**федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ» (КНИТУ-КАИ)**

**ОТДЕЛЕНИЕ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**ДИПЛОМНЫЙ ПРОЕКТ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Тема: Разработка приложения для изучения русского языка, как иностранного, для испаноговорящих .** | | | | |
| **Руководитель**  Преподаватель |  |  |  | И.Ф. Гимадеев |
| *(должность)* |  | *(подпись)* |  | *(И.О. Фамилия)* |
| **Обучающийся**  4432 |  |  |  | П.А. Валова |
| *(группа)* |  | *(подпись)* |  | *(И.О. Фамилия)* |
| **Специальность** 09.02.07 Информационные системы и программирование . | | | | |
| (шифр и наименование специальности) | | | | |

Казань 2022 г.

**АННОТАЦИЯ**

Данный документ представляет собой пояснительную записку к выпускной квалификационной работе по направлению обучения 09.02.07 «Информационные системы и программирование» на тему «Разработка приложения для изучения русского языка, как иностранного, для испаноговорящих».

Пояснительная записка состоит из: количество страниц – 60; количество приложений – 5; количество иллюстраций – 59; количество таблиц – 9; количество использованных источников – 20.

Целью дипломного проекта является обучение испаноговорящих русскому языку, как иностранному, с помощью теоретических и практических занятий.

Данная информационная система предусмотрена для образовательных учреждений, в которых осуществляется подготовка иностранцев к поступлению в университет.

Результат дипломного проекта – информационная система “Calanthe” для изучения русского языка, как иностранного, для испаноговорящих.

Ключевые слова: desktop приложение, обучение, Windows Presentation Foundation, Microsoft SQL Server, Visual Studio.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 6](#_Toc105456040)

[ГЛАВА 1 ПОСТАНОВКА ЗАДАЧИ 8](#_Toc105456041)

[1.1 Цели и задачи информационной системы 8](#_Toc105456042)

[1.2 Описание предметной области 9](#_Toc105456043)

[1.2.1 Регистрация/авторизация обучающегося 9](#_Toc105456044)

[1.2.2 Профиль обучающегося 10](#_Toc105456045)

[1.2.3 Занятия 10](#_Toc105456046)

[1.2.4 Словарь 11](#_Toc105456047)

[ГЛАВА 2 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 12](#_Toc105456048)

[2.1 Выбор средств реализации 12](#_Toc105456049)

[2.1.1 Язык программирования 12](#_Toc105456050)

[2.1.2 Среда разработки 13](#_Toc105456051)

[2.1.3 Платформа разработки 14](#_Toc105456052)

[2.1.4 Система управления базами данных 15](#_Toc105456053)

[2.2 Выявление требований к программному обеспечению 16](#_Toc105456054)

[2.3 Проектирование диаграммы состояний 20](#_Toc105456055)

[2.4 Проектирование диаграммы деятельности 21](#_Toc105456056)

[2.5 Проектирование диаграммы классов 22](#_Toc105456057)

[2.6 Проектирование диаграммы последовательностей 23](#_Toc105456058)

[2.7 Проектирование диаграммы сущность-связь 26](#_Toc105456059)

[2.8 Разработка пользовательского интерфейса 27](#_Toc105456060)

[ГЛАВА 3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 37](#_Toc105456061)

[ГЛАВА 4 ПОЛЬЗОВАТЕЛЬСКАЯ ДОКУМЕНТАЦИЯ 44](#_Toc105456062)

[4.1 Руководство по установке программного обеспечения 44](#_Toc105456063)

[4.2 Руководство пользователя 47](#_Toc105456064)

[ЗАКЛЮЧЕНИЕ 57](#_Toc105456065)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 58](#_Toc105456066)

[ПРИЛОЖЕНИЕ А 60](#_Toc105456067)

[ПРИЛОЖЕНИЕ Б 61](#_Toc105456068)

[ПРИЛОЖЕНИЕ В 62](#_Toc105456069)

[ПРИЛОЖЕНИЕ Г 67](#_Toc105456070)

[ПРИЛОЖЕНИЕ Д 103](#_Toc105456071)

# ВВЕДЕНИЕ

Число иностранных студентов увеличивается с каждым годом, но, к сожалению, обучения на подготовительных курсах, которые предоставляют образовательные учреждения, недостаточно.

Во время обучения русскому языку иностранцы встречаются с такими трудностями, как: старые обучающие материалы, не имеющие визуальных решений; плохая коммуникация с преподавателями русского языка из-за лингвистического барьера. Как следствие, иностранные студенты поступают в образовательные учреждения, но не получают информацию в полном объёме, из-за чего, чаще всего, у них появляется множество академических задолженностей или они понимают, что не справляются с данным объемом учебного материала, и покидают страну.

Именно для решения проблем, описанных выше, было принято решение создать информационную систему “Calanthe”, позволяющую иностранцам, говорящих на испанском языке, ознакомиться с базой русского языка и научиться активно применять полученные знания в жизни.

