Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

НТУ «Дніпровська Політехніка»
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Кречуняк Євгеній Альбертович*
(ПІБ)

академічної групи *122-20ск*
(шифр)

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

освітньої програми *Комп'ютерні науки*
(назва освітньої програми)

на тему: Розробка онлайн-сервісу для вивчення української мови

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи				
розділів:				
спеціальний	*Кабак Л.В.*			
економічний	*проф. Вагонова О.Г.*			
Рецензент	*доц. Каштан В.Ю.*			
Нормоконтролер	*доц. Гуліна І.Г.*			

Дніпро
2023

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ М.О. Алексєєв
(підпис) (прізвище, ініціали)

« » _____ 2023 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *122-20ск* *Кречуняк Є.А.*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи Розробка онлайн-сервісу для вивчення української мови

затверджена наказом ректора НТУ «ДП» від *16.05.2023* № *350-с*

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів проєктно-технологічної практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі. Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***13.05.2023 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***27.05.2023 р.***

Завдання видав _____ *Кабак Л.В.*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Кречуняк Є.А.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *14.01.2023 р.*

Термін подання кваліфікаційної роботи до ЕК: *3,07.2023 р.*

РЕФЕРАТ

Пояснювальна записка: 110 с., 33 рис., 5 дод., 11 джерел.

Об'єкт розробки: Розробка онлайн-сервісу для вивчення української мови.

Мета кваліфікаційної роботи: розробити веб-додаток, який буде виступати в якості онлайн-сервісу для інтерактивного проходження уроків з української мови.

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточнюється постановка завдання.

У першому розділі проведено аналіз предметної області, визначено актуальність завдання та призначення розробки, розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі виконано аналіз існуючих рішень, обрано вибір мов для розробки, виконано розробку веб-додатку та його проєктування, наведено опис алгоритму і структури функціонування системи, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження застосунку, описана робота веб-сайту.

В економічному розділі визначено трудомісткість розробленої інформаційної підсистеми, проведений підрахунок вартості роботи по створенню онлайн-сервісу та розраховано час на його створення.

Список ключових слів: ВЕБ-ДОДАТОК, ОНЛАЙН-СЕРВІС, HTML, CSS, SASS, TYPESCRIPT, JAVASCRIPT, PHP, JSON, MYSQL, DATABASE, PHPMYADMIN, LOCALSTORAGE, DOM.

ABSTRACT

Explanatory note: 110 pages, 33 pics, 5 apps, 11 sources.

Object of development: Development of an online-service for learning the Ukrainian language.

The purpose of the diploma project: to develop a web-application that will act as an online-service for interactive lessons on the Ukrainian language.

The introduction considers the analysis and the current state of the problem, specifies the purpose of the qualification work and the field of its application, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the analysis of the subject area is carried out, the urgency of the task and purpose of development is defined, the statement of the task is developed, requirements to software realization, technologies and software are set.

The second section analyzes the existing solutions, selects the platform for development, performs design and development of the web-application, describes the algorithm and structure of the system, determines the input and output data, provides the characteristics of the parameters of website, describes the call and application load, describes the website.

In the economic section, the complexity of the developed information subsystem is determined, the cost of work on creating an online-service is calculated and the time for its creation is calculated.

List of keywords: WEB-APPLICATION, ONLINE-SERVICE, HTML, CSS, SASS, TYPESCRIPT, JAVASCRIPT, PHP, JSON, MYSQL, DATABASE, PHPMYADMIN, LOCALSTORAGE, DOM.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

HTML – HyperText Markup Language (Мова розмітки гіпертексту).

CSS – Cascading Style Sheets (Каскадні таблиці стилів).

SASS - Syntactically Awesome Style Sheets (метамова на основі CSS, призначена для збільшення рівня абстракції CSS-коду та спрощення файлів каскадних таблиць стилів).

TYPESCRIPT – Programming language (мова програмування, представлена як засіб розробки веб-додатків, що розширює можливості JavaScript).

PHP – Hypertext Preprocessor (Препроцесор гіпертексту).

MySQL – Structured Query Language (СКБД).

HTTP – HyperText Transfer Protocol (Протокол передачі гіпертексту).

API – Application Programming Interface (Інтерфейс програмування додатків).

W3C – World Wide Web Consortium (перелік технологічних стандартів для Всесвітнього павутиння).

DOM – Document Object Model (об’єктна модель документа).

ВСТУП

Нескінченний відкритий доступ до глобальної мережі даних кожного дня значно спрощує та покращує життя кожної людини у сучасному світі.

Завдяки цьому, люди мають змогу розвиватися у тих сферах діяльності, в яких вони хочуть, будь-то робота чи навчання, самовираження або розважання.

Інтернет дозволяє людям з усього світу легко спілкуватися між собою. За допомогою соціальних мереж, електронної пошти, месенджерів та відео-дзвінків, ми можемо знаходитися на зв'язку зі своїми родичами, друзями та колегами в будь-якому куточку світу. Це розширює наші можливості для спілкування, обміну думками та ідеями, а також сприяє взаєморозумінню та культурному обміну. Тому зараз як ніколи виникає потреба у розробці онлайн-сервісів для вивчення мов спілкування.

Мова має величезний вплив на життя людини. Вона не лише є засобом спілкування, але й передає культуру, традиції та історію народу. Мова формує наше сприйняття світу, мислення, емоції, поведінку та взаємодію з іншими людьми.

Вона є не тільки засобом комунікації, але й інструментом для розвитку наукових досліджень та технічного прогресу. Багато сучасних технологій та інноваційні ідеї були здійснені завдяки співпраці між науковцями та фахівцями з різних країн, що здійснювали комунікацію на різних мовах.

Мова також впливає на соціальну інтеграцію людини в суспільство, її культурну та економічну діяльність. Навички володіння іноземними мовами допомагають людині отримати кращі можливості в кар'єрному рості, подорожувати та брати участь у міжнародних проектах.

Вивчення мови в сучасному житті є надзвичайно важливим з наступних причин:

- Міжнародна комунікація: у світі, де глобалізація стає все більш помітною, вміння спілкуватися на іноземній мові стає ключовим;
- Академічні та професійні переваги: знання мови допомагає збільшити шанси на успіх у навчанні, розширити кругозір та забезпечити конкурентоспроможність на ринку праці;
- Розвиток когнітивних навичок: завдяки навчанню мови, у людини розвивається критичне мислення, зосередженість, пам'ять та творчість;
- Особисті задоволення: вивчення мови може стати джерелом задоволення та розваги.

Метою кваліфікаційної роботи є створення онлайн-сервісу, який буде надавати усі необхідні можливості для того, щоб кожна бажаюча людина змогла навчитися та опанувати рідну українську мову.

Підбиваючи підсумки, було сформовано тему кваліфікаційної роботи: «Розробка онлайн-сервісу для вивчення української мови».

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Загальні відомості з предметної галузі

Основний принцип роботи онлайн-сервісів з вивчення будь-якої мови полягає в поєднанні різноманітних методів та інструментів, що сприяють продуктивному навчанню мови через Інтернет. Основні елементи, на яких ґрунтується робота таких сервісів, можуть включати:

- Навчальний матеріал: клієнту надають доступ до різноманітного навчального матеріалу, який може включати як текстові, так й відео-уроки, які містять в собі аудіо-файли, граматичні правила, словники та інші ресурси. Цей матеріал зазвичай структурований за рівнями складності та тематикою, щоб надати користувачам послідовний прогрес у вивченні мови.
- Вправи та тести: онлайн-сервіси надають клієнтам різноманітні завдання для практики навичок читання, письма, говоріння та слухання. Ці завдання можуть включати виконання завдань з вибором відповіді, заповнення пропусків, переклад, вимову та інше. Також, деякі сервіси використовують інтерактивні вправи, диктанти та ігри для забезпечення цікавого та захоплюючого навчання.
- Система прогресу: працюючи з онлайн-сервісом, користувачі можуть зручно відстежувати свої досягнення, бачити, які навички вони вже вивчили, і визначати області, в яких їм слід ще працювати. Завдяки звітам та статистиці, що надає онлайн-сервіс, клієнт може визначитися з подальшим планом свого навчання.
- Інтерактивність: онлайн-сервіси намагаються створити інтерактивне навчальне середовище, в якому користувачі можуть проходити вправи з вимови, створювати відео чи аудіо-записи, для подальшого порівняння

свого голосу з носіями мови, отримання коментарів та пояснень від вчителів або інших користувачів.

- Допомога та зворотній зв’язок: онлайн-сервіси надають усі необхідні пояснення та настанови з приводу того, як правильно працювати з навчальним середовищем сервісу, для того щоб отримувати весь максимум з навчання.

Ці елементи поєднуються з метою створення привабливого та ефективного середовища для навчання мови, що дозволяє користувачам отримувати знання, практикувати навички та вдосконалювати свою мову в зручний для них час і місце.

Першим відомим онлайн-сервісом для вивчення мови був сервіс LiveMocha, створений у 2007 році парою інженерів - Беном Хасселтоном і Джі Ти. Головний офіс компанії знаходиться у США, Сіетл.

LiveMocha був популярним веб-сайтом, що надавав можливість вивчати різні мови та спілкуватися з носіями мови з усього світу. Сервіс пропонував навчальні матеріали, вправи, оцінювання та можливості спілкування з іншими користувачами. У 2013 році LiveMocha був придбаний компанією Rosetta Stone, а пізніше його функціонал був інтегрований у продукт Rosetta Stone. На теперішній стан, сервіс має понад 8,5 мільйонів членів зі 195 країн та пропонує безплатні курси вивчення 38 мов та додаткові платні курси з 5 мов. Курси LiveMocha містять базові дані з граматики, викладені у прикладах, вправи на читання та розуміння, інтерактивні рольові ігри (діалоги), письмові та усні вправи, тести тощо. Вивчення мов базується на взаємній допомозі користувачів один одному.

Рисунок 1.1 - загальний вигляд сервісу LiveMocha

Однак найбільшої популярності та оцінки отримав зовсім інший сервіс під назвою Duolingo.

Duolingo - це популярний онлайн-сервіс для вивчення мови, який пропонує інтерактивні уроки та вправи. Він був розроблений Луїсом фон Аном і Севастіаном Тельфайром (рисунок 1.2) у 2011 році з метою створення безкоштовної платформи для навчання мови, яка б поєднувала навчання та гральний процес.

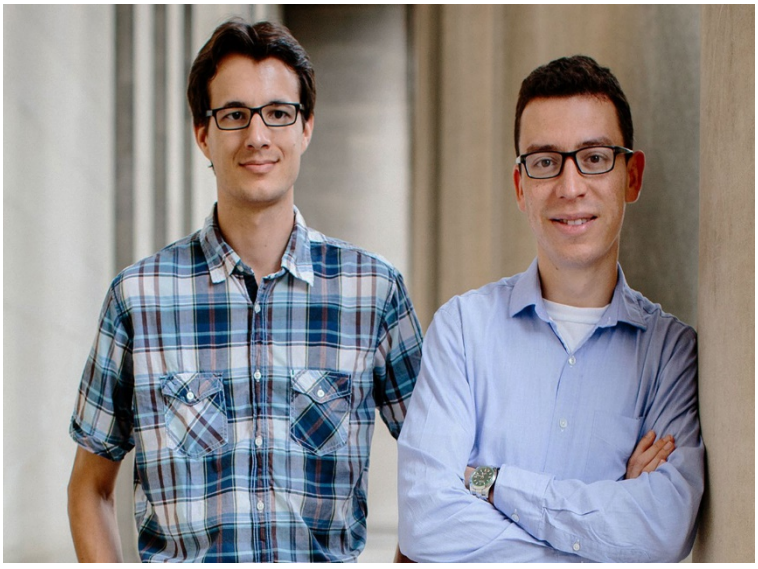

Рисунок 1.2 - засновники Duolingo

Duolingo пропонує навчання багатьох популярних мов, таких як англійська, іспанська, французька, німецька, італійська та багато інших. Сервіс використовує групу різних методів навчання, включаючи читання, письмо, прослуховування та говоріння, щоб розвивати всі аспекти мовлення. Сервіс використовує систему гейміфікації, де користувачі отримують бали, рівні та досягнення за успішне виконання вправ. Він також має вбудовану систему повторення, яка допомагає закріплювати вивчений матеріал через періодичне повторення попередніх уроків.

Однією з особливостей Duolingo є його безкоштовна модель, яка дозволяє вивчати мову без необхідності оплачувати підписку. Проте, також доступна платна версія, Duolingo Plus, яка пропонує додаткові переваги, такі як відсутність реклами та можливість використовувати сервіс офлайн.

Сьогодні Duolingo є лідером у сфері сервісів з онлайн-навчання мов, який надає широкі можливості усім користувачам, які бажають ознайомитися з новою мовою та розвинути свої навички мовлення.

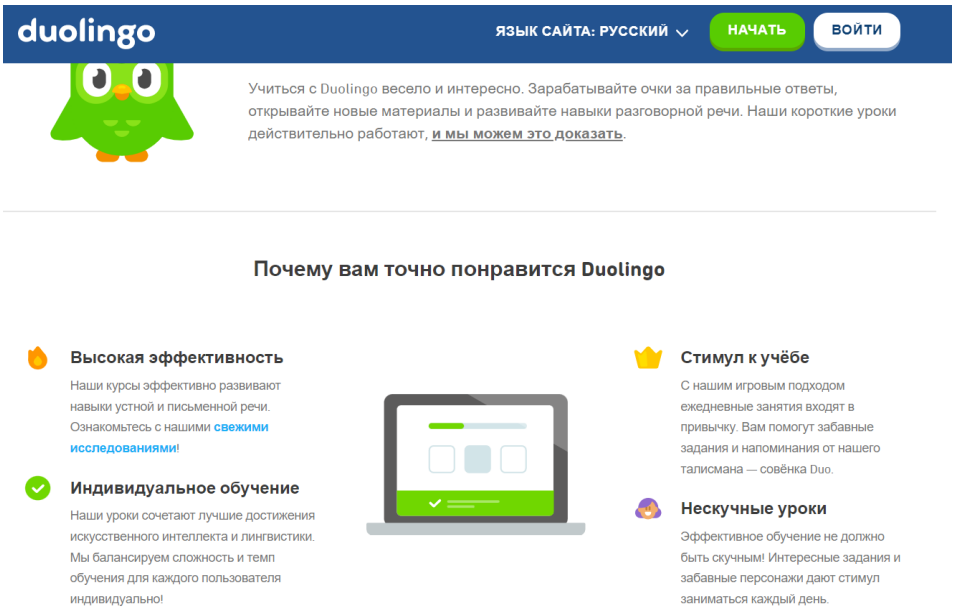

Рисунок 1.3 - загальний вигляд сервісу Duolingo

Не менш важливим і ключовим елементом при побудові веб-додатку є грамотне створення інтерфейсу для візуалізації.

Інтерфейс - це механізм, який дозволяє користувачеві комунікувати з програмним забезпеченням, вводити дані, отримувати результати і керувати його функціоналом. Він може бути реалізований у різних формах, основними з яких є:

- Графічний інтерфейс користувача (GUI) - надає візуальний зв'язок з програмою за допомогою графічних об'єктів, таких як кнопки, меню, поля вводу, вкладки тощо. Користувач може взаємодіяти з програмою, використовуючи мишу, клавіатуру або сенсорний екран;
- Текстовий інтерфейс командного рядка (CLI) - дозволяє користувачу вводити команди або запити у вигляді тексту. Користувач взаємодіє з програмою, набираючи команди і отримуючи результати у вигляді текстового виводу;
- Веб-інтерфейси - надають доступ до програмного забезпечення через веб-браузер. Вони можуть включати різні елементи, такі як форми, кнопки, посилання, таблиці тощо;
- Голосові інтерфейси - дозволяють користувачеві взаємодіяти з програмним забезпеченням за допомогою голосових команд і відповідей. Такі інтерфейси все більше набувають популярності у голосових помічниках та інших додатках для розпізнавання мови.

Важливим аспектом інтерфейсу є його вигляд, організація елементів та способи взаємодії, які зрозумілі, зручні та ефективні для користувача. Добре розроблений інтерфейс допомагає полегшити використання програмного забезпечення та покращити користувацький досвід.

Для створення інтерфейсу веб-додатків існує багато графічних редакторів та інструментів. Найпоширенішими є:

- Adobe Photoshop - photoshop є одним з найпопулярніших графічних редакторів, який широко використовується для проектування веб-інтерфейсів. Він надає багатий набір інструментів для створення макетів, дизайну кнопок, іконок, фонів та інших елементів інтерфейсу;
- Sketch - sketch є популярним графічним редактором, спеціально розробленим для дизайну інтерфейсів. Він має інтуїтивний інтерфейс, широкі можливості для створення векторної графіки, символів, компонентів та багато інших функцій, спрямованих на роботу з веб-дизайном;
- Figma - figma є онлайн-інструментом для дизайну інтерфейсу, який дозволяє розробникам та дизайнерам спільно працювати над проектами в реальному часі. Він має потужні можливості для створення макетів, прототипів, а також для спільного взаємодії з командою та замовниками;
- Adobe XD - adobe XD є інструментом для дизайну та прототипування веб-додатків і мобільних додатків. Він має інтуїтивний інтерфейс, можливості для швидкого створення макетів, взаємодії елементів та перетворення їх в функціональні прототипи;
- InVision - inVision є інструментом для створення і презентації прототипів веб-додатків. Він дозволяє створювати інтерактивні макети, збирати зворотній зв'язок від команди та клієнтів, а також забезпечує можливість спільної роботи та взаємодії.

Це лише кілька прикладів графічних редакторів та інструментів, які можна використовувати для створення інтерфейсу для програмного забезпечення. Вибір конкретного інструменту залежить від особистих вподобань, рівня досвіду та вимог проекту.

При розробці дизайну веб-інтерфейсу кваліфікаційної роботи був обраний онлайн-інструмент Figma.

1.2 Призначення розробки та галузь застосування

В якості бакалаврської кваліфікаційної роботи виступає тема «розробка онлайн-сервісу для вивчення української мови». Функціональна основа будується завдяки мовам програмування TypeScript та PHP. Головною метою роботи є створення веб-додатку, який є сервісом для інтерактивного проходження уроків з української мови.

Головними перевагами розроблювального веб-сайту є:

- ✓ Зручний та зрозумілий інтерфейс;
- ✓ Гнучкість та доступність;
- ✓ Різноманітність навчальних матеріалів;
- ✓ Можливість самостійного навчання;
- ✓ Наявність інтерактивних інструментів.

Веб-додаток призначений для:

- ✓ Створення середовища для вивчення української мови;
- ✓ Взаємодія з функціональними можливостями середовища;
- ✓ Збереження будь-яких змін у середовищі.

Розроблювальна система описується як веб-додаток, що надає можливість проходити уроки з української мови онлайн у будь-який час.

1.3 Підстави для розробки

Згідно з навчальним планом та графіком навчального процесу, а також зі стандартами освітньо-кваліфікаційної характеристики(ОКХ) й освітньо-професійної програми(ОПП), наприкінці навчання, кожен студент має виконати підсумкову кваліфікаційну роботу у вигляді кваліфікаційної роботи, тема якої насамперед узгоджується з керівником проекту від кафедри, а потім затверджується наказом ректора.

Перелік підстав для розробки проекту:

- Навчальний план;
- Графік навчального процесу;
- ОПП 122 напрямку “Комп’ютерні науки”;
- Кваліфікаційна робота на тему “Розробка онлайн-сервісу для вивчення української мови”.

1.4 Постановка завдання

Розроблювальна система онлайн-сервісу призначена для того, щоб допомогти кожній бажаючий людині ефективно удосконалити рівень володіння українською мовою. Тобто, додаток повинен виконати реалізацію таких дій:

- Можливість входу чи реєстрації користувача;
- Можливість взаємодії з елементами(уроками) веб-додатку;
- Надати можливість переглядати раніше пройдений матеріал за допомогою спеціальних інтерактивних інструментів;
- Виконання CRUD-операцій;
- Збереження усіх необхідних змін до БД.

