РЕФЕРАТ

Пояснювальна записка: ___ с., ___ рис., ___ табл., __ дод., ___ джерел.

Об'єкт розробки: програмний додаток інформаційного забезпечення навчально-тестуючого процесу.

Мета кваліфікаційної роботи: створення програмного забезпечення для об’єднання, зберігання та автоматизації ведення прогресу вивчення англійської мови, які будуть корисні та якими зможуть користуватися викладачі та школяри.

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

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

У другому розділі виконано аналіз існуючих рішень, обрано платформу для розробки, виконано проєктування і розробка програми, наведено опис алгоритму і структури функціонування програми, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження програми, описана робота програми.

В економічному розділі визначено трудомісткість розробленої інформаційної системи, проведений підрахунок вартості роботи по створенню програми та розраховано час на його створення.

Практичне значення полягає у створенні програмного додатка, що надає можливість користувачеві швидко та цікаво розширювати свій словарний запас, використовуючи різні режими програми, вивчати граматику, додавати власні переліки слів для вивчення.

Актуальність даного програмного продукту визначається тим, що через останні тенденції в освітньому просторі, які набувають чинності у зв’язку із переходом на нові стандарти навчання, дуже гостро постає питання вивчання іноземних мов. І не тільки вивчення, оскільки це відбувалося в загальноосвітніх навчальних закладах і раніше, а зміни вектору навчання іноземним мовам та якомога більшого поглиблення знань із цього напрямку.

Список ключових слів: АНГЛІЙСЬКА МОВА, УЧЕНЬ, ВЧИТЕЛЬ, ТРЕНАЖЕР, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ІНТЕРФЕЙС, ДОДАТОК, АЛГОРИТМ, БАЗА ДАНИХ.

ABSTRACT

Object of development: software application for information support of the educational and testing process.

The purpose of the qualification work is to create software for combining, storing and automating the progress of English language learning, which will be useful and which can be used by teachers and students.

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

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

In the second section the analysis of existing solutions is performed, the platform for development is chosen, the program is designed and developed, the algorithm and structure of program operation are described, input and output data are defined, characteristics of technical means parameters are given, call and program loading are described.

The economic section determines the complexity of the developed information system, calculates the cost of work to create a program and calculates the time for its creation.

The practical significance is to create a software application that allows the user to quickly and interestingly expand their vocabulary, using different modes of the program, learn grammar, add your own lists of words to learn.

The relevance of this software product is determined by the fact that due to the latest trends in the educational space, which are becoming effective in connection with the transition to new standards of learning, the issue of learning foreign languages is very acute. And not only the study, as it has happened in secondary schools before, but changes in the vector of teaching foreign languages and the greatest possible deepening of knowledge in this area.

List of keywords: ENGLISH, STUDENT, TEACHER, EXERCISE MACHINE, SOFTWARE, INTERFACE, APPENDIX, ALGORITHM, DATABASE.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – База даних;

ІС – інформаційна система;

ОС – операційна система;

ПЗ – програмне забезпечення;

ПК – персональний комп'ютер;

СУБД – система управління базами даних.

ЗМІСТ

ВСТУП

Мета навчання іноземної мови - це комунікативна діяльність учнів, тобто практичне володіння іноземною мовою. Завдання вчителя активізувати діяльність кожного учня в процесі навчання, створити ситуації для їх творчої активності.

В умовах аудиторного форми навчання викладач не завжди має можливість приділити належну увагу кожному учневі. Тому багато хто з них втрачають мотивацію до навчання, що приводить до істотного зниження рівня їх знань. Тому, одним з основних завдань вчителя є активізація діяльності кожного учня в процесі навчання, створення ситуації для їх творчої активності. У зв'язку з цим використання комп'ютера і мультимедійних засобів допомагає реалізувати особистісно-орієнтований підхід в навчанні та забезпечує індивідуалізацію з урахуванням особливостей учнів.

Виходячи з вищезазначеного тема кваліфікаційної роботи «Розробка програмного забезпечення для вивчення англійської мови дітей молодшого шкільного віку засобами середовища Qt Creator» є актуальною, адже наше суспільство перебуває на етапі глобального технічного розвитку, як наслідок, виникає необхідність в вивченні іноземних мов за допомогою програмних додатків.

Актуальність розробки визначається тим, що через останні тенденції в освітньому просторі, які набувають чинності у зв’язку із переходом на нові стандарти навчання, дуже гостро постає питання вивчання іноземних мов. І не тільки вивчення, оскільки це відбувалося в загальноосвітніх навчальних закладах і раніше, а зміни вектору навчання іноземним мовам та якомога більшого поглиблення знань із цього напрямку.

Мета створення даної розробки полягає у підвищенні ефективності засвоєння матеріалу учнями та обрання індивідуального темпу навчання, згідно з особистими можливостями.

Основна функція програми полягає у забезпеченні комфортного вивчення мови та перевірки знань в ігровій формі.

Таким чином це дає змогу користувачеві швидко та цікаво розширювати свій словарний запас, використовуючи різні режими програми, вивчати граматику, додавати власні переліки слів для вивчення.

В даній кваліфікаційній роботі засобами середовища Qt Creator розроблено програмний модуль, який призначений для полегшення роботи викладачів та підвищення освітнього рівня у школярів.

Програмний продукт пройде технічне випробування серед вчителів англійської мови навчального закладу з метою практичної користі при застосуванні на заняттях. Очікується позитивна оцінка естетики та нетрадиційного графічного інтерфейсу, який робить можливим сприйняття матеріалу як ігрової практики, а не як навчального матеріалу.

Наступними кроками у розвитку інновацій освітнього процесу мають стати логічні продовження даного програмного проекту із різних напрямків вивчення англійської мови. Також планується передати дані проекти для поточного технічного і навчального тестування в інші навчальні заклади.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Загальні відомості з предметної галузі

1.1.1. Аналіз існуючих методів та програм для вивчення англійської мови

Процес вивчення англійської мови потребує запам’ятовування величезної кількості інформації.

Пам'ять – це властивість людського мозку, яка дозволяє записувати, зберігати і при необхідності відтворювати інформацію. Пам'ять можна розділити на 3 основні види: сенсорна пам'ять, короткочасна пам'ять і довгострокова пам'ять [15].

Найчастіше при вивченні англійської мови застосовують оперативну пам’ять, але її можливості досить обмежені. Залежно від деяких факторів, інформація може зберігатися в цій частині пам'яті від 5 годин до 3-х місяців (але не більше), а далі забувається. Більшість програм для вивчення англійської орієнтовані на цей вид пам’яті, через що вони є доволі неефективними.

При вивченні іноземних мов переклад інформації в довготривалу пам'ять грає ключову роль для ефективного запам'ятовування. У довготривалій пам'яті інформація зберігається тривалий час, але доводиться постійно виконувати (від 2 до 8 разів) повторення.

Тому вибір правильної частоти повторення матеріалу є важливим при вивченні слів та граматики в англійській мові.

Основним методом самостійного вивчення англійської мови є граматичний метод.

Граматичний метод базується на читанні, вивченні граматики, перекладу з однієї мови на іншу, заучуванні слів іноземної мови. Головними джерелами інформації при цьому є посібники. Кожна подана тема поділяється на окремі підтеми до яких подаються правила, винятки з правил та надається контроль

отриманих знань. Значним недоліком цього методу є те, що обсяг розглянутих прикладів, як правило, є невеликим і не охоплює весь матеріал, що викликає необхідність в пошуку інших друкованих матеріалів. Також недоліком є складність перевірки правильності відповіді й виклад перевірочних завдань у сталому порядку.

1.1.2. Огляд існуючих рішень

Зараз існує велика кількість програм та веб-сторінок для вивчення англійської мови, зокрема й до завдань з частини Reading and Use of English. Значна частина таких програмних засобів це електронні тести, які повторюють матеріал з підручника та автоматично оцінюють правильність відповіді. Сайти [4] надають велику кількість словникових вправ потрібних для підготовки до екзамену та містять навчальний матеріал, але вони більш націлені на перевірку знань, аніж на їх отримання, бо користувач повинен сам знати свій рівень підготовки та щоб вивчити матеріал йому потрібно декілька разів виконати одну й ту саму вправу, так як словникова база вправ різна. Також мінусами є виклад завдання в сталому порядку й наявність лише однією правильної відповіді, в той час коли їх може бути декілька.

На сьогодні розроблено багато різноманітних мобільних додатків для вивчення мови. Такі додатки як «FCE Academy 2018 Lite – for the B2 First» та «Ready4FCE», які встановлюються на пристрої, що використовують операційну систему Android за своїм змістом вони схожі на веб-сайти та мають такі ж недоліки. Суттєвою перевагою є те, що мобільні додатки слідкують за особистим прогресом користувача.

Ще недоліки розглянутих методів: неможливість конфігурації бази даних навчальних завдань, те що не надається переклад, або тлумачення правильної відповіді, а також відсутність поступового ускладнення завдань в залежності від рівня знань користувача.

Найпоширенішим прикладом подібного до теми програмного продукту кваліфікаційної роботи є мобільний додаток Words.

Особливості Words:

– 10 унікальних тренувань, кожне тренування вчить конкретному навичку, що вивчається;

– вивчення готових наборів слів або додавання своїх;

– вбудований словник із зручним пошуком;

– система досягнень і докладна статистика навчання;

– розумні нагадування про заняття;

– вимова, приклади використання і картинки для кожного слова;

– індивідуальна програма навчання;

– не потрібно доступ до інтернету — займатися можна де завгодно.

Words дозволяє легко вчити іноземну мову за допомогою цікавих тренувань в ігровій формі, в які хочеться грати знову і знову. Кожне тренування вчить конкретному навику нової мови, створюючи всебічний досвід. За допомогою Words можна дуже швидко запам'ятовувати нові слова і легко перевіряти свої знання, тренувати правопис і сприйняття слів на слух.

Унікальний алгоритм покращує процес навчання, використовуючи перевірені методи з галузі дослідження пам'яті. Words адаптується індивідуально до користувача, вибірково повторює слова, з якими у нього були проблеми. Завершуючи уроки і граючи в ігри, користувач буде набирати досвід, переходити на більш високі рівні і колекціонувати нагороди.

Тематичні блоки дозволяють навчати саме ті теми, які цікаві зараз: людина, їжа, транспорт, здоров'я, будинок, спорт, природа, одяг, гроші та ін. [7].

1.2. Призначення розробки та область застосування

Розробка програми кваліфікаційної роботи, якою зможуть користуватися вчителі та школярі, призначена для об’єднання, зберігання та автоматизації ведення прогресу вивчення англійської мови. Основна функція програми полягає у забезпеченні комфортного вивчення мови та перевірки знань в ігровій формі. Таким чином це дає змогу користувачеві швидко та цікаво розширювати свій словарний запас, використовуючи різні режими програми, вивчати граматику, додавати власні переліки слів для вивчення.

В результаті використання даного програмного забезпечення має підвищитися загальний рівень успішності у школярів та полегшиться певним чином робота викладачів.

1.3. Підстава для розробки

Підставою для розробки кваліфікаційної роботи бакалавра на тему «Розробка програмного забезпечення для вивчення англійської мови дітей молодшого шкільного віку засобами середовища Qt Creator» є наказ по Національному технічному університету «Дніпровська політехніка» від __.__. 2021р. № ____-_.

1.4. Постановка завдання

Виходячи з відомостей з огляду галузі застосування та існуючих аналогів системи, стає зрозуміло що багато методів самостійного вивчення англійської мови є доволі неефективними. Тому ставиться мета роботи, якою є створення програмного забезпечення, яке б підвищило ефективність існуючих методів навчання англійської мови та полегшало б засвоєння матеріалу учнями та обрання індивідуального темпу навчання, згідно з особистими можливостями.

.

В ході виконання кваліфікаційної роботи має бути розроблений програмний додаток, що дозволяє вивчати англійську мову за допомогою ігрової форми в різних режимах запам’ятовування слів.

Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Аналіз існуючих методів та навчальних програм для вивчення англійської мови.
2. Пошук шляхів удосконалення існуючих програм навчання англійській мові.
3. Складання бази даних.
4. Створення алгоритму програми для самостійного вивчення.
5. Розробка програмного забезпечення у вигляді Windows-додатку.
6. Практичне використання програмного забезпечення при вивченні англійської мови.

Програмний додаток повинен мати простий інтуїтивно-зрозумілий інтерфейс та містити підказки та перевірку коректності введених даних, та дозволяти працювати з ним людям з мінімальними навиками володіння комп’ютером. Передбачено, що додатком може користуватися кожний.

1.5. Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

В межах даної кваліфікаційної роботи потрібно створити програмний додаток для вивчення англійської мови учнями молодшого шкільного віку.

Програма повинна виконувати такі функції по веденню бази даних:

– додавання, редагування та видалення слів;
– додавання, редагування та видалення тем;
– додавання, редагування та видалення учнів.

Слова вивчаються за допомогою карток зі словом, перекладом та відповідним зображенням.

Перевірка вивчення слів може відбуватися в таких режимах (за кожний

режим користувачу начисляється визначена кількість балів):

1. Збір слова із заданих букв: 10 балів.
2. Обирання вірного перекладу: 7 балів.
3. Обирання вірного перекладу із 5 пар слів: 5 бала.
4. Виділення слова серед випадкової матриці букв: 8 балів.
5. Обирання «правильний» або «неправильний» переклад слова: 5 балів.

Коли балів стає більше 20, слово вважається вивченим і його вже не використовують в режимах, де в цьому немає необхідності (необхідність може виникнути, наприклад, в третьому режимі, тому що кількість слів в ньому 5, а невивчених може залишитися менше).

Користувач повинен мати можливість додавати свою тему. Програма повинна виконувати перевірки на коректність введених даних цієї теми:

– назва теми;

– слова (не більше 40).

Програмний додаток повинен перевіряти усі введені дані на коректність та видавати повідомлення у випадку помилкового вводу.

Дані про слова:

– слово на англійській;

– слово на російській;

– зображення до слова.

Дані про тему:

– назва теми;

– зображення до теми.

Дані про учнів:

– ім’я та фамілія.

Завдання кваліфікаційної роботи повинно бути реалізоване з використанням таких ролей користувачів: «Вчитель» та «Учень».

Для ролі користувача «Вчитель» повинні бути реалізовані функції для створення, редагування та видалення списків тем та учнів, налаштування

звукового супроводу, перегляду успіхів учнів.

Для ролі користувача «Учень» повинні бути реалізовані функції для вивчення слів та тестування на їх знання.

1.5.2. Вимоги до інформаційної безпеки

Для надійної роботи системи необхідно:

1. Використовувати ліцензійне програмне забезпечення на сервері.
2. Здійснювати захист від вірусів на сервері.
3. Здійснювати захист від несанкціонованого доступу.
4. Застосовувати на сервері джерело безперебійного живлення для захисту від перепадів напруги або збоїв у живленні.
5. Здійснювати контроль даних, що вводяться користувачами.
6. Автоматично завершувати сеанс роботи з користувачем у разі тривалої перерви його активності.

Надійність роботи розроблюваного програмного забезпечення залежить від надійності операційної системи, під управлінням якої вона буде функціонувати і розроблюваного ПЗ. Також за допомогою валідатора QValidator можна забезпечити контроль введених користувачем даних.

1.5.3. Вимоги до складу та параметрів технічних засобів

Системні вимоги:

– 1 GB RAM (рекомендується 1.5 GB +);

– 1-2 GB вільного простору на жорсткому диску;

– стандартний GPU з підтримкою DirectX 9.0 та вище;

– роздільна здатність екрану 1024x768 та вище;

– Intel® Pentium® або сумісний, мінімум 1.4 GHz (рекомендується 2.2GHz +);

– переферійні пристрої введення-виведення: клавіатура, миша.

1.5.4. Вимоги до інформаційної та програмної сумісності

Дане програмне забезпечення повинне бути розроблене з використанням наступних програмно-апаратних засобів:

– операційна система Microsoft Windows XP/Vista/7;

– мови програмування C++ у середовищі QTCreator.

Програма повинна являти собою самостійний виконуваний модуль, бути структурована і закоментована.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1. Функціональне призначення програми

Розроблений програмний додаток дозволяє вивчати англійську мову за допомогою ігрової форми в різних режимах запам’ятовування слів.

Програма виконує такі функції по веденню бази даних:

– додавання, редагування та видалення слів;

– додавання, редагування та видалення тем;

– додавання, редагування та видалення учнів.

Слова вивчаються за допомогою карток зі словом, перекладом та відповідним зображенням. Виконується перевірка вивчення слів

Додаток реалізований з використанням таких ролей користувачів: «Вчитель» та «Учень».

Програмний додаток перевіряє усі введені дані на коректність та видає повідомлення у випадку помилкового вводу.

2.2. Опис застосованих математичних методів

В даному програмному додатку ні під час розробки, ні під час тестування та роботи програми, не використовуються та не розглядаються ніякі математичні методи.

2.3. Опис використаної архітектури та шаблонів проектування

Model-view-controller (MVC) − схема використання декількох шаблонів проектування, за допомогою яких модель даних програми, призначений для користувача інтерфейс і взаємодія з користувачем розділені на три окремих

компонента так, що модифікація одного з компонентів надає мінімальний вплив на інші. Дана схема проектування часто використовується для побудови архітектурного каркаса, коли переходять від теорії до реалізації в конкретній предметній області.

Основна мета застосування цієї концепції полягає в поділі бізнес-логіки (моделі) від її візуалізації (уявлення, виду). За рахунок такого поділу підвищується можливість повторного використання. Найбільш корисне застосування даної концепції, коли користувач повинен бачити ті ж самі дані одночасно в різних контекстах і/або з різних точок зору. Зокрема, виконуються наступні завдання:

1. До однієї моделі можна приєднати кілька видів, при цьому не зачіпаючи реалізацію моделі. Наприклад, деякі дані можуть бути одночасно представлені у вигляді електронної таблиці, гістограми і кругової діаграми.

2. Не зачіпаючи реалізацію видів, можна змінити реакції на дії користувача (натискання мишею на кнопці, введення даних), для цього досить використовувати інший контролер.

3. Ряд розробників спеціалізуються тільки в одній з областей: або розробляють графічний інтерфейс або розробляють бізнес-логіку. Тому можливо добитися, що програмісти, які займаються розробкою бізнес-логіки (моделі), взагалі не будуть обізнані про те, яке уявлення буде використовуватися.

Концепція MVC дозволяє розділити дані, подання та обробку дій користувача на три окремих компонента (рис. 2.1):

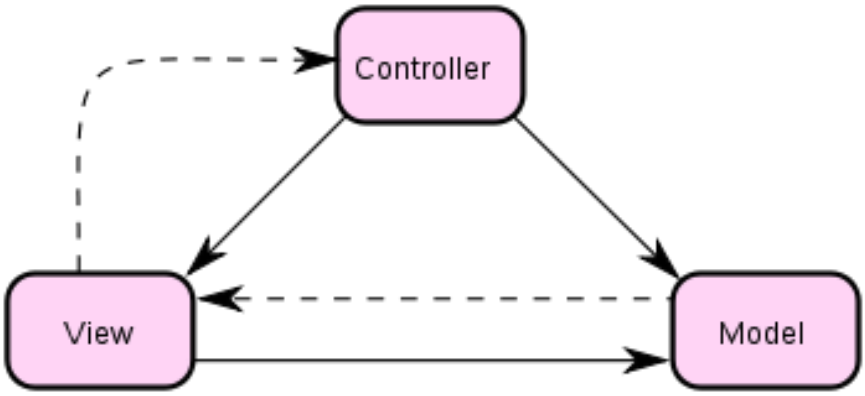

Рис. 2.1. Концепція MVC

1. Модель (англ. Model). Модель надає знання: дані та методи роботи з цими даними, реагує на запити, змінюючи свій стан. Не містить інформації, як ці знання можна візуалізувати.

2. Представлення, вид (англ. View). Відповідає за відображення інформації (візуалізація). Часто в якості уявлення виступає форма (вікно) з графічними елементами.

3. Контролер (англ. Controller). Забезпечує зв'язок між користувачем і системою: контролює введення даних користувачем і використовує модель і уявлення для реалізації необхідної реакції.

Важливо відзначити, що як уявлення, так і контролер залежать від моделі. Однак модель не залежить ні від уявлення, ні від контролера. Тим самим досягається призначення такого поділу: воно дозволяє будувати модель незалежно від візуального представлення, а також створювати кілька різних уявлень для однієї моделі.

Для реалізації схеми Model-View-Controller використовується досить велика кількість шаблонів проектування (в залежності від складності архітектурного рішення), основні з яких Спостерігач, Стратегія, Компоновщик.

Найбільш типова реалізація відокремлює вид від моделі, шляхом встановлення між ними протоколу взаємодії, використовуючи апарат подій (підписка / оповіщення). При кожній зміні внутрішніх даних в моделі, модель оповіщає всіх залежних від неї уявлення, і уявлення оновлюється. Для цього

використовується шаблон проектування наблюдача. При обробці реакції користувача, вид вибирає залежно від потрібної реакції потрібний контролер, який забезпечить ту чи іншу зв'язок з моделлю. Для цього використовується шаблон проектування стратегія, або замість цього може бути модифікація з використанням шаблону проектування команда. А для можливості однотипного поводження з підоб'єкти складно-складеного ієрархічного виду, може використовуватися шаблон проектування Компоновщик. Крім того, можуть використовуватися і інші шаблони проектування, наприклад, фабричний метод, який дозволить задати за замовчуванням тип контролера для відповідного виду.

2.4. Опис використаних технологій та мов програмування

Для реалізації програмного продукту було використано середовище Qt Creator, програмний продукт написаний на мові програмування C++, СУБД було використано MySQL.

2.4.1. Опис середовища розробки Qt Creator

Qt Creator - це повністю інтегроване середовище розробки (IDE), яке надає інструменти проєктування і розробки складних додатків для безлічі настільних і мобільних платформ.

Одним з найголовніших досягнень Qt Creator є те, що воно дозволяє команді розробників працювати над проєктом на різних платформах з використанням загальних інструментів для розробки і налагодження.

Щоб бути в змозі збирати і запускати додатки, Qt Creator потребує тієї ж інформації, яка буде потрібно компілятору. Ця інформація вказана в налаштуваннях збірки і запуску проєкту.

Створення проєкту дозволить:

– групувати файли разом;

– додати власні кроки збірки;

– включити форми і файли ресурсів;

– вказати налаштування для запуску.

Є можливість або створити проєкт з нуля, або імпортувати існуючий проєкт. Qt Creator генерує всі необхідні файли в залежності від типу створюваного проєкту. Наприклад, якщо обрати створення додатка з графічним інтерфейсом користувача (GUI), Qt Creator створить порожній .ui файл, який можна змінити в інтегрованому Qt Designer. Qt Creator інтегроване з кросплатформеними системами автоматизації збирання: qmake і CMake. Також є можливість імпортувати існуючі проєкти, які не використовують qmake або CMake, і вказати Qt Creator просто проігнорувати систему збирання.

Qt Creator поставляється з редактором коду і Qt Designer для проєктування і складання графічних інтерфейсів користувача з віджетів Qt.

Так як воно є IDE, Qt Creator відрізняється від текстового редактора тим, що знає як збирати і запускати додатки. Воно розуміє мови C ++ і QML як код, а не як простий текст. Це дозволяє йому:

– надавати можливість писати добре форматований код;

– аналізувати, що користувач хоче написати, і доповнювати код;

– відображати повідомлення про помилки і попередження;

– надавати можливість переміщатися між класами, функціями і символами;

– надавати контекстно-залежну довідку по класах, функцій і символам;

– осмислено перейменовувати символи так, що інші символи з таким же ім'ям, але які належать іншим областям дії, не будуть перейменовані;

– показувати вам місце в коді де функція була описана або викликана.

Можна використовувати Qt Designer щоб мати у своєму розпорядженні і налаштовувати ваші віджети або діалоги і тестувати їх використовуючи різні стилі. Створені за допомогою Qt Designer віджети і форми легко інтегруються в програмний код з використанням механізму сигналів і слотів Qt, які дозволять вам легко визначити поведінку графічних елементів. Всі властивості,

встановлені в Qt Designer, можуть бути динамічно змінені в коді. Більш того, такі особливості як просування віджетів і власні модулі дозволять вам використовувати власні віджети з Qt Designer.

Qt Creator надає підтримку для збірки і запуску додатків на Qt для настільних комп'ютерів (Windows, Linux і Mac OS) і мобільних пристроїв (Symbian, Maemo і MeeGo). Налаштування збірки дозволять вам швидко перемикатися між цілями збірки.

Qt Creator інтегроване з набором корисних інструментів, такі як системи управління версіями і емулятор Qt.

Доступні вам функції в Qt Creator залежать від системи управління версіями. Базові функції доступні для всіх підтримуваних систем. Вони включають порівняння файлів з останньою версією, що знаходиться в сховищі, і відображення різниці, перегляд історії версій і подробиць змін, анотацію файлів і прийняття та скасування змін [1].

2.4.2. Опис мови програмування C++

C++ компільована, статично типізована мова програмування загального призначення.

Підтримує такі парадигми програмування як процедурне програмування, об'єктно-орієнтоване програмування, узагальнене програмування, забезпечує модульність, роздільну компіляцію, обробку винятків, абстракцію даних, оголошення типів (класів) об'єктів, віртуальні функції. Стандартна бібліотека включає, в тому числі, загальновживані контейнери і алгоритми. C ++ поєднує властивості як високорівневих, так і низькорівневих мов. У порівнянні з її попередником мовою C, - найбільшу увагу приділено підтримці об'єктно-орієнтованого і узагальненого програмування.

C++ широко використовується для розробки програмного забезпечення, будучи однією з найпопулярніших мов програмування. Область її застосування включає створення операційних систем, різноманітних прикладних програм,

драйверів пристроїв, додатків для вбудованих систем, високопродуктивних серверів, а також розважальних програм. Існує безліч реалізацій мови C++, як безкоштовних, так і комерційних і для різних платформ [2].

C++ мова, що складалася еволюційно. Кожен елемент C++ запозичувався з інших мов окремо і незалежно від інших елементів (ніщо з запропонованого C++ за всю історію його розвитку не було нововведенням в Computer Science), що зробило мову надзвичайно складною, з безліччю дублюючих і взаємно суперечливих елементів, блоки яких засновані на різних формальних базах.

Критики C++ не протиставляють їй який-небудь конкретну мову, а навпаки, стверджують, що для будь-якого випадку застосування C++ завжди існує альтернативний інструментарій, що дозволяє вирішити ту ж задачу більш ефективно і якісно. У свою чергу, прихильники C++ вважають некоректним порівнювати різні аспекти C++ з абсолютно різними мовами, так як загальний набір засобів і можливостей C++ істотно ширше, ніж в більшості мов, з якими проводиться порівняння, і сама по собі широта можливостей , на їх погляд, є вагомим виправданням недосконалості кожної окремо взятої можливості. Більш того, на їхню думку, висока сумісність з C і є однією з важливих характеристик мови, і тому всі недоліки C++ виправдані перевагами, наданими цій сумісністю.

Переваги:

- висока сумісність з мовою C;
- обчислювальна продуктивність;
- підтримка різних стилів програмування: структурне, об'єктно-орієнтоване, узагальнене програмування, функціональне програмування;
- автоматичний виклик деструкторів об'єктів (в порядку зворотному виклику конструкторів) спрощує і підвищує надійність управління пам'яттю і іншими ресурсами (відкритими файлами, мережевими з'єднаннями, тощо);
- перевантаження операторів;
- шаблони (дають можливість побудови узагальнених контейнерів і алгоритмів для різних типів даних);

− можливість розширення мови для підтримки парадигм, які не підтримуються компіляторами безпосередньо

− доступність (для C ++ існує величезна кількість навчальної літератури, перекладеної на різні мови).

Недоліки:

− погано продуманий синтаксис звужує спектр застосування мови;

− мова не містить багатьох важливих можливостей;

− мова містить небезпечні можливості;

− продуктивність праці програмістів на мові виявляється невиправдано низька;

− громіздкість синтаксису [3].

2.4.3. Опис СУБД MySQL

MySQL це реляційна система управління базами даних з відкритим вихідним кодом. В даний час ця СУБД одна з найбільш популярних в веб-додатках - переважна більшість CMS використовує саме MySQL (часто тільки її, без альтернатив), а майже всі веб-фреймворки підтримують MySQL вже на рівні базової конфігурації (без додаткових модулів) [4].

З переваг СУБД MySQL варто відзначити простоту використання, гнучкість, низьку вартість володіння (щодо платних СУБД), а також масштабованість і продуктивність.

MySQL дозволяє зберігати цілочисельні значення зі знаком і беззнакові, довжиною в 1, 2, 3, 4 і 8 байтів, працює із строковими і текстовими даними фіксованої і змінної довжини, дозволяє здійснювати SQL-команди SELECT, DELETE, INSERT, REPLACE і UPDATE, забезпечує повну підтримку операторів і функцій в SELECT- і WHERE- частинах запитів, працює з GROUP BY і ORDER BY, підтримує групові функції COUNT, AVG, STD, SUM, MAX і MIN, дозволяє використовувати JOIN в запитах, в т.ч. LEFT OUTER JOIN і

RIGHT OUTER JOIN, підтримує реплікацію, транзакції, роботу з зовнішніми ключами і каскадні зміни на їх основі, а також забезпечує багато інших функціональних можливостей.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Є й інші типи таблиць, розроблені спільнотою.

СУБД MySQL з'явилася в 1995. Написана на C і C++, протестована на безлічі різних компіляторів і працює на різних платформах. С 2010 року розроблення та підтримку MySQL здійснює корпорація Oracle. Продукт поширюється як під GNU GPL, так і під власною комерційною ліцензією. Однак за умовами GPL, якщо яка-небудь програма включає вихідні коди MySQL, то і ця програма теж повинна розповсюджуватися за ліцензією GPL. Для небажаючих відкривати вихідні тексти своїх програм якраз передбачена комерційна ліцензія, яка, на додаток до можливості розробки під «закритою» ліцензією, забезпечує якісну сервісну підтримку. Спільнотою розробників MySQL створені різні відгалуження - Drizzle, OurDelta, Percona Server і MariaDB, всі ці відгалуження вже існували на момент отримання прав на MySQL корпорацією Oracle.

Зараз MySQL разом з форком MariaDB займають почесне перше місце, а слідом за ними йде PostgreSQL. Решта СУБД в веб-проєктах використовуються значно рідше [5].

2.5. Опис структури програми та алгоритмів її функціонування

2.5.1. Діаграма сценаріїв

Діаграма сценаріїв демонструє розподіл функціоналу, який припадає на кожного користувача програми, тим самим чудово демонструє рівні доступу до програми. У кваліфікаційній роботі передбачено два типи авторів, на яких

припадає різний обсяг функціоналу:

– вчитель;

– учень.

Автор «Вчитель» має найбільший рівень доступу до програми, відповідно на нього припадає функціонал налаштування, може контролювати всі дані про теми та учнів, переглядати успіхи учнів, налаштовувати звуковий супровід.

Автор «Учень» має найменший рівень доступу до програми, відповідно на нього припадає найменше функціонала, може вивчати слова та проходити тести.

Якщо актори визначають зовнішню сутність системи, то внутрішню відображають сценарії. Розглядаючи окремі сценарії, можна розділити функціональність системи на окремі складові, з якими можливо працювати паралельно.

Для сценаріїв визначають два типи відношень:

– відношення «розширює» означає те, що функції одного сценарію є доповненнями до функцій іншого сценарію;

– відношення «використовує» означає те, що конкретний сценарій може бути використаний для розширення кількома іншими сценаріями.

Діаграма сценаріїв предметної області індивідуального завдання зображена на рис. 2.2 .