Для достижения поставленной цели было решено реализовать следующие модули:

1. регистрации;
2. авторизации;
3. просмотра теоретических занятий;
4. просмотра практических занятий;
5. редактирования профиля;
6. просмотра статистики;
7. ведения словаря.

Разделы дипломного проекта, их название и описание:

1. Глава 1 “Постановка задачи”: эта глава содержит в себе формулировку решаемой задачи;
2. Глава 2 “Проектирование информационной системы”: в этой главе определяется обобщенная архитектура программного обеспечения (пакета программ, системы или модуля), описывается назначение выделенных блоков и связи между ними;
3. Глава 3 “Проектирование базы данных”: эта глава описывает процесс создания схемы базы данных и определение ограничений целостности.
4. Глава 4 “Пользовательская документация”: в данную главу входят следующие руководства: руководство по установке программного обеспечения, руководство пользователя.

# ГЛАВА 1 ПОСТАНОВКА ЗАДАЧИ

Эта глава содержит в себе формулировку решаемой задачи и описание всех требований к системе.

## **1.1 Цели и задачи информационной системы**

Для того, чтобы выявить цели и задачи данного проекта, был проведён социальный опрос среди иностранных обучающихся. Мы спросили у них, с какими проблемами они сталкивались, учась на подготовительном курсе, которые предоставляют образовательные учреждения.

Далее рассмотрим основные проблемы, и каким способом их будет решать данная ИС. Они продемонстрированы в таблицах 1.1 – 1.4.

Таблица 1.1 – Проблема №1

|  |  |
| --- | --- |
| Проблема | Отсутствие визуальных решений и старый материал |
| Решение | Занятия, загруженные в программу, имеют иллюстрации, красочные схемы для облегченного усвоения информации. А также каждое теоретическое занятие прошло несколько этапов проверки и тестирование на реальных студентах |

Таблица 1.2 – Проблема №2

|  |  |
| --- | --- |
| Проблема | Преподаватели дают одинаковые задания для иностранных обучающихся в университете и обучающихся на подготовительном курсе, поэтому студенты, только приехавшие в Россию, начинают отставать от программы |
| Решение | Данная ИС предназначена только для студентов, обучающихся на подготовительном курсе, поэтому преподавателю не придётся брать задания, предназначенные для тех, кто уже учится в университете |

Таблица 1.3 – Проблема №3

|  |  |
| --- | --- |
| Проблема | Лингвистический барьер |
| Решение | Программа, а также теоретические и практические занятия, полностью переведены на испанский язык |

Таблица 1.4 – Проблема №4

|  |  |
| --- | --- |
| Проблема | Частая смена преподавателей, что влияет на образовательный процесс обучающихся, так как у каждого преподавателя своя форма обучения |
| Решение | Обучаясь по данной программе, студентам не придётся беспокоиться о том, что придёт новый преподаватель с абсолютной другой формой обучения |

А также есть ещё одно преимущество в данной ИС: в этой программе есть всё необходимое для комфортной и интенсивной учёбы: возможность ведения словаря, отслеживание статистики, просмотра теоретических занятий, выполнение практических занятий.

Самая главная цель этого проекта – дать иностранным студентам возможность получить качественное образование и коммуницировать с людьми, живущими в России.

## **1.2 Описание предметной области**

«Calanthe» – это приложение для изучения русского языка, как иностранного, для испаноговорящих. Ранее образовательная деятельность осуществлялась без сторонних приложений и для удобства было решено написать систему, которая бы облегчала работу преподавателя и студентов.

* + 1. **Регистрация/авторизация обучающегося**

Весь материал доступен обучающемуся только после регистрации или авторизации.

Для регистрации пользователь должен ввести свою действующую электронную почту, логин и пароль. Электронная почта необходима для того, чтобы обучающийся, потеряв пароль, мог восстановить его. Восстановление будет происходить следующим образом: после нажатия на кнопку, пользователю придёт сгенерированный пароль на его электронную почту, после чего необходимо повторно авторизоваться.

При авторизации пользователь должен ввести электронную почту и пароль.

Входные данные для регистрации и авторизации должны иметь следующий вид:

1. электронная почта: \_\_\_\_\_ @ \_\_ . \_\_ ;
2. логин: любые символы;
3. пароль: более 8 символов.
   * 1. **Профиль обучающегося**

Зайдя в профиль, обучающийся видит свою статистику и фото. После прохождения каждого занятия модуль в статистике будет загораться ярким цветом, чтобы пользователь мог отслеживать, какое количество занятий он прошёл.

Перейдя в окно редактирования профиля, он может увидеть свои данные, которые он вводил при регистрации, и в случае необходимости, изменить их.

### **1.2.3 Занятия**

Курс состоит из 16 теоретических и практических занятий:

1. алфавит;
2. существительные;
3. прилагательные;
4. местоимения;
5. глаголы 1;
6. глаголы 2;
7. винительный падеж;
8. дательный падеж;
9. родительный падеж;
10. творительный падеж;
11. предложный падеж;
12. прошедшее время;
13. будущее время.
14. частицы;
15. вопросы;
16. предлоги.