Для якісного виконання проекту, були виконані наступні дії:

- Переглянуто вже існуючі рішення для розробки онлайн-сервісів для навчання мов спілкування;
- Виконано проектування архітектури додатку;
- Розроблено веб-дизайн додатку;
- Розроблено базу даних для системи;
- Виконано Front-End та Back-End частини веб-додатку.

1.5. Вимоги до програми або програмного виробу.

1.5.1. Вимоги до функціональних характеристик.

Розроблювальна програма повинна виконувати наступні вимоги:

- Коректне відображення усіх візуальних елементів сайту;
- Коректна взаємодія з елементами сайту;
- Збереження усіх змін згідно з діями користувача.

Для того, щоб усі функціональні вимоги якісно працювали, додаток має налічувати:

- Браузерну підтримку;
- Програмну та апаратну сумісність.

1.5.2 Вимоги до інформаційної безпеки.

Програмна система повинна включати наступні аспекти інформаційної безпеки:

- Наявність механізмів перевірки якісної аутентифікації та авторизації користувачів сайту;
- Коректна екранізація та очищення вхідних даних;
- Перевірка SQL-запитів до БД на коректність;
- Створювання резервної копії даних сайту у разі втрати;
- Зберігання усіх необхідних даних у локальному середовищі;
- Захист даних сайту та користувачів від несанкціонованого доступу.

1.5.3 Вимоги до складу та параметрів технічних засобів.

Обчислювальна машина, на якій буде запускатися веб-додаток, повинна мати наступні характеристики для надійної роботи:

- Клавіатура;

- Миша або тачпад;
- Монітор;
- Наявність інтернет-зв’язку;
- Не менш ніж 1 Гб місця на жорсткому диску;
- Процесор Intel Pentium 4 з підтримкою SSE3;
- Не менш ніж 4 Гб оперативної пам’яті.

Перелічені вище характеристики, в цілому, є мінімально-рекомендованими для того, щоб програмне забезпечення коректно функціонувало. У випадку невідповідності заданим критеріям, веб-додаток не зможе працювати.

1.5.4 Вимоги до інформаційної та програмної сумісності.

Програмне забезпечення машини, на якій буде експлуатуватися веб-додаток, повинно відповідати наступним вимогам програмної сумісності:

- Наявність веб-браузеру, наприклад Internet-Explorer, Microsoft Edge, Firefox, Opera чи Google Chrome;
- Операційна система Windows 7/10/11.

Функціонал розроблювального веб-додатку має бути реалізований використовуючи наступні засоби:

- Мова програмування TypeScript для Front-End частини;
- Каскадні таблиці стилів(Css) з використанням технології Sass для візуальної складової сайту;
- Мова програмування Sql для створення бази даних веб-додатку;
- Мова програмування Php для Back-End частини.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Функціональне призначення системи

Під час виконання кваліфікаційної роботи було розроблено веб-додаток, а саме онлайн-сервіс для вивчення української мови. Головною метою розробки була можливість інтерактивного проходження уроків з української мови користувачем.

Призначення розробленого інтернет-магазину:

− Надання можливості реєстрації аккаунту користувачем;

− Надання можливості перегляду переліку уроків;

− Надання можливості обрання та проходження уроку;

− Збереження усіх даних та усього прогресу користувача;

− Візуальне оновлення інтерфейсу веб-додатку згідно з прогресом користувача;

− Надання можливості звернення до підтримки онлайн-сервісу.

Для досягнення всього перерахованого функціоналу, веб-додаток повинен мати:

− Сумісність зі стандартною апаратною конфігурацією обчислювальної машини;

− Операційну систему Windows 11/10/7.

− Підтримку сучасних браузерів, таких як Google Chrome, Mozilla Firefox чи Microsoft Edge.

2.2 Опис застосованих математичних методів

При розробці веб-додатку не використовувалися ніякі математичні методи, так як особливості предметної області не передбачають використання та застосування математичних методів.

2.3 Опис використаних технологій та мов програмування

При розробці онлайн-сервісу було використано декілька мов програмування. Серверна частина веб-додатку була спроектована та розроблена за допомогою СКБД MySQL на базі PhpMyAdmin. Програмна частина додатку була написана на мові TypeScript та PHP. Візуальна складова була написана використовуючи препроцесор SASS та CSS.

Опис мови програмування TypeScript.

TypeScript - це мова програмування, яка є розширенням JavaScript. Вона надає статичну типізацію, що дозволяє розробникам виявляти помилки під час компіляції коду. TypeScript компілюється до чистого JavaScript, тому його можна використовувати на будь-якому середовищі, де підтримується JavaScript.

TypeScript додаває до JavaScript декілька нових функцій, які полегшують розробку складних програм. Основні особливості TypeScript включають:

1) Статична типізація: TypeScript дозволяє оголошувати типи для змінних, параметрів функцій та повертаємих значень. Це допомагає виявляти помилки на етапі компіляції і полегшує розуміння коду;
2) Підтримка об'єктно-орієнтованого програмування: TypeScript має підтримку класів, спадкування, інтерфейсів, модифікаторів доступу і багато іншого. Це дозволяє розробникам створювати більш структурований і модульний код;

3) Екосистема JavaScript: TypeScript є сумісним з існуючими бібліотеками та фреймворками JavaScript. Це означає, що розробники можуть використовувати наявний код JavaScript і поступово переходити до TypeScript, додаючи типізацію і інші переваги мови;
4) Розширені можливості редактора: TypeScript надає багатий набір інструментів для підказок і автодоповнення коду у редакторах, таких як Visual Studio Code. Це полегшує розробку, покращує продуктивність і допомагає уникнути помилок.

TypeScript є дуже популярним у веб-розробці, особливо в проектах на базі фреймворків, таких як Angular, React і Vue.js. Він дозволяє створювати більш безпечний і підтримуваний код, сприяючи швидшому розробленню і підтримці великих проектів.

Основними етапами обробки TypeScript-коду є наступні кроки:

1. Написання коду: Розробник створює програмний код на мові TypeScript. TypeScript використовує синтаксис, подібний до JavaScript, з додатковими можливостями типізації та функціональними можливостями;
2. Компіляція: Файли з розширенням .ts або .tsx (якщо використовується JSX) компілюються в чистий JavaScript. Компілятор TypeScript (tsc) перетворює код TypeScript у валідний JavaScript, зберігаючи типи, інтерфейси та інші TypeScript-конструкції, які можуть бути використані на етапі розробки, але не підтримуються самим JavaScript;
3. Виявлення помилок: Під час компіляції, компілятор TypeScript перевіряє код на наявність помилок типізації та інших потенційних проблем. Він надає діагностичні повідомлення про виявлені помилки, які допомагають розробнику виправити проблеми;
4. Генерація JavaScript: Після успішної компіляції, результатом є один або кілька файлів з розширенням .js, які містять вихідний код

JavaScript. Ці файли можуть бути використані в будь-якому середовищі, що підтримує JavaScript;

5. Виконання коду: Згенерований JavaScript-код може бути виконаний в будь-якому середовищі, що підтримує JavaScript, такому як веб-браузер або Node.js. Виконання коду TypeScript немає жодних особливостей порівняно з виконанням звичайного JavaScript-коду.

Ці етапи показують, що TypeScript використовується для створення типізованого коду на етапі розробки, який потім компілюється в чистий JavaScript для виконання в браузерах або на сервері. Компіляція TypeScript до JavaScript дозволяє розробникам використовувати багатий набір інструментів та функцій TypeScript, але забезпечує сумісність з існуючим екосистемою JavaScript.

Опис мови програмування PHP.

PHP (Hypertext Preprocessor) є скриптовою мовою програмування загального призначення, яка використовується для розробки веб-додатків і динамічних веб-сторінок. Вона була спеціально створена для обробки сторінок на стороні сервера, але також може використовуватися як мова програмування загального призначення.

PHP є однією з найпопулярніших мов програмування для розробки веб-додатків, особливо для динамічних сторінок, які взаємодіють з базами даних. Вона працює на більшості веб-серверів і здатна генерувати HTML-сторінки на льоту, коли користувач запитує веб-сторінку.

PHP має простий і легкий для вивчення синтаксис, що дозволяє швидко створювати динамічні веб-сторінки і розробляти складні веб-додатки. Вона підтримує широкий спектр функцій, включаючи роботу з базами даних, роботу з файлами, обробку форм, роботу з мережевими протоколами і багато іншого.

PHP може бути вбудована безпосередньо в HTML-код або використовуватися в поєднанні з іншими технологіями веб-розробки, такими як

CSS і JavaScript. Багато відомих систем управління вмістом (CMS), таких як WordPress і Joomla, розроблені з використанням PHP.

Однією з переваг PHP є велика спільнота розробників і багата документація, яка допомагає вирішувати проблеми і навчатися новим технікам.

PHP є скриптовою мовою програмування, тому процес його виконання відрізняється від традиційних компільованих мов, таких як C++ чи Java.

Основні етапи обробки PHP-коду включають наступні кроки:

1) Сервер отримує запит: Коли веб-сторінка з PHP-кодом запитується клієнтом (наприклад, веб-браузером), веб-сервер отримує цей запит;
2) Виявлення PHP-файлу: Веб-сервер перевіряє розширення файлу запиту та визначає, чи є цей файл PHP-файлом. Якщо це так, веб-сервер знає, що потрібно обробити PHP-код;
3) Запуск інтерпретатора PHP: Після визначення файлу як PHP-файлу, веб-сервер запускає інтерпретатор PHP. Інтерпретатор PHP відповідає за виконання PHP-коду;
4) Обробка PHP-коду: Інтерпретатор PHP проходить по PHP-файлу починаючи з першого рядка і виконує код по одній інструкції за раз. Він обробляє PHP-теги (наприклад, <?php та ?>), виконує змінні, функції, оператори, умовні конструкції та інші PHP-елементи;
5) Взаємодія з сервером та іншими сервісами: Під час обробки PHP-коду можуть відбуватися взаємодії з сервером баз даних, файловою системою, зовнішніми API, сесіями користувача та іншими сервісами;
6) Генерація вихідного HTML: Коли PHP-код виконано, результатом є згенерований HTML-код. Цей HTML-код буде включати як статичний контент, так і динамічні дані, залежно від виконаного PHP-коду;

7) Відправка відповіді клієнту: Після генерації HTML-коду веб-сервер відправляє цей HTML-код у відповідь клієнту, який отримує веб-сторінку для відображення у веб-браузері.

Ці етапи відбуваються позаду лаштування серверного середовища PHP, яке включає встановлення та налаштування веб-сервера, PHP-інтерпретатора і додаткових компонентів, які можуть бути використані, наприклад, бази даних.

Опис MySQL

MySQL - це відкрите програмне забезпечення для керування базами даних (DBMS - Database Management System), що використовує мову запитів SQL (Structured Query Language). Він є одним з найпопулярніших та найширше використовуваних реляційних СУБД у світі.

MySQL надає можливості для створення, зберігання, оновлення та керування базами даних. Він забезпечує швидкий доступ до даних і підтримує велику кількість одночасних підключень. MySQL може бути використаний як веб-сервером баз даних або локально на комп'ютері.

Основні особливості MySQL:

- Реляційна структура: MySQL базується на реляційній моделі даних, де дані зберігаються в таблицях з рядками і стовпцями. Він підтримує використання ключів для встановлення взаємозв'язків між таблицями;
- Мова запитів SQL: MySQL використовує стандартну мову запитів SQL для взаємодії з базою даних. SQL дозволяє виконувати різні операції, такі як вставка, оновлення, видалення і запити даних;
- Масштабованість: MySQL підтримує масштабованість з допомогою реплікації, кластерів та інших методів розподіленої обробки даних. Це дозволяє обробляти великі обсяги даних та забезпечувати високу доступність системи;

- Багатопоточність: MySQL може обробляти багато одночасних підключень, що дозволяє багатьом користувачам одночасно взаємодіяти з базою даних;
- Безпека: MySQL надає можливості для захисту бази даних, включаючи автентифікацію, авторизацію та шифрування даних;
- Підтримка різних платформ: MySQL доступний для різних операційних систем, включаючи Windows, Linux і Mac OS.

MySQL використовується в широкому спектрі додатків, включаючи веб-сайти, блоги, електронну комерцію, системи управління вмістом і багато інших. Він є потужним та надійним інструментом для роботи з базами даних.

Опис SASS

Sass (Syntactically Awesome Stylesheets) є мовою препроцесора, яка розширює можливості CSS. Вона надає додаткові функції, синтаксис та покращену організацію стилів, що спрощує розробку та підтримку стилей веб-сайтів і додатків.

Основні особливості Sass включають:

- Перемінні: Sass дозволяє використовувати перемінні для збереження значень, таких як кольори, шрифти, розміри, шляхи до зображень і т. д. Це спрощує управління і одночасне використання значень по всьому коду;
- Вкладеність: Sass дозволяє вкладати CSS-правила одне в одне, замість повторного писання селекторів. Це поліпшує організацію коду та зрозумілість структури стилів;
- Модульність і імпорт: Sass дозволяє розділяти стилі на окремі файли і потім імпортувати їх в основний файл. Це дозволяє розбити код на логічні частини, полегшує його підтримку та сприяє повторному використанню;

- Математичні операції: Sass дозволяє використовувати математичні операції для обчислення значень, таких як додавання, віднімання, множення та ділення. Це зручно для створення адаптивних стилів та виконання складних обчислень;
- Міксіни: Sass дозволяє створювати міксини, що є перевикористованими блоками стилів, які можуть приймати аргументи. Це дозволяє створювати зручні шаблони для стилів, які можна використовувати повторно та змінювати залежно від потреби;
- Функції: Sass підтримує власні функції, що дозволяють виконувати додаткові операції зі значеннями, обробляти рядки та кольори, створювати градієнти та багато іншого.

Sass компілюється в стандартний CSS, який браузери можуть розуміти. Компіляція виконується за допомогою Sass-компілятора, який перетворює код Sass в CSS перед розгортанням веб-сайту або додатку.

Опис CSS

CSS (Cascading Style Sheets) є мовою, що використовується для опису вигляду та форматування веб-сторінок. Вона визначає, як HTML-елементи мають відображатися на екрані, включаючи розмір, кольори, шрифти, розташування елементів та інші властивості.

Основні риси CSS:

- Селектори: CSS використовує селектори для вибору конкретних HTML-елементів або груп елементів, до яких будуть застосовуватися стилі. Наприклад, селектор може вибрати всі заголовки h1 на сторінці або елементи з певним класом;
- Властивості та значення: CSS має велику кількість властивостей, які можуть бути застосовані до вибраних елементів. Кожна властивість має своє значення, яке задається в CSS-правилі. Наприклад, властивість "color" встановлює колір тексту, а властивість "font-size" встановлює розмір шрифту;
- Каскадування та спадковість: CSS використовує механізм каскадування, що дозволяє встановлювати пріоритетність стилів в залежності від їхнього джерела та специфічності селекторів. Крім того, CSS дозволяє успадковувати стилі від батьківських елементів до дочірніх, що спрощує структуру та управління стилями;
- Розмітка і позиціонування: CSS надає можливості для контролю розмітки та позиціонування елементів на сторінці. За допомогою властивостей, таких як "display", "position", "float", "margin" та інші, можна керувати розміщенням та відступами елементів;
- Адаптивність: CSS дозволяє створювати адаптивні стилі, що змінюються в залежності від розміру екрану або пристрою. За допомогою медіа-запитів і властивостей, таких як "@media" та "flexbox", можна створювати резинові та реагуючі дизайни.

CSS використовується веб-розробниками для стилізації веб-сторінок, забезпечення їх зовнішнього вигляду та взаємодії з користувачем. Він є невід'ємною частиною фронтенд-розробки і співпрацює з HTML та JavaScript для створення повноцінних веб-додатків.

2.4 Опис структури системи та алгоритмів її функціонування

Структура онлайн-сервісу була побудована за допомогою MVC-підходу.

MVC (Model-View-Controller) - це архітектурний шаблон розробки програмного забезпечення, який використовується для розділення компонентів і логіки додатку на три основних шари: модель (Model), представлення (View) та контролер (Controller). Кожен шар виконує свою відповідальність та взаємодіє з іншими шарами за допомогою певних правил.

Основні компоненти MVC:

1) Модель (Model): Модель представляє дані та бізнес-логіку додатку. Вона відповідає за обробку, зберігання та маніпулювання даними, а також за виконання операцій і правил, пов'язаних з бізнес-процесами. Модель може включати базу даних, класи для роботи з даними, логіку обробки даних тощо;
2) Представлення (View): Представлення відображає дані користувачу та відповідає за візуальну частину додатку. Воно відображає дані з моделі і дозволяє користувачу взаємодіяти з додатком. Представлення може бути у вигляді HTML-сторінок, шаблонів, графічних елементів і т.д;
3) Контролер (Controller): Контролер приймає вхідні дані від користувача і взаємодіє з моделлю та представленням. Він керує потоком даних та логікою додатку, обробляє вхідні запити, оновлює модель та передає необхідні дані представленню для відображення. Контролер також може включати маршрутизацію, яка визначає, які дії повинні бути виконані відповідно до вхідних запитів.

Переваги використання MVC:

- Розділення відповідальностей: Модель, представлення і контролер виконують свої специфічні функції, що спрощує розробку,

підтримку та тестування коду. Кожен шар може бути розроблений та модифікований незалежно від інших, що полегшує розширення та модифікацію додатку;

- Підтримка багатоплатформеності: Завдяки розділенню логіки додатку від його візуальної частини, MVC дозволяє створювати додатки, які можуть бути легко адаптовані до різних платформ і пристроїв. Модель може залишатися незмінною, а представлення та контролер можуть бути змінені для різних інтерфейсів;
- Легкість управління кодом: Розділення логіки додатку на різні шари робить код більш організованим та легким у підтримці. Модульність і розсортованість дозволяють змінювати або замінювати окремі компоненти без необхідності переписування всього додатку.

Незважаючи на усі перелічені переваги MVC-підходу, він має й свої недоліки, а саме:

- Складність: MVC може бути складним для розуміння та впровадження, особливо для початківців у розробці програмного забезпечення. Розбиття програми на три шари може створювати додаткову складність в управлінні логікою та взаємодією між компонентами;
- Зайва складність: Для простих проектів або проектів з невеликою кількістю функціональності використання повноцінної MVC-архітектури може бути надмірним. Вона може вимагати більше часу та зусиль для налаштування та підтримки, ніж простіші альтернативи;
- Складність маршрутизації: MVC часто вимагає розробки системи маршрутизації, яка визначає, які запити відправляються до яких контролерів. Це може бути складно і часом заплутано для розробників, особливо при великій кількості сторінок та функцій;
- Недостатня гнучкість: Модель-View-Controller може бути досить жорсткою, особливо якщо виникають нестандартні вимоги або потрібні

великі зміни в логіці додатку. Виконання змін у всіх трьох шарах може бути часом- та ресурсозатратним;

– Навантаження на контролер: В MVC контролери відповідають за обробку багатьох різних запитів та керування логікою додатку. Це може призводити до зайвої залежності та перевантаження контролерів, що ускладнює їх підтримку та тестування.

Однак, ці недоліки не означають, що MVC не є ефективним шаблоном архітектури. Вони варто розглядати як фактори, які потрібно враховувати та збалансувати при виборі підходящої архітектури для конкретного проекту. MVC широко використовується в веб-розробці для побудови складних додатків з покращеними можливостями управління даними та взаємодії з користувачем.