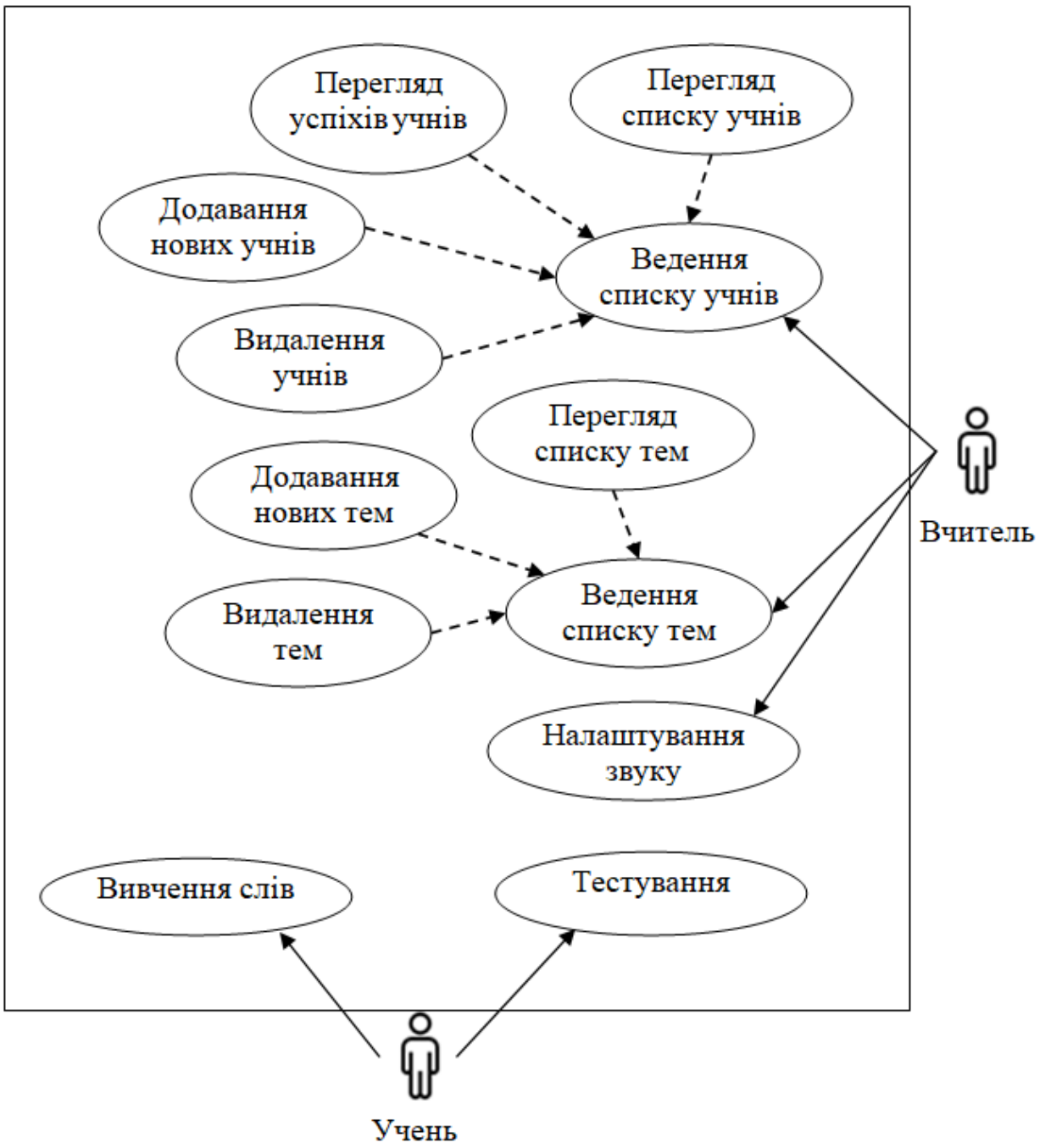

Рис. 2.2. Діаграма сценаріїв предметної області

2. 5.2. Проєктування бази даних

Для реалізації програмного додатку була обрана СУБД MySQL 8.0.15, у якій створена база даних, що містить шість таблиць, а саме:

В таблиці «card» зберігається інформація про теми. Дана таблиця зв’язана

з таблицею «having_c». Структура таблиці «card» представлена на рис. 2.3.

Field	Type	Null	Key	Default	Extra
name_c	varchar(40)	NO		NULL	
id_c	int(6)	NO	PRI	NULL	auto_increment

Рис. 2.3. Структура таблиці «card»

В таблиці «users» зберігається інформація про користувачів. Дана таблиця зв’язана з таблицями «having_u» та «stopka». Структура таблиці «users» представлена на рис. 2.4.

Field	Type	Null	Key	Default	Extra
id_u	int(6)	NO	PRI	NULL	auto_increment
name_user	varchar(40)	NO		NULL	

Рис. 2.4. Структура таблиці «users»

В таблиці «word» зберігається інформація про слова. Дана таблиця зв’язана з таблицями «having_c», «having_u» та «stopka». Структура таблиці «word» представлена на рис. 2.5.

Field	Type	Null	Key	Default	Extra
eng	varchar(40)	NO		NULL	
rus	varchar(40)	NO		NULL	
id_w	int(6)	NO	PRI	NULL	auto_increment

Рис. 2.5. Структура таблиці «word»

Таблиця «having_c» створена для розбиття відношення багато до багатьох

між таблицями «card» та «word». Структура таблиці «having_c» представлена на рис. 2.6.

Field	Type	Null	Key	Default	Extra
id_card	int(6)	NO		NULL	
id_word	int(6)	NO		NULL	

Рис. 2.6. Структура таблиці «having_c»

Таблиця «having_u» створена для розбиття відношення багато до багатьох між таблицями «user» та «word». Структура таблиці «having_u» представлена на рис. 2.7.

Field	Type	Null	Key	Default	Extra
id_word	int(6)	NO		NULL	
id_user	int(6)	NO		NULL	

Рис. 2.7. Структура таблиці «having_u»

Таблиця «stopka» створена для збереження балів для слів, які вивчаються на даний момент. Дана таблиця пов’язана з таблицями «word» та «user». Структура таблиці «stopka» представлена на рис. 2.8.

Field	Type	Null	Key	Default	Extra
bal	int(2)	NO		NULL	
id_word	int(6)	NO		NULL	
id_user	int(6)	NO		NULL	

Рис. 2.8. Структура таблиці «stopka»

В таблиці «regime» зберігається інформація про режими. Дана таблиця

зв’язана з таблицею «bal_regime». Структура таблиці «regime» представлена на рис. 2.9.

Field	Type	Null	Key	Default	Extra
bal	int(2)	NO		NULL	
id_word	int(6)	NO		NULL	
id_user	int(6)	NO		NULL	

Рис. 2.9. Структура таблиці «regime»

Таблиця «bal_regime» створена для збереження балів, які отримав кожен учень в кожному режимі. Дана таблиця пов’язана з таблицями «regime» та «user». Структура таблиці «bal_regime» представлена на рис. 2.10.

Field	Type	Null	Key	Default	Extra
name_r	varchar(50)	NO		NULL	
id_r	int(6)	NO	PRI	NULL	
score	int(2)	NO		NULL	

Рис. 2.10. Структура таблиці «bal_regime»

Загальна схема таблиць та зв’язки між ними продемонстровано на рис. 2.11.

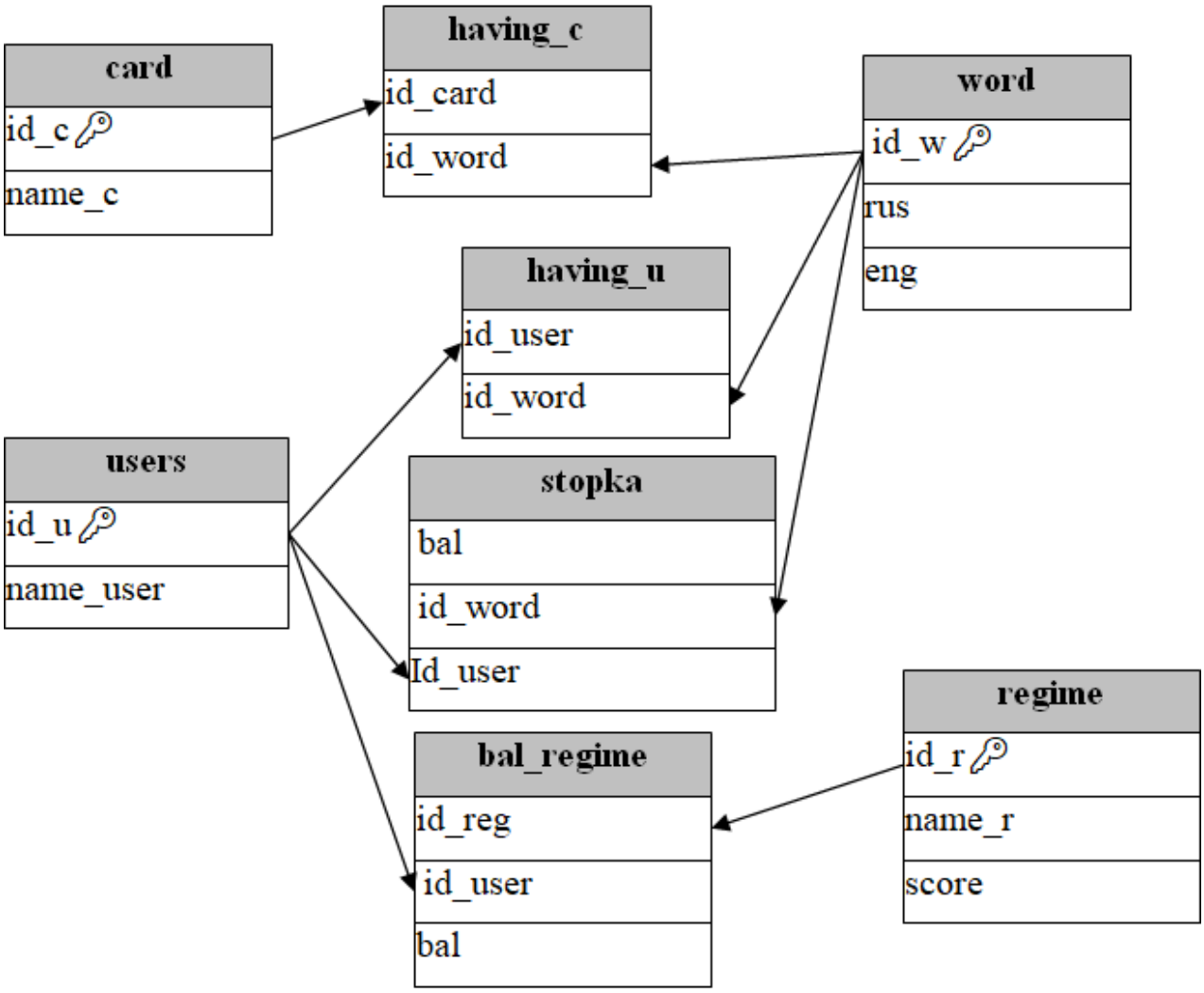

Рис. 2.11. Загальна схема бази даних

Заповнені таблиці даними зображені на рисунках 2.12-2.16.

name_c	id_c
Фрукты	1
Космос	2
Животные	3

Рис. 2.12. Таблиця «card»

eng	rus	id_w
Fruit	Фрукт	1
Apricot	Абрикос	2
Pineapple	Ананас	3
Banana	Банан	4
Bergamot	Бергамот	5
Grape	Виноград	6
Grapefruit	Грейпфрут	7
Pear	Груша	8
Melon	Дыня	9
Lemon	Лимон	10
Mandarin	Мандарин	11
Peach	Персик	12
Plum	Слива	13
Apple	Яблоко	14
Orange	Апельсин	15
Persimmon	Хурма	16
Pomegranate	Гранат	17
Watermelon	Арбуз	18
Kiwi	Киви	19
Fig	Инжир	20
Solar system	Солнечная система	21
Comet	Комета	22
Constellation	Созвездие	23
Planet	Планета	24
Universe	Вселенная	25
Nebula	Туманность	26
Booster	Ракетоноситель	27
Space probe	Космический зонд	28
UFO	НЛО	29
Telescope	Телескоп	30
Meteor	Метеор	31
Space shuttle	Космический корабль	32
Eclipse	Затмение	33
Orbit	Орбита	34
Rocket	Ракета	35
Dog	Собака	36
Cat	Кошка	37
Cow	Корова	38
Hen	Курица	39
Bear	Медведь	40
Horse	Лошадь	41
Pig	Свинья	42
Wolf	Волк	43
Lion	Лев	44
Elephant	Слон	45

Рис. 2.13. Таблиця «word»

id_card	id_word
1	1
1	2
1	3
1	4
1	5
1	6
1	7
1	8
1	9
1	10
1	11
1	12
1	13
1	14
1	15
1	16
1	17
1	18
1	19
1	20
2	21
2	22
2	23
2	24
2	25
2	26
2	27
2	28
2	29
2	30
2	31
2	32
2	33
2	34
2	35
3	36
3	37
3	38
3	39
3	40
3	41
3	42
3	43
3	44
3	45
3	36
3	36
3	36
3	46

Рис. 2.14. Таблиця «having_c»

id_u	name_user
8	Мирослав Тетерин
9	Милан Сафонов
10	Юрий Ковальчук
11	Татьяна Коломоец
12	Ева Рогова
13	Кристина Тимошенко
14	Виктория Рымар
15	Олеся Хитрук
16	Давид Стегайло
17	Регина Калинина
18	Ольга Жукова
19	Ростислав Сиверстов
20	Марк Палий

Рис. 2.15. Таблиця «users»

name_r	id_r	score
Сбор слова из заданных букв	1	10
Выбирание правильного перевода	2	7
Выбирание правильного перевода из 5 пар слов	3	5
Выделение слова	4	8
Выбирание правильный или неправильный перевод	5	5

Рис. 2.16. Таблиця «regime»

2.5.3. Проєктування інтерфейсу та опис складових частин програми

Схема взаємодії модулів програмного додатку наведена на рисунку 2.17.

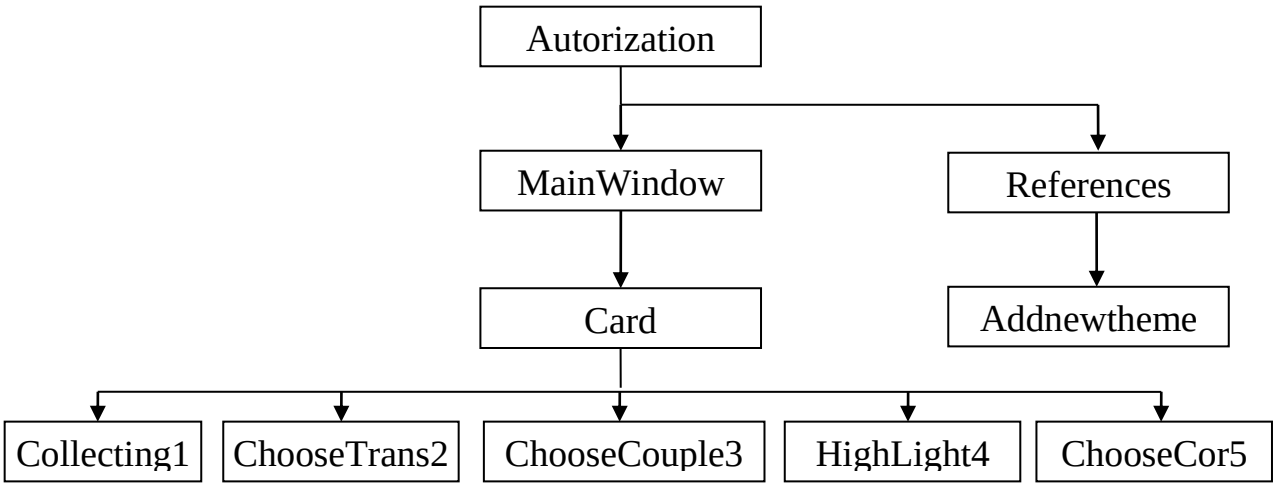

Рис. 2.17. Схема взаємодії модулів програмного додатку

Графічний інтерфейс забезпечує взаємодію користувача з програмним додатком.

Згідно з постановкою задачі було розроблено інтерфейс програми, який описано нижче.

Перша форма, яку зустрічає користувач – авторизація, яка зображена на рис. 2.18-2.19. На ній розміщено такі компоненти: background, choose, choose_school, enter, learning_english, parol:

– компонент background призначений виведення зображення на фон форми;

– компонент choose призначений для вибору актора;

– компонент choose_school призначений для вибору фамілії та ім’я учня;

– компонент enter призначений для відкриття інших форм;

– компонент learning_english призначений для розміщення логотипу;

– компонент parol призначений для введення паролю за актора «Вчитель».

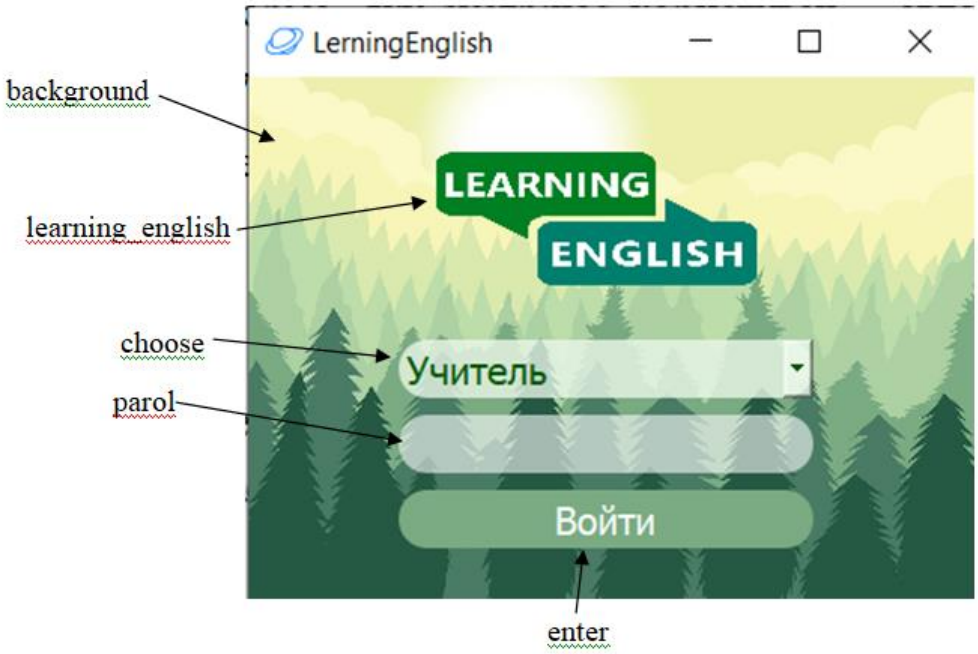

Рис. 2.18. Форма авторизації вчителя

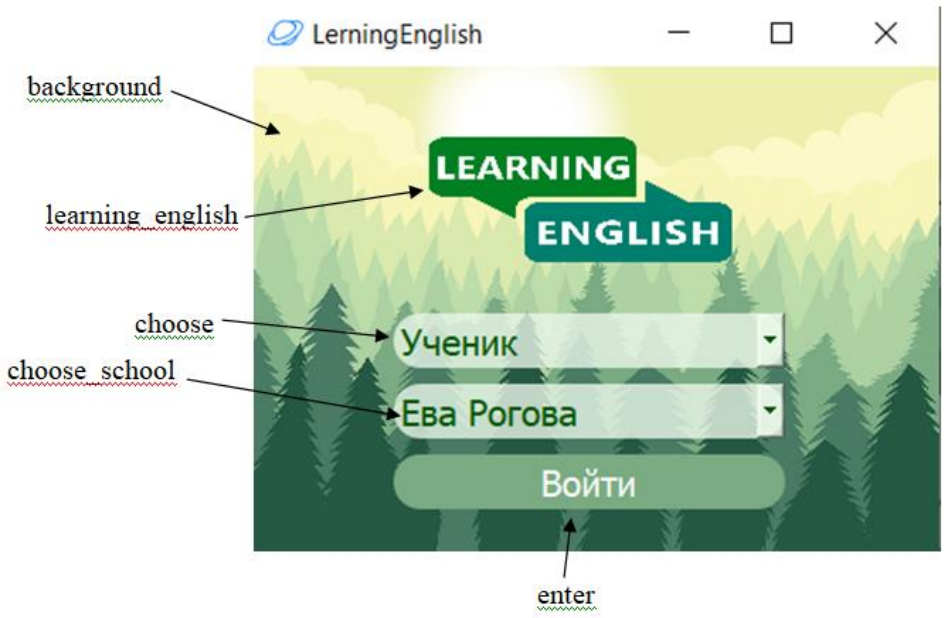

Рис. 2.19. Форма авторизації учня

На формі вибору теми MainWindow, що зображено на рис. 2.20 розміщено такі компоненти goout, w_theme, table_theme, text_theme:

– компонент goout призначений для виходу з програми;

– компонент widget_stat призначений для виведення графіку;

– компонент w_theme - віджет для розміщення на ньому компонентів;

– компонент table_theme призначений для виведення тем;

– компонент text_theme призначений для виведення рядка «Оберіть тему».

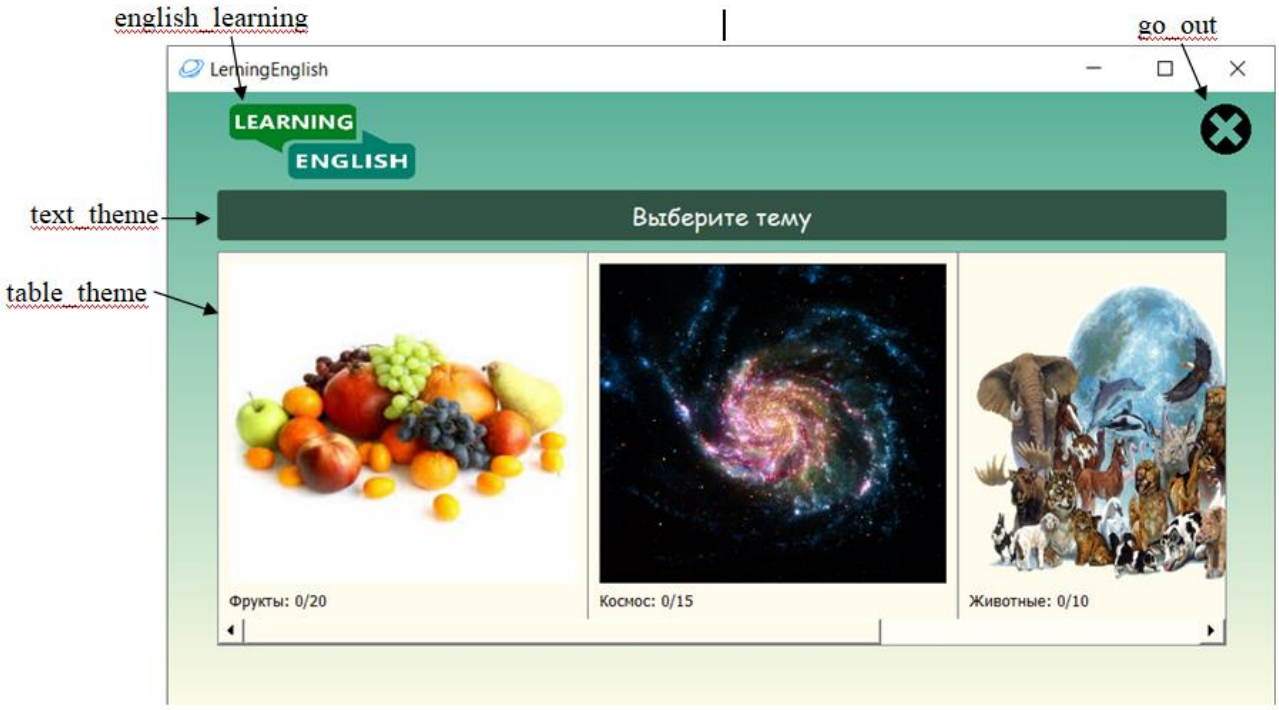

Рис. 2.20. Форма MainWindow

Форма Addnewtheme призначена для додавання нової теми, форма має наступні компоненти add_photo_theme, addtheme, delete_w, exit, frame, add_photo_word, addword, eng, iden_word, link_word, rus, text_eng, text_rus, iden_theme, link_theme, list_Widget, text_theme, theme. Форма представлена на рис. 2.21:

– компонент add_photo_theme призначений для додавання зображення до теми;

– компонент addtheme призначений для додавання теми;

– компонент delete_w призначений для видалення слова;

– компонент exit призначений для повернення до попередньої форми;

– компонент frame призначена для розміщення на ньому компонентів для додавання слова;

– компонент add_photo_word призначений для додавання зображення до слова;

– компонент addword призначена для додавання слова;

– компонент eng призначений для введення слова на англійській;

– компонент iden_word призначений для того, щоб показати наявність файлу по шляху, введеному в link_word;

– компонент link_word призначений для введення шляху до зображення до слова;

– компонент rus призначений для введення слова на російській;

– компонент text_eng призначений для виведення рядка «На англійській:»;

– компонент text_ rus призначений для виведення рядка «На російській:»;

– компонент iden_theme призначений для того, щоб показати наявність файлу по шляху, введеному в link_theme;

– компонент link_word призначений для введення шляху до зображення до теми;

– компонент list_Widget призначений для виведення слів, які були добавлені;

– компонент text_theme призначений для виведення рядка «Назва теми:»;

– компонент theme призначений для введення назви теми.

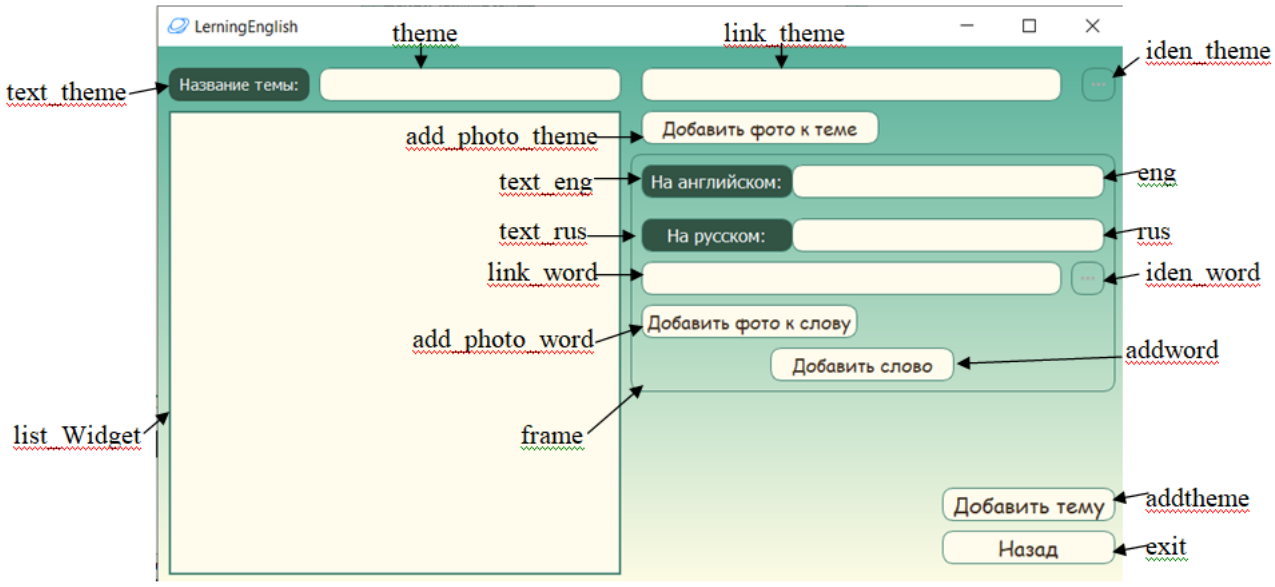

Рис. 2.21. Додавання теми, форма Addnewtheme

Форма Collecting1 призначена для першого режиму (збір слова із заданих букв), форма містить такі компоненти cor_photo, go_out, grid, progressBar, text_regime, vvod, widget_noph, back_noph, vuvod_noph, widget_ph, back_ph, vuvod_ph, word_photo. Форма представлена на рис. 2.22:

– компонент cor_photo призначений для того, щоб показати, чи було вірно пройдено режим;

– компонент go_out призначений для перевірки введеного слова та переходу далі;

– компонент grid призначений для виведення літер;

– компонент progressBar призначений для виведення прогресу;

– компонент text_regime призначений для виведення рядка «Зберіть слово із пропонованих літер»;

– компонент vvod призначений для виведення перекладу слова, яке потрібно ввести;

– компонент widget_noph призначений для виведення компонентів, якщо зображення до слова відсутнє;

– компонент back_noph призначений для видалення останньої літери з компонента vuvod_noph;

– компонент vuvod_noph призначений для введення слова, до якого немає зображення;

– компонент widget_ph призначений для виведення компонентів, якщо зображення до слова існує;

– компонент back_ph призначений для видалення останньої літери з компонента vuvod_ph;

– компонент vuvod_ph призначений для введення слова, до якого є зображення;

– компонент word_photo призначений для виведення зображення, якщо воно існує.

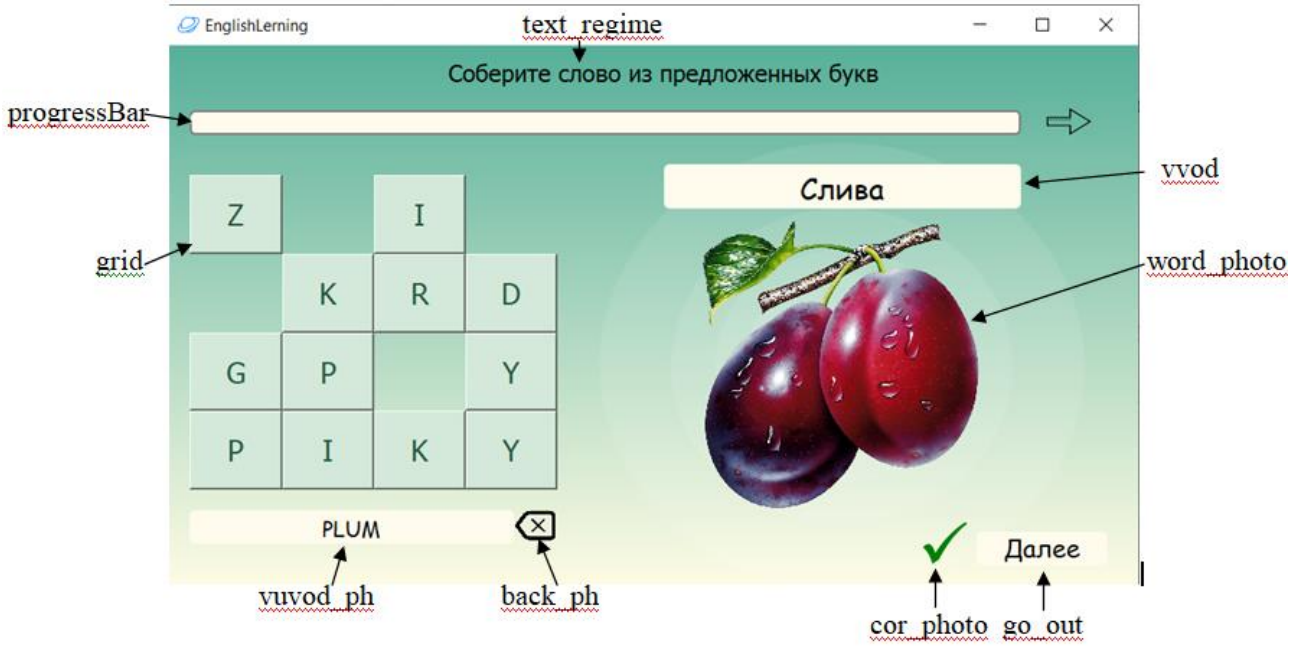

Рис. 2.22. Перший режим - збір слова із заданих букв, форма Collecting1

Форма Choosetrans2 призначена для другого режиму (обирання вірного перекладу), форма містить такі компоненти progressBar, text_regime, translate, widget_noph, b1, b2, b3, b4, b5, widget_ph, b1_ph, b2_ph, b3_ph, b4_ph, b5_ph, word_photo. Форма проставлена на рис. 2.23:

– компонент progressBar призначений для виведення прогресу;

– компонент text_regime призначений для виведення рядка «Оберіть вірний переклад»;

– компонент translate призначений для виведення перекладу слова, яке потрібно ввести;

– компонент widget_noph призначений для виведення компонентів, якщо зображення до слова відсутнє;

– компоненти b1, b2, b3, b4, b5 призначені для обирання користувачем перекладу, якщо зображення відсутнє;

– компонент widget_ph призначений для виведення компонентів, якщо зображення до слова існує;

– компоненти b1_ph, b2_ph, b3_ph, b4_ph, b5_ph призначені для обирання користувачем перекладу, якщо зображення існує;

– компонент word_photo призначений для виведення зображення, якщо воно існує.

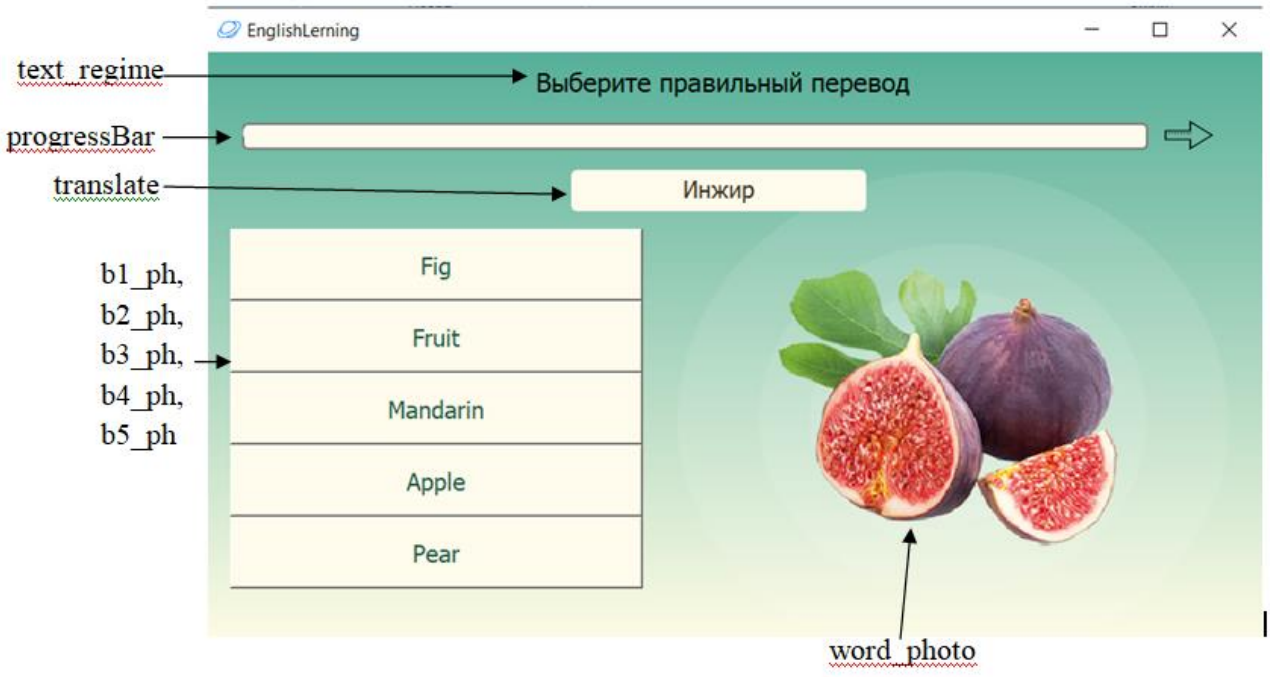

Рис. 2.23. Другий режим - обрання вірного перекладу, форма Choosetrans2

Форма Choosecouple3 призначена для третього режиму (обирання вірного перекладу із 5 пар слів), форма містить такі компоненти b1, b2, b3, b4, b5, b6, b7, b8, b9, b10, godali, progressBar, text_regime. Форма представлена на рис. 2.24:

– компонент progressBar призначений для виведення прогресу;

– компонент text_regime призначений для виведення рядка «Оберіть пари слів»;

– компонент godali призначений для того, щоб пропустити режим;

– компоненти b1, b2, b3, b4, b5, b6, b7, b8, b9, b10 призначені для обирання користувачем пар слів.

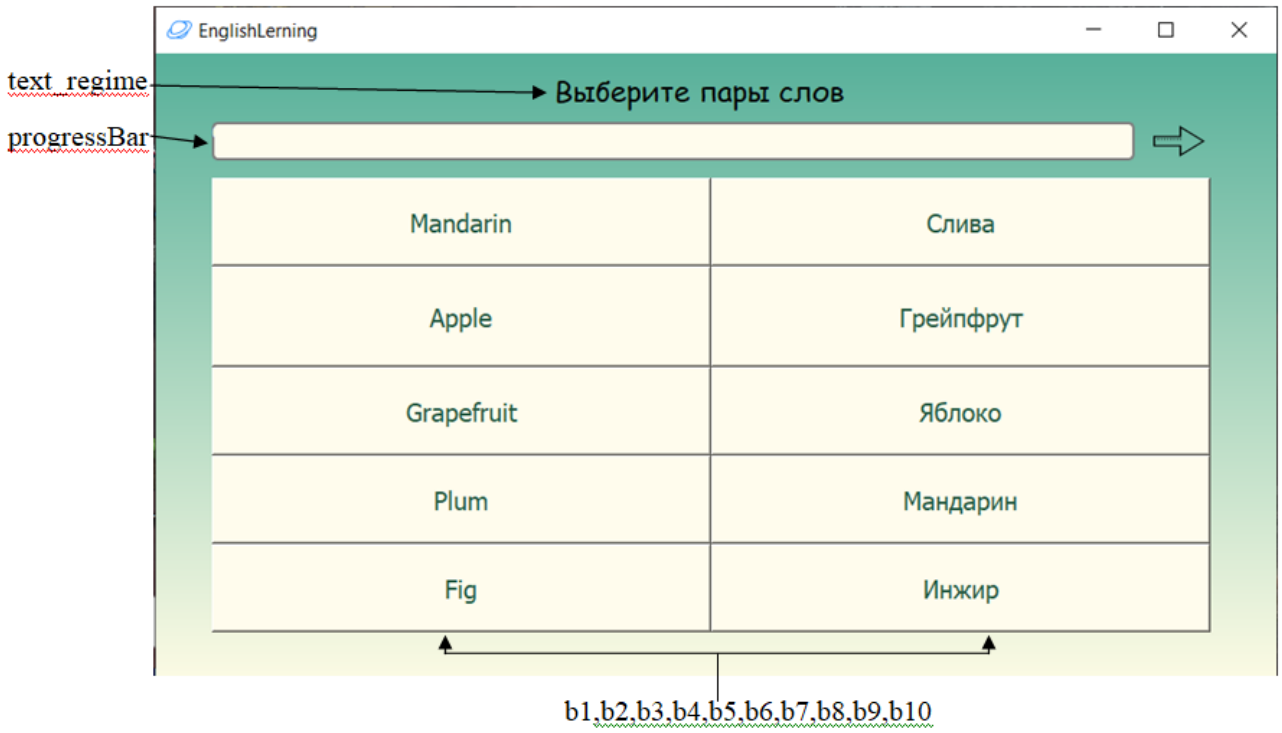

Рис. 2.24. Третій режим - обирання вірного перекладу із 5 пар слів, форма Choosecouple3

Форма Highlight4 призначена для четвертого режиму (виділення слова серед випадкової матриці букв), форма містить такі компоненти cor_photo, go_out, progressBar, text_regime, Grid, vvod, widget_noph, vuvod_noph, widget_ph, vuvod_ph, word_photo. Форма представлена на рис. 2.25.

– компонент progressBar призначений для виведення прогресу;

– компонент text_regime призначений для виведення рядка «Виділіть слово із запропонованих літер»;

– компонент go_out призначений для того, щоб пропустити режим;

– компонент Grid призначений для виведення літер;

– компонент vvod призначений для виведення перекладу слова, яке потрібно ввести;

– компонент widget_noph призначений для виведення компонентів, якщо зображення до слова відсутнє;

– компонент vuvod_noph призначений для введення слова, до якого немає зображення;

– компонент widget_ph призначений для виведення компонентів, якщо зображення до слова існує;

– компонент vuvod_ph призначений для введення слова, до якого є зображення:

– компонент word_photo призначений для виведення зображення, якщо воно існує.

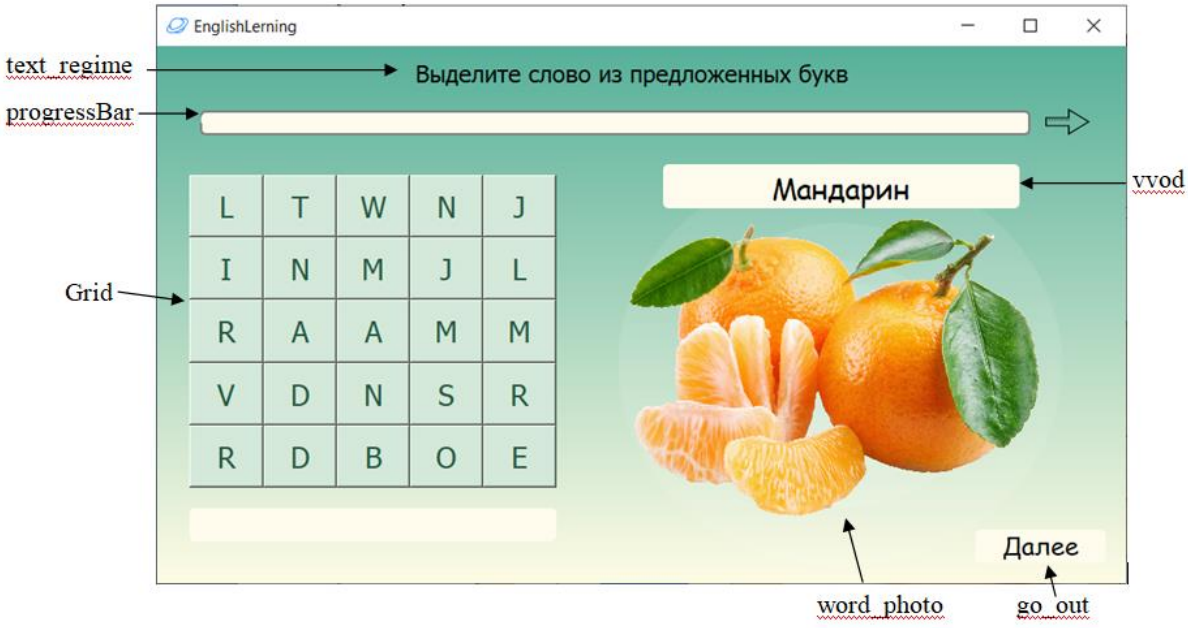

Рис. 2.25. Четвертий режим - виділення слова серед випадкової матриці букв, форма Highlight4

Форма Choosecor5 призначена для п’ятого режиму (обирання «правильний» або «неправильний» переклад слова), форма містить такі компоненти cor, notcor, progressBar, text_regime, transl, word. Форма представлена на рис. 2.26:

– компонент progressBar призначений для виведення прогресу;

– компонент text_regime призначений для виведення рядка «Оберіть, це вірний або не вірний варіант»;

– компонент cor призначений для того, щоб користувач міг обрати варіант «Вірно»;

– компонент notcor призначений для того, щоб користувач міг обрати варіант «Не вірно»;

– компонент transl призначений для виведення слова на російській;

– компонент word призначений виведення слова на англійській.

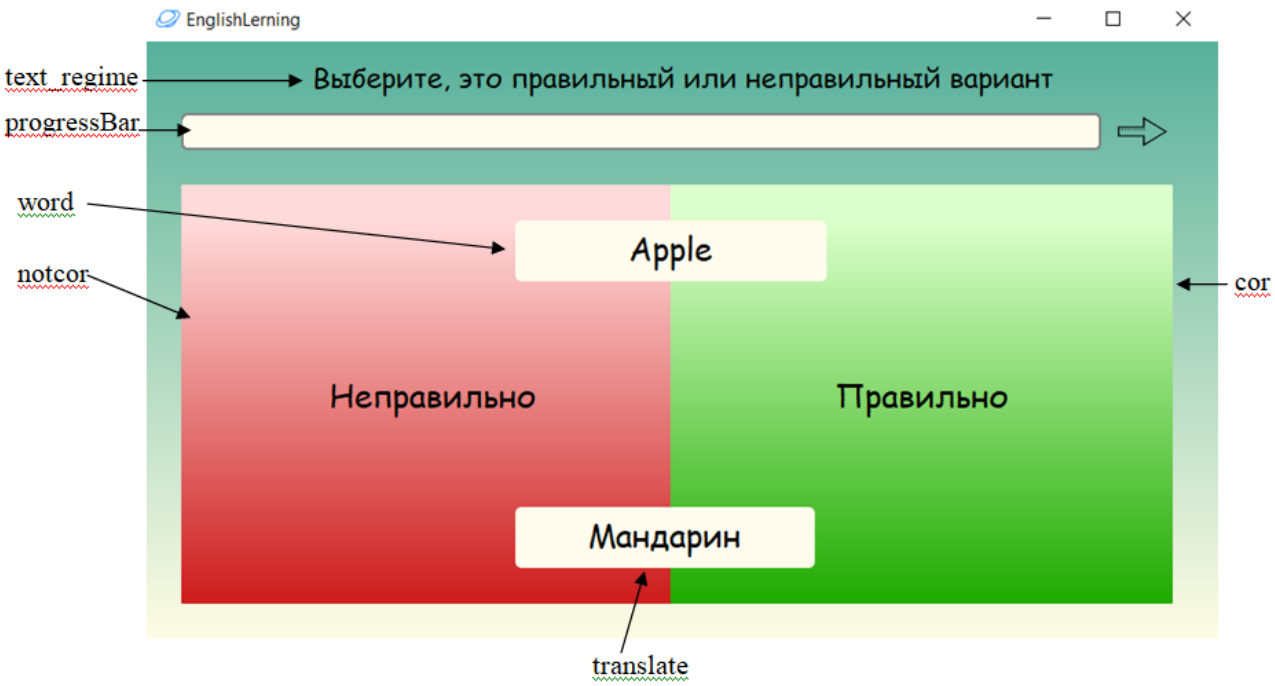

Рис. 2.26. П’ятий режим - обирання «правильний» або «неправильний» переклад слова, форма Choosecor5

Форма refences призначена для налаштування, форма містить такі компоненти: list, sound, add, add_s, delete_student, delete_theme, input_student, save, speed, speed_text, student, themes, volume, volume_text, stat, text. Форма преставлена на рис. 2.27-2.28:

– компонент list – це вкладка «Процес вивчення»;

– компонент sound – це вкладка «Звук та теми»;

– компонент add призначений для переходу на форму додавання теми;

– компонент add_s призначений для додавання нового учня;

– компонент delete_student призначений для видалення учня;

– компонент delete_theme призначений для видалення теми;

– компонент input_student призначений для додавання учня;

– компонент save призначений для збереження налаштувань звуку;

– компонент speed призначений для налаштування параметра «Швидкість програвання голосу»;

– компонент speed_text призначений для виведення рядка «Швидкість програвання голосу»;

– компонент student призначений для виведення списку учнів;

– компонент themes призначений для виведення списку тем;

– компонент volume призначений для налаштування параметра «Гучність»;

– компонент volume_text призначений для виведення рядка «Гучність»;

– компонент stat призначений виведення успішності учнів;

– компонент text призначений виведення рядка «Успіхи учнів».

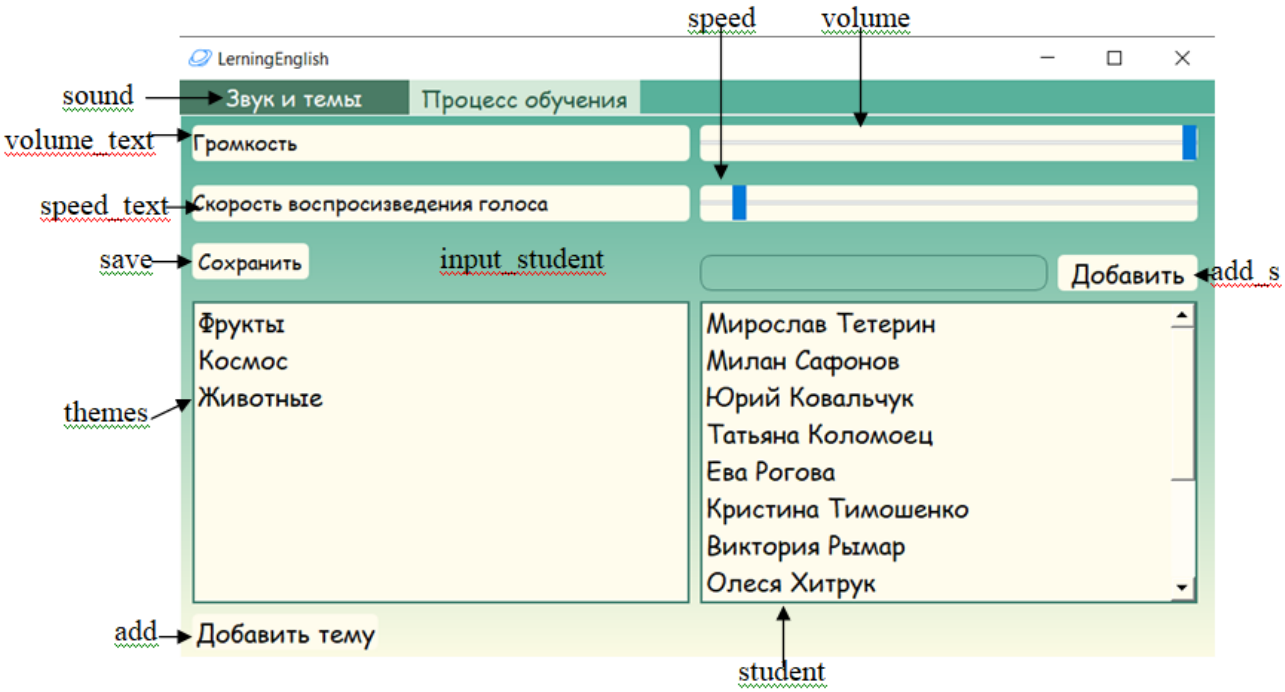

Рис. 2.27. Налаштування, форма References (вкладка «Звук та теми»)

Рис. 2.28. Налаштування, форма References (вкладка «Процес вивчення»)

Форма Card призначена для вивчення учнем нових слів, форма містить такі компоненти: back, before, current, eng, go_test, image, listen, next, rus. Форма преставлена на рис. 2.29:

– компонент back призначений для повернення на попередню форму;

– компонент before призначений для показу попереднього слова;

– компонент current призначений для виведення поточного номеру слова;

– компонент eng призначений для виведення слова на англійській;

– компонент go_test призначений для переходу на форму тестування;

– компонент image призначений для виведення зображення;

– компонент listen призначений для програвання слова голосом;

– компонент next призначений для показу наступного слова;

– компонент rus призначений для виведення слова на російській.

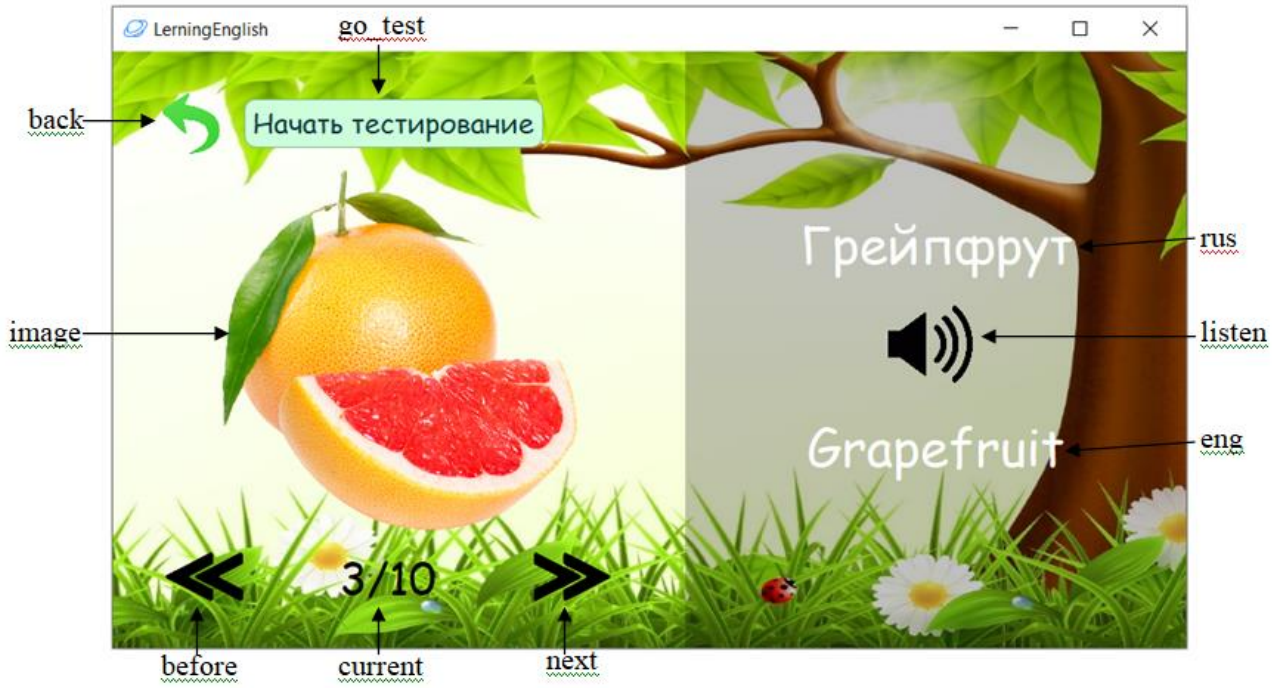

Рис. 2.29. Форма Card

2.5.4. Опис основних алгоритмів програми

Підключення до бази даних представлене на наступному лістингу:

```
QSqlDatabase db =  QSqlDatabase::addDatabase("QODBC");
db.setDatabaseName("kurs");
db.setHostName("localhost");
db.setPort(3306);
db.setUserName("root");
 db.setPassword("1234");
```

Якщо база даних за яких-небудь причин не відкрилась, то програма виведе відповідне повідомлення. Приклади представлені на рис. 2.30 на в лістингу:

```
if(!db.open()) messange("База данных не подкючена!");
```

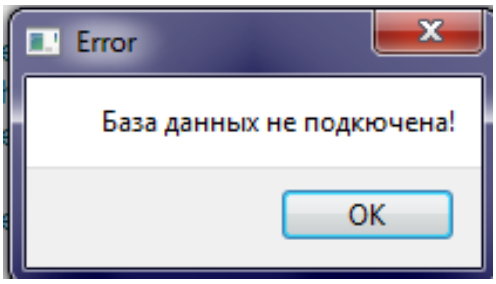

Рис. 2.30. Повідомлення

В четвертому режимі користувач може натиснути лише на визначені букви завдяки функції на лістингу, що забезпечує унеможливлення натиснення на неправильні кнопки:

```
void HighLight4::signalbutton_wp()
{
int index=ui->Grid->indexOf(static_cast<QPushButton*>(sender()));
   if(static_cast<QPushButton*>(sender())->styleSheet()!="background-color:
#0055ff; font: 12pt")
{
```

```
      if(touch[position-1]-1==index || touch[position-1]+1==index || touch[position-
1]-5==index || touch[position-1]+5==index || position==0)
  {
      QString word = static_cast<QPushButton*>(sender())->text();
      ui->vuvod_ph->setText(ui->vuvod_ph->toPlainText()+word);
       ui->vuvod_ph->setAlignment(Qt::AlignCenter);
      static_cast<QPushButton*>(sender())->setStyleSheet("background-color:
#0055ff; font: 12pt");
       touch[position]=index;
       position++;
       }
  }
    else
{
 if(touch[position-1]==index)
{
    QString str =ui->vuvod_ph->toPlainText();
    str.chop(1);
    ui->vuvod_ph->setText(str);
    ui->vuvod_ph->setAlignment(Qt::AlignCenter);
    static_cast<QPushButton*>(sender())->setStyleSheet("background-color: rgb(255,
252, 237); font: 12pt");
    position--;
       }
   }
}
```

2.6. Обґрунтування та організація вхідних та вихідних даних програми

Для реалізації програмного додатку була створена база даних, що містить шість таблиць та містить в собі наступні вхідні дані програми:

1. Дані про слова:
   - слово на англійській;
   - слово на російській;
   - зображення до слова.
2. Дані про тему:
   - назва теми;
   - зображення до теми.

Дані про користувачів: ім’я та прізвище.

На основі вхідних даних формується вихідний потік даних, що забезпечує відображення відповідних форм за запитами користувача, зображення слів для вивчення, кількість отриманих балів.

2.7. Опис роботи розробленого програмного продукту

2.7.1. Використані технічні засоби

Для коректної роботи програми рекомендовані наступні системні вимоги:

– 1 GB RAM (рекомендується 1.5 GB +);

– 1-2 GB вільного простору на жорсткому диску;

– стандартний GPU з підтримкою DirectX 9.0 та вище;

– роздільна здатність екрану 1024x768 та вище;

– Intel® Pentium® або сумісний, мінімум 1.4 GHz (рекомендується 2.2GHz +);

– переферійні пристрої введення-виведення: клавіатура, миша.

2.7.2. Використані програмні засоби

Дане програмне забезпечення розроблене з використанням наступних програмно-апаратних засобів:

– операційна система Microsoft Windows XP/Vista/7;

– мови програмування C++ у середовищі QTCreator ;

– СУБД MySQL..

2.7.3. Виклик та завантаження програми

Для установки програмного додатку потрібно папку English скопіювати на потрібний комп’ютер і відкрити файл English.exe.

Вхід в систему виконується після завантаження додатку.

2.7.4. Опис інтерфейсу користувача

Після запуску програми з’явиться вікно авторизації, що зображено на рис. 2.31- 2.32.

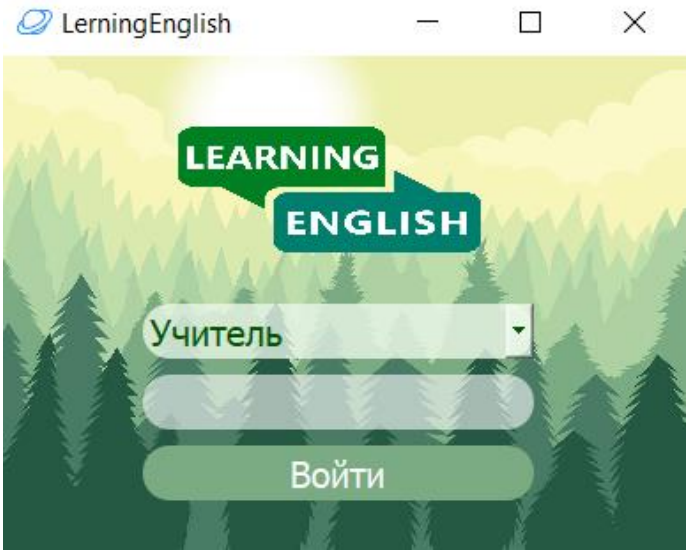

Рис. 2.31. Вікно авторизації (учитель)

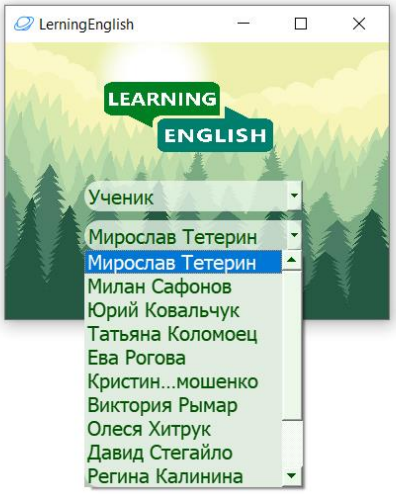

Рис. 2.32. Вікно авторизації (учень)

Якщо користувач обирає актора «Учитель», то йому, після введення пароля, виводиться форма налаштування, що зображено на рис. 2.33 – 2. 34. На першій вкладці «Звук та теми» можна зробити налаштування звуку: гучність та швидкість програвання голосу. Можна додати нового учня, ввівши його ім'я та прізвище в рядок вводу та натиснувши кнопку «Додати».

На другій вкладці «Процес вивчення» виведено таблицю зі списком учнів та їх успіхами у кожній темі.

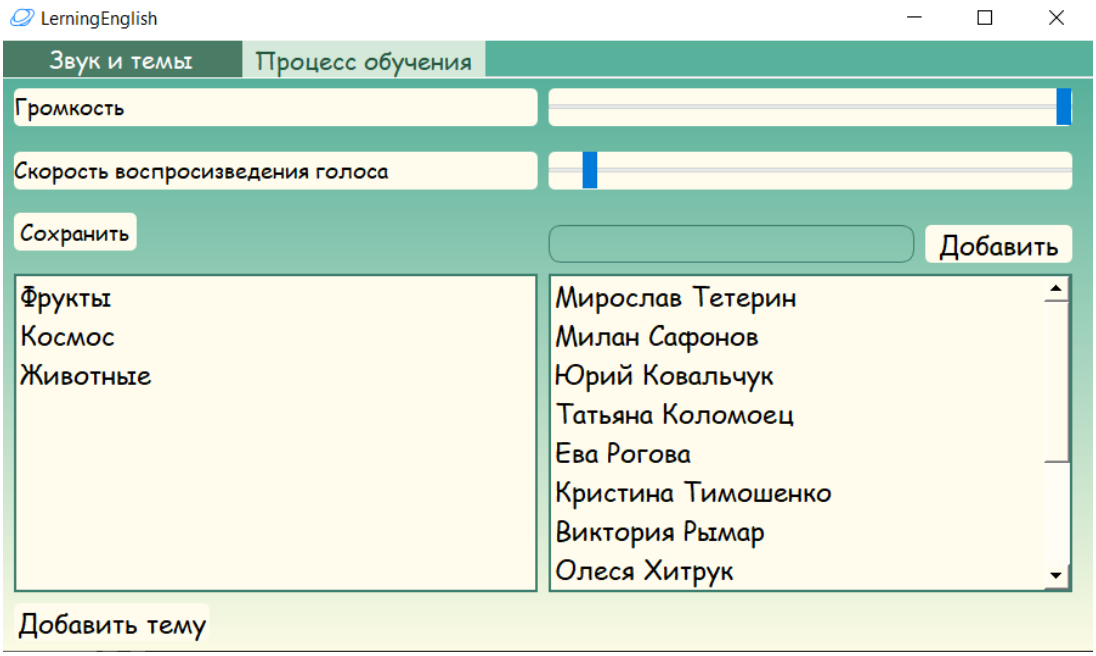

Рис. 2.33. Налаштування «Звуки та теми»

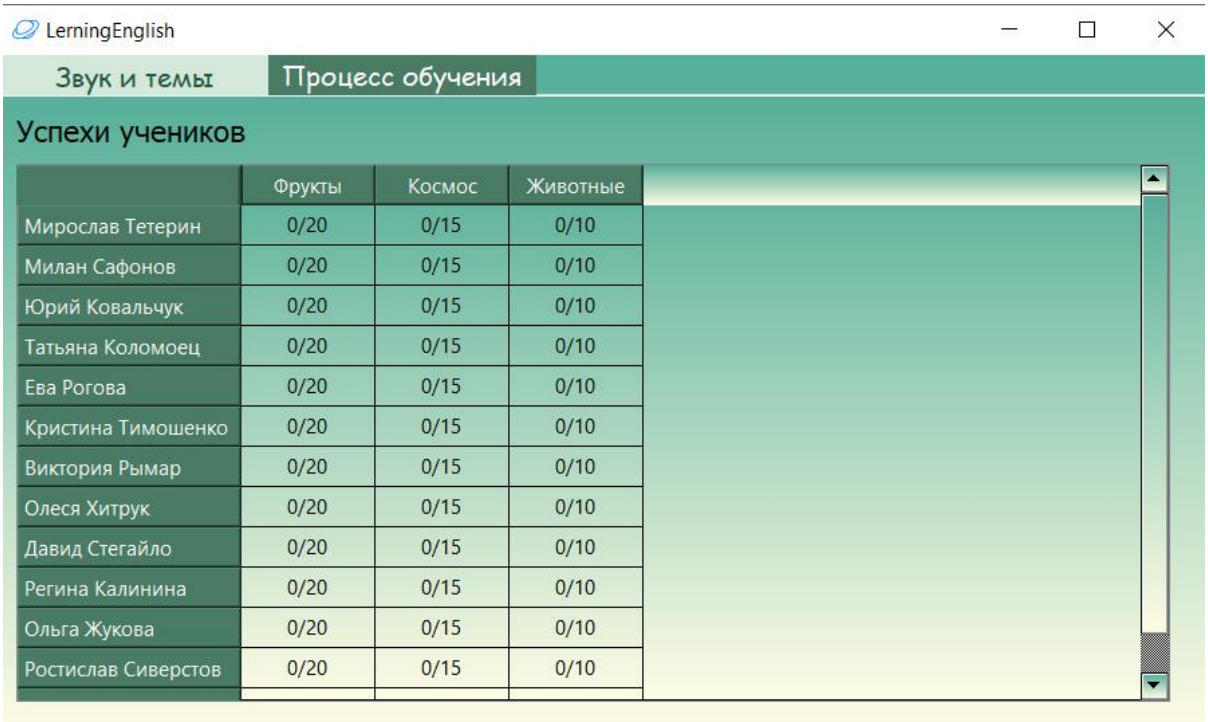

Рис. 2.34. Налаштування «Процес вивчення»

Якщо користувач натисне кнопку «Додати тему», то відкриється вікно додавання теми, яке можна побачити на рисунку 2. 35.

