ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка застосунку для навчання маленьких дітей літерам та звукам»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Тимофій ЯРИНА
(підпис)

Виконав: здобувач вищої освіти групи ПД-42

Тимофій ЯРИНА

Керівник: Наталя АНДРУСЕВИЧ
ст. викладач

Рецензент: _____

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
« ____ » _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Ярині Тимофію Олеговичу

1. Тема кваліфікаційної роботи: «Розробка застосунку для навчання маленьких дітей літерам та звукам»
керівник кваліфікаційної роботи ст. викладач кафедри ІПЗ Наталя АНДРУСЕВИЧ, затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025 р. № 56.

2. Строк подання кваліфікаційної роботи «02» червня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: теоретичні відомості про методи навчання маленьких дітей, опис методів навчання маленьких дітей, технічна документація з описом бібліотек.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Огляд та аналіз існуючих застосунків для навчання маленьких дітей літерам та звукам.
2. Проектування застосунку для навчання маленьких дітей літерам та звукам".

3. Програмна реалізація та опис функціонування застосунку для навчання маленьких дітей літерам та звукам.
4. Тестування застосунку.

5. Перелік графічного матеріалу: *презентація*
1. Аналіз аналогів.
2. Вимоги за стосунку.
3. Програмні засоби реалізації.
4. Діаграма варіантів використання.
5. Діаграма послідовності.
6. Діаграма пакетів.
7. Блок-схема алгоритму роботи вікторини.
8. Екранні форми.
9. Апробація результатів дослідження

6. Дата видачі завдання «25» лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	25.02.-04.03.2025	
2	Аналіз та дослідження існуючих аналогів	05.03-13.03.2025	
3	Огляд та аналіз застосунку для навчання маленьких дітей літерам та звукам.	14.03-20.03.2025	
4	Огляд та аналіз засобів реалізації застосунку для навчання маленьких дітей літерам та звукам.	21.03-28.03.2025	
5	Програмна реалізація модулів: абетка для вивчення літер та звуків, вікторина, детальне вивчення літер.	29.03-18.04.2025	
6	Тестування реалізованої системи та її компонентів.	19.04-10.05.2025	
7	Оформлення роботи: вступ, висновки, реферат	11.05-12.05.2025	
8	Розробка демонстраційних матеріалів	13.05-18.05.2025	
9	Попередній захист роботи	19.05-30.05.2025	

Здобувач вищої освіти _____ Тимофій ЯРИНА
(підпис)

Керівник
кваліфікаційної роботи _____ Наталя АНДРУСЕВИЧ
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 50 стор., 2 табл., 27 рис 20 джерел.

Мета роботи – вдосконалення навчання маленьких дітей літерам та звукам.

Об'єкт дослідження – процеси навчання маленьких дітей літерам та звукам.

Предмет дослідження – вебзастосунок для навчання маленьких дітей літерам та звукам.

Короткий зміст роботи: В роботі описано підходи до створення вебзастосунку, який покращує навчання маленьких дітей літерам та звукам.

Розроблено алгоритм роботи застосунку та реалізовані ключові функціональні можливості, такі як: інтерактивне навчання літерам та звукам, малювання літер, вибір мови, прослуховування англійської та українською абеток, вікторина із вгадуванням літер. Проведено функціональне та мануальне тестування додатку.

Розроблено застосунок використовуючи Java, HTML, CSS та JavaScript з використанням інтерактивних бібліотек, серверна частина розроблена на Spring Boot. Особливу увагу приділено зручному та інтуїтивно зрозумілому інтерфейсу, адаптованому під вікові особливості цільової аудиторії.

У роботі використано сучасні технології та методики розробки, що дозволяє легко адаптувати систему під майбутні розширення функціоналу.

Сферою використання застосунку є домашнє навчання, дошкільні навчальні заклади.

КЛЮЧОВІ СЛОВА: JAVA, JAVASCRIPT, SPRING BOOT, HTML, CSS.

ЗМІСТ

ВСТУП

У сучасному світі цифрові технології відіграють дедалі важливішу роль у процесі навчання. Зокрема, інтерактивні освітні застосунки для дітей молодшого віку стають ефективним інструментом пізнання та розвитку. Дошкільний вік є критичним періодом для формування мовленнєвих навичок, засвоєння звуків та ознайомлення з літерами, що в подальшому закладає основу для читання, письма й загального інтелектуального розвитку. У зв’язку з цим розробка навчального застосунку, який поєднує ігрову подачу інформації з аудіовізуальними елементами, є актуальним завданням.

Метою кваліфікаційної роботи є створення інтерактивного застосунку для навчання дітей дошкільного віку літерам і звукам української та англійської мов. Застосунок має допомогти дитині у легкій і зрозумілій формі засвоїти алфавіт, розпізнавати звуки, асоціювати літери з відповідними словами та зображеннями, а також розвивати фонематичний слух.

У процесі розробки було враховано особливості сприйняття інформації дітьми цього віку: простий інтерфейс, яскравий візуальний стиль, інтерактивність, а також наявність звукового супроводу. Важливим аспектом стала підтримка мультимовності, що дозволяє дитині не лише вивчати рідну мову, а й знайомитися з основами іноземної.

У роботі описано етапи проєктування, реалізації та тестування застосунку, розглянуто використані технології, зокрема HTML, CSS, JavaScript, а також фреймворки й бібліотеки, необхідні для реалізації звукових ефектів і адаптивного дизайну. Проведено функціональне тестування за методом "чорного ящика" з метою перевірки відповідності програмного продукту початковим вимогам.

1 ОГЛЯД ТА АНАЛІЗ ІСНУЮЧИХ ЗАСОБІВ ДЛЯ НАВЧАННЯ МАЛЕНЬКИХ ДІТЕЙ ЛІТЕРАМ ТА ЗВУКАМ

1.1 Особливості розробки програмного забезпечення для навчання маленьких дітей літерам та звукам

Застосунок для навчання маленьких дітей літерам та звукам — це програма, яка допоможе маленьким дітям у легкій та захоплюючій формі ознайомитися з основами читання. Завдяки цьому застосунку діти зможуть вивчати літери, їхнє звучання та основні правила звуко-буквеної відповідності у зручному для себе ритмі, незалежно від місця перебування чи часу доби. Усі навчальні елементи розроблені так, щоб відповідати психологічним та віковим особливостям дітей дошкільного віку, стимулювати інтерес до пізнання та розвивати мовлення.

Інтерактивні вправи дозволяють дитині краще засвоювати матеріал, адже вони поєднують візуальні, слухові та моторні елементи, що є надзвичайно важливим для комплексного розвитку. Всі завдання розподілено за категоріями — вивчення літер, розпізнавання звуків, фонематичний слух — це дозволяє зробити навчання послідовним, логічним і ефективним.

Основними користувачами застосунку є діти віком від 3 до 6 років — періоду, коли формується мовлення, активізується розвиток пам’яті та уваги, а навчання повинно відбуватись у формі гри. Саме тому інтерфейс програми створено максимально простим, яскравим та адаптованим, з урахуванням особливостей сприйняття маленьких дітей.

Основа методики навчання ґрунтується на поєднанні візуального та аудіального сприйняття, що відповідає особливостям розвитку дітей дошкільного віку. Для ефективного засвоєння матеріалу використовуються картки з літерами та зображеннями предметів, які починаються на відповідні звуки. Додаток містить озвучення літер з можливістю багаторазового повторення, що сприяє формуванню правильної звуковимови.

Навчальний процес подається у формі ігрових завдань: діти обирають потрібну літеру серед кількох варіантів, малюють літери, прослуховують абетку. У застосунку передбачена можливість зміни мови інтерфейсу та навчального контенту на англійську. Це дає змогу дітям не лише вивчати українську абетку, а й знайомитися з англійською алфавітом, звуками англійської мови. Навчання проходить у такій же інтерактивній формі: з картками, озвученням, іграми, які допомагають дитині поступово засвоювати англійську абетку, розвивати фонематичний слух і навички правильної вимови.

Це особливо корисно для дітей, які планують у майбутньому вивчати англійську мову, або для сімей з двомовним вихованням. Така функція робить застосунок універсальним і відкриває широкі можливості для раннього вивчення іноземної мови.

1.2 Аналіз аналогів

Аналіз існуючих застосунків для навчання маленьких дітей літерам і звукам є ключовим етапом у процесі проєктування та розробки нових навчальних програм. Цей процес дозволяє не лише зібрати корисну інформацію про функціональні можливості наявних рішень, але й виявити слабкі місця, обмеження та незадоволені потреби користувачів. Завдяки цьому розробники отримують змогу створити конкурентоспроможний продукт, що буде краще відповідати потребам цільової аудиторії — дітей дошкільного віку та молодших школярів, а також їхніх батьків і педагогів.

Під час аналізу варто звертати увагу на такі аспекти, як:

- чи охоплюються застосунок всі літери алфавіту, як саме подається інформація (через зображення, звуки, анімації);
- наскільки зручний інтерфейс користувача, зрозумілий та адаптований для маленьких дітей;
- яка мова доступна у застосунку, наявність аудіо супроводу;
- наявність ігрових режимів для перевірки знань.

1.2.1 Вчимося читати – вчимо літери

Мобільний застосунок «Вчимося читати – вчимо літери» розроблений спеціально для дітей віком від 3 до 6 років. Його основна мета — навчити дитину розпізнавати літери, слова за допомогою гри[1].

Рис. 1.1 Головний екран Вчимося читати – вчимо літери

Освітній процес подано у вигляді сюжетної гри, де дитина допомагає тваринам вибратися з крижаного полону, виконуючи навчальні завдання. Такий ігровий підхід стимулює інтерес до навчання і забезпечує емоційну залученість, що особливо важливо для дітей дошкільного віку. Застосунок доступний українською мовою.

Рис. 1.2 Екран вікторини

Переваги:

- застосунок пропонує інтерактивні вправи, які допомагають дітям легко засвоювати букви та звуки через гру з вгадування звірів;
- проста та зрозуміла навігація, великі кнопки роблять користування комфортним;
- кожна літера супроводжується чітким звуковим відтворенням, що сприяє розвитку фонематичного слуху і правильної вимови.

Недоліки:

- застарілий інтерфейс користувача — хоча дизайн яскравий, деякі елементи виглядають технічно застарілими й неадаптованими для нових екранів з високою роздільною здатністю;
- повільне завантаження екранів — іноді спостерігається затримка при переході між рівнями або відкритті нових розділів.
- обмеженість функціоналу — застосунок має лише один ігровий режим;
- обмежена підтримка мовних варіантів, лише українська мова.

1.2.2 Українська абетка для дітей

Українська абетка для дітей — це застосунок розроблений українською компанією Onartsoft, який допомагає малюкам ознайомитися з літерами українського алфавіту у веселій та доступній формі. Програма пропонує два режими, серед яких — вивчення української абетки з аудіо супроводом кожної літери, а також проста гра у вигляді збирання пазла[2].

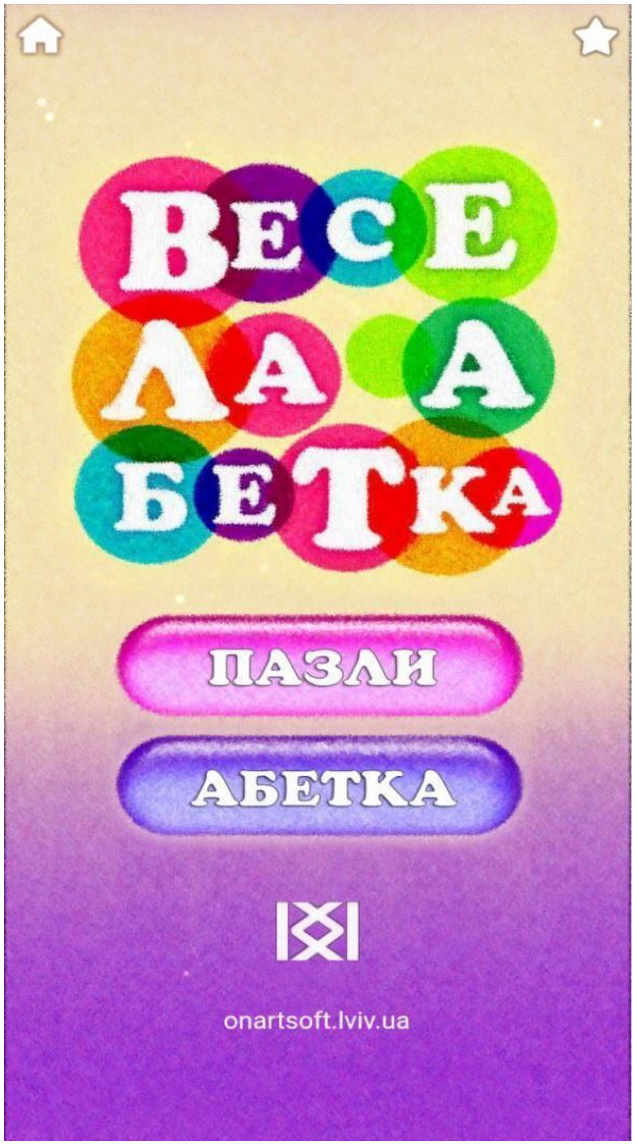


Рис. 1.3 Головний екран Українська абетка для дітей

Кожна літера супроводжується тематичним зображенням, що дозволяє дитині асоціювати зображення з початковою літерою.

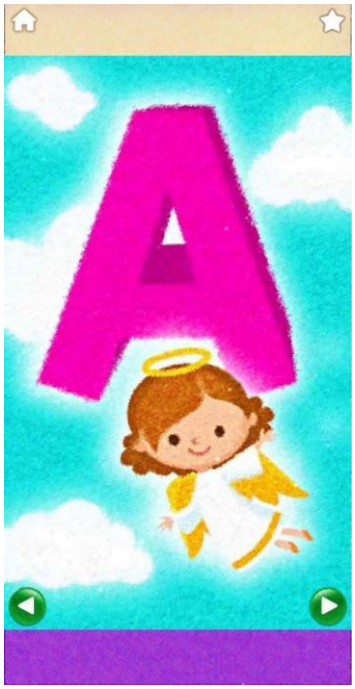


Рис. 1.4 Екран з літерами у розділі абетки

Яскравий, дружній та інтуїтивно зрозумілий інтерфейс створений з урахуванням вікових особливостей дітей, що дозволяє їм легко орієнтуватися в додатку без допомоги дорослих, а весела музика і звукові ефекти підтримують увагу та інтерес протягом усього навчального процесу.

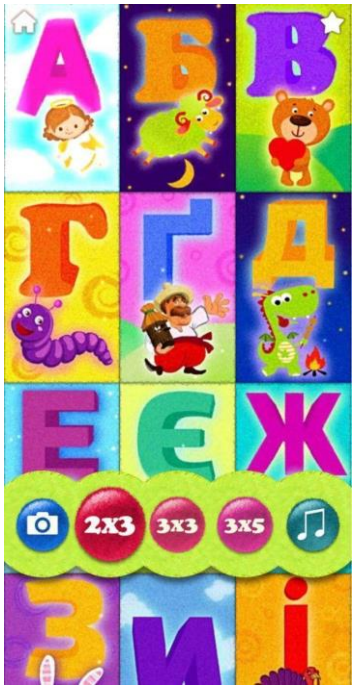


Рис. 1.4 Екран розділу зі збиранням пазлів

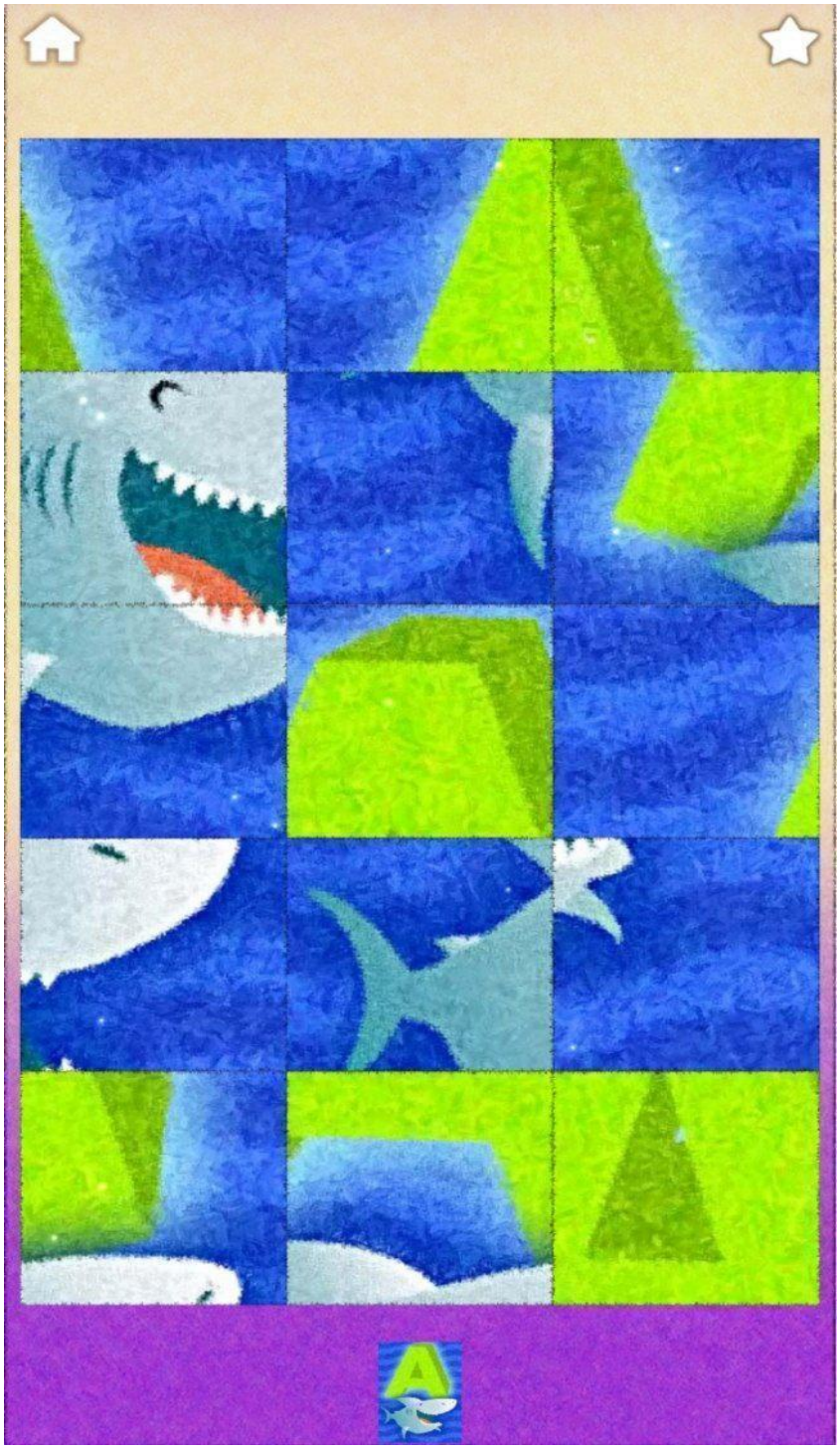


Рис. 1.5 Екран прикладу зі збирання пазла

Переваги:

- можливість повторення матеріалу та поступове ускладнення завдань для адаптації до рівня дитини;
- інтерфейс адаптований під вікові критерії;

- ігровий режим зі збирання пазлів.

Недоліки:

- застарілий інтерфейс — елементи виглядають технічно застарілими;
- обмежена кількість інтерактивних елементів;
- відсутність перевірки знань.

1.2.3 Розумники(Smart Kids)

«Розумники (Smart Kids)» розроблений українським мультимедійним видавництвом «Розумники», яке спеціалізується на створенні інтерактивних освітніх ресурсів для початкової школи. Проєкт реалізується за підтримки Міністерства освіти і науки України та Національної академії педагогічних наук України, зокрема Інституту цифровізації освіти. Застосунок «Розумники» є частиною всеукраїнського педагогічного експерименту, спрямованого на впровадження технологій навчання учнів початкової школи за допомогою електронних освітніх ігрових ресурсів[3].

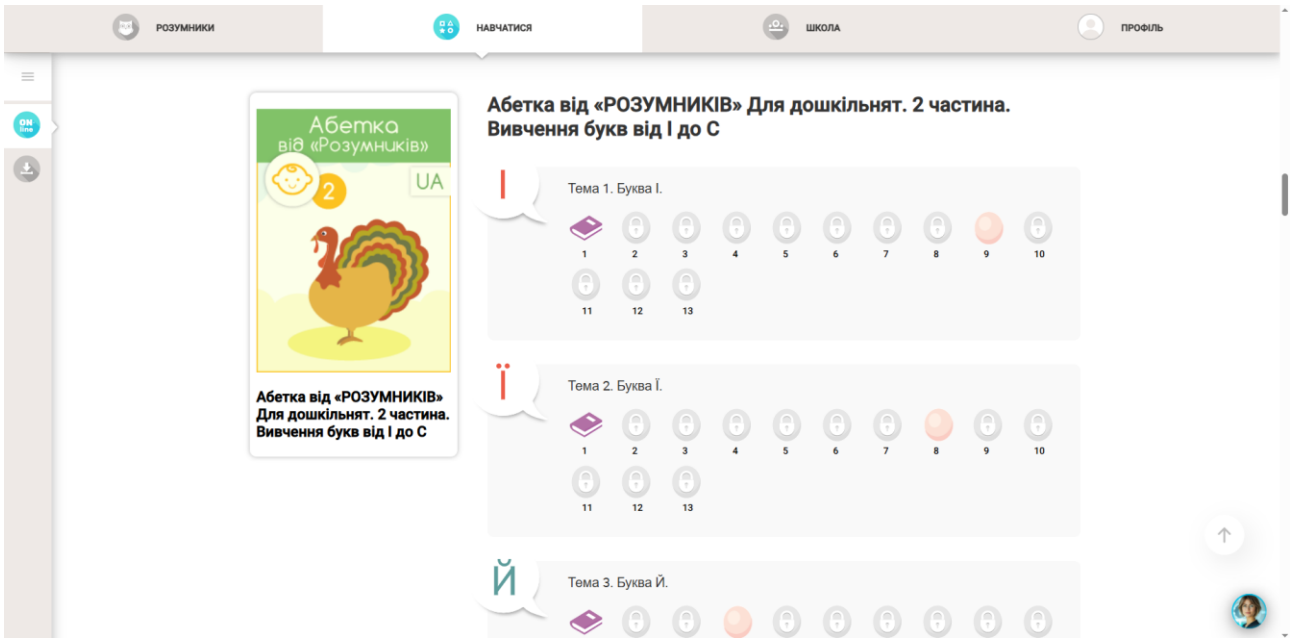


Рис. 1.6 Головний екран Розумники (Smart Kids)

Особливістю «Розумників» є його модульність — додаткові навчальні блоки можуть завантажуватись окремо, що дозволяє розширювати функціональність відповідно до потреб дитини. Деякі модулі передбачають інтеграцію із системою контролю знань, що дає змогу батькам або вчителям відстежувати прогрес учня.

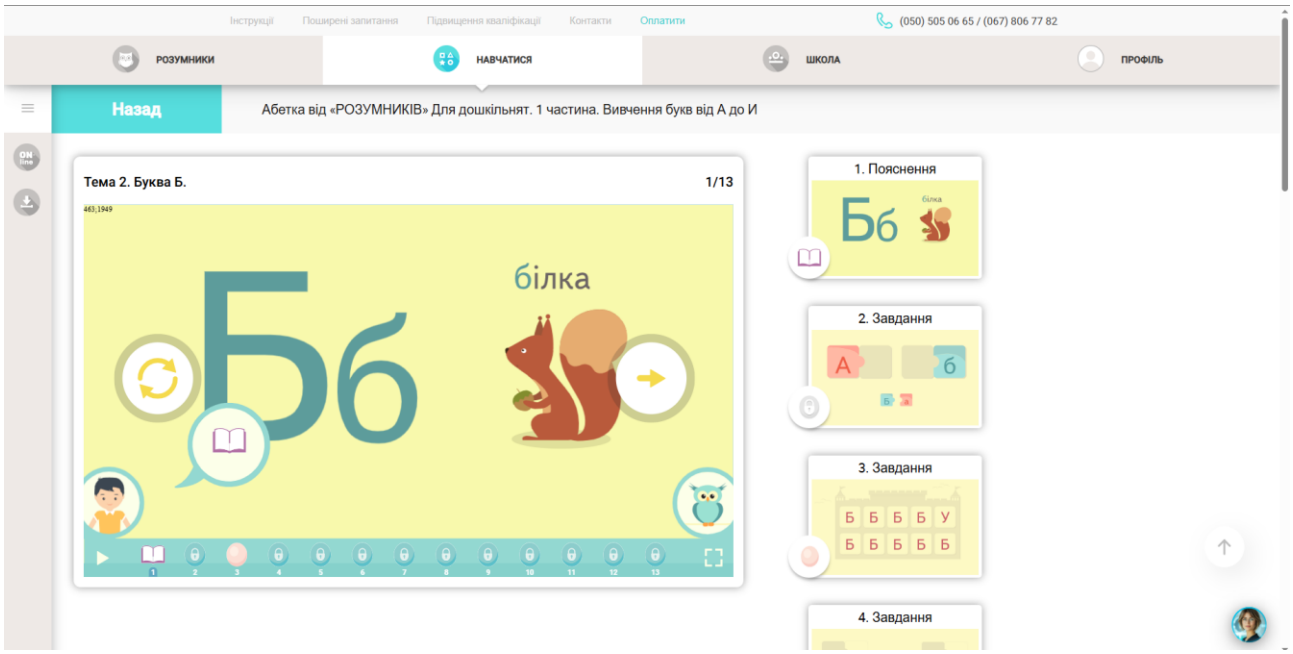


Рис. 1.7 Екран із розділу з вивчення літер

Застосунок адаптований для самостійного використання дитиною: всі елементи меню інтуїтивно зрозумілі, управління здійснюється через великі кнопки з малюнками, а інструкції озвучуються голосом. Завдяки цьому діти можуть навчатися без постійної участі дорослих.

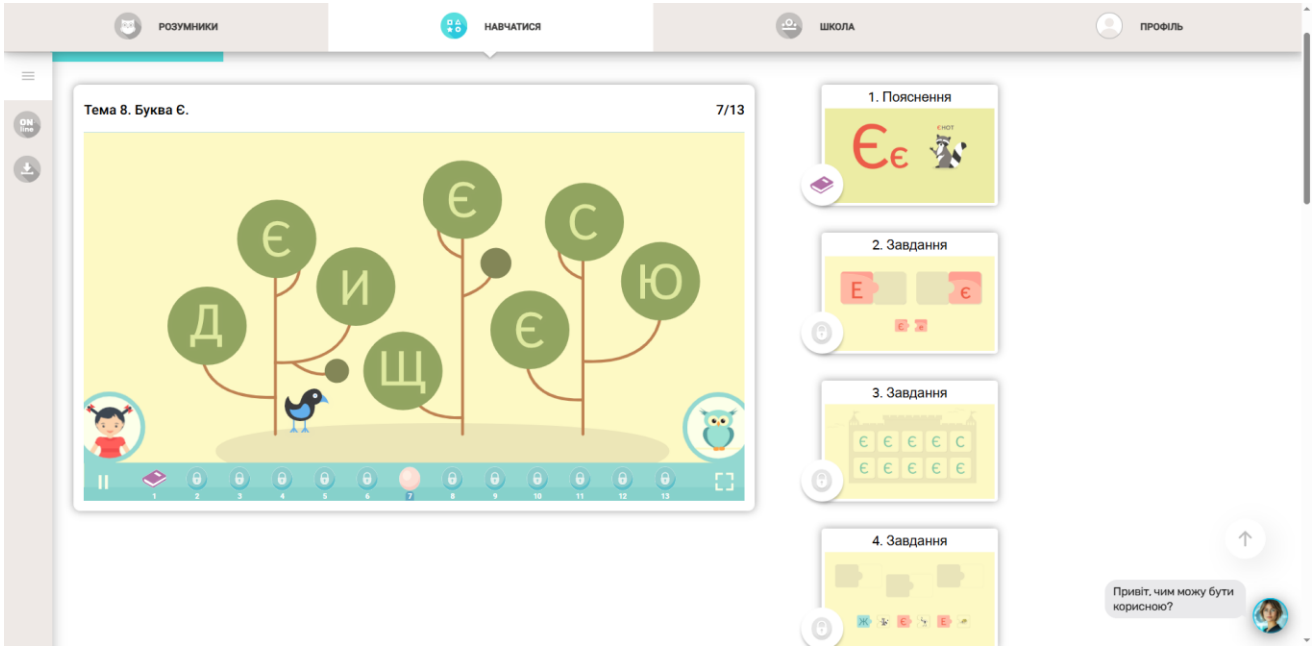


Рис. 1.8 Екран із розділу з вивчення літер, «Знайди літер»

Видавництво «Розумники» також проводить тренінги для вчителів у рамках проєкту «Smart Kids», забезпечуючи педагогів необхідними знаннями та навичками для ефективного використання електронних ресурсів у навчальному процесі. Інтерфейс програми барвистий, з великою кількістю ілюстрацій.

Переваги:

- багатофункціональний додаток, що охоплює різні напрямки розвитку дитини;
- розділ із вивченням літер містить як візуальні, так і аудіальні елементи навчання;
- система контролю знань для батьків і педагогів.

Недоліки:

- більшість функції та модулів є платними, що обмежує доступ до повного обсягу навчального контенту;
- через велику кількість тем додаток може здаватися перевантаженим для дітей, які тільки починають знайомство з абеткою;
- відсутність підтримки іноземної мови.

1.3 Зведений аналіз аналогів

З метою виявлення сильних і слабких сторін існуючих рішень у сфері навчання дітей дошкільного віку вивченню літер та звуків було проведено аналіз трьох найпопулярніших мобільних додатків: Вчимося читати — вчимо літери, Українська абетка для дітей та Розумники (Smart Kids). Усі три застосунки орієнтовані на формування у дітей розвитку фонематичного слуху та ознайомлення з літерами української абетки. Проте кожен із них має свої особливості реалізації, функціональні обмеження та переваги.

«Вчимося читати — вчимо літери» — це простий у використанні додаток, що має інтуїтивно зрозумілий інтерфейс та якісне озвучення. Його основною перевагою є доступність для самостійного використання дитиною без участі дорослих. Проте додаток має лише один режим навчання, обмежену кількість вправ і доволі застаріле візуальне оформлення, що знижує мотивацію до тривалого використання.

«Українська абетка для дітей», розроблений компанією Onarsoft, пропонує барвистий дизайн, озвучення професійними дикторами та вправи на складання пазлів. Додаток створено з урахуванням вікових особливостей дітей, що сприяє кращому засвоєнню матеріалу. Проте функціональність обмежена, відсутня перевірка знань, наприклад — вікторина.

«Розумники (Smart Kids) » — це освітній застосунок від українського видавництва «Розумники», який входить до всеукраїнського проєкту з цифровізації початкової освіти. Він охоплює широкий спектр предметів, зокрема й українську абетку.

Перевагами є модульність (можливість додаткового завантаження навчальних блоків), інтеграція з системою контролю знань і підтримка педагогів. Проте, немає підтримки іноземних мов, що є важливим фактором для двомовних сімей, або дітей за кордоном.

Результати зведеного аналізу застосунків для навчання маленьких дітей літерам та звукам наведено в таблиці 1.1.

Таблиця 1.1

Результати зведеного аналізу застосунків для навчання маленьких дітей літерам та звукам

Критерії	Вчимося читати — вчимо літери	Українська абетка для дітей	Розумники (Smart Kids)
Платформа	Android,iOS	Android,iOS	Web
Мовна доступність	Українська	Українська	Українська
Наявність абетки	ні	так	так
Перевірка знань	так	ні	так
Аудіо супровід	так	так	так
Адаптація до віку	ні	ні	так
Дизайн	Застарілий	Застарілий	Сучасний

2 ЗАСОБИ РОЗРОБКИ ТА ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Обґрунтування вибору мови програмування Java

Java — це одна з найстаріших, найстабільніших і водночас найактуальніших мов програмування у світі. Її створили в середині 1990-х років компанією Sun Microsystems (тепер належить корпорації Oracle), і з того часу вона стала основою для мільйонів програмних систем, від банківського ПЗ до мобільних застосунків. Її головна концепція — "Write once, run anywhere" (Напиши один раз — запускай всюди), що означає: код, написаний мовою Java, можна запускати на будь-якому пристрої або операційній системі, де встановлена Java Virtual Machine (JVM). Це забезпечує максимальну переносимість та гнучкість, особливо при розробці кросплатформених систем.

Java є об’єктно-орієнтованою мовою, що сприяє логічному та структурованому підходу до проєктування програм. Це дозволяє легко створювати масштабовані модульні додатки, які зручно підтримувати й розвивати.

Мова має великий набір стандартних бібліотек, які дозволяють виконувати широкий спектр задач — від роботи з файлами і базами даних до побудови графічних інтерфейсів користувача та реалізації мережевих з'єднань. Це значно скорочує час розробки, оскільки багато типових функціональних блоків уже реалізовано.

Java активно використовується в критично важливих галузях — фінансах, телекомунікаціях, електронному урядуванні — саме через свою надійну систему обмеження доступу до ресурсів, багаторівневу обробку винятків, контроль пам’яті JVM і відсутність прямого доступу до пам’яті (як у C/C++).

Для додатків, які взаємодіють із персональними даними користувачів (наприклад, освітні застосунки з профілями дітей або батьків), ці властивості є ключовими[4].

2.2 Обґрунтування вибору середовища розробки IntelliJ IDEA

У процесі створення програмного забезпечення особливу роль відіграє правильний вибір інтегрованого середовища розробки (IDE), яке забезпечить зручність програмування, ефективне налагодження та швидкий доступ до інструментів для управління проєктом. Одним із найкращих середовищ для розробки програм на мові Java є IntelliJ IDEA, розроблене компанією JetBrains.

IntelliJ IDEA — це потужне, інтелектуальне середовище розробки, яке надає широкий спектр функцій для ефективної роботи з Java, а також з іншими мовами програмування. Воно стало галузевим стандартом для Java-розробників завдяки своїй стабільності, зручному інтерфейсу, багатій функціональності та регулярній підтримці оновлень.

Однією з основних переваг IntelliJ IDEA є автоматизація рутинних процесів та інтелектуальне доповнення коду. Середовище самостійно підказує назви змінних, методів, класів, а також пропонує швидкі варіанти виправлення помилок. Завдяки цьому суттєво зменшується кількість синтаксичних та логічних помилок, а розробник може зосередитись на логіці застосунку, а не на технічних деталях.

Іншою перевагою є глибока інтеграція з фреймворками, зокрема Spring Framework, а також з системами збірки (Maven, Gradle), системами контролю версій (Git), інструментами тестування (JUnit) та базами даних (MySQL). Це забезпечує повноцінний цикл розробки в одному середовищі: від написання та тестування коду — до збірки, налагодження та розгортання проєкту.

Інструменти швидкого рефакторингу, навігації по коду, візуалізації структури проєкту, автогенерації коду, шаблонів і конфігурацій значно підвищують продуктивність розробника. Вбудований емулятор запуску проєктів і серверів дозволяє швидко тестувати зміни без потреби в зовнішніх сервісах.

Наявність як безкоштовної версії (Community Edition), так і розширеної комерційної версії (Ultimate Edition) робить IntelliJ IDEA доступним як для індивідуальних розробників, так і для великих команд. При цьому навіть

безкоштовна версія забезпечує більшість необхідних функцій для розробки повноцінного Java-додатку [5].

2.3 Обґрунтування вибору фреймворку Spring Framework

Spring Framework — це один з найпопулярніших фреймворків у Java-екосистемі, який забезпечує зручні засоби для розробки надійних, масштабованих та безпечних веб-додатків. Він базується на принципах інверсії управління (IoC), аспектно-орієнтованого програмування (AOP) та модульності, що дозволяє створювати гнучкі й добре структуровані системи.

Spring забезпечує усю необхідну інфраструктуру для серверної логіки: обробка запитів, маршрутизація, валідація, робота з базами даних, безпека, API-інтерфейси тощо. Завдяки модулю Spring Boot розробка стає ще простішою — багато налаштувань конфігурується автоматично, що дозволяє швидко створити працюючий застосунок із мінімальним обсягом шаблонного коду.

Однією з найбільших переваг є його інтеграція з іншими технологіями. Наприклад, він чудово поєднується з MySQL для збереження та обробки даних, підтримує REST API для взаємодії з frontend'ом (написаним, наприклад, з використанням HTML, CSS та JavaScript), а також забезпечує захист через Spring Security.

Завдяки розподіленості структури, Spring дає змогу розробникам використовувати лише ті компоненти, які їм потрібні, що робить проєкти менш громіздкими й легшими в обслуговуванні. Spring активно підтримується спільнотою, постійно оновлюється, і має багату документацію, що особливо важливо для стабільної роботи програмного забезпечення[6].

2.4 Система керування базами даних: MySQL

MySQL — це реляційна система керування базами даних з відкритим вихідним кодом, яка є однією з найпопулярніших у світі завдяки своїй

продуктивності, надійності та простоті у використанні. У контексті створення освітнього застосунку для дітей MySQL використовується як основне сховище даних, де зберігається інформація про користувачів, їхній прогрес, налаштування, результати проходження вправ тощо.

Однією з ключових переваг MySQL є її висока швидкодія навіть при роботі з великими обсягами даних. Це особливо актуально в освітніх проєктах, де потрібно оперативно зчитувати й оновлювати дані про активність великої кількості користувачів. Крім того, MySQL добре масштабується, що дозволяє в перспективі обслуговувати більшу кількість користувачів без втрати продуктивності.

Ще однією важливою перевагою MySQL є її повна сумісність зі Spring Framework, завдяки чому інтеграція з backend-частиною відбувається швидко та без ускладнень. За допомогою ORM-бібліотек на кшталт Hibernate або Spring Data JPA можна ефективно працювати з базою даних, не пишучи надто багато SQL-коду вручну.

MySQL також підтримує високий рівень безпеки завдяки механізмам автентифікації, контролю доступу та шифрування з’єднань. Це важливо при роботі з персональними даними користувачів, особливо дітей[7].

