МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

# Кваліфікаційна робота бакалавра

на тему: Розробка навчального веб-сервісу з іноземних мов

Виконав студент групи **К-21і**
спеціальності 122 Комп'ютерні науки
Туз Герман Вячеславович

Керівник **Ковальчук В.В.,**
**д.ф.-м.н., професор**

Консультант

Рецензент **Квасніков В.П.,**
**д.т.н, проф, зав каф.інф.**
**технологій Київського**
**національного авіаційного**
**університету**

Одеса 2023

**ЗМІСТ**

## ТЕРМІНИ І СКОРОЧЕННЯ

ANSI – об'єднання американських промислових і ділових груп, що розробляє торгові і комунікаційні стандарти.

API – це абривіатура «Application Programming Interface», інтерфейс програмування додатків, програмний інтерфейс програми.

БД – база даних.

ОС – операційна система.

ПК – персональний комп’ютер.

СУБД – система керування базами даних.

# ВСТУП

Інформаційні технології в сучасному світі є невід'ємною складовою життя людей, що значно впливає на їхній спосіб навчання та отримання нових знань. Розвиток та вдосконалення програмних систем сприяє підвищенню доступності та зручності отримання нових знань та навичок.

У зв'язку з останніми подіями, багато сімей з дітьми були вимушені виїхати з України до інших країн. В більшості випадків діти не знають мови країни до якої вони переїхали. Одним з ефективних шляхів виховання інтересу до вивчення іноземної мови є ігри. По суті, гра – це тренажер, на якому виробляються вміння і навички. Інтерес і задоволення – найважливіші психологічні ефекти гри.

Для поглиблення і узагальнити знання про основні правила мови, основні фрази, словосполучення, сприяти активізації пізнавальних інтересів завдяки створенню тренажера в ігровій формі, привабливій для дітей.

Для досягнення мети роботи вирішуються такі задачі:

- аналіз предметної області;
- виявлення та аналіз наявних аналогів;
- розробка програмного забезпечення для вивчення іноземних мов;
- створення бази даних;
- створення каталогу уроків для вивчення та закріплення мови;
- створення систем відстеження прогресу та винагород;
- створення словника;
- створення посібника основ мови.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

## 1.1 Роль веб-технологій в сучасній освіті

В наш час особливої актуальності набуває проблема розробки веб-сайтів дистанційного навчання, використання Internet-технологій у системі освіти та різних сервісів що дозволять учням більш продуктивно засвоювати навчальний матеріал. Будь-який імпульс до нового завжди і творчий та реакційний. Історія показує, нові технології займають свої ніші у суспільних відносинах, не замінюючи традиційні, а доповнюючи їх можливості. Підростаюче покоління більш сприйнятливе до різних новацій. Отже, актуальність постановки, вивчення та вирішення проблеми цілеспрямованого використання Internet-технологій в освіті загалом та для навчальних закладів зокрема, не викликає сумнівів.

Сучасний освітній процес неможливо уявити без використання інноваційних технологій. Інформаційно-комунікаційні технології дозволяють підвищити мотивацію та ефективність навчання, інтелектуальний рівень учнів, урізноманітнити форми міжособистісного спілкування всіх учасників освітнього процесу, удосконалювати методики проведення занять, в такий спосіб відкриваючи нові можливості у викладанні.

Використання технологій допомагає учням вирішувати складні завдання, вдосконалювати навички спілкування та лідерські якості, підвищувати рівень мотивації, продуктивності та творчий потенціал.

Застосування інформаційно-комунікаційних технологій у навчанні дозволяє самостійно набувати знання, а також зменшує потребу в посередницькій діяльності викладача, оскільки самостійна робота з електронними засобами навчання є основою прогресивного навчального процесу. Сучасні мережеві технології надають доступ до величезної кількості навчально-методичної та наукової літератури, допомагають проводити віртуальні заняття у режимі реального часу.

Інформаційно-комунікаційні технології включають різноманітні методи та програмно-технічні засоби роботи з даними. до них належать електронні підручники та посібники, електронні енциклопедії та довідники, ігрові платформи та програми тестування, широкий спектр освітніх ресурсів Інтернету.

Перелічені засоби дозволяють удосконалювати методики проведення занять, забезпечують доступ до різноманітних інформаційних ресурсів, надаючи можливість швидко знаходити літературу у різних пошукових системах, обробляти інформацію, працювати з текстами, зберігати інформацію на електронних носіях.

### 1.2 Аналіз існуючих аналогів

Був проведений аналіз існуючих аналогів застосунків для навчання, що демонструють ефективність короткого цілеспрямованого навчання. Вони пропонують поєднання мультимедійного вмісту, інтерактивних вправ та персоналізованих навчальних шляхів для покращення взаємодії та збереження знань. У застосуваннях використовуються різні методи, як-от інтервал повторення, гейміфікація та адаптивне навчання, щоб забезпечити ефективне та приємне навчання для користувачів.

Для кращого порівняння було обрано декілька готових програмних продуктів, які користуються найбільшою популярністю. Під час вибору основними критеріями оцінки були кількість завантажень та загальний рейтинг.

Далі опишемо деякі з них та виконаємо порівняльну характеристику.

Duolingo – це застосунок для вивчення мов. Завдяки своєму інноваційному підходу, Duolingo надає користувачам індивідуальний та ефективний шлях до опанування нової мови, бо вміло адаптується до потреб кожного студента. Застосунок постійно аналізує та оцінює прогрес

користувача, враховуючи його сильні та слабкі сторони [1]. Заснований на цих даних, Duolingo пропонує персоналізовані уроки та вправи, які підходять саме для конкретного користувача, щоб забезпечити оптимальний процес навчання. Застосунок надає миттєвий зворотний зв'язок щодо правильності відповідей, а також пропонує додаткові пояснення та вправи для подальшого зміцнення знань (рис.1.1).

Застосунок використовує гейміфікацію, нагороди та досягнення, що збуджує конкурентний дух та сприяє регулярному вивченню мови. Усі ці інноваційні використання в Duolingo роблять процес вивчення мови більш зручним, ефективним та захоплюючим. Користувачі можуть насолоджуватися індивідуальним підходом, точним оцінюванням та персоналізованими рекомендаціями, що прискорює їх прогрес і допомагає досягати високої якості вивчення мови. Таким чином, Duolingo – найпопулярніший додаток у світі в категорії "Навчання".

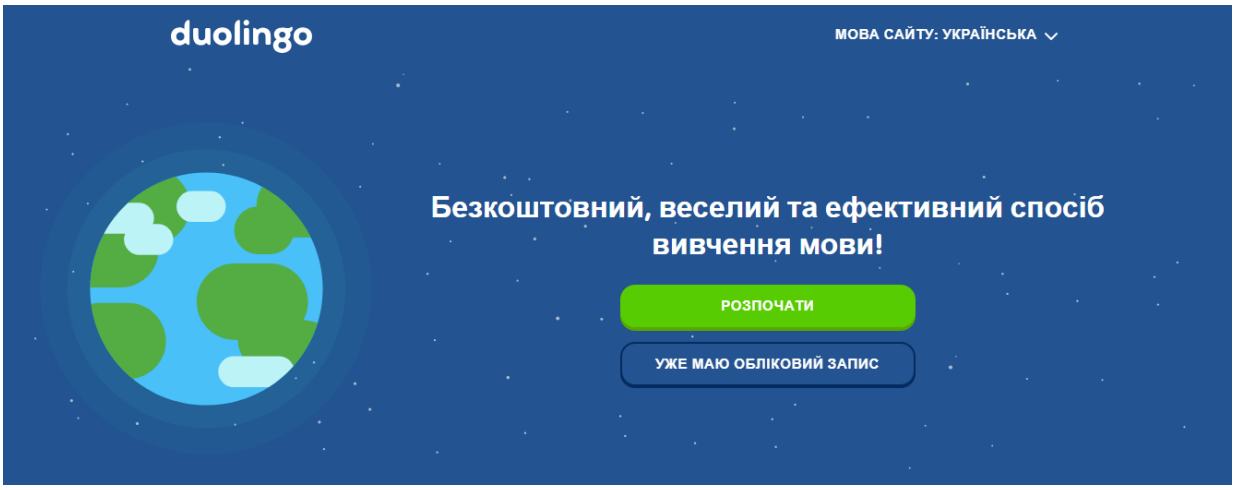


Рисунок 1.1 – Застосунок Duolingo для вивчення іноземної мови

Загальна кількість завантажень – 100.000.000+.

Загальний рейтинг – 4.3

Ще були виділені наступні конкуренти.

Додаток «Busuu» (рис.1.2) призначений для навчання 10 різних мов, у тому числі й англійської. Усі завдання у додатку розділені на 4 рівні, від A1 до B2. "Busuu" використовує наступний підхід для навчання англійської мови. Кожне завдання відноситься до різних життєвих ситуацій (предметних областей), а також воно розділене на кілька підзадань. Спочатку користувач поповнює свій словниковий запас з предметної області, що вивчається. Далі користувачеві дається виконання кілька граматичних вправ, у яких використовуються слова з обраної предметної області [2].

Для відстеження індивідуального прогресу програма підсвічує виконані завдання, тоді як інші залишаються затемненими. Таким чином, Busuu – це платформа для вивчення мов, яка дає користувачам змогу взаємодіяти з носіями мови.

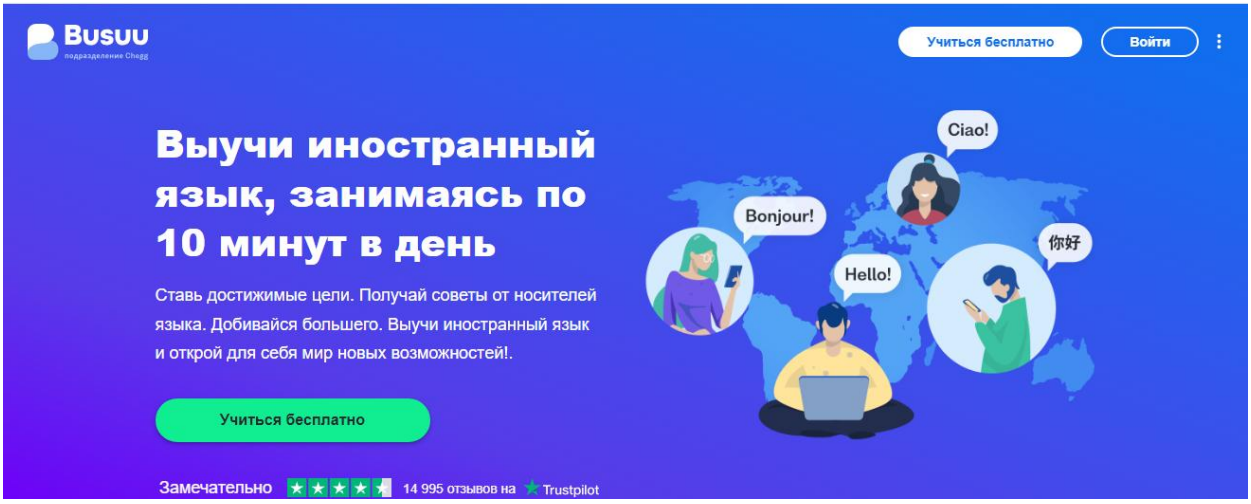


Рисунок 1.2 – Застосунок Busuu для вивчення іноземної мови

Загальна кількість завантажень – 10.000.000+.

Загальний рейтинг – 4.8.

Ще було розглянуто декілька застосунків. Їх порівняльна характеристика наведена у таблиці 1.1.

FunEasyLearn – безкоштовне та ефективне вивчення іноземних мов за допомогою ігор [3].

Загальна кількість завантажень – 1.000.000+.

Загальний рейтинг – 4.8.

Babbel – короткі інтерактивні уроки переосмислюють мовну освіту старої школи.

Загальна кількість завантажень – 50.000.000+.

Загальний рейтинг – 4.7.

Аналіз аналогів виявляється вельми цінним, оскільки надає нам можливість отримати цінні уроки, внутрішню інформацію та рекомендації, які відіграють ключову роль у розробці та успішному запровадженні нового продукту або послуги. Це дозволяє нам знизити ризики, підвищити конкурентоспроможність та надати нашим користувачам більш задоволений досвід, відповідаючи їх потребам і очікуванням на більш високому рівні.

Таблиця 1.1 – Порівняння систем навчання

| Назва системи | Різноманітність тестів | Артикль основ мов | Система нагород | Ситуаційні теми рівнів |
|---|---|---|---|---|
| Система, що розробляється | + | + | + | + |
| FunEasyLearn | + | - | + | + |
| Duolingo | + | - | + | - |
| Busuu | - | - | - | + |
| Babbel | + | + | - | + |

Отже, розроблювальна система повинна мати ситуаційну тематику рівнів, відповідати критеріям різноманітності тестів, наявність інформації про основи мов, та систему нагород

### 1.3 Постановка задачі

В роботі необхідно розробити веб-сервіс для навчання у вигляді мобільного додатку, який повинен працювати швидко та ефективно, щоб користувачі могли отримувати необхідну інформацію швидко і вчасно.

Додаток повинен бути зручним у використанні та мати інтуїтивно зрозумілий інтерфейс.

У додатку реалізувати:

- функції реєстрації/авторизації;
- можливість вибору мови;
- вибір уроку та навчального матеріалу по тематикам;
- обов’язкове тестування знань;
- словник.

Протестувати додаток.

# 2 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## 2.1 Особливості розробки мобільних додатків

Найчастіше мобільні програми використовують компанії, які працюють у наступних областях:

- онлайн-магазини;
- туризм, продаж квитків та бронювання готелів;
- медицина;
- освітні проекти;
- продаж, оренда, ремонт та обслуговування автомобілів;
- ресторани, кафе, розважальні заклади;
- салони краси та фітнес-центри;
- банки та компанії, що надають фінансові послуги.

Насамперед важливо визначитися з технологією розробки. Існує кілька видів технологій, проте найчастіше замовник обирає між нативними чи гібридними додатками.

Нативні програми розробляють під конкретну ОС. Наприклад, додаток для Android пишуть мовами Kotlin або Java, а для iOS – на Swift та Objective C. Такі програми відрізняються швидкою роботою та приємним інтерфейсом. Проте їхня розробка коштує вдвічі дорожче, ніж розробка гібридних рішень.

Гібридні або кросплатформні програми містять риси нативного та веб-додатків. Для розробки використовують фреймворки, засновані на HTML5/JavaScript (React Native, Ionic). Гібридні програми працюють на будь-яких платформах, але мають більше обмежень та багів.

Вибір технології залежить від того, чи потрібна вам програма iOS і Android або на перший час достатньо однієї ОС. У першому випадку варто вибрати нативний варіант. У другому – кросплатформовий.

Розглянемо порядок створення мобільного додатку.

В першу чергу потрібно сформулювати завдання, яке вирішуватиме програма, і написати User Story (як виглядає користувач, як він заходитиме у програму, з якими проблемами може зіткнутися).

На наступному етапі продумується функціонал, від якого залежить внутрішня архітектура мобільного додатку. Також важливо вирішити, як виглядатимуть екрани і які елементи та дані потрібно на них розмістити.