Рис. 2. 35. Додавання теми

Щоб додати слово, потрібно ввести його на англійській та переклад на російській. За бажанням можна додати зображення до слова, для цього потрібно скопіювати шлях до файлу та його ім’я (наприклад, «C:\Users\Meshtli\Desktop\hi\image.jpg») та натиснути на кнопку «Додати фото до слова». Після цього з’явиться відповідне зображення в формі зліва, це представлене на рис. 2.36 – 2.37.

Аналогічно діє додавання зображення до теми.

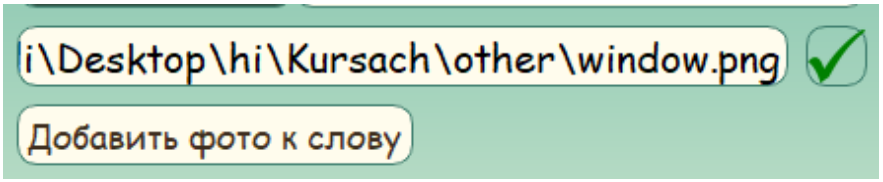

Рис. 2.36. Шлях до файлу існує

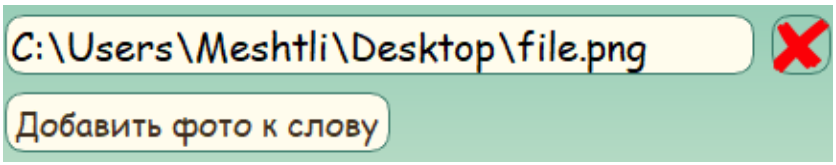

Рис. 2.37. Шлях до файлу не існує

Після натискання на «Додати слово», якщо рядки «на англійській» та «на російській» не порожні, слово переміщується в зону зліва. Додавання слів представлене на рис. 2.38. Якщо хоча б один з рядків порожній, то він на короткий час стає червоним, слово не додається, такий випадок представлений на рис. 2.39.

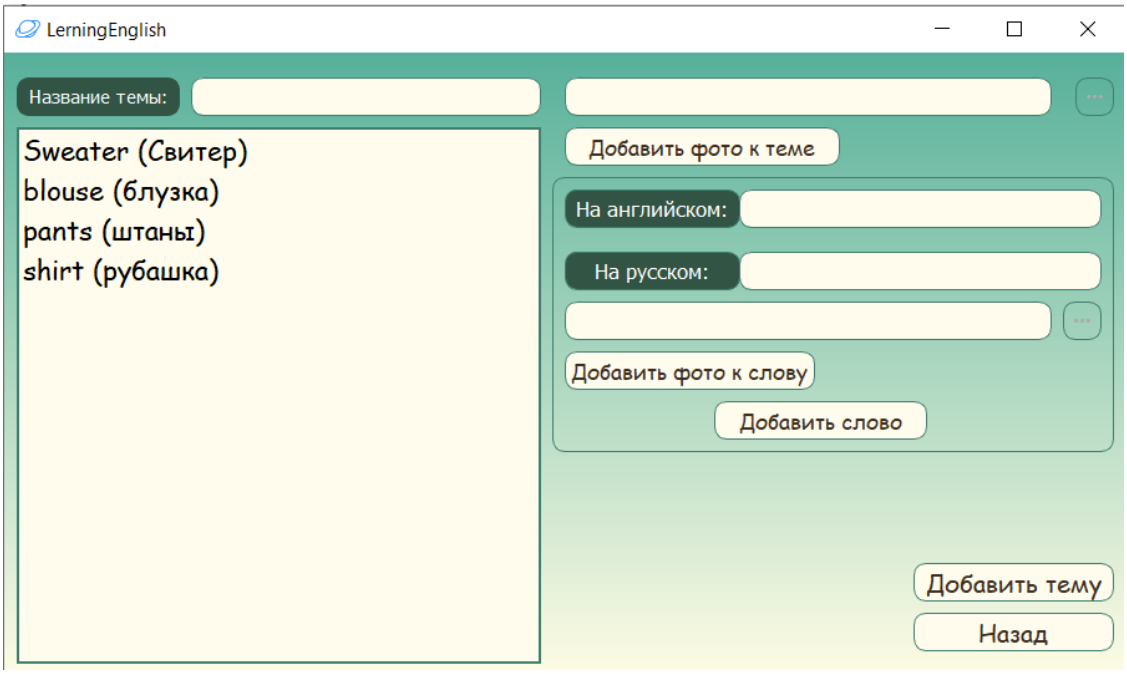

Рис. 2.38. Додані слова

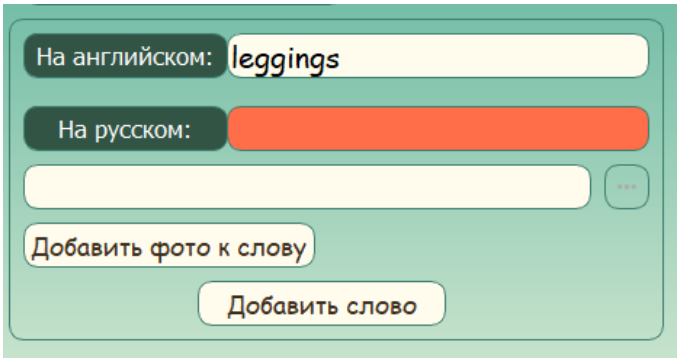

Рис. 2.39. Рядок з перекладом на російську порожній

Якщо натиснути додати тему, коли рядок з її назвою порожній, він стане червоним, це можна побачити на рис. 2.40. Якщо натиснути на ту ж кнопку, коли не додано жодного слова, червоною стане зона зі словами, це можна побачити на рисунку 2.41.

Рис. 2.40. Рядок з назвою теми порожній

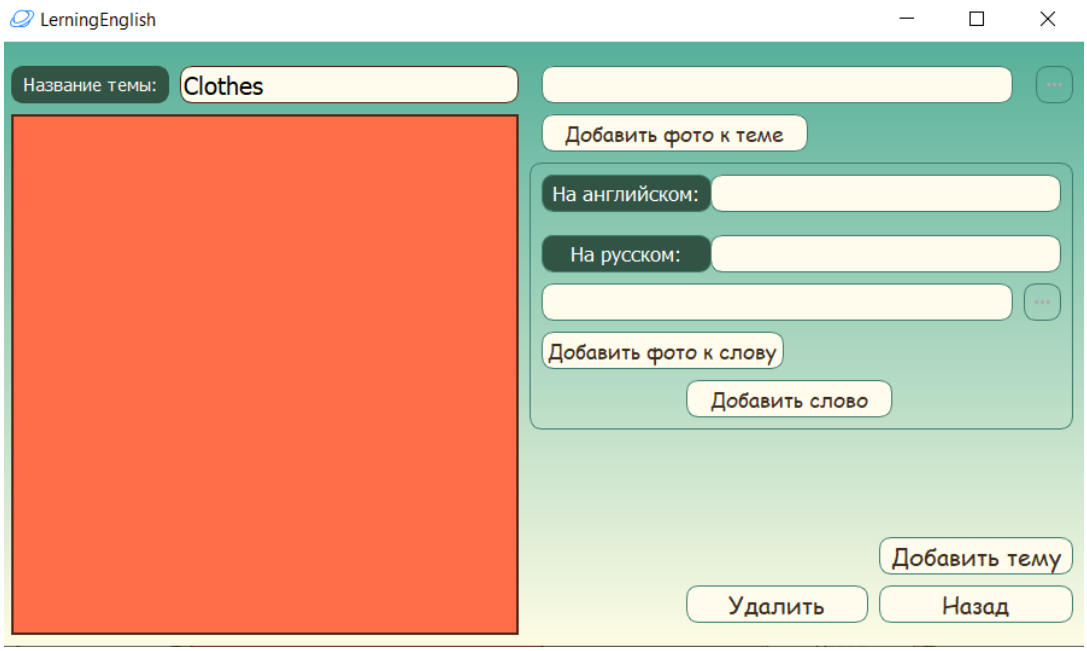

Рис. 2.41. Не додано жодного слова

Якщо користувач обирає актора «Учень», то йому потрібно обрати своє ім’я та прізвище. Далі відкривається вікно вибору теми, яке можна подивитись на рис. 2.42.

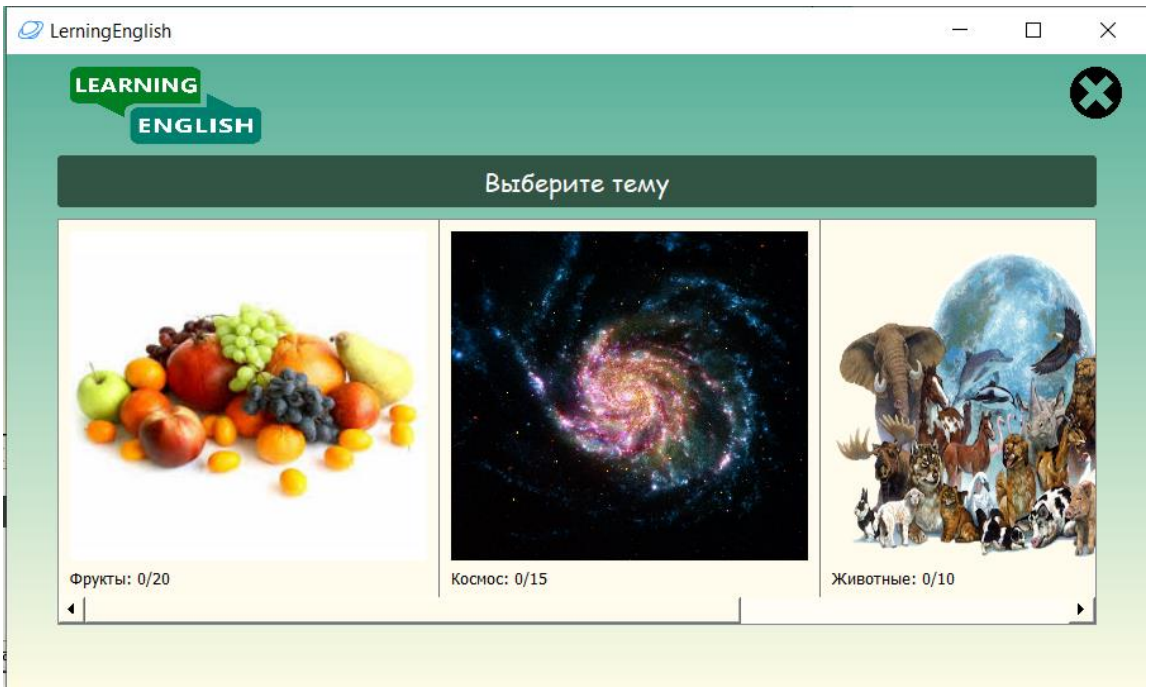

Рис. 2.42. Обирання теми

Для початку роботи потрібно обрати бажану тему та натиснути на її зображення. Відкриється вікно, за допомогою якого зручно вивчати переклад слова, це зображено на рис. 2.43. Учневі дається на вивчення 10 випадкових слів обраної теми, після цього, якщо вони успішно вивчені, програма готує ще 10, доки не буде вивчена уся тема

.

Рис. 2.43. Вивчення слова

Коли усі слова з 10 вивчені, можна переходити до тестування, натиснувши кнопку «Розпочати тестування», відобразиться випадково вікно одного з п’яти режимів. Якщо тему було обрано невірно, можна повернутись назад, настигнувши на зелену стрілку.

При запуску першого режиму, вам потрібно з поданих зліва літер зібрати слово на англійській. Вікно представлене на рис. 2.44.

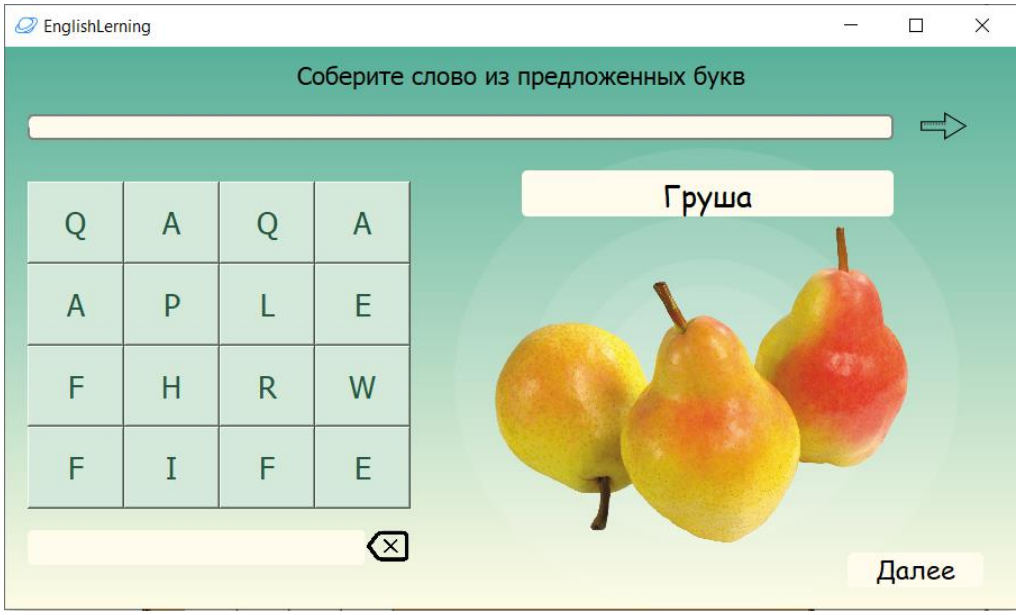

Рис. 2.44. Перший режим

Якщо введено слово вірно, то приблизно на одну секунду з’являється зображення біля кнопки далі, що підтверджує його правильність. Вікно з вірним результатом представлено на рис. 2.45. Якщо введено не вірно, то з’являється інше відповідне зображення. Вікно з не вірним результатом представлено на рис. 2.46.

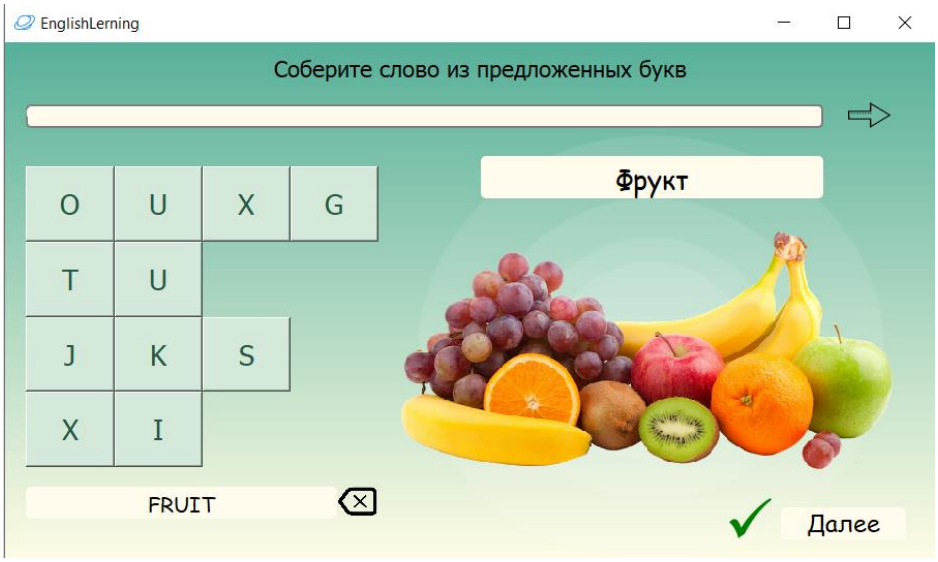

Рис. 2.45. Перший режим з вірно введеним словом

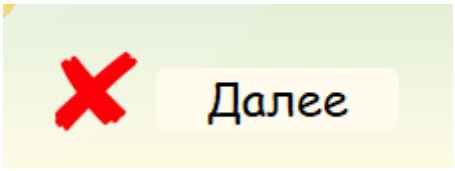

Рис. 2.46. Перший режим з невірно введеним словом

При запуску другого режиму потрібно обрати переклад слова на російській мові. Вікно представлене на рис. 2.47.

Рис. 2.47. Другий режим

Якщо слово обрано вірно, то приблизно на одну секунду колір кнопки зі словом на англійській змінюється на зелений. А якщо обрано не вірно, то — на червоний. Вікна з вірним та не вірним результатом представлені на рис. 2. 48-2.49.

Рис. 2.48. Другий режим з вірно обраним словом

Рис. 2.49 Другий режим з невірно обраним словом

При запуску третього режиму потрібно обрати відповідність між англійськими словами та їх перекладами. Вікно представлено на рисунку 2.50.

Рис. 2.50. Третій режим

При натисканні на одне з десяти слів, воно змінює колір на блакитний. Якщо натиснути ще раз на те ж саме слово, то колір зміниться назад. Вікно зі зміною кольору на блакитний представлено на рисунку 2.51.

Рис. 2.51. Третій режим - зміна кольору на блакитний

Після обирання другого слова з іншого стовпця (обрати слово з того ж неможливо), якщо пара була обрана вірно, то вона зникає. Вікно із правильним вибором пари представлено на рис. 2.52.

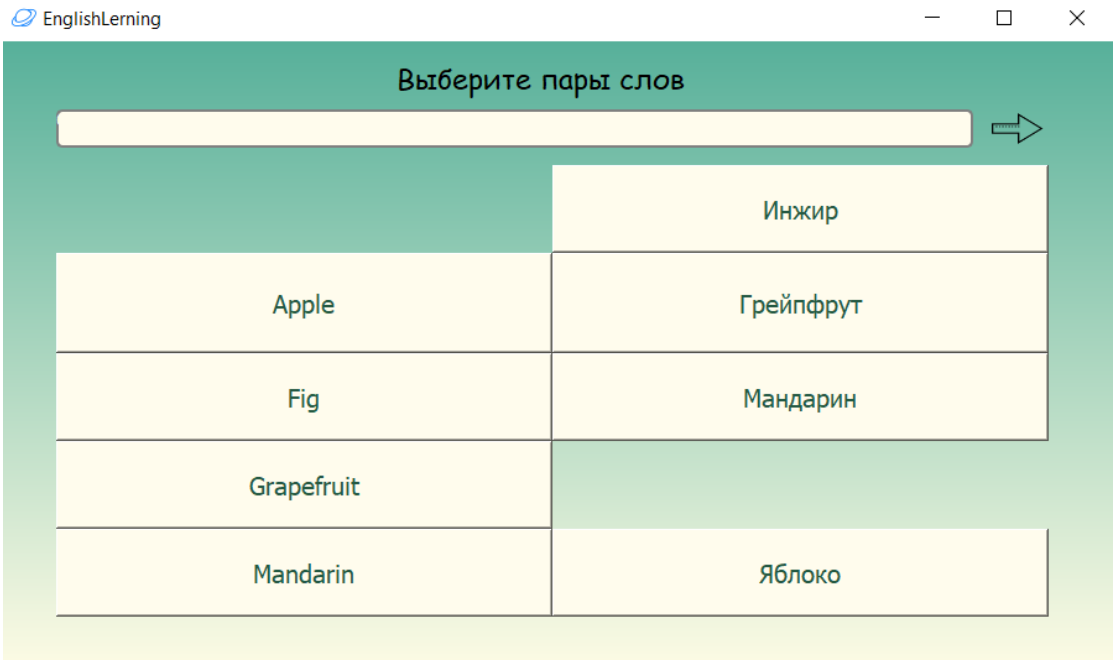

Рис. 2.52. Третій режим - правильний вибір пари

Якщо пара обрана неправильно, то обидва слова змінюють колір на червоний. Вікно із неправильним вибором пари представлено на рис. 2.53.

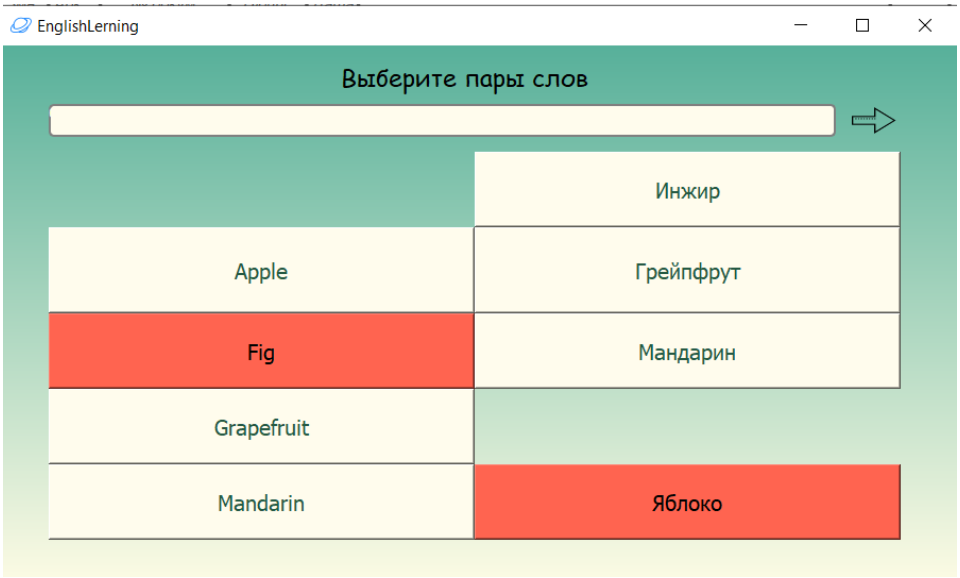

Рис. 2.53. Третій режим - не правильний вибір пари

Після обирання всіх десяти слів, вікно ненадовго затримується, щоб можна було побачити результат останньої пари слів, та переходить далі.

При запуску четвертого режиму потрібно виділити по літерам переклад слова. Вікно представлене на рис. 2.54.

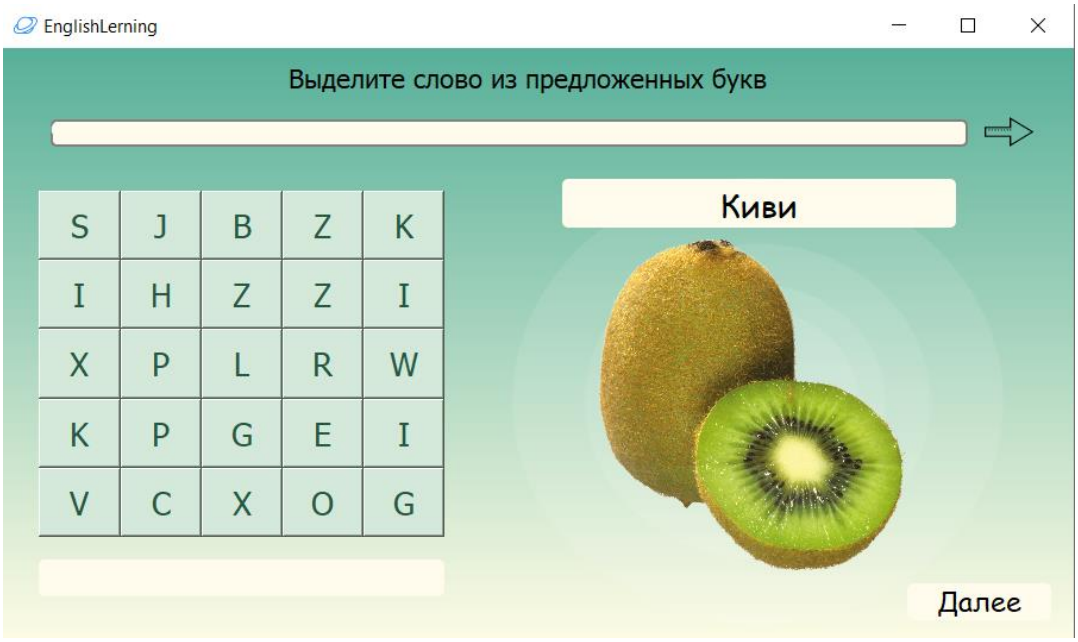

Рис. 2.54. Четвертий режим

При натисканні на літеру, вона змінює колір на темно-зелений. Якщо виділено слово вірно, то приблизно на одну секунду з’являється зображення біля кнопки далі, що підтверджує його правильність. Вікно з вірним результатом представлено на рис. 2.55. Якщо введено не вірно, то з’являється інше відповідне зображення. Вікно з не вірним результатом представлено на рисунку 2.56.

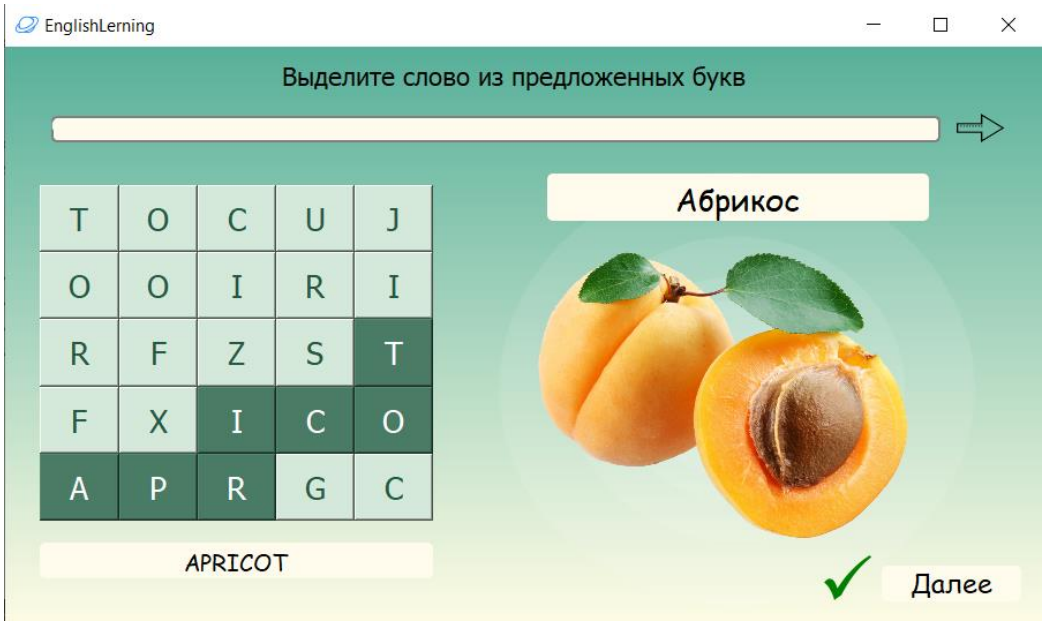

Рис. 2.55. Четвертий режим з вірно введеним словом

Рис. 2.56. Четвертий режим з невірно введеним словом

При запуску п’ятого режиму потрібно натиснути, чи співпадають слово на англійській та його переклад. Вікно п’ятого режиму представлено на рис. 2.57.

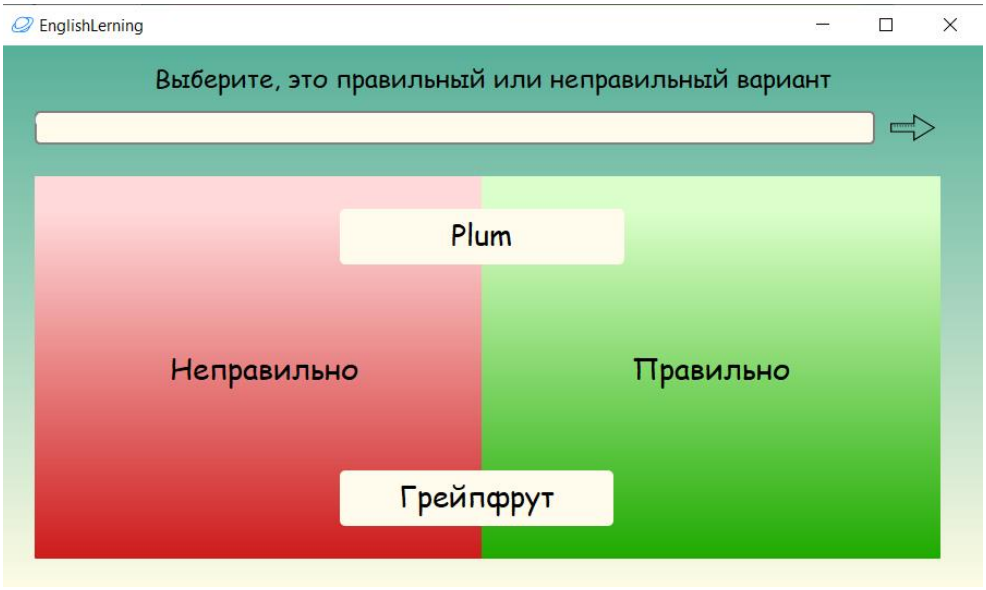

Рис. 2.57. П’ятий режим

Якщо обрано «Вірно» чи «Невірно» у відповідності з вірною відповіддю, то ненадовно поля зі словом та перекладом змінюють колір на зелений і після цього відбувається перехід далі, а якщо обрано неправильно, то на червоний. Вікно з вірним результатом представлено на рис. 2.58. Якщо введено не вірно, то з’являється інше відповідне зображення. Вікно з не вірним результатом представлено на рис. 2.59.

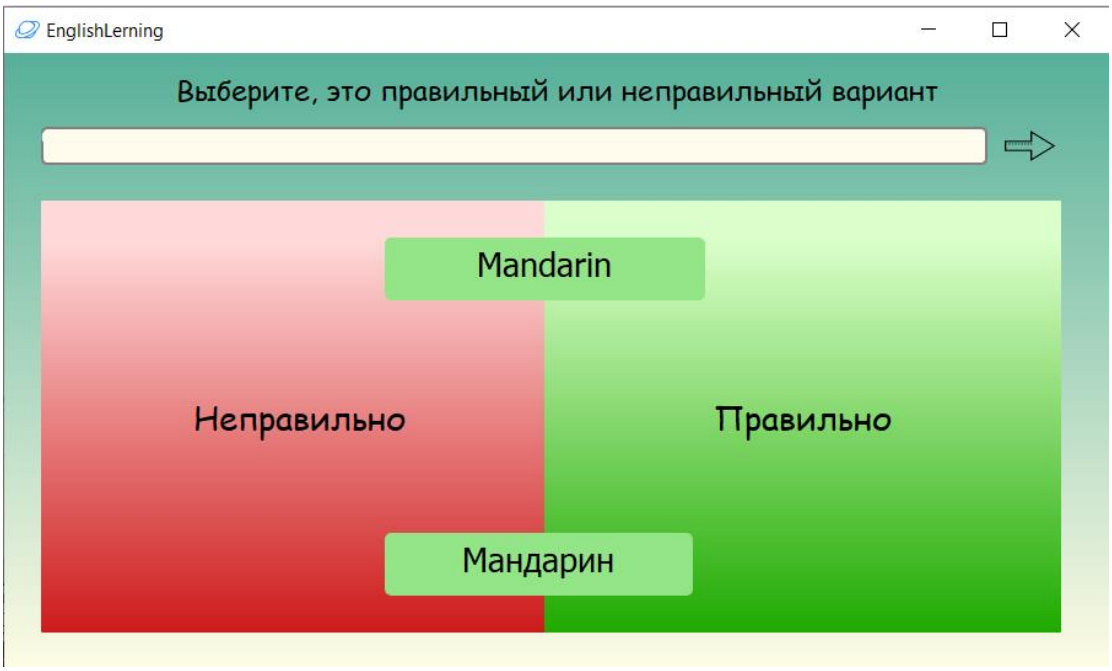

Рис. 2.58. П’ятий режим з вірною відповіддю

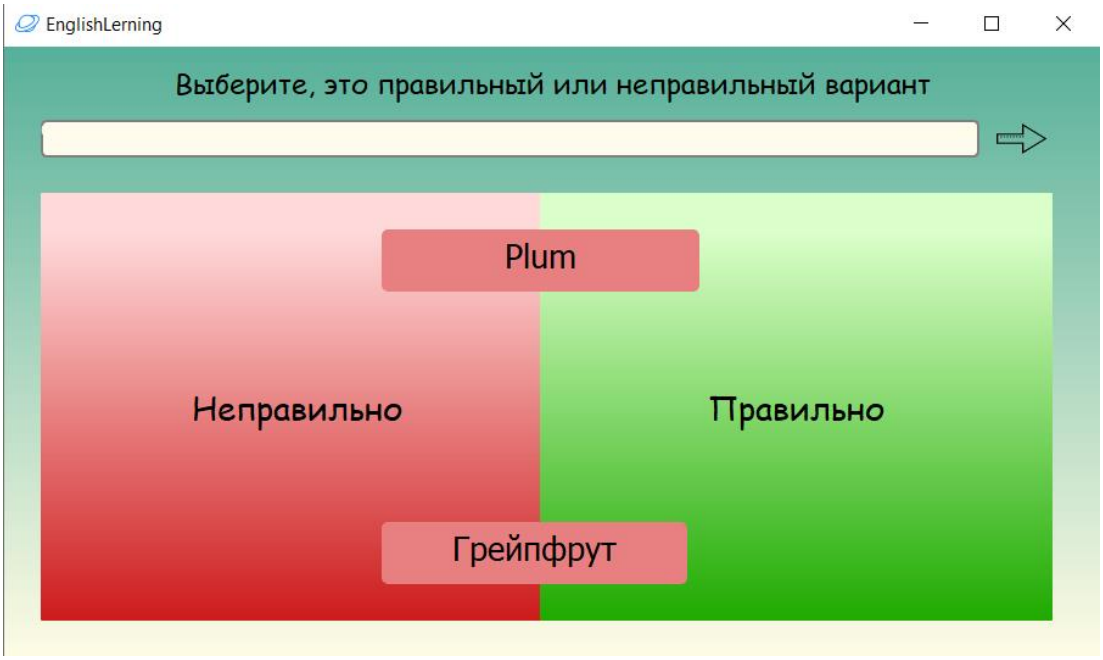

Рис. 2.59. П’ятий режим з невірною відповіддю

2.7.5. Тестування програмного продукту

Для тестування програмного продукту було обрано метод «чорного ящику» (техніка тестування, заснована на роботі виключно з зовнішніми інтерфейсами тестованої системи) з таких причин:

– тестування проводиться з позиції кінцевого користувача і може допомогти виявити неточності і протиріччя в специфікації;

– тестувальнику немає необхідності знати мови програмування і заглиблюватися в особливості реалізації програми, що допомагає уникнути упередженого ставлення;

– можна починати писати тест-кейси, як тільки готова специфікація.

Метою цієї техніки є пошук помилок в таких категоріях:

– неправильно реалізовані або відсутні функції;

– помилки інтерфейсу;

– помилки в структурах даних або організації доступу до зовнішніх баз даних;

– помилки поведінки або недостатня продуктивність системи.

Техніки тестування «чорним ящиком»:

– еквівалентне розбиття: ця техніка включає в себе поділ вхідних значень на допустимі і неприпустимі розділи і вибір репрезентативних значень з кожного розділу в якості тестових даних. Вона може бути використана для зменшення кількості тестових випадків;

– аналіз граничних значень: техніка, яка включає в себе визначення меж вхідних значень і вибір в якості тестових даних значень, що знаходяться на кордонах, всередині і поза межами. Багато систем мають тенденцію вести себе некоректно при граничних значеннях, тому оцінка значень меж додатка дуже важлива. При перевірці ми беремо наступні величини: мінімум, (мінімум-1), максимум, (максимум + 1), стандартні значення;

– тестування таблиці переходів: при цій техніці сценарії тестування вибираються на основі виконання коректних і некоректних переходів станів;

– тестування за сценаріями використання: ця техніка використовується при написанні тестів для індивідуального сценарію користувача з метою перевірки його роботи.

Контроль на коректне введення, стабільне та нормалізоване відображення даних, оптимізований пошук даних, стабільна взаємодія компонентів - це все можна описати як надійність програмного продукту, якій необхідно приділяти багато часу, бо від результату залежить комфортність використання даної програми.

У даному розділі описано аналіз можливих ситуацій, що були проаналізовані при проєктуванні додатку, тобто ситуації, що можуть призвести до виникнення помилок.

В коді програми прописані унеможливення введення некоректних даних під час додавання нової теми. Приклади представлені в наступних лістингах:

1. Унеможливлення введення символів або літер іншої мови під час введення слова на англійській мові:

```
bool filter::eventFilter(QObject *watched, QEvent *event)
{
```

```
   if(event->type() == QEvent::KeyPress)
   {
       QKeyEvent * kEvent = static_cast<QKeyEvent*>(event);
       if((kEvent->key() >= Qt::Key_A && kEvent->key() <= Qt::Key_Z) || kEvent-
>key() == Qt::Key_Backspace)
       {
          return false;
       }
      return true;
   }
  return false;
        }
```

2. Унеможливлення введення символів або літер іншої мови під час введення слова на російській мові:

```
bool filter_rus::eventFilter(QObject *watched, QEvent *event)
{
   if(event->type() == QEvent::KeyPress)
   {
       QKeyEvent * kEvent = static_cast<QKeyEvent*>(event);
       if((kEvent->key() >= 1040 && kEvent->key() <= 1071) || kEvent->key() ==
Qt::Key_Backspace)
       {
          return false;
       }
      return true;
   }
  return false;
        }
```

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Вхідні дані:

- передбачуване число операторів - 1500;
- коефіцієнт складності програми - 2;
- коефіцієнт корекції програми в ході її розробки - 0,08;
- годинна заробітна плата програміста, грн./год - 30.

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

Трудомісткість розробки ПЗ можна розрахувати за формулою:

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

де to – витрати праці на підготовку й опис поставленої задачі (50),

ти – витрати праці на дослідження алгоритму рішення задачі,

та – витрати праці на розробку блок-схеми алгоритму,

тп – витрати праці на програмування по готовій блок-схемі,

тотл – витрати праці на налагодження програми на ЕОМ,

тд – витрати праці на підготовку документації.

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

Умовне число операторів (підпрограм):

$$Q = q \times C \times (1+ p) \text{ , людино-годин,} \quad (3.2)$$

де q - передбачуване число операторів,

C - коефіцієнт складності програми;

p - коефіцієнт кореляції програми в ході її розробки.

$$Q = 1500 \cdot 2 \cdot (1 + 0{,}08) = 3240 \text{ людино-годин.} \quad (3.3)$$

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

$$t_u = \frac{QB}{(75...85)K}, \text{ людино-годин,} \quad (3.4)$$

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

k – коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.

Витрати праці на розробку алгоритму рішення задачі:

$$t_u = \frac{3240 \cdot 1{,}3}{80 \cdot 1{,}2} = 44, \text{ людино-годин.} \quad (3.5)$$

Витрати на складання програми по готовій блок-схемі:

$$t_a = \frac{Q}{(20...25)K} \quad \text{людино-годин.} \quad (3.6)$$

$$t_a = \frac{3240}{22 \cdot 1{,}2} = 123 \text{ людино-годин.} \quad (3.7)$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20...25)K} \quad \text{людино-годин.} \tag{3.8}$$

$$t_n = \frac{3240}{23 \cdot 1{,}2} = 117 \text{ людино-годин.} \tag{3.9}$$

Витрати праці на налагодження програми на ЕОМ:

$$t_{отл} = \frac{Q}{(4...5)K} \quad \text{людино-годин.} \tag{3.10}$$

$$t_{отл} = \frac{3240}{5 \cdot 1{,}2} = 540 \text{ людино-годин.} \tag{3.11}$$

Витрати праці на підготовку документації:

$$t_д = t_{др} + t_{до}, \text{ людино-годин,} \tag{3.12}$$

де tдр - трудомісткість підготовки матеріалів і рукопису.

$$t_{др} = \frac{Q}{(15...20)K}, \text{ людино-годин.} \tag{3.13}$$

$$t_{др} = \frac{3240}{18 \cdot 1{,}2} = 150 \text{ людино-годин,} \tag{3.14}$$

де tдо - трудомісткість редагування, печатки й оформлення документації.

$$t_{до} = 0{,}75 \cdot t_{др}, \text{ людино-годин.} \tag{3.15}$$

$$t_{до} = 150 + 73 = 223 \text{ людино-годин.} \tag{3.16}$$

Отримуємо трудомісткість розробки програмного забезпечення:

$$t = 50 + 44 + 123 + 117 + 540 + 223 = 1097 \text{ людино-годин.} \tag{3.17}$$

3.2. Розрахунок витрат на створення програми

Витрати на створення ПЗ Кпо включають витрати на заробітну плату виконавця програми Зз/п і витрат машинного часу, необхідного на налагодження програми на ЕОМ:

$$K_{ПО} = З_{ЗП} + З_{МВ}, \text{ грн.} \quad (3.18)$$

Заробітна плата виконавців визначається за формулою:

$$З_{ЗП} = t \cdot C_{ПР}, \text{ грн,} \quad (3.19)$$

де t - загальна трудомісткість, людино-годин,

Спр - середня годинна заробітна плата програміста, грн/година.

$$З_{ЗП} = 1097 \cdot 30 = 32910 \text{ грн.} \quad (3.20)$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{МВ} = t_{отл} \times C_{М}, \text{ грн,} \quad (3.21)$$

де tотл - трудомісткість налагодження програми на ЕОМ, год.,

Смч - вартість машино-години ЕОМ, грн/год.

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення АСУП.

$$З_{МВ} = 540 \times 5 = 2700 \text{ грн.}, \quad (3.22)$$

$$Кпо = 2700 + 32910 = 35610 \text{ грн.} \quad (3.23)$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p} \text{ міс.,} \tag{3.24}$$

де Bk - число виконавців,

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

$$T = \frac{1097}{1 \cdot 176} = 6.3 \text{ міс.} \tag{3.25}$$

Визначено трудомісткість розробленої інформаційної системи (1097 людгод), проведений підрахунок вартості роботи по створенню програми (39610 грн.) та розраховано час на його створення (6,3 міс).

ВИСНОВКИ

В ході виконання кваліфікаційної роботи був розроблений програмний додаток, що дозволяє вивчати англійську мову за допомогою ігрової форми в різних режимах запам’ятовування слів.

Програмний додаток має простий інтуїтивно-зрозумілий інтерфейс та містить підказки та перевірки коректності введених даних, що дозволяє працювати з ним людям з мінімальними навиками володіння комп’ютером. Передбачено, що додатком може користуватися кожний зацікавлений користувач.

Для реалізації програмного продукту було використано середовище Qt Creator, програмний продукт написаний на мові програмування C++, СУБД було використано MySQL.

В наступному даний програмний продукт пройде технічне випробування серед вчителів англійської мови навчального закладу з метою практичної користі при застосуванні на заняттях. Очікується позитивна оцінка естетики та нетрадиційного графічного інтерфейсу, який робить можливим сприйняття матеріалу як ігрової практики, а не як навчального матеріалу.

Наступними кроками у розвитку інновацій освітнього процесу мають стати логічні продовження даного програмного проекту із різних напрямків вивчення англійської мови.

В економічному розділі визначено трудомісткість розробленої інформаційної системи (1097 люд-год), проведений підрахунок вартості роботи по створенню програми (39610 грн) та розраховано час на його створення (6,3 міс).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання : (ГОСТ 7.1-2003, IDT) : ДСТУ ГОСТ 7.1:2006. – Чинний з 2007–07–01. – К. : Держспоживстандарт України, 2007. – 47 с. – (Система стандартів з інформації, бібліотечної та видавничої справи) (Національний стандарт України).

2. Бусыгин Б.С., Дивизинюк М.М., Коротенко Г.М., Коротенко Л.М. Введение в современную информатику. Учебник. – Севастополь: Издательство СНУЯЭиП, 2005. – 644 с. / URL: http://www.programmer.dp.ua/book-ru-k02.php. дата звернення: 15.01.2018.

3. Гузеева, К. А. Практическое пособие по переводу с английского языка на русский : уч. пособие / К. А. Гузеева. – Москва : СОЮЗ, 2018. – 258 с.

4. Даминова, С.О. The Rainbow of Chemistry. Учебное пособие по чтению на английском языке для студентов химических специальностей. Уровень B1-B2 / С.О. Даминова. – Москва : URSS, 2020. – 224 с.

5. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення : ДСТУ 3008-95. – Чинний від 1996–01–01. – К. : Держстандарт України, 1996. – 39 с.

6. ДСТУ 2394-94 Інформація та документація. Комплектування фонду, бібліографічний опис, аналіз документів. Терміни та визначення. – Чинний від 01.01.1995. - Київ : Держстандарт України, 1994. – 88 с.

7. Методичні вказівки з виконання економічного розділу в дипломних проектах студентів спеціальності “Комп’ютерні системи ” / Укладачі О.Г. Вагонова, Нікітіна О.Б. Н.Н. Романюк – Дніпропетровськ: Національний гірничий університет. – 2013. – 23с.

8. Методичні рекомендації до виконання кваліфікаційних робіт бакалаврів напряму підготовки 121 «Програмна інженерія» галузі знань 12 Інформаційні технології/, Л.М. Коротенко , О.С. Шевцова; Нац. гірн. ун-т. – Д : ДВНЗ НГУ, 2019. – 65 с.

9. Методичні рекомендації щодо написання, оформлення та представлення учнівських науково-дослідницьких робіт учнів – членів Малої академії наук України / Г.Г. Півняк, Л.М. Коротенко, І.М. Удовик, Є.М. Головня – Д.: ДВНЗ «Національний гірничий університет», 2017. – 24 с.

10. Офіційний сайт розробника середовища програмування Visual Studio URL:visualstudio.microsoft.com/ru/vs/features/ide/.

11. Сайт 4brain. URL: https://4brain.ru/memory/povtorenie.php. дата звернення: 26.01.2020

12. Скорочення слів в українській мові у бібліографічному описі. Загальні правила та вимоги : ДСТУ 3582-97. – Чинний від 1998–07–01. – К. : Держстандарт України, 1998. – 24 с. – (Державний стандарт України).

13. Скорочення слів в українській мові у бібліографічному описі. Загальні правила та вимоги : ДСТУ 3582-97. – Чинний від 1998–07–01. – К. : Держстандарт України, 1998. – 24 с. – (Державний стандарт України).

14. Троелсен Э. Язык программирования C# 2010 и платформа .NET 4.0, 5-

е изд. / Э Троелсен. - М.: ООО “И.Д. Вильямс”, 2011. – 1392 с.

15. Allen M.P., Tildesley D.J. Computer Simulation of Liquids, Clarendon Press: Oxford,1987. – 50 с.

16. English Tests FCE.. URL: http://englishtests.ucoz.com/FCE/test1/use/fce_paper3_use_test1_part4.htm. дата звернення: 16.01.2021

17. FCE Word Formation. URL: https://cefrexambot.com/exams/fce-word-formation-1/. дата звернення: 6.02.2021

18. FCE word formation card game.. URL: https:// www.lessonplansdigger.com/2015/07/27/fce-word-formation-card-game/ Дата звернення: 1.02.2021

19. Ivunina, E. Word Formation for FCE. URL: https://itunes.apple.com/us/app/word-formation-for-fce/id1352759672mt=8. дата звернення: 16.01.2021

20. WF041 - Word Formation Sentences. Gap-fill exercise. URL: https://www.english-grammar.at/online_exercises/word-formation/wf041-sentences.htm дата звернення: 10.01.2021

ДОДАТОК А

КОД ПРОГРАМИ

```
#include <QMainWindow>
#include <QBoxLayout>
#include <QLabel>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QSqlError>
#include <QDebug>
#include <QSqlTableModel>
#include <QWidget>
#include <QPushButton>
#include <QFileInfo>
#include <QPicture>
#include <QtTest/QTest>
#include <QCompleter>
#include <QMessageBox>
namespace Ui {
class AddTheme;
}
class AddTheme : public QWidget
{    Q_OBJECT
public:
    explicit AddTheme(QWidget *parent = nullptr);
    ~AddTheme();
signals:
    void goout();
private slots:
    void on_listWidget_itemSelectionChanged();
    void on_add_photo_word_clicked();
    void on_add_photo_theme_clicked();
    void on_addword_clicked();
    void on_addtheme_clicked();
    void on_delete_w_clicked();
    void on_exit_clicked();
private:
    void messange(QString a);
    QString rus[40];
    QString eng[40];
    QString link[40];
    int count_word;
    Ui::AddTheme *ui;
};
#include <QWidget>
#include <QDebug>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QMessageBox>
#include "mainwindow.h"
#include "refences.h"
#include <QFile>
namespace Ui {
class Autorization;
}
class Autorization : public QWidget
{    Q_OBJECT
public:
    explicit Autorization(QWidget *parent = nullptr);
```

```
    ~Autorization();
private slots:
   void on_enter_clicked();
    void on_choose_activated(int index);
private:
    Ui::Autorization *ui;
    MainWindow * window;
    Refences * ref;
};
#endif // AUTORIZATION_H


#include <QWidget>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QDebug>
#include "choosemode.h"
namespace Ui {
class Card;
}
class Card : public QWidget
{    Q_OBJECT
public:
    explicit Card(int id_u, int id_c, QWidget *parent = nullptr);
    ~Card();
private slots:
    void on_go_test_clicked();
    void on_next_clicked();
    void on_before_clicked();
    void on_listen_clicked();
   void on_back_clicked();
   signals:
  void go_back();
private:
    int id_user;
    int id_theme;
    int current=1;
    Ui::Card *ui;
};
#endif // CARD_H

#include <QMainWindow>
#include <QBoxLayout>
#include <QLabel>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QMessageBox>
#include <QSqlError>
#include <QDebug>
#include <QTableView>
#include <QSqlTableModel>
#include <QWidget>
#include <QPushButton>
#include <QDate>
#include <QtTest/QTest>
#include <QTextToSpeech>
#include <QSettings>
namespace Ui {
class ChooseCor5;
}
class ChooseCor5 : public QWidget
{    Q_OBJECT
```

```
int id_user, branch;
public:
    explicit ChooseCor5(int id, int id_c, QWidget *parent = nullptr);
    ~ChooseCor5();
signals:
    void gochoose(int progress);
private slots:
    void on_cor_clicked();
    void on_notcor_clicked();
    void on_skip_clicked();
    void on_sound_clicked();
private:
    int id_theme;
    Ui::ChooseCor5 *ui;
};
#endif // CHOOSECOR5_H
#ifndef CHOOSECOUPLE3_H
#define CHOOSECOUPLE3_H

#include <QMainWindow>
#include <QBoxLayout>
#include <QLabel>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QMessageBox>
#include <QSqlError>
#include <QDebug>
#include <QTableView>
#include <QSqlTableModel>
#include <QWidget>
#include <QPushButton>
#include <QtTest/QTest>
#include <QTextToSpeech>
#include <QSettings>
namespace Ui {
class ChooseCouple3;
}
class ChooseCouple3 : public QWidget
{    Q_OBJECT
int id_user;
int koln;
QString knock;
public:
    explicit ChooseCouple3(int id, int id_c, QWidget *parent = nullptr);
    ~ChooseCouple3();
signals:
    void gochoose();
private slots:
    void on_b1_clicked();
    void on_godali_clicked();
    void on_b2_clicked();
    void on_b3_clicked();
    void on_b4_clicked();
    void on_b5_clicked();
    void on_b6_clicked();
    void on_b7_clicked();
    void on_b8_clicked();
    void on_b9_clicked();
    void on_b10_clicked();
    void on_sound1_clicked();
    void on_sound2_clicked();
    void on_sound3_clicked();
```

```
    void on_sound4_clicked();
    void on_sound5_clicked();
private:
    int id_theme;
    void exit();
    void play_sound(int index);
    QPushButton * click_on_1_5(QPushButton * b);
    QPushButton * click_on_6_10(QPushButton * b);
    Ui::ChooseCouple3 *ui;
};
#endif // CHOOSECOUPLE3_H
#ifndef CHOOSEMODE_H
#define CHOOSEMODE_H


#include <QWidget>
#include <collecting1.h>
#include <choosecouple3.h>
#include <choosecor5.h>
#include <highlight4.h>
#include <choosetrans2.h>
namespace Ui {
class ChooseMode;
}
class ChooseMode : public QWidget
{    Q_OBJECT
public:
    explicit ChooseMode(int id, int id_u,QWidget *parent = nullptr);
     ~ChooseMode();
signals:
    void toolbar1();
public slots:
    void setword(int id);
    void getsignalregim1();
    void getsignalregim2();
    void getsignalregim5();
    void getsignalregim4();
    void getsignalregim3();
    void on_back_clicked();
    void on_reg1_clicked();
    void on_reg2_clicked();
    void on_reg3_clicked();
    void on_reg4_clicked();
    void on_reg5_clicked();
    void on_reg6_clicked();
    void on_delete_theme_clicked();
    void on_learning_clicked();
private:
    void chooseregime(int current);
     struct srandom{srandom(){srand(time(NULL));}}orandom;
    int id_user;
    int id_theme;
    int flag;
    ChooseCor5 * wreg5;
    HighLight4 * wreg4;
    ChooseCouple3 * wreg3;
    Collecting1 * wreg1;
    ChooseTrans2 * wreg2;
    Ui::ChooseMode *ui;
    void messange(QString);
};
#endif // CHOOSEMODE_H
```

```
#ifndef CHOOSETRANS2_H
#define CHOOSETRANS2_H

#include <QMainWindow>
#include <QBoxLayout>
#include <QLabel>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QSqlError>
#include <QDebug>
#include <QSqlTableModel>
#include <QWidget>
#include <QPushButton>
#include <QFileInfo>
#include <QPicture>
#include <QtTest/QTest>
#include <QDate>
namespace Ui {
class ChooseTrans2;
}
class ChooseTrans2 : public QWidget
{    Q_OBJECT
public:
    explicit ChooseTrans2(int id, int id_c, QWidget *parent = nullptr);
    ~ChooseTrans2();
signals:
    void gochoose();
private slots:
    void on_b1_clicked();
    void on_b2_clicked();
    void on_b3_clicked();
    void on_b4_clicked();
    void on_b5_clicked();
    void on_b1_ph_clicked();
    void on_b2_ph_clicked();
    void on_b3_ph_clicked();
    void on_b4_ph_clicked();
    void on_b5_ph_clicked();
private:
    QPushButton * onbuttonclick(QPushButton * b);
    QPushButton * changetext(QPushButton * b, int kol, QString id_t);
    void exit();
    int id_user;
    int id_theme;
    int id_w;
    QString wordatbutton[4];
    int count_word;
    Ui::ChooseTrans2 *ui;
};
#endif // CHOOSETRANS2_H
#ifndef COLLECTING1_H
#define COLLECTING1_H

#include <QBoxLayout>
#include <QLabel>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QMessageBox>
#include <QSqlError>
#include <QDebug>
#include <QTableView>
#include <QSqlTableModel>
```

```
#include <QWidget>
#include <QPushButton>
#include <QFileInfo>
#include <QPicture>
#include <QtTest/QTest>
#include <QMessageBox>
#include <QDate>
namespace Ui {
class Collecting1;
}
class Collecting1 : public QWidget
{    Q_OBJECT
public:
    explicit Collecting1(int id, int id_c, QWidget *parent = nullptr);
    ~Collecting1();
private slots:
void signalbutton();
void signalbutton_wp();
void on_go_out_clicked();
void on_back_ph_clicked();
void on_back_noph_clicked();
void on_skip_clicked();
signals:
    void gochoose();
private:
void messange(QString a);
    int id_user;
    int id_theme;
    int id_word_g;
    int *touch;
    int position;
    QPushButton * setbutton(QString text);
    QPushButton * setbutton_wp(QString text);
    Ui::Collecting1 *ui;
};
#endif // COLLECTING1_H
#ifndef FILTER_H
#define FILTER_H

#include <QDebug>
#include <QObject>
#include <QKeyEvent>
class filter : public QObject
{    Q_OBJECT
public:
    explicit filter(QObject *parent = nullptr);
signals:
public slots:
    // QObject interface
public:
    bool eventFilter(QObject *watched, QEvent *event);
};
#endif // FILTER_H
#ifndef FILTER_RUS_H
#define FILTER_RUS_H

#include <QDebug>
#include <QObject>
#include <QKeyEvent>
class filter_rus : public QObject
{    Q_OBJECT
public:
```

```
    explicit filter_rus(QObject *parent = nullptr);
signals:
public slots:
     bool eventFilter(QObject *watched, QEvent *event);
};
#endif // FILTER_RUS_H
#ifndef HIGHLIGHT4_H
#define HIGHLIGHT4_H

#include <QBoxLayout>
#include <QLabel>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QMessageBox>
#include <QSqlError>
#include <QDebug>
#include <QTableView>
#include <QSqlTableModel>
#include <QWidget>
#include <QPushButton>
#include <QFileInfo>
#include <QPicture>
#include <QtTest/QTest>
#include <QDate>
namespace Ui {
class HighLight4;
}
class HighLight4 : public QWidget
{    Q_OBJECT
int id_user;
int id_word_g;
int *touch;
int position;
public:
    explicit HighLight4(int id, int id_c, QWidget *parent = nullptr);
    ~HighLight4();
signals:
    void gochoose(int progress);
private slots:
void signalbutton();
void signalbutton_wp();
void on_go_out_clicked();
void on_skip_clicked();
private:
int id_theme;
QPushButton * setbutton(QString text);
QPushButton * setbutton_wp(QString text);
    Ui::HighLight4 *ui;
};
#endif // HIGHLIGHT4_H
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include "startlearning.h"
#include <QMainWindow>
#include <choosemode.h>
#include <refences.h>
#include <addtheme.h>
#include <QBoxLayout>
#include <QLabel>
#include <QSqlDatabase>
#include <QSqlQuery>
```

```
#include <QMessageBox>
#include <QSqlError>
#include <QDebug>
#include <QTableView>
#include <QSqlTableModel>
#include <QWidget>
#include <QPushButton>
#include <stdlib.h>
#include <QtWidgets/QApplication>
#include <QtWidgets/QMainWindow>
#include <QtCharts/QChartView>
#include <QtCharts/QBarSeries>
#include <QtCharts/QBarSet>
#include <QtCharts/QLegend>
#include <QtCharts/QBarCategoryAxis>
#include <QtCharts/QValueAxis>
#include <QPen>
#include <QColor>
namespace Ui {
class MainWindow;
}
class MainWindow : public QMainWindow
{    Q_OBJECT
public:
    explicit MainWindow(int id_u, QWidget *parent = nullptr);
    ~MainWindow();
signals:
    void setword_s(int id1, int id2);
private slots:
void getsignalstart();
void on_theme_clicked();
void newcolumn(QString st, int v, int k, int id);
void gotochoose();
void on_goout_clicked();
private:
    int id_user;
    Ui::MainWindow *ui;
    ChooseMode * choosemodeui;
    StartLearning * learn;
    QWidget * AddNewButton(QString st, int, int, int);
};
#endif // MAINWINDOW_H
#ifndef REFENCES_H
#define REFENCES_H

#include <QWidget>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QSqlError>
#include <QDebug>
#include <QSettings>
#include "addtheme.h"
#include <QtWidgets/QApplication>
#include <QtWidgets/QMainWindow>
#include <QtCharts>
#include <QtCharts/QChartView>
#include <QtCharts/QBarSeries>
#include <QtCharts/QBarSet>
#include <QtCharts/QLegend>
#include <QtCharts/QBarCategoryAxis>
#include <QtCharts/QValueAxis>
#include <QPen>
```

```
#include <QColor>
#include <QBoxLayout>
#include <QLabel>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QMessageBox>
#include <QSqlError>
#include <QDebug>
#include <QTableView>
#include <QSqlTableModel>
#include <QWidget>
#include <QPushButton>
#include <stdlib.h>
namespace Ui {
class Refences;
}
class Refences : public QWidget
{    Q_OBJECT
public:
    explicit Refences(QWidget *parent = nullptr);
    ~Refences();
private slots:
  void  messange(QString a);
    void on_save_clicked();
    void on_add_clicked();
    void on_themes_currentRowChanged(int currentRow);
    void on_delete_theme_clicked();
    void on_delete_student_clicked();
    void on_add_s_clicked();
    void on_student_currentRowChanged(int currentRow);
    void getsignaladdtheme();
    void on_tabWidget_currentChanged(int index);
    void on_choose_school_currentIndexChanged(int index);
private:
    Ui::Refences *ui;
    QSettings *settings;
    AddTheme * addtheme;
     QHBoxLayout * ch= new QHBoxLayout;
};
#endif // REFENCES_H
#ifndef STARTLEARNING_H
#define STARTLEARNING_H

#include <QDebug>
#include "card.h"
#include <QWidget>
#include <QSqlDatabase>
#include <QSqlQuery>
namespace Ui {
class StartLearning;
}
class StartLearning : public QWidget
{
    Q_OBJECT
signals:
    void signal_start();
public:
    explicit StartLearning(QWidget *parent = nullptr);
    ~StartLearning();
 void setword(int id_theme, int id_user);
public slots:
   void on_start_b_clicked();
```

```
private slots:
void getsignalback();
signals:
void go_main();
private:
    int id_theme_text;
    int id_user_sent;
    Ui::StartLearning *ui;
    Card * learn;
};
#endif // STARTLEARNING_H

#include "addtheme.h"
#include "ui_addtheme.h"
#include "filter.h"
#include "filter_rus.h"
AddTheme::AddTheme(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::AddTheme)
{
    ui->setupUi(this);
    count_word=0;
    setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
    setWindowTitle("LerningEnglish");
QPixmap pict;
   pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/dots.png");
   ui->iden_word->setPixmap(pict);
   ui->iden_theme->setPixmap(pict);
ui->delete_w->hide();
QSqlQuery query;
QStringList list_rus, list_eng;
query.exec("select eng, rus from word");
while(query.next())
{
    list_eng<<query.value(0).toString();
    list_rus<<query.value(1).toString();
}
QCompleter *completer_rus = new QCompleter(list_rus,this);
QCompleter *completer_eng = new QCompleter(list_eng,this);
ui->rus->setCompleter(completer_rus);
ui->eng->setCompleter(completer_eng);
ui->eng->installEventFilter(new filter(ui->eng));
ui->rus->installEventFilter(new filter_rus(ui->rus));
}
AddTheme::~AddTheme()
{
    delete ui;
}
void AddTheme::on_listWidget_itemSelectionChanged()
{
ui->delete_w->show();
}
void AddTheme::on_add_photo_word_clicked()
{ QPixmap pict;
    QFileInfo check_file(ui->link_word->text());
    if(check_file.exists())
    {
        pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/correct.png");
        ui->iden_word->setPixmap(pict);
    }
    else
    {
```

```
        pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/nocorrect.png");
        ui->iden_word->setPixmap(pict);
    }
}
void AddTheme::on_add_photo_theme_clicked()
{
    QPixmap pict;
       QFileInfo check_file(ui->link_theme->text());
       if(check_file.exists())
       {
          pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/correct.png");
          ui->iden_theme->setPixmap(pict);
       }
       else
       {
           pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/nocorrect.png");
           ui->iden_theme->setPixmap(pict);
       }
}

void AddTheme::on_addword_clicked()
{int otv=0;
    for(int i=0;i<count_word;i++)
    {
       if(ui->rus->text()==rus[i] || ui->eng->text()==eng[i]) otv=1;
    }
    if(ui->rus->text().size()>0 && ui->eng->text().size()>0 && otv==0)
    {
rus[count_word]=ui->rus->text();
eng[count_word]=ui->eng->text();
QFileInfo check_file(ui->link_word->text());
if(check_file.exists())
{
 link[count_word]=ui->link_word->text();
}
else link[count_word]="0";
ui->listWidget->addItem(ui->eng->text()+" ("+ui->rus->text()+")");
count_word++;
    }
    else
    {if(otv==1)
       {
         messange("Такое слово уже есть!");
       }
       else{
       if(!(ui->rus->text().size()>0))ui->rus->setStyleSheet(" font: 12pt; background-color: rgb(255, 109, 73);");
       if(!(ui->eng->text().size()>0)) ui->eng->setStyleSheet(" font: 12pt; background-color: rgb(255, 109, 73);");
       QTest::qWait(1500);
       ui->rus->setStyleSheet(" font: 12pt; background-color: rgb(255, 252, 237);");
       ui->eng->setStyleSheet(" font: 12pt; background-color: rgb(255, 252, 237);");
       }
    }
   ui->eng->clear();
   ui->rus->clear();
   ui->link_word->clear();
   QPixmap pict;
      pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/dots.png");
      ui->iden_word->setPixmap(pict);
}
void AddTheme::messange(QString a)
{
    QMessageBox mes;
```

```
    mes.setWindowTitle("Error");
    mes.resize(150,100);
    mes.setText(a);
    mes.exec();
}
void AddTheme::on_addtheme_clicked()
{if(ui->theme->text().size()>0 && ui->listWidget->count()>0){
    QSqlQuery query;
    QString kol_w, id_theme;
    kol_w.setNum(count_word);
    query.exec("insert into card(name_c) value('"+ui->theme->text()+"')");
    query.clear();
    query.exec("select name_c from card where id_c=(select max(id_c) from card)");
    qDebug()<<query.value(0).toString();
    query.clear();
    query.exec("select max(id_c) from card");
    query.next();
    id_theme=query.value(0).toString();
    query.clear();
    QFileInfo check_file(ui->link_theme->text());
    if(check_file.exists())
    QFile::copy(ui->link_theme->text(),"C:/Users/Meshtli/Desktop/hi/Kursach/card/"+id_theme+".jpg");
    for(int i=0;i<count_word;i++){
        int otv=0;
        QString b=rus[i];
        QString d=eng[i];
        QString id_word;
        b=b.toLower();
        d=d.toLower();
        query.exec("select eng, rus, id_w from word");
        while(query.next())
        {
        QString a=query.value(1).toString(), c=query.value(0).toString();
        a=a.toLower();
        c=c.toLower();
        if(a==c || c==d)
        {
            otv=1;
            id_word=query.value(2).toString();
        }
        }
        query.clear();
        if(otv==0)
        {
    d[0]=d[0].toUpper();
    b[0]=b[0].toUpper();
    query.exec("insert into word(eng, rus) value('"+d+"','"+b+"')");
    query.clear();
    query.exec("insert into having_c(id_card, id_word) value((select max(id_c) from card), (select max(id_w) from
word))");
    query.clear();
    query.exec("select max(id_w) from word");
    query.next();
    id_word=query.value(0).toString();
    query.clear();
        }
        else
        {
            query.exec("insert into having_c(id_card, id_word) value((select max(id_c) from card), "+id_word+")");
            query.clear();
        }
        if(link[i]!="0")
```

```
        {
            QFile::copy(link[i],"C:/Users/Meshtli/Desktop/hi/Kursach/word/"+id_word+".jpg");
        }
    }
    emit goout();
    }
    else
    {if(!(ui->listWidget->count()>0))
        ui->listWidget->setStyleSheet(" font: 13pt; background-color: rgb(255, 109, 73); border: 2px solid rgb(76, 39,
19);");
      if (!(ui->theme->text().size()>0))
        ui->theme->setStyleSheet(" font: 12pt; background-color: rgb(255, 109, 73); border: 1px solid rgb(76, 39, 19);
border-radius: 10px;");
        QTest::qWait(1500);
        ui->theme->setStyleSheet(" font: 12pt; background-color: rgb(255, 252, 237); border: 1px solid rgb(76, 39, 19);
border-radius: 10px;");
  ui->listWidget->setStyleSheet(" font: 13pt; background-color: rgb(255, 252, 237); border: 2px solid rgb(76, 39, 19);");
    }
}
void AddTheme::on_delete_w_clicked()
{qDeleteAll(ui->listWidget->selectedItems());
}
void AddTheme::on_exit_clicked()
{
emit goout();
}

#include "autorization.h"
#include "ui_autorization.h"
void messange(QString a)
{
    QMessageBox mes;
  mes.setWindowTitle("Error");
    mes.resize(200,100);
    mes.setText(a);
    mes.exec();
}
Autorization::Autorization(QWidget *parent) :
    QWidget(parent),
    ui(new Ui::Autorization)
{    ui->setupUi(this);
     setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
     setWindowTitle("LerningEnglish");
    ui->parol->show();
    ui->choose_school->hide();
    QSqlDatabase db =  QSqlDatabase::addDatabase("QODBC");
      db.setDatabaseName("kurs");
      db.setHostName("localhost");
      db.setPort(3306);
    db.setUserName("root");
    db.setPassword("1234");
   if(!db.open()) messange("База данных не подкючена!");
    QSqlQuery school;
    school.exec("select name_user from users");
    while(school.next()){
ui->choose_school->addItem(school.value(0).toString());
    }
}
Autorization::~Autorization()
{
    delete ui;
}
```

```
void Autorization::on_enter_clicked()
{ qDebug()<<ui->choose->currentText();
    if(ui->choose->currentText()=="Учитель")
    {
QFile file("C:/Users/Meshtli/Desktop/hi/Kursach/auto.txt");
  if((file.exists()&&(file.open(QIODevice::ReadOnly))))
   {
    if(ui->parol->text()==file.readAll()) {
        this->close();
        ref = new Refences();
       // ref->setGeometry(0, 0, 900, 500);
        ref->show();
      //  connect(wreg5, &ChooseCor5::gochoose,this, &ChooseMode::getsignalregim5);
    }
    else messange("Введен неверный пароль!");
    }
   }
    else
    {int i=0;
        QSqlQuery school;
        school.exec("select name_user, id_u from users");
         while(school.next()){
            if(i==ui->choose_school->currentIndex()) {
   window = new MainWindow(school.value(1).toInt());
   window->show();
   this->close();
             }
                i++;
        }
        }
}
void Autorization::on_choose_activated(int index)
{
    if(index==0)
    {
        ui->parol->show();
        ui->choose_school->hide();
           }
    else
    {
        ui->parol->hide();
        ui->choose_school->show();
           }
}

#include "card.h"
#include "ui_card.h"
Card::Card(int id_u, int id_c, QWidget *parent) :
    QWidget(parent),
    ui(new Ui::Card)
{
    ui->setupUi(this);
    setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
    setWindowTitle("LerningEnglish");
    id_user=id_u;
    id_theme=id_c;
    ui->current->setText(QString::number(current)+"/10");
QSqlQuery query;
query.exec("select rus, eng, id_w from word where id_w in (select id_word from stopka where
id_user="+QString::number(id_u)+") && id_w in (select id_word from having_c where
id_card="+QString::number(id_c)+")");
query.next();
```

```
QPixmap p;
p.load("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+query.value(2).toString()+".png");
int w = ui->image->width();
int h = ui->image->height();
ui->image->setPixmap(p.scaled(w,h,Qt::KeepAspectRatio));
ui->rus->setText(query.value(0).toString());
ui->eng->setText(query.value(1).toString());
}

Card::~Card()
{
   delete ui;
}
void Card::on_go_test_clicked()
{
   ChooseMode * a= new ChooseMode(id_user, id_theme);
    a->setGeometry(this->pos().x(), this->pos().y(), 900, 500);
       a->on_reg6_clicked();
   this->close();
}
void Card::on_next_clicked()
{
   if(current<10){
       current++;
       ui->current->setText(QString::number(current)+"/10");
   QSqlQuery query;
   int i=1;
  query.exec("select rus, eng, id_w from word where id_w in (select id_word from stopka where
id_user="+QString::number(id_user)+") && id_w in (select id_word from having_c where
id_card="+QString::number(id_theme)+")");
   while(query.next()){
       qDebug()<<current<<i;
    if(current==i){
        QPixmap p;
        p.load("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+query.value(2).toString()+".png");
        int w = ui->image->width();
        int h = ui->image->height();
        ui->image->setPixmap(p.scaled(w,h,Qt::KeepAspectRatio));
        ui->rus->setText(query.value(0).toString());
        ui->eng->setText(query.value(1).toString());
    }
   i++;
   }
}
}
void Card::on_before_clicked()
{if(current>1){
   QSqlQuery query;
   int i=1;
   current--;
   ui->current->setText(QString::number(current)+"/10");
   query.exec("select rus, eng, id_w from word where id_w in (select id_word from stopka where
id_user="+QString::number(id_user)+") && id_w in (select id_word from having_c where
id_card="+QString::number(id_theme)+")");
   while(query.next()){
    if(current==i){
        QPixmap p;
        p.load("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+query.value(2).toString()+".png");
        int w = ui->image->width();
        int h = ui->image->height();
        ui->image->setPixmap(p.scaled(w,h,Qt::KeepAspectRatio));
        ui->rus->setText(query.value(0).toString());
```

```
        ui->eng->setText(query.value(1).toString());
     }
    i++;
    }
}
}
void Card::on_listen_clicked()
{
    QString word=ui->eng->text();
   QTextToSpeech * speech= new QTextToSpeech;
   QSettings * settings = new QSettings("C:/Users/Meshtli/Desktop/hi/Kursach/other/settings.conf",
QSettings::IniFormat);
   qDebug()<<settings->value("sound/speed").toDouble()/100;
   speech->setRate(settings->value("sound/speed").toDouble()/100);
   speech->setVolume(settings->value("sound/volume").toDouble()/100);
   speech->say(word);
}
void Card::on_back_clicked()
{
    emit go_back();
}

#include "choosecor5.h"
#include "ui_choosecor5.h"
ChooseCor5::ChooseCor5(int id, int id_c, QWidget *parent) :
    QWidget(parent),
    ui(new Ui::ChooseCor5)
{ struct srandom{srandom(){srand(time(NULL));}}orandom;
    ui->setupUi(this);
    setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
    setWindowTitle("LerningEnglish");
    id_user=id;
    id_theme=id_c;
    qDebug()<<"cor5";
    ui->skip->setStyleSheet("border-image: url(C:/Users/Meshtli/Desktop/hi/Kursach/other/skip.png) stretch;");
    QSqlQuery progress;
    progress.exec("select count(*) from stopka where id_user="+QString::number(id)+") && id_word in (select id_word
from having_c where id_card="+QString::number(id_theme)+") && bal>20");
    ui->progressBar->setRange(0,100);
    ui->progressBar->setValue(progress.value(0).toInt()*10);
    if(progress.value(0).toInt()==0)
        ui->progressBar->setStyleSheet("QProgressBar { color: rgb(255, 252, 237);  background-color:  rgb(255, 252,
237); border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
    else ui->progressBar->setStyleSheet("QProgressBar { color: rgb(76, 39, 19);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
    this->setWindowTitle("EnglishLerning");
    QSqlQuery query, query2;
    int kol=10;
    QString id_u;
    id_u.setNum(id);
    query.clear();
    int flag=0;
    while(flag!=-1){
        int random=rand()% kol, i=0;
        query.exec("select eng, rus, id_w from word where id_w in (select id_word from stopka where id_user="+id_u+")
&& id_w in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
        while(query.next())
        {
            random=rand()% kol;
            QSqlQuery bal;
            bal.exec("select bal from stopka where id_word="+query.value(2).toString());
            bal.next();
```

```
        if(random==0 && bal.value(0).toInt()<20 && flag!=-1){
            ui->word->setText(query.value(0).toString());
            ui->word->setAlignment(Qt::AlignCenter);
            int r=rand()%3;
            if(r==1) {ui->transl->setText(query.value(1).toString());
                ui->transl->setAlignment(Qt::AlignCenter);
                branch=0;
                flag=-1;
            }
            else {
                do{
                    query2.exec("select rus, eng, id_w from word where id_w in (select id_word from stopka where
id_user="+id_u+") && id_w in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
                    while(query2.next() && flag!=-1 )
                    {int ran=rand()% kol;
                        QSqlQuery bal;
                        bal.exec("select bal from stopka where id_word="+query.value(2).toString());
                        bal.next();
                        if(ran==0)
                            if (bal.value(0).toInt()<20 && query2.value(1).toString()!=ui->word->toPlainText()) {
                                ui->transl->setText(query2.value(0).toString());
                                ui->transl->setAlignment(Qt::AlignCenter);
                                flag=-1;
                                branch=1;
                            }
                    }
                    query2.clear();

                }while(flag!=-1);
            }
        }
        else i++;
    }
    query.clear();
  }
}
ChooseCor5::~ChooseCor5()
{
  delete ui;
}
void ChooseCor5::on_cor_clicked()
{ QSqlQuery query, upd;
  if(branch==0)
  {       query.exec("select eng, rus, id_w from word where id_w in (select id_word from stopka where
id_user="+QString::number(id_user)+") && id_w in (select id_word from having_c where
id_card="+QString::number(id_theme)+")");
      while(query.next())
      {
        if(query.value(0).toString()==ui->word->toPlainText()) {
            QString rab2;
            QSqlQuery bal;
            bal.exec("select bal from stopka where id_word="+query.value(1).toString());
            bal.next();
            rab2.setNum(query.value(1).toInt());
            QSqlQuery table;
            table.exec("select score from regime where id_r=5");
            table.next();
            upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where id_word="+rab2);
            table.clear();
        }
      }
      query.clear();
```

```
    }
    if(branch==0)
    {
        ui->word->setStyleSheet("font: 16pt;  background-color:rgb(148, 228, 136); border: 1px solid rgb(148, 228, 136);
border-radius: 5px");
        ui->transl->setStyleSheet("font: 16pt;  background-color:rgb(148, 228, 136); border: 1px solid rgb(148, 228, 136);
border-radius: 5px");
    }
    else
    {
        ui->word->setStyleSheet("font: 16pt;  background-color:rgb(231, 128, 128); border: 1px solid rgb(231, 128, 128);
border-radius: 5px");
        ui->transl->setStyleSheet("font: 16pt;  background-color:rgb(231, 128, 128); border: 1px solid rgb(231, 128, 128);
border-radius: 5px");
    }
    QTest::qWait(1000);
    qDebug()<<"one";
    emit gochoose(0);
}
void ChooseCor5::on_notcor_clicked()
{ QSqlQuery query, upd;
    QString id_t;
    id_t.setNum(id_user);
    if(branch==1)
    {
        query.exec("select eng, rus, id_w from word where id_w in (select id_word from stopka where
id_user="+QString::number(id_user)+") && id_w in (select id_word from having_c where
id_card="+QString::number(id_theme)+")");
        while(query.next())
        {
            if(query.value(0).toString()==ui->word->toPlainText()) {
                QString rab1, rab2;
                QSqlQuery bal;
                bal.exec("select bal from stopka where id_word="+query.value(1).toString());
                bal.next();
                rab2.setNum(query.value(0).toInt());
                QSqlQuery table;
                table.exec("select score from regime where id_r=5");
                table.next();
                upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where id_word="+rab2+" &&
id_user="+id_t);
                upd.clear();
                upd.exec("select count(*) from bal_regime where id_user="+id_t);
                upd.next();
                if(upd.value(0).toInt()==0)
                {
                    QSqlQuery create;
                    create.exec("insert into bal_regime(id_reg,id_user,bal) value(1,"+id_t+",0)");
                    create.exec("insert into bal_regime(id_reg,id_user,bal) value(2,"+id_t+",0)");
                    create.exec("insert into bal_regime(id_reg,id_user,bal) value(3,"+id_t+",0)");
                    create.exec("insert into bal_regime(id_reg,id_user,bal) value(4,"+id_t+",0)");
                    create.exec("insert into bal_regime(id_reg,id_user,bal) value(5,"+id_t+",0)");
                }
                upd.clear();
                upd.exec("update bal_regime set bal=bal+"+table.value(0).toString()+" where id_r=5 && id_u"+id_t);
                table.clear();
            }
        }
        query.clear();
    }
    if(branch==1)
    {
```

```
        ui->word->setStyleSheet("font: 16pt;  background-color:rgb(148, 228, 136); border: 1px solid rgb(148, 228, 136);
border-radius: 5px");
        ui->transl->setStyleSheet("font: 16pt;  background-color:rgb(148, 228, 136); border: 1px solid rgb(148, 228, 136);
border-radius: 5px");
    }
    else
    {
        ui->word->setStyleSheet("font: 16pt;  background-color:rgb(231, 128, 128); border: 1px solid rgb(231, 128, 128);
border-radius: 5px");
        ui->transl->setStyleSheet("font: 16pt;  background-color:rgb(231, 128, 128); border: 1px solid rgb(231, 128, 128);
border-radius: 5px");
    }
    QTest::qWait(1000);
    qDebug()<<"two";
    emit gochoose(0);
}
void ChooseCor5::on_skip_clicked()
{
    emit gochoose(100);
}
void ChooseCor5::on_sound_clicked()
{
    QTextToSpeech * speech= new QTextToSpeech;
    QSettings * settings = new QSettings("C:/Users/Meshtli/Desktop/hi/Kursach/other/settings.conf",
QSettings::IniFormat);
    speech->setRate(settings->value("sound/speed").toDouble()/100);
    speech->setVolume(settings->value("sound/volume").toDouble()/100);
    speech->say(ui->word->toPlainText());
}

#include "choosecouple3.h"
#include "ui_choosecouple3.h"
ChooseCouple3::ChooseCouple3(int id1,  int id_c, QWidget *parent) :
    QWidget(parent),
    ui(new Ui::ChooseCouple3)
{   ui->setupUi(this);
    setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
    setWindowTitle("LerningEnglish");
this->setWindowTitle("EnglishLerning");
    knock="0";
    koln=0;
    struct srandom{srandom(){srand(time(NULL));}}orandom;
id_theme=id_c;
 ui->skip->setStyleSheet("border-image: url(C:/Users/Meshtli/Desktop/hi/Kursach/other/skip.png) stretch;");
id_user=id1;
ui->progressBar->setRange(0,100);
QSqlQuery progress;
progress.exec("select count(*) from stopka where id_user="+QString::number(id_user)+") && id_word in (select
id_word from having_c where id_card="+QString::number(id_theme)+") && bal>20");
progress.next();
 ui->progressBar->setValue(progress.value(0).toInt()*10);
if(progress.value(0).toInt()*10==0)
ui->progressBar->setStyleSheet("QProgressBar { color: rgb(255, 252, 237);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
else ui->progressBar->setStyleSheet("QProgressBar { color: rgb(76, 39, 19);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
    QSqlQuery query;
    int id[5]={-1,-1,-1,-1,-1};
    int i=0, kol=0;
    QString id_u;
    id_u.setNum(id_user);
```

```
    query.exec("select id_w from word where id_w in (select id_word from stopka where id_user="+id_u+") && id_w in
(select id_word from having_c where id_card="+QString::number(id_theme)+")");
    while(query.next())
    {
       int bal=query.value(0).toInt();
       if(bal<20) kol++;
    }
    qDebug()<<kol;
    if(kol>5)
    {
    while(i<=4){
 query.exec("select id_word from stopka where id_user="+id_u+" && id_word in (select id_word from having_c where
id_card="+QString::number(id_theme)+")" );
    while(query.next() && i<=4)
    { QSqlQuery balq;
       balq.exec("select bal from stopka where id_word="+query.value(0).toString());
       balq.next();
       int bal=balq.value(0).toInt();
       int random=rand()% kol;
         if(bal<20) if(random==0) {
           int id_w=query.value(0).toInt();
           int otv=1;
           for(int o=0;o<5;o++)
              if( id[o]==id_w) otv=0;
           if(otv==1){
           id[i]=id_w;
           i++;}
       }
    }
    query.clear();
    }
    }
    else
    {
       while(i<kol){
     query.exec("select id_word from stopka where id_user="+id_u);
       while(query.next() && i<kol)
       {QSqlQuery balq;
          balq.exec("select bal from stopka where id_word="+query.value(0).toString());
          balq.next();
          int bal=balq.value(0).toInt();
          int random=rand()% kol;
            if(bal<20) if(random==0) {
              int id_w=query.value(0).toInt();
              int otv=1;
              for(int o=0;o<5;o++)
                 if( id[o]==id_w) otv=0;
              if(otv==1){
              id[i]=id_w;
              i++;}
           }
       }
       query.clear();
       }
       while(i<5){
         query.exec("select id_word from stopka where id_user="+id_u);
       while(query.next() && i<=4)
       {     int random=rand()% kol;
            if(random==0) {
              int id_w=query.value(0).toInt();
              int otv=1;
              for(int o=0;o<5;o++)
```

```
                if( id[o]==id_w) otv=0;
            if(otv==1){
            id[i]=id_w;
            i++;}
          }
       }

       query.clear();
       }
   }