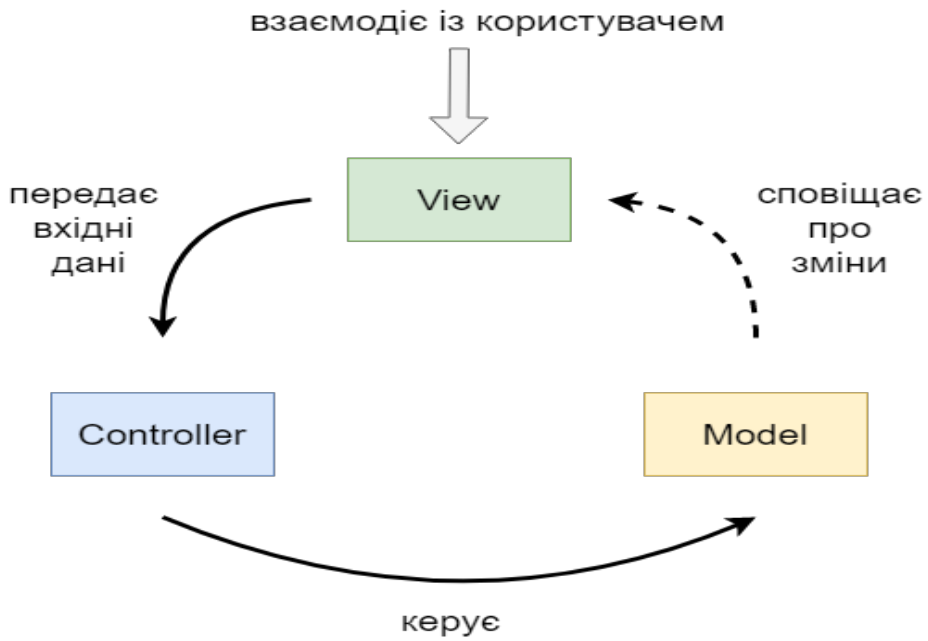

Рис. 2.4.1 – Архітектурний підхід MVC

Розгляд серверної частини проєкту:

Серверна частина веб-додатку побудована за допомогою спеціального веб-сервісу, який надає не тільки можливість хостингу файлів проєкту, а й надає онлайн-сервер Apache для обробки усіх необхідних запитів до веб-додатку та має вбудовану СКБД PhpMyAdmin для зручної роботи з базами даних.

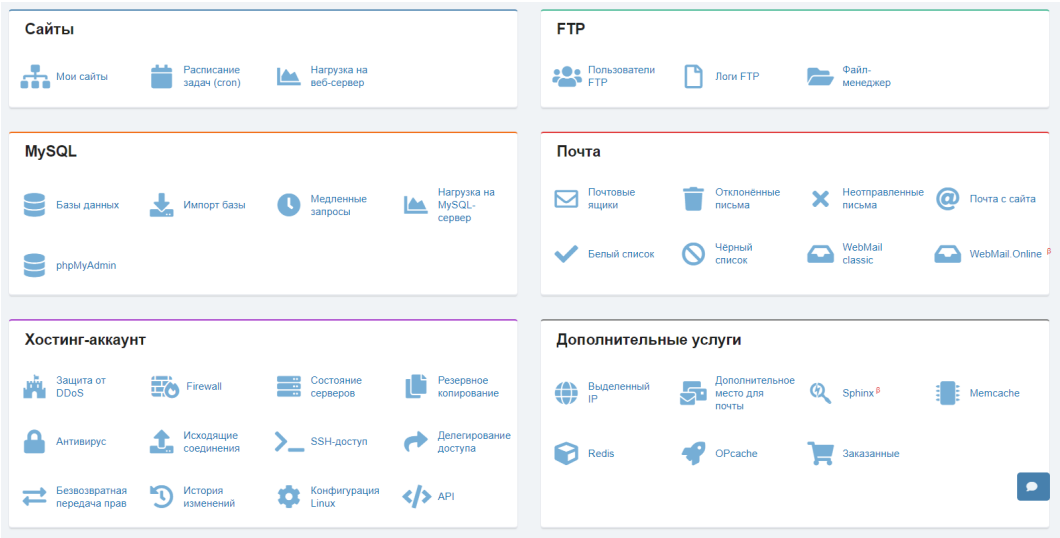

Рис. 2.4.2 – Загальний вигляд веб-сервісу

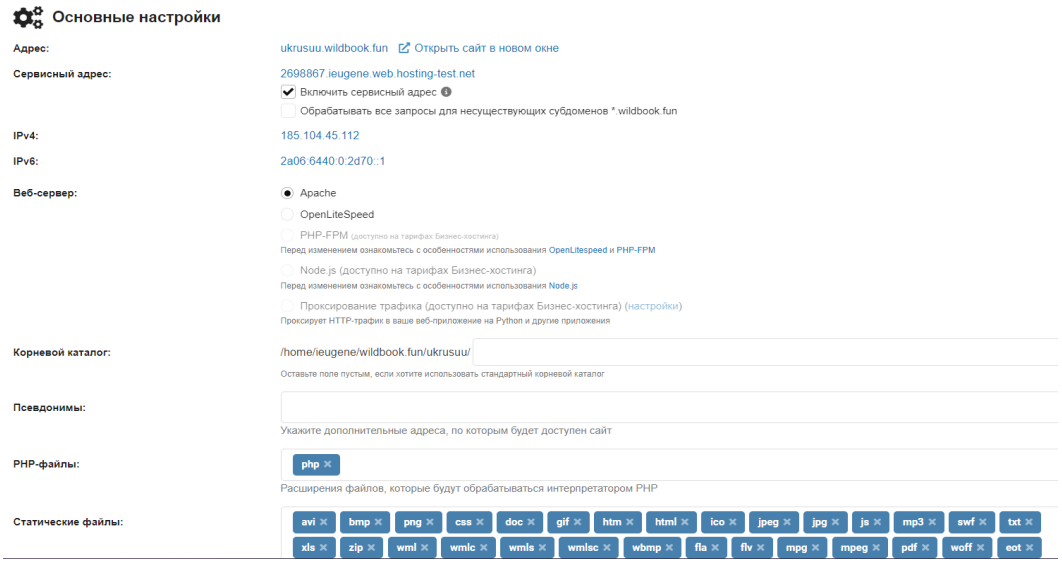

Рис. 2.4.3 – Налаштування веб-додатку

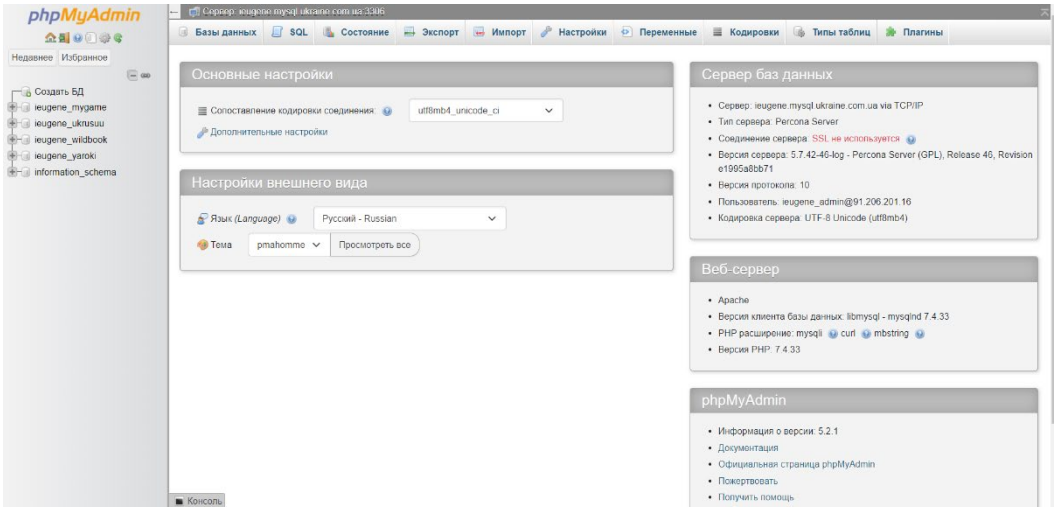

Рис. 2.4.4 – Загальний вигляд вікна PhpMyAdmin

MySQL є реляційною базою даних. Реляційна база даних (Relational Database) - це тип бази даних, який зберігає дані у вигляді таблиць зі зв'язками між ними. Вона ґрунтується на реляційній моделі даних, яка була розроблена Едгаром Коддом у 1970-х роках.

У реляційних базах даних (РБД) дані організовані за допомогою таблиць. Кожна таблиця представляє собою набір рядків (записів) і стовпців (полів). Кожен рядок таблиці відповідає окремому запису або об'єкту, а кожен стовпець містить певну інформацію про цей запис.

Основні компоненти організації даних у реляційних базах даних:

- Таблиці: Таблиці є основними структурними компонентами реляційної бази даних. Кожна таблиця має унікальне ім'я і складається зі стовпців і рядків. Кожен стовпець має назву і тип даних, що визначає, який тип інформації може бути збережений у цьому стовпці. Кожен рядок таблиці відповідає конкретному запису і містить значення для кожного стовпця;
- Ключі: Ключі використовуються для ідентифікації та забезпечення унікальності записів у таблиці. Головний ключ (Primary Key) визначає унікальний ідентифікатор кожного запису в таблиці.

Зовнішній ключ (Foreign Key) використовується для встановлення зв'язків між таблицями, посилаючись на головний ключ іншої таблиці;

- Запити: Запити використовуються для отримання, оновлення, вставки або видалення даних з таблиць. Запити можуть бути складними, включати фільтри, з'єднання таблиць та інші операції, які дозволяють отримувати потрібні дані з бази даних;
- Нормалізація: Нормалізація є процесом організації даних у реляційній базі даних з метою усунення аномалій та забезпечення ефективності та цілісності даних. Вона розбиває таблиці на менші компоненти, що допомагає уникнути повторення даних і забезпечує їх логічну структуру;
- Зв'язки між таблицями: Реляційні бази даних можуть містити зв'язки між таблицями, що визначають спосіб взаємодії даних між ними. Наприклад, зв'язки можуть бути один-до-одного (One-to-One), один-до-багатьох (One-to-Many) або багато-до-багатьох (Many-to-Many), в залежності від потреб додатку та структури даних.

Організація даних у реляційних базах даних дозволяє зберігати, оновлювати, вибирати та взаємодіяти з даними ефективно та структуровано, забезпечуючи цілісність та безпеку інформації.

Реляційні бази даних широко використовуються у різних галузях і сферах діяльності, де потрібно зберігати та керувати великими обсягами даних. Ось декілька галузей, де реляційні бази даних є популярними:

1) Веб-додатки: Багато веб-додатків використовують реляційні бази даних для зберігання інформації, такої як користувачі, замовлення, коментарі, статті тощо. Вони забезпечують структуровану організацію даних та дозволяють ефективно виконувати запити до бази даних;

2) Електронна комерція: Реляційні бази даних використовуються для зберігання інформації про товари, клієнтів, замовлення, оплати та інші елементи, пов'язані з електронною комерцією. Вони дозволяють ефективно керувати запасами, відстежувати замовлення та забезпечувати безпеку даних клієнтів;
3) Фінанси: У фінансовому секторі реляційні бази даних використовуються для зберігання інформації про транзакції, рахунки, клієнтів та інші фінансові дані. Вони забезпечують точність, цілісність та безпеку фінансової інформації;
4) Управління відносинами зі споживачами (CRM): Реляційні бази даних використовуються для зберігання інформації про клієнтів, контакти, історію взаємодії та інші дані, необхідні для управління відносинами зі споживачами. Вони допомагають відстежувати та аналізувати дані про клієнтів для покращення взаємодії та збільшення продажів;
5) Системи управління контентом (CMS): Багато систем управління контентом використовують реляційні бази даних для зберігання контенту, як-от статті, зображення, відео, коментарі та інше. Вони забезпечують структуроване зберігання та ефективну доставку контенту до користувачів.

Це лише кілька прикладів галузей, де реляційні бази даних широко використовуються. Завдяки своїй ефективності, надійності та розширюваності, реляційні бази даних залишаються одними з найпопулярніших інструментів для зберігання та управління даними.

Нормалізація у реляційних базах даних включає набір форм (нормальних форм), які допомагають усунути повторення даних і забезпечити логічну структуру бази даних. Ось основні форми нормалізації:

- Перша нормальна форма: Вимагає, щоб кожен стовпець у таблиці мав атомарні значення, тобто не може містити масиви або повторювані

групи значень. Кожен рядок у таблиці повинен бути унікальним ідентифікатором;

- Друга нормальна форма: Вимагає, щоб кожен стовпець у таблиці залежав від всього первинного ключа, а не від часткового ключа. Це означає, що таблиця повинна бути розбита на дві окремі таблиці, які взаємодіють через зв'язки;
- Третя нормальна форма: Вимагає, щоб кожен стовпець у таблиці залежав від первинного ключа безпосередньо, а не через інші стовпці. Якщо є залежність між стовпцями, вони також повинні бути розбиті на окремі таблиці;
- Нормальна форма Бойса-Кодда: Ця форма відображає умови для усунення аномалій оновлення, вставки та видалення в базі даних. Вона вимагає, щоб в таблиці не було неприродних залежностей, коли один стовпець залежить від іншого стовпця, що не є первинним ключем;
- Четверта нормальна форма: Вимагає, щоб в таблиці відсутні були множинні залежності, коли один стовпець залежить від групи стовпців, які не є первинним ключем.

Існують також п'ята нормальна форма і шоста нормальна форма, які стосуються більш специфічних ситуацій. Однак, перша-четверта нормальні форми є найбільш поширеними і часто використовуються для нормалізації реляційних баз даних.

Важливо зауважити, що нормалізація бази даних залежить від конкретного контексту і вимог проекту, і може варіюватись в залежності від конкретних сценаріїв використання та потреб користувачів.

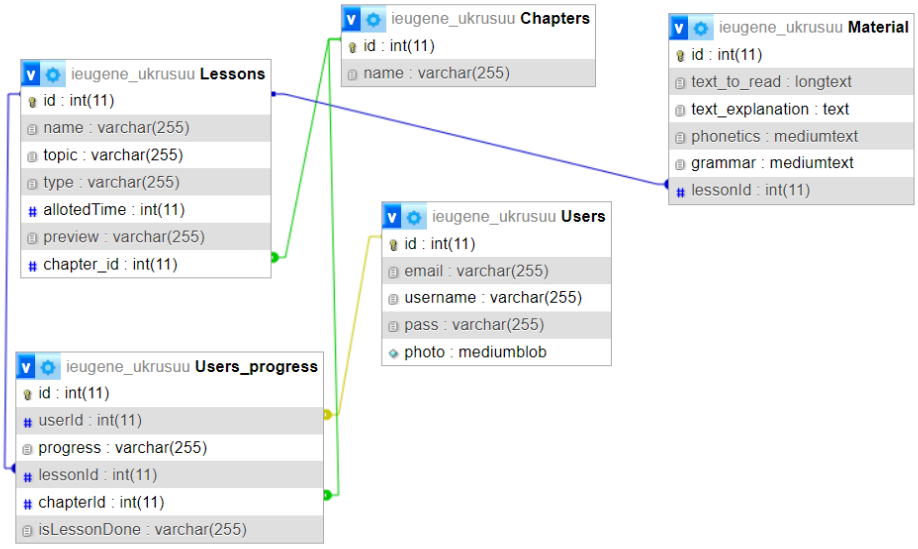

Рис. 2.4.5 – ER-діаграма БД веб-додатку

При розробці проєкту була обрана клієнт-серверна архітектура.

Клієнт-серверна архітектура є моделлю розподіленого обчислення, в якій задачі і функції системи розподілені між двома типами компонентів: клієнтами і серверами. Ця архітектура дозволяє розділити обов'язки і відповідальності між цими компонентами для покращення ефективності, масштабованості та доступності системи.

Основна ідея клієнт-серверної архітектури полягає в тому, що клієнти (користувачі або програми) взаємодіють з серверами (центральними обчислювальними ресурсами) для отримання необхідних ресурсів і послуг. Зазвичай, клієнти і сервери знаходяться на різних фізичних комп'ютерах або пристроях, і вони спілкуються між собою через мережу, таку як Інтернет або локальну мережу.

Основні характеристики клієнт-серверної архітектури:

- Розподілений доступ до ресурсів: Сервери зберігають ресурси, такі як дані, файли, програми, послуги тощо, і надають доступ до них клієнтам за запитом. Клієнти можуть виконувати запити на сервери для отримання потрібних ресурсів;

- Розділення обов'язків: Клієнти відповідають за представлення інтерфейсу користувача та взаємодію з користувачем, тоді як сервери забезпечують обробку запитів, зберігання даних і надання послуг. Це дозволяє розділити завдання і сконцентруватись на специфічних областях відповідальності;
- Відсутність складності: Клієнти можуть бути простими пристроями або програмами з обмеженими обчислювальними можливостями, тоді як сервери можуть бути потужними обчислювальними системами. Це дозволяє розподілити навантаження обчислювальних завдань між різними компонентами системи;
- Масштабованість: Клієнт-серверна архітектура дозволяє масштабувати систему, додаючи більше серверів або клієнтів за потреби. Це допомагає забезпечити високу доступність та швидкодію системи навіть при зростаючому навантаженні;
- Безпека: Клієнт-серверна архітектура дозволяє застосовувати механізми безпеки на рівні серверів для захисту даних і ресурсів. Наприклад, сервери можуть використовувати аутентифікацію та авторизацію для контролю доступу клієнтів до ресурсів.

Клієнт-серверна архітектура широко використовується в різних системах, таких як веб-додатки, мобільні додатки, бази даних, електронна пошта, хмарні сервіси та інші. Вона надає зручну та ефективну модель для розподіленого обчислення та обміну даними між різними компонентами системи.

HTTP (Hypertext Transfer Protocol) є протоколом передачі гіпертекстових даних через мережу, таку як Інтернет. Він використовується для комунікації між клієнтами (наприклад, веб-браузерами) та веб-серверами для отримання веб-сторінок, зображень, відео, даних форм та іншого вмісту.

Основні характеристики HTTP:

- Запит-відповідь: Протокол HTTP базується на моделі запит-відповідь, де клієнт надсилає запит серверу, а сервер повертає відповідь на цей запит. Запити та відповіді можуть містити заголовки та тіло, що містять метадані та дані відповідно;
- Без стану: Протокол HTTP є безстановним, що означає, що кожний запит-відповідь обробляється незалежно, і сервер не зберігає жодної інформації про попередні запити. Це забезпечує простоту та масштабованість веб-систем;
- URI (Uniform Resource Identifier): В HTTP запити та відповіді використовують URI для ідентифікації ресурсів, таких як веб-сторінки, зображення або API-маршрути. URI включає URL (Uniform Resource Locator) або URN (Uniform Resource Name), які вказують на місцезнаходження або ідентифікатор ресурсу в мережі;
- Методи запиту: HTTP визначає різні методи запиту, такі як GET, POST, PUT, DELETE, для виконання різних дій над ресурсами. Наприклад, GET використовується для отримання ресурсу, POST - для створення нового ресурсу, PUT - для оновлення ресурсу, DELETE - для видалення ресурсу;
- Заголовки: HTTP використовує заголовки для передачі метаданих в запитах та відповідях. Заголовки можуть включати інформацію про тип вмісту, кодування, кешування, автентифікацію та інші параметри, які допомагають управляти комунікацією між клієнтом та сервером.

HTTP є основним протоколом веб-комунікації і використовується у багатьох системах, включаючи веб-додатки, RESTful API, веб-служби та багато іншого. Він забезпечує стандартизований спосіб взаємодії між клієнтами та серверами для передачі даних через мережу.

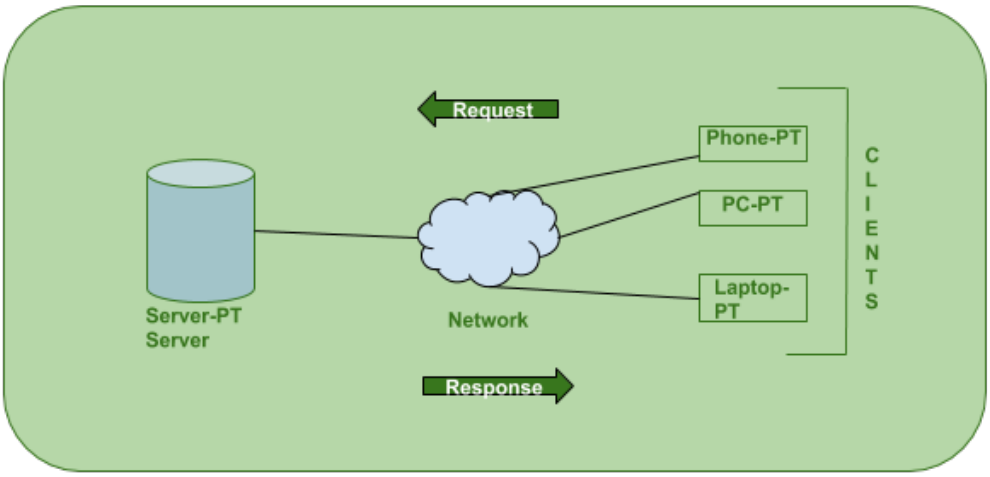

Рис. 2.4.6 – Алгоритм роботи серверної частини веб-додатку

2.5 Обгрунтування та організація вхідних та вихідних даних програми

Вхідними даними веб-додатку є заповнені веб-компоненти та поля форм системи. Після отримання усіх вхідних даних, система оброблює їх за допомогою TypeScript та PHP-коду та відправляє отриманий результат до бази даних.

Вихідними даними є уся інформація щодо переліку уроків та їх внутрішнього матеріалу, а також весь прогрес користувача. Все це береться безпосередньо з бази даних.

2.6 Опис розробленої системи.

2.6.1 Використані технічні засоби.

Наступні рекомендовані характеристики обчислювальної машини наведені виключно для клієнтського обладнання:

- Процесор Intel Core i5-11400H 2.70GHz;
- Оперативна пам’ять в розмірі 16 ГБ;
- Вільне місце на SSD-диску 2 ГБ;
- Відеокарта Nvidia GeForce GTX1650 GDDR6 1780MHz;

- Монітор 15.6 дюймів та розширенням 1920x1080;
- Маніпулятор “миша” або тачпад;
- Клавіатура;
- Доступ до глобальної інтернет-мережі.

Вище перелічені характеристики є здебільш рекомендованими для комфортної праці веб-додатку. При наявності характеристик нижче або вище зазначених, може змінитися час відповіді сайту або серверу.

2.6.2 Використані програмні засоби.

Веб-додаток реалізований за допомогою мов програмування TypeScript та PHP. Цих технологій достатньо для роботи з даними на стороні клієнту. Візуальна частина сайту побудована використовуючи препроцесор SASS та каскадні таблиці стилів (CSS).

Необхідні програмні засоби:

- Веб-браузер Google Chrome, Mozilla Firefox або Microsoft Edge;
- Операційна система Windows 11/10/7;
- Локальний/онлайн веб-сервер для зберігання даних;
- Редактор коду для подальшої підтримки веб-додатку, наприклад Visual Studio Code.

2.6.3 Виклик та завантаження програм

Розроблений онлайн-сервіс потребує веб-браузер, який підтримує HTML5, JavaScript та PHP для його подальшої якісної експлуатації.

2.6.4 Опис інтерфейсу користувача

Для того, щоб безпосередньо розпочати роботу з веб-додатком, першим чином необхідно перейти до нього за посиланням у веб-браузері, яке направить користувача до головної сторінки сайту (рис. 2.6.1).

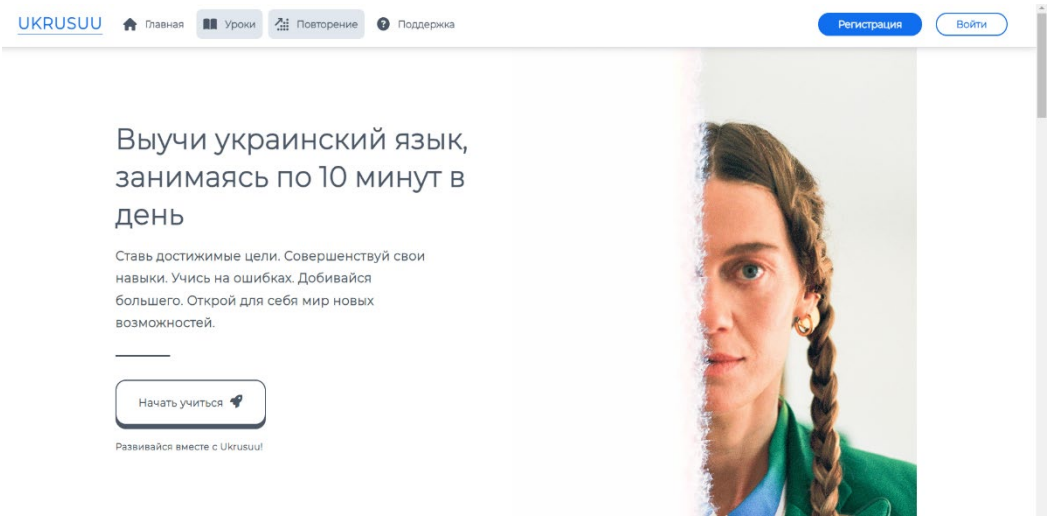

Рис 2.6.1 – Головна сторінка веб-додатку

Головна сторінка представляє з себе ознайомчу сторінку, завдяки якій користувач сайту може дізнатися про переваги вивчання української мови (рис 2.6.2), плюсах навчання мови у веб-додатку (рис. 2.6.3) та отримати усю необхідну йому інформацію. Завдяки кнопці “Начать учиться” користувач може перейти безпосередньо до навчання. Також на сторінці присутнє навігаційне меню, яке дозволяє зручно керувати веб-сайтом та “меню-підвал”, завдяки якій користувач може звернутися до підтримки сайту або увійти до особистого

акаунту (рис 2.6.4). У цілому, головна сторінка сайту мотивує користувача залишитися на сайті та розпочати своє навчання.

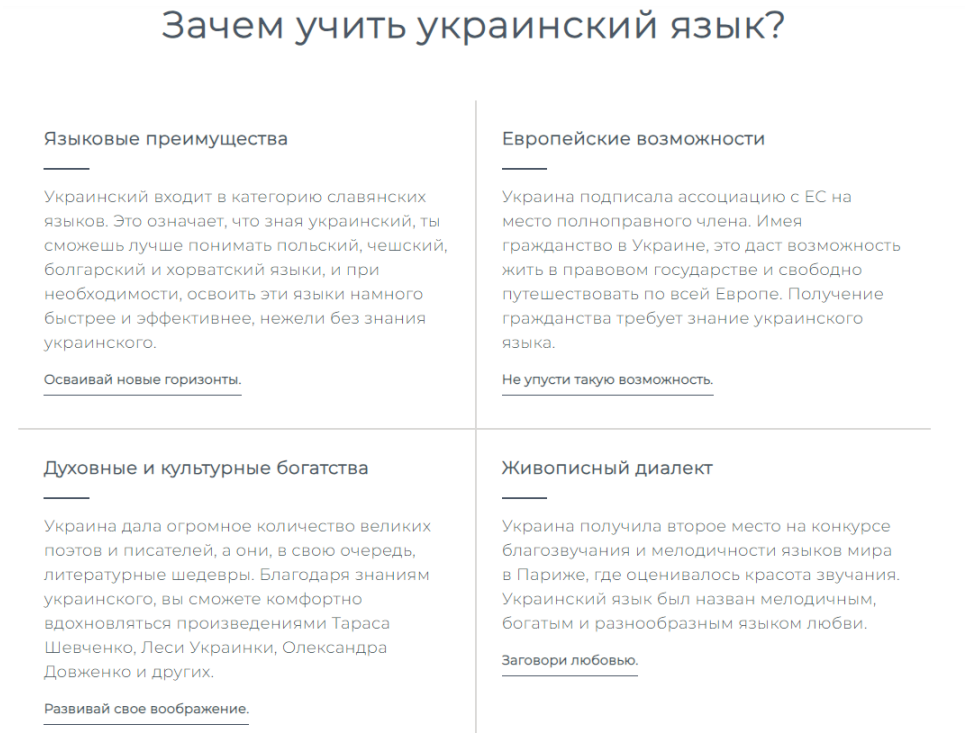

Рис 2.6.2 – Перелік переваг навчання української мови.

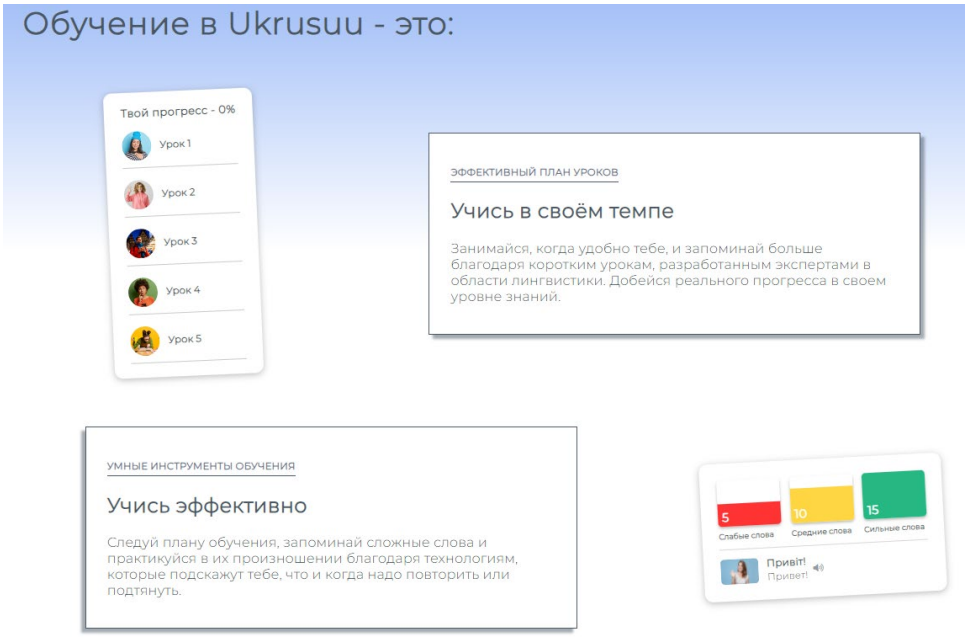

Рис 2.6.3 – Перелік плюсів навчання у веб-додатку.

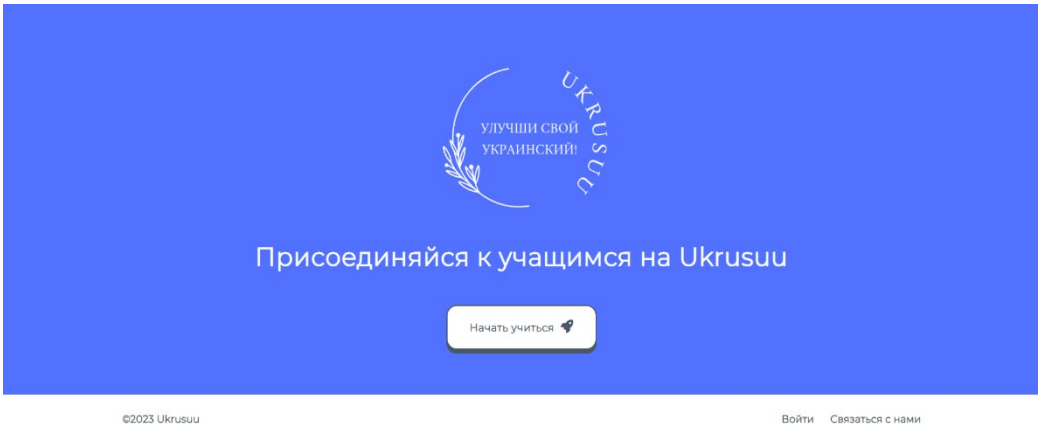

Рис 2.6.4 – “Меню-підвал” головної сторінки сайту.

Веб-додаток має біло-синій з сірими відтінками дизайн, стиль якого було обрано після переглядання інших сайтів-сервісів з навчання будь-яких мов, таких як Duolingo, Lingualeo, Busuu та інші.

Використовуючи навігаційне меню, користувач може перейти до розділу сайту “Уроки” та “Повторение”, однак, вони можуть бути заблоковані у разі, якщо користувач не увійшов до свого акаунту. У цьому разі, користувач отримує сповіщення про необхідність входу до акаунту (рис. 2.6.5).

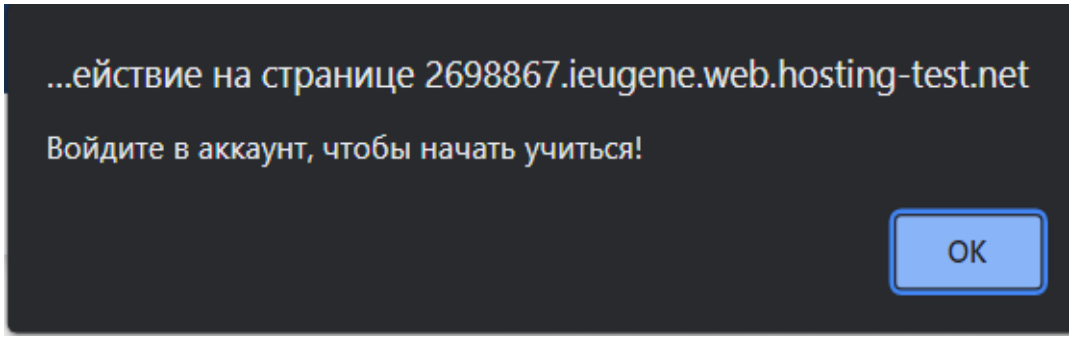

Рис 2.6.5 – Сповіщення про необхідність входу до акаунту.

Натискаючи на кнопку “Регистрация”, користувач може перейти до реєстрації власного акаунту (рис. 2.6.6).

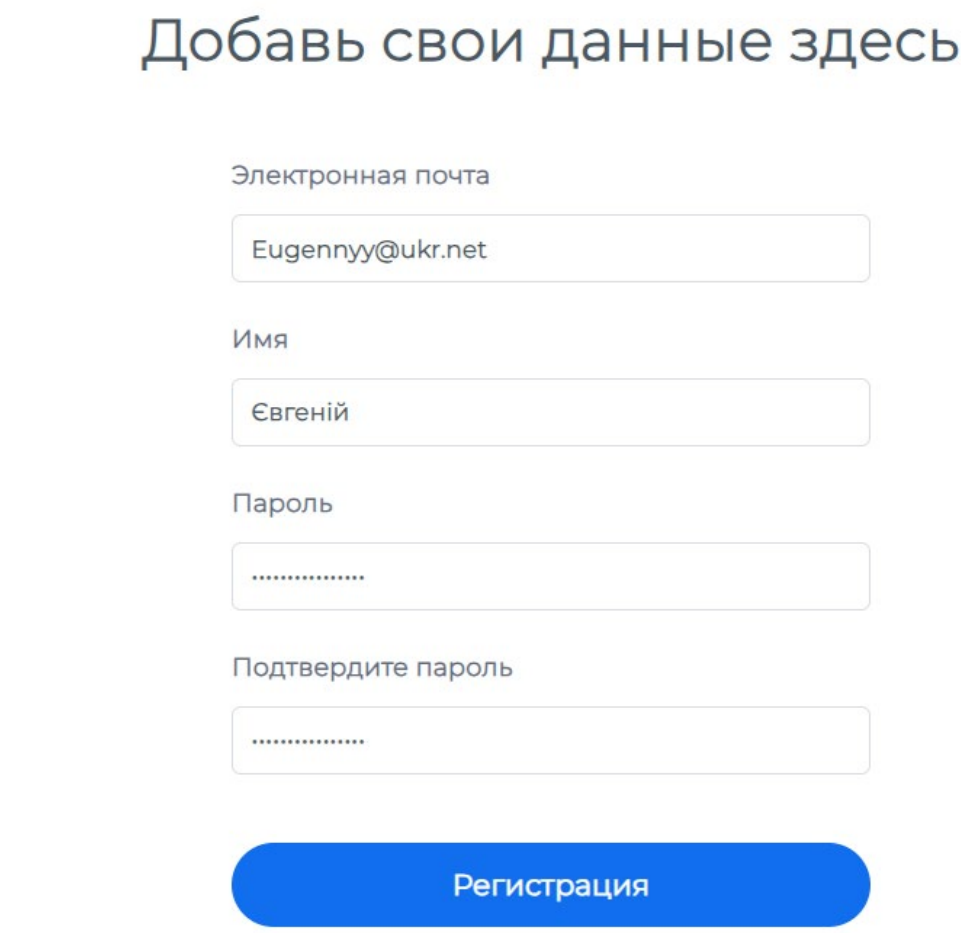

Рис 2.6.6 – Реєстрація акаунту.

Після успішної вказівки усіх необхідних даних для реєстрації, користувач повинен підтвердити завершення реєстрації спеціальним кодом, який прийде до електронної пошти (рис. 2.6.7).

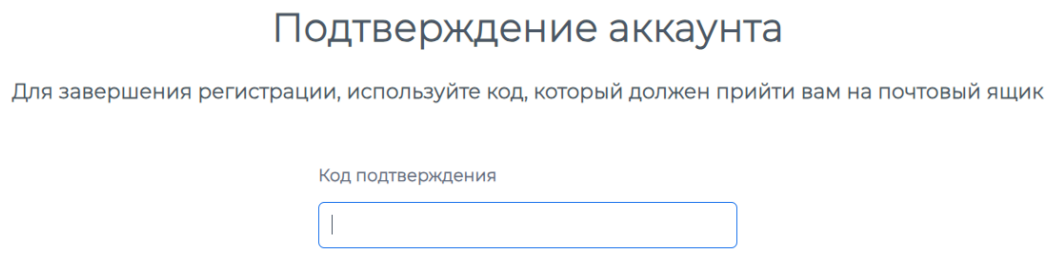

Рис 2.6.7 – Вікно підтвердження реєстрації акаунту.

При вводі правильного коду, користувач отримує сповіщення про успішне створення акаунту (рис. 2.6.8) та перейде до сторінки входу до акаунту, на якій він може авторизуватися до акаунту (рис 2.6.9).

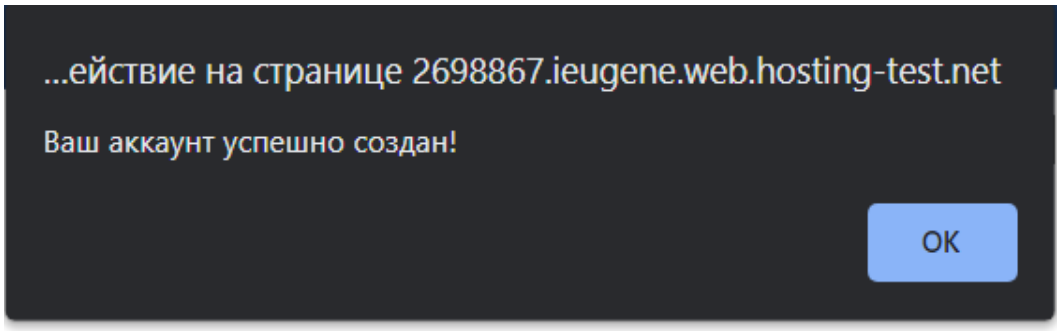

Рис 2.6.8 – Сповіщення про успішне створення акаунту.