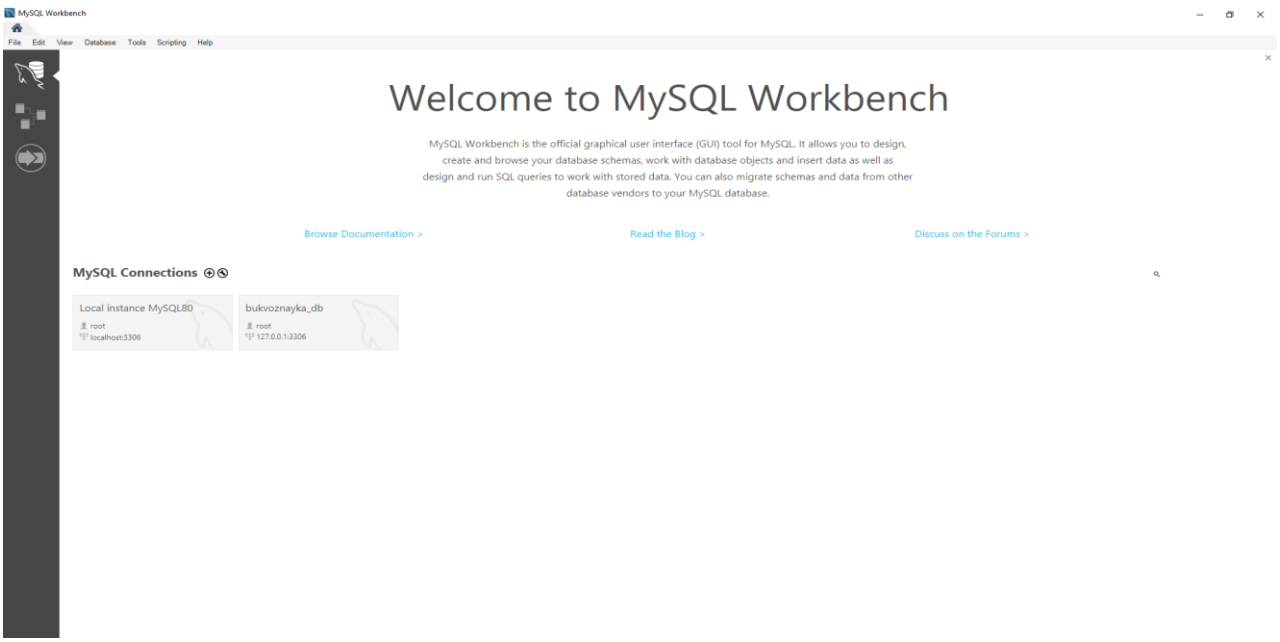


Рис. 2.1 Інтерфейс MySQL

Серед інших переваг — інструменти адміністрування, такі як MySQL Workbench, які дозволяють зручно проєктувати структуру бази даних, візуалізувати зв’язки між таблицями та виконувати запити вручну для тестування або налагодження.

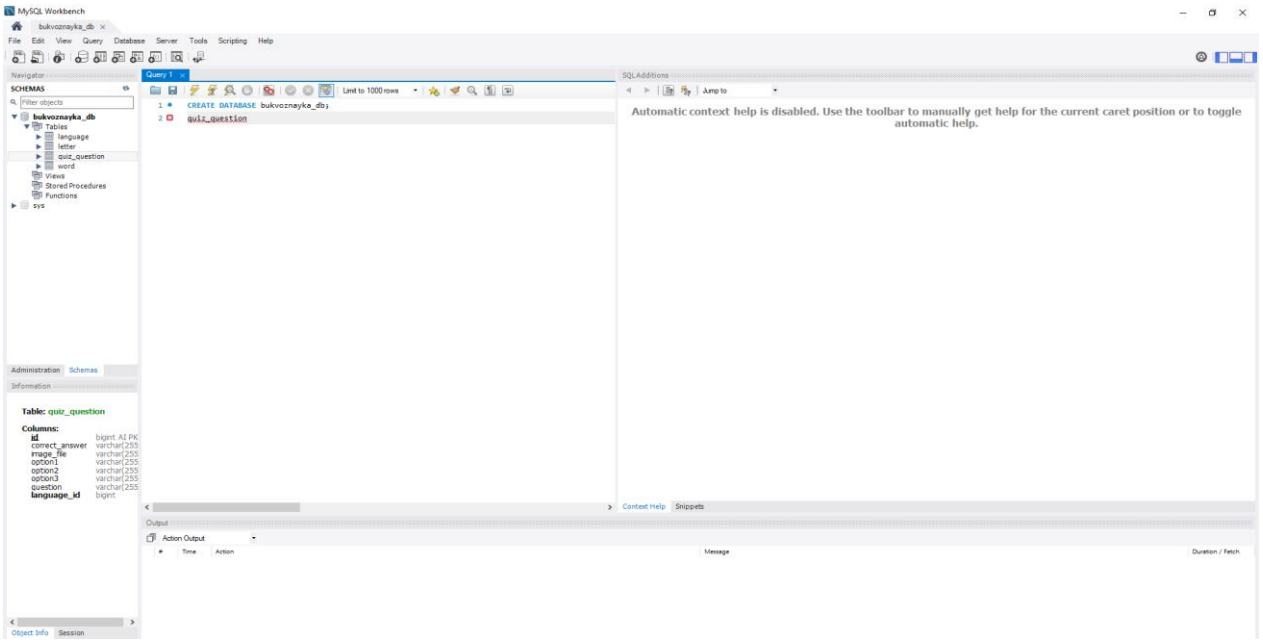


Рис. 2.2 Інтерфейс MySQL

2.5 Мова розмітки HTML

HTML (HyperText Markup Language) є стандартною мовою розмітки, яка використовується для формування структури веб-сторінок і визначає логічну організацію контенту у веб-застосунках. HTML забезпечує фундаментальну основу інтерфейсу користувача шляхом створення семантично коректних та структурованих елементів інтерфейсу. Завдяки використанню відповідних тегів розмітки формуються різноманітні компоненти інтерфейсу, такі як заголовки, текстові блоки, кнопки, поля введення, списки, зображення та контейнери, що логічно групують пов’язані між собою елементи (наприклад, меню, навчальні вправи, картки з літерами).

HTML дозволяє інтегрувати мультимедійні елементи, такі як аудіо та відео, що є важливим для створення інтерактивних компонентів навчального процесу. Адаптивність веб-сторінки, що досягається завдяки коректній структурі HTML у поєднанні з каскадними таблицями стилів (CSS), забезпечує належне відображення інтерфейсу на різних пристроях та розмірах екранів, що є особливо актуальним при розробці застосунків для дітей із урахуванням різних платформ доступу.

Таким чином, HTML виступає як базовий каркас інтерфейсу користувача, на основі якого реалізуються подальші шари стилізації та функціональності, забезпечуючи зручність, доступність та ефективність взаємодії дитини з навчальним застосунком[8].

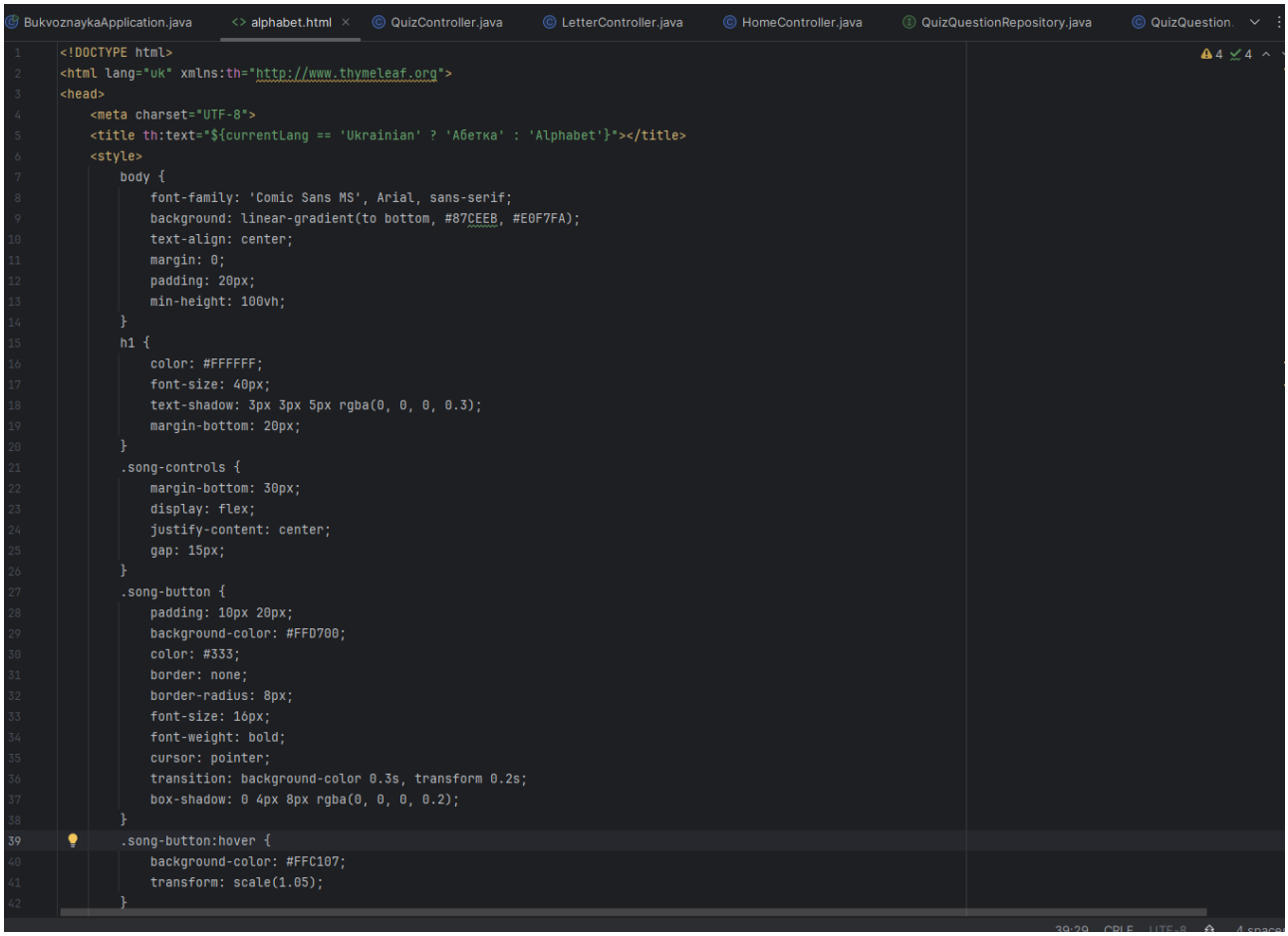


Рис. 2.3 Приклад коду мовою розмітки HTML

2.6 Стилізація інтерфейсу користувача за допомогою CSS

CSS (Cascading Style Sheets) є мовою опису стилів, яка використовується для визначення візуального оформлення та розташування елементів веб-інтерфейсу. У рамках розробки CSS відіграє ключову роль у створенні привабливого, інтуїтивно зрозумілого та доступного користувацького інтерфейсу.

За допомогою CSS здійснюється налаштування кольорової гами, шрифтів, розмірів, відступів, меж, тіней та інших візуальних характеристик, що сприяють підвищенню естетичної привабливості та зручності використання застосунку. Особлива увага приділяється застосуванню яскравих, контрастних кольорів і великих інтерактивних елементів, що відповідають потребам маленьких користувачів, які можуть мати обмежені навички точного керування мишею чи сенсорним екраном. За допомогою медіа-запитів CSS забезпечується адаптивність інтерфейсу, що дозволяє коректно відображати контент на різних пристроях і розмірах екранів — від настільних комп’ютерів до мобільних телефонів і планшетів.

CSS також підтримує реалізацію анімацій та плавних переходів, що підвищують інтерактивність і привабливість застосунку, сприяючи утриманню уваги дітей під час навчального процесу. Завдяки гнучкості та широким можливостям CSS забезпечується єдиний стиль інтерфейсу, що сприяє формуванню позитивного користувацького досвіду та полегшує навчання[9].

2.7 Інтерактивність веб-інтерфейсів за допомогою JavaScript

JavaScript є високорівневою інтерпретованою мовою програмування, яка виконується на стороні клієнта та призначена для реалізації динамічної та інтерактивної поведінки веб-інтерфейсів. Мова дозволяє створювати вебзастосунки, які реагують на дії користувача в режимі реального часу,

забезпечуючи плавну та ефективну взаємодію без необхідності перезавантаження сторінки. Основною функціональністю JavaScript є обробка подій користувача, таких як кліки мишею, введення тексту, навігація по елементах інтерфейсу, а також складніші дії, наприклад, перетягування об’єктів або взаємодія з мультимедійними елементами.

JavaScript також відповідає за реалізацію логіки роботи вебзастосунку, включаючи валідацію введених даних, перевірку правильності виконання завдань, керування станом інтерфейсу та відображення динамічного контенту, що робить користувацький досвід більш привабливим і зручним.

Завдяки асинхронним можливостям, таким як AJAX-запити та робота з API, JavaScript забезпечує взаємодію клієнтської частини з сервером без необхідності повного оновлення сторінки, що значно підвищує швидкодію та комфорт користувача.

Підтримує інтеграцію з численними бібліотеками та фреймворками (зокрема React.js, Vue.js, Angular), що дозволяє ефективно реалізовувати складні користувацькі інтерфейси та масштабувати проекти. Завдяки цьому, розробники можуть більш гнучко організовувати код, підвищувати його повторне використання і забезпечувати більш високий рівень підтримки та розвитку застосунку.

JavaScript є невід’ємною складовою сучасної фронтенд-розробки, що надає вебзастосункам інтерактивність, динамічність та адаптивність, необхідні для створення ефективного і зручного користувацького інтерфейсу[10].

3 ПРОЕКТУВАННЯ ТА РОЗРОБКА ЗАСТОСУНКУ ДЛЯ НАВЧАННЯ МАЛЕНЬКИХ ДІТЕЙ ЛІТЕРАМ ТА ЗВУКАМ

3.1 Оцінка та планування вебзастосунку

Для початку розробки вебзастосунку було вирішено сформулювати функціональні та нефункціональні вимоги. Визначення вимог є одним з найважливіших етапів у процесі створення програмного забезпечення, оскільки саме вони формують основу для подальшого проєктування, реалізації, тестування та оцінювання якості програмного продукту.

Функціональні вимоги:

1. Відображення літер з можливістю їх прослуховувати – аудіо-озвучення кожної літери.
2. Написання літери мишею – тренування моторики дитини.
3. Повне прослуховування абетки за допомогою пісні.
4. Візуальні асоціації кожної літери, відповідна картинка до кожної літери.
5. Ігрові розділи, які дозволяють користувачу вивчати літери та звуки.
6. Можливість перемикання між різними мовами – англійська та українська.

Нефункціональні вимоги:

1. Україномовна та англомовна локалізація – повна підтримка мови інтерфейсу.
2. Інтерфейс має бути адаптованим для дітей віком 3–6 років.
3. Продуктивна робота застосунку – час відгуку на дії користувача не більше 0,2 секунд.
4. Мінімум текстової інформації – зосередження на малюнках.
5. Інтуїтивні взаємодії – натиснув та отримав відгук.

Сформульовані функціональні та нефункціональні вимоги стали основою для наступних етапів розробки, зокрема для побудови архітектури застосунку, проєктування інтерфейсу та реалізації логіки взаємодії. Вони також використовуються як критерії при тестуванні та оцінці якості кінцевого продукту.

3.2 Засоби для створення UML-діаграм. UMLetino.

UMLetino — це сучасний онлайн-інструмент для створення UML-діаграм, який забезпечує простий та швидкий спосіб моделювання структури та поведінки програмних систем. Цей редактор дозволяє розробникам легко візуалізувати різні компоненти проекту, спрощуючи процес планування та аналізу.

Інтуїтивний інтерфейс та широкий набір функцій роблять UMLetino ефективним рішенням для команд будь-якого рівня досвіду. Завдяки відкритому доступу та безкоштовності, UMLetino здобуває популярність серед розробників по всьому світу.

Інструмент підтримує багато видів UML-діаграм, включаючи діаграми класів, послідовностей, станів та активностей, що робить його універсальним засобом для моделювання різноманітних аспектів програмних продуктів і складних систем[11].

Рис. 3.1 Інтерфейс UMLetino

3.3 Моделювання вимог користувачів. Діаграма варіантів використання

У процесі розробки навчального програмного забезпечення для дітей дошкільного віку одним із важливих етапів є моделювання функціональних можливостей системи. Для цього доцільним є використання діаграми варіантів використання (Use Case Diagram), яка дозволяє наочно відобразити взаємодію користувача із ключовими компонентами програмного продукту.

У діаграмі варіантів використання, на рисунку 3.1, визначено одного зовнішнього актора — Користувача. Ним може виступати як сама дитина, що працює із застосунком, так і доросла особа, яка супроводжує взаємодію дитини із програмою[12].

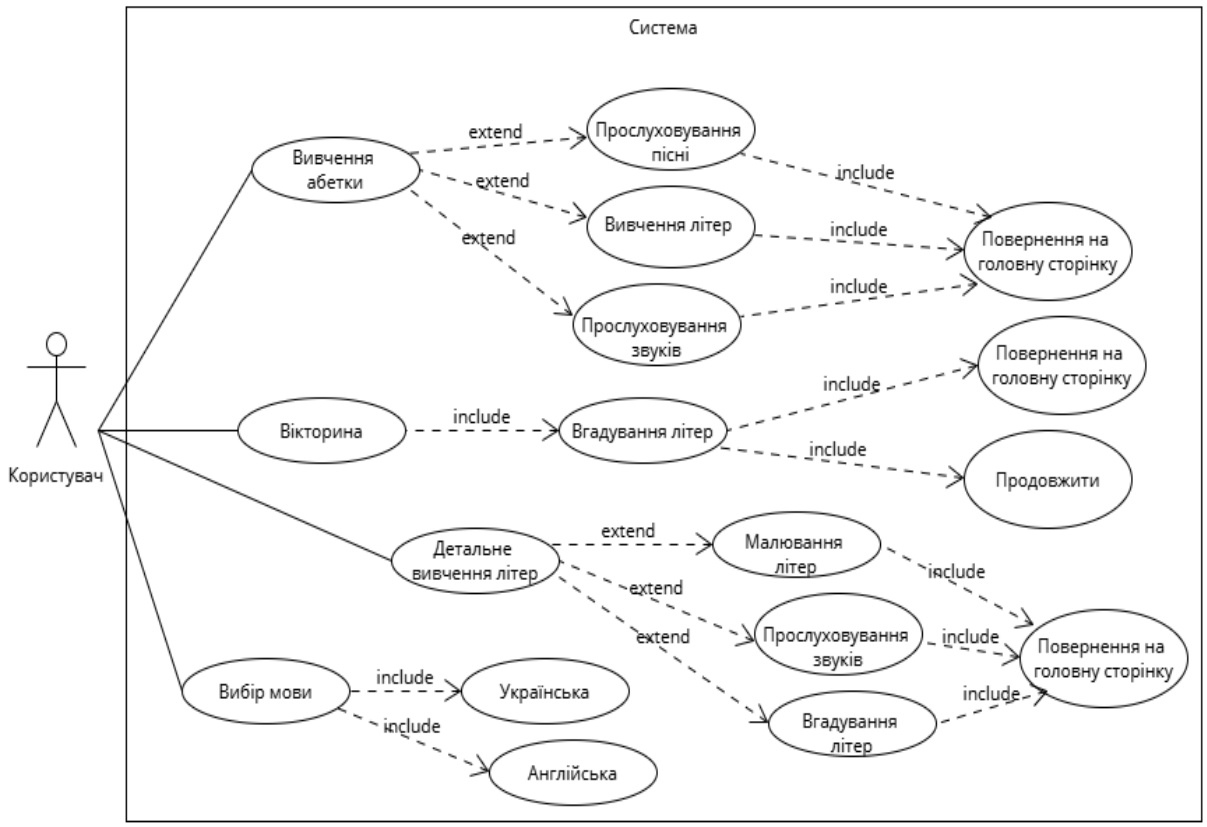


Рис. 3.1 Діаграма варіантів використання

Вивчення абетки — базова активність, що включає в себе:

- вивчення літер (extend) — реалізує ознайомлення з окремими літерами;
- прослуховування пісні (extend) — активує аудіоматеріал для полегшеного засвоєння матеріалу;
- прослуховування звуків (extend) — забезпечує можливість почути звуки, які відповідають літерам.

Усі зазначені дії включають (include) підсценарій — повернення на головну сторінку для зручності навігації користувача.

Вікторина — інтерактивний режим перевірки знань, у якому передбачено:

- вгадування літер (include) — обов’язковий підсценарій для реалізації основної мети вікторини;
- продовжити (include) — перехід до наступного питання чи завдання;
- повернення на головну сторінку (include) — завершення або вихід із режиму.

Детальне вивчення літер — режим поглибленого ознайомлення з літерами, який включає:

- малювання літер (extend) — активність, що розвиває моторні навички;
- прослуховування літер (extend) — аналогічна аудіопідтримка;
- вгадування літер (extend) — інтерактивний компонент на впізнавання.

Кожен із розширених варіантів включає можливість повернення на головну сторінку (include).

Вибір мови — користувач обирає мовну версію застосунку, зокрема:

- українська мова (include);
- англійська мова (include).

У діаграмі використано два основних типи відношень:

- <<include>> — позначає обов’язкові підсценарії, що завжди виконуються в контексті основного варіанту використання. Наприклад, вгадування літер є невіддільною частиною функціоналу вікторини, тому позначене як включене;
- <<extend>> — вказує на опціональні або додаткові функції, які можуть бути реалізовані за потреби. Наприклад, малювання літер не є обов’язковою частиною навчання, але розширює його можливості.

3.4 Блок-схема взаємодії у системі вікторини

Блок-схема яка зображена на рисунку 3.2, описує процес проходження користувачем вікторини, починаючи з активації системи, отримання питань, надання відповідей, перевірки їх правильності та завершення або переходу до наступного питання. Вона ілюструє основні стани системи та логіку переходів між ними.

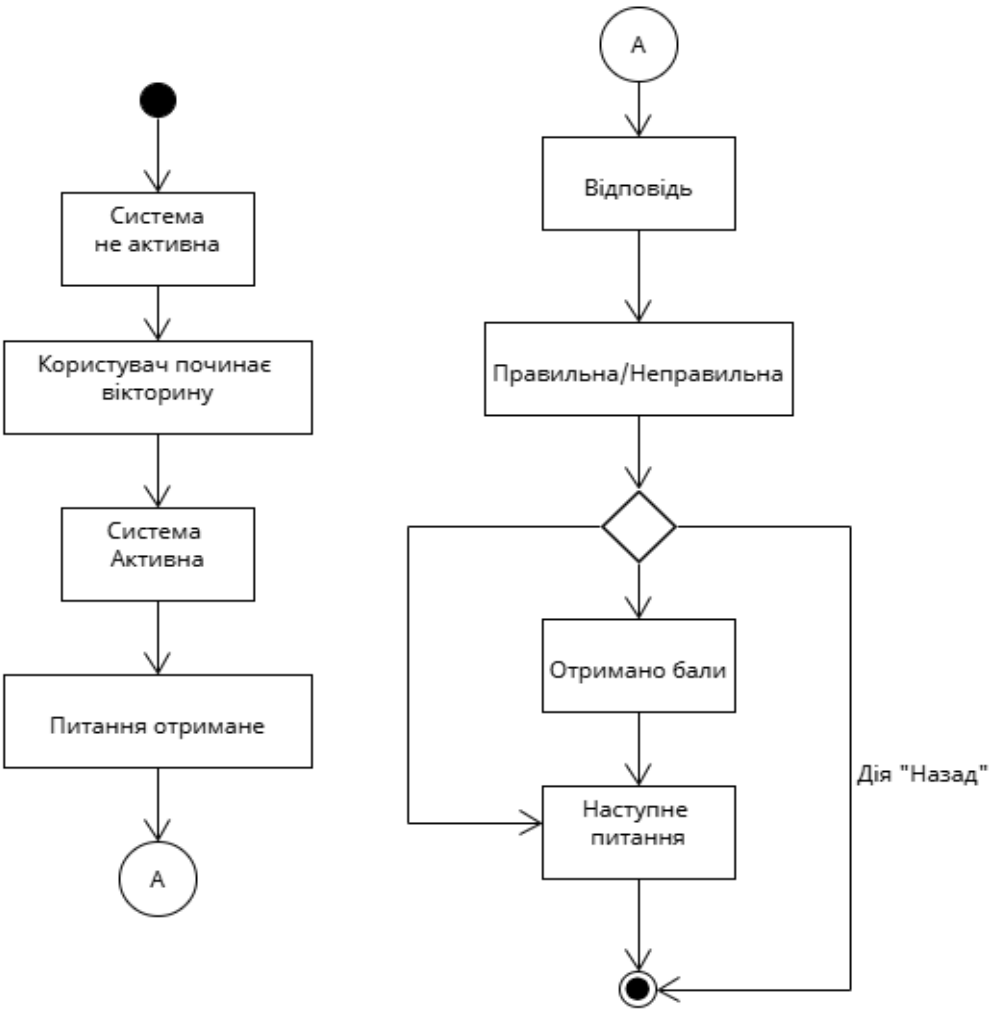


Рис 3.2 Блок-схема взаємодії вікторини

Система не активна:

- початковий стан системи, коли вікторина ще не запущена або перебуває в режимі очікування;
- система не взаємодіє з користувачем.

Користувач починає вікторину:

- користувач ініціює процес;
- система переходить у активний стан.

Система активна:

- основний робочий стан, у якому система готова до проведення вікторини;
- відбувається завантаження або питання.

Питання отримане:

- система відображає користувачу питання з варіантами відповідей;
- користувач має можливість обрати відповідь.

Відповідь:

- користувач надає відповідь на запитання;
- система фіксує відповідь і готується до її перевірки.

Правильна / Неправильна:

- система аналізує відповідь і визначає її правильність;
- результат зберігається та нараховуються бали.

Отримано бали:

- користувачу відображається результат.

Наступне питання:

- система переходить до наступного питання у вікторині.

Процес повторюється зі стану "Питання отримане":

- користувач може повернутися до попереднього питання або вийти з вікторини;
- якщо вікторина завершена, система переходить у стан "Не активна".

3.5 Структура архітектури програми. Діаграма пакетів

У сучасній розробці програмного забезпечення чітке розмежування обов'язків між компонентами системи є критично важливим для забезпечення масштабованості, підтримки та ефективної колаборації між командами. Представлена на рисунку 3.3 діаграма пакетів візуалізує ключові архітектурні рішення, що лежать в основі організації кодової бази програми.

Діаграма відображає архітектурну організацію програми, розділену на шари (пакети) з чітко визначеними ролями. Вона побудована за принципом багаторівневої архітектури (multi-layer architecture), де кожен рівень відповідає за певний аспект функціонування системи[13].

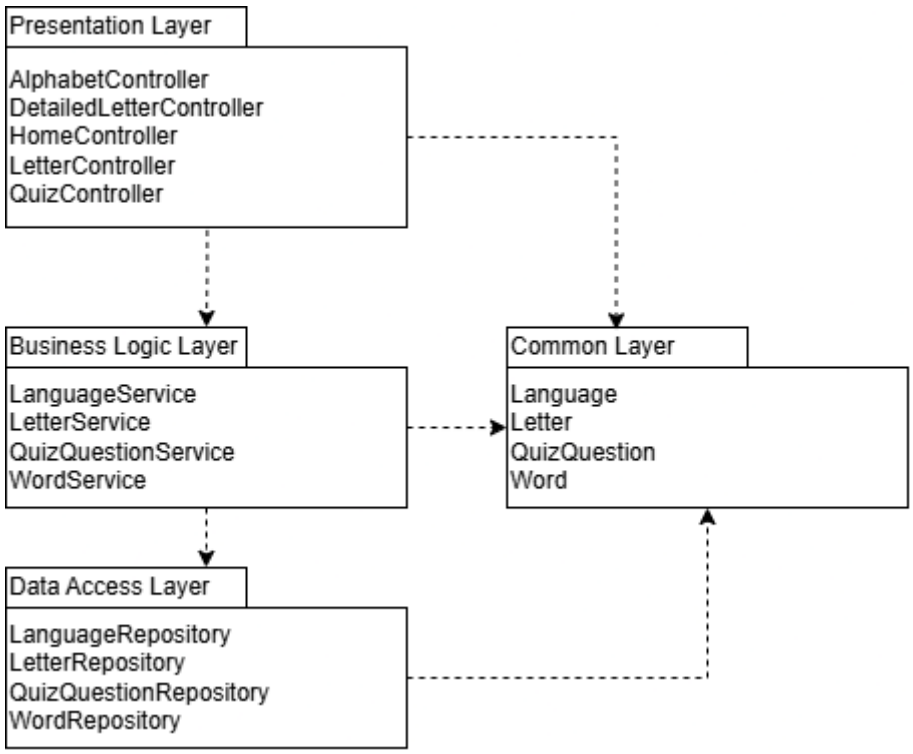


Рис 3.3 Діаграма пакетів

Presentation Layer (Рівень представлення):

- відповідає за взаємодію з користувачем. Містить контролери, які обробляють HTTP-запити та відповідають за відображення даних.

Контролери:

- AlphabetController – робота з алфавітом (відображення списку літер);
- DetailedLetterController – детальна інформація про конкретну літеру;
- HomeController – головна сторінка;
- LetterController – базові операції з літерами;
- QuizController – управління вікторинами (перевірка відповідей, перехід між питаннями).

Business Logic Layer (Рівень бізнес-логіки):

- містить сервіси, які реалізують основну логіку програми (обробка даних, алгоритми).

Сервіси:

- LanguageService – робота з мовами (наприклад, переключення мови інтерфейсу);
- LetterService – логіка, пов’язана з літерами (пошук, фільтрація);
- QuizQuestionService – генерація питань, перевірка відповідей у вікторин;
- WordService – операції зі словами (додавання, пошук).

Data Access Layer (Рівень доступу до даних):

- відповідає за взаємодію з базою даних. Містить репозиторії, які виконують CRUD-операції.

Репозиторії:

- LanguageRepository – запити до таблиці мов;
- LetterRepository – робота з даними про літери;
- QuizQuestionRepository – отримання/збереження питань вікторини;
- WordRepository – управління словами в БД.

Common Layer (Загальний рівень):

- містить моделі даних (класи-сутності), які використовуються всіма шарами. Наприклад: Language, Letter, QuizQuestion, Word – об’єкти, що описують основні сутності програми.

3.6 Діаграма послідовності

Розроблена діаграма послідовності відображає повний цикл взаємодії користувача із системою під час проходження вікторини (див. рисунок 3.4).

Така модель дозволяє чітко структурувати логіку програми, забезпечує зрозуміле розділення відповідальностей між компонентами та слугує основою для реалізації відповідного функціоналу в програмному коді. Вона також сприяє підвищенню якості програмного продукту за рахунок прозорості процесів обробки подій [14].

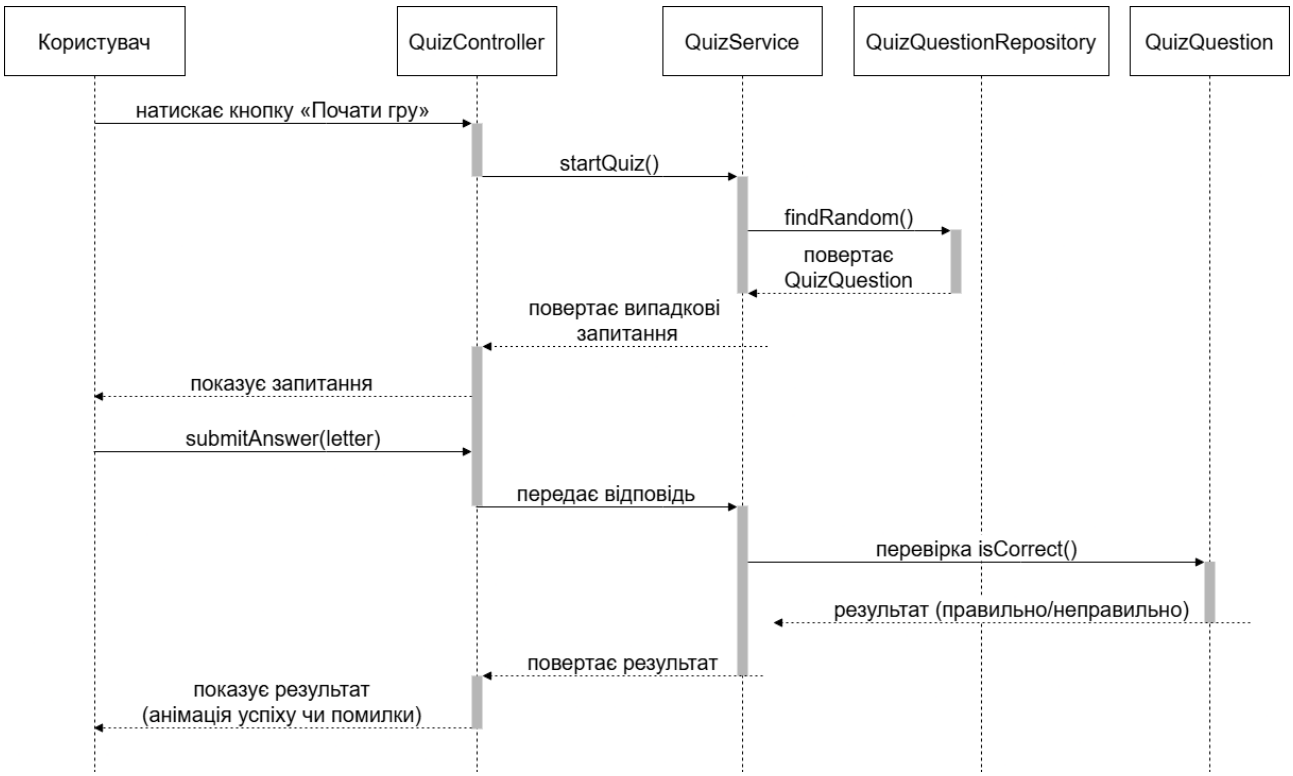


Рис 3.4 Діаграма послідовності

Користувач ініціює процес взаємодії з системою:

- QuizController — контролер, що обробляє запити користувача з інтерфейсу;
- QuizService — сервісний компонент, який реалізує бізнес-логіку вікторини;
- QuizQuestionRepository — компонент доступу до даних, відповідальний за отримання запитань із джерела (наприклад, бази даних);
- QuizQuestion — модель, яка представляє конкретне запитання вікторини та містить метод для перевірки правильності відповіді.

Процес проходження вікторини складається з наступних кроків:

1. Користувач натискає кнопку «Почати гру».
2. Контролер QuizController викликає метод startQuiz() сервісу QuizService.

3. Сервіс, у свою чергу, звертається до репозиторію QuizQuestionRepository методом findRandom() для отримання випадкового запитання.

4. Репозиторій повертає об’єкт QuizQuestion, який передається назад через сервіс до контролера.

5. Контролер повертає запитання користувачу для відображення в інтерфейсі.

Надсилання відповіді:

1. Користувач вводить відповідь (букву) та надсилає її через метод submitAnswer(letter).

2. Контролер передає отриману відповідь до сервісу.

3. Сервіс викликає метод isCorrect() у моделі QuizQuestion для перевірки правильності відповіді.

4. Модель повертає результат (правильно/неправильно), який передається через сервіс до контролера.

5. Контролер надсилає результат користувачу, який бачить відповідну анімацію (успіх або помилка).

3.7 Діаграма розгортання

Діаграма розгортання моделює фізичне розміщення програмних компонентів системи на апаратних ресурсах (див. рисунок 3.5).

У контексті проєкту, діаграма відображає клієнт-серверну архітектуру, де компоненти взаємодіють через мережу (HTTP/HTTPS)[15].

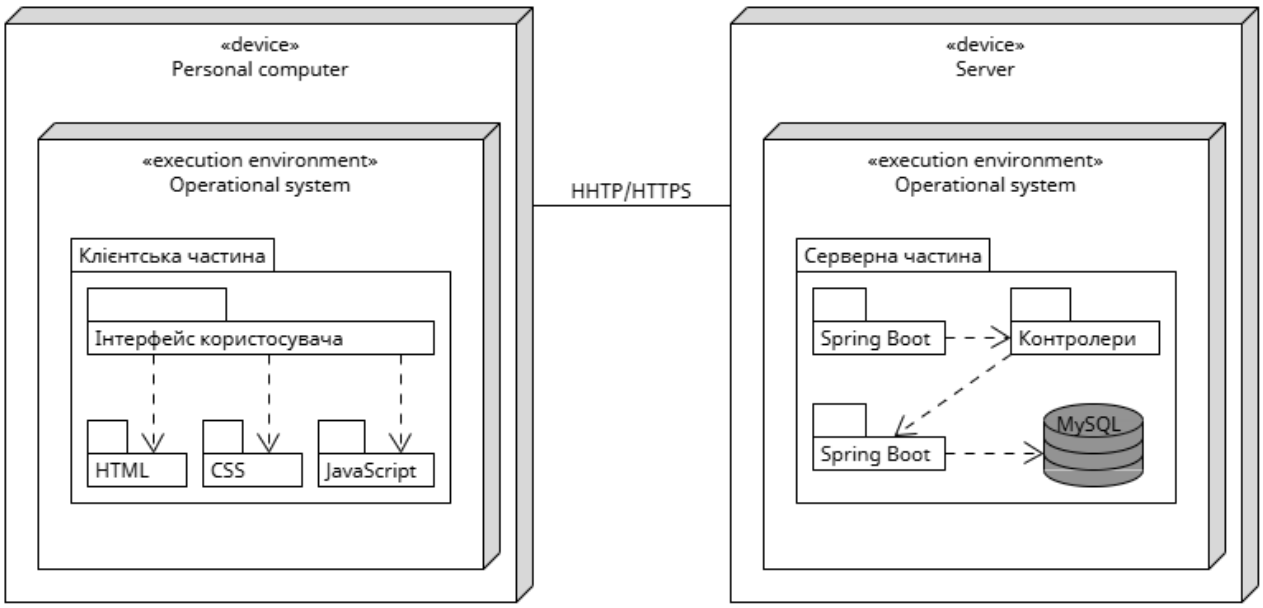


Рис 3.5 Діаграма розгортання

Вузол клієнта (Personal computer):

- клієнтська частина – візуалізує браузерну частину за стосунку;
- інтерфейс користувача (UI) – основна точка взаємодії користувача із системою;
- HTML / CSS / JavaScript – технології, що відповідають за відображення, стиль та динамічну поведінку інтерфейсу.

Взаємодія:

- користувач взаємодіє з UI, що реалізовано за допомогою HTML, CSS та JavaScript, через веб-браузер.

Вузол сервера (Server):

- серверна частина – логіка обробки запитів і бізнес-процесів;
- Spring Boot – фреймворк, що використовується для побудови серверного за стосунку;
- контролери – REST-контролери, що обробляють запити від клієнта та передають дані;
- MySQL – реляційна база даних, яка зберігає інформацію про питання, букви, слова, мови тощо.

Взаємодія:

- контролери працюють у середовищі Spring Boot і обробляють HTTP-запити;
- контролери надсилають запити до бази даних MySQL для отримання або збереження інформації.