query.exec("select eng, rus, id_w from word where id_w in (select id_word from stopka where id_user="+id_u+") &&
id_w in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
int random[5]={-1,-1,-1,-1,-1};
i=0;
while(i<=4){
int otv=1, button = rand() % 5;
for(int o=0;o<5;o++)
   if( random[o]==button) otv=0;
if(otv==1) {random[i]=button;i++;}
}
while(query.next())
{
   int id_w=query.value(2).toInt();
 if(id_w==id[0]) {ui->b1->setText(query.value(0).toString());
if(random[0]==0) ui->b6->setText(query.value(1).toString());
if(random[0]==1) ui->b7->setText(query.value(1).toString());
if(random[0]==2) ui->b8->setText(query.value(1).toString());
if(random[0]==3) ui->b9->setText(query.value(1).toString());
if(random[0]==4) ui->b10->setText(query.value(1).toString());
    }
 if(id_w==id[1]) {ui->b2->setText(query.value(0).toString());
    if(random[1]==0) ui->b6->setText(query.value(1).toString());
    if(random[1]==1) ui->b7->setText(query.value(1).toString());
    if(random[1]==2) ui->b8->setText(query.value(1).toString());
    if(random[1]==3) ui->b9->setText(query.value(1).toString());
    if(random[1]==4) ui->b10->setText(query.value(1).toString());}
 if(id_w==id[2]) {ui->b3->setText(query.value(0).toString());
    if(random[2]==0) ui->b6->setText(query.value(1).toString());
    if(random[2]==1) ui->b7->setText(query.value(1).toString());
    if(random[2]==2) ui->b8->setText(query.value(1).toString());
    if(random[2]==3) ui->b9->setText(query.value(1).toString());
    if(random[2]==4) ui->b10->setText(query.value(1).toString());}
 if(id_w==id[3]) {ui->b4->setText(query.value(0).toString());
    if(random[3]==0) ui->b6->setText(query.value(1).toString());
    if(random[3]==1) ui->b7->setText(query.value(1).toString());
    if(random[3]==2) ui->b8->setText(query.value(1).toString());
    if(random[3]==3) ui->b9->setText(query.value(1).toString());
    if(random[3]==4) ui->b10->setText(query.value(1).toString());}
 if(id_w==id[4]) {ui->b5->setText(query.value(0).toString());
    if(random[4]==0) ui->b6->setText(query.value(1).toString());
    if(random[4]==1) ui->b7->setText(query.value(1).toString());
    if(random[4]==2) ui->b8->setText(query.value(1).toString());
    if(random[4]==3) ui->b9->setText(query.value(1).toString());
    if(random[4]==4) ui->b10->setText(query.value(1).toString());}
}
query.clear();
}
ChooseCouple3::~ChooseCouple3()
{
   delete ui;
}
void ChooseCouple3::on_godali_clicked()
```

```
{
 emit gochoose();
}
QPushButton * ChooseCouple3::click_on_1_5(QPushButton * b)
{
   koln++;
       if(b->styleSheet()=="background-color: rgb(85, 170, 255); font: 12pt") {b->setStyleSheet("font: 12pt;
background-color: rgb(255, 252, 237); color:rgb(37, 89, 67);"); knock="0"; koln=koln-2;}
       else
       if(b->styleSheet()!="background-color: #ff6450; font: 12pt"&& (ui->b6->text()==knock || ui->b7->text()==knock
|| ui->b8->text()==knock || ui->b9->text()==knock || ui->b10->text()==knock|| knock=="0"))
       {
   b->setStyleSheet("background-color: rgb(85, 170, 255); font: 12pt");
   if(knock=="0") knock=b->text();
   else
   {QSqlQuery query, upd;
      QString id_u;
      id_u.setNum(id_user);
      query.exec("select rus, eng, id_w from word where id_w in (select id_word from stopka where id_user="+id_u+")
&& id_w in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
       while(query.next())
       {
     if(b->text()==query.value(1).toString() && knock == query.value(0).toString())
     {b->hide();
   if(ui->b6->text()==knock) ui->b6->hide();
   if(ui->b7->text()==knock) ui->b7->hide();
   if(ui->b8->text()==knock) ui->b8->hide();
   if(ui->b9->text()==knock) ui->b9->hide();
   if(ui->b10->text()==knock) ui->b10->hide();
   QString rab2;
   rab2.setNum(query.value(2).toInt());
   QSqlQuery table;
   table.exec("select score from regime where id_r=3");
   table.next();
   upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where id_word="+rab2+" && id_user="+id_u);
   upd.clear();
   upd.exec("select count(*) from bal_regime where id_user="+id_u);
   upd.next();
   if(upd.value(0).toInt()==0)
   {
     QSqlQuery create;
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(1,"+id_u+",0)");
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(2,"+id_u+",0)");
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(3,"+id_u+",0)");
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(4,"+id_u+",0)");
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(5,"+id_u+",0)");
   }
   upd.clear();
   upd.exec("update bal_regime set bal=bal+"+table.value(0).toString()+" where id_r=3 && id_u"+id_u);
   upd.clear();
     }
     else {
   b->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b6->text()==knock) ui->b6->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b7->text()==knock) ui->b7->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b8->text()==knock) ui->b8->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b9->text()==knock) ui->b9->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b10->text()==knock) ui->b10->setStyleSheet("background-color: #ff6450; font: 12pt");
     }
       }
       query.clear();
       knock="0";
```

```
    }
       }

else koln--;
       return b;
}
QPushButton * ChooseCouple3::click_on_6_10(QPushButton * b)
{koln++;
   if(b->styleSheet()=="background-color: rgb(85, 170, 255); font: 12pt") {b->setStyleSheet("font: 12pt; background-
color: rgb(255, 252, 237); color:rgb(37, 89, 67);"); knock="0";koln=koln-2;}
   else
   if(b->styleSheet()!="background-color: #ff6450; font: 12pt" && (ui->b1->text()==knock || ui->b2->text()==knock ||
ui->b3->text()==knock || ui->b4->text()==knock || ui->b5->text()==knock|| knock=="0"))
       {
   b->setStyleSheet("background-color: rgb(85, 170, 255); font: 12pt");
   if(knock=="0") knock=b->text();
   else
   {QSqlQuery query, upd;
      QString id_u;
      id_u.setNum(id_user);
      query.exec("select rus, eng, id_w from word where id_w in (select id_word from stopka where id_user="+id_u+")
&& id_w in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
       while(query.next())
       {
     if(query.value(1).toString()==knock &&  b->text() == query.value(0).toString())
     {b->hide();
   if(ui->b1->text()==knock) ui->b1->hide();
   if(ui->b2->text()==knock) ui->b2->hide();
   if(ui->b3->text()==knock) ui->b3->hide();
   if(ui->b4->text()==knock) ui->b4->hide();
   if(ui->b5->text()==knock) ui->b5->hide();
   QString rab2;
   rab2.setNum(query.value(2).toInt());
   QSqlQuery table;
   table.exec("select score from regime where id_r=3");
   table.next();
   upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where id_word="+rab2+" && id_user="+id_u);
   upd.clear();
   upd.exec("select count(*) from bal_regime where id_user="+id_u);
   upd.next();
   if(upd.value(0).toInt()==0)
   {
     QSqlQuery create;
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(1,"+id_u+",0)");
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(2,"+id_u+",0)");
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(3,"+id_u+",0)");
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(4,"+id_u+",0)");
     create.exec("insert into bal_regime(id_reg,id_user,bal) value(5,"+id_u+",0)");
   }
   upd.clear();
   upd.exec("update bal_regime set bal=bal+"+table.value(0).toString()+" where id_r=3 && id_u"+id_u);
     }
     else {
   b->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b1->text()==knock) ui->b1->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b2->text()==knock) ui->b2->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b3->text()==knock) ui->b3->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b4->text()==knock) ui->b4->setStyleSheet("background-color: #ff6450; font: 12pt");
   if(ui->b5->text()==knock) ui->b5->setStyleSheet("background-color: #ff6450; font: 12pt");
     }
       }
       query.clear();
```

```
        knock="0";
    }
        }
    else koln--;
return b;
}
void ChooseCouple3::exit(){
    QTest::qWait(1000);
   emit gochoose();
}
void ChooseCouple3::on_b1_clicked()
{ui->b1=click_on_1_5(ui->b1);
    if(koln==10) exit();
 }
void ChooseCouple3::on_b2_clicked()
{ui->b2=click_on_1_5(ui->b2);
    if(koln==10) exit();
 }
void ChooseCouple3::on_b3_clicked()
{ui->b3=click_on_1_5(ui->b3);
    if(koln==10) exit();
 }
void ChooseCouple3::on_b4_clicked()
{ui->b4=click_on_1_5(ui->b4);
    if(koln==10) exit();
 }
void ChooseCouple3::on_b5_clicked()
{ui->b5=click_on_1_5(ui->b5);
    if(koln==10) exit();
 }
void ChooseCouple3::on_b6_clicked()
{ui->b6=click_on_6_10(ui->b6);
    if(koln==10) exit();
}
void ChooseCouple3::on_b7_clicked()
{ui->b7=click_on_6_10(ui->b7);
    if(koln==10) exit();
}
void ChooseCouple3::on_b8_clicked()
{   ui->b8=click_on_6_10(ui->b8);
    if(koln==10) exit();
}
void ChooseCouple3::on_b9_clicked()
{ui->b9=click_on_6_10(ui->b9);
    if(koln==10) exit();
}
void ChooseCouple3::on_b10_clicked()
{ui->b10=click_on_6_10(ui->b10);
    if(koln==10) exit();
}
void ChooseCouple3::play_sound(int index)
{
 QString word;
 if(index==1) word=ui->b1->text();
 if(index==2) word=ui->b2->text();
 if(index==3) word=ui->b3->text();
 if(index==4) word=ui->b4->text();
 if(index==5) word=ui->b5->text();
QTextToSpeech * speech= new QTextToSpeech;
QSettings * settings = new QSettings("C:/Users/Meshtli/Desktop/hi/Kursach/other/settings.conf",
QSettings::IniFormat);
qDebug()<<settings->value("sound/speed").toDouble()/100;
```

```
speech->setRate(settings->value("sound/speed").toDouble()/100);
speech->setVolume(settings->value("sound/volume").toDouble()/100);
speech->say(word);
}
void ChooseCouple3::on_sound1_clicked()
{
  play_sound(1);
}
void ChooseCouple3::on_sound2_clicked()
{
   play_sound(2);
}
void ChooseCouple3::on_sound3_clicked()
{
   play_sound(3);
}
void ChooseCouple3::on_sound4_clicked()
{
   play_sound(4);
}
void ChooseCouple3::on_sound5_clicked()
{
   play_sound(5);
}

#include "choosemode.h"
#include "ui_choosemode.h"
ChooseMode::ChooseMode(int id, int id_u, QWidget *parent) :
   QWidget(parent),
   ui(new Ui::ChooseMode)
{
   ui->setupUi(this);
   this->setWindowTitle("EnglishLerning");
   flag=0;
   id_user=id;
   id_theme=id_u;
   //update stopka set bal=0;
}
ChooseMode::~ChooseMode()
{
   delete ui;
}
void ChooseMode::messange(QString a)
{
   QMessageBox mes;
   mes.setWindowTitle("Error");
   mes.resize(200,100);
   mes.setText(a);
   mes.exec();
}
void ChooseMode::on_back_clicked()
{
   emit toolbar1();
}
void ChooseMode::chooseregime(int current)
{int otv=0;
   while(otv==0){
      int random=rand()%5;
      switch (random) {
      case 0: {
         if(random!=current-1) {
            otv=1;
```

```
        int i=0,j=0;
        if(current==1){
           i=wreg1->pos().x()+1; j=wreg1->pos().y()+38;
           wreg1->close();
        }
        if(current==2){
           i=wreg2->pos().x()+1; j=wreg2->pos().y()+38;
           wreg2->close();
        }
        if(current==3){
           i=wreg3->pos().x()+1; j=wreg3->pos().y()+38;
           wreg3->close();
        }
        if(current==4){
           i=wreg4->pos().x()+1; j=wreg4->pos().y()+38;
           wreg4->close();
        }
        if(current==5){
           i=wreg5->pos().x()+1; j=wreg5->pos().y()+38;
           wreg5->close();
        }
        wreg1 = new Collecting1(id_user, id_theme);
        wreg1->show();
        qDebug()<<i<<j;
        wreg1->setGeometry(i, j, 900, 500);
        connect(wreg1, &Collecting1::gochoose,this, &ChooseMode::getsignalregim1);
    }
    break;
}
case 1: {
    if(random!=current-1) {
        otv=1;
        int i=0,j=0;
        if(current==1){
           i=wreg1->pos().x()+1; j=wreg1->pos().y()+38;
           wreg1->close();
        }
        if(current==2){
           i=wreg2->pos().x()+1; j=wreg2->pos().y()+38;
           wreg2->close();
        }
        if(current==3){
           i=wreg3->pos().x()+1; j=wreg3->pos().y()+38;
           wreg3->close();
        }
        if(current==4){
           i=wreg4->pos().x()+1; j=wreg4->pos().y()+38;
           wreg4->close();
        }
        if(current==5){
           i=wreg5->pos().x()+1; j=wreg5->pos().y()+38;
           wreg5->close();
        }
        wreg2 = new ChooseTrans2(id_user, id_theme);
        wreg2->show();
        qDebug()<<i<<j;
        wreg2->setGeometry(i, j, 900, 500);
        connect(wreg2, &ChooseTrans2::gochoose,this, &ChooseMode::getsignalregim2);
    }
    break;
}
case 2: {
```

```
    if(random!=current-1) {
        otv=1;
        int i=0,j=0;
        if(current==1){
            i=wreg1->pos().x()+1; j=wreg1->pos().y()+38;
            wreg1->close();
        }
        if(current==2){
            i=wreg2->pos().x()+1; j=wreg2->pos().y()+38;
            wreg2->close();
        }
        if(current==3){
            i=wreg3->pos().x()+1; j=wreg3->pos().y()+38;
            wreg3->close();
        }
        if(current==4){
            i=wreg4->pos().x()+1; j=wreg4->pos().y()+38;
            wreg4->close();
        }
        if(current==5){
            i=wreg5->pos().x()+1; j=wreg5->pos().y()+38;
            wreg5->close();
        }
        wreg3 = new ChooseCouple3(id_user, id_theme);
        qDebug()<<i<<j;
        wreg3->setGeometry(i, j, 900, 500);
        connect(wreg3, &ChooseCouple3::gochoose,this, &ChooseMode::getsignalregim3);
        wreg3->show();
    }
    break;
}
case 3: {
    if(random!=current-1) {
        otv=1;
        int i=0,j=0;
        if(current==1){
            i=wreg1->pos().x()+1; j=wreg1->pos().y()+38;
            wreg1->close();
        }
        if(current==2){
            i=wreg2->pos().x()+1; j=wreg2->pos().y()+38;
            wreg2->close();
        }
        if(current==3){
            i=wreg3->pos().x()+1; j=wreg3->pos().y()+38;
            wreg3->close();
        }
        if(current==4){
            i=wreg4->pos().x()+1; j=wreg4->pos().y()+38;
            wreg4->close();
        }
        if(current==5){
            i=wreg5->pos().x()+1; j=wreg5->pos().y()+38;
            wreg5->close();
        }
        wreg4 = new HighLight4(id_user, id_theme);
        wreg4->show();
        qDebug()<<i<<j;
        wreg4->setGeometry(i, j, 900, 500);
        connect(wreg4, &HighLight4::gochoose,this, &ChooseMode::getsignalregim4);
    }
    break;
```

```
        }
        case 4: {
           if(random!=current-1) {
              otv=1;
              int i=0,j=0;
              if(current==1){
                 i=wreg1->pos().x()+1; j=wreg1->pos().y()+38;
                 wreg1->close();
              }
              if(current==2){
                 i=wreg2->pos().x()+1; j=wreg2->pos().y()+38;
                 wreg2->close();
              }
              if(current==3){
                 i=wreg3->pos().x()+1; j=wreg3->pos().y()+38;
                 wreg3->close();
              }
              if(current==4){
                 i=wreg4->pos().x()+1; j=wreg4->pos().y()+38;
                 wreg4->close();
              }
              if(current==5){
                 i=wreg5->pos().x()+1; j=wreg5->pos().y()+38;
                 wreg5->close();
              }
              wreg5 = new ChooseCor5(id_user, id_theme);
              wreg5->show();
              qDebug()<<i<<j;
              wreg5->setGeometry(i, j, 900, 500);
              connect(wreg5, &ChooseCor5::gochoose,this, &ChooseMode::getsignalregim5);
           }
           break;
        }
        }
    }
}
void ChooseMode::getsignalregim3()
{
    QSqlQuery query;
    query.exec("select count(*) from stopka where id_user="+QString::number(id_user)+"&& bal>20");
    query.next();
    if(query.value(0).toInt()<10){
        if(flag==1)
        {
            chooseregime(3);
        }
        else{qDebug()<<"sadsd";
            int i=wreg3->pos().x(), j=wreg3->pos().y();
            wreg3->close();
            wreg3 = new ChooseCouple3(id_user, id_theme);
            wreg3->show();
            wreg3->setGeometry(i, j, 900, 500);
            connect(wreg3, &ChooseCouple3::gochoose,this, &ChooseMode::getsignalregim3);
        }
    }
    else
    {//do
    }
}
void ChooseMode::getsignalregim5()
{
    QSqlQuery query;
```

```
    query.exec("select count(*) from stopka where id_user="+QString::number(id_user)+"&& bal>20");
    query.next();
    if(query.value(0).toInt()<10){
        if(flag==1)
        {
            chooseregime(5);
        }
        else{qDebug()<<"sadsd";
            int i=wreg5->pos().x(), j=wreg5->pos().y();
            wreg5->close();
            wreg5 = new ChooseCor5(id_user, id_theme);
            wreg5->show();
            wreg5->setGeometry(i, j, 900, 500);
            connect(wreg5, &ChooseCor5::gochoose,this, &ChooseMode::getsignalregim5);
        }
    }
    else
    {//do
    }
}
void ChooseMode::getsignalregim4()
{
    QSqlQuery query;
    query.exec("select count(*) from stopka where id_user="+QString::number(id_user)+"&& bal>20");
    query.next();
    if(query.value(0).toInt()<10){
        if(flag==1)
        {
            chooseregime(4);
        }
        else{qDebug()<<"sadsd";
            int i=wreg4->pos().x(), j=wreg4->pos().y();
            wreg4->close();
            wreg4 = new HighLight4(id_user, id_theme);
            wreg4->show();
            wreg4->setGeometry(i, j, 900, 500);
            connect(wreg4, &HighLight4::gochoose,this, &ChooseMode::getsignalregim4);
        }
    }
    else
    {//do
    }
}
void ChooseMode::getsignalregim1()
{
    QSqlQuery query;
    query.exec("select count(*) from stopka where id_user="+QString::number(id_user)+"&& bal>20");
    query.next();
    if(query.value(0).toInt()<10){
        if(flag==1)
        {
            chooseregime(1);
        }
        else{qDebug()<<"sadsd";
            int i=wreg1->pos().x(), j=wreg1->pos().y();
            wreg1->close();
            wreg1 = new Collecting1(id_user, id_theme);
            wreg1->show();
            wreg1->setGeometry(i, j, 900, 500);
            connect(wreg1, &Collecting1::gochoose,this, &ChooseMode::getsignalregim1);
        }
    }
```

```
    else
    {this->show();
        this->setGeometry(wreg1->pos().x()+1, wreg1->pos().y()+38, 900, 500);
        wreg1->close();
        flag=0;
    }
}
void ChooseMode::getsignalregim2()
{
    QSqlQuery query;
    query.exec("select count(*) from stopka where id_user="+QString::number(id_user)+"&& bal>20");
    query.next();
    if(query.value(0).toInt()<10){
        if(flag==1)
        {
            chooseregime(2);
        }
        else{qDebug()<<"sadsd";
            int i=wreg2->pos().x(), j=wreg2->pos().y();
            wreg2->close();
            wreg2 = new ChooseTrans2(id_user, id_theme);
            wreg2->show();
            wreg2->setGeometry(i, j, 900, 500);
            connect(wreg2, &ChooseTrans2::gochoose,this, &ChooseMode::getsignalregim2);
        }
    }
    else
    {this->show();
        this->setGeometry(wreg2->pos().x()+1, wreg2->pos().y()+38, 900, 500);
        wreg2->close();
        flag=0;
    }
}
void ChooseMode::on_reg1_clicked()
{
    wreg1 = new Collecting1(id_user, id_theme);
    connect(wreg1, &Collecting1::gochoose,this, &ChooseMode::getsignalregim1);
    wreg1->setGeometry(this->pos().x()+1, this->pos().y()+38, 900, 500);
    wreg1->show();
    this->close();
}
void ChooseMode::on_reg2_clicked()
{
    wreg2 = new ChooseTrans2(id_user, id_theme);
    connect(wreg2, &ChooseTrans2::gochoose,this, &ChooseMode::getsignalregim2);
    wreg2->show();
    wreg2->setGeometry(this->pos().x()+1, this->pos().y()+38, 900, 500);
    this->close();
}
void ChooseMode::on_reg3_clicked()
{
    wreg3 = new ChooseCouple3(id_user, id_theme);
    connect(wreg3, &ChooseCouple3::gochoose,this, &ChooseMode::getsignalregim3);
     wreg3->setGeometry(this->pos().x()+1, this->pos().y()+38, 900, 500);
    wreg3->show();
    this->close();
}
void ChooseMode::on_reg4_clicked()
{
    wreg4 = new HighLight4(id_user, id_theme);
    connect(wreg4, &HighLight4::gochoose,this, &ChooseMode::getsignalregim4);
    wreg4->setGeometry(this->pos().x()+1, this->pos().y()+38, 900, 500);
```

```
    wreg4->show();
    this->close();
}
void ChooseMode::on_reg5_clicked()
{
    wreg5 = new ChooseCor5(id_user, id_theme);
    connect(wreg5, &ChooseCor5::gochoose,this, &ChooseMode::getsignalregim5);
        wreg5->setGeometry(this->pos().x()+1, this->pos().y()+38, 900, 500);
    wreg5->show();
    this->close();
}
void ChooseMode::setword(int id)
{   id_user=id;
}
void ChooseMode::on_reg6_clicked()
{struct srandom{srandom(){srand(time(NULL));}}orandom;
    flag=1;
    switch(random)
    {
    case 0: {
        on_reg1_clicked();
        break;
    }
    case 1: {
        on_reg2_clicked();
        break;
    }
    case 2: {
        on_reg3_clicked();
        break;
    }
    case 3: {
        on_reg4_clicked();
        break;
    }
    case 4: {
        on_reg5_clicked();
        break;
    }
    }
}

 #include "choosetrans2.h"
#include "ui_choosetrans2.h"
ChooseTrans2::ChooseTrans2(int id,  int id_c, QWidget *parent) :
    QWidget(parent),
    ui(new Ui::ChooseTrans2)
{struct srandom{srandom(){srand(time(NULL));}}orandom;
    ui->setupUi(this);
    setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
    setWindowTitle("LerningEnglish");
    count_word=0;
    id_user=id;
    id_theme=id_c;
this->setWindowTitle("EnglishLerning");
 ui->skip->setStyleSheet("border-image: url(C:/Users/Meshtli/Desktop/hi/Kursach/other/skip.png) stretch;");
    ui->progressBar->setRange(0,100);
    QSqlQuery progress;
    progress.exec("select count(*) from stopka where id_user="+QString::number(id)+") && id_word in (select id_word
from having_c where id_card="+QString::number(id_theme)+") && bal>20");
    progress.next();
     ui->progressBar->setValue(progress.value(0).toInt()*10);
```

```
    if(progress.value(0).toInt()*10==0)
   ui->progressBar->setStyleSheet("QProgressBar { color: rgb(255, 252, 237);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
else ui->progressBar->setStyleSheet("QProgressBar { color: rgb(76, 39, 19);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
   QString id_u;
   id_u.setNum(id_user);
   QSqlQuery query;
   int kol=10;
   query.clear();
   int flag=0, count=0;
   QString trans, word;
   do{count++;
      query.exec("select id_w, eng, rus from word where id_w in (select id_word from stopka where id_user="+id_u+")
&& id_w in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
      while(query.next() && flag!=-1 )
      {int ran=rand()% kol;
         QSqlQuery bal;
         bal.exec("select bal from stopka where id_word="+query.value(2).toString());
         bal.next();
         if(ran==0 && bal.value(0)<20)  {
            flag=-1;
            trans=query.value(2).toString();
            word=query.value(1).toString();
            id_w=query.value(0).toInt();
                  }
      }
      query.clear();
   }while(flag!=-1 && count<200);
ui->translate->setText(trans);
ui->translate->setAlignment(Qt::AlignCenter);
   QString word_id;
   word_id.setNum(id_w);
   QFileInfo check_file("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
   if(!check_file.exists())
   {
ui->widget_ph->hide();
int random=rand()%5;
if(random==0)    ui->b1->setText(word);
else ui->b1=changetext(ui->b1, kol, id_u);
if(random==1)    ui->b2->setText(word);
else ui->b2=changetext(ui->b2, kol, id_u);
if(random==2)    ui->b3->setText(word);
else ui->b3=changetext(ui->b3, kol, id_u);
if(random==3)    ui->b4->setText(word);
else ui->b4=changetext(ui->b4, kol, id_u);
if(random==4)    ui->b5->setText(word);
else ui->b5=changetext(ui->b5, kol, id_u);
   }
   else
   {
      QPixmap pict("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
      QPixmap p;
      p.load("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
      int w = ui->image->width();
      int h = ui->image->height();
      ui->image->setPixmap(p.scaled(w,h,Qt::KeepAspectRatio));
ui->widget_noph->hide();
int random=rand()%5;
if(random==0)    ui->b1_ph->setText(word);
else ui->b1_ph=changetext(ui->b1_ph, kol, id_u);
if(random==1)    ui->b2_ph->setText(word);
```

```
else ui->b2_ph=changetext(ui->b2_ph, kol, id_u);
if(random==2)    ui->b3_ph->setText(word);
else ui->b3_ph=changetext(ui->b3_ph, kol, id_u);
if(random==3)    ui->b4_ph->setText(word);
else ui->b4_ph=changetext(ui->b4_ph, kol, id_u);
if(random==4)    ui->b5_ph->setText(word);
else ui->b5_ph=changetext(ui->b5_ph, kol, id_u);
   }
}
QPushButton * ChooseTrans2::changetext(QPushButton * b, int kol, QString id_u)
{QSqlQuery query, id;
   int flag=0;
   do{
     query.exec("select eng from word where id_w in (select id_word from stopka where id_user="+id_u+") && id_w
in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
      while(query.next() && flag!=-1 )
      {int ran=rand()% kol;
         if(ran==0)  {
            int otv=0;
            QString word;
            word.setNum(id_w);
            id.exec("select eng from word where id_w="+word);
            id.next();
              for(int i=0;i<count_word;i++)
                { //qDebug()<<query.value(0).toString()<<id.value(0).toString();
               if(wordatbutton[i]==query.value(0).toString() || query.value(0).toString()==id.value(0).toString()) otv=1;
             }
               if(otv==0){
               b->setText(query.value(0).toString());
               flag=-1;
               wordatbutton[count_word]=query.value(0).toString();
               count_word++;
             }
            id.clear();
         }
      }
      query.clear();
   }while(flag!=-1 );
   return b;
}
ChooseTrans2::~ChooseTrans2()
{
   delete ui;
}
QPushButton * ChooseTrans2::onbuttonclick(QPushButton * b)
{    QSqlQuery query, upd;
         query.exec("select eng from word where id_w ="+QString::number(id_w));
         query.next();
         if(query.value(0).toString()==b->text()){
            QSqlQuery table;
            table.exec("select score from regime where id_r=2");
            table.next();
            upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where
id_word="+QString::number(id_w)+" && id_user="+QString::number(id_user));
            upd.clear();
            upd.exec("select count(*) from bal_regime where id_user="+QString::number(id_user));
            upd.next();
            if(upd.value(0).toInt()==0)
            {
              QSqlQuery create;
              create.exec("insert into bal_regime(id_reg,id_user,bal) value(1,"+QString::number(id_user)+",0)");
              create.exec("insert into bal_regime(id_reg,id_user,bal) value(2,"+QString::number(id_user)+",0)");
```

```
            create.exec("insert into bal_regime(id_reg,id_user,bal) value(3,"+QString::number(id_user)+",0)");
            create.exec("insert into bal_regime(id_reg,id_user,bal) value(4,"+QString::number(id_user)+",0)");
            create.exec("insert into bal_regime(id_reg,id_user,bal) value(5,"+QString::number(id_user)+",0)");
          }
          upd.clear();
          upd.exec("update bal_regime set bal=bal+"+table.value(0).toString()+" where id_r=2 &&
id_u"+QString::number(id_user));
          b->setStyleSheet("background-color: #05cb23; font: 12pt");
        }
        else
        {
          b->setStyleSheet("background-color: #ff6450; font: 12pt");
        }
      return b;
}
void ChooseTrans2::exit(){
   QTest::qWait(1000);
  emit gochoose();
}
void ChooseTrans2::on_b1_clicked()
{
   ui->b1=onbuttonclick(ui->b1);
   exit();
}
void ChooseTrans2::on_b2_clicked()
{
ui->b2=onbuttonclick(ui->b2);
exit();
}
void ChooseTrans2::on_b3_clicked()
{
ui->b3=onbuttonclick(ui->b3);
exit();
}
void ChooseTrans2::on_b4_clicked()
{
ui->b4=onbuttonclick(ui->b4);
exit();
}
void ChooseTrans2::on_b5_clicked()
{
ui->b5=onbuttonclick(ui->b5);
exit();
}
void ChooseTrans2::on_b1_ph_clicked()
{
   ui->b1_ph=onbuttonclick(ui->b1_ph);
   exit();
}
void ChooseTrans2::on_b2_ph_clicked()
{
   ui->b2_ph=onbuttonclick(ui->b2_ph);
   exit();
}
void ChooseTrans2::on_b3_ph_clicked()
{
   ui->b3_ph=onbuttonclick(ui->b3_ph);
   exit();
}
void ChooseTrans2::on_b4_ph_clicked()
{
   ui->b4_ph=onbuttonclick(ui->b4_ph);
```

```
    exit();
}
void ChooseTrans2::on_b5_ph_clicked()
{
    ui->b5_ph=onbuttonclick(ui->b5_ph);
    exit();
}

#include "collecting1.h"
#include "ui_collecting1.h"
Collecting1::Collecting1(int id,  int id_c, QWidget *parent) :
    QWidget(parent),
    ui(new Ui::Collecting1)
{struct srandom{srandom(){srand(time(NULL));}}orandom;
id_theme=id_c;
    ui->setupUi(this);
    setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
    setWindowTitle("LerningEnglish");
    this->setWindowTitle("EnglishLerning");
ui->skip->setStyleSheet("border-image: url(C:/Users/Meshtli/Desktop/hi/Kursach/other/skip.png) stretch;");
    ui->cor_photo->hide();
    id_user=id;
    position=0;
    ui->progressBar->setRange(0,100);
    QSqlQuery progress;
    progress.exec("select count(*) from stopka where id_user="+QString::number(id)+") && id_word in (select id_word
from having_c where id_card="+QString::number(id_theme)+") && bal>20");
    progress.next();
     ui->progressBar->setValue(progress.value(0).toInt()*10);
    if(progress.value(0).toInt()*10==0)
    ui->progressBar->setStyleSheet("QProgressBar { color: rgb(255, 252, 237);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
else ui->progressBar->setStyleSheet("QProgressBar { color: rgb(76, 39, 19);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
     QString id_u, word;
     id_u.setNum(id);
     QSqlQuery query;
     int kol=10;
     query.clear();
     int flag=0, id_word, size_w, count=0;
     QString trans;
     do{count++;
        query.exec("select eng, rus, id_w from word where id_w in (select id_word from stopka where id_user="+id_u+")
&& id_w in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
        while(query.next() && flag!=-1 )
        {int ran=rand()% kol;
           QSqlQuery bal;
           bal.exec("select bal from stopka where id_word="+query.value(2).toString());
           bal.next();
           if(ran==0 && bal.value(0)<20 && query.value(0).toString().size()<=16)  {
              flag=-1;
              trans=query.value(1).toString();
              word=query.value(0).toString();
              id_word=query.value(2).toInt();
                      }
        }
        query.clear();
    }while(flag!=-1 && count<200);
    if(flag==-1){
        id_word_g=id_word;
    size_w=word.size();
    touch=new int[size_w];
```

```
    ui->vvod->setText(trans);
    ui->vvod->setAlignment(Qt::AlignCenter);
    QString
alpha[26]={"A","B","C","D","E","F","G","H","I","J","K","L","M","N","O","P","Q","R","S","T","U","V","W","X","Y"
,"Z"};
    QString mas[4][4];
    int ** tochki=new int*[size_w];
    for(int j=0; j<size_w; j++)
       tochki[j]=new int[2];
    for(int i=0;i<4;i++)
       for(int j=0;j<4;j++)
          mas[i][j]="0";
    for(int i=0; i<size_w; i++)
    {int otv;
       do{
       otv=0;
       int t1,t2;
       t1=rand()%4;
       t2=rand()%4;

       if(i>0)
       for(int j=0; j<i; j++)
       if(tochki[j][0]==t1 && tochki[j][1]==t2) otv=1;
          if(otv==0) {
          tochki[i][0]=t1;
          tochki[i][1]=t2;
       }
}while(otv==1);
    }
     for(int i=0; i<size_w; i++)
        mas[tochki[i][0]][tochki[i][1]]=word[i].toUpper();
   for(int i=0; i<4; i++)
      for(int j=0; j<4; j++)
      {
         if(mas[i][j]=="0") {
            int ran=rand()%26;
            mas[i][j]=alpha[ran];
         }
      }
   QString word_id;
   word_id.setNum(id_word);
   QFileInfo check_file("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
   if(check_file.exists())
   {       ui->widget_noph->hide();
      QPixmap p;
      p.load("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
      int w = ui->image->width();
      int h = ui->image->height();
      ui->image->setPixmap(p.scaled(w,h,Qt::KeepAspectRatio));
      ui->Grid->setSpacing(0);
      for(int i=0; i<4;i++)
      {
         for(int j=0; j<4;j++)
         {
            ui->Grid->addWidget(setbutton_wp(mas[i][j]),i,j);
         }
      }
   }
   else
   {ui->widget_ph->hide();
      ui->Grid->setSpacing(0);
      for(int i=0; i<4;i++)
```

```
        {
            for(int j=0; j<4;j++)
            {
                ui->Grid->addWidget(setbutton(mas[i][j]),i,j);
            }
        }
    }
    for(int j=0; j<size_w; j++)
        delete[]tochki[j];
    delete[]tochki;
     }
}
void Collecting1::messange(QString a)
{
    QMessageBox mes;
  mes.setWindowTitle("Error");
    mes.resize(150,100);
    mes.setText(a);
    mes.exec();
}
void Collecting1::signalbutton()
{   int index=ui->Grid->indexOf(static_cast<QPushButton*>(sender()));
    QString word = static_cast<QPushButton*>(sender())->text();
    ui->vuvod_noph->setText(ui->vuvod_noph->toPlainText()+word);
    ui->vuvod_noph->setAlignment(Qt::AlignCenter);
    static_cast<QPushButton*>(sender())->hide();
    touch[position]=index;
    position++;
    }
QPushButton *Collecting1::setbutton(QString text)
{
    QPushButton * button = new QPushButton;
    connect(button,&QPushButton::clicked,this,&Collecting1::signalbutton);
    button->setText(text);
    QSizePolicy pls;
    pls.setVerticalPolicy(QSizePolicy::Expanding);
    pls.setHorizontalPolicy(QSizePolicy::Expanding);
    pls.setRetainSizeWhenHidden(true);
    button->setSizePolicy(pls);
    button->setStyleSheet("background-color: rgb(212, 233, 217); color: rgb(37, 89, 67); font: 16pt");
    return button;
}
void Collecting1::signalbutton_wp()
{   int index=ui->Grid->indexOf(static_cast<QPushButton*>(sender()));
    QString word = static_cast<QPushButton*>(sender())->text();
    ui->vuvod_ph->setText(ui->vuvod_ph->toPlainText()+word);
    ui->vuvod_ph->setAlignment(Qt::AlignCenter);
    static_cast<QPushButton*>(sender())->hide();
    touch[position]=index;
    position++;
    }
QPushButton *Collecting1::setbutton_wp(QString text)
{
    QPushButton * button = new QPushButton;
    connect(button,&QPushButton::clicked,this,&Collecting1::signalbutton_wp);
    button->setText(text);
    QSizePolicy pls;
    pls.setVerticalPolicy(QSizePolicy::Expanding);
    pls.setHorizontalPolicy(QSizePolicy::Expanding);
    pls.setRetainSizeWhenHidden(true);
    button->setSizePolicy(pls);
    button->setStyleSheet("background-color: rgb(212, 233, 217); color: rgb(37, 89, 67); font: 16pt");
```

```
    return button;
}
Collecting1::~Collecting1()
{
    delete ui;
}
void Collecting1::on_go_out_clicked()
{
    QString id_u;
        int cor=0;
        id_u.setNum(id_user);
        QSqlQuery query, upd;
        QString word_id;
        word_id.setNum(id_word_g);
        query.exec("select id_w, eng, rus from word where id_w in (select id_word from stopka where id_user="+id_u+")
&& id_w in (select id_word from having_c where id_card="+QString::number(id_theme)+")");
        QFileInfo check_file("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
        if(check_file.exists()){
        while(query.next() )
        {
            if(query.value(0).toInt()==id_word_g){
                QString a=query.value(1).toString(), b=ui->vuvod_ph->toPlainText();
                a=a.toLower();
                b=b.toLower();
                qDebug()<<a<<b;
                if(a==b){cor=1;
                    QString rab;
                    QSqlQuery table;
                    rab.setNum(id_word_g);
                    table.exec("select score from regime where id_r=1");
                    table.next();
                    upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where id_word="+rab+" &&
id_user="+QString::number(id_user));
                    upd.clear();
                    upd.exec("select count(*) from bal_regime where id_user="+QString::number(id_user));
                    upd.next();
                    if(upd.value(0).toInt()==0)
                    {