Після цього передається проект на розробку. Фахівець продумає архітектуру системи та вибере відповідні інструменти. На етапі backend-розробки фахівець визначається з мовою програмування, створює інтерфейс та вибирає сервер. Від цього залежатиме масштабованість та продуктивність програми. Після цього настає час frontend-розробки, тобто створення тієї частини, яку бачитиме користувач [4].

Готове рішення тестуються на різних пристроях.

### 2.2 Проєктування структури веб-сервісу

При проєктуванні системи для початку потрібно визначитись для чого вона потрібна і які послуги надаватиме, або які функції виконуватиме. Бо від цього напряму залежатиме напрямок проєктування.

В даній роботі система націлена на допомогу в процесі навчання. Відповідно користуватися системою будуть користувачі практично будь-якого віку:

- школярі (включаючи учнів молодших класів);
- студенти;
- викладачі навчальних закладів;
- будь-яка людина;
- система може використовуватись в різних компаніях, для проведення тренінгів та підвищення рівня кваліфікації працівників.

Веб-сервіс для вивчення іноземних мов включатиме в себе наступні сторінки (екрани), які необхідно розробити.

1) Домашня сторінка або головний екран відображатиме головне меню, де можна обрати особистий кабінет, навчальні матеріали, вибрати тематику навчання, скористатися словником та перейти до налаштувань.
2) Екран реєстрації/авторизації є способом ідентифікувати користувача, щоб він міг використовувати систему в своїх цілях.
3) Екран навчальних матеріалів буде відображати розділи з навчальним матеріалом.
4) У словнику можна знайти потрібне слово з перекладом.
5) Екран тестування буде з’являтися після вивчення кожної теми.
6) Особистий кабінет буде відображати інформацію про користувача з можливістю її редагування, а також прогрес, тобто накопичені бали.

### 2.3 Специфікація вимог до розроблюваного програмного засобу

Визначимо основні функціональні вимоги та побудуємо діаграму варіантів використання (рис.2.1).

У системі беруть участь 3 актори – користувач, система словника (Dictionary API), та система переводу (Translation API).

Користувач – цільовий користувач веб-сервісу, для якого передбачається додаток.

Dictionary – це стороннє АПІ, що дозволяє отримувати дані про подане слово.

Translation – це стороннє АПІ, що дозволяє переводити подане слово на слово в потрібній мові.

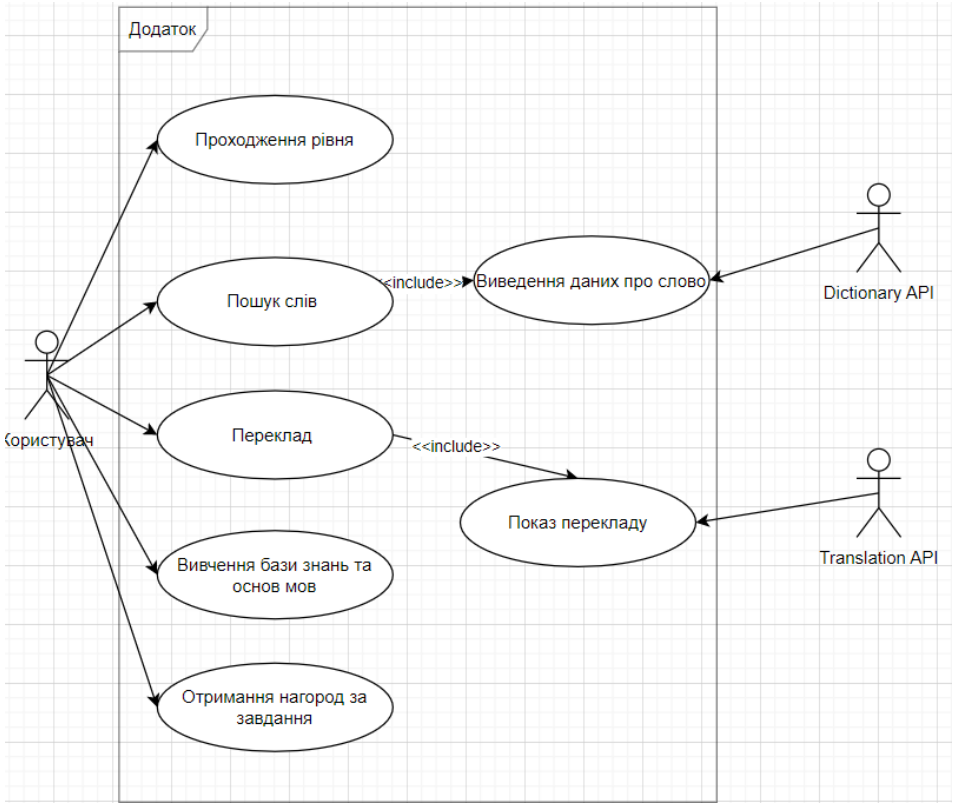


Рисунок 2.1 – Діаграма варіантів використання

Розглянемо основні прецеденти.

Прецедент – Проходження рівня.

Гарантія успіху: Рівень пройдено успішно.

Передумова: Користувач має бути в системі та мати доступ до рівнів.

Основний сценарій:

1) Користувач входить до системи. Система показує усі рівні.
2) Користувач обирає рівень. Система повертає рівень.
3) Користувач проходить рівень. Система зберігає інформацію.

Розширення:

– Користувач не зміг пройти рівень:
    – система відображає повідомлення про невдачу;
    – рівень розпочинається спочатку.

Прецедент – Пошук слова.

Гарантія успіху: Інформацію про слово знайдено.

Передумова: Користувач має бути в системі та мати доступ до функції пошуку слова.

Основний сценарій:

1) Користувач вводить слово для пошуку. Система виводить доступну інформацію.

Розширення:

– Введене слово не існує/не відомо/введено некоректно:
  – система відображає повідомлення що такого слова немає;
  – користувач вводить інше слово.
– Відсутній зв’язок з Dictionary API:
  – система відображає повідомлення про помилку, та пропонує повторити пошук.

Прецедент – Переклад.

Гарантія успіху: Слова перекладене на обрану мову

Передумова: Користувач має бути в системі та мати доступ до функції перекладу.

Основний сценарій:

1) Користувач обирає мову з якої йде переклад, та мову на яку перекладається. Система зберігає інформацію.
2) Користувач вводить слова для перекладу. Система виводить перекладені слова.

Розширення:

– Введене слово не існує/не відомо/введено некоректно:
  – система відображає повідомлення що такого слова немає;
  – користувач вводить інше слово.
– Відсутній зв’язок з Translation API:
  – система відображає повідомлення про помилку, та пропонує повторити переклад.

Прецедент – Вивчення бази знань та основ мови.

Гарантія успіху: Користувач отримав бажані знання

Передумова: Користувач має бути в системі та мати доступ до розділів знань і основ мови.

Основний сценарій:

1) Користувач обирає бажаний розділ. Система виводить інформацію.

Прецедент – Отримання нагород за завдання.

Гарантія успіху: отримання Користувачем нагороди.

Передумова: Користувач має бути в системі та завершити рівень для отримання нагороди.

Основний сценарій:

1) Користувач закінчив рівень. Система виводить нагороду.

Розширення:

– Користувач закінчив рівень з занадто великою кількістю помилок:
    – система відображає повідомлення про занадто велику кількість помилок.

Розглянемо соновні нефункціональні вимоги до системи.

Додаток повинен працювати швидко та ефективно, щоб користувачі могли отримувати необхідну інформацію вчасно.

Додаток повинен бути зручним у використанні та мати інтуїтивно зрозумілий інтерфейс.

Додаток повинен бути надійним і відповідати вимогам, описаними в цій роботі.

Додаток повинен працювати стабільно, без перебоїв та попереджувати будь які помилки, що можуть заважати роботі системи.

## 2.4 Діаграма діяльності

Для того, щоб продемонструвати які процеси відбуваються у системі використовуються діаграми діяльності.

На рис. 2.2 показана діаграма діяльності для проходження уроку.

Користувач заходить до веб-сервісу, та в залежності від того чи він авторизований чи ні може:

- вибрати мову;
- перейти до форми для входу;
- після вибору мови йде вибір тематики, а після цього, вибір уроку по цій тематиці;
- далі користувач проходить урок, якщо він зробив n помилок, він може пройти урок знову.

Якщо зроблено більше n помилок користувач може обрати:

- продовжити навчання;
- при продовженні навчання користувач повертається до вибору тематики.

Якщо користувач не авторизований, він переходить до форми для входу, далі в залежності від того чи користувач має аккаунт, він може:

- ввести дані;
- перейти до форми для реєстрації;
- після того як користувач ввів дані для входу, проходить їх перевірк;
- якщо дані вірні, користувач переходить до вибору мови, якщо дані введено невірно, він повинен ввести дані заново;
- якщо користувач перейшов до форми для реєстрації, він вводить свої дані для створення аккаунту.

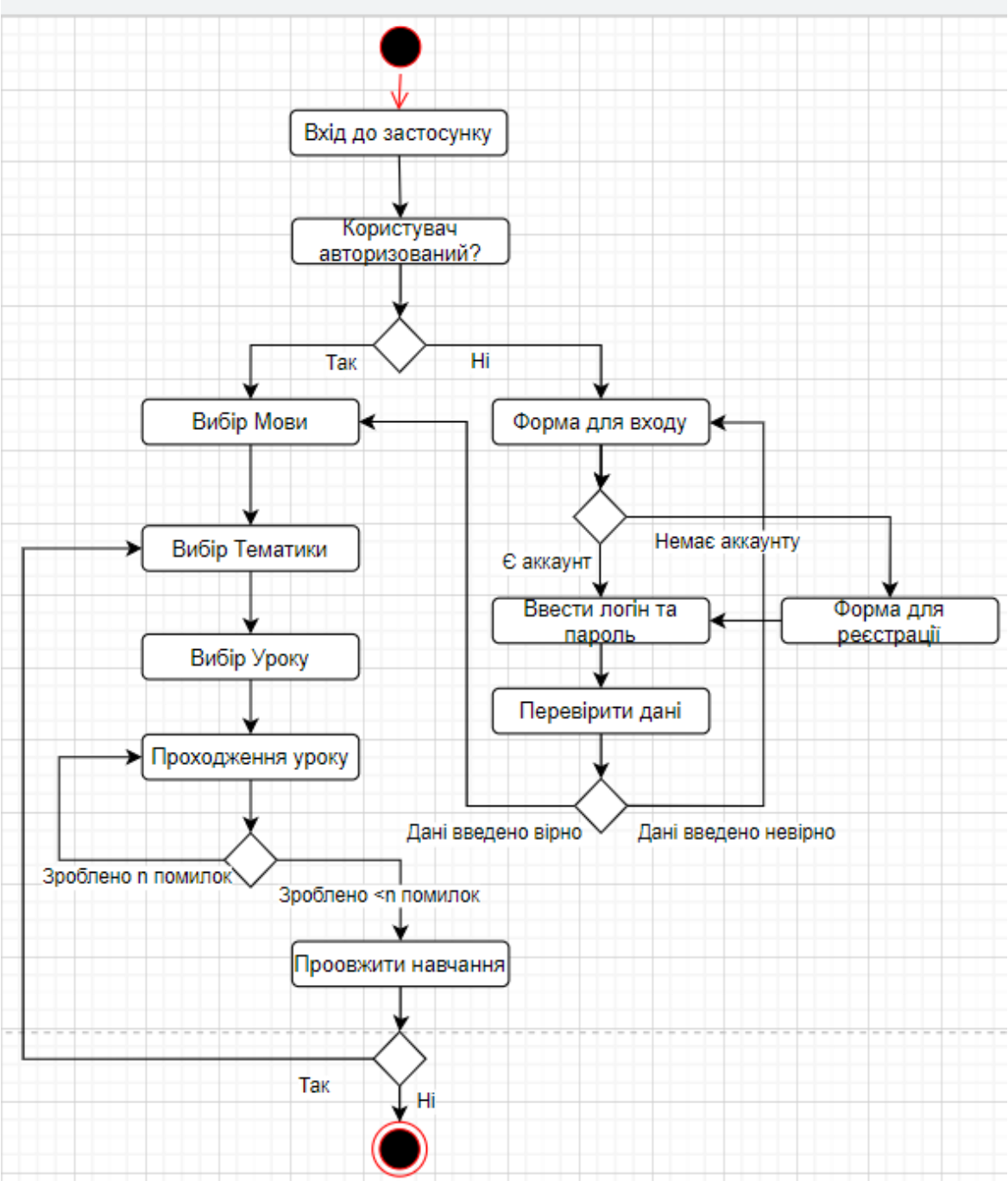


Рисунок 2.2 – Діаграма діяльності «Проходження уроку»

## 2.5 Діаграми послідовності

Щоб продемонструвати взаємодію об’єктів, використаємо діаграму послідовності. Діаграми послідовностей використовуються для уточнення діаграм прецедентів, більш детального опису логіки використання. Це

відмінний засіб документування проекту з точки зору сценаріїв використання.

Діаграми послідовностей зазвичай містять об'єкти, які взаємодіють в рамках сценарію, повідомлення, якими вони обмінюються, і результати, пов'язані з повідомленнями. Втім, часто результати позначають лише в тому випадку, якщо це не очевидно з контексту.

Розглянемо діаграму послідовності для варіанту використання «Пошук слова» (рис. 2.3).

Користувач запитує слово. Для отримання даних перекладу, синонімів та можливих замін слова будемо використовувати YandexDictionary API. Для отримання даних прикладів використання слова використовуємо DictionaryAPI.

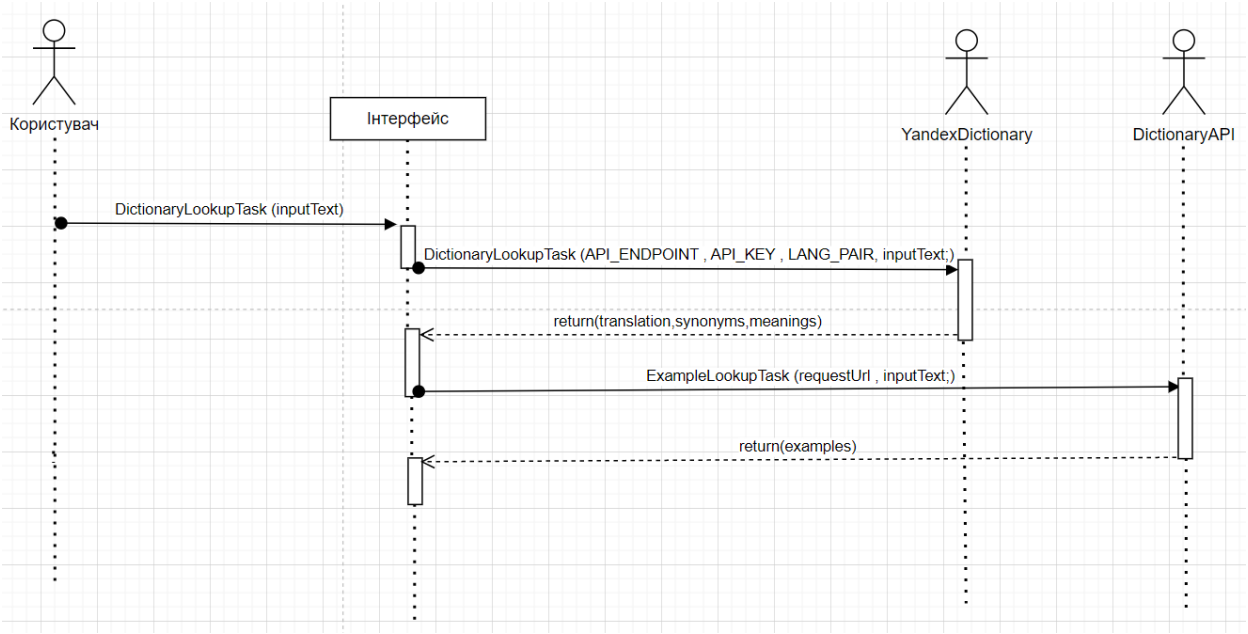


Рисунок 2.3 – Діаграма послідовності «Пошук Слова»

Діаграма послідовності для варіанту використання «Переклад» (рис.2.4): користувач запрошує переклад слова, використовуємо Deep

Translate API для перекладу, отримуємо текст введений користувачем, та обрані ним мови для перекладу.

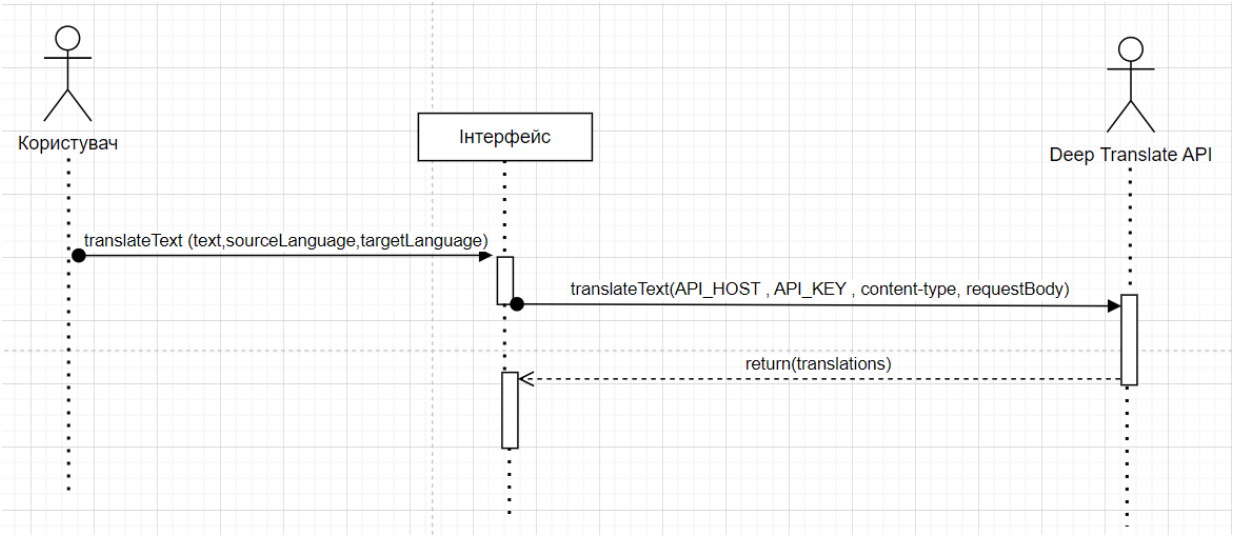


Рисунок 2.4 – Діаграма послідовності «Переклад»

### 2.6 Розробка структури бази даних

Для збереження даних використовується реляційна база даних. Вона має такі таблиці:

User – користувач системи;

Language – таблиця з мовами;

Theme – таблиця з темами;

Lesson – таблиця з уроками;

Testing – тестувальна частина уроку;

Learning – навчальна частина уроку;

Theory – навчальний матеріал.

Структура БД представлена на рис.2.5.

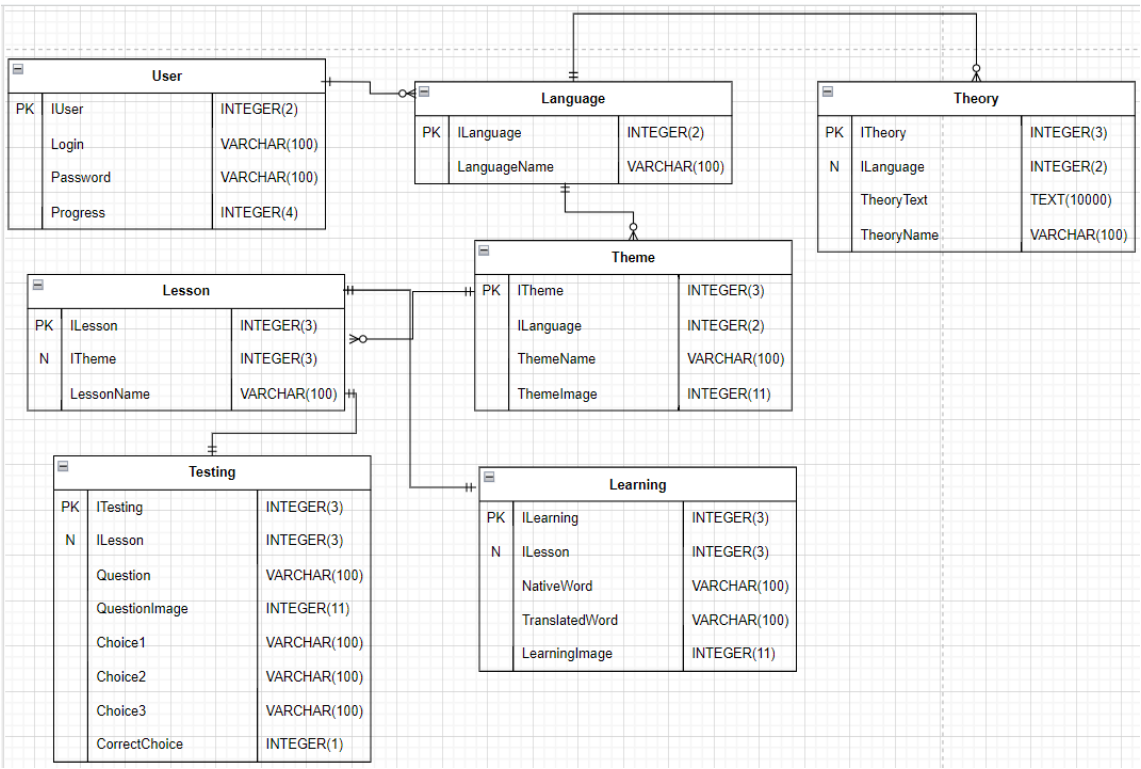


Рисунок 2.5 – Схема бази даних

Для кожної таблиці сформуємо структуру з визначенням полів.

Таблиця 2.1 – Структура даних таблиці User

| Назва поля | Тип даних | Опис | Ключ |
|---|---|---|---|
| IUser | integer (2) | Ідентифікатор користувача | PK |
| Login | varchar (100) | Логін користувача системи | |
| Password | varchar (100) | Пароль користувача системи | |
| Progress | integer (4) | Прогрес користувача системи | |

Таблиця 2.2 – Структура таблиці Language

| Назва поля | Тип даних | Опис | Ключ |
|---|---|---|---|
| ILanguage | integer (2) | Ідентифікатор мови | PK |
| LanguageName | varchar (100) | Назва мови | |

Таблиця 2.3 – Структура таблиці Theme

| Назва поля | Тип даних | Опис | Ключ |
|---|---|---|---|
| ITheme | integer (3) | Ідентифікатор теми | PK |
| ILanguage | integer (2) | Ідентифікатор мови | |
| ThemeName | varchar (100) | Назва теми | |
| ThemeImage | integer (11) | Зображення теми | |

Таблиця 2.4 – Структура таблиці Lesson

| Назва поля | Тип даних | Опис | Ключ |
|---|---|---|---|
| ILesson | integer (3) | Ідентифікатор уроку | PK |
| ITheme | integer (3) | Ідентифікатор теми | |
| LessonName | varchar (100) | Назва уроку | |

Таблиця 2.5 – Структура таблиці Testing

| Назва поля | Тип даних | Опис | Ключ |
|---|---|---|---|
| ITesting | integer (3) | Ідентифікатор частини тестування | PK |
| Question | varchar (100) | Питання | |
| QuestionImage | integer (11) | Зображення до питання | |
| Choice1 | varchar (100) | Перший варіант | |
| Choice2 | varchar (100) | Другий варіант | |
| Choice3 | varchar (100) | Третій варіант | |
| CorrectChoice | integer (1) | Який з варіантів правильний | |
| ILesson | integer (3) | Ідентифікатор уроку | |

Таблиця 2.6 – Структура таблиці Learning

| Назва поля | Тип даних | Опис | Ключ |
|---|---|---|---|
| ILearning | integer (3) | Ідентифікатор частини навчання | PK |
| NativeWord | varchar (100) | Слово в рідній мові | |
| TranslatedWord | varchar (100) | Слово в іноземні мові | |
| LearningImage | integer (11) | Зображення до слова | |
| ILesson | integer (3) | Ідентифікатор уроку | |

Таблиця 2.7 – Структура таблиці Theory

| Назва поля | Тип даних | Опис | Ключ |
|---|---|---|---|
| ITheory | integer (3) | Ідентифікатор навчального матеріалу | PK |
| TheoryText | text (10000) | Текст навчального матеріалу | |
| TheoryName | varchar (100) | Назва навчального матеріалу | |
| ILanguage | integer (2) | Ідентифікатор мови | |

Представлена база даних є нормалізованою, у ній усунуті надмірності даних, а також виявлені функціональні залежності. У разі виключення надмірності даних гарантується компактність наборів даних, уникнення зайвого дублювання даних та відсутність аномалій вставки, видалення, редагування після програмної реалізації БД.

### 2.7 Проєктування структури класів

Перейдемо до формування структури у вигляді програмних класів, їх атрибутів та методів. Для цього була створена діаграма програмних класів (рис. 2.6).

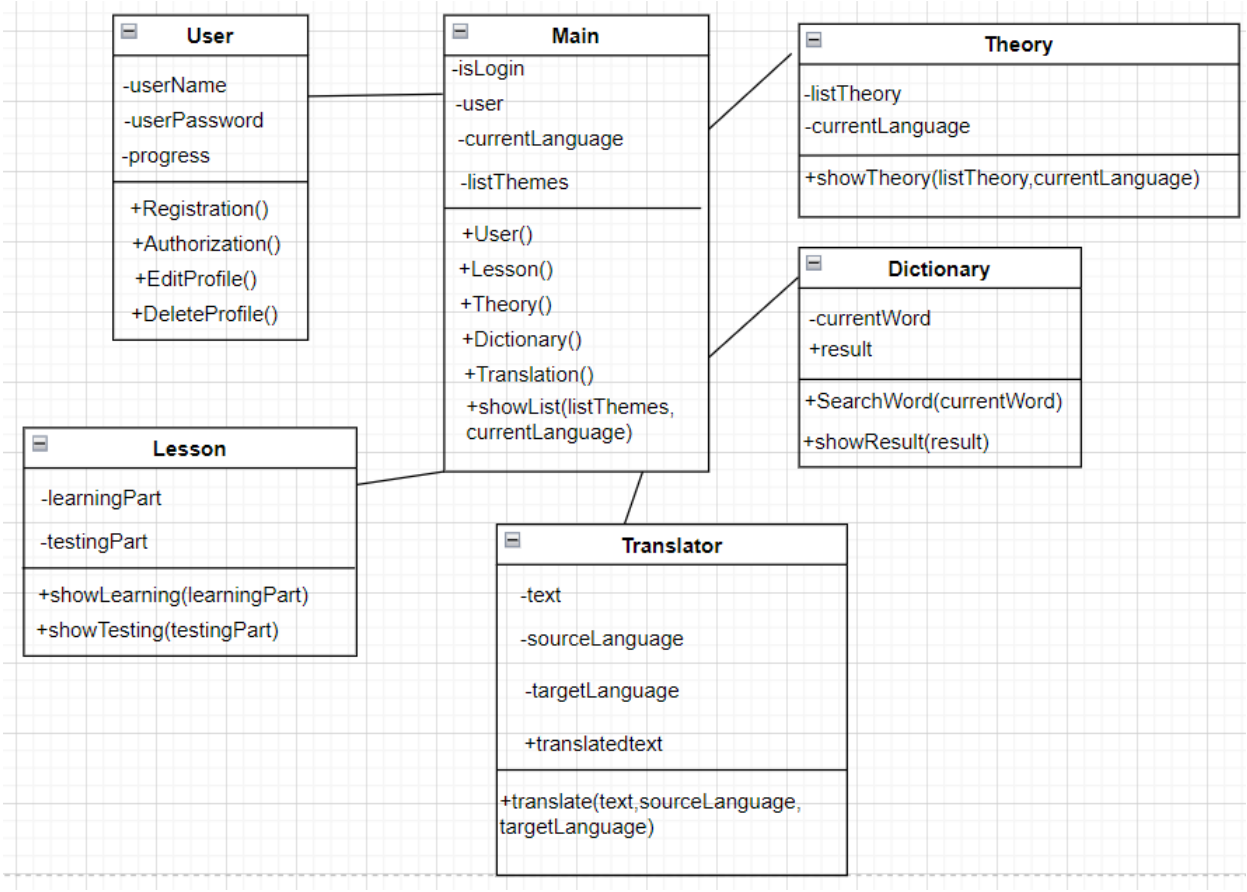


Рисунок 2.6 – Діаграма класів системи

Клас User містить дані про ім’я, пароль та прогрес користувача, та методи реєстрації, авторизації, редагування та видалення профілю.

Клас Theory має дані про навчальних матеріали та метод для показу списку навчальних матеріалів за поточною мовою.

Клас Lesson має дані навчальної частини та тестувальної частини уроків. Також методи для показу навчальної частини та роботи в тестувальній частині.

Клас Dictionary має дані про шукане слово, та виведену інформації по ньому. Також методи для пошуку слова та виведення інформації.

Клас Translator має про текст для переводу, мови переводу та метод для переводу.

Клас Main містить інформацію про логін користувача та поточну мову. Також методи для переходу на інші частини додатку, показу списку тем за поточною мовою.

Розподіл обов'язків:

Клас User:

- Атрибути: ім'я, пароль, прогрес користувача.
- Методи: реєстрація, авторизація, редагування та видалення профілю.

Клас Theory:

- Атрибути: навчальні матеріали, поточна мова.
- Методи: показ списку навчальних матеріалів за поточною мовою.

Клас Lesson:

- Атрибути: навчальна частина, тестувальна частина уроків.
- Методи: показ навчальної частини, робота в тестувальній частині.

Клас Dictionary:

- Атрибути: шукане слово, інформація по ньому.
- Методи: пошук слова, виведення інформації.

Клас Translator:

- Атрибути: текст для перекладу, мови перекладу.
- Методи: переклад тексту.

Клас Main:

- Атрибути: логін користувача, поточна мова.
- Методи: перехід на інші частини додатку, показ списку тем за поточною мовою.