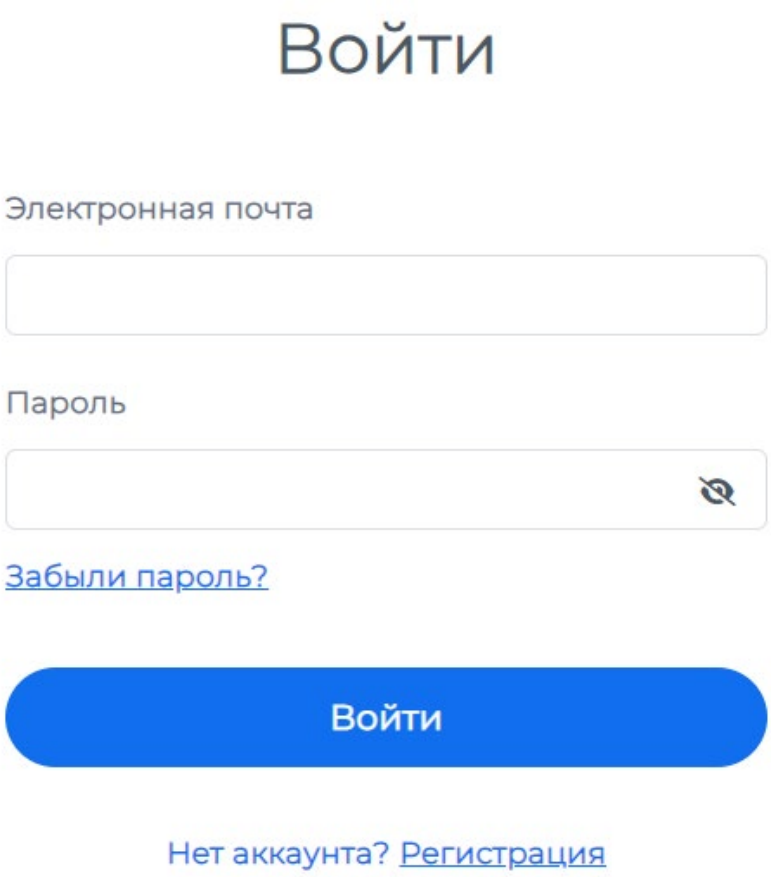

Рис 2.6.9 – Сторінка входу до акаунту.

У разі, якщо користувач забув поточний пароль, він може скористатися функцією скидання паролю, процес якого зображений на рисунках 2.6.10 - 2.6.12.

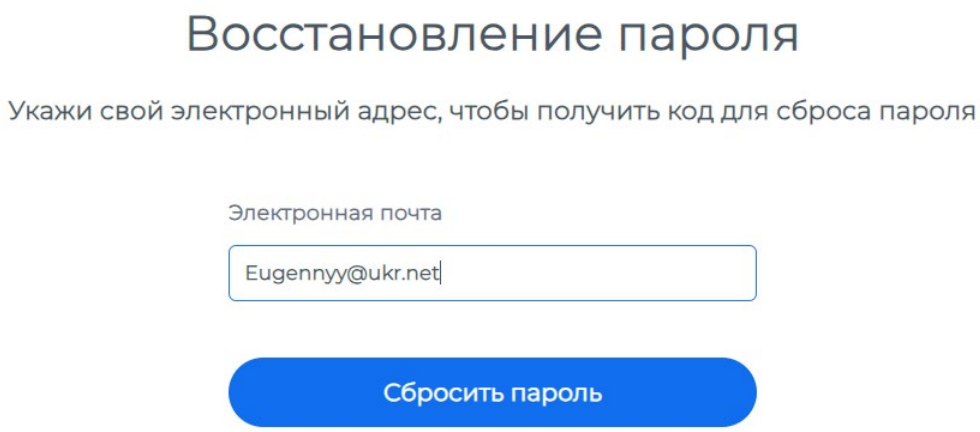

Рис 2.6.10 – Скидання поточного паролю.

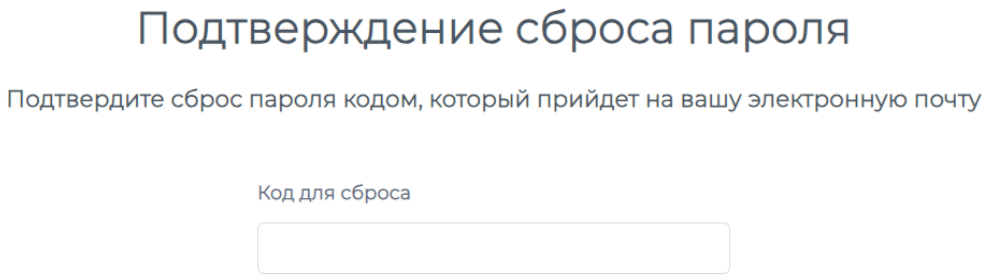

Рис 2.6.11 – Скидання поточного паролю.

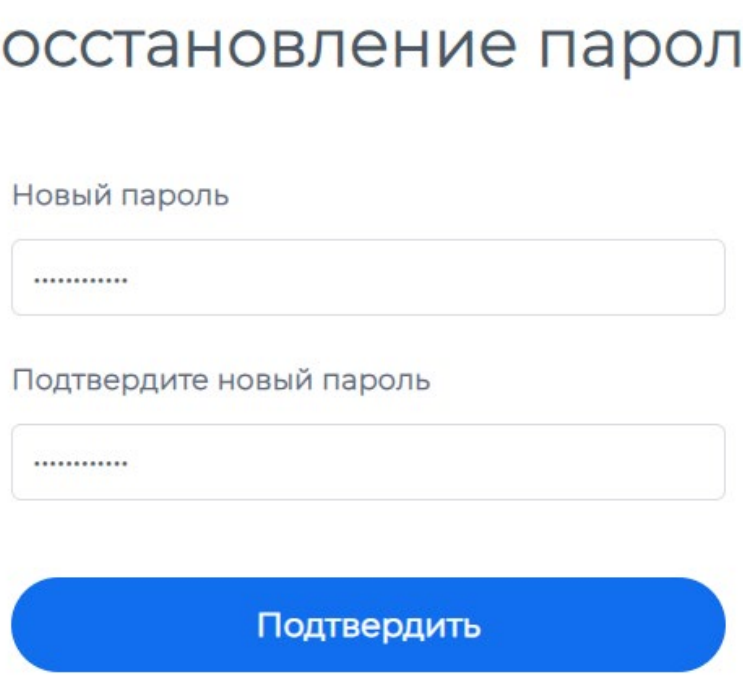

Рис 2.6.12 – Скидання поточного паролю.

Після успішного скидання паролю, користувач отримає відповідне сповіщення (рис. 2.6.13) та зможе заново авторизуватися.

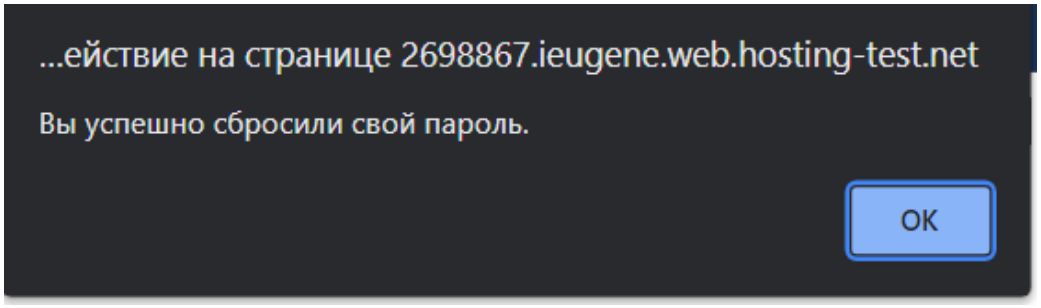

Рис 2.6.13 – Сповіщення про успішне скидання паролю.

Якщо вхід було виконано успішно, користувач побачить сторінку керування акаунтом, де він може змінити власні дані, видалити акаунт чи вийти з нього (рис. 2.6.14).

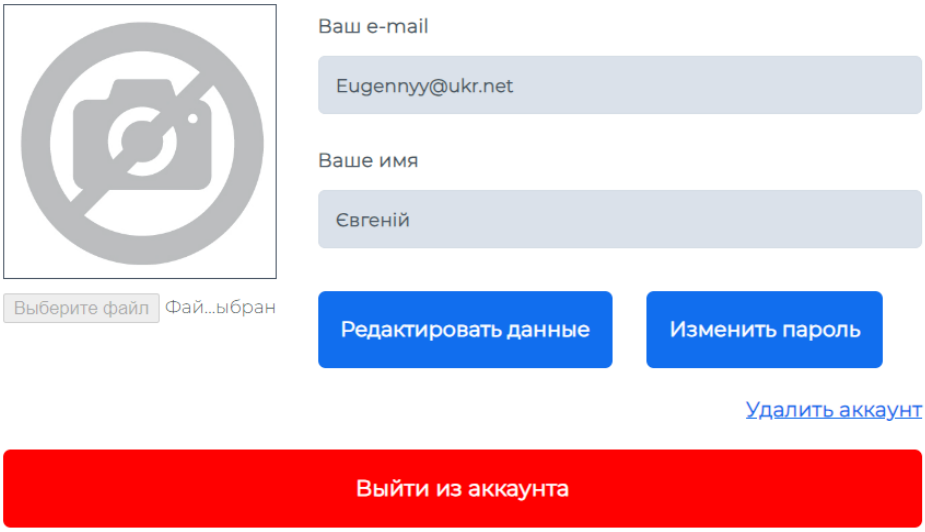

Рис 2.6.14 – Сторінка керування акаунту.

Після успішної авторизації, користувач має змогу приступити до навчання української мови використовуючи посилання “Уроки”, перейшовши до якої, він побачить свій поточний прогрес навчання та повний перелік уроків (рис. 2.6.15).

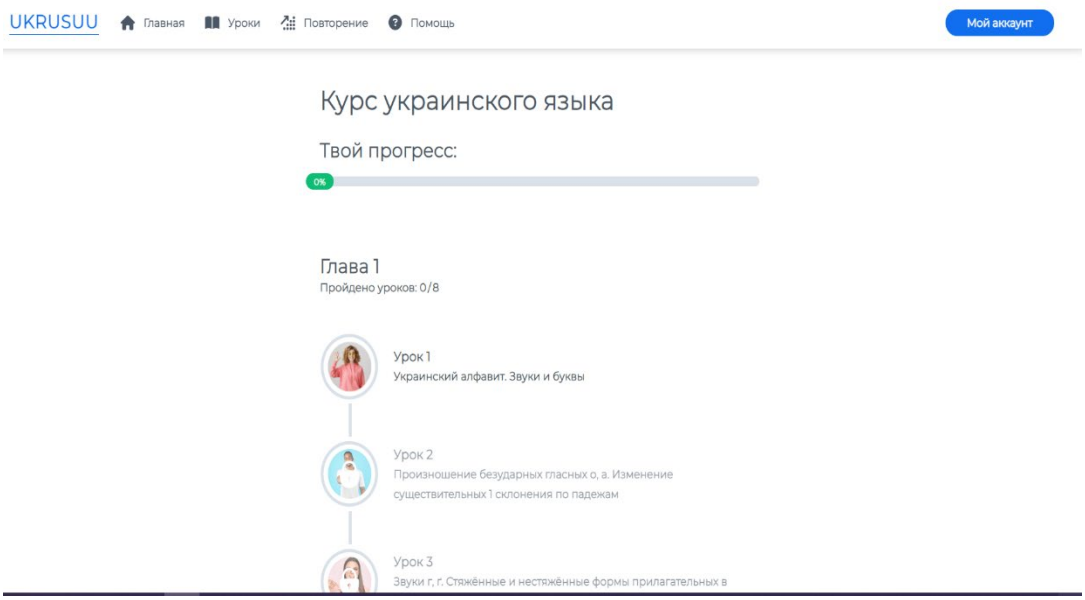

Рис 2.6.15 – Сторінка “Уроки”.

Натискаючи на урок, користувач отримає повну інформацію стосовно уроку та може розпочати його, натиснувши на кнопку “Начать урок” (рис. 2.6.16).

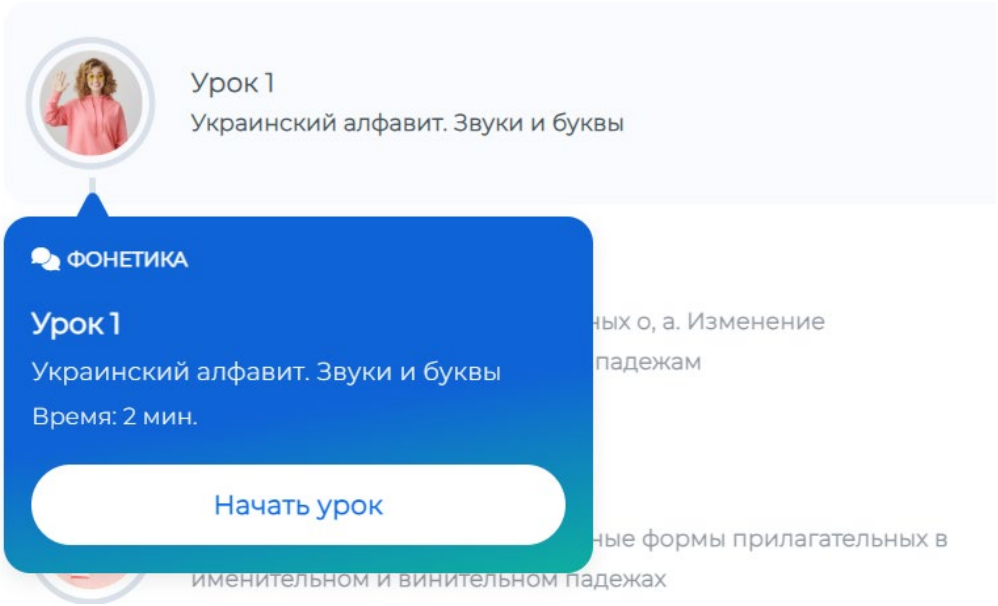

Рис 2.6.16 – Обрання уроку.

Розпочавши урок, користувач буде проходити та навчатися матеріалу, наприклад, читати текст, вивчати правила та проходити завдання. У процесі його проходження, користувач може побачити, яку кількість уроку він вже пройшов завдяки смужці прогресу, яка інтерактивно змінюється впродовж проходження. Також, якщо потрібно, користувач завжди може вийти з уроку, використовуючи відповідну кнопку. Процес проходження уроку зображений на рисунках 2.6.17 - 2.6.18.

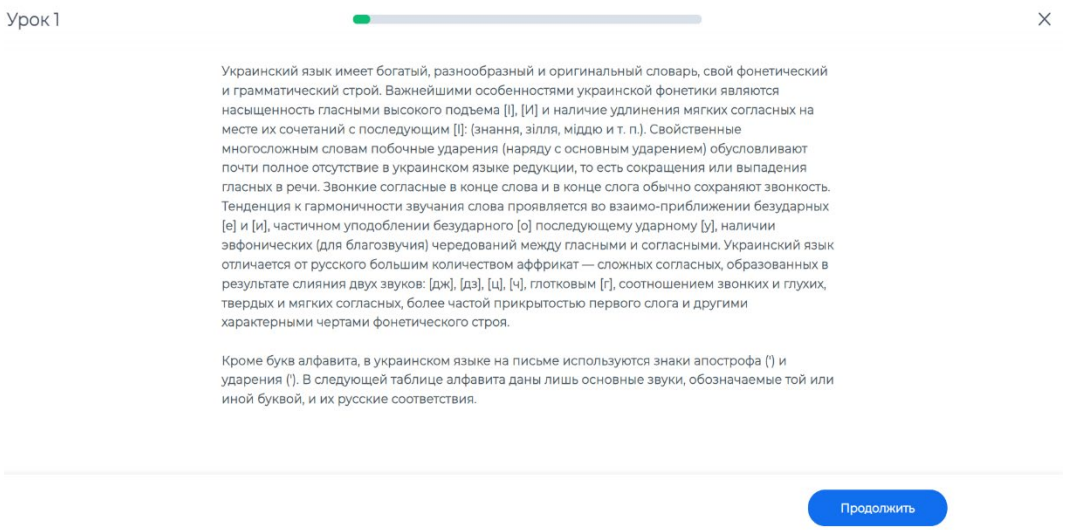

Рис 2.6.17 – Проходження уроку.

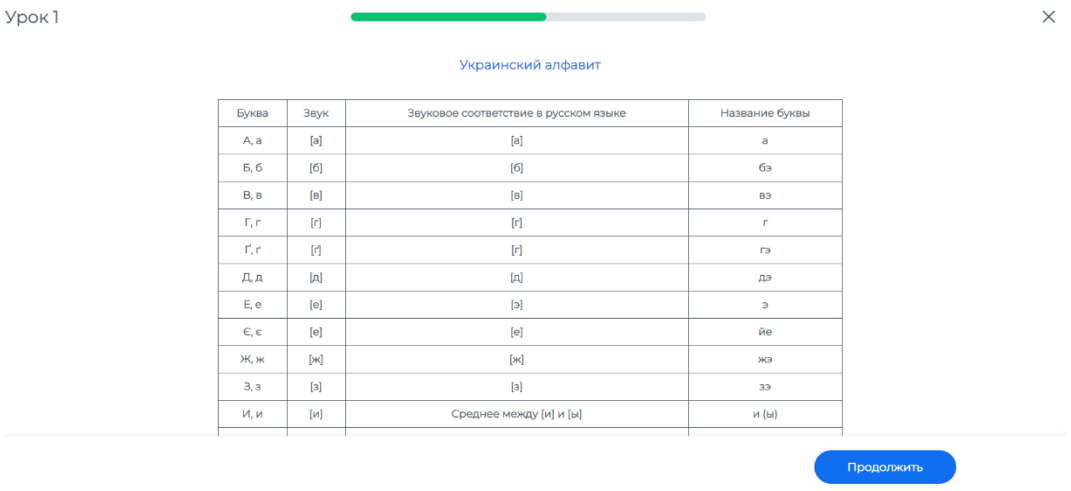

Рис 2.6.18 – Проходження уроку.

У разі успішного проходження, користувач отримає сповіщення про вдале завершення уроку. Його прогрес оновиться, а також він отримає доступ до наступного уроку. Все це зображено на рисунках 2.6.19 - 2.6.20.

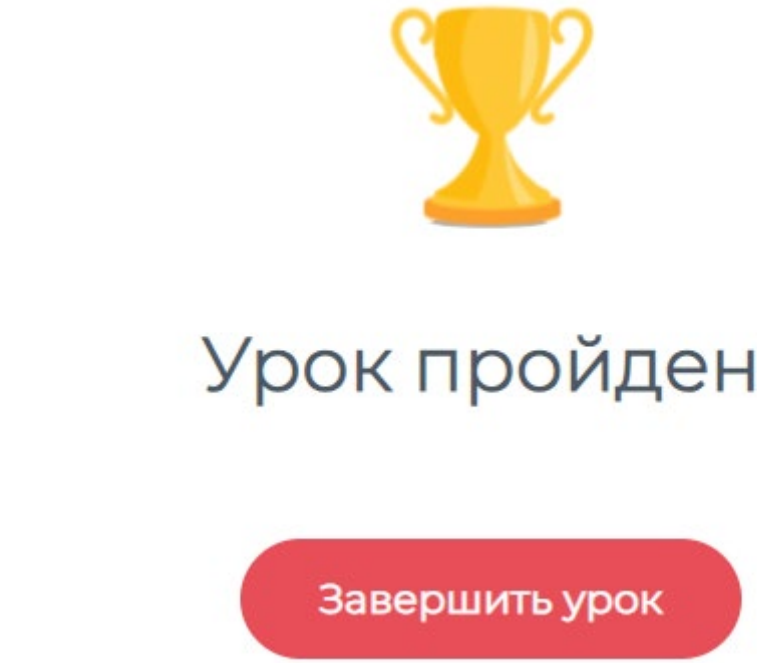

Рис 2.6.19 – Сповіщення про вдале завершення уроку.

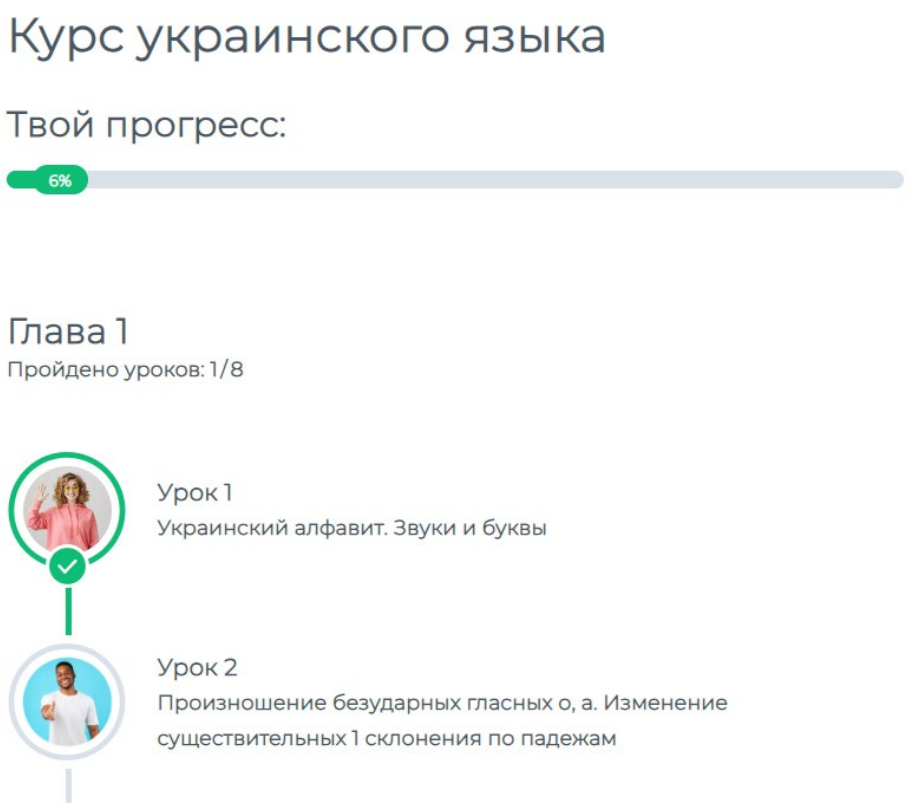

Рис 2.6.20 – Оновлений прогрес користувача.

У випадку, якщо користувач спробує розпочати урок, який він ще не відкрив, він отримає сповіщення-підказку (рис. 2.6.21).

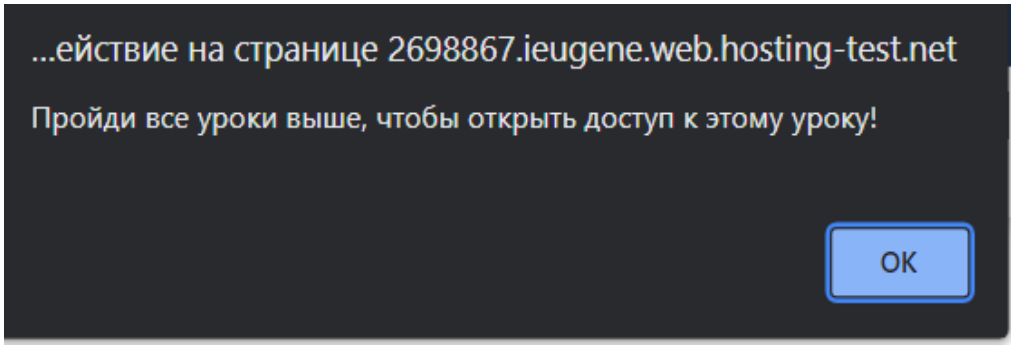

Рис 2.6.21 – Сповіщення-підказка.

Проходячи уроки, які мають в собі тексти для читання, користувач навчається новим словам, які згодом можна повторити завдяки сторінці “Повторение”, куди зберігаються усі слова з текстів, які були пройдені користувачем. Тут можна побачити, скільки слів було вивчено користувачем, їх “силу” та значення на українській-російській мові (рис. 2.6.22).

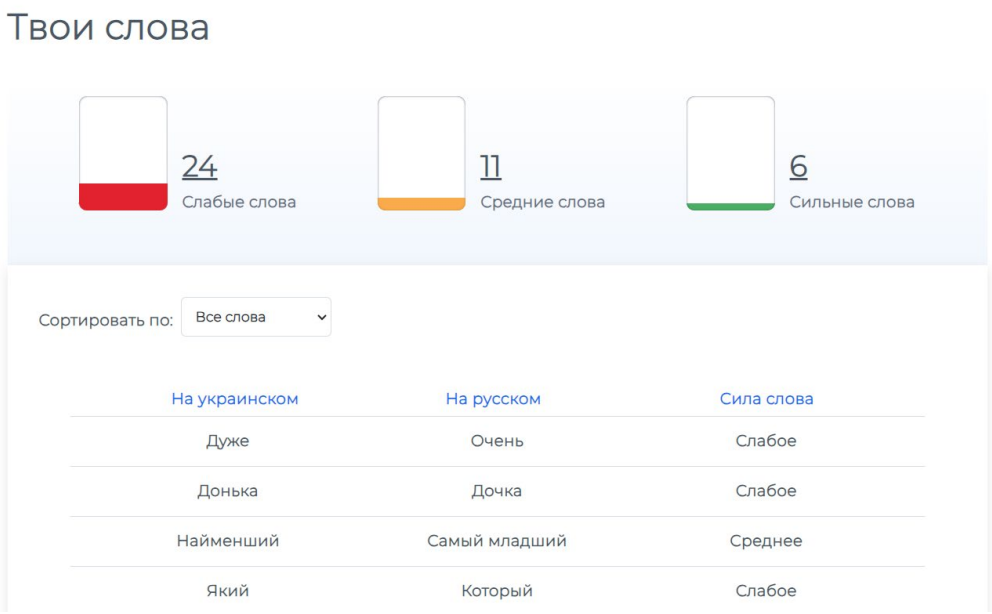

Рис 2.6.22 – Сторінка “Повторение”.

Використовуючи кнопку “Сортировать по” можна відфільтрувати перелік слів за складністю слова (рис. 2.6.23).

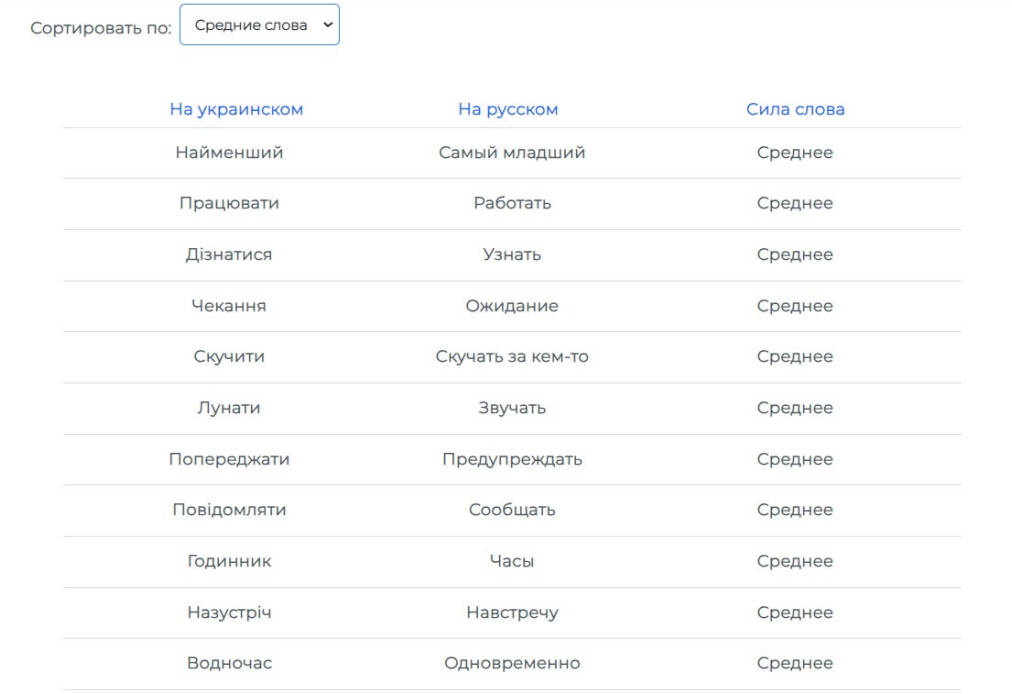

Рис 2.6.23 – Приклад фільтрації переліку слів.

Останньою сторінкою веб-додатку є сторінка підтримки, до якої можна перейти використовуючи посилання “Помощь”. Заповнив відповідну форму та натиснувши на кнопку “Отправить”, можна відправити запитання до підтримки сайту (рис. 2.6.24).

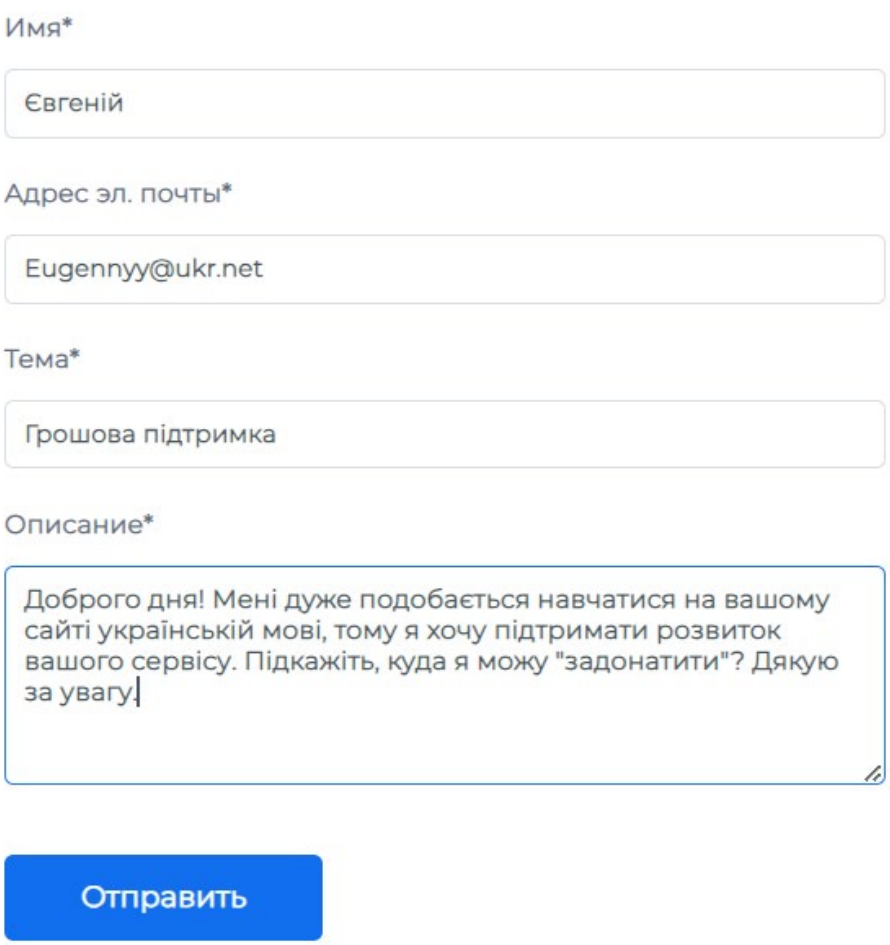

Рис 2.6.24 – Відправлення запитання.

У разі успішного заповнення форми та відправки, користувач отримає відповідне сповіщення (рис. 2.6.25).

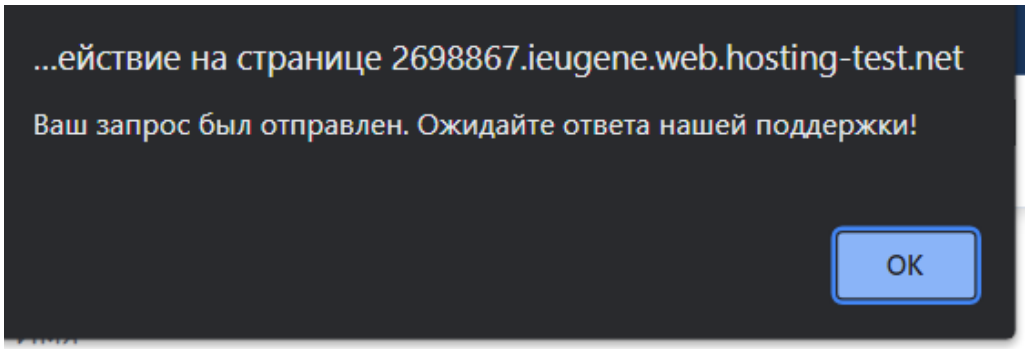

Рис 2.6.25 – Сповіщення про успішну відправку запитання.

РОЗДІЛ 3. ЕКОНОМІЧНИЙ РОЗДІЛ

В економічній частині було розраховано трудомісткість розробки і розрахунок витрат на створення.

3.1 Розрахунок трудомісткості та вартості розробки програмного продукту.

Задані дані:

1. Передбачуване число операторів – 8000;
2. Коефіцієнт складності програми – 2;
3. Коефіцієнт корекції програми в ході її розробки – 0,5;
4. Годинна заробітна плата програміста – 156,25;
5. Коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі – 1,3;
6. Коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності – 1,3;
7. Вартість машинно-години ЕОМ – 22.35.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д \text{, людино-годин,} \quad (3.1)$$

де t_o - витрати праці на підготовку й опис поставленої задачі (приймається 50);

$t_и$ - витрати праці на дослідження алгоритму рішення задачі;

$t_а$ - витрати праці на розробку блок-схеми алгоритму;

$t_п$ - витрати праці на програмування по готовій блок-схемі;

$t_{отл}$ - витрати праці на налагодження програми на ЕОМ;

$t_д$ - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \cdot C(1 + p), \text{ де} \qquad (3.2)$$

q – передбачуване число операторів;

C – коефіцієнт складності програми;

P – коефіцієнт кореляції програми в ході її розробки.

$$Q = 8000 \times 2(1 + 0{,}5) = 24000$$

Витрати праці на вивчення опису задачі $t_и$ визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_и = \frac{Q \cdot B}{(75...85)K}, \text{ людино-годин,} \qquad (3.3)$$

де B – коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі, В=1.2 ... 1.5;

K – коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності. до 2 – 0,8;

$$t_u = \frac{24000 \times 1{,}3}{85 \times 1{,}3} = 282{,}35, \text{ людина} - \text{годин}$$

Витрати праці на розробку алгоритму рішення задачі:

$$t_a = \frac{Q}{(20...25)K} \tag{3.4}$$

$$t_a = \frac{24000}{20 \times 1{,}3} = 923{,}07, \text{ людина} - \text{годин}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20...25)K} \tag{3.5}$$

$$t_n = \frac{24000}{25 \times 1{,}3} = 738{,}4, \text{ людина} - \text{годин}$$

Витрати праці на налагодження програми на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4...5)K} \tag{3.6}$$

$$t_{отл} = \frac{24000}{5 \times 1{,}3} = 3692{,}3, \text{ людина} - \text{годин}$$

- за умови комплексного налагодження завдання:

$$t_{отл}^{к} = 1{,}2 \cdot t_{отл} \tag{3.7}$$

$$t_{отл}^{к} = 1{,}2 \times 3692{,}3 = 4430{,}76, \text{ людина } - \text{ годин}$$

Витрати праці на підготовку документації:

$$t_{д} = t_{др} + t_{до} \tag{3.8}$$

де $t_{др}$ - трудомісткість підготовки матеріалів і рукопису

$$t_{др} = \frac{Q}{(15...20)K} \tag{3.9}$$

$$t_{dр} = \frac{24000}{20 \times 1{,}3} = 923{,}07, \text{ людина } - \text{ годин}$$

$t_{до}$ – трудомісткість редагування, печатки й оформлення документації

$$t_{до} = 0{,}75 \cdot t_{др} \tag{3.10}$$

$$t_{do} = 0{,}75 \times 923{,}07 = 692{,}3, \text{ людина } - \text{ годин}$$

$$t_{d} = 923{,}07 + 692{,}3 = 1615{,}37, \text{ людина } - \text{ годин}$$

Отримаємо трудомісткість розробки програмного забезпечення:

$$t = 282{,}35 + 923{,}07 + 738{,}4 + 4430{,}76 + 692{,}3 + 1615{,}37 = 8682{,}25, \text{ людина } - \text{ годин}$$

У результаті ми розрахували, що в загальній складності необхідно 8682,25 людино-годин для розробки даного програмного забезпечення.

3.2. Рахунок витрат на створення програми

Витрати на створення ПЗ $К_{по}$ включають витрати на заробітну плату виконавця програми $З_{зп}$ і витрат машинного часу, необхідного на налагодження програми на ЕОМ.

$$КПО = ЗЗП + ЗМВ \text{ , грн,} \tag{3.11}$$

де $З_{ЗП}$ – заробітна плата виконавців, яка визначається за формулою:

$$ЗЗП = t \cdot СПР \text{ , грн,} \tag{3.12}$$

де t – загальна трудомісткість, людино-годин;

$С_{ПР}$ – середня годинна заробітна плата програміста, грн/година.

$$З_{зп}\text{грн.} = 8682{,}25 \times 156{,}25 = 1356601{,}56$$

$З_{МВ}$ – Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$ЗМВ = t_{отл} \cdot СМ \text{ , грн,} \tag{3.13}$$

де $t_{отл}$ – трудомісткість налагодження програми на ЕОМ, год.

$С_{МЧ}$ – вартість машино-години ЕОМ, грн/год.

$$З_{мв}\text{грн.} = 4430{,}76 \times 22.35 = 99027{,}486$$

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення ПЗ:

$$К_{по}грн. = 1356601,56 + 99027,486 = 1455629,05$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p}, \text{ мес.} \quad (3.14)$$

де B_k - число виконавців;

F_p - місячний фонд робочого часу (при 40 годинному робочому тижні Fp=176 годин).

$$T = \frac{8682,25}{1 \times 176} = 49,3 \text{ міс.}$$

Висновок: Час розробки даного програмного забезпечення складає 8682,25 людино-годин. Таким чином, очікувана тривалість розробки складе 49,3 місяця при 40 годинному робочому тижні (місячний фонд робочого часу 176 годин), а витрати на створення програмного забезпечення складатимуть 1455629,05 грн.

ВИСНОВКИ

В рамках кваліфікаційної роботи був створений веб-додаток в якості онлайн-сервісу для вивчання української мови. При розробці сайту була використана мова програмування TypeScript та PHP, каскадна мова стилів CSS та препроцесор Sass для візуальної складової веб-додатку, мова розмітки HTML у якості “каркасу”, а також база даних використовуючи СКБД MySQL.

Даний програмний продукт призначений для того, щоб будь-який користувач міг легко та зручно навчатися новим знанням в області української мови завдяки зручному інтерфейсу, який практично дозволяє інтерактивно проходити уроки та відслідковувати свій прогрес у реальному часі.

Під час розробки кваліфікаційної роботи був виконаний наступний перелік задач:

- ✓ Визначено предметну область задачі;
- ✓ Створено дизайн веб-додатку;
- ✓ Виконано проектування внутрішньої архітектури системи;
- ✓ Створено серверну частину додатку;
- ✓ Створено клієнтську частину додатку;
- ✓ Розроблено базу даних та локальне сховище даних;
- ✓ Визначено трудомісткість розробленого додатку;
- ✓ Підрахована загальна вартість додатку.

В результаті, розроблений веб-застосунок дозволяє рендерити усі необхідні елементи веб-сторінки, надає можливість зручної навігації по сайту, інтерактивно взаємодіяти з компонентами системи та зберігати усі внесені дані.

Застосування мови TypeScript допомогло набагато пришвидшити написання коду веб-додатку та зекономити багато часу, а мова PHP дозволила

реалізувати багато важких програмних рішень. Тому час створення застосунку становить 49,3 місяця, а вартість додатку становить 1455629,05 грн.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Документація TypeScript - https://www.typescriptlang.org/docs/
2) TypeScript для початківців - https://codeguida.com/post/475
3) Документація PHP - https://www.php.net/manual/ru/index.php
4) Посібник для розробників - https://developer.mozilla.org/en-US/
5) Документація Sass - https://sass-lang.com/documentation/
6) Документація по розробці баз даних MySQL - https://bohdan-books.com/upload/iblock/ebe/ebe73ff67804bc6dc1f6f464a3ca4469.pdf
7) Як підключати БД використовуючи мову програмування PHP - https://www.php.net/manual/ru/mysqli.quickstart.statements.php
8) Д. Б. Буй, А.В. Пузікова. Теорія нормалізації в реляційних базах даних: сучасний стан 10с.
9) Що таке MVC простими словами - https://ru.hexlet.io/blog/posts/chto-takoe-mvc-rasskazyvaem-prostymi-slovami
10) Використання MVC архітектури - https://habr.com/ru/post/249263/
11) Архітектура веб-застосунків: моделі, типи, визначення - https://hackr.io/blog/web-application-architecture-definition-models-types-and-more

ДОДАТОК А

КОД ПРОГРАМИ

Зміст файлу Ukrusuu.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="Ukrusuu - Улучши свой уровень украинского языка!">
    <meta name="keywords" content="Ukrusuu, Онлайн-обучение украинскому языку, Изучение украинского языка, Улучшение владения украинским языком">
    <title>Ukrusuu - Улучши свой уровень украинского языка!</title>
    <link rel="stylesheet" id="styles" href="./Css/main.css">
</head>
<body>
    <div class="layout">
        <header class="header">
            <div class="header__logotype">
                <a href="./Ukrusuu.html" class="header__link">Ukrusuu</a>
            </div>
            <nav class="header__menu">
                <ul class="header__list">
                    <li class="header__item" id="main">
                        <i class="fa-solid fa-house header__icon"></i>
                        Главная
                    </li>
                    <li class="header__item" id="lessons">
                        <i class="fa-solid fa-book-open header__icon header__icon_position"></i>
```

```
                Уроки
            </li>
            <li class="header__item" id="repetition">
                <i class="fa-solid fa-arrow-up-right-dots header__icon"></i>
                Повторение
            </li>
            <li class="header__item" id="support">
                <i class="fa-solid fa-circle-question header__icon header__icon_size"></i>
                Поддержка
            </li>
            <div class="header__controls">
                <button class="header__route" id="signUp">Регистрация</button>
                <button class="header__route" id="signIn">Войти</button>
            </div>
        </ul>
    </nav>
</header>
<div class="content">
    <section class="intro">
        <div class="container">
            <div class="intro__main">
                <div class="intro__info">
                    <h1>Выучи украинский язык, занимаясь по 10 минут в день</h1>
                    <p class="intro__text">Ставь достижимые цели. Совершенствуй свои навыки. Учись на ошибках. Добивайся большего. Открой для себя мир новых возможностей.</p>
                    <div class="intro__line"></div>
                    <div class="intro__action">
                        <button class="intro__btn">
                            Начать учиться
                            <i class="fa-solid fa-rocket intro__icon"></i>
                        </button>
```

```
                <p class="intro__smallText">Развивайся вместе с Ukrusuu!</p>
              </div>
            </div>
            <div class="intro__pic"></div>
          </div>
        </div>
      </section>
      <section class="causes">
        <div class="container">
          <div class="causes__main">
            <h2 class="causes__title">Зачем учить украинский язык?</h2>
            <div class="causes__item">
              <div class="causes__tile causes__tile_margin">
                <div class="causes__shell">
                  <h3 class="causes__name">Языковые преимущества</h3>
                  <div class="causes__line"></div>
                  <p class="causes__desc">Украинский входит в категорию славянских языков. Это означает, что зная украинский, ты сможешь лучше понимать польский, чешский, болгарский и хорватский языки, и при необходимости, освоить эти языки намного быстрее и эффективнее, нежели без знания украинского.</p>
                  <span class="causes__msg">Осваивай новые горизонты.</span>
                </div>
              </div>
              <div class="causes__tile">
                <div class="causes__shell">
                  <h3 class="causes__name">Европейские возможности</h3>
                  <div class="causes__line"></div>
                  <p class="causes__desc">Украина подписала ассоциацию с ЕС на место полноправного члена. Имея гражданство в Украине, это даст возможность жить в правовом государстве и свободно путешествовать по всей Европе. Получение гражданства требует знание украинского языка.</p>
                  <span class="causes__msg">Не упусти такую возможность.</span>
                </div>
              </div>
```

```
            </div>
            <div class="causes__item">
              <div class="causes__tile">
                <div class="causes__shell">
                  <h3 class="causes__name">Духовные и культурные богатства</h3>
                  <div class="causes__line"></div>
                  <p class="causes__desc">Украина дала огромное количество великих поэтов и писателей, а они, в свою очередь, литературные шедевры. Благодаря знаниям украинского, вы сможете комфортно вдохновляться произведениями Тараса Шевченко, Леси Украинки, Олександра Довженко и других.</p>
                  <span class="causes__msg">Развивай свое воображение.</span>
                </div>
              </div>
              <div class="causes__tile">
                <div class="causes__shell">
                  <h3 class="causes__name">Живописный диалект</h3>
                  <div class="causes__line"></div>
                  <p class="causes__desc">Украина получила второе место на конкурсе благозвучания и мелодичности языков мира в Париже, где оценивалось красота звучания. Украинский язык был назван мелодичным, богатым и разнообразным языком любви.</p>
                  <span class="causes__msg">Заговори любовью.</span>
                </div>
              </div>
            </div>
          </div>
        </section>
        <section class="parallax">
          <svg class="parallax__waves" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" viewBox="0 24 150 28" preserveAspectRatio="none" shape-rendering="auto">
            <defs>
              <path id="wave" d="M-160 44c30 0 58-18 88-18s 58 18 88 18 58-18 88-18 58 18 88 18 v44h-352z"></path>
```

```
        </defs>
        <g class="parallax__animation">
          <use xlink:href="#wave" x="48" y="0" fill="rgba(37, 104, 237, 0.3)" class="parallax__wave"></use>
          <use xlink:href="#wave" x="48" y="5" fill="rgba(37, 104, 237, 0.1)" class="parallax__wave"></use>
          <use xlink:href="#wave" x="48" y="3" fill="rgba(37, 104, 237, 0.2)" class="parallax__wave"></use>
          <use xlink:href="#wave" x="48" y="7" fill="rgba(37, 104, 237, 0.4)" class="parallax__wave"></use>
        </g>
      </svg>
    </section>
    <section class="features">
      <div class="container">
        <h3 class="features__title">Обучение в Ukrusuu - это:</h3>
        <div class="features__row">
          <div class="features__info features__info_position">
            <span class="features__name">Эффективный план уроков</span>
            <h3 class="features__subname">Учись в своём темпе</h3>
            <p class="features__desc">Занимайся, когда удобно тебе, и запоминай больше благодаря коротким урокам, разработанным экспертами в области лингвистики. Добейся реального прогресса в своем уровне знаний.</p>
          </div>
          <div class="features__demo">
            <h5 class="features__progress">Твой прогресс - 0%</h5>
            <div class="features__shell">
              <div class="features__circle">
                <img src="./Images/Main-page/Lesson-preview1.jpg" alt="Обложка-превью урока" class="features__preview">
              </div>
              <p class="features__lesson">Урок 1</p>
            </div>
            <div class="features__shell">
              <div class="features__circle">
```

<img src="./Images/Main-page/Lesson-preview2.jpg" alt="Обложка-превью урока" class="features__preview">

Урок 2

<img src="./Images/Main-page/Lesson-preview3.jpg" alt="Обложка-превью урока" class="features__preview">

Урок 3

<img src="./Images/Main-page/Lesson-preview4.jpg" alt="Обложка-превью урока" class="features__preview">

Урок 4

<img src="./Images/Main-page/Lesson-preview5.jpg" alt="Обложка-превью урока" class="features__preview">

Урок 5

Умные инструменты обучения

<h3 class="features__subname">Учись эффективно</h3>

```
                <p class="features__desc">Следуй плану обучения, запоминай сложные слова и практикуйся в их произношении благодаря технологиям, которые подскажут тебе, что и когда надо повторить или подтянуть.</p>
            </div>
            <div class="features__dictionary">
                <div class="features__power">
                    <div class="features__extra">
                        <div class="features__word features__word_style_red">
                            <span class="features__num">5</span>
                        </div>
                        <span class="features__strength">Слабые слова</span>
                    </div>
                    <div class="features__extra">
                        <div class="features__word features__word_style_yellow">
                            <span class="features__num">10</span>
                        </div>
                        <span class="features__strength">Средние слова</span>
                    </div>
                    <div class="features__extra">
                        <div class="features__word features__word_style_green">
                            <span class="features__num">15</span>
                        </div>
                        <span class="features__strength">Сильные слова</span>
                    </div>
                </div>
                <div class="features__line"></div>
                <div class="features__list">
                    <div class="features__cover">
                        <img src="./Images/Word-preview.jpg" alt="Обложка-превью слова" class="features__wordPreview">
                    </div>
                    <div class="features__text">
```

```
                    <h5 class="features__wordName">Привіт!</h5>
                    <span class="features__wordTranslation">Привет!</span>
                </div>
                <i class="fa-solid fa-volume-high features__icon"></i>
            </div>
        </div>
    </div>
</section>
<section class="accession">
    <div class="container">
        <div class="accession__main">
            <div class="accession__shell">
                <img src="./Images/Ukrusuu-logo.png" alt="Логотип Ukrusuu" class="accession__img">
            </div>
            <h2 class="accession__title">Присоединяйся к учащимся на Ukrusuu</h2>
            <div class="accession__action">
                <button class="accession__btn">
                    Начать учиться
                    <i class="fa-solid fa-rocket accession__icon"></i>
                </button>
            </div>
        </div>
    </div>
</section>
<footer class="credits">
    <div class="container">
        <div class="credits__main">
            <span>©2023 Ukrusuu</span>
            <div>
                <span class="credits__link credits__link_entrance">Войти</span>
```

```
                <span class="credits__link credits__link_support">Связаться с нами</span>
              </div>
            </div>
          </div>
        </footer>
        <nav class="viewport__nav">
          <div class="viewport__mobilenav">
            <div class="viewport__mobile">
              <li class="header__item header__item_mobile" id="main">
                <i class="fa-solid fa-house header__icon"></i>
                <span class="viewport__mobileSpan">Главная</span>
              </li>
              <li class="header__item header__item_mobile" id="lessons">
                <i class="fa-solid fa-book-open header__icon header__icon_position"></i>
                <span class="viewport__mobileSpan">Уроки</span>
              </li>
              <li class="header__item header__item_mobile" id="repetition">
                <i class="fa-solid fa-arrow-up-right-dots header__icon"></i>
                <span class="viewport__mobileSpan">Повторение</span>
              </li>
              <li class="header__item header__item_mobile" id="support">
                <i class="fa-solid fa-circle-question header__icon header__icon_size"></i>
                <span class="viewport__mobileSpan">Поддержка</span>
              </li>
            </div>
          </div>
        </nav>
      </div>
    </div>
    <script src="https://kit.fontawesome.com/62f4bec11e.js" crossorigin="anonymous"></script>
```

```
    <script src="https://code.jquery.com/jquery-3.7.0.min.js" integrity="sha256-
2Pmvv0kuTBOenSvLm6bvfBSSHrUJ+3A7x6P5Ebd07/g=" crossorigin="anonymous"></script>
    <script src="./Scripts/Js/vocabulary.js"></script>
    <script src="./Scripts/Js/main.js"></script>
</body>
</html>
```

Зміст файлу main.ts

```
'use strict';
window.scrollTo({
    top: 0,
    behavior: "smooth"
});
let accountStatus,
    access,
    authSession;
(function checkStorage() {
    if (localStorage.getItem('status') != null && localStorage.getItem('access') != null) {
        accountStatus = parseInt(localStorage.getItem('status'));
        access = parseInt(localStorage.getItem('access'));
        return;
    }

    else {
        accountStatus = 0;
        access = 0;
        localStorage.setItem('status', JSON.stringify(accountStatus));
        localStorage.setItem('access', JSON.stringify(access));
    };
}());
(function checkAuth() {
```

```
    if (Boolean(parseInt(localStorage.getItem('status'))) === true) {
        document.querySelector('#signIn').textContent = "Мой аккаунт";
        document.querySelector('#signIn').setAttribute('id', 'signed');
        setAuthListener();
        document.querySelector('#signUp').remove();
    }
    else {
        localStorage.removeItem('auth');
        setListenerToBtnReg();
        setListenerToBtnEntrance();
        return;
    };
    authSession = localStorage.getItem('auth');
}());
function checkAccess() {
    if (Boolean(parseInt(localStorage.getItem('access'))) === false) {
        document.querySelector('#lessons').classList.add('header__item_disabled');
        document.querySelector('#lessons').setAttribute('id', 'lessons_closed');
        document.querySelector('#repetition').classList.add('header__item_disabled');
        document.querySelector('#repetition').setAttribute('id', 'repetition_closed');
    } else {
        return;
    };
};
checkAccess();
(function checkMobileMenuAccess() {
    let mobileMenu: any = document.querySelector('.viewport__mobile') as HTMLElement,
        mobileMenuStyles: any = getComputedStyle(mobileMenu),
        mobileMenuDisplay: any = mobileMenuStyles.getPropertyValue('display');
    if (mobileMenuDisplay !== "none") {
        checkAccess();
```

```
    } else {
        return;
    };
} () );
(function checkIsPassed() {
    if (localStorage.getItem('isPassed') !== null) {
        localStorage.removeItem('isPassed');
    } else {
        return;
    };
}());
window.addEventListener('storage', function () {
    switch (true) {
        case (localStorage.getItem('status') !== accountStatus):
            localStorage.setItem('status', JSON.stringify(accountStatus));
        case (localStorage.getItem('access') !== access):
            localStorage.setItem('access', JSON.stringify(access));
        case (localStorage.getItem('auth') !== authSession):
            if (localStorage.getItem('auth') !== null) {
                localStorage.setItem('auth', authSession);
            }
            else {
                if (authSession !== undefined) {
                    localStorage.setItem('auth', authSession);
                }
                else {
                    return;
                };
            };
    };
});
```

```
setListenerToMainBtn();
const regexp1 = /\./g,
    regexp2 = /\?/g,
    regexp3 = /\*|!|,|:|;|"|'|-|`|\$/g;
function setListenersToMenuItems() {
    document.querySelectorAll('.header__item').forEach(element => {
        element.addEventListener('click', function () {
            checkItemId(element.id);
        });
    });
};
setListenersToMenuItems();
function checkItemId(id: string) {
    switch (id) {
        case ("main"):
            renderMainPage();
            break;

        case ("lessons"):
            renderLessonsPage();
            break;

        case ("repetition"):
            renderRepetitionPage();
            break;

        case ("support"):
            renderSupportPage();
            break;

        case ("lessons_closed"):
```

```
            alert('Войдите в аккаунт, чтобы начать учиться!');
            break;

        case ("repetition_closed"):
            alert('Войдите в аккаунт, чтобы начать учиться!');
            break;
    };
};
function renderMainPage() {
    document.querySelector('.content').innerHTML = createMainPage();
    document.querySelector('#styles').setAttribute('href', './Css/main.css');
    window.scrollTo({
        top: 0
    });
    setListenerToMainBtn();
    let mobileMenu: any = document.querySelector('.viewport__mobile') as HTMLElement,
        mobileMenuStyles: any = getComputedStyle(mobileMenu),
        mobileMenuDisplay: any = mobileMenuStyles.getPropertyValue('display');

    if (mobileMenuDisplay !== "none") {
        checkAccess();
        setListenersToMenuItems();
    } else {
        return;
    };
};
function setListenerToMainBtn() {
    document.querySelector('.intro__btn').addEventListener('click', function () {
        if (Boolean(parseInt(localStorage.getItem('status'))) === true && Boolean(parseInt(localStorage.getItem('access')))
=== true) {
            renderLessonsPage();
```

```
    } else {
        document.querySelector('.layout').innerHTML = createEntranceContent();
        document.querySelector('#styles').setAttribute('href', './Css/account.css')
        document.querySelector('#closeAcc').addEventListener('click', function () {
            if (confirm('Отменить все изменения и вернуться на главную страницу?')) {
                document.querySelector('.layout').innerHTML = returnToHomePage();
                document.querySelector('#styles').setAttribute('href', './Css/main.css');
                setListenersToMenuItems();
                setListenerToBtnReg();
                setListenerToBtnEntrance();
                checkAccess();
                setListenerToMainBtn();
            } else {
                return;
            };
        });

        enterAccount();
    };
});
document.querySelector('.accession__btn').addEventListener('click', function () {
    if (Boolean(parseInt(localStorage.getItem('status'))) === true && Boolean(parseInt(localStorage.getItem('access')))
=== true) {
        renderLessonsPage();
    } else {
        document.querySelector('.layout').innerHTML = createEntranceContent();
        document.querySelector('#styles').setAttribute('href', './Css/account.css')
        document.querySelector('#closeAcc').addEventListener('click', function () {
            if (confirm('Отменить все изменения и вернуться на главную страницу?')) {
                document.querySelector('.layout').innerHTML = returnToHomePage();
                document.querySelector('#styles').setAttribute('href', './Css/main.css');
```

```
            setListenersToMenuItems();
            setListenerToBtnReg();
            setListenerToBtnEntrance();
            checkAccess();
            setListenerToMainBtn();
        } else {
            return;
        };
    });
    enterAccount();
};
});
document.querySelector('.credits__link_entrance').addEventListener('click', function () {
    if (Boolean(parseInt(localStorage.getItem('status'))) === true && localStorage.getItem('auth') !== null) {
        document.querySelector('.layout').innerHTML = createSignedContent();

        document.querySelector('#styles').setAttribute('href', './Css/account.css');
        document.querySelector('#signUp').remove();
        document.querySelector('#signIn').textContent = "Мой аккаунт";
        document.querySelector('#signIn').setAttribute('id', 'signed');
        setAuthListener();
        setListenersToMenuItems();
        let userPhoto: any = document.querySelector('#userPhoto'),
            userEmail: any = document.querySelector('#userEmail'),
            userLogin: any = document.querySelector('#userLogin');
        $.ajax({
            type: 'POST',
            url: './Php/getUserData.php',
            cache: false,
            data: {
                email: JSON.parse(localStorage.getItem('auth'))
```

```
        },
        success(data) {
            let userObject = JSON.parse(data);
            if (userObject.photo !== null) {
                userPhoto.setAttribute('src', userObject.photo);
            };
            userEmail.value = userObject.email;
            userLogin.value = userObject.username;
        },
        error() {
            alert("Произошла ошибка соединения. Страница будет перезагружена.");
            location.reload();
        },
    });

    document.querySelector('#choosePhoto').addEventListener('change', choosePhoto);
    editAccount();
    setListenerToEditPassword()
    setListenerToDeleteAccount();
    setListenerToAccountExit();
} else {
    document.querySelector('.layout').innerHTML = createEntranceContent();
    document.querySelector('#styles').setAttribute('href', './Css/account.css')
    document.querySelector('#closeAcc').addEventListener('click', function () {
        if (confirm('Отменить все изменения и вернуться на главную страницу?')) {
            document.querySelector('.layout').innerHTML = returnToHomePage();
            document.querySelector('#styles').setAttribute('href', './Css/main.css');
            setListenersToMenuItems();
            setListenerToBtnReg();
            setListenerToBtnEntrance();
            checkAccess();
```

```
                setListenerToMainBtn();
            } else {
                return;
            };
        });
        enterAccount();
    };
});
document.querySelector('.credits__link_support').addEventListener('click', function () {
    renderSupportPage();
});
};
```

Зміст файлу createNewAccount.php

```
<?php
    $connection = mysqli_connect("ieugene.mysql.tools", "ieugene_admin", "23042000Eugennyy_Admin", "ieugene_ukrusuu");
    $answer = false;
    mysqli_set_charset($connection, "UTF8");
    if (!$connection) {
        echo "Возникла проблема во время установки соединения с базой данных. Пожалуйста, перезагрузите страницу.";
        exit;
    };
    if (isset($_POST['email']) && isset($_POST['login']) && isset($_POST['password'])) {
        $email = trim($_POST['email']);
        $login = trim($_POST['login']);
        $password = trim($_POST['password']);
        $passwordHash = md5($password);
```

```
    $query = mysqli_query($connection, "INSERT INTO `Users` SET `id` = not null, `email` = '$email', `username` = '$login', `pass` = '$passwordHash'");
    if ($query) {
        $answer = true;
        echo $answer;
        exit;
    }
    else {
        echo $answer;
        exit;
    };
  }
  else {
    exit;
  };
  mysqli_close($connection);
?>
```

Зміст файлу getUserData.php

```
<?php
  $connection = mysqli_connect("ieugene.mysql.tools", "ieugene_admin", "23042000Eugennyy_Admin", "ieugene_ukrusuu");
  mysqli_set_charset($connection, "UTF8");
  if (!$connection) {
    echo "Возникла проблема во время установки соединения с базой данных. Пожалуйста, перезагрузите страницу.";
    exit;
  };
  if (isset($_POST['email'])) {
    $email = trim($_POST['email']);
    $dbData = mysqli_query($connection, "SELECT `email`, `username`, `photo` FROM `Users` WHERE `email` = '$email'");
```

```
    while ($row = $dbData->fetch_assoc()) {
        $data = $row;
    };
    echo json_encode($data);
};
?>
```

Зміст файлу createNewUserProgress.php

```
<?php
$connection = mysqli_connect("ieugene.mysql.tools", "ieugene_admin", "23042000Eugennyy_Admin", "ieugene_ukrusuu");
$answer = false;
mysqli_set_charset($connection, "UTF8");
if (!$connection) {
    echo "Возникла проблема во время установки соединения с базой данных. Пожалуйста, перезагрузите страницу.";
    exit;
};
if (isset($_POST['userid']) && isset($_POST['progress']) && isset($_POST['lessonid']) && isset($_POST['chapterid']) && isset($_POST['isdone'])) {
    $userId = trim($_POST['userid']);
    $progress = trim($_POST['progress']);
    $lessonId = trim($_POST['lessonid']);
    $chapterId = trim($_POST['chapterid']);
    $isDone = trim($_POST['isdone']);
    $query = mysqli_query($connection, "INSERT INTO `Users_progress` SET `id` = not null, `userId` = $userId, `progress` = '$progress', `lessonId` = $lessonId, `chapterId` = $chapterId, `isLessonDone` = $isDone");
    if ($query) {
        $answer = true;
        echo $answer;
        exit;
    }
```

```
        else {
            echo $answer;
            exit;
        };
    }
    else {
        exit;
    };
    mysqli_close($connection);
?>
```

Зміст файлу main.css

```
@import
url("https://fonts.googleapis.com/css2?family=Josefin+Sans&family=Montserrat:wght@300&display=swap");
.features__title, .causes__title {
  font-size: 2.3rem;
  line-height: 2.5625rem;
  letter-spacing: 1px;
}
.intro__pic {
  background-size: cover;
  background-repeat: no-repeat;
  background-position: center center;
}
 * {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Montserrat", sans-serif;
```

```
  color: rgb(75, 87, 102);
}
a {
  text-decoration: none;
  outline: none;
}
ul, ol, li {
  list-style-type: none;
}
svg {
  pointer-events: none;
}
button {
  outline: none;
  border: none;
  pointer-events: all;
  cursor: pointer;
  -moz-user-select: none;
      user-select: none;
  -moz-appearance: none;
      appearance: none;
  -webkit-user-select: none;
  -webkit-appearance: none;
}
h1 {
  font-size: 2.5rem;
  line-height: 3.25rem;
  font-weight: 800;
  margin-bottom: 1rem;
  letter-spacing: 0.3px;
}
```

```
.header {
  display: flex;
  position: fixed;
  top: 0;
  left: 0;
  right: 0;
  box-shadow: 0 0 10px 0 rgba(0, 0, 0, 0.2);
  border-bottom: 1px solid rgb(218, 225, 234);
  padding: 0.5rem 2rem;
  height: 4rem;
  align-items: center;
  z-index: 999999;
  background-color: #fff;
}
.header__menu {
  width: 100%;
}
.header__logotype {
  display: inline-flex;
  margin-right: 1rem;
  border-bottom: 1px solid rgb(17, 110, 238);
  -webkit-user-select: none;
     -moz-user-select: none;
          user-select: none;
}
.header__link {
  color: rgb(17, 110, 238);
  font-size: 1.406rem;
  text-transform: uppercase;
  letter-spacing: 1px;
  font-weight: bold;
```

```
  padding-bottom: 1.5px;
}
.header__list {
  display: flex;
}
.header__item {
  display: inline-flex;
  align-items: center;
  margin: 0.25rem;
  padding: 0.7rem 0.625rem;
  font-size: 0.875rem;
  font-weight: bold;
  cursor: pointer;
  outline: none;
  border-radius: 0.5rem;
  background-color: transparent;
  transition: all 0.2s ease-in-out 0s;
}
.header__item:hover {
  color: rgb(17, 110, 238);
  background-color: rgb(240, 247, 255);
}
.header__item:hover .header__icon {
  color: rgb(17, 110, 238);
}
.header__item:active {
  background-color: rgb(240, 247, 255);
  color: rgb(17, 110, 238);
}
.header__item:active .header__icon {
  color: rgb(17, 110, 238);
```

```
}
.header__item_disabled {
  background-color: rgba(218, 225, 234, 0.8);
  cursor: default;
}
.header__item_disabled:hover {
  color: rgb(75, 87, 102);
  background-color: rgba(218, 225, 234, 0.8);
  cursor: default;
}
.header__item_disabled:hover .header__icon {
  color: rgb(75, 87, 102);
  cursor: default;
}
.header__item_disabled:active {
  color: rgb(75, 87, 102);
  background-color: rgba(218, 225, 234, 0.8);
  cursor: default;
}
.header__item_disabled:active .header__icon {
  color: rgb(75, 87, 102);
  cursor: default;
}
.header__icon {
  font-size: 1.1rem;
  margin-right: 0.75rem;
  transition: color 0.2s ease-in-out 0s;
}
.header__icon_position {
  margin-top: 1px;
}
```

```
.header__icon_size {
  font-size: 1.15rem;
}
.header__controls {
  display: inline-flex;
  align-items: center;
  justify-content: flex-end;
  width: 100%;
}
.header__route {
  margin: 0 0.625rem;
  padding: 0.375rem 1.75rem;
  font-size: 0.875rem;
  border: 0.125rem solid rgb(17, 110, 238);
  background-color: transparent;
  font-weight: bold;
  color: rgb(17, 110, 238);
  border-radius: 2.815rem;
  text-align: center;
  -webkit-appearance: none;
  cursor: pointer;
  word-wrap: break-word;
  transition-property: background-color, color;
  transition: 0.2s ease-in-out;
}
.header__route:hover {
  background-color: rgb(17, 110, 238);
  color: #fff;
}
.header__route:active {
  background-color: rgb(17, 110, 238);
```

```
  color: #fff;
}
.header__route:first-child {
  background-color: rgb(17, 110, 238);
  color: #fff;
}
.header__route:first-child:hover {
  background-color: transparent;
  color: rgb(17, 110, 238);
}
.header__route:first-child:active {
  background-color: transparent;
  color: rgb(17, 110, 238);
}
.container {
  max-width: 78.125rem;
  width: 100%;
  margin: 0 auto;
  padding: 0 2.5rem 0 2.5rem;
  height: 100%;
}
.content {
  padding-top: 3.938rem;
}
.intro {
  background-color: #fff;
  box-shadow: 0 2px 5px rgba(0, 0, 0, 0.25), 0 4px 6px rgba(0, 0, 0, 0.25);
}
.intro__main {
  display: flex;
  align-items: center;
```

```
}
.intro__info {
  padding: 1.875rem 0 1.25rem 0;
}
.intro__text {
  font-size: 1.125rem;
  line-height: 1.7;
  font-weight: 600;
  max-width: 29.375rem;
  margin-bottom: 1.85rem;
  letter-spacing: 0.5px;
}
.intro__line {
  width: 5rem;
  height: 2px;
  background-color: rgb(75, 87, 102);
  margin-bottom: 2rem;
}
.intro__action {
  min-height: 6.25rem;
  max-height: 6.25rem;
  height: 100%;
  position: relative;
}
.intro__btn {
  left: 0;
  top: 0;
}
.intro__btn:hover {
  box-shadow: none;
  margin-top: 0.37em;
```

```
}
.intro__btn:active {
  background-color: rgba(75, 87, 102, 0.2);
}
.intro__icon {
  font-size: 1.15rem;
  margin-left: 5px;
}
.intro__smallText {
  font-size: 0.813rem;
  font-weight: bold;
  letter-spacing: 0.5px;
  position: absolute;
  left: 0;
  bottom: 0;
}
.intro__pic {
  max-width: 40.625rem;
  min-height: 40.625rem;
  width: 100%;
  height: 100%;
  background-image: url(../Images/Woman-photo.webp);
  background-position: center top;
}
.causes {
  margin-top: 0.938rem;
  background-color: #fff;
}
.causes__main {
  display: flex;
  flex-direction: column;
```

```
  margin: 0 auto;
  max-width: 56.25rem;
  width: 100%;
  padding-top: 3.125rem;
}
.causes__title {
  text-align: center;
  padding-bottom: 3.438rem;
}
.causes__item {
  display: flex;
  -webkit-user-select: none;
     -moz-user-select: none;
          user-select: none;
}
.causes__tile {
  width: 50%;
  min-height: 20.313rem;
  height: 100%;
  border-left: 2px solid #dad9d7;
  border-bottom: 2px solid #dad9d7;
  line-height: 1.5;
  position: relative;
}
.causes__tile:first-child {
  border-left: none;
}
.causes__tile:before {
  content: "";
  background-color: #fff;
  position: absolute;
```

```
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  transition-property: box-shadow, transform;
  transition: 0.3s ease-out;
  cursor: pointer;
  border-radius: 3px;
  z-index: 1;
}
.causes__tile:hover:before {
  box-shadow: 0 7px 21px 0 rgba(0, 0, 0, 0.25);
  transform: scale(1.05, 1.05);
  -webkit-transform: scale(1.05, 1.05);
  z-index: 5;
}
.causes__shell {
  position: relative;
  z-index: 5;
  padding: 1.563rem;
  cursor: pointer;
}
.causes__name {
  font-size: 1.1rem;
  letter-spacing: 0.5px;
}
.causes__line {
  width: 2.813rem;
  height: 2px;
  background-color: rgb(75, 87, 102);
  margin: 0.938rem 0 0.938rem 0;
```

```
}
.causes__desc {
  letter-spacing: 0.3px;
}
.causes__msg {
  display: inline-block;
  margin-top: 0.938rem;
  padding-bottom: 4.5px;
  font-weight: bold;
  font-size: 0.813rem;
  letter-spacing: 0.2px;
  border-bottom: 0.5px solid rgb(75, 87, 102);
}
.parallax {
  padding-top: 1.875rem;
}
.parallax__wave {
  animation: endless-wave-motion 25s cubic-bezier(0.55, 0.5, 0.45, 0.5) infinite;
  -webkit-animation: endless-wave-motion 25s cubic-bezier(0.55, 0.5, 0.45, 0.5) infinite;
}
.parallax__wave:nth-child(1) {
  animation-delay: -2s;
  animation-duration: 7s;
  -webkit-animation-delay: -2s;
  -webkit-animation-duration: 7s;
}
.parallax__wave:nth-child(2) {
  animation-delay: -3s;
  animation-duration: 10s;
  -webkit-animation-delay: -3s;
  -webkit-animation-duration: 10s;
```

```
}
.parallax__wave:nth-child(3) {
  animation-delay: -4s;
  animation-duration: 13s;
  -webkit-animation-delay: -4s;
  -webkit-animation-duration: 13s;
}
.parallax__wave:nth-child(4) {
  animation-delay: -5s;
  animation-duration: 20s;
  -webkit-animation-delay: -5s;
  -webkit-animation-duration: 20s;
}
@keyframes endless-wave-motion {
  0% {
    transform: translate3d(-90px, 0, 0);
    -webkit-transform: translate3d(-90px, 0, 0);
  }
  100% {
    transform: translate3d(85px, 0, 0);
    -webkit-transform: translate3d(85px, 0, 0);
  }
}
.features {
  margin-top: -4px;
  padding-top: 13rem;
  padding-bottom: 50px;
  background: linear-gradient(to bottom, rgba(37, 104, 237, 0.7) 0, rgba(255, 255, 255, 0) 50%);
  background: -webkit-gradient(linear, left top, left bottom, color-stop(0, rgba(37, 104, 237, 0.7)), color-stop(50%, rgba(255, 255, 255, 0)));
}
```

```
.features__title {
  padding-bottom: 3.75rem;
}
.features__row {
  display: flex;
  justify-content: space-around;
  align-items: center;
  padding-top: 3.75rem;
}
.features__row:first-of-type {
  padding-top: 0;
}
.features__info {
  width: 50%;
  background-color: #fff;
  border: 1px solid rgb(75, 87, 102);
  box-shadow: 5px 5px 3px rgba(75, 87, 102, 0.5);
  padding: 2.188rem 1.563rem 2.188rem 1.563rem;
}
.features__info_position {
  order: 2;
}
.features__info_shadow {
  box-shadow: -5px 5px 3px rgba(75, 87, 102, 0.5);
}
.features__name {
  display: inline-block;
  text-transform: uppercase;
  font-size: 0.719rem;
  line-height: 1.125rem;
  color: #666e7e;
```

```
  font-weight: bold;
  letter-spacing: 0.3px;
  border-bottom: 0.5px solid rgb(75, 87, 102);
  padding-bottom: 2px;
  margin-bottom: 1.031rem;
}
.features__subname {
  font-size: 1.5rem;
  line-height: 2.25rem;
  font-weight: bold;
  letter-spacing: 0.5px;
  margin-bottom: 1.031rem;
}
.features__desc {
  letter-spacing: 0.2px;
}
.features__demo {
  padding: 1.031rem 1.25rem 1.031rem 1.25rem;
  transform: rotate(-2.5deg);
}
.features__progress {
  font-size: 0.813rem;
  letter-spacing: 0.4px;
}
.features__shell {
  display: flex;
  align-items: center;
  padding: 0.531rem 0 0.531rem 0;
  margin-top: 5px;
  border-bottom: 1px solid rgba(0, 0, 0, 0.2);
}
```

```
.features__circle {
  border-radius: 50%;
  width: 2.188rem;
  height: 2.188rem;
  -o-object-fit: cover;
     object-fit: cover;
}
.features__preview {
  width: 100%;
  height: 100%;
  border-radius: inherit;
}
.features__lesson {
  font-weight: bold;
  font-size: 0.719rem;
  letter-spacing: 0.2px;
  padding-left: 0.625rem;
}
.features__dictionary {
  padding: 1.25rem;
  transform: rotate(-3deg);
}
.features__power {
  display: flex;
}
.features__extra {
  display: flex;
  flex-direction: column;
  margin-left: 0.719rem;
}
.features__extra:first-child {
```

```
  margin-left: 0;
}
.features__word {
  width: 4.688rem;
  height: 3.438rem;
  border-radius: 4.5px;
  background-color: #fff;
  box-shadow: 0 2px 3px rgba(0, 0, 0, 0.25);
  position: relative;
}
.features__word_style_red::before {
  background-color: rgba(255, 0, 0, 0.8);
  height: 50%;
}
.features__word_style_yellow::before {
  background-color: rgba(255, 209, 46, 0.9);
  height: 75%;
}
.features__word_style_green::before {
  background-color: #26b883;
  height: 100%;
  border-radius: inherit;
}
.features__num {
  position: absolute;
  bottom: 2.5px;
  left: 5px;
  color: #fff;
  font-size: 0.875rem;
  font-weight: bold;
  letter-spacing: 0.3px;
```

```
}
.features__strength {
  font-size: 0.563rem;
  letter-spacing: 0.1px;
  font-weight: bold;
  padding-top: 0.375rem;
}
.features__line {
  width: 100%;
  height: 1px;
  background-color: #dad9d7;
  margin: 0.625rem 0 0.625rem 0;
}
.features__list {
  display: flex;
  align-items: center;
}
.features__cover {
  width: 2.813rem;
  height: 1.875rem;
  border-radius: 4.5px;
  -o-object-fit: cover;
     object-fit: cover;
}
.features__wordPreview {
  width: 100%;
  height: 100%;
  border-radius: inherit;
}
.features__text {
  display: flex;
```

```
  flex-direction: column;
  padding-left: 0.625rem;
}
.features__wordName {
  font-size: 0.75rem;
  letter-spacing: 0.2px;
}
.features__wordTranslation {
  font-size: 0.719rem;
  letter-spacing: 0.2px;
  padding-top: 2px;
}
.features__icon {
  font-size: 0.625rem;
  color: rgba(75, 87, 102, 0.6);
  margin-left: 0.406rem;
}
.accession {
  background-color: #5271ff;
}
.accession__main {
  display: flex;
  flex-direction: column;
  align-items: center;
  padding-bottom: 3.75rem;
}
.accession__shell {
  max-width: 23.75rem;
  max-height: 23.75rem;
  height: 100%;
  -o-object-fit: cover;
```

```
    object-fit: cover;
  -webkit-user-select: none;
     -moz-user-select: none;
          user-select: none;
}
.accession__img {
  width: 100%;
  height: 100%;
}
.accession__title {
  font-size: 2.25rem;
  line-height: 1.5;
  letter-spacing: 1px;
  color: #fff;
  margin-top: -3.438rem;
}
.accession__action {
  display: flex;
  justify-content: center;
  width: 100%;
  min-height: 4.063rem;
  max-height: 4.063rem;
  height: 100%;
  position: relative;
  margin-top: 2.5rem;
}
.accession__btn {
  top: 0;
}
.accession__btn:hover {
  box-shadow: none;
```

```
  margin-top: 0.37em;
}
.accession__btn:active {
  background-color: rgba(255, 255, 255, 0.5);
}
.accession__icon {
  font-size: 1.15rem;
  margin-left: 5px;
}
.credits {
  background-color: #fff;
  padding: 1.563rem 0 1.563rem 0;
  font-size: 0.906rem;
  font-weight: bold;
  letter-spacing: 0.2px;
}
.credits__main {
  display: flex;
  justify-content: space-between;
}
.credits__link {
  display: inline-block;
  cursor: pointer;
  margin-left: 1.25rem;
  padding-bottom: 0.5px;
}
.credits__link:first-child {
  margin-left: 0;
}
.credits__link:after {
  content: "";
```

```
  display: block;
  border-bottom: 1px solid rgb(75, 87, 102);
  transform: scaleX(0);
  transition: transform 0.2s ease-in-out;
}
.credits__link:hover:after {
  transform: scaleX(1);
}
.viewport__mobile {
  display: none;
}
```

ДОДАТОК Б

ВІДГУК

на кваліфікаційну роботу бакалавра на тему: «Розробка онлайн-сервісу для вивчення української мови» студента групи 122-20ск Кречуняка Євгенія Альбертовича

ДОДАТОК В

РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра на тему: «Розробка онлайн-сервісу для вивчення української мови» студента групи 122-20ск Кречуняка Євгенія Альбертовича

ДОДАТОК Г

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК Д

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файла	Опис
Пояснювальні документи	
Кваліфікаційна робота.docx	Пояснювальна записка до кваліфікаційної роботи в форматі Word документу.
Кваліфікаційна робота.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF.
Програма	
Ukrusuu.rar	Архів проекту. Містить в собі весь код веб-додатку.
Ukrusuu_DB.sql	База даних MySQL проекту.
Презентація	
Презентація.pptx	Презентація кваліфікаційної роботи.