Взаємозв’язок між вузлами:

- HTTP/HTTPS – стандартний протокол взаємодії між клієнтською та серверною частиною;
- клієнт відправляє запити на сервер (отримати питання).

3.8 Опис інтерфейсу користувача

Інтерфейс користувача описує механізми взаємодії системи з користувачем і дозволяє зрозуміти логіку її використання.

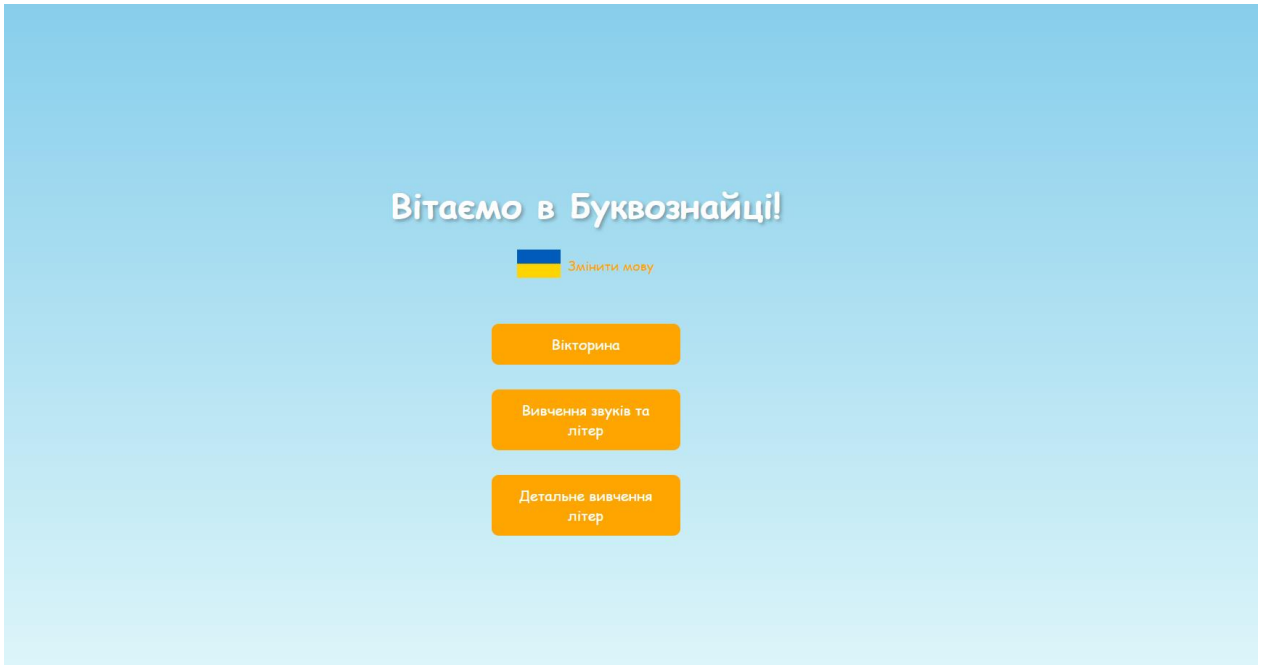


Рис. 3.6 Головний екран застосунку

Головний екран вебзастосунку є початковою точкою взаємодії користувача з системою. Його структура розроблена з урахуванням особливостей цільової аудиторії — дітей віком від 3 до 6 років. Інтерфейс є простим, інтуїтивно зрозумілим та візуально привабливим. На головному екрані розміщено чотири основні інтерактивні елементи.

Під заголовком знаходиться кнопка зміни мови, яка дозволяє перемикатися між українською та англійською локалізацією. При натисканні інтерфейс миттєво адаптується до обраної мови, змінюючи як текст, так і озвучення відповідно до мовного середовища.

Ця функція сприяє двомовному навчанню та є важливою складовою функціоналу застосунку.

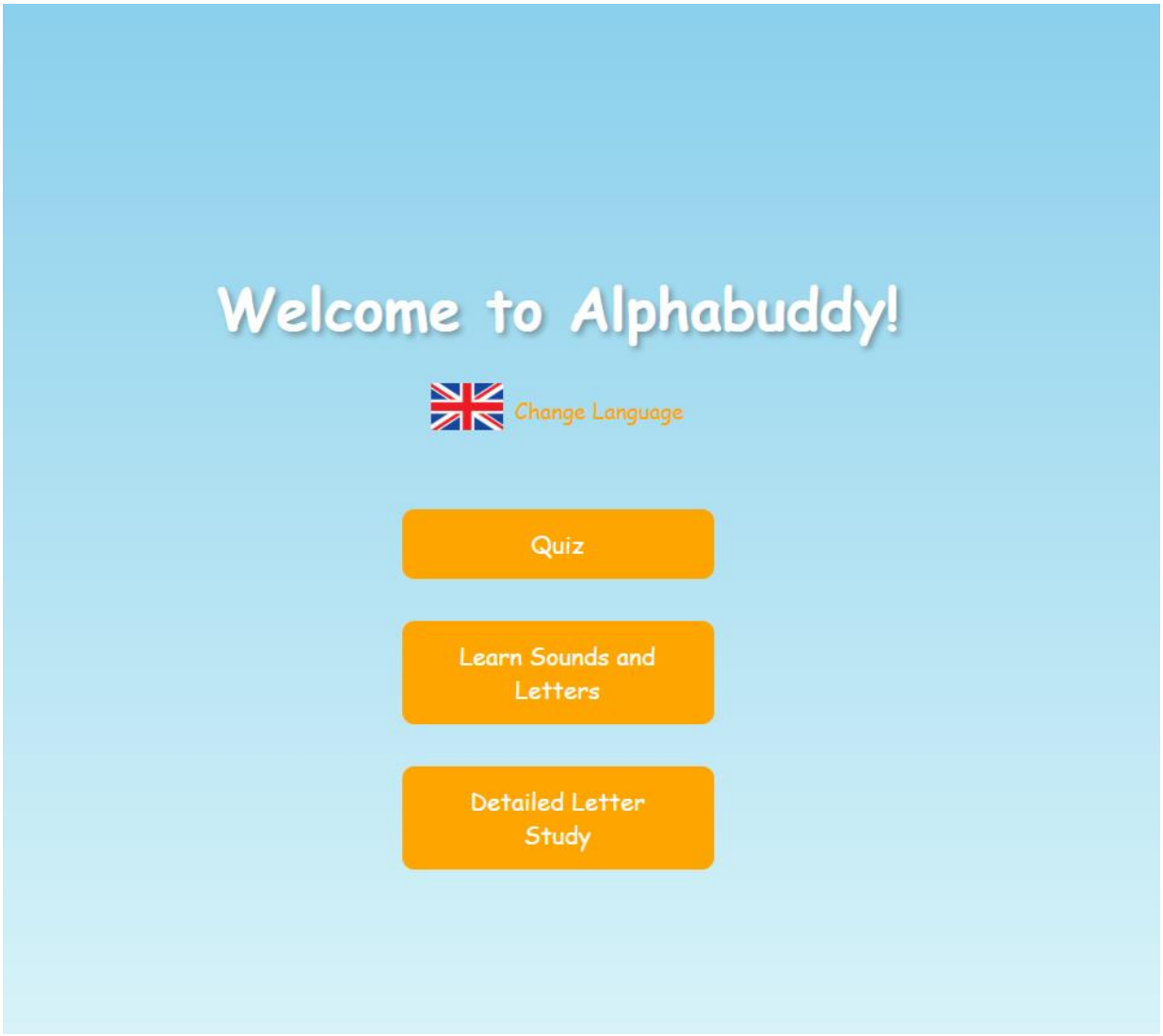


Рис. 3.7 Головний екран застосунку у англійській локалізації

3.9 Розробка екрану з інтерактивною абеткою

На рисунку 3.8 – веб-екран реалізує інтерактивну абетку з аудіо-супроводом для кожної літери, що дозволяє користувачам навчатися через візуальні та звукові асоціації. Використовуючи сучасні веб-технології, сторінка адаптована під дві мови (українську та англійську) за допомогою механізму шаблонів Thymeleaf[16].

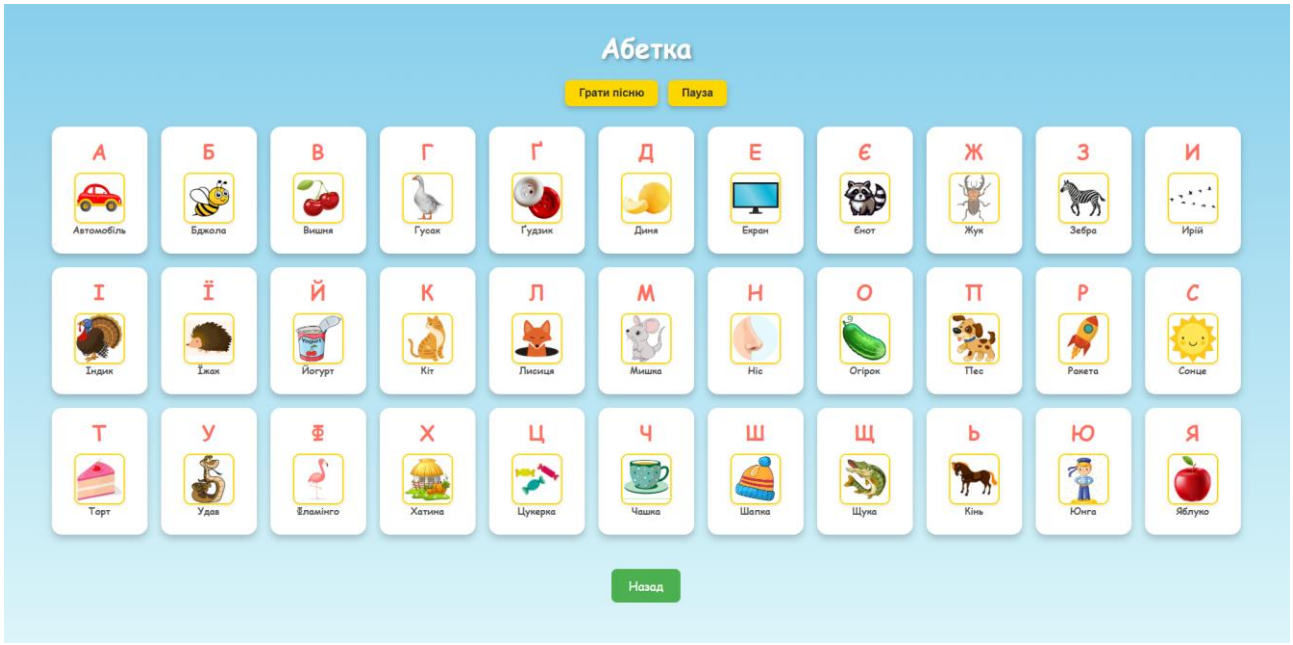


Рис. 3.8 Інтерактивна абетка українською мовою

При розробці екрану з інтерактивною абеткою, було розроблено наступне:

1. Кожна літера представлена у вигляді клікабельного блоку з зображенням та прикладом слова.
2. При натисканні на літеру відтворюється відповідний звуковий файл з вимовою цієї літери.
3. Користувачі можуть увімкнути або призупинити фонову пісню абетки з плавним ефектом збільшення/зменшення гучності.
4. Сторінка має приємний градієнтний фон, стильні кнопки та анімації для покращення користувацького досвіду.

5. Заголовки, кнопки і медіа-файли змінюються (див. рисунок 3.9) залежно від обраної мови (українська або англійська).

Рис. 3.9 Інтерактивна абетка англійською мовою

Технічні рішення:

- використано стандартний HTML, CSS та JavaScript із обробкою аудіо через Web Audio API та об’єкти Audio для кращої взаємодії зі звуком;
- звуки літер та фонова пісня зберігаються у форматі MP3 та завантажуються асинхронно, що забезпечує швидкий відгук на взаємодію[17].

3.10 Розробка екрану з вікториною

Цей екран (див. рисунок 3.10) реалізує навчальну вікторину з кількома запитаннями, зображеннями та варіантами відповідей. Його основна мета — перевірка знань у цікавому ігровому форматі з моментальним зворотним зв’язком та підрахунком балів.

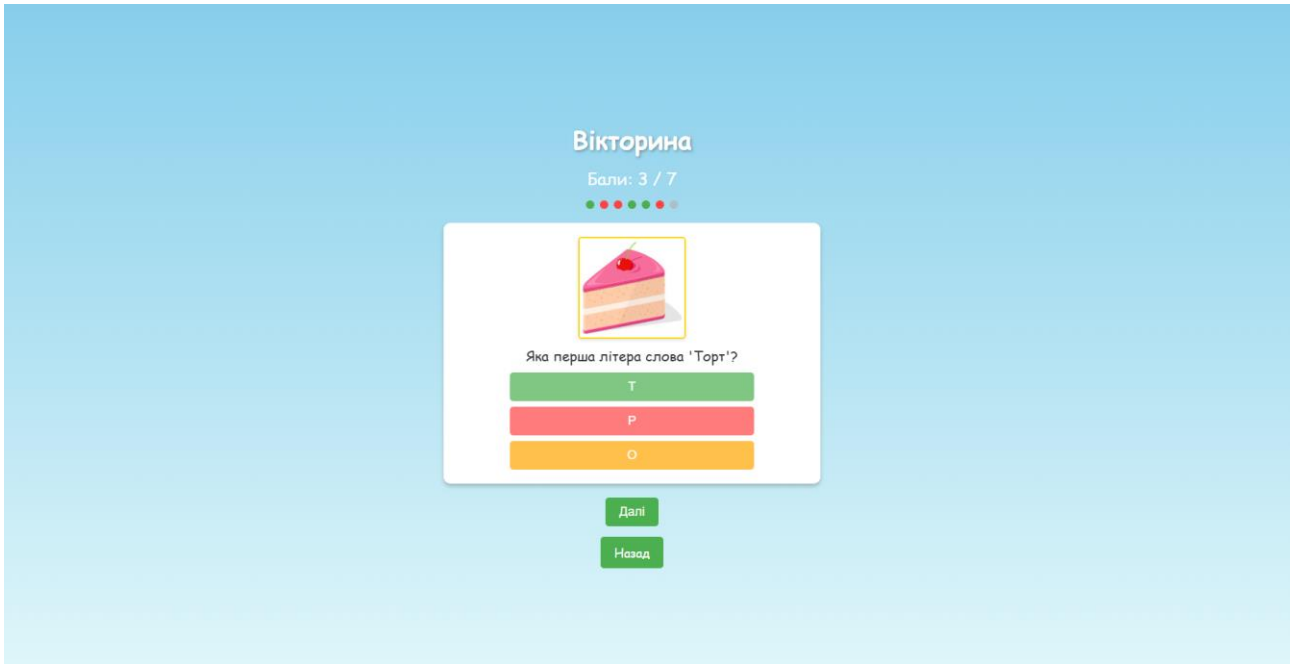


Рис. 3.10 Екран проходження вікторини

При розробці екрану з вікториною, було розроблено наступне:

1. Кожне запитання відображається окремо, з можливістю переходу до наступного або пропуску.

2. Для кожного запитання виводиться відповідне зображення, яке допомагає краще сприймати інформацію.

1. Відповідь користувача одразу підсвічується зеленим або червоним кольором залежно від правильності.

2. Угорі відображається кількість набраних балів у реальному часі.

3. Кольорові кружечки показують поточне запитання, а також правильність попередніх відповідей.

4. Елементи плавно з’являються, підсвічуються та змінюють колір для покращення візуального сприйняття.

5. Усі тексти та медіа-ресурси автоматично адаптуються до вибраної мови (українська або англійська).

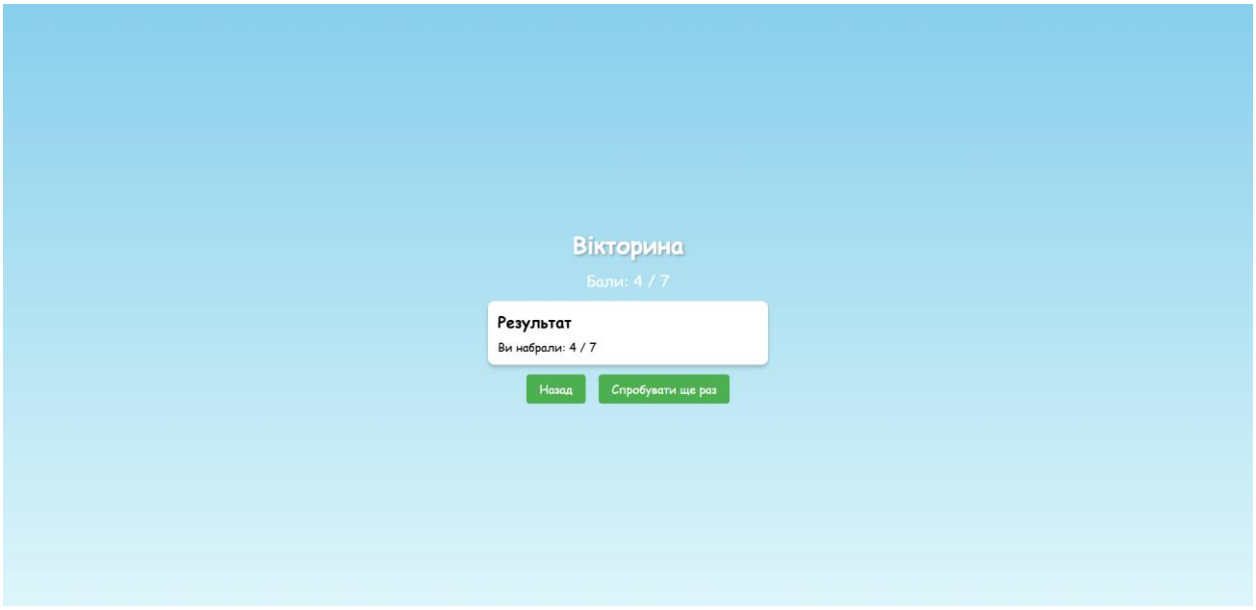


Рис. 3.11 Екран проходження вікторини

Технічні рішення:

- реалізація побудована на базі HTML, CSS, JavaScript та механізму шаблонів Thymeleaf для динамічного виводу запитань і мовної адаптації;
- логіка перемикання, підрахунку балів і взаємодії з елементами реалізована на стороні клієнта без перезавантаження сторінки[18].

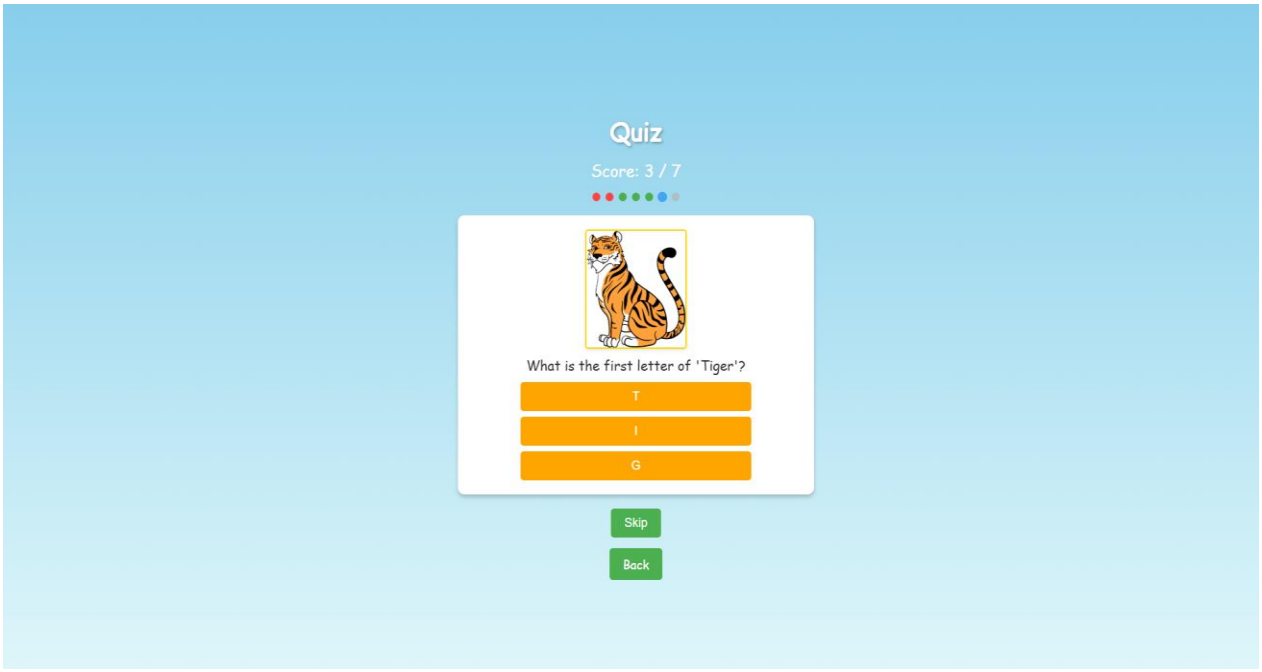


Рис. 3.11 Екран проходження вікторини англійською

3.11 Розробка екрану з детальним вивченням літер

Розділ детального вивчення літер є розширенням функціональності основного екрану абетки та спрямований на поглиблення знайомства дитини з кожною окремою літерою. Його реалізація (вид.рис 3.12) дозволяє забезпечити багатоканальне сприйняття інформації — через візуальні, слухові та тактильні елементи.

У даному контексті багатоканальність означає одночасну взаємодію з матеріалом за допомогою:

- візуальних елементів — зображення літери, люстрації до кожної літери;
- слухових елементів — озвучення літери;
- інтерактивних елементів — можливість малювати літери за допомогою елемента «canvas» або клікати на ілюстрації для повторного прослуховування.

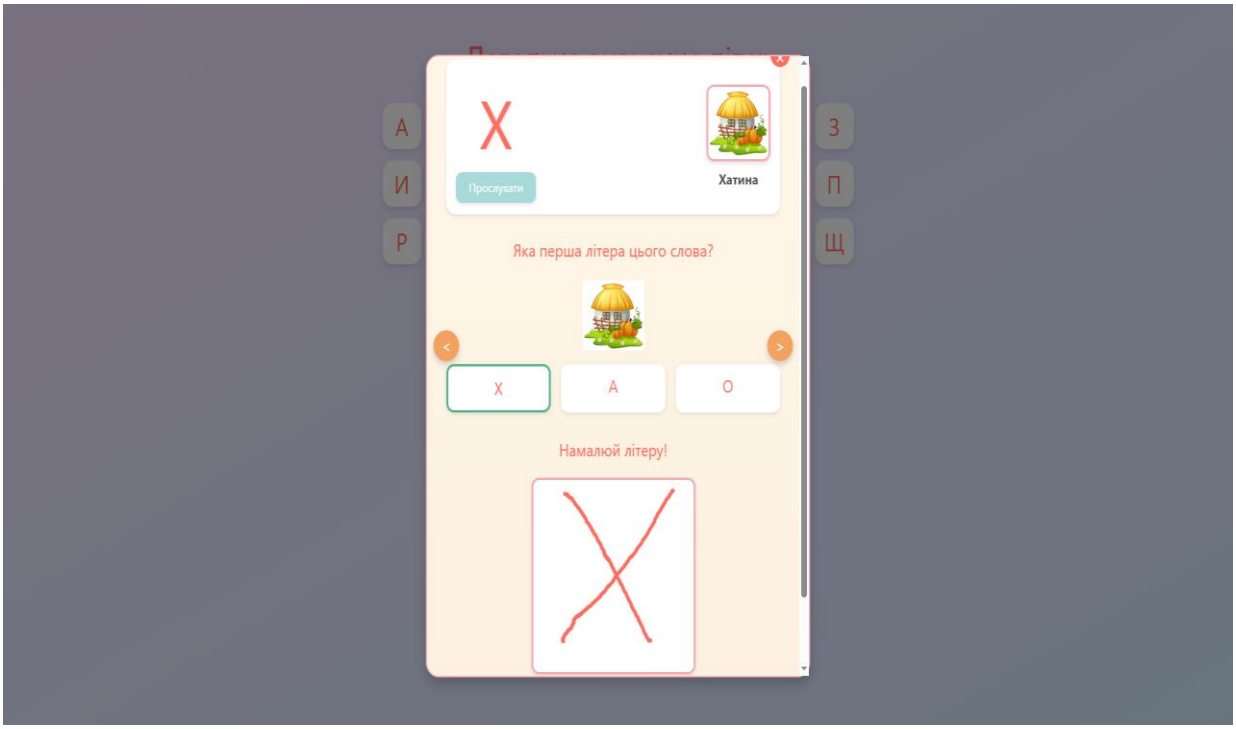


Рис. 3.12 Екран розділу детального навчання літер

При розробці екрану з детальним вивченням літер, було розроблено наступний функціонал:

1. Модальне вікно з інформацією про літеру — при натисканні на літеру в основній абетці відкривається модальне вікно, яке містить навчальний контент, спрямований на детальне опрацювання вибраного символу.
2. Аудіо-вимова літери — при відкритті модального вікна відтворюється звук з правильною вимовою літери.
3. Ілюстрація до прикладу слова — поряд із літерою відображається зображення об’єкта, який починається на цю літеру (наприклад, «П — пес»).
4. Полотно для малювання (canvas) — інтерактивне поле дозволяє користувачеві власноруч писати літеру[19].
5. Присутні кнопки для закриття вікна та повернення до абетки, що забезпечує простоту користування та збереження логічного потоку навчання.

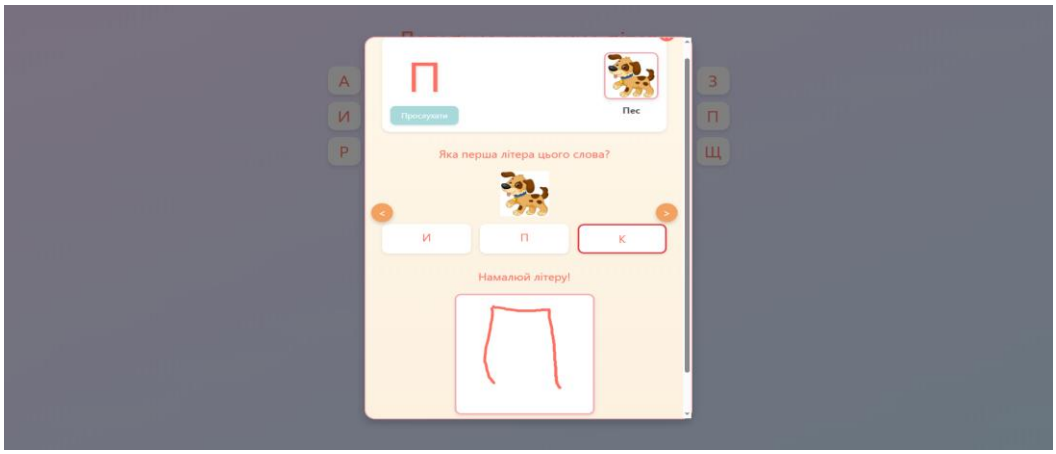


Рис. 3.13 Екран розділу детального навчання літер

Технічні рішення:

- розмітка реалізована у вигляді семантичного модального блоку, стилізованого за допомогою адаптивного CSS для коректного відображення на різних екранах;
- JavaScript забезпечує відкриття/закриття модального вікна, програвання звуку та ініціалізацію інтерактивного малювання;
- Canvas API реалізує можливість малювання з підтримкою подій миші та сенсорних дотиків;
- вимова літери та слова активується за подією завантаження модального вікна, для цього використовуються об'єкти Audio.

4 ТЕСТУВАННЯ ЗАСТОСУНКУ

Для перевірки функціональності та зручності користування розробленим веб-додатком було проведено тестування за методом «чорного ящика». Цей метод передбачає оцінку поведінки системи без аналізу її внутрішньої структури чи коду, тобто з позиції кінцевого користувача.

Тестування охоплювало основні сценарії взаємодії з інтерфейсом, зокрема відкриття та закриття модальних вікон, натискання на інтерактивні елементи, перемикання мов, запуск фонової музики, відтворення звуків і використання елемента малювання. Особливу увагу приділено перевірці доступності функцій для дітей — цільової аудиторії проекту[20].

Тестування буде проводитись за допомогою тест-кейсів. Кожен тест-кейс включає в себе детальний опис дій, які потрібно виконати, а також тестові дані. Структура кожного тест-кейсу включає наступне:

- Ідентифікатор тест-кейсу (Test Case ID) — унікальний номер тест-кейсу;
- Назва тест-кейсу (Test Case Title) — короткий опис тесту, що перевіряється;
- Тестові дані (Test Data) — вхідні параметри та умови для виконання тесту;
- Очікуваний результат (Expected Result) — передбачувана коректна поведінка системи;
- Статус виконання (Status) — результат тестування (Успішно/Невдало).

В таблиці 4.1. наведені тест-кейси функціонального тестування застосунку.

Таблиця 4.1.

Результати функціонального тестування застосунку

Test Case ID	Test Case Title	Test Steps	Test Data	Expected Result	Status
1	Вибір мови	1. Перейти на головний екран застосунку. 2. Натиснути на "Змінити мову".	-	Мова інтерфейсу змінюється на англійську.	Успішно
2	Перехід до екрану вікторини	1. Перейти на головний екран. 2. Натиснути на кнопку "Вікторина".	-	Вікторина успішно відкривається.	Успішно
3	Правильна відповідь у вікторині	1. Почати вікторину. 2. Обрати правильну літеру.	-	Правильна відповідь буде відображена зеленим кольором.	Успішно
4	Неправильна відповідь у вікторині	1. Почати вікторину. 2. Обрати не правильну літеру.	-	Неправильна відповідь буде відображена червоним кольором.	Успішно
5	Повернення на головну сторінки з розділу "Вікторина"	1. Під час проходження вікторини натиснути кнопку "Назад".	-	Користувач повертається на головну сторінку.	Успішно

Продовження таблиці 4.1.

Результати функціонального тестування застосунку

Test Case ID	Test Case Title	Test Steps	Test Data	Expected Result	Status
6	Пройти вікторину заново	1. Під час проходження вікторини натиснути кнопку "Спробувати ще раз".	-	Користувач починає вікторини заново.	Успішно
7	Перехід до екрану інтерактивної абетки	1. Перейти на головний екран. 2. Натиснути на кнопку "Вивчення звуків та літер".	-	Абетка успішно відкривається.	Успішно
8	Програвання абетки у вигляді пісні	1. Перейти у розділ "Вивчення звуків та літер". 2. Натиснути кнопку "Грати пісню".	-	Програється абетка у вигляді пісні.	Успішно
9	Аудіо-супровід літери "А"	1. Перейти у розділ "Вивчення звуків та літер". 2. Натиснути на літеру "А".	-	Програється звук літери "А".	Успішно

Продовження таблиці 4.1.

Результати функціонального тестування застосунку

Test Case ID	Test Case Title	Test Steps	Test Data	Expected Result	Status
10	Аудіо-супровід літери "Б"	1. Перейти у розділ "Вивчення звуків та літер". 2. Натиснути на літеру "Б".	-	Програється звук літери "Б".	Успішно
11	Аудіо-супровід літери "В"	1. Перейти у розділ "Вивчення звуків та літер". 2. Натиснути на літеру "В".	-	Програється звук літери "В".	Успішно
12	Аудіо-супровід літери "Г"	1. Перейти у розділ "Вивчення звуків та літер "Г". 2. Натиснути на літеру "Г".	-	Програється звук літери "Г".	Успішно
13	Аудіо-супровід літери "Ґ"	1. Перейти у розділ "Вивчення звуків та літер "Ґ". 2. Натиснути на літеру "Ґ".	-	Програється звук літери "Ґ".	Успішно

Продовження таблиці 4.1.

Результати функціонального тестування застосунку

Test Case ID	Test Case Title	Test Steps	Test Data	Expected Result	Status
14	Аудіо-супровід літери "Д"	1. Перейти у розділ "Вивчення звуків та літер "Д". 2. Натиснути на літеру "Д".	-	Програється звук літери "Д".	Успішно
15	Аудіо-супровід літери "Е"	1. Перейти у розділ "Вивчення звуків та літер "Е". 2. Натиснути на літеру "Е".	-	Програється звук літери "Е".	Успішно
16	Аудіо-супровід літери "Є"	1. Перейти у розділ "Вивчення звуків та літер "Є". 2. Натиснути на літеру "Є".	-	Програється звук літери "Є".	Успішно
17	Час відгуку на дії користувача не більше 0,2 секунд	1. Перейти у розділ "Вікторина". 2. Натиснути на будь-який варіант відповіді.	-	Система відповідає на дії користувача не більше 0,2 секунд.	Успішно

Продовження таблиці 4.1.

Результати функціонального тестування застосунку

Test Case ID	Test Case Title	Test Steps	Test Data	Expected Result	Status
18	Повернення на головний екран застосунку	1. Натиснути кнопку "Назад".	-	Користувач повертається на головний екран застосунку.	Успішно
19	Перехід до екрану детального вивчення літер	1. Перейти на головний екран. 2.Натиснути на кнопку "Детальне вивчення літер".	-	Розділ детального вивчення літер відкривається.	Успішно
20	Полотно для малювання (canvas)	1. Перейти у розділ детального вивчення літер. 2. У спеціальном у віконці намалювати обрану літеру.	-	На екрані буде відображатись намальована літера.	Успішно
21	Очистка полотна для малювання.	1. Натиснути кнопку "Очистити".	-	Полотно очищається від намальованої літери.	Успішно
22	Повернення на головний екран застосунку	1. Натиснути кнопку "Назад".	-	Користувач повертається на головний екран застосунку.	Успішно

Було протестовано як українську, так і англійську версію інтерфейсу користувача. Особливу увагу приділено перевірці коректності локалізації елементів управління. У тест-кейсах використовувалися вхідні дані, характерні для дій звичайного користувача, зокрема введення правильних і неправильних відповідей, навігація між питаннями, спроба дострокового завершення вікторини тощо.

Кожен тест-кейс включав детальний опис кроків, що потрібно виконати, зазначення тестових даних, а також критерії успішного проходження тесту. Це забезпечувало систематичне покриття всіх функціональних можливостей програми, включаючи обробку помилок, відображення зворотного зв’язку та логіку переходів між етапами вікторини.

Результати тестування показали, що всі ключові елементи застосунку працюють коректно відповідно до очікуваної поведінки. Не було виявлено критичних помилок чи збоїв, що свідчить про високу стабільність системи..

ВИСНОВКИ

1. Проведено детальний аналіз існуючих застосунків для навчання дітей літерам та звукам, що дало змогу виявити їхні сильні сторони та недоліки.
2. Визначено основні потреби користувачів — дітей та їх батьків, а також сформульовано функціональні та нефункціональні вимоги до застосунку.
3. Проведено огляд та аналіз доступних технологій і засобів розробки застосунків, що дало змогу вибрати оптимальний стек технологій для реалізації проєкту з урахуванням специфіки цільової аудиторії.
4. Спроєктовано архітектуру застосунку, яка забезпечує логічну структуру та гнучкість у подальшому розвитку, а також розроблено інтерфейс, адаптований до потреб маленьких дітей.
5. Реалізовано ключові функціональні модулі: абетку для знайомства з літерами, вікторину для перевірки знань та інтерактивний розділ з малювання літер, що сприяє кращому засвоєнню матеріалу через гру та творчість.
6. Проведено мануальне тестування, яке підтвердило відповідність вимогам.

Робота пройшла апробацію. За її результатами було опубліковано наступні тези доповідей:

1) Ярина Т.О., Андрусевич Н.В. Діаграма предметної області застосунку для навчання маленьких дітей літерам та звукам: Матеріали VI Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях». 24.04.2025 Державний університет інформаційно-комунікаційних технологій, Київ, Україна, 2025. Збірник тез К.: ДУІКТ, С.117-119.

2) Ярина Т.О., Андрусевич Н.В. Огляд програм-аналогів застосунку для навчання маленьких дітей літерам та звукам: Матеріали VI Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях». 24.04.2025 Державний університет інформаційно-комунікаційних технологій, Київ, Україна, 2025. Збірник тез К.: ДУІКТ, С.172-174.

ПЕРЕЛІК ПОСИЛАНЬ

1. CLEVERBIT. Вчимося читати – вчимо літери - Apps on Google Play. Android Apps on Google Play. URL: https://play.google.com/store/apps/details?id=net.cleverbit.SaveAnimalsFree

2. onartsoft. Українська Абетка для дітей - Apps on Google Play. Android Apps on Google Play. URL: https://play.google.com/store/apps/details?id=com.onartsoft.abetka

3. Навчатися. EDUGAMES. URL: https://edugames.rozumniki.com/task-online/

4. Top 10 Reasons to Learn Java in 2025 - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/reasons-to-learn-java/

5. Advantages of IntelliJ IDEA. Alibaba Cloud Community. URL: https://www.alibabacloud.com/blog/advantages-of-intellij-idea_595539

6. Why Choose Spring as Your Java Framework? | Baeldung. Baeldung. URL: https://www.baeldung.com/spring-why-to-choose

7. MySQL Explained: Your Guide to Mastering This Powerful Database. URL: https://www.oracle.com/mysql/what-is-mysql/

8. HTML: The Markup Language. W3C. URL: https://www.w3.org/TR/2012/WD-html-markup-20120329/spec.html

9. CSS: Cascading Style Sheets | MDN. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/CSS

10. JavaScript: Adding interactivity - Learn web development | MDN. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Learn_web_development/Getting_started/Your_first_website/Adding_interactivity

11. UMLetino - Free Online UML Tool for Fast UML Diagrams. UMLetino - Free Online UML Tool for Fast UML Diagrams. URL: https://www.umletino.com/

12. UML Use Case Diagram Tutorial. Lucidchart. URL: https://www.lucidchart.com/pages/uml-use-case-diagram

13. Package Diagram Tutorial. Lucidchart. URL: https://www.lucidchart.com/pages/uml-package-diagram

14. Sequence Diagram for Online Quiz System | Creately. Creately | Visual Collaboration & Diagramming Platform. URL: https://creately.com/diagram/example/hwmv3zx6/sequence-diagram-for-online-quiz-system

15. What is Deployment Diagram?. Ideal Modeling & Diagramming Tool for Agile Team Collaboration. URL: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-deployment-diagram/

16. Guide to Internationalization in Spring Boot | Baeldung. Baeldung. URL: https://www.baeldung.com/spring-boot-internationalization

17. Web Audio API - Web APIs | MDN. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API

18. Event reference | MDN. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/Events

19. The Graphics Canvas element - HTML: HyperText Markup Language | MDN. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/HTML/Reference/Elements/canvas

20. GeeksforGeeks. Black Box Testing - Software Engineering - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/software-engineering-black-box-testing/

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

РОЗРОБКА ЗАСТОСУНКУ ДЛЯ НАВЧАННЯ МАЛЕНЬКИХ ДІТЕЙ ЛІТЕРАМ ТА ЗВУКАМ

Виконав студент 4 курсу
групи ПД-42
Ярина Тимофій Олегович
Керівник роботи
старший викладач кафедри ІПЗ Андрусевич Наталя Володимирівна

Київ – 2025

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи:** вдосконалення навчання маленьких дітей літерам та звукам.
- **Об'єкт дослідження:** процеси навчання маленьких дітей літерам та звукам.
- **Предмет дослідження:** вебзастосунок для навчання маленьких дітей літерам та звукам.