Каждому теоретическому занятию соответствует одно практическое, которое представлено в виде теста.

### **1.2.4 Словарь**

Все слова в словаре хранятся в базе данных. В случае необходимости обучающийся может добавить свои слова, а также удалить их или изменить.

Для каждого пользователя открываются только те слова, которые добавил данный пользователь.

# ГЛАВА 2 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

В этом разделе описывается выбор средств реализации, определены требования к ПО, спроектированы UML – диаграммы, база данных и пользовательский интерфейс.

## **2.1 Выбор средств реализации**

В данной части даётся обзор языка программирования, среды разработки, платформы, системы управления базами данных

### **2.1.1 Язык программирования**

C# - это язык программирования, который Microsoft сделал, взяв лучшее из языков C и C++, и продолжил добавлять к нему функциональные возможности, заимствуя из других языков, таких как java. Он ориентировал его для всей своей платформы .NET (как Framework, так и Core), и со временем адаптировал средства создания кода, что сделало его таким же универсальным и простым в освоении.[1].

Как мы уже говорили ранее, C# - это язык, предназначенный для использования на .NET, целью которого является создание приложений простым способом. Таким образом, этот язык используется для разработки приложений на этой платформе.[18].

Рассмотрим особенности этого ЯП:

1. простота: C# не содержит элементы, которые не нужны в .NET;
2. современность: этот ЯП автоматически и интуитивно включает в себя элементы, которые необходимы для разработки приложения;
3. безопасность: встроенный механизм для обеспечения правильного доступа к типам данных, что позволяет избежать ошибок, которые трудно обнаружить;
4. расширяемость: возможность добавлять основные типы данных, операторы и модификаторы;
5. совместимость: C# поддерживает синтаксис, очень похожий на C, C++, Java и многие другие языки программирования, чтобы облегчить работу программиста;
6. эффективность: несмотря на ограничения, которые имеет C#, вы можете пропустить эти ограничения, используя объекты с помощью указателей.

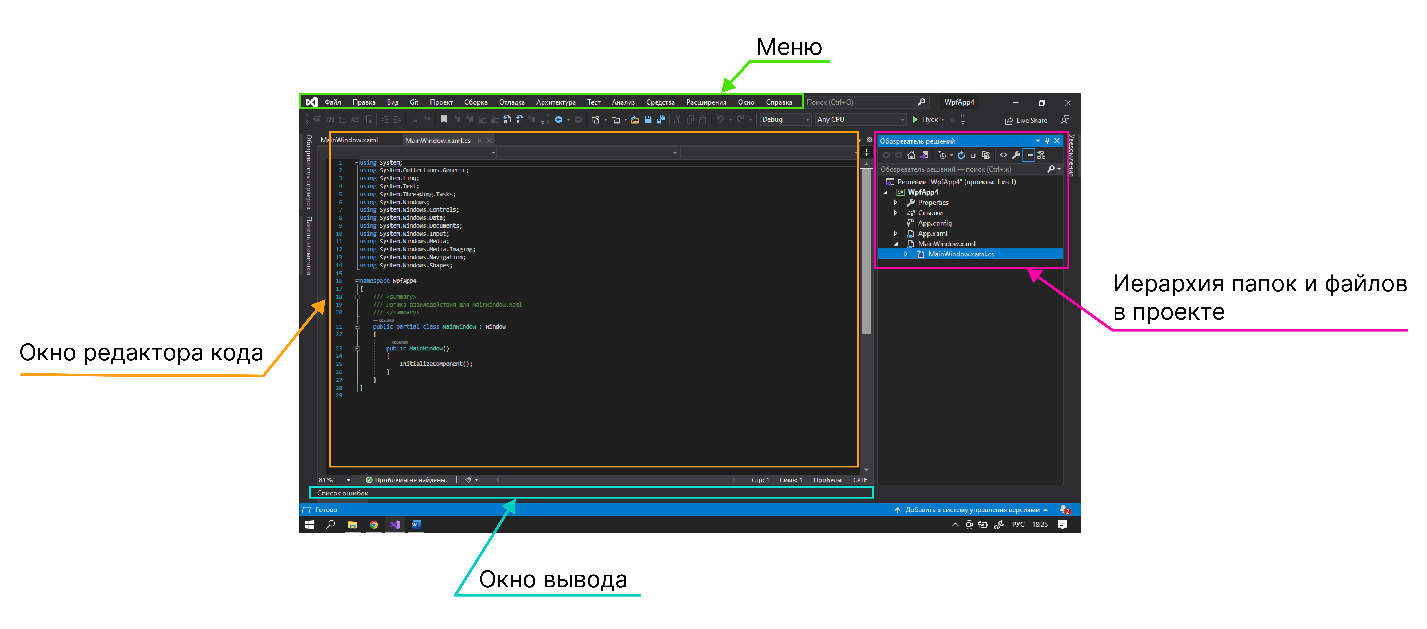
### **2.1.2 Среда разработки**

Visual Studio – это интегрированная среда разработки для просмотра и редактирования любого типа кода после отладки, сборки и публикации Android, