px;
    height: 10px;
    background-color: white;
    border-radius: 50%;
}

.robot-mouth {
    width: 60%;
    height: 5px;
    background-color: white;
    margin: 5px auto;
}

.robot-body {
    width: 70%;
    height: 60%;
    background-color: #555;
    margin: 10px auto;
}

.robot-arms {
    display: flex;
    justify-content: space-between;
    position: absolute;
    top: 30px;
    width: 100%;
}

.robot-arm {
    width: 20px;
    height: 40px;
    background-color: #444;
    border-radius: 5px;
}

.robot-arm.left {
    transform: rotate(45deg);
    transform-origin: top left;
}

.robot-arm.right {
    transform: rotate(-45deg);
    transform-origin: top right;
}
```

```

.robot-legs {
  display: flex;
  justify-content: space-between;
  width: 100%;
}

.robot-leg {
  width: 20px;
  height: 40px;
  background-color: #444;
  border-radius: 5px;
}

.robot.move-forward {
  animation: moveForward 1s forwards;
}

.robot.jump {
  animation: jump 1s forwards;
}

@keyframes moveForward {
  0% {
    transform: translateX(0);
  }
  100% {
    transform: translateX(100px);
  }
}

@keyframes jump {
  0% {
    transform: translateY(0);
  }
  50% {
    transform: translateY(-50px);
  }
  100% {
    transform: translateY(0);
  }
}

.robot.raise-hand .robot-arm.right {
  animation: raiseHand 1s forwards;
}

@keyframes raiseHand {
  0% {
    transform: rotate(0deg);
  }
}

```

```

100% {
  transform: rotate(-90deg); /* Підніmemo руку на 90 градусів */
}
}

.language-selection {
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  height: 100vh;
  text-align: center;
}

.button-container {
  display: flex;
  gap: 20px;
  margin-top: 20px;
}

.language-button {
  background-color: #5351d4;
  color: white;
  padding: 10px 20px;
  font-size: 16px;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.language-button:hover {
  background-color: #130ff5;
}

.language-button:focus {
  outline: none;
}

/* Стили для впливаючого блоку контактів */
.contact-dropdown {
  position: absolute;
  top: 50px;
  right: 0;
  background-color: #fff;
  padding: 10px 20px;
  border-radius: 5px;
  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
  z-index: 1000;
  opacity: 0;
}

```

```

visibility: hidden;
transition: opacity 0.3s ease, visibility 0.3s ease;
}

.contact-dropdown.show {
  opacity: 1;
  visibility: visible;
}

.contact-dropdown p {
  margin: 0;
  font-size: 14px;
  color: #333;
}

.contact-dropdown p + p {
  margin-top: 10px;
}

/* Стили для меню при наведенні */
.header-menu a:hover {
  color: #5351d4;
  text-decoration: underline;
}

a#contacts {
  position: relative;
}

a#contacts:hover + .contact-dropdown {
  opacity: 1;
  visibility: visible;
}

.highlight {
  background-color: #f0f8ff; /* Колір підсвічування */
  color: #5351d4; /* Колір тексту при підсвічуванні */
  padding: 40px 20px;
  margin-top: 30px;
  border-radius: 10px;
}

.about-us.highlight {
  background-color: #f6e125; /* Підсвічуємо жовтим */
  transition: background-color 0.5s ease-out; /* Плавне згасання */
}

.main-heading {
  font-size: 2.5rem; /* Збільшуємо розмір шрифту */
  font-weight: bold; /* Робимо текст жирним */
}

```

```
color: #333; /* Темно-сірий колір для тексту */
text-align: center; /* Вирівнюємо текст по центру */
margin-top: 50px; /* Відступ знизу від попереднього елемента */
margin-bottom: 20px; /* Відступ знизу */
text-transform: uppercase; /* Перетворюємо текст на великі літери */
letter-spacing: 2px; /* Додаємо відстань між літерами */
line-height: 1.2; /* Збільшуємо міжрядковий інтервал */
font-family: 'Montserrat', sans-serif; /* Вибираємо сучасний шрифт */
position: relative; /* Для додавання додаткових ефектів */
}

/* Додатковий стиль для підсвічування заголовка при наведенні */
.main-heading:hover {
  color: #5351d4; /* Колір при наведенні (синій) */
  text-shadow: 2px 2px 5px rgba(0, 0, 0, 0.1); /* Легке тіньове підсвічування */
  transition: color 0.3s ease, text-shadow 0.3s ease; /* Плавний перехід */
}
```