ЗАДАЧІ КВАЛІФІКАЦІЙНОЇ РОБОТИ

1. Провести аналіз існуючих застосунків для навчання маленьких дітей літерам та звукам.
2. Визначити потреби користувачів та сформулювати функціональні та нефункціональні вимоги.
3. Провести огляд та аналіз засобів реалізації застосунку для навчання маленьких дітей літерам та звукам.
4. Спроєктувати архітектуру та інтерфейс застосунку для навчання маленьких дітей літерам та звукам.
5. Розробити застосунок відповідно до визначених вимог.
6. Перевірити систему на відповідність вимогам.

3

АНАЛІЗ АНАЛОГІВ

Критерії	Вчимося читати - вчимо літери	Українська абетка для дітей	Розумники(Smart Kids)	Буквознайка/Alphabuddy
Платформа	Android, iOS	Android, iOS	Web	Web
Перевірка знань	+	-	+	+
Прослуховування абетки у вигляді пісні	-	-	-	+
Аудіо супровід кожної літери	+	+	+	+
Малювання літер	-	-	-	+
Англомовна версія застосунку	-	-	-	+

4

ВИМОГИ ДО ЗАСТОСУНКУ

Функціональні вимоги:

1. Відображення літер з можливістю їх прослуховувати – аудіо-озвучення кожної літери.
2. Написання літери мишею – тренування моторики дитини.
3. Повне прослуховування абетки за допомогою пісні.
4. Візуальні асоціації кожної літери, відповідна картинка до кожної літери.
5. Ігрові розділи, які дозволяють користувачу вивчати літери та звуки.
6. Можливість перемикання між різними мовами – англійська та українська.

Нефункціональні вимоги:

1. Україномовна та англомовна локалізація – повна підтримка мови інтерфейсу.
2. Інтерфейс має бути адаптованим для дітей віком 3–6 років.
3. Продуктивна робота застосунку – час відгуку на дії користувача не більше 0,2 секунд.
4. Мінімум текстової інформації – зосередження на малюнках.
5. Інтуїтивні взаємодії – натиснув та отримав відгук.

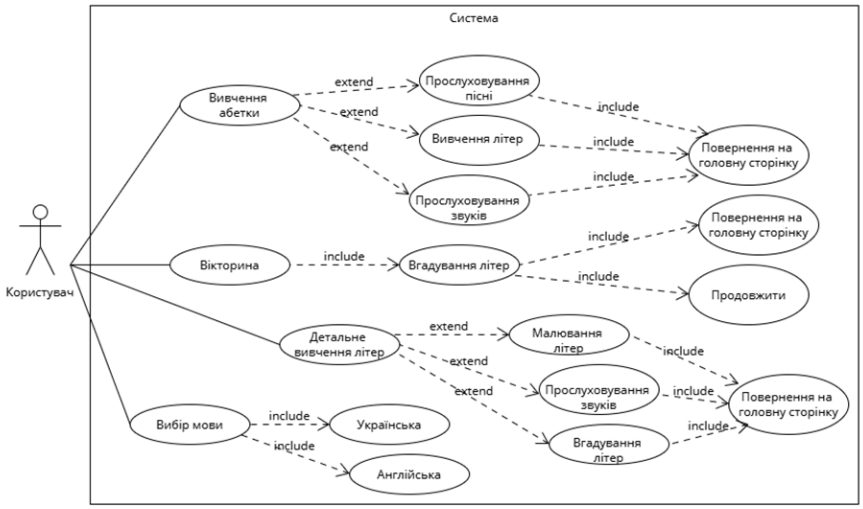
5

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

intellij idea

6

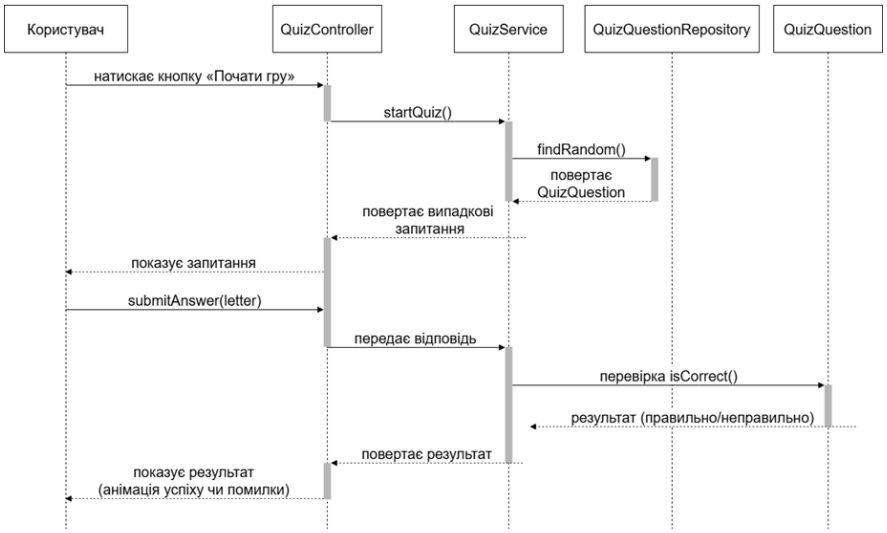
МОДЕЛЮВАННЯ ФУНКЦІОНАЛЬНОСТІ



Діаграма варіантів використання

7

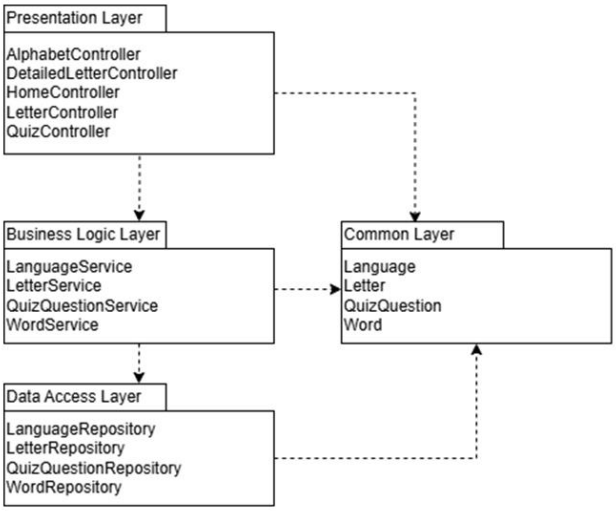
АЛГОРИТМ ПРОХОДЖЕННЯ ВІКТОРИНИ



Діаграма послідовності

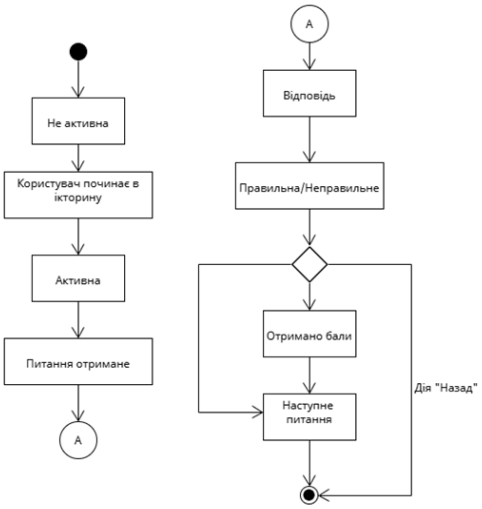
8

Бізнес логіка



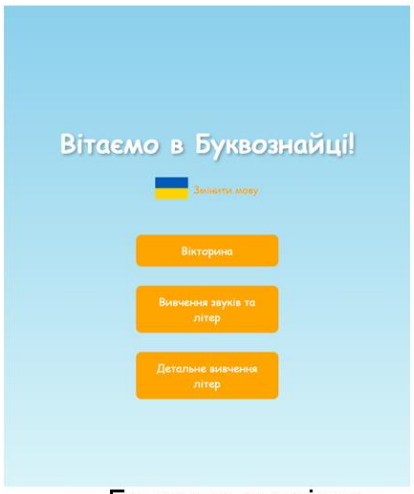
Діаграма пакетів 9

Алгоритм роботи застосунку

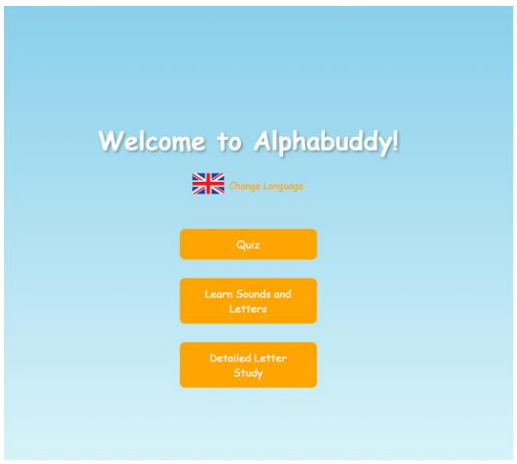


Блок-схема алгоритму вікторини 10

ЕКРАННІ ФОРМИ



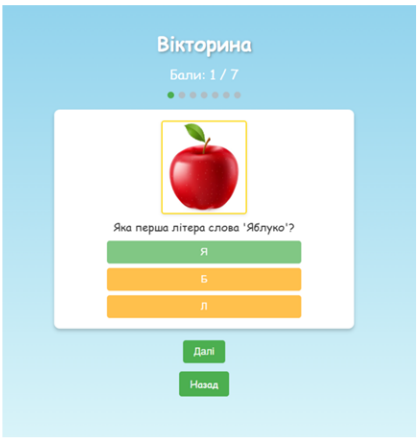
Головна сторінка



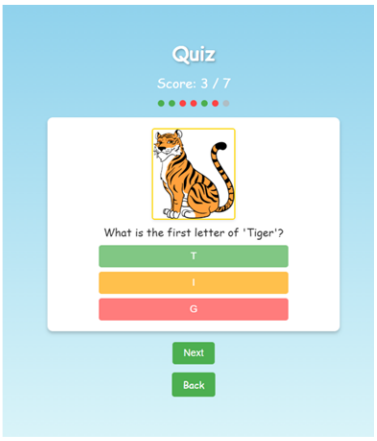
Головна сторінка англомовної версії

11

ЕКРАННІ ФОРМИ



Вікторина українською мовою

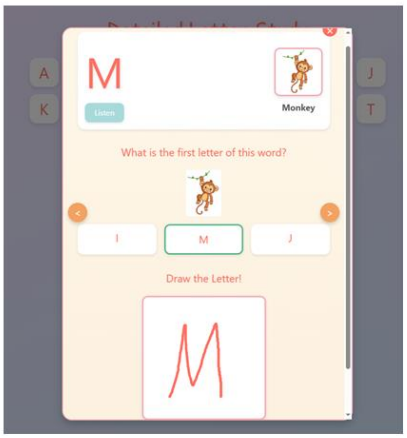


Вікторина англійською мовою

12

ЕКРАННІ ФОРМИ

Розділ з інтерактивною абеткою



Розділ детального вивчення літер

13

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1. Ярина Т.О., Андрусевич Н.В. Діаграма предметної області застосунку для навчання маленьких дітей літерам та звукам: Матеріали VI Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях». 24.04.25 Державний університет інформаційно-комунікаційних технологій, Київ, Україна, 2025. Збірник тез К.: ДУІКТ, С.117-119.

2. Ярина Т.О., Андрусевич Н.В Огляд програм-аналогів застосунку для навчання маленьких дітей літерам та звукам: Матеріали VI Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях». 24.04.25 Державний університет інформаційно-комунікаційних технологій, Київ, Україна, 2025. Збірник тез К.: ДУІКТ, С.172-174.

14

ВИСНОВКИ

1. Проведено детальний аналіз існуючих застосунків для навчання дітей літерам та звукам, що дало змогу виявити їхні сильні сторони та недоліки.
2. Визначено основні потреби користувачів — дітей та їх батьків, а також сформульовано функціональні та нефункціональні вимоги до застосунку.
3. Проведено огляд та аналіз доступних технологій і засобів розробки застосунків, що дало змогу вибрати оптимальний стек технологій для реалізації проєкту з урахуванням специфіки цільової аудиторії.
4. Спроєктовано архітектуру застосунку, яка забезпечує логічну структуру та гнучкість у подальшому розвитку, а також розроблено інтерфейс, адаптований до потреб маленьких дітей.
5. Реалізовано ключові функціональні модулі: абетку для знайомства з літерами, вікторину для перевірки знань та інтерактивний розділ з малювання літер, що сприяє кращому засвоєнню матеріалу через гру та творчість.
6. Проведено мануальне тестування, яке підтвердило відповідність вимогам.