Цей розподіл обов'язків допомагає організувати функціональність програми, розділити логічні блоки функцій та забезпечити зрозумілість та читабельність коду.

# 3 ПРОГРАМНА РЕАЛІЗАЦІЯ НАВЧАЛЬНОГО ВЕБ-СЕРВІСУ З ІНОЗЕМНИХ МОВ

## 3.1 Середовище розробки

Android Studio – інтегроване середовище розробки виробництва Google, за допомогою якого розробникам стають доступні інструменти для створення програм на платформі Android OS. Android Studio можна встановити на Windows, Mac та Linux. Android Studio створене з урахуванням IntelliJ IDEA.

Studio містить інструменти для розробки рішень для смартфонів і планшетів, а також нові технологічні рішення для Android TV, Android Wear, Android Auto, Glass і додаткові контекстуальні модулі [5].

Середовище Android Studio призначене як для невеликих команд розробників мобільних додатків (навіть однієї людини), так і для великих міжнародних організацій з GIT або іншими подібними системами управління версіями. Досвідчені розробники зможуть вибрати інструменти, які найбільше підходять для масштабних проектів. Рішення для Android розробляються в Android Studio за допомогою Java або C++.

В основі робочого процесу Android Studio закладено концепт безперервної інтеграції, що дозволяє відразу виявляти наявні проблеми.

Тривала перевірка коду забезпечує можливість ефективного зворотного зв'язку з розробниками. Така опція дозволяє швидше опублікувати версію мобільного додатка в Google Play App Store. Для цього є також підтримка інструментів LINT, Pro-Guard і App Signing.

За допомогою інструмента для візуалізації пам'яті розробник дізнається, коли його додаток буде використовувати занадто багато оперативної пам'яті і коли відбудеться складання сміття.

Інструменти для аналізу батареї показують, яке навантаження лягає на пристрій.

Android Studio сумісна з платформою Google App Engine для швидкої інтеграції у хмари нових API та функцій. У розробці є різні API, такі як Google Play, Android Pay і Health.

Є підтримка всіх платформ Android, починаючи з Android 1.6.

Особливості Android Studio

Нові функції відображаються з кожною новою версією Android Studio. На даний момент доступні такі функції:

Розширений редактор макетів: WYSIWYG, здатність працювати з UI компонентами за допомогою Drag-and-Drop, функція попереднього перегляду макета на декількох конфігураціях екрана.

Складання додатків, засноване на Gradle.

Різні види збірок та генерація декількох .apk файлів

Рефакторинг коду

Статичний аналізатор коду (Lint), що дозволяє знаходити проблеми продуктивності, несумісності версій та інше.

Вбудований ProGuard та утиліта для підписування додатків.

Шаблони основних макетів та компонентів Android.

Підтримка розробки додатків для Android Wear та Android TV.

Вбудована підтримка Google Cloud Platform, яка включає інтеграцію з сервісами Google Cloud Messaging і App Engine.

Android Studio 2.1 підтримує Android Preview SDK, а це означає, що розробники зможуть розпочати роботу зі створення програми для нової програмної платформи.

Нова версія Android Studio 2.1 здатна працювати з оновленим компілятором Jack, а також має покращену підтримку Java 8 та вдосконалену функцію Instant Run.

Починаючи з Platform-tools 23.1.0 для Linux винятково 64-розрядна.

В Android Studio 3.0 будуть за стандартом включені інструменти мови Kotlin, засновані на JetBrains IDE.

Недоліки Android Studio:

– досить повільна збірка проекту без додаткових інструментів;
– високі системні вимоги до ПК;
– повільна робота емуляторів.

### 3.2 Мова програмування

Одна з найпопулярніших мов для розробки додатків по під операційну систему Android є Java. Застосування Java в створенні мобільних додатків є невипадковим вибором, бо це відкрита і потужна мова. Java застосовується в розробці додаткового функціонала веб-серверів, повномасштабних корпоративних додатків, проектують додатки для різних пристроїв крім смартфонів і планшетів, і це тільки мала частина можливостей і області застосування цієї мови. Java – багатоплатформна мова, з її допомогою програмісти можуть створювати додатки, не звертаючи увагу на апаратні особливості пристрою, а з використання Android API та інших допоміжних засобів розробники здатні швидко вивчити проектування додатків для операційної системи Android [6].

Головною перевагою операційної системи Android є відкритість. Дана платформа поширюється на вільній основі і заснована на використанні відкритого коду. Завдяки цьому, розробники можуть отримати доступ до вихідного коду Android і на його основі реалізовувати нові функції програми.

За допомогою мови Java, яка відноситься до об'єктно орієнтованим мов програмування, розробники отримують доступ до великої кількості потужних бібліотек, що прискорюють написання програми. Створення призначеного для користувача графічного інтерфейсу є контрольованим подією. З його допомогою можна зібрати інтерфейс з заготовлених елементів, у вигляді текстових полів, кнопок і перемикачів змінюючи їх розмір, перетягуючи в будь-які частини екрану і додаючи підписи.

Розглянемо основні переваги Java.

Безпека: для перевірки достовірності даних використовується методи, засновані на шифруванні з відкритим ключем.

Динаміка: Java легко адаптується під мінливі умови та є більш динамічною мовою на відміну від C і C ++. Програми можуть виконувати велику кількість під час обробки інформації, яка може бути використана для перевірки й дозволу доступу до об'єктів на час виконання.

Сильна підтримка спільноти: Java має велику та активну спільноту розробників по всьому світу. Ця спільнота сприяє розробці численних бібліотек, фреймворків і інструментів з відкритим кодом, що полегшує розробникам пошук рішень, обмін знаннями та співпрацю над проектами.

Портативність: архітектурно-нейтральний і не має залежності від реалізації аспектів специфікацій – все це робить Java портативним. Компілятор в Java написаний на ANSI C з чистою переносимістю, який є підмножиною POSIX.

Інтерпретування: Java байт-код переводиться на льоту в машинні інструкції та ніде не зберігається, роблячи процес більш швидким і аналітичним, оскільки зв'язування відбувається як додаткове з невеликою вагою процесу.

Висока продуктивність: введення Just-In-Time компілятора, дозволило отримати високу продуктивність.

Багата стандартна бібліотека: Java поставляється з розширеною стандартною бібліотекою, яка надає широкий спектр попередньо створених класів і методів для типових завдань програмування. Ця бібліотека спрощує розробку, пропонуючи готові до використання компоненти для мереж, обробки файлів, підключення до бази даних, розробки інтерфейсу користувача тощо.

Обробка винятків: Java має надійний механізм обробки винятків, який дозволяє розробникам акуратно обробляти та відновлювати помилки

виконання. Перехоплюючи та обробляючи винятки, розробники можуть створювати більш надійні та відмовостійкі програми.

Розглянемо основні недоліки Java.

Накладні витрати на продуктивність: порівняно з мовами нижчого рівня, такими як C або C++, програми Java можуть мати більше споживання пам’яті та нижчу швидкість виконання. Однак з прогресом технології розрив у продуктивності значно зменшився.

Багатослівність: Java може бути більш багатослівною, ніж інші мови програмування, вимагаючи більше рядків коду для виконання певних завдань. Однак ця багатослівність часто сприяє читабельності та зручності підтримки коду.

Обмежений доступ до апаратного забезпечення: Java була розроблена з урахуванням безпеки, і, як наслідок, вона має обмежений прямий доступ до апаратних компонентів низького рівня. Хоча це забезпечує більш безпечне середовище, це може бути недоліком для певних програм, які вимагають детального контролю над апаратними ресурсами.

Крива навчання: Java має великий набір функцій, що може ускладнити її опанування початківцям. У цій мові є багато концепцій і API, які потрібно вивчити, включаючи принципи об’єктно-орієнтованого програмування і бібліотеки Java.

Відсутність сучасних мовних можливостей: Java розвивалася протягом багатьох років, але її критикували за те, що вона повільніше засвоює деякі сучасні мовні функції та парадигми програмування. Однак останні версії Java представили кілька нових функцій і вдосконалень для вирішення цієї проблеми.

Незважаючи на недоліки, надійність, портативність і розгалуженість Java зробили її популярним вибором для розробки широкого спектру програм, включаючи корпоративне програмне забезпечення, веб-програми, мобільні програми тощо.

### 3.3 Система керування базами даних SQLite

Sqlite – це вбудована реляційна база даних, з відкритим вихідним кодом. Тобто вона не використовує модель роботи бази даних клієнт-сервер і не є окремим працюючим процесом. Наприклад, у базі даних Mysql движок Sqlite стає ніби частиною застосунку. При такому підході база даних Sqlite являє собою звичайний текстовий файл, який можна розташувати у зручному місці.

Переваги SQLite:

- файлова структура – вся база даних складається з одного файлу, тому її легко переносити на різні машини;
- відсутність необхідності налаштування сервера СУБД;
- повністю вільна ліцензія;
- кросплатформеність;
- висока швидкість простих операцій вибірки даних;
- підтримка транзакцій, тригерів, уявлень (views), вкладених запитів;
- безпека. БД зберігається в одному файлі, права доступу до якого можна контролювати стандартними засобами ОС;
- дуже економічна, щодо ресурсів, архітектура.

Недоліки SQLite:

1) Відсутність системи користувачів – більші СУБД включають до складу системи управління правами доступу користувачів. Зазвичай застосування цієї функції негаразд критично, оскільки ця СУБД використовується у невеликих додатках.
2) Відсутність можливості збільшення продуктивності – знову, виходячи з проектування, досить складно дістати щось продуктивніше з цієї СУБД.
3) Це досить швидкий та зручний спосіб збереження та опрацювання даних у додатку, котрий не вимагає окремого серверу БД, не

навантажує мобільний пристрій і дозволяє в короткі терміни обробляти та записувати інформацію в БД [7].

### 3.4 Програмна реалізація функцій

#### 3.4.1 Реалізація функцій «Проходження уроку»

Для реалізації функцій проходження уроку створено класи LearningActivity, TestingActivity, ResultActivity їх класи даних та методи, описані далі:

```
    private void loadLearningData() {
        learningDataList =
dbHelper.getLearningDataByLesson(lessonId);

        if (!learningDataList.isEmpty()) {
          currentIndex = 0;
          LearningData learningData =
learningDataList.get(currentIndex);
          displayLearningData(learningData);

          if (learningDataList.size() == 1) {
            startQuestionActivity();
          }
        }
      }
```

Використовується для отримання інформацію з бази даних шляхом заповнення листу класа даних LearningData.

```
    public List<LearningData> getLearningDataByLesson(int
lessonId) {
        List<LearningData> learningDataList = new
ArrayList<>();
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery("SELECT * FROM learning
WHERE lesson_id = ?", new String[]{String.valueOf(lessonId)});
        if (cursor.moveToFirst()) {
          do {
            int learningId =
cursor.getInt(cursor.getColumnIndex("learning_id"));
            String nativeWord =
cursor.getString(cursor.getColumnIndex("native_word"));
            int imageResId =
cursor.getInt(cursor.getColumnIndex("image_res_id"));
```

```
            String translation =
cursor.getString(cursor.getColumnIndex("translation"));
            String transcription =
cursor.getString(cursor.getColumnIndex("transcription"));
            LearningData learningData = new
LearningData(learningId, nativeWord, imageResId, translation,
transcription);
            learningDataList.add(learningData);
          } while (cursor.moveToNext());
        }
        cursor.close();
        db.close();
        return learningDataList;
      }
```

Метод в DBHelper для заповнення даних:

```
      private void displayLearningData(LearningData
learningData) {
        wordTextView.setText(learningData.getNativeWord());

translationTextView.setText(learningData.getTranslation());

transcriptionTextView.setText(learningData.getTranscription());

lessonImageView.setImageResource(learningData.getImageResId());
      }
```

Використовується для відображення отриманих даних:

```
     private void showNextLearningData() {
        currentIndex++;
        if (currentIndex < learningDataList.size()) {
          LearningData learningData =
learningDataList.get(currentIndex);
          displayLearningData(learningData);
        } else {
          startQuestionActivity();
        }
      }
```

Цей метод використовується кнопкою для переходу на наступну частину даних:

```
      private void startQuestionActivity() {
        Intent intent = new Intent(this,
QuestionActivity.class);
        intent.putExtra("lessonId", lessonId);
        startActivity(intent);
```

```
        finish();
    }
```

Цей метод використовується для початку QuestionActivity після того як всі дані LearningActivity були показані:

```
    private void loadQuestionData() {
        if (currentQuestionIndex < testingDataList.size()) {
          TestingData testingData =
testingDataList.get(currentQuestionIndex);
          questionTextView.setText(testingData.getQuestion());
questionImageView.setImageResource(testingData.getImageResId());
          choice1Button.setText(testingData.getChoice1());
          choice2Button.setText(testingData.getChoice2());
          choice3Button.setText(testingData.getChoice3());
          resetButtonColors();
          choice1Button.setOnClickListener(v ->
handleChoiceButtonClick(1, choice1Button));
          choice2Button.setOnClickListener(v ->
handleChoiceButtonClick(2, choice2Button));
          choice3Button.setOnClickListener(v ->
handleChoiceButtonClick(3, choice3Button));
        } else {
          navigateToResult();
        }
    }
```

Використовується для отримання інформацію з бази даних шляхом заповнення листу класа даних TestingData:

```
    private void handleChoiceButtonClick(int choice, Button
selectedButton) {
        TestingData testingData =
testingDataList.get(currentQuestionIndex);
        int correctChoice = testingData.getCorrectChoice();
        if (choice != correctChoice) {
          numMistakes++;
        }

        if (choice == correctChoice) {
          selectedButton.setBackgroundColor(Color.GREEN);
          score++;
        } else {
          selectedButton.setBackgroundColor(Color.RED);
          if (correctChoice == 1) {
            choice1Button.setBackgroundColor(Color.GREEN);
          } else if (correctChoice == 2) {
            choice2Button.setBackgroundColor(Color.GREEN);
          } else if (correctChoice == 3) {
            choice3Button.setBackgroundColor(Color.GREEN);
          }
```

```
        }
        choice1Button.setEnabled(false);
        choice2Button.setEnabled(false);
        choice3Button.setEnabled(false);
        new Handler().postDelayed(() -> {
          if (currentQuestionIndex < testingDataList.size() -
1) {        choice1Button.setEnabled(true);
            choice2Button.setEnabled(true);
            choice3Button.setEnabled(true);
            currentQuestionIndex++;
            loadQuestionData();
          } else {
            if (numMistakes > 2) {
              showTooManyMistakesMessage();
            } else {
              navigateToResult();
            }
          }
        }, 1500);
      }
```

Цей метод використовується для обробки вибору варіанту відповіді:

```
      private void showTooManyMistakesMessage() {
        Toast.makeText(this, "Зробленено занадто багато
помилок, спробуй знову.", Toast.LENGTH_SHORT).show();
        score = 0;
        numMistakes = 0;
        recreate();
      }
```

Цей метод використовується якщо зроблено забагато помилок:

```
      private void resetButtonColors() {
        choice1Button.setBackgroundColor(Color.LTGRAY);
        choice2Button.setBackgroundColor(Color.LTGRAY);
        choice3Button.setBackgroundColor(Color.LTGRAY);
      }
```

### 3.4.2 Реалізація функції «Пошук слова»

Для реалізації функцій пошуку слова створено клас DictionaryActivity та методи, описані далі:

```
    private class DictionaryLookupTask extends
AsyncTask<String, Void, String>
```

Використовує YandexDictionary API для отримання даних перекладу, синонімів та можливих замін слова. Використовує DictionaryAPI для отримання даних прикладів використання слова.