                      QSqlQuery create;
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(1,"+QString::number(id_user)+",0)");
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(2,"+QString::number(id_user)+",0)");
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(3,"+QString::number(id_user)+",0)");
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(4,"+QString::number(id_user)+",0)");
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(5,"+QString::number(id_user)+",0)");
                    }
                    upd.clear();
                    upd.exec("update bal_regime set bal=bal+"+table.value(0).toString()+" where id_r=1 &&
id_u"+QString::number(id_user));
            }
        }
        }
        }
        else
        {
            while(query.next() )
            {
                if(query.value(0).toInt()==id_word_g){
                    QString a=query.value(1).toString(), b=ui->vuvod_noph->toPlainText();
                    a=a.toLower();
                    b=b.toLower();
                    if(a==b){cor=1;
                        QString rab;
```

```
                    QSqlQuery table;
                    rab.setNum(id_word_g);
                    table.exec("select score from regime where id_r=1");
                    table.next();
                    upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where id_word="+rab+" &&
id_user="+QString::number(id_user));
                    upd.clear();
                    upd.exec("select count(*) from bal_regime where id_user="+QString::number(id_user));
                    upd.next();
                    if(upd.value(0).toInt()==0)
                    {
                      QSqlQuery create;
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(1,"+QString::number(id_user)+",0)");
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(2,"+QString::number(id_user)+",0)");
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(3,"+QString::number(id_user)+",0)");
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(4,"+QString::number(id_user)+",0)");
                      create.exec("insert into bal_regime(id_reg,id_user,bal) value(5,"+QString::number(id_user)+",0)");
                    }
                    upd.clear();
                    upd.exec("update bal_regime set bal=bal+"+table.value(0).toString()+" where id_r=1 &&
id_u"+QString::number(id_user));
if(query.value(2).toInt()+table.value(0).toInt()>=20) {
id_c =(select id_card from having_c where id_word="+rab+")");
history(id_word,id_reg,date_walk) value("+rab+",1,'"+dates+"')");
//                    }
                }
              }
            }
        }
    query.clear();
    QPixmap pict;
       if(cor==1) pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/correct.png");
       else  pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/nocorrect.png");
       ui->cor_photo->setPixmap(pict);
       ui->cor_photo->show();
     QTest::qWait(1500);
    emit gochoose();
}
void Collecting1::on_back_ph_clicked()
{
    if(position>0){
       position--;
       ui->Grid->itemAt(touch[position])->widget()->show();
       QString str =ui->vuvod_ph->toPlainText();
       str.chop(1);
       ui->vuvod_ph->setText(str);
       ui->vuvod_ph->setAlignment(Qt::AlignCenter);
       }
}
void Collecting1::on_back_noph_clicked()
{
    if(position>0){
       position--;
       ui->Grid->itemAt(touch[position])->widget()->show();
       QString str =ui->vuvod_noph->toPlainText();
       str.chop(1);
       ui->vuvod_noph->setText(str);
       ui->vuvod_noph->setAlignment(Qt::AlignCenter);
       }
}
void Collecting1::on_skip_clicked()
{
```

```
        emit gochoose();
}

#include "filter.h"
filter::filter(QObject *parent) : QObject(parent)
{
}
bool filter::eventFilter(QObject *watched, QEvent *event)
{
   if(event->type() == QEvent::KeyPress)
   {
        QKeyEvent * kEvent = static_cast<QKeyEvent*>(event);

        if((kEvent->key() >= Qt::Key_A && kEvent->key() <= Qt::Key_Z) || kEvent->key() == Qt::Key_Backspace)
        {
           return false;
        }
       return true;
   }
  return false;
}
#include "filter_rus.h"

filter_rus::filter_rus(QObject *parent) : QObject(parent)
{
}
bool filter_rus::eventFilter(QObject *watched, QEvent *event)
{
   if(event->type() == QEvent::KeyPress)
   {
        QKeyEvent * kEvent = static_cast<QKeyEvent*>(event);

        if((kEvent->key() >= 1040 && kEvent->key() <= 1071) || kEvent->key() == Qt::Key_Backspace)
        {
           return false;
        }
       return true;
   }
  return false;
}

#include "highlight4.h"
#include "ui_highlight4.h"
struct srandom{srandom(){srand(time(NULL));}}orandom;
HighLight4::HighLight4(int id, int id_c, QWidget *parent) :
   QWidget(parent),
   ui(new Ui::HighLight4)
{id_theme=id_c;
   ui->setupUi(this);
   setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
   setWindowTitle("LerningEnglish");
ui->skip->setStyleSheet("border-image: url(C:/Users/Meshtli/Desktop/hi/Kursach/other/skip.png) stretch;");
   ui->cor_photo->hide();
   QPalette palette=ui->progressBar->palette();
this->setWindowTitle("EnglishLerning");
   ui->progressBar->setRange(0,100);
   QSqlQuery progress;
   progress.exec("select count(*) from stopka where id_user="+QString::number(id)+") && id_word in (select id_word
from having_c where id_card="+QString::number(id_theme)+") && bal>20");
   progress.next();
    ui->progressBar->setValue(progress.value(0).toInt()*10);
   if(progress.value(0).toInt()*10==0)
```

```
    ui->progressBar->setStyleSheet("QProgressBar { color: rgb(255, 252, 237);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
else ui->progressBar->setStyleSheet("QProgressBar { color: rgb(76, 39, 19);  background-color:  rgb(255, 252, 237);
border: 2px solid grey; border-radius: 5px;} QProgressBar::chunk {background-color: rgb(76, 39, 19);}");
    int count=0;
    id_user=id;
    QString id_t, word;
    QSqlQuery query;
    int kol=10;
    int flag=0, id_word, size_w;
    QString trans;
    do{
        query.exec("select id_w, eng, rus from word where id_w in (select id_word from stopka where
id_user="+QString::number(id_user)+") && id_w in (select id_word from having_c where
id_card="+QString::number(id_theme)+")");
        while(query.next() && flag!=-1 )
        {int ran=rand()% kol;
            QSqlQuery bal;
            bal.exec("select bal from stopka where id_word="+query.value(2).toString());
            bal.next();
            if(ran==0 && bal.value(0)<20)  {
                flag=-1;
                trans=query.value(2).toString();
                word=query.value(1).toString();
                id_word=query.value(0).toInt();
                id_word_g=id_word;
            }
        }
        query.clear();
    }while(flag!=-1);
    size_w=word.size();
    ui->vvod->setText(trans);
    ui->vvod->setAlignment(Qt::AlignCenter);
    touch=new int[size_w];
    position=0;
    QString
alpha[26]={"A","B","C","D","E","F","G","H","I","J","K","L","M","N","O","P","Q","R","S","T","U","V","W","X","Y"
,"Z"};
    QString mas[5][5];
    int ** tochki=new int*[size_w];
    for(int j=0; j<size_w; j++)
        tochki[j]=new int[2];
    while(count==200 || count==0){
        count=0;
        for(int i=0;i<5;i++)
            for(int j=0;j<5;j++)
                mas[i][j]="0";
        int nach1=rand()%5, nach2=rand()%5;
        mas[nach1][nach2]=word[0];
        int i=1;
        tochki[0][0]=nach1;
        tochki[0][1]=nach2;
        //qDebug()<<nach1<<nach2;
        int t_i=1;
        while(i<size_w && count!=200)
        {int random=rand()%4;
            count++;
          //  qDebug()<<count<<word;
            switch(random)
            {
            case 0: {
                int otv=0;
```

```
            for(int j=0;j<=t_i;j++)
                if(nach1+1==tochki[j][0] && nach2==tochki[j][1]) otv=1;
            if(nach1+1<5 && otv==0){
                nach1++;
               // qDebug()<<nach1<<nach2;
                mas[nach1][nach2]=word[i].toUpper();
                tochki[t_i][0]=nach1;
                tochki[t_i][1]=nach2;
                i++; t_i++;
            }
            break;
        }
        case 1: {
            int otv=0;
            for(int j=0;j<=t_i;j++)
                if(nach1-1==tochki[j][0] && nach2==tochki[j][1]) otv=1;
            if(nach1-1>=0 && otv==0){
                nach1--;
                mas[nach1][nach2]=word[i].toUpper();
               // qDebug()<<nach1<<nach2;
                tochki[t_i][0]=nach1;
                tochki[t_i][1]=nach2;
                i++; t_i++;
            }
            break;
        }
        case 2: {
            int otv=0;
            for(int j=0;j<=t_i;j++)
                if(nach1==tochki[j][0] && nach2+1==tochki[j][1]) otv=1;
            if(nach2+1<5 && otv==0){
                nach2++;
               // qDebug()<<nach1<<nach2;
                mas[nach1][nach2]=word[i].toUpper();
                tochki[t_i][0]=nach1;
                tochki[t_i][1]=nach2;
                i++; t_i++;
            }
            break;
        }
        case 3: {
            int otv=0;
            for(int j=0;j<=t_i;j++)
                if(nach1==tochki[j][0] && nach2-1==tochki[j][1]) otv=1;
            if(nach2-1>=0 && otv==0){
                nach2--;
                //qDebug()<<nach1<<nach2;
                mas[nach1][nach2]=word[i].toUpper();
                tochki[t_i][0]=nach1;
                tochki[t_i][1]=nach2;
                i++; t_i++;
            }
            break;
        }
        }
    }
    count++;
}
for(int i=0; i<5; i++)
    for(int j=0; j<5; j++)
    {
        if(mas[i][j]=="0") {
```

```
            int ran=rand()%26;
            mas[i][j]=alpha[ran];
        }
    }
    QString word_id;
    word_id.setNum(id_word);
    QFileInfo check_file("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
    if(check_file.exists())
    {
        ui->widget_noph->hide();
        QPixmap p;
        p.load("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
        int w = ui->image->width();
        int h = ui->image->height();
        ui->image->setPixmap(p.scaled(w,h,Qt::KeepAspectRatio));
        ui->Grid->setSpacing(0);
        for(int i=0; i<5;i++)
        {
            for(int j=0; j<5;j++)
            {
                ui->Grid->addWidget(setbutton_wp(mas[i][j]),i,j);
            }
        }
    }
    else
    {ui->widget_ph->hide();
        ui->Grid->setSpacing(0);
        for(int i=0; i<5;i++)
        {
            for(int j=0; j<5;j++)
            {
                ui->Grid->addWidget(setbutton(mas[i][j]),i,j);
            }
        }
    }
    for(int j=0; j<size_w; j++)
        delete[]tochki[j];
    delete[]tochki;

}
HighLight4::~HighLight4()
{    delete ui;
}
void HighLight4::signalbutton()
{int index=ui->Grid->indexOf(static_cast<QPushButton*>(sender()));
    if(static_cast<QPushButton*>(sender())->styleSheet()!="color: rgb(255, 255, 255); background-color: rgb(74, 123,
101); font: 16pt"){
        if(touch[position-1]-1==index || touch[position-1]+1==index || touch[position-1]-5==index || touch[position-
1]+5==index || position==0)
        {
            QString word = static_cast<QPushButton*>(sender())->text();
            ui->vuvod_noph->setText(ui->vuvod_noph->toPlainText()+word);
            ui->vuvod_noph->setAlignment(Qt::AlignCenter);
            static_cast<QPushButton*>(sender())->setStyleSheet("background-color: rgb(212, 233, 217); color: rgb(37, 89,
67); font: 16pt");
            touch[position]=index;
            position++;
        }
    }
    else {
        if(touch[position-1]==index)
        {
```

```
            QString str =ui->vuvod_noph->toPlainText();
            str.chop(1);
            ui->vuvod_noph->setText(str);
            ui->vuvod_noph->setAlignment(Qt::AlignCenter);
            static_cast<QPushButton*>(sender())->setStyleSheet("background-color: rgb(212, 233, 217); color: rgb(37, 89,
67); font: 16pt");
            position--;
        }
    }
}
QPushButton *HighLight4::setbutton(QString text)
{    QPushButton * button = new QPushButton;
 connect(button,&QPushButton::clicked,this,&HighLight4::signalbutton);
    button->setText(text);
    QSizePolicy pls;
    pls.setVerticalPolicy(QSizePolicy::Expanding);
    pls.setHorizontalPolicy(QSizePolicy::Expanding);
    button->setSizePolicy(pls);
    button->setStyleSheet("background-color: rgb(212, 233, 217); color: rgb(37, 89, 67); font: 16pt");
    return button;
}
void HighLight4::signalbutton_wp()
{int index=ui->Grid->indexOf(static_cast<QPushButton*>(sender()));
    if(static_cast<QPushButton*>(sender())->styleSheet()!="color: rgb(255, 255, 255); background-color: rgb(74, 123,
101); font: 16pt"){
        if(touch[position-1]-1==index || touch[position-1]+1==index || touch[position-1]-5==index || touch[position-
1]+5==index || position==0)
        {           QString word = static_cast<QPushButton*>(sender())->text();
            ui->vuvod_ph->setText(ui->vuvod_ph->toPlainText()+word);
            ui->vuvod_ph->setAlignment(Qt::AlignCenter);
            static_cast<QPushButton*>(sender())->setStyleSheet("color: rgb(255, 255, 255); background-color: rgb(74,
123, 101); font: 16pt");
            touch[position]=index;
            position++;
        }
    }
    else {
        if(touch[position-1]==index)
        {
            QString str =ui->vuvod_ph->toPlainText();
            str.chop(1);
            ui->vuvod_ph->setText(str);
            ui->vuvod_ph->setAlignment(Qt::AlignCenter);
            static_cast<QPushButton*>(sender())->setStyleSheet("background-color: rgb(212, 233, 217); color: rgb(37, 89,
67); font: 16pt");
            position--;
        }
    }
}
QPushButton *HighLight4::setbutton_wp(QString text)
{
    QPushButton * button = new QPushButton;
    connect(button,&QPushButton::clicked,this,&HighLight4::signalbutton_wp);
    button->setText(text);
    QSizePolicy pls;
    pls.setVerticalPolicy(QSizePolicy::Expanding);
    pls.setHorizontalPolicy(QSizePolicy::Expanding);
    button->setSizePolicy(pls);
    button->setStyleSheet("background-color: rgb(212, 233, 217); color: rgb(37, 89, 67); font: 16pt");
    return button;
}
void HighLight4::on_go_out_clicked()
```

```
{
   int cor=0;
     QSqlQuery query, upd;
   QString word_id;
   word_id.setNum(id_word_g);
   query.exec("select id_w, eng, rus from word where id_w in (select id_word from stopka where
id_user="+QString::number(id_user)+") && id_w in (select id_word from having_c where
id_card="+QString::number(id_theme)+")");
   QFileInfo check_file("C:/Users/Meshtli/Desktop/hi/Kursach/word/"+word_id+".png");
   if(check_file.exists()){
   while(query.next() )
   {
      if(query.value(0).toInt()==id_word_g){
         QString a=query.value(1).toString(), b=ui->vuvod_ph->toPlainText();
         a=a.toLower();
         b=b.toLower();
         qDebug()<<a<<b;
         if(a==b){cor=1;

            QSqlQuery table;
            table.exec("select score from regime where id_r=4");
            table.next();
            upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where
id_word="+QString::number(id_word_g)+" && id_user="+QString::number(id_user));
            upd.clear();
            upd.exec("select count(*) from bal_regime where id_user="+QString::number(id_user));
            upd.next();
            if(upd.value(0).toInt()==0)
            {
              QSqlQuery create;
              create.exec("insert into bal_regime(id_reg,id_user,bal) value(1,"+QString::number(id_user)+",0)");
              create.exec("insert into bal_regime(id_reg,id_user,bal) value(2,"+QString::number(id_user)+",0)");
              create.exec("insert into bal_regime(id_reg,id_user,bal) value(3,"+QString::number(id_user)+",0)");
              create.exec("insert into bal_regime(id_reg,id_user,bal) value(4,"+QString::number(id_user)+",0)");
              create.exec("insert into bal_regime(id_reg,id_user,bal) value(5,"+QString::number(id_user)+",0)");
            }
            upd.clear();
            upd.exec("update bal_regime set bal=bal+"+table.value(0).toString()+" where id_r=4 &&
id_u"+QString::number(id_user));
=(select id_card from having_c where id_word="+QString::number(id_word_g)+")");
history(id_word,id_reg,date_walk) value("+QString::number(id_word_g)+",4,'"+dates+"')");
      }
   }
   }
   }
   else
   {
      while(query.next() )
      {
         if(query.value(0).toInt()==id_word_g){
            QString a=query.value(1).toString(), b=ui->vuvod_noph->toPlainText();
            a=a.toLower();
            b=b.toLower();
            if(a==b){cor=1;
                QSqlQuery table;
              table.exec("select score from regime where id_r=4");
              table.next();
              upd.exec("update stopka set bal=bal+"+table.value(0).toString()+" where
id_word="+QString::number(id_word_g)+" && id_user="+QString::number(id_user));
              upd.clear();
              upd.exec("select count(*) from bal_regime where id_user="+QString::number(id_user));
              upd.next();
```

```
                if(upd.value(0).toInt()==0)
                {
                  QSqlQuery create;
                  create.exec("insert into bal_regime(id_reg,id_user,bal) value(1,"+QString::number(id_user)+",0)");
                  create.exec("insert into bal_regime(id_reg,id_user,bal) value(2,"+QString::number(id_user)+",0)");
                  create.exec("insert into bal_regime(id_reg,id_user,bal) value(3,"+QString::number(id_user)+",0)");
                  create.exec("insert into bal_regime(id_reg,id_user,bal) value(4,"+QString::number(id_user)+",0)");
                  create.exec("insert into bal_regime(id_reg,id_user,bal) value(5,"+QString::number(id_user)+",0)");
                }
                upd.clear();
                upd.exec("update bal_regime set bal=bal+"+table.value(0).toString()+" where id_r=4 &&
id_u"+QString::number(id_user));
bal=bal+"+table.value(0).toString()+" where id_w="+rab);
history(id_word,id_reg,date_walk) value("+rab+",4,'"+dates+"')");
              }
           }
        }
     }
query.clear();
QPixmap pict;
   if(cor==1) pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/correct.png");
   else  pict.load("C:/Users/Meshtli/Desktop/hi/Kursach/other/nocorrect.png");
   ui->cor_photo->setPixmap(pict);
   ui->cor_photo->show();
 QTest::qWait(1500);
emit gochoose(ui->progressBar->value()+10);
}
void HighLight4::on_skip_clicked()
{
        emit gochoose(100);
}

#include "autorization.h"
#include <QApplication>
#include "mainwindow.h"
int main(int argc, char *argv[])
{
   QApplication a(argc, argv);
   Autorization w;
   w.show();
   return a.exec();
}

#include "ui_mainwindow.h"
#include "mainwindow.h"

QT_CHARTS_USE_NAMESPACE
MainWindow::MainWindow(int id_u, QWidget *parent) :
   QMainWindow(parent),
   ui(new Ui::MainWindow)
{ ui->setupUi(this);
setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
setWindowTitle("LerningEnglish");
ui->statistic->setStyleSheet("background-color: rgb(255, 252, 237); font: 11pt; color:rgb(62, 46, 31); border-radius:
10px;");
ui->history->setStyleSheet("background-color: rgb(255, 252, 237); font: 11pt; color:rgb(62, 46, 31); border-radius:
10px;");
ui->theme->setStyleSheet("background-color: rgb(76, 39, 19); border-radius: 10px; font: 11pt ; color: rgb(255, 232,
193);");
ui->goout->setStyleSheet("border-image: url(C:/Users/Meshtli/Desktop/hi/Kursach/other/exit.png) stretch; font: 1pt");
   learn = new StartLearning();
  connect(learn, &StartLearning::go_main,this, &MainWindow::getsignalstart);
```

```
connect(this, &MainWindow::setword_s,learn, &StartLearning::setword);
id_user=id_u;
ui->table_theme->setStyleSheet("background-color: rgb(130, 135, 144);");
   //Задаем начальное количество столбиков и строк
      ui->table_theme->setRowCount(1);
      ui->table_theme->setColumnCount(0);
       //Устанавливаем ширину
      ui->table_theme->setRowHeight(0, ui->table_theme->size().height()-20);//
      ui->table_theme->setColumnWidth(0, ui->table_theme->rowHeight(0));//
      QSqlQuery query, all, learn;
      query.exec("Select name_c, id_c from card");
      while(query.next())
      {    all.exec("select count(*) from having_c where id_card="+query.value(1).toString());
         all.next();
         //сделать нахождение
      newcolumn(query.value(0).toString(), 0, all.value(0).toInt(), query.value(1).toInt());
            }
       ui->table_theme->horizontalHeader()->hide();//
       ui->table_theme->verticalHeader()->hide();//
ui->history->hide();
ui->statistic->hide();
ui->addnew->hide();
ui->theme->hide();
}
MainWindow::~MainWindow()
{
   delete ui;
}
QWidget * MainWindow::AddNewButton(QString st, int v, int k, int id)
{
   //Создаем кнопку и сразу подключаем ее к слоту
   QPushButton * pb = new QPushButton();
   connect(pb, &QPushButton::clicked, this, &MainWindow::gotochoose);
   QString ids;
   ids.setNum(id);
pb->setText(ids);
QSqlQuery query;
int kol=0;
query.exec("select bal from word where id_w in (select id_word from having_c where id_card="+ids+")");
while(query.next())
{
int bal=query.value(0).toInt();
if(bal>=20) kol++;
}
query.clear();
QString kols;
kols.setNum(kol);
query.exec("update card set v_s="+kols+" where id_c ="+ids);
  // connect(pb, &QPushButton::clicked, choosemodeui, &ChooseMode::setword(st));
   //Делаем чтобы кнопка расстягивалась на всю площадь
   QSizePolicy pls;
   pls.setVerticalPolicy(QSizePolicy::Expanding);
   pls.setHorizontalPolicy(QSizePolicy::Expanding);
   pb->setSizePolicy(pls);
   pb->setStyleSheet("border-image: url(C:/Users/Meshtli/Desktop/hi/Kursach/card/"+ids+".png) stretch; font: 1pt");
QString v1, k1;
v1.setNum(v);
k1.setNum(k);
   QLabel * lbl = new QLabel(st+": "+kols+"/"+k1);
   //Размещаем кнопку и лейбл на слое
   QVBoxLayout * lay = new QVBoxLayout();
   lay->addWidget(pb, 1);
```

```
    lay->addWidget(lbl, 0);
    //Пихаем слой в виджет, потому что в таблицу можно вставлять только виджеты
    QWidget * wg = new QWidget();
    wg->setStyleSheet("background-color: rgb(255, 252, 237);");
    wg->setLayout(lay);
    return wg;
}
void MainWindow::getsignalstart()
{
    this->show();
    this->setGeometry(learn->pos().x()-7, learn->pos().y()+8, 900, 500);
   // on_history_clicked();
   // on_statistic_clicked();
    on_theme_clicked();
  }
void MainWindow::on_theme_clicked()
{if(ui->theme->styleSheet()!="background-color: rgb(76, 39, 19); border-radius: 10px; font: 11pt ; color: rgb(255, 232,
193);")
          {      ui->w_theme->show();
          ui->history->setStyleSheet("background-color: rgb(255, 252, 237); font: 11pt; color:rgb(62, 46, 31); border-
radius: 10px;");
          ui->statistic->setStyleSheet("background-color: rgb(255, 252, 237); font: 11pt; color:rgb(62, 46, 31); border-
radius: 10px;");
          ui->theme->setStyleSheet("background-color: rgb(76, 39, 19); border-radius: 10px; font: 11pt ; color: rgb(255,
232, 193);");
          //Задаем начальное количество столбиков и строк
            ui->table_theme->setRowCount(1);
            ui->table_theme->setColumnCount(0);
             //Устанавливаем ширину
            ui->table_theme->setRowHeight(0, ui->table_theme->size().height()-20);//
            ui->table_theme->setColumnWidth(0, ui->table_theme->rowHeight(0));//
            QSqlQuery query;
            query.exec("Select name_c, v_s, k_s, id_c from card");
            while(query.next())
            {QString name=query.value(0).toString();
               int v=query.value(1).toInt(), k=query.value(2).toInt();
            newcolumn(name, v, k, query.value(3).toInt());
                  }
             //Скрываем заголовки у таблицы
             ui->table_theme->horizontalHeader()->hide();//
             ui->table_theme->verticalHeader()->hide();//
    }
}
void MainWindow::newcolumn(QString st, int v, int k, int id)
{
    //Вставляем новый столбец
       ui->table_theme->setColumnCount(ui->table_theme->columnCount()+1);
       //Добавляем в новый столбик виджет
       ui->table_theme->setCellWidget(0, ui->table_theme->columnCount()-1, AddNewButton(st, v, k, id));
       //И подгоняем размер под первого
       ui->table_theme->setColumnWidth(ui->table_theme->columnCount()-1, ui->table_theme->rowHeight(0));
}
void MainWindow::gotochoose()
{QString st = static_cast<QPushButton*>(sender())->text();
    int st2=st.toInt();
    emit setword_s(st2, id_user);
    learn->setGeometry(this->pos().x()+1, this->pos().y()+38, 900, 500);
    learn->on_start_b_clicked();
    this->close();
void MainWindow::on_goout_clicked()
{
exit(EXIT_FAILURE);}
```

```
#include "refences.h"
#include "ui_refences.h"
void Refences::messange(QString a)
{
   QMessageBox mes;
   mes.setWindowTitle("Error");
   mes.resize(200,100);
   mes.setText(a);
   mes.exec();
}
Refences::Refences(QWidget *parent) :
   QWidget(parent),
   ui(new Ui::Refences)
{
   ui->setupUi(this);
   setWindowIcon(QIcon("C:/Users/Meshtli/Desktop/hi/Kursach/other/window.png"));
   setWindowTitle("LerningEnglish");
   ui->tabWidget->setStyleSheet("QTabBar::tab {height: 30px; width: 200px; font: 75 12pt \"Comic Sans MS\"; color:
rgb(255, 255, 255); background-color: rgb(74, 123, 101); border-color: rgb(159, 209, 186);}"
                                "QTabBar::tab:!selected {background-color: rgb(212, 233, 217); color: rgb(37, 89, 67);   border-
bottom-color: rgb(37, 89, 67); }");
   ui->tabWidget->setTabText(0,"Звук и темы");
   ui->tabWidget->setTabText(1,"Процесс обучения");
ui->delete_theme->hide();
   ui->speed->setRange(0,30);
   settings = new QSettings("C:/Users/Meshtli/Desktop/hi/Kursach/other/settings.conf", QSettings::IniFormat);
   ui->speed->setValue(settings->value("sound/speed").toInt());
   ui->volume->setValue(settings->value("sound/volume").toInt());
   QSqlQuery query;
   query.exec("select name_user from users");
   while(query.next()){
      ui->student->addItem(query.value(0).toString());
   }
query.clear();
ui->delete_student->hide();
query.exec("select name_c from card");
int i=0;
while(query.next()){
   ui->themes->addItem(query.value(0).toString());
   ui->stat->setColumnCount(ui->stat->columnCount()+1);
   ui->stat->setHorizontalHeaderItem(i, new QTableWidgetItem(query.value(0).toString()));
   i++;
}
query.clear();
query.exec("select name_user from users");
i=0;
while(query.next()){
   ui->choose_school->addItem(query.value(0).toString());
   ui->stat->setRowCount(ui->stat->rowCount()+1);
   ui->stat->setVerticalHeaderItem(i, new QTableWidgetItem(query.value(0).toString()));
   i++;
}
for(int i=0;i<ui->stat->columnCount();i++)
   for(int j=0;j<ui->stat->rowCount();j++)
   {   QString card, user, count_word;
      int i2=0, j2=0;
      query.clear();
      query.exec("select id_c from card");
      while(query.next() && i2!=i) i2++;
      card=query.value(0).toString();
      query.clear();
      query.exec("select id_u from users");
```

```
        while(query.next() && j2!=j) j2++;
        user=query.value(0).toString();
        query.clear();
        query.exec("select count(*) from having_c where id_card="+card);
        query.next();
        count_word=query.value(0).toString();
        query.clear();
        query.exec("select count(*) from having_u where id_word in (select id_word from having_c where
id_card="+card+") && id_user="+user);
        query.next();
        QTableWidgetItem * item =new QTableWidgetItem(query.value(0).toString()+"/"+count_word);

        item->setTextAlignment(Qt::AlignCenter);
        ui->stat->setItem(j,i,item);
    }
 ui->stat->setStyleSheet("QHeaderView::section {color: rgb(255, 255, 255); background-color: rgb(74, 123, 101);}; ");
}
Refences::~Refences()
{    delete ui;
}
void Refences::on_save_clicked()
{    settings->setValue("sound/speed", ui->speed->value());
    settings->setValue("sound/volume", ui->volume->value());
    settings->sync();
}
void Refences::on_add_clicked()
{
    addtheme=new AddTheme;
    connect(addtheme, &AddTheme::goout,this, &Refences::getsignaladdtheme);
    addtheme->show();
    addtheme->setGeometry(this->pos().x()+8, this->pos().y()+30, 900, 500);
     this->close();
}
void Refences::getsignaladdtheme()
{
    this->setGeometry(this->pos().x()+8, this->pos().y()+30, 900, 500);
    this->show();
    addtheme->close();
}
void Refences::on_themes_currentRowChanged(int currentRow)
{
    ui->delete_theme->show();
}

void Refences::on_delete_theme_clicked()
{
    int n = QMessageBox::warning(0,
                                 "Внимание!",
                                 "Тема будет удалена без возможности восстановления."
                                 "\n Вы хотите продолжить?",
                                 "Да",
                                 "Нет",
                                 QString(),
                                 0,
                                 1
                                 );
    if(!n) {
        QString id;
        QSqlQuery query;
        query.exec("select id_c from card where name_c="+ui->themes->item(ui->themes->currentRow())->text());
        query.next();
        id=query.value(0).toString();
```

```
        query.clear();
        query.exec("delete from card where name_c="+ui->themes->item(ui->themes->currentRow())->text());
        query.next();
        query.clear();
        query.exec("delete from having_c where id_card="+id);
        query.next();
    }
    ui->student->takeItem(ui->student->currentRow());
}
void Refences::on_delete_student_clicked()
{
    int n = QMessageBox::warning(0,
                                 "Внимание!",
                                 "Ученик будет удален без возможности восстановления."
                                 "\n Вы хотите продолжить?",
                                 "Да",
                                 "Нет",
                                 QString(),
                                 0,
                                 1
                                 );
    if(!n) {
        QString id;
        QSqlQuery query;
        query.exec("select id_u from users where name_user='"+ui->student->item(ui->student->currentRow())-
>text()+"'");
        query.next();
        id=query.value(0).toString();
        query.clear();
        query.exec("delete from having_u where id_user="+id);
        query.next();
        query.clear();
        query.exec("delete from users where name_user='"+ui->student->item(ui->student->currentRow())->text()+"'");
        query.next();
    }
ui->student->takeItem(ui->student->currentRow());
}
void Refences::on_add_s_clicked()
{bool flag=true;
    QSqlQuery query;
    query.exec("select name_user from users");
    while(query.next())
    {
        if(query.value(0).toString()==ui->input_student->toPlainText()) flag=false;
    }
    if(flag==true)
    {
 query.clear();
 query.exec("insert into users(name_user) value('"+ui->input_student->toPlainText()+"');");
 query.next();
 ui->student->addItem(ui->input_student->toPlainText());
    }
    else messange("Такой ученик уже есть!");
}
void Refences::on_student_currentRowChanged(int currentRow)
{
    ui->delete_student->show();
}
void Refences::on_tabWidget_currentChanged(int index)
{
    if(index==2)
    {
```

```
                QBarSet *set0 = new QBarSet("Сбор слова");
                QBarSet *set1 = new QBarSet("Выбор перевода");
                QBarSet *set2 = new QBarSet("Выбор перевода среди пар");
                QBarSet *set3 = new QBarSet("Выделение слова");
                QBarSet *set4 = new QBarSet("Правильный или неправильный перевод");
                QSqlQuery query, user;
                user.exec("select id_u from users");
                int i=0;
                QString id_user;
                while(user.next())
                {
                   if(ui->choose_school->currentIndex()==i) {id_user=user.value(0).toInt(); break;}
                   i++;
                }
qDebug()<<id_user;
                int mas[5], range=100;
                query.exec("select bal from bal_regime where id_reg=1 && id_user="+id_user);
                query.next();
                query.clear();
                mas[0]=query.value(0).toInt();
                *set0 << mas[0] ;
                *set1 << mas[1] ;
                *set2 << mas[2] ;
                *set3 << mas[3] ;
                *set4 << mas[4] ;
                QBarSeries *series = new QBarSeries();
                series->append(set0);
                series->append(set1);
               series->append(set2);
                series->append(set3);
                series->append(set4);
                QChart *chart = new QChart();
                chart->addSeries(series);
                chart->setAnimationOptions(QChart::SeriesAnimations);
                QValueAxis *axisY = new QValueAxis();
                axisY->setRange(0,range);
                chart->addAxis(axisY, Qt::AlignLeft);
                series->attachAxis(axisY);
                chart->legend()->setVisible(true);
                chart->legend()->setAlignment(Qt::AlignRight);
                QChartView *chartView = new QChartView(chart);
                chartView->setRenderHint(QPainter::Antialiasing);
      chartView->setStyleSheet("background-color: rgb(255, 252, 237); font: 11pt; color:rgb(62, 46, 31);");
      ch = new QHBoxLayout;
      delete ch;
         ch = new QHBoxLayout;
         ch->addWidget(chartView);
         ui->grafic->setLayout(ch) ;
   }
}
void Refences::on_choose_school_currentIndexChanged(int index)
{
   on_tabWidget_currentChanged(2);
}

#include "startlearning.h"
#include "ui_startlearning.h"
StartLearning::StartLearning( QWidget *parent) :
   QWidget(parent),
   ui(new Ui::StartLearning)
{    ui->setupUi(this);
}
```

```
StartLearning::~StartLearning()
{
   delete ui;
}
void StartLearning::on_start_b_clicked()
{QString id_u;
   id_u.setNum(id_user_sent);
   QSqlQuery new_s;
   new_s.exec("select count(*) from stopka where id_user="+id_u+" && id_word in (select id_word from having_c
where id_card="+QString::number(id_theme_text)+")");

   new_s.next();
   if(new_s.value(0).toInt()==0)
   {int count=0;
    int words[10];
      do{
      QSqlQuery word;
      word.exec("select id_w from word where id_w in (select id_word from having_c where
id_card="+QString::number(id_theme_text)+")");

       while(word.next())
       {
         int random=rand()%3, rab=0;
         for(int i=0;i<count;i++)
            if(word.value(0).toInt()==words[i]) rab++;
         if(random==0 && rab==0)
         {
            words[count]=word.value(0).toInt();
            count++;
         }
       }
       word.clear();
      }while(count<10);
    for(int i=0; i<10; i++)
    {QString str;
       str.setNum(words[i]);
       QSqlQuery ins;
       ins.exec("insert into stopka(bal, id_word, id_user) value(0, "+str+", "+id_u+")");
       ins.clear();
    }
   }
   learn = new Card(id_user_sent, id_theme_text);
    connect(learn, &Card::go_back,this, &StartLearning::getsignalback);
   learn->setGeometry(this->pos().x()-7, this->pos().y()+8, 900, 500);
   learn->show();
   this->close();
}
void StartLearning::getsignalback()
{    this->setGeometry(this->pos().x(), this->pos().y(), 900, 500);
    learn->close();
   emit go_main();
}
void StartLearning::setword(int id_theme, int id_user)
{   id_theme_text=id_theme;
   id_user_sent=id_user;
   QString s;
   s.setNum(id_theme);
   QSqlQuery query;
   query.exec("select name_c from card where id_c="+s);
   query.next();
   query.clear();
}
```

ДОДАТОК Б

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК В

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
Диплом_ doc	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Диплом_.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація_.ppt	Презентація кваліфікаційної роботи