15

ДОДАТОК Б. ЛІСТИНГИ ОСНОВНИХ МОДУЛЕЙ

```
package com.yaryna.bukvoznayka.controller;

import
com.yaryna.bukvoznayka.service.LanguageService;
import
com.yaryna.bukvoznayka.service.LetterService;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import
org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.PathVariable;

@Controller
public class AlphabetController {
    @Autowired
    private LanguageService languageService;
    @Autowired
    private LetterService letterService;

    @GetMapping("/alphabet/{lang}")
    public String showAlphabet(@PathVariable
String lang, Model model) {
        var language =
languageService.getLanguageByName(lang);
        var letters =
letterService.getLettersByLanguage(language);
        model.addAttribute("letters", letters);
        model.addAttribute("currentLang", lang);
        return "alphabet";
    }
}
```

```
package com.yaryna.bukvoznayka.controller;

import com.yaryna.bukvoznayka.model.Letter;
import com.yaryna.bukvoznayka.model.Word;
import
com.yaryna.bukvoznayka.service.LanguageService;
import
com.yaryna.bukvoznayka.service.LetterService;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import
org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.PathVariable;

import java.util.*;

@Controller
public class DetailedLetterController {
    @Autowired
    private LanguageService languageService;
    @Autowired
    private LetterService letterService;

    @GetMapping("/letters/{lang}")
    public String
showDetailedLetters(@PathVariable String lang, Model
model) {
        var language =
languageService.getLanguageByName(lang);
        if (language == null) {
            model.addAttribute("errorMessage", "Мова
не знайдена");
            return "error";
        }
        var letters =
letterService.getLettersByLanguage(language);
        if (letters == null || letters.isEmpty()) {
            model.addAttribute("letters",
Collections.emptyList());
            model.addAttribute("currentLang", lang);
            return "detailed-letters";
        }

        // Додаємо індекси до списку літер
        List<Map<String, Object>> indexedLetters =
new ArrayList<>();
        for (int i = 0; i < letters.size(); i++) {
            Map<String, Object> letterData = new
HashMap<>();
            letterData.put("letter", letters.get(i));
            letterData.put("index", i);
            indexedLetters.add(letterData);
        }

        // Для кожної літери додаємо дані для
завдання "Яка перша літера цього слова?"
        Map<Long, Map<String, Object>> gameData
= new HashMap<>();
        Random random = new Random();
        List<Letter> allLetters = new
ArrayList<>(letters);

        for (Letter letter : letters) {
            Map<String, Object> letterGameData =
new HashMap<>();

            // Вибираємо слово для завдання
            List<Word> words = letter.getWords();
            Word selectedWord =
words.get(random.nextInt(words.size()));
            letterGameData.put("selectedWord",
selectedWord);

            // Вибираємо два неправильні варіанти
(літери)
            String correctLetter = letter.getLetter();
            String wrongLetter1, wrongLetter2;
            do {
                wrongLetter1 =
allLetters.get(random.nextInt(allLetters.size())).getLetter();
            } while
(wrongLetter1.equals(correctLetter));
            do {
                wrongLetter2 =
allLetters.get(random.nextInt(allLetters.size())).getLetter();
            } while (wrongLetter2.equals(correctLetter)
|| wrongLetter2.equals(wrongLetter1));

            // Перемішуємо варіанти
            List<String> options =
Arrays.asList(correctLetter, wrongLetter1, wrongLetter2);
            Collections.shuffle(options);
            letterGameData.put("option1",
options.get(0));
            letterGameData.put("option2",
options.get(1));
            letterGameData.put("option3",
options.get(2));
```

```
                gameData.put(letter.getId(),
letterGameData);
            }

            model.addAttribute("indexedLetters",
indexedLetters);
            model.addAttribute("letters", letters);
            model.addAttribute("gameData", gameData);
            model.addAttribute("currentLang", lang);
            return "detailed-letters";
        }
    }
```

```
package com.yaryna.bukvoznayka.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import
org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.RequestParam;

@Controller
public class HomeController {

    @GetMapping("/")
    public String home(@RequestParam(value =
"lang", defaultValue = "Ukrainian") String lang, Model
model) {
        // Перевіряємо значення lang і дозволяємо
лише "Ukrainian" або "English"
        String currentLang = "Ukrainian"; // Значення
за замовчуванням
        if ("English".equalsIgnoreCase(lang)) {
            currentLang = "English";
        } else if ("Ukrainian".equalsIgnoreCase(lang))
{
            currentLang = "Ukrainian";
        }

        // Встановлюємо шлях до прапора залежно
від currentLang
        String flag = currentLang.equals("Ukrainian")
? "/images/flags/ua.jpg" : "/images/flags/uk.jpg";

        // Додаємо атрибути до моделі
        model.addAttribute("currentLang",
currentLang);
        model.addAttribute("flag", flag);

        return "home";
    }
}
```

```
package com.yaryna.bukvoznayka.controller;

import
com.yaryna.bukvoznayka.service.LetterService;
import
com.yaryna.bukvoznayka.service.WordService;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import
org.springframework.web.bind.annotation.GetMapping;

package com.yaryna.bukvoznayka.service;
```

```
import
org.springframework.web.bind.annotation.PathVariable;

@Controller
public class LetterController {
    @Autowired
    private LetterService letterService;
    @Autowired
    private WordService wordService;

    @GetMapping("/letter/{id}")
    public String showLetter(@PathVariable Long
id, Model model) {
        var letter = letterService.getLetterById(id);
        var words =
wordService.getWordsByLetter(letter);
        model.addAttribute("letter", letter);
        model.addAttribute("words", words);
        model.addAttribute("currentLang",
letter.getLanguage());
        return "letter";
    }
}
```

```
package com.yaryna.bukvoznayka.controller;

import
com.yaryna.bukvoznayka.model.QuizQuestion;
import
com.yaryna.bukvoznayka.service.LanguageService;
import
com.yaryna.bukvoznayka.service.QuizQuestionService;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import
org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.PathVariable;

@Controller
public class QuizController {
    @Autowired
    private LanguageService languageService;
    @Autowired
    private QuizQuestionService
quizQuestionService;

    @GetMapping("/quiz/{lang}")
    public String showQuiz(@PathVariable String
lang, Model model) {
        var language =
languageService.getLanguageByName(lang);
        if (language == null) {
            model.addAttribute("errorMessage", "Мова
не знайдена");
            return "error";
        }
        var questions =
quizQuestionService.getQuestionsByLanguage(language);
        model.addAttribute("questions", questions !=
null ? questions : java.util.Collections.emptyList());
        model.addAttribute("currentLang", lang);
        return "quiz";
    }
}
import com.yaryna.bukvoznayka.model.Language;
import
com.yaryna.bukvoznayka.repository.LanguageRepository;
```

```
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class LanguageService {
   @Autowired
   private LanguageRepository
languageRepository;
}

package com.yaryna.bukvoznayka.service;

import com.yaryna.bukvoznayka.model.Language;
import com.yaryna.bukvoznayka.model.Letter;
import
com.yaryna.bukvoznayka.repository.LetterRepository;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class LetterService {
   @Autowired
   private LetterRepository letterRepository;

   public List<Letter>
getLettersByLanguage(Language language) {
      return
letterRepository.findByLanguage(language);
   }

   public Letter getLetterById(Long id) {
      return
letterRepository.findById(id).orElseThrow(() -> new
RuntimeException("Letter not found"));
   }
}

package com.yaryna.bukvoznayka.service;

import com.yaryna.bukvoznayka.model.Language;
import
com.yaryna.bukvoznayka.model.QuizQuestion;
import
com.yaryna.bukvoznayka.repository.QuizQuestionRepositor
y;
import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class QuizQuestionService {
   @Autowired
   private QuizQuestionRepository
quizQuestionRepository;

   public List<QuizQuestion>
getQuestionsByLanguage(Language language) {
      return
quizQuestionRepository.findByLanguage(language);
   }
}

package com.yaryna.bukvoznayka.service;

   public Language getLanguageByName(String
name) {
      return
languageRepository.findByName(name).orElseThrow(() ->
new RuntimeException("Language not found"));
   }


import com.yaryna.bukvoznayka.model.Letter;
import com.yaryna.bukvoznayka.model.Word;
import
com.yaryna.bukvoznayka.repository.WordRepository;
import
org.springframework.beans.factory.annotation.Autowired;

package com.yaryna.bukvoznayka.config;

import com.yaryna.bukvoznayka.model.Language;
import com.yaryna.bukvoznayka.model.Letter;
import
com.yaryna.bukvoznayka.model.QuizQuestion;
import com.yaryna.bukvoznayka.model.Word;
import
com.yaryna.bukvoznayka.repository.LanguageRepository;
import
com.yaryna.bukvoznayka.repository.LetterRepository;
import
com.yaryna.bukvoznayka.repository.QuizQuestionRepositor
y;
import
com.yaryna.bukvoznayka.repository.WordRepository;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import
org.springframework.beans.factory.annotation.Autowired;
import
org.springframework.boot.CommandLineRunner;
import
org.springframework.stereotype.Component;

import java.util.Arrays;
import java.util.Collections;
import java.util.List;
import java.util.Locale;
import java.util.Optional;

@Component
public class DataInitializer implements
CommandLineRunner {
   private static final Logger logger =
LoggerFactory.getLogger(DataInitializer.class);

   @Autowired
   private LanguageRepository
languageRepository;
   @Autowired
   private LetterRepository letterRepository;
   @Autowired
   private WordRepository wordRepository;
   @Autowired
   private QuizQuestionRepository
quizQuestionRepository;

   @Override
   public void run(String... args) {
      logger.info("Початок ініціалізації даних");

      // Створюємо мови, якщо їх немає
```

```
        Language ukrainian = createLanguageIfNotExists("Ukrainian");
        Language english = createLanguageIfNotExists("English");

        // Український алфавіт (33 літери)
        String[] ukrLetters = {
            "А", "Б", "В", "Г", "Ґ", "Д", "Е", "Є", "Ж", "З", "И",
            "І", "Ї", "Й", "К", "Л", "М", "Н", "О", "П", "Р", "С",
            "Т", "У", "Ф", "Х", "Ц", "Ч", "Ш", "Щ", "Ь", "Ю", "Я"
        };
        String[] ukrWords = {
            "Автомобіль", "Бджола", "Вишня", "Гусак", "Ґудзик", "Диня", "Екран", "Єнот",
            "Жук", "Зебра", "Ірій", "Індик", "Їжак", "Йогурт", "Кіт", "Лисиця",
            "Мишка", "Ніс", "Огірок", "Пес", "Ракета", "Сонце", "Торт", "Удав",
            "Фламінго", "Хатина", "Цукерка", "Чашка", "Шапка", "Щука", "Кінь", "Юнга",
            "Яблуко"
        };
        String[] ukrImages = {
            "avtomobil.jpg", "bdzhola.jpg", "vyshnya.jpg", "husak.jpg", "gudzyk.jpg",
            "dynya.jpg", "ekran.jpg", "yenot.jpg", "zhuk.jpg", "zebra.jpg",
            "yriy.jpg", "indyk.jpg", "yizhak.jpg", "yohurt.jpg", "kit.jpg",
            "lysytsya.jpg", "myshka.jpg", "nis.jpg", "ogirok.jpg", "pes.jpg",
            "raketa.jpg", "sontse.jpg", "tort.jpg", "udav.jpg", "flamingo.jpg",
            "hatyna.jpg", "tsukerka.jpg", "chashka.jpg", "shapka.jpg", "shchuka.jpg",
            "kin.jpg", "yunga.jpg", "yabluko.jpg"
        };
        String[] ukrSoundFiles = {
            "letter_a.mp3", "letter_b.mp3", "letter_v.mp3", "letter_h.mp3", "letter_g.mp3",
            "letter_d.mp3", "letter_e.mp3", "letter_ye.mp3", "letter_zh.mp3", "letter_z.mp3",
            "letter_y.mp3", "letter_i.mp3", "letter_yi.mp3", "letter_yot.mp3", "letter_k.mp3",
            "letter_l.mp3", "letter_m.mp3", "letter_n.mp3", "letter_o.mp3", "letter_p.mp3",
            "letter_r.mp3", "letter_s.mp3", "letter_t.mp3", "letter_u.mp3", "letter_f.mp3",
            "letter_kh.mp3", "letter_ts.mp3", "letter_ch.mp3", "letter_sh.mp3", "letter_shch.mp3",
            "letter_miakyi_znak.mp3", "letter_yu.mp3", "letter_ya.mp3"
        };

        for (int i = 0; i < ukrLetters.length; i++) {
            try {
                logger.info("Обробка літери: {}", ukrLetters[i]);
                if (!letterExists(ukrainian, ukrLetters[i])) {
                    logger.info("Літера {} ще не існує, додаємо її", ukrLetters[i]);
                    Letter letter = new Letter(ukrainian, ukrLetters[i], ukrSoundFiles[i], ukrImages[i]);
                    letter = letterRepository.save(letter);
                    logger.info("Літера {} успішно додана з id={}", ukrLetters[i], letter.getId());
                    if (!wordExists(letter, ukrWords[i])) {
                        logger.info("Додаємо слово для літери {}: {}", ukrLetters[i], ukrWords[i]);
                        wordRepository.save(new Word(letter, ukrWords[i], ukrImages[i]));
                        logger.info("Слово {} успішно додане", ukrWords[i]);
                    }
                } else {
                    logger.info("Літера {} вже існує", ukrLetters[i]);
                }
            } catch (Exception e) {
                logger.error("Помилка при додаванні літери {}: {}", ukrLetters[i], e.getMessage(), e);
            }
        }

        // Англійський алфавіт (26 літер)
        String[] engLetters = {
            "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M",
            "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z"
        };
        String[] engWords = {
            "Apple", "Ball", "Cat", "Dog", "Elephant",
            "Fish", "Goose", "Hat", "Ice", "Jar",
            "Kite", "Lion", "Monkey", "Nest", "Orange",
            "Pen", "Queen", "Rabbit", "Sun", "Tiger",
            "Umbrella", "Violin", "Whale", "Xylophone", "Yogurt",
            "Zebra"
        };
        String[] engImages = {
            "apple.jpg", "ball.jpg", "cat.jpg", "dog.jpg", "elephant.jpg",
            "fish.jpg", "goose.jpg", "hat.jpg", "ice.jpg", "jar.jpg",
            "kite.jpg", "lion.jpg", "monkey.jpg", "nest.jpg", "orange.jpg",
            "pen.jpg", "queen.jpg", "rabbit.jpg", "sun.jpg", "tiger.jpg",
            "umbrella.jpg", "violin.jpg", "whale.jpg", "xylophone.jpg", "yogurt.jpg",
            "zebra.jpg"
        };
        String[] engSoundFiles = {
            "letter-a.mp3", "letter-b.mp3", "letter-c.mp3", "letter-d.mp3", "letter-e.mp3",
            "letter-f.mp3", "letter-g.mp3", "letter-h.mp3", "letter-i.mp3", "letter-j.mp3",
            "letter-k.mp3", "letter-l.mp3", "letter-m.mp3", "letter-n.mp3", "letter-o.mp3",
            "letter-p.mp3", "letter-q.mp3", "letter-r.mp3", "letter-s.mp3", "letter-t.mp3",
            "letter-u.mp3", "letter-v.mp3", "letter-w.mp3", "letter-x.mp3", "letter-y.mp3",
            "letter-z.mp3"
        };

        for (int i = 0; i < engLetters.length; i++) {
            try {
                logger.info("Обробка літери: {}", engLetters[i]);
                if (!letterExists(english, engLetters[i])) {
```

```
                    logger.info("Літера {} ще не існує, додаємо її", engLetters[i]);
                    Letter letter = new Letter(english, engLetters[i], engSoundFiles[i], engImages[i]);
                    letter = letterRepository.save(letter);
                    logger.info("Літера {} успішно додана з id={}", engLetters[i], letter.getId());
                    if (!wordExists(letter, engWords[i])) {
                        logger.info("Додаємо слово для літери {}: {}", engLetters[i], engWords[i]);
                        wordRepository.save(new Word(letter, engWords[i], engImages[i]));
                        logger.info("Слово {} успішно додане", engWords[i]);
                    }
                } else {
                    logger.info("Літера {} вже існує", engLetters[i]);
                }
            } catch (Exception e) {
                logger.error("Помилка при додаванні літери {}: {}", engLetters[i], e.getMessage(), e);
            }
        }

        // Ініціалізація питань для вікторини
        initializeQuizQuestions();
        logger.info("Ініціалізація даних завершена");
    }

    private Language createLanguageIfNotExists(String name) {
        Optional<Language> existingLanguage = languageRepository.findByName(name);
        if (existingLanguage.isPresent()) {
            logger.info("Мова {} вже існує з id={}", name, existingLanguage.get().getId());
            return existingLanguage.get();
        }
        Language language = new Language(name);
        language = languageRepository.save(language);
        logger.info("Мова {} створена з id={}", name, language.getId());
        return language;
    }

    private boolean letterExists(Language language, String letter) {
        logger.debug("Перевіряємо, чи існує літера {} для мови {}", letter, language.getName());
        Optional<Letter> existingLetter = letterRepository.findByLanguageAndLetter(language.getId(), letter);
        if (existingLetter.isPresent()) {
            logger.debug("Літера {} вже існує з id={}", letter, existingLetter.get().getId());
        } else {
            logger.debug("Літера {} не існує", letter);
        }
        return existingLetter.isPresent();
    }

    private boolean wordExists(Letter letter, String word) {
        return wordRepository.findByLetterAndWord(letter, word).isPresent();
    }

    private void shuffleOptions(QuizQuestion question) {
        List<String> options = Arrays.asList(question.getOption1(), question.getOption2(), question.getOption3());
        Collections.shuffle(options);
        question.setOption1(options.get(0));
        question.setOption2(options.get(1));
        question.setOption3(options.get(2));
    }

    private void initializeQuizQuestions() {
        Language ukrainian = createLanguageIfNotExists("Ukrainian");
        Language english = createLanguageIfNotExists("English");

        // Питання для української мови (7 питань)
        if (quizQuestionRepository.findByQuestion("Яка перша літера слова 'Яблуко'?").isEmpty()) {
            QuizQuestion q1 = new QuizQuestion();
            q1.setLanguage(ukrainian);
            q1.setQuestion("Яка перша літера слова 'Яблуко'?");
            q1.setCorrectAnswer("Я");
            q1.setOption1("Б");
            q1.setOption2("Я");
            q1.setOption3("Л");
            q1.setImageFile("yabluko.jpg");
            shuffleOptions(q1);
            quizQuestionRepository.save(q1);
        }

        if (quizQuestionRepository.findByQuestion("Яка перша літера слова 'Бджола'?").isEmpty()) {
            QuizQuestion q2 = new QuizQuestion();
            q2.setLanguage(ukrainian);
            q2.setQuestion("Яка перша літера слова 'Бджола'?");
            q2.setCorrectAnswer("Б");
            q2.setOption1("Д");
            q2.setOption2("Б");
            q2.setOption3("Ж");
            q2.setImageFile("bdzhola.jpg");
            shuffleOptions(q2);
            quizQuestionRepository.save(q2);
        }

        if (quizQuestionRepository.findByQuestion("Яка перша літера слова 'Вишня'?").isEmpty()) {
            QuizQuestion q3 = new QuizQuestion();
            q3.setLanguage(ukrainian);
            q3.setQuestion("Яка перша літера слова 'Вишня'?");
            q3.setCorrectAnswer("В");
            q3.setOption1("Ш");
            q3.setOption2("В");
            q3.setOption3("Н");
            q3.setImageFile("vyshnya.jpg");
            shuffleOptions(q3);
            quizQuestionRepository.save(q3);
        }

        if (quizQuestionRepository.findByQuestion("Яка перша літера слова 'Кіт'?").isEmpty()) {
```

```
                QuizQuestion q4 = new QuizQuestion();
                q4.setLanguage(ukrainian);
                q4.setQuestion("Яка перша літера слова
'Кіт'?");
                q4.setCorrectAnswer("К");
                q4.setOption1("І");
                q4.setOption2("Т");
                q4.setOption3("К");
                q4.setImageFile("kit.jpg");
                shuffleOptions(q4);
                quizQuestionRepository.save(q4);
            }

            if
(quizQuestionRepository.findByQuestion("Яка перша
літера слова 'Сонце'?").isEmpty()) {
                QuizQuestion q5 = new QuizQuestion();
                q5.setLanguage(ukrainian);
                q5.setQuestion("Яка перша літера слова
'Сонце'?");
                q5.setCorrectAnswer("С");
                q5.setOption1("О");
                q5.setOption2("С");
                q5.setOption3("Н");
                q5.setImageFile("sontse.jpg");
                shuffleOptions(q5);
                quizQuestionRepository.save(q5);
            }

            if
(quizQuestionRepository.findByQuestion("Яка перша
літера слова 'Торт'?").isEmpty()) {
                QuizQuestion q6 = new QuizQuestion();
                q6.setLanguage(ukrainian);
                q6.setQuestion("Яка перша літера слова
'Торт'?");
                q6.setCorrectAnswer("Т");
                q6.setOption1("О");
                q6.setOption2("Р");
                q6.setOption3("Т");
                q6.setImageFile("tort.jpg");
                shuffleOptions(q6);
                quizQuestionRepository.save(q6);
            }

            if
(quizQuestionRepository.findByQuestion("Яка перша
літера слова 'Ракета'?").isEmpty()) {
                QuizQuestion q7 = new QuizQuestion();
                q7.setLanguage(ukrainian);
                q7.setQuestion("Яка перша літера слова
'Ракета'?");
                q7.setCorrectAnswer("Р");
                q7.setOption1("А");
                q7.setOption2("Р");
                q7.setOption3("К");
                q7.setImageFile("raketa.jpg");
                shuffleOptions(q7);
                quizQuestionRepository.save(q7);
            }

            // Питання для англійської мови (7 питань)
            if
(quizQuestionRepository.findByQuestion("What is the first
letter of 'Apple'?").isEmpty()) {
                QuizQuestion q1 = new QuizQuestion();
                q1.setLanguage(english);
                q1.setQuestion("What is the first letter of
'Apple'?");
                q1.setCorrectAnswer("A");
                q1.setOption1("B");
                q1.setOption2("A");
                q1.setOption3("P");
                q1.setImageFile("apple.jpg");
                shuffleOptions(q1);
                quizQuestionRepository.save(q1);
            }

            if
(quizQuestionRepository.findByQuestion("What is the first
letter of 'Ball'?").isEmpty()) {
                QuizQuestion q2 = new QuizQuestion();
                q2.setLanguage(english);
                q2.setQuestion("What is the first letter of
'Ball'?");
                q2.setCorrectAnswer("B");
                q2.setOption1("A");
                q2.setOption2("B");
                q2.setOption3("L");
                q2.setImageFile("ball.jpg");
                shuffleOptions(q2);
                quizQuestionRepository.save(q2);
            }

            if
(quizQuestionRepository.findByQuestion("What is the first
letter of 'Cat'?").isEmpty()) {
                QuizQuestion q3 = new QuizQuestion();
                q3.setLanguage(english);
                q3.setQuestion("What is the first letter of
'Cat'?");
                q3.setCorrectAnswer("C");
                q3.setOption1("A");
                q3.setOption2("C");
                q3.setOption3("T");
                q3.setImageFile("cat.jpg");
                shuffleOptions(q3);
                quizQuestionRepository.save(q3);
            }

            if
(quizQuestionRepository.findByQuestion("What is the first
letter of 'Dog'?").isEmpty()) {
                QuizQuestion q4 = new QuizQuestion();
                q4.setLanguage(english);
                q4.setQuestion("What is the first letter of
'Dog'?");
                q4.setCorrectAnswer("D");
                q4.setOption1("O");
                q4.setOption2("D");
                q4.setOption3("G");
                q4.setImageFile("dog.jpg");
                shuffleOptions(q4);
                quizQuestionRepository.save(q4);
            }

            if
(quizQuestionRepository.findByQuestion("What is the first
letter of 'Sun'?").isEmpty()) {
                QuizQuestion q5 = new QuizQuestion();
                q5.setLanguage(english);
                q5.setQuestion("What is the first letter of
'Sun'?");
                q5.setCorrectAnswer("S");
                q5.setOption1("U");
                q5.setOption2("S");
                q5.setOption3("N");
                q5.setImageFile("sun.jpg");
                shuffleOptions(q5);
                quizQuestionRepository.save(q5);
```

```
                }

                if
(quizQuestionRepository.findByQuestion("What is the first
letter of 'Tiger'?").isEmpty()) {
                    QuizQuestion q6 = new QuizQuestion();
                    q6.setLanguage(english);
                    q6.setQuestion("What is the first letter of
'Tiger'?");
                    q6.setCorrectAnswer("T");
                    q6.setOption1("I");
                    q6.setOption2("T");
                    q6.setOption3("G");
                    q6.setImageFile("tiger.jpg");
                    shuffleOptions(q6);
                    quizQuestionRepository.save(q6);
                }

                if
(quizQuestionRepository.findByQuestion("What is the first
letter of 'Zebra'?").isEmpty()) {
                    QuizQuestion q7 = new QuizQuestion();
                    q7.setLanguage(english);
                    q7.setQuestion("What is the first letter of
'Zebra'?");
                    q7.setCorrectAnswer("Z");
                    q7.setOption1("E");
                    q7.setOption2("Z");
                    q7.setOption3("B");
                    q7.setImageFile("zebra.jpg");
                    shuffleOptions(q7);
                    quizQuestionRepository.save(q7);
                }
            }
        }
        import org.springframework.stereotype.Service;

        import java.util.List;

        @Service
        public class WordService {
            @Autowired
            private WordRepository wordRepository;

            public List<Word> getWordsByLetter(Letter
letter) {
                return wordRepository.findByLetter(letter);
            }
        }
```