### 3.4.3 Реалізація функції «Переклад»

Для реалізації функцій перекладу створено клас TranslatorActivity та методи, описані далі:

```
private void translateText(String text, String sourceLanguage, String targetLanguage)
```

Використовує Deep Translate API для перекладу, отримує текст введений користувачем, та обрані ним мови для перекладу.

### 3.4.4 Реалізація функції «Читання матеріалу»

Для реалізації функцій читання матеріалів створено клас MaterialsActivity, його адаптер, клас даних та методи, описані далі:

```
int languageId = 1;
    if ("Polish".equals(selectedLanguage)) {
      languageId = 2;
    }
```

Отримуємо поточну мову, для якої виводити матеріали:

```
   materials = dbHelper.getAllMaterials(languageId);
```

Метод для отримання даних з бази даних:

```
        materialAdapter = new MaterialAdapter(this, materials);
        materialsListView.setAdapter(materialAdapter);
```

Підключаємо адаптер для виводу даних.

### 3.4.5 Реалізація функції «Отримання нагород»

Відображаємо інформацію про досягнення:

```
    int score = getIntent().getIntExtra("Score", 0);
        SharedPreferences preferences =
getSharedPreferences("MyPrefs", MODE_PRIVATE);
        int userId = preferences.getInt("USER_ID", 0);
        int currentProgress = dbHelper.getUserProgress(userId);
    //Отримуємо інформацію потрібну для відображення та
оновлення даних
        int newProgress = currentProgress + score;
        dbHelper.updateUserProgress(userId, newProgress);
    //Оновлюємо інформацію
        int maxProgress = 1000;
        double progressPercentage = (newProgress / (double)
maxProgress) * 100;
       progressBar.setProgress((int) progressPercentage);
    progressTextView.setText(getString(R.string.result_progress
, newProgress));
        scoreTextView.setText(getString(R.string.result_score,
score));
        if (newProgress >= 1000) {
          // TODO:
        } else if (newProgress >= 500) {
          // TODO:
        } else if (newProgress >= 100) {
          // TODO:
        } else {
          // TODO:
        }
```

### 3.5 Реалізація системи

Додаток складається з декількох екранів:

– головний екран;

– реєстрація;

– авторизація;

– вибір мови;

– вибір уроку;

– навчання;

– тестування;

– профіль;

– словник.

Кожному з екранів відведена конкретна задача. При відкриванні веб-сервісу з навчання мов побачимо головний екран. Екран реєстрації користувача є початковим, після його проходження, або проходження екрану авторизації користувачеві відкривається доступ до іншого функціоналу.

Вибрати мову та змінити її в процесі навчання можна за допомогою відповідного вибору елементів. В системі можна вибрати одну з двох мов: англійську та польську.

Також є тематика уроків, вибравши яку починається навчання.

Після проходження навчання буде тестування по вивченій темі.

Якщо перейти до навчання без вибору тематики, відобразяться теми для навчання: основи граматики, читання, тощо.

Також є пошук слова, яке необхідно перекласти у словнику.

Наведемо головний екран та екран реєстрації та авторізації на рис.3.1 та рис.3.2 відповідно.

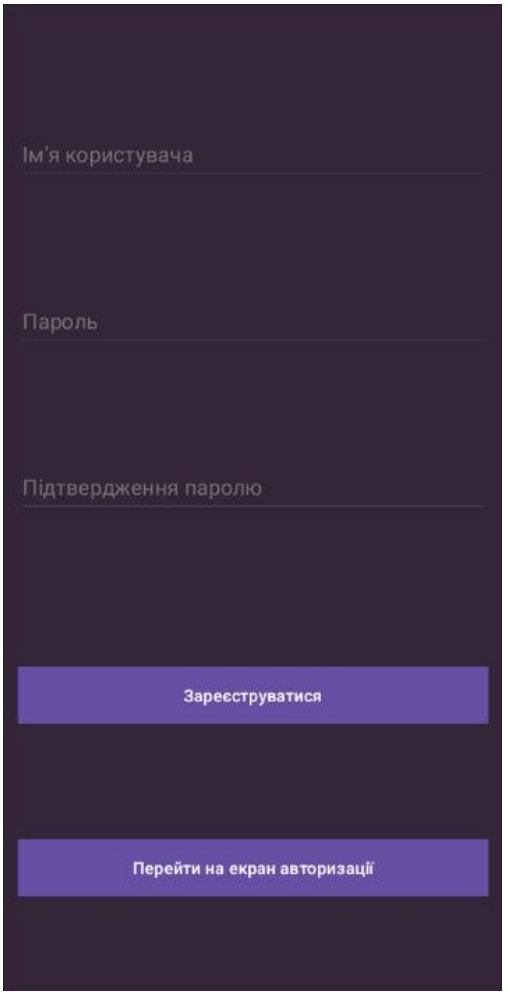


Рисунок 3.1 – Головний екран програми

Далі є можливість зареєструватися або авторизуватися, якщо вже пройшла реєстрація. Під час реєстрації користувачу потрібно заповнити необхідні поля. Перше поле – ім'я або логін, друге поле – пароль, третє поле – пароль.

При авторизації користувач вводить свій логін та пароль.

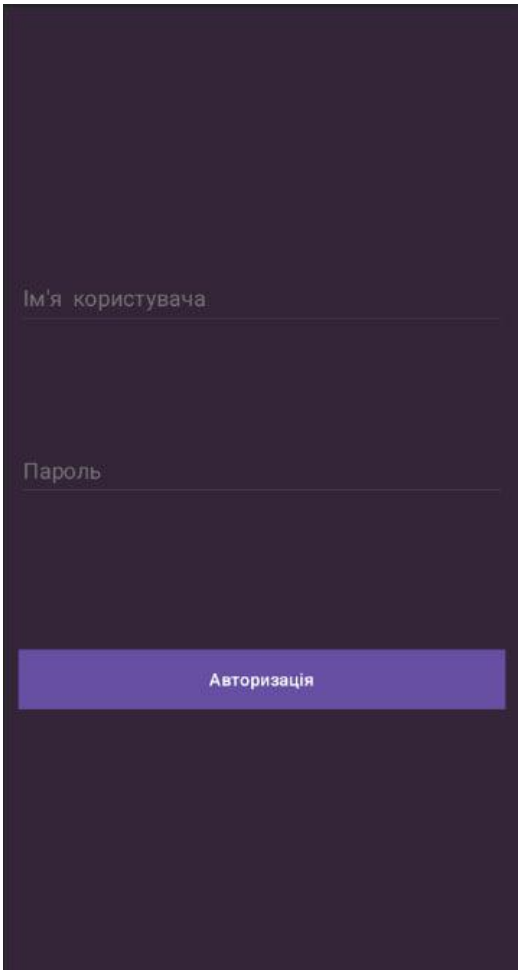


Рисунок 3.2 – Екран «Авторизація»

На рис. 3.3 показано екран мов. В ньому користувачу обирає мову навчання. На даний час у за стосунку є можливість вибору однієї мови з двох: англійської та польської. Після вибори мови значок з відповідним прапором буде відображатися зверху у головному меню. Натиснувши на прапор, завжди можна обрати іншу мову.


Рисунок 3.3 – Вибір мови

На рис. 3.4 показано головне меню після авторизації. В ньому користувач може:

- вибрати тему навчання;
- змінити мову;
- перейти на сторінку профілю;
- перейти до вивчення навчального матеріалу;
- увійти в налаштування;
- скористатися словником.


Рисунок 3.4 – Головне меню веб-сервісу

На рис. 3.5 показано список уроків по вибраній тематиці. Тут користувач може обрати цікавлячий його урок.

Якщо користувач не обирає тематику, а хоче перейти до всіх матеріалів у відповідному пункті меню, йому відобразиться список основних правил з граматики, читання та інші.

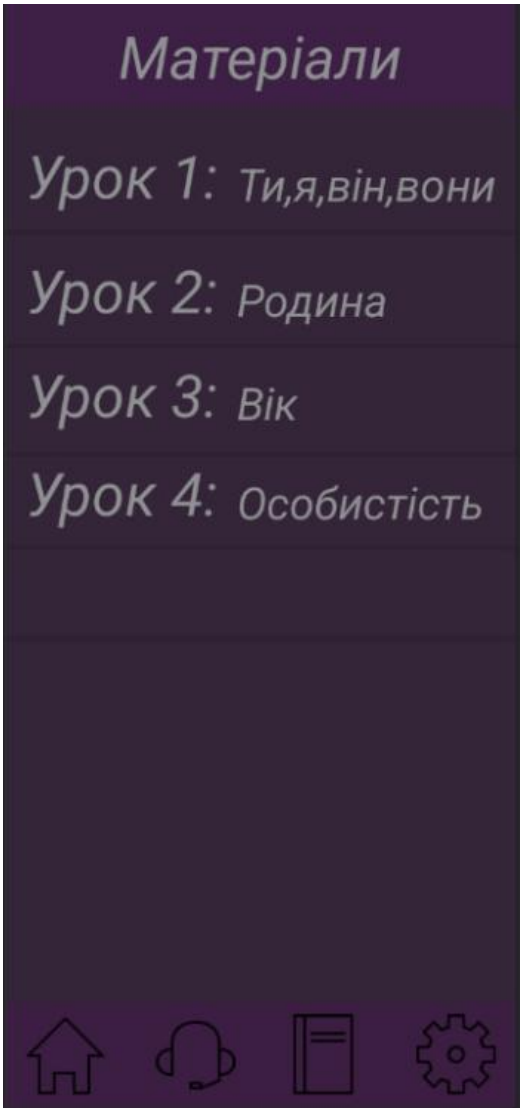


Рисунок 3.5 – Вибір уроку

На рис. 3.6 показано навчальну частину уроку. Показуються всі слова, що відповідають обраній тематиці з можливістю переключати слова тільки вперед.

Відображується слово на українській мові, виводиться відповідне зображення, це виводиться переклад слова на іноземній мові з транскрипцією.


Рисунок 3.6 – Проходження навчання

На рис. 3.7 показано тестувальну частину уроку. Після вивчення матеріалу уроку обов’язково треба пройти тестування для перевірки засвоєного матеріалу. В тестовому завданні використовується слово на українській мові, відповідне зображення та 3 варіанти відповідей.


Рисунок 3.7 – Проходження тестування

На рис. 3.8 показана сторінка словника. Саме в словнику користувач може ввести цікавляче його слово та отримати інформацію про нього: переклад слова, його значення та приклад використання.

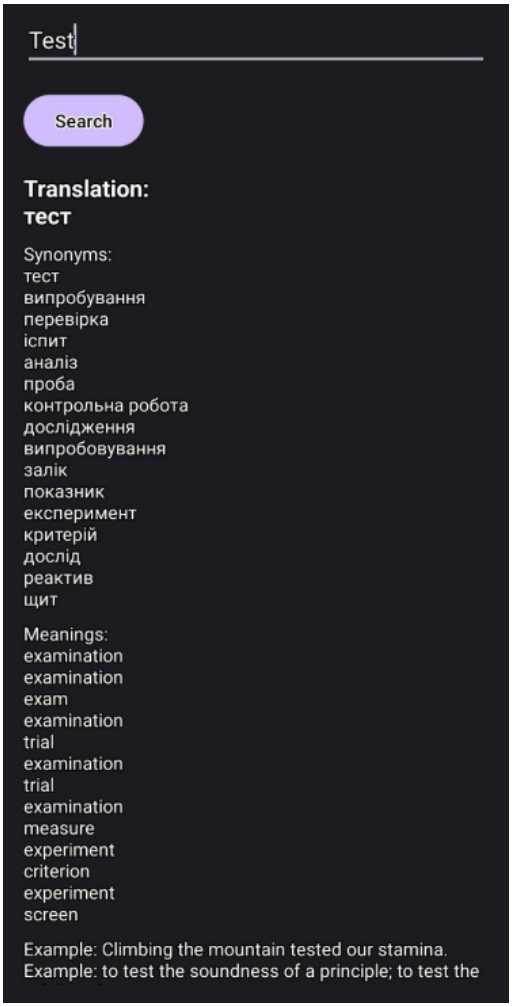


Рисунок 3.8 – Словник

На рис. 3.9 наведено список навчальних матеріалів для користувача. В цій розділ розробник може додавати основні граматичні, фонетичні правила та будь-який навчальний матеріал.

Матеріал подається невеликими порціями, після вивчення обов’язково складається тест для визначення рівня засвоєння матеріалу. Якщо тест пройдено, дається винагорода.

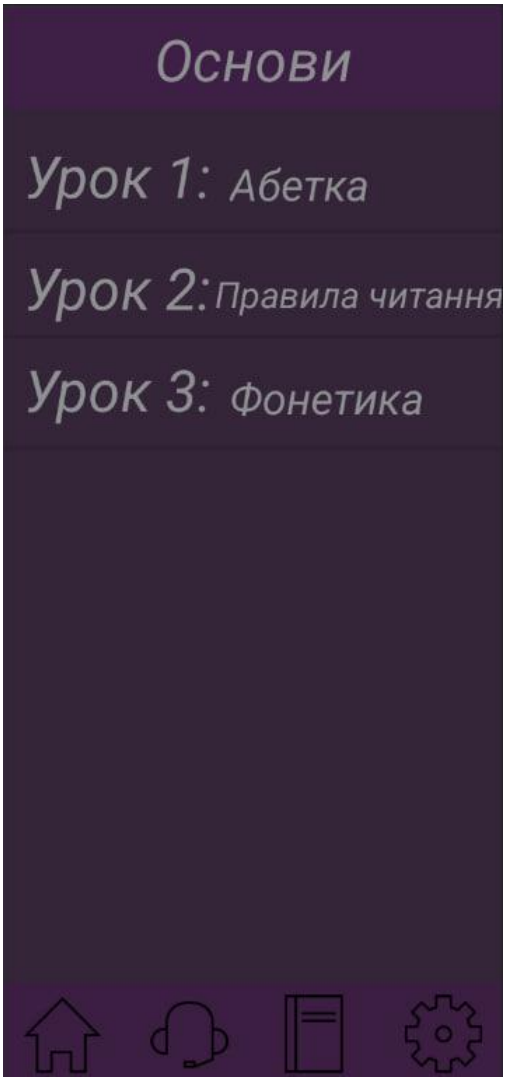


Рисунок 3.9 – Список навчальних матеріалів

На рис. 3.10 показано екран профілю користувача.

В цьому розділі буде виводитися прогрес користувача після вивчення тематики и проходження тестів. Якщо потрібна кількість балів не буде набрана, необхідно повторити навчання і знову пройти тестування. До тих пір, доки вивчення теми не буде задовольняти прийнятим показникам, навчання потрібно буде продовжувати.

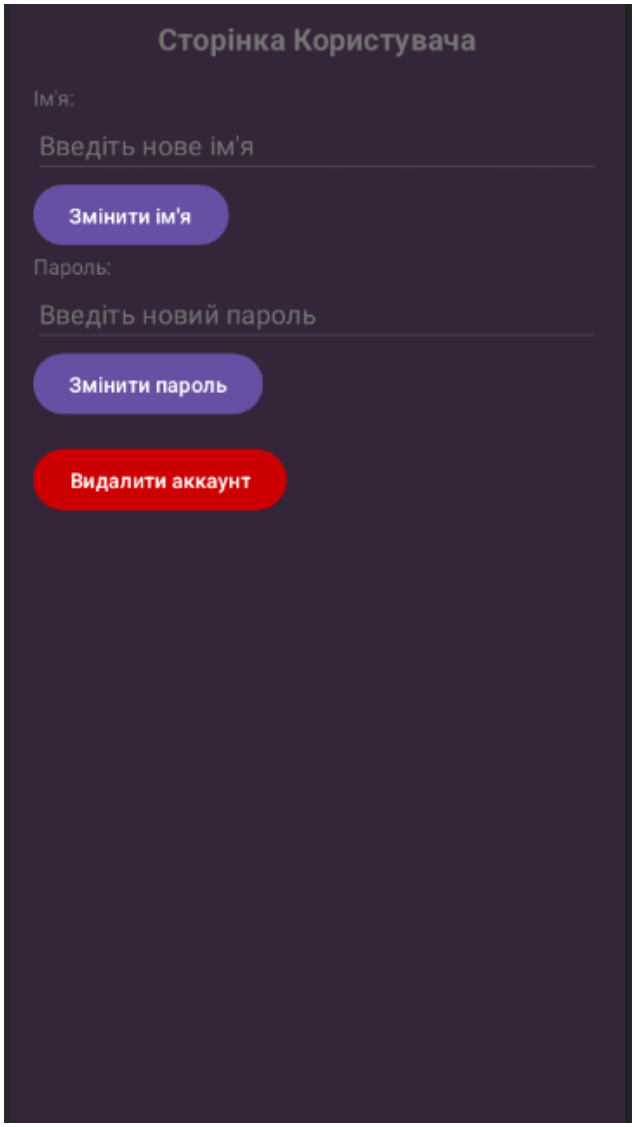


Рисунок 3.10 – Екран користувача

Розроблений веб-сервіс був протестований на ОС Android 13, в БД було внесено матеріал по тематикам, внесені питання для тестування.

# ВИСНОВКИ

Кваліфікаційна робота була присвячена розробці програмного забезпечення для вивчення іноземної мови. Було вирішено створювати систему як додаток для мобільного пристрою.

Був проведений аналіз предметної області та пошук і вивчення існуючих аналогів. Далі було визначено основний функціонал системи у вигляді діаграми варіантів використання.

Основний функціонал програмних модулів був реалізований на мові програмування Java в середі розробки Android Studio разом з додатковими бібліотеками. Наступним етапом було її тестування, де були складені тестові випадки.

Також було додано посібник користувача для більш детального пояснення основних кроків користування застосунком.

Розроблена система не є кінцевим результатом та може бути розширена за допомогою нового функціоналу.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Duolingo [Електронний ресурс] – Режим доступу: URL: https://play.google.com/store/apps/details?id=com.duolingo&hl=ru
2. Busuu [Електронний ресурс] – Режим доступу: URL: https://play.google.com/store/apps/details?id=com.busuu.android.enc&hl=ru
3. FunEasyLearn[Електронний ресурс] – Режим доступу: URL: https://play.google.com/store/apps/details?id=com.funeasylearn.languages&hl=ru&pli=1
4. How to Create a Language Learning [Електронний ресурс]. – Режим доступу:URL:https://belitsoft.com/custom-elearning-development/how-develop-language-learning-app-duolingo
5. Android Studio: переваги та особливості [Електронний ресурс]. – Режим доступу: URL: https://qagroup.com.ua/publications/android-studio-perevagy-ta-osoblyvosti/
6. Переваги мови програмування Java [Електронний ресурс]. – Режим доступу: URL: https://qagroup.com.ua/publications/benefits-of-java/
7. SQlite стоит ли использовать в web-разработке? [Електронний ресурс]. – Режим доступу: URL: https://blog-programmista.ru/post/36-sqlite-stoit-li-ispolzovat-v-web-razrabotke.html

# ДОДАТОК А

## Лістинг програми

WelcomeActivity

```
package com.example.diplom;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.os.Handler;
import androidx.appcompat.app.AppCompatActivity;
public class WelcomeActivity extends AppCompatActivity {
  private static final long DELAY_TIME_MS = 2000;
  private DBHelper dbHelper;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_welcome);
    SharedPreferences preferences =
getSharedPreferences("MyPrefs", MODE_PRIVATE);
    boolean isFirstRun = preferences.getBoolean("isFirstRun",
true);
    if (isFirstRun) {
      SharedPreferences.Editor editor = preferences.edit();
      editor.putBoolean("isFirstRun", false);
      editor.apply();
      dbHelper = new DBHelper(this);
      dbHelper.copyDatabase();
      new Handler().postDelayed(() -> {
        startActivity(new Intent(WelcomeActivity.this,
RegistrationActivity.class));
        finish();
      }, DELAY_TIME_MS);
    } else {
      startActivity(new Intent(WelcomeActivity.this,
RegistrationActivity.class));
      finish();
    }
  }
  @Override
  protected void onDestroy() {
    super.onDestroy();
    if (dbHelper != null) {
      dbHelper.close();
    }
  }
}
```

DBHelper

```
package com.example.diplom;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;
import java.io.FileOutputStream;
import java.io.IOException;
```

```
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.List;
public class DBHelper extends SQLiteOpenHelper {
public static final String DB_NAME = "Diplom.db";
private static final int DB_VERSION = 1;
private final Context context;
private static final String TABLE_MATERIALS = "materials";
  private    static    final    String    COLUMN_MATERIAL_ID    =
"material_id";
  public    static    final    String    COLUMN_MATERIAL_TEXT    =
"material_text";
  private    static    final    String    COLUMN_MATERIAL_NAME    =
"material_name";
  private    static    final    String    COLUMN_LANGUAGE_ID    =
"language_id";
 public DBHelper(Context context) {
    super(context, DB_NAME, null, DB_VERSION);
    this.context = context;
  }
  @Override
  public void onCreate(SQLiteDatabase db) {
  }
  @Override
  public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {
  }
  public void copyDatabase() {
    try {
      InputStream                  inputStream                =
context.getAssets().open("databases/" + DB_NAME);
      String                    outFileName                   =
context.getDatabasePath(DB_NAME).getPath();
      OutputStream          outputStream          =          new
FileOutputStream(outFileName);
      byte[] buffer = new byte[1024];
      int length;
      while ((length = inputStream.read(buffer)) > 0) {
        outputStream.write(buffer, 0, length);
      }
      outputStream.flush();
      outputStream.close();
      inputStream.close();
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
  public long insertData(String username, String password) {
    SQLiteDatabase MyDB = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("username", username);
    contentValues.put("password", password);
    long userId = MyDB.insert("users", null, contentValues);
    MyDB.close();
    return userId;
  }
  public Boolean checkusername(String username) {
    SQLiteDatabase MyDB = this.getWritableDatabase();
```

```
    Cursor cursor = MyDB.rawQuery("Select * from users where
username = ?", new String[]{username});
    if (cursor.getCount() > 0)
      return true;
    else
      return false;
  }
  public Boolean checkusernamepassword(String username, String
password){
    SQLiteDatabase MyDB = this.getWritableDatabase();
    Cursor cursor = MyDB.rawQuery("Select * from users where
username = ? and password = ?", new String[]
{username,password});
    if(cursor.getCount()>0)
      return true;
    else
      return false;
  }
  public int getCurrentUserId(String username) {
    SQLiteDatabase MyDB = this.getReadableDatabase();
    int userId = 0;
    Cursor cursor = MyDB.rawQuery("SELECT user_id FROM users
WHERE username = ?", new String[]{username});
    if (cursor.moveToFirst()) {
      userId = cursor.getInt(cursor.getColumnIndex("user_id"));
    }
    cursor.close();
    MyDB.close();
    Log.d("DBHelper", "Current UserData ID: " + userId);
    return userId;
  }
  public UserData getUserById(int userId) {
    SQLiteDatabase db = this.getReadableDatabase();
    String[] columns = {"user_id", "username", "password"};
    String selection = "user_id = ?";
    String[] selectionArgs = {String.valueOf(userId)};
    Cursor cursor = db.query("users", columns, selection,
selectionArgs, null, null, null);
    UserData userData = null;
    if (cursor.moveToFirst()) {
      int userIdFromDB =
cursor.getInt(cursor.getColumnIndex("user_id"));
      String username =
cursor.getString(cursor.getColumnIndex("username"));
      String password =
cursor.getString(cursor.getColumnIndex("password"));
      userData = new UserData(userIdFromDB, username, password);
    }
    cursor.close();
    db.close();
    return userData;
  }
  public boolean updateUsername(int userId, String newUsername)
{
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put("username", newUsername);
    int rowsAffected = db.update("users", values, "user_id=?",
new String[]{String.valueOf(userId)});
```

```
    db.close();
    return rowsAffected > 0;
  }
  public boolean updatePassword(int userId, String newPassword)
{
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put("password", newPassword);
    int rowsAffected = db.update("users", values, "user_id=?",
new String[]{String.valueOf(userId)});
    db.close();
    return rowsAffected > 0;
  }
  public boolean deleteUser(int userId) {
    SQLiteDatabase db = this.getWritableDatabase();
    int rowsAffected = db.delete("users", "user_id=?", new
String[]{String.valueOf(userId)});
    db.close();
    return rowsAffected > 0;
  }
  public List<MyThemeData> getAllThemes(int languageId) {
    List<MyThemeData> themeList = new ArrayList<>();

    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cursor = db.rawQuery("SELECT theme_id, theme_name,
theme_image FROM themes WHERE language_id = ?", new
String[]{String.valueOf(languageId)});
    if (cursor.moveToFirst()) {
      do {
        int themeId =
cursor.getInt(cursor.getColumnIndex("theme_id"));
        String themeName =
cursor.getString(cursor.getColumnIndex("theme_name"));
        int themeImage =
cursor.getInt(cursor.getColumnIndex("theme_image"));
        Log.d("Theme Image Integer",
String.valueOf(themeImage));

        MyThemeData themeData = new MyThemeData(themeId,
themeName, themeImage);
        themeList.add(themeData);
      } while (cursor.moveToNext());
    }
    cursor.close();
    db.close();
    return themeList;
  }
  public List<LessonData> getLessonsByThemeId(int themeId) {
    List<LessonData> lessonList = new ArrayList<>();
    SQLiteDatabase db = this.getReadableDatabase();
    String query = "SELECT * FROM lessons WHERE theme_id = " +
themeId;
    Cursor cursor = db.rawQuery(query, null);
    if (cursor.moveToFirst()) {
      do {
        int lessonId =
cursor.getInt(cursor.getColumnIndex("lesson_id"));
        String lessonName =
cursor.getString(cursor.getColumnIndex("lesson_name"));
```

```
        int                    lessonThemeId                    =
cursor.getInt(cursor.getColumnIndex("theme_id"));
        LessonData lesson = new LessonData(lessonId, lessonName,
lessonThemeId);
        lessonList.add(lesson);
      } while (cursor.moveToNext());
    }
    cursor.close();
    Log.d("LessonsActivity", "Lessons for Theme ID " + themeId +
": " + lessonList);
    return lessonList;
  }
  public     List<LearningData>     getLearningDataByLesson(int
lessonId) {
    List<LearningData> learningDataList = new ArrayList<>();

    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cursor = db.rawQuery("SELECT * FROM learning WHERE
lesson_id = ?", new String[]{String.valueOf(lessonId)});
    if (cursor.moveToFirst()) {
      do {
        int                    learningId                    =
cursor.getInt(cursor.getColumnIndex("learning_id"));
        String                    nativeWord                 =
cursor.getString(cursor.getColumnIndex("native_word"));
        int                    imageResId                    =
cursor.getInt(cursor.getColumnIndex("image_res_id"));
        String                    translation                =
cursor.getString(cursor.getColumnIndex("translation"));
        String                    transcription              =
cursor.getString(cursor.getColumnIndex("transcription"));

        LearningData learningData = new LearningData(learningId,
nativeWord, imageResId, translation, transcription);
        learningDataList.add(learningData);
      } while (cursor.moveToNext());
    }
    cursor.close();
    db.close();

    return learningDataList;
  }
  public List<TestingData> getQuestionDataByLesson(int lessonId)
{
    List<TestingData> testingDataList = new ArrayList<>();
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cursor = db.rawQuery("SELECT * FROM questions WHERE
lesson_id = ?", new String[]{String.valueOf(lessonId)});
    if (cursor.moveToFirst()) {
      do {
        int                    questionId                    =
cursor.getInt(cursor.getColumnIndex("question_id"));
        String                    question                   =
cursor.getString(cursor.getColumnIndex("question"));
        int                    imageResId                    =
cursor.getInt(cursor.getColumnIndex("image_res_id"));
        String                    choice1                    =
cursor.getString(cursor.getColumnIndex("choice1"));
```

```
        String                      choice2                     =
cursor.getString(cursor.getColumnIndex("choice2"));
        String                      choice3                     =
cursor.getString(cursor.getColumnIndex("choice3"));
        int                    correctChoice                    =
cursor.getInt(cursor.getColumnIndex("correct_choice"));
        TestingData  testingData  =  new  TestingData(questionId,
question, imageResId, choice1, choice2, choice3, correctChoice);
        testingDataList.add(testingData);
      } while (cursor.moveToNext());
    }
    cursor.close();
    db.close();
    return testingDataList;
  }
  public List<Material> getAllMaterials(int languageId) {
    List<Material> materials = new ArrayList<>();
    SQLiteDatabase db = this.getReadableDatabase();
    String[] columns = {
        COLUMN_MATERIAL_ID,
        COLUMN_MATERIAL_NAME,
        COLUMN_MATERIAL_TEXT
    };
    String selection = COLUMN_LANGUAGE_ID + " = ?";
    String[] selectionArgs = {String.valueOf(languageId)};
    Cursor   cursor   =   db.query(TABLE_MATERIALS,   columns,
selection, selectionArgs, null, null, null);
    if (cursor != null && cursor.moveToFirst()) {
      do {
        int                     materialId                      =
cursor.getInt(cursor.getColumnIndex(COLUMN_MATERIAL_ID));
        String                   materialName                   =
cursor.getString(cursor.getColumnIndex(COLUMN_MATERIAL_NAME));
        String                   materialText                   =
cursor.getString(cursor.getColumnIndex(COLUMN_MATERIAL_TEXT));
        Material    material    =    new    Material(materialId,
materialName, materialText);
        materials.add(material);
      } while (cursor.moveToNext());
    }
    if (cursor != null) {
      cursor.close();
    }
    db.close();
    return materials;
  }
  public boolean updateUserProgress(int userId, int newProgress)
{
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put("user_progress", newProgress);
    int rowsAffected = db.update("users", values, "user_id = ?",
new String[]{String.valueOf(userId)});
    return rowsAffected > 0;
  }
  public int getUserProgress(int userId) {
    SQLiteDatabase db = this.getReadableDatabase();
    String[] columns = {"user_progress"};
    String selection = "user_id = ?";
```

```
    String[] selectionArgs = {String.valueOf(userId)};
    Cursor  cursor  =  db.query("users",  columns,  selection,
selectionArgs, null, null, null);
    int userProgress = 0;
    if (cursor.moveToFirst()) {
      int columnIndex = cursor.getColumnIndex("user_progress");
      userProgress = cursor.getInt(columnIndex);
    }
    cursor.close();
    return userProgress;
  }
}
RegistrationActivity
package com.example.diplom;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
public class RegistrationActivity extends AppCompatActivity {
  EditText username, password, repassword;
  Button signup, signin;
  DBHelper DB;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    username = (EditText) findViewById(R.id.username);
    password = (EditText) findViewById(R.id.password);
    repassword = (EditText) findViewById(R.id.repassword);
    signup = (Button) findViewById(R.id.btnsignup);
    signin = (Button) findViewById(R.id.btnsignin);
    DB = new DBHelper(this);
    signup.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View view) {
        String user = username.getText().toString();
        String pass = password.getText().toString();
        String repass = repassword.getText().toString();
        if(user.equals("")||pass.equals("")||repass.equals(""))
          Toast.makeText(RegistrationActivity.this, "Будь ласка,
заповніть всі поля", Toast.LENGTH_SHORT).show();
        else{
          if(pass.equals(repass)){
            Boolean checkuser = DB.checkusername(user);
            if(checkuser==false){
              long userId = DB.insertData(user, pass);
              if(userId != -1){
                Toast.makeText(RegistrationActivity.this,
"Успішна реєстрація", Toast.LENGTH_SHORT).show();
                int userIdd = DB.getCurrentUserId(user);
                SharedPreferences          preferences          =
getSharedPreferences("MyPrefs", MODE_PRIVATE);
                SharedPreferences.Editor         editor         =
preferences.edit();
                editor.putInt("USER_ID", userIdd);
```

```
                editor.apply();
                Intent intent = new
Intent(getApplicationContext(), LanguageActivity.class);
                intent.putExtra("USER_ID", userId);
                startActivity(intent);
                finish();
              } else {
                Toast.makeText(RegistrationActivity.this,
"Невдала Реєстрація", Toast.LENGTH_SHORT).show();
              }
            } else {
              Toast.makeText(RegistrationActivity.this,
"Користувач вже існує! Увійдіть будь ласка",
Toast.LENGTH_SHORT).show();
            }
          } else {
            Toast.makeText(RegistrationActivity.this, "Різні
паролі", Toast.LENGTH_SHORT).show();
          }
        }
      }
    });
    signin.setOnClickListener(view -> {
      Intent intent = new Intent(getApplicationContext(),
LoginActivity.class);
      startActivity(intent);
    });
  }
}
LoginActivity
package com.example.diplom;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
public class LoginActivity extends AppCompatActivity {
  EditText username, password;
  Button btnlogin;
  DBHelper DB;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_login);

    username = findViewById(R.id.username1);
    password = findViewById(R.id.password1);
    btnlogin = findViewById(R.id.btnsignin1);
    DB = new DBHelper(this);
    btnlogin.setOnClickListener(view -> {
      String user = username.getText().toString();
      String pass = password.getText().toString();
      if(user.equals("")||pass.equals(""))
        Toast.makeText(LoginActivity.this, "Будь ласка,
заповніть всі поля", Toast.LENGTH_SHORT).show();
      else{
```

```
        Boolean checkuserpass = DB.checkusernamepassword(user,
pass);
        if (checkuserpass) {
          Toast.makeText(LoginActivity.this, "Успішний вхід",
Toast.LENGTH_SHORT).show();
          int userId = DB.getCurrentUserId(user);
          SharedPreferences preferences =
getSharedPreferences("MyPrefs", MODE_PRIVATE);
          SharedPreferences.Editor editor = preferences.edit();
          editor.putInt("USER_ID", userId);
          editor.apply();
          Intent languageIntent = new Intent(LoginActivity.this,
LanguageActivity.class);
          startActivity(languageIntent);
          finish();
        }
        else {
          Toast.makeText(LoginActivity.this, "Невірні дані",
Toast.LENGTH_SHORT).show();
        }
      }
    });
  }
}
MainPageActivity
package com.example.diplom;
import static
com.example.diplom.LanguageActivity.selectedLanguage;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import java.util.List;
public class MainPageActivity extends AppCompatActivity {
  DBHelper DB;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(getLayoutForSelectedLanguage());
    RecyclerView recyclerView =
findViewById(R.id.themeRecyclerView);
    recyclerView.setLayoutManager(new
LinearLayoutManager(this));
    DB = new DBHelper(this);
    setupImageViewClickListener(R.id.langicon,
LanguageActivity.class, this);
    setupImageViewClickListener(R.id.searchicon,
DictionaryActivity.class, this);
    setupImageViewClickListener(R.id.homeicon,
MainPageActivity.class, this);
    setupImageViewClickListener(R.id.usericon,
UserPageActivity.class, this);
    setupImageViewClickListener(R.id.settingsicon,
TranslatorActivity.class, this);
```

```
    setupImageViewClickListener(R.id.materialsicon,
MaterialsActivity.class, this);
    int languageId = 1;
    if ("Polish".equals(selectedLanguage)) {
      languageId = 2;
    }
    List<MyThemeData> themeList = DB.getAllThemes(languageId);
    if (themeList.isEmpty()) {
      Toast.makeText(this,       "Теми       не       знайдено",
Toast.LENGTH_SHORT).show();
    } else {
      ThemeAdapter  themeAdapter  =  new  ThemeAdapter(themeList,
MainPageActivity.this);
      recyclerView.setAdapter(themeAdapter);
    }
  }
  private int getLayoutForSelectedLanguage() {
    int languageId;
    if ("English".equals(selectedLanguage)) {
      languageId = 1;
      return R.layout.eng_main;
    }
    else if ("Polish".equals(selectedLanguage)) {
      languageId = 2;
      return R.layout.pol_main;
    }
    else{return R.layout.eng_main;}
  }
  private  void  setupImageViewClickListener(int  viewId,  final
Class<?>      targetActivity,      final      AppCompatActivity
currentActivity) {
    ImageView imageView = currentActivity.findViewById(viewId);
    imageView.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        Intent    intent    =    new    Intent(currentActivity,
targetActivity);
        currentActivity.startActivity(intent);
        if (targetActivity != currentActivity.getClass()) {
          currentActivity.finish();
        }
      }
    });
  }
}
ThemeAdapter
package com.example.diplom;
import android.content.Context;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
import java.util.List;
public           class           ThemeAdapter           extends
RecyclerView.Adapter<ThemeAdapter.MyViewHolder> {
```

```
  private Context context;
  private List<MyThemeData> myThemeDataList;
  public ThemeAdapter(List<MyThemeData> myThemeDataList, Context
context) {
    this.myThemeDataList = myThemeDataList;
    this.context = context;
  }
  @NonNull
  @Override
  public   MyViewHolder   onCreateViewHolder(@NonNull   ViewGroup
parent, int viewType) {
    LayoutInflater inflater = LayoutInflater.from(context);
    View   view   =   inflater.inflate(R.layout.my_row,   parent,
false);
    return new MyViewHolder(view);
  }
  @Override
  public void onBindViewHolder(@NonNull MyViewHolder holder, int
position) {
    final           MyThemeData           myThemeData           =
myThemeDataList.get(position);
    holder.theme_name.setText(myThemeData.getThemeName());

holder.theme_image.setImageResource(myThemeData.getThemeImage())
;
    holder.itemView.setOnClickListener(new
View.OnClickListener() {
      @Override
      public void onClick(View v) {
        int selectedThemeId = myThemeData.getThemeId();
        Intent      intent      =      new      Intent(context,
LessonsActivity.class);
        intent.putExtra("themeId", selectedThemeId);
        context.startActivity(intent);
      }
    });
  }
  @Override
  public int getItemCount() {
    return myThemeDataList.size();
  }
  public class MyViewHolder extends RecyclerView.ViewHolder {
    TextView theme_name;
    ImageView theme_image;
    public MyViewHolder(@NonNull View itemView) {
      super(itemView);
      theme_name = itemView.findViewById(R.id.theme_name_txt);
      theme_image                                                =
itemView.findViewById(R.id.theme_image_blob);
    }
  }
}
LessonsActivity
package com.example.diplom;

import android.content.Intent;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
```

```
import androidx.recyclerview.widget.RecyclerView;
import java.util.ArrayList;
import java.util.List;
public class LessonsActivity extends AppCompatActivity {
  private RecyclerView recyclerView;
  private LessonsAdapter adapter;
  private DBHelper dbHelper;
  private int selectedThemeId;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_lessons);
    Intent intent = getIntent();
    selectedThemeId = intent.getIntExtra("themeId", -1);
    recyclerView = findViewById(R.id.recyclerViewLessons);
    adapter = new LessonsAdapter(this, new ArrayList<>());
    dbHelper = new DBHelper(this);
    recyclerView.setAdapter(adapter);
    recyclerView.setLayoutManager(new
LinearLayoutManager(this));
    List<LessonData>                lessonList                =
dbHelper.getLessonsByThemeId(selectedThemeId);
    adapter.setLessonList(lessonList);
  }
}
LessonsAdapter
package com.example.diplom;
import android.content.Context;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
import java.util.List;
public           class           LessonsAdapter           extends
RecyclerView.Adapter<LessonsAdapter.ViewHolder> {
  private Context context;
  private List<LessonData> lessonList;
  public   LessonsAdapter(Context   context,   List<LessonData>
lessonList) {
    this.context = context;
    this.lessonList = lessonList;
  }
  @NonNull
  @Override
  public   ViewHolder   onCreateViewHolder(@NonNull   ViewGroup
parent, int viewType) {
    View                            view                            =
LayoutInflater.from(context).inflate(R.layout.item_lesson,
parent, false);
    return new ViewHolder(view);
  }
  @Override
  public void onBindViewHolder(@NonNull ViewHolder holder, int
position) {
    LessonData lesson = lessonList.get(position);
    holder.lessonNameTextView.setText(lesson.getLessonName());
```

```
    holder.itemView.setOnClickListener(new
View.OnClickListener() {
      @Override
      public void onClick(View v) {
        Intent    intent    =    new    Intent(v.getContext(),
LearningActivity.class);
        intent.putExtra("lessonId", lesson.getLessonId());
        v.getContext().startActivity(intent);
      }
    });
  }
  @Override
  public int getItemCount() {
    return lessonList.size();
  }
  public class ViewHolder extends RecyclerView.ViewHolder {
    TextView lessonNameTextView;
    public ViewHolder(@NonNull View itemView) {
      super(itemView);
      lessonNameTextView                                          =
itemView.findViewById(R.id.lessonNameTextView);
    }
  }
  public void setLessonList(List<LessonData> lessonList) {
    this.lessonList = lessonList;
    notifyDataSetChanged();
  }
}
LearningActivity
package com.example.diplom;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
import java.util.List;
public class LearningActivity extends AppCompatActivity {
  private TextView wordTextView;
  private ImageView lessonImageView;
  private TextView translationTextView;
  private TextView transcriptionTextView;
  private Button nextButton;
  private DBHelper dbHelper;
  private int lessonId;
  private List<LearningData> learningDataList;
  private int currentIndex;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.learning_activity);
    lessonId = getIntent().getIntExtra("lessonId", 0);
    dbHelper = new DBHelper(this);
    lessonImageView = findViewById(R.id.lessonimageView);
    wordTextView = findViewById(R.id.nativetextView);
    translationTextView = findViewById(R.id.translatedtextView);
    transcriptionTextView                                         =
findViewById(R.id.transcriptiontextView);
```

```
    nextButton = findViewById(R.id.nextbutton);
    nextButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        showNextLearningData();
      }
    });
    loadLearningData();
  }
  private void loadLearningData() {
    learningDataList                                    =
dbHelper.getLearningDataByLesson(lessonId);
    if (!learningDataList.isEmpty()) {
      currentIndex = 0;
      LearningData                learningData           =
learningDataList.get(currentIndex);
      displayLearningData(learningData);
      if (learningDataList.size() == 1) {
        startQuestionActivity();
      }
    }
  }
  private void displayLearningData(LearningData learningData) {
    wordTextView.setText(learningData.getNativeWord());
    translationTextView.setText(learningData.getTranslation());

transcriptionTextView.setText(learningData.getTranscription());

lessonImageView.setImageResource(learningData.getImageResId());
  }
  private void showNextLearningData() {
    currentIndex++;
    if (currentIndex < learningDataList.size()) {
      LearningData                learningData           =
learningDataList.get(currentIndex);
      displayLearningData(learningData);
    } else {
      startQuestionActivity();
    }
  }
  private void startQuestionActivity() {
    Intent intent = new Intent(this, TestingActivity.class);
    intent.putExtra("lessonId", lessonId);
    startActivity(intent);
    finish();
  }
}
TestingActivity
package com.example.diplom;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.os.Handler;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.util.Collections;
```

```
import java.util.List;
public class TestingActivity extends AppCompatActivity {
  private TextView questionTextView;
  private ImageView questionImageView;
  private Button choice1Button;
  private Button choice2Button;
  private Button choice3Button;
  private List<TestingData> testingDataList;
  private int currentQuestionIndex;
  private int score;
  private int numMistakes;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.question_activity);
    questionTextView = findViewById(R.id.questionTextView);
    questionImageView = findViewById(R.id.questionImageView);
    choice1Button = findViewById(R.id.choiceButton1);
    choice2Button = findViewById(R.id.choiceButton2);
    choice3Button = findViewById(R.id.choiceButton3);
    int lessonId = getIntent().getIntExtra("lessonId", 0);
    DBHelper dbHelper = new DBHelper(this);
    testingDataList                                    =
dbHelper.getQuestionDataByLesson(lessonId);
    dbHelper.close();
    Collections.shuffle(testingDataList);
    currentQuestionIndex = 0;
    score = 0;
    loadQuestionData();
  }
  private void loadQuestionData() {
    if (currentQuestionIndex < testingDataList.size()) {
      TestingData                  testingData                  =
testingDataList.get(currentQuestionIndex);

      questionTextView.setText(testingData.getQuestion());

questionImageView.setImageResource(testingData.getImageResId());
      choice1Button.setText(testingData.getChoice1());
      choice2Button.setText(testingData.getChoice2());
      choice3Button.setText(testingData.getChoice3());
      resetButtonColors();
      choice1Button.setOnClickListener(v                         ->
handleChoiceButtonClick(1, choice1Button));
      choice2Button.setOnClickListener(v                         ->
handleChoiceButtonClick(2, choice2Button));
      choice3Button.setOnClickListener(v                         ->
handleChoiceButtonClick(3, choice3Button));
    } else {
      navigateToResult();
    }
  }
  private   void   handleChoiceButtonClick(int   choice,   Button
selectedButton) {
    TestingData                  testingData                  =
testingDataList.get(currentQuestionIndex);
    int correctChoice = testingData.getCorrectChoice();
    if (choice != correctChoice) {
      numMistakes++;
```

```
    }
    if (choice == correctChoice) {
      selectedButton.setBackgroundColor(Color.GREEN);
      score++;
    } else {
      selectedButton.setBackgroundColor(Color.RED);
      if (correctChoice == 1) {
        choice1Button.setBackgroundColor(Color.GREEN);
      } else if (correctChoice == 2) {
        choice2Button.setBackgroundColor(Color.GREEN);
      } else if (correctChoice == 3) {
        choice3Button.setBackgroundColor(Color.GREEN);
      }
    }
    choice1Button.setEnabled(false);
    choice2Button.setEnabled(false);
    choice3Button.setEnabled(false);
    new Handler().postDelayed(() -> {
      if (currentQuestionIndex < testingDataList.size() - 1) {
        choice1Button.setEnabled(true);
        choice2Button.setEnabled(true);
        choice3Button.setEnabled(true);
        currentQuestionIndex++;
        loadQuestionData();
      } else {
        if (numMistakes > 2) {
          showTooManyMistakesMessage();
        } else {
          navigateToResult();
        }
      }
    }, 1500);
  }
  private void showTooManyMistakesMessage() {
    Toast.makeText(this, "Зробленено занадто багато помилок,
спробуй знову.", Toast.LENGTH_SHORT).show();

    score = 0;
    numMistakes = 0;
    recreate();
  }
  private void resetButtonColors() {
    choice1Button.setBackgroundColor(Color.LTGRAY);
    choice2Button.setBackgroundColor(Color.LTGRAY);
    choice3Button.setBackgroundColor(Color.LTGRAY);
  }
  private void navigateToResult() {
    Intent intent = new Intent(TestingActivity.this,
ResultActivity.class);
    intent.putExtra("Score", score);
    startActivity(intent);
    finish();
  }
}
ResultActivity
package com.example.diplom;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.ProgressBar;
```

```
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
public class ResultActivity extends AppCompatActivity {
  private TextView scoreTextView;
  private ProgressBar progressBar;
  private TextView progressTextView;
  private DBHelper dbHelper;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_result);
    scoreTextView = findViewById(R.id.scoreTextView);
    progressBar = findViewById(R.id.progressBar);
    progressTextView = findViewById(R.id.progressTextView);
    dbHelper = new DBHelper(this);
    int score = getIntent().getIntExtra("Score", 0);
    SharedPreferences                preferences                =
getSharedPreferences("MyPrefs", MODE_PRIVATE);
    int userId = preferences.getInt("USER_ID", 0);
    int currentProgress = dbHelper.getUserProgress(userId);
    int newProgress = currentProgress + score;
    dbHelper.updateUserProgress(userId, newProgress);
    int maxProgress = 1000;
    double   progressPercentage   =   (newProgress   /   (double)
maxProgress) * 100;
    progressBar.setProgress((int) progressPercentage);
    progressTextView.setText(getString(R.string.result_progress,
newProgress));
    scoreTextView.setText(getString(R.string.result_score,
score));
    if (newProgress >= 1000) {
      // TODO:
    } else if (newProgress >= 500) {
      // TODO:
    } else if (newProgress >= 100) {
      // TODO:
    } else {
      // TODO:
    }
  }
  @Override
  protected void onDestroy() {
    super.onDestroy();
    dbHelper.close();
  }
}
DictionaryActivity
package com.example.diplom;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
```

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.ArrayList;
import java.util.List;
import java.util.StringJoiner;
public class DictionaryActivity extends AppCompatActivity {
  private EditText inputEditText;
  private Button searchButton;
  private TextView translationTextView;
  private TextView synonymsTextView;
  private TextView meaningsTextView;
  private TextView examplesTextView;
  private static final String API_KEY =
"dict.1.1.20230605T174930Z.36223f412153d170.32f93cd9d82f6a51b449
b9cd62fe81a86d9cf8fc";
  private static final String API_ENDPOINT =
"https://dictionary.yandex.net/api/v1/dicservice.json/lookup";
  private static final String LANG_PAIR = "en-uk";
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.dictionary_activity);
    inputEditText = findViewById(R.id.inputEditText);
    searchButton = findViewById(R.id.searchButton);
    translationTextView =
findViewById(R.id.translationTextView);
    synonymsTextView = findViewById(R.id.synonymsTextView);
    meaningsTextView = findViewById(R.id.meaningsTextView);
    examplesTextView = findViewById(R.id.examplesTextView);
    searchButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        String inputText = inputEditText.getText().toString();
        if (!inputText.isEmpty()) {
          new DictionaryLookupTask().execute(inputText);
        }
      }
    });
  }
  private class DictionaryLookupTask extends AsyncTask<String,
Void, String> {
    @Override
    protected String doInBackground(String... params) {
      String inputText = params[0];
      String requestUrl = API_ENDPOINT + "?key=" + API_KEY +
"&lang=" + LANG_PAIR + "&text=" + inputText;
      try {
        URL url = new URL(requestUrl);
        HttpURLConnection connection = (HttpURLConnection)
url.openConnection();
        connection.setRequestMethod("GET");
        int responseCode = connection.getResponseCode();
        if (responseCode == HttpURLConnection.HTTP_OK) {
          InputStream inputStream = connection.getInputStream();
```

```
          BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(inputStream));
          StringBuilder response = new StringBuilder();
          String line;
          while ((line = bufferedReader.readLine()) != null) {
            response.append(line);
          }
          bufferedReader.close();
          inputStream.close();
          return response.toString();
        } else {
          return "Error: " + responseCode;
        }
      } catch (IOException e) {
        e.printStackTrace();
        return "Error: " + e.getMessage();
      }
    }
    @Override
    protected void onPostExecute(String result) {
      try {
        JSONObject responseJson = new JSONObject(result);
        JSONArray               definitionsArray               =
responseJson.getJSONArray("def");
        if (definitionsArray.length() > 0) {
          JSONObject              firstDefinition               =
definitionsArray.getJSONObject(0);
          JSONArray              translationsArray              =
firstDefinition.getJSONArray("tr");
          if (translationsArray.length() > 0) {
            JSONObject             firstTranslation             =
translationsArray.getJSONObject(0);
            String                  translation                 =
firstTranslation.getString("text");
            translationTextView.setText("Translation:\n"       +
translation);
            List<String> synonymsList = new ArrayList<>();
            List<String> meaningsList = new ArrayList<>();
            for (int i = 0; i < translationsArray.length(); i++)
{
              JSONObject             translationObj             =
translationsArray.getJSONObject(i);
              String synonym = translationObj.getString("text");
              synonymsList.add(synonym);

              JSONArray              meaningsArray              =
translationObj.optJSONArray("mean");
              if      (meaningsArray      !=      null      &&
meaningsArray.length() > 0) {
                for (int j = 0; j < meaningsArray.length(); j++)
{
                  JSONObject             meaningObj             =
meaningsArray.getJSONObject(j);
                  String meaning = meaningObj.getString("text");
                  meaningsList.add(meaning);
                }
              }
            }
```

```
            synonymsTextView.setText("Synonyms:\n"              +
getListAsString(synonymsList));
            if (meaningsList.size() > 0) {
              meaningsTextView.setText("Meanings:\n"            +
getListAsString(meaningsList));
            } else {
              meaningsTextView.setText("Meanings:");
            }
            String                    inputText                 =
inputEditText.getText().toString();
            new ExampleLookupTask().execute(inputText);
          } else {
            translationTextView.setText("Translation:");
            synonymsTextView.setText("Synonyms:");
            meaningsTextView.setText("Meanings:");
            examplesTextView.setText("");
          }
        } else {
          translationTextView.setText("No definitions found.");
          synonymsTextView.setText("");
          meaningsTextView.setText("");
          examplesTextView.setText("");
        }
      } catch (JSONException e) {
        e.printStackTrace();
        translationTextView.setText("Error: " + e.getMessage());
        synonymsTextView.setText("");
        meaningsTextView.setText("");
        examplesTextView.setText("");
      }
    }
  }
  private  class  ExampleLookupTask  extends  AsyncTask<String,
Void, String> {
    @Override
    protected String doInBackground(String... params) {
      String inputText = params[0];
      String                      requestUrl                    =
"https://api.dictionaryapi.dev/api/v2/entries/en/" + inputText;
      try {
        URL url = new URL(requestUrl);
        HttpURLConnection   connection   =   (HttpURLConnection)
url.openConnection();
        connection.setRequestMethod("GET");
        int responseCode = connection.getResponseCode();
        if (responseCode == HttpURLConnection.HTTP_OK) {
          InputStream inputStream = connection.getInputStream();
          BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(inputStream));
          StringBuilder response = new StringBuilder();
          String line;
          while ((line = bufferedReader.readLine()) != null) {
            response.append(line);
          }
          bufferedReader.close();
          inputStream.close();
          return response.toString();
        } else {
          return "Error: " + responseCode;
```

```
        }
      } catch (IOException e) {
        e.printStackTrace();
        return "Error: " + e.getMessage();
      }
    }
    @Override
    protected void onPostExecute(String result) {
      try {
        JSONArray responseArray = new JSONArray(result);

        if (responseArray.length() > 0) {
          StringBuilder examplesBuilder = new StringBuilder();
          for (int i = 0; i < responseArray.length(); i++) {
            JSONObject                wordObject                =
responseArray.getJSONObject(i);
            JSONArray              meaningsArray               =
wordObject.getJSONArray("meanings");
            for (int j = 0; j < meaningsArray.length(); j++) {
              JSONObject             meaningObject             =
meaningsArray.getJSONObject(j);
              JSONArray            definitionsArray            =
meaningObject.getJSONArray("definitions");
              for (int k = 0; k < definitionsArray.length();
k++) {
                JSONObject           definitionObject          =
definitionsArray.getJSONObject(k);
                String                  example                =
definitionObject.optString("example");
                if (!example.isEmpty()) {
                  examplesBuilder.append("Example:
").append(example).append("\n");
                }
              }
            }
          }
          if (examplesBuilder.length() > 0) {

examplesTextView.setText(examplesBuilder.toString());
          } else {
            examplesTextView.setText("No examples found.");
          }
        } else {
          examplesTextView.setText("No results found.");
        }
      } catch (JSONException e) {
        e.printStackTrace();
        examplesTextView.setText("Error: " + e.getMessage());
      }
    }
  }
  private String getListAsString(List<String> list) {
    StringJoiner joiner = new StringJoiner("\n");
    for (String item : list) {
      joiner.add(item);
    }
    return joiner.toString();
  }
}
```

TranslatorActivity

```
package com.example.diplom;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import org.jetbrains.annotations.NotNull;
import org.json.JSONException;
import org.json.JSONObject;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.MediaType;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
public class TranslatorActivity extends AppCompatActivity {
  private static final String TAG = "TranslatorActivity";
  private   static   final   String   API_URL   =   "https://deep-
translate1.p.rapidapi.com/language/translate/v2";
  private      static      final      String      API_KEY      =
"657bc24dbfmsh4e8b4e1d0a29eb0p17f565jsn2a1612290d91";
  private    static    final    String    API_HOST    =    "deep-
translate1.p.rapidapi.com";
  private EditText sourceEditText;
  private Button translateButton;
  private TextView translatedTextView;
  private Spinner sourceLanguageSpinner;
  private Spinner targetLanguageSpinner;
  private ArrayAdapter<String> languageAdapter;
  private List<String> languageList;
  private String selectedSourceLanguage;
  private String selectedTargetLanguage;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.translator_activity);
    sourceEditText = findViewById(R.id.source_text_edittext);
    translateButton = findViewById(R.id.translate_button);
    translatedTextView                                          =
findViewById(R.id.translated_text_textview);
    sourceLanguageSpinner                                       =
findViewById(R.id.source_language_spinner);
    targetLanguageSpinner                                       =
findViewById(R.id.target_language_spinner);

    languageList = new ArrayList<>();
    languageList.add("English");
    languageList.add("Ukrainian");
```

```
    languageList.add("Polish");
    languageAdapter = new ArrayAdapter<>(this,
android.R.layout.simple_spinner_item, languageList);
languageAdapter.setDropDownViewResource(android.R.layout.simple_
spinner_dropdown_item);
    sourceLanguageSpinner.setAdapter(languageAdapter);
    targetLanguageSpinner.setAdapter(languageAdapter);
    sourceLanguageSpinner.setSelection(0);
    targetLanguageSpinner.setSelection(1);
    selectedSourceLanguage = "en";
    selectedTargetLanguage = "uk";
    sourceLanguageSpinner.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
      @Override
      public void onItemSelected(AdapterView<?> parent, View
view, int position, long id) {
        selectedSourceLanguage = getLanguageCode(position);
      }
      @Override
      public void onNothingSelected(AdapterView<?> parent) {
      }
    });
    targetLanguageSpinner.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
      @Override
      public void onItemSelected(AdapterView<?> parent, View
view, int position, long id) {
        selectedTargetLanguage = getLanguageCode(position);
      }
      @Override
      public void onNothingSelected(AdapterView<?> parent) {
      }
    });
    translateButton.setOnClickListener(new
View.OnClickListener() {
      @Override
      public void onClick(View v) {
        String sourceText =
sourceEditText.getText().toString().trim();
        if (!sourceText.isEmpty()) {
          translateText(sourceText, selectedSourceLanguage,
selectedTargetLanguage);
        }
      }
    });
  }
  private void translateText(String text, String sourceLanguage,
String targetLanguage) {
    OkHttpClient client = new OkHttpClient();
    MediaType mediaType = MediaType.parse("application/json");
    String requestBody = "{\"q\":\"" + text + "\",\"source\":\""
+ sourceLanguage + "\",\"target\":\"" + targetLanguage + "\"}";
    RequestBody body = RequestBody.create(mediaType,
requestBody);
    Request request = new Request.Builder()
        .url(API_URL)
        .post(body)
        .addHeader("content-type", "application/json")
        .addHeader("X-RapidAPI-Key", API_KEY)
```

```
        .addHeader("X-RapidAPI-Host", API_HOST)
        .build();
    client.newCall(request).enqueue(new Callback() {
      @Override
      public void onFailure(@NonNull Call call, @NonNull
IOException e) {
      }
      @Override
      public void onResponse(@NotNull Call call, @NotNull
Response response) throws IOException {
        if (response.isSuccessful()) {
          String jsonResponse = response.body().string();
          try {
            JSONObject jsonObject = new
JSONObject(jsonResponse);
            JSONObject translations =
jsonObject.getJSONObject("data").getJSONObject("translations");
            final String translatedText =
translations.getString("translatedText");

            runOnUiThread(new Runnable() {
              @Override
              public void run() {
                translatedTextView.setText(translatedText);
              }
            });
          } catch (JSONException e) {
            e.printStackTrace();
          }
        } else {
          Log.e(TAG, "Response error: " + response.code());
        }
      }
    });
  }
  private String getLanguageCode(int position) {
    switch (position) {
      case 0:
        return "en";
      case 1:
        return "uk";
      case 2:
        return "pl";
      default:
        return "";
    }
  }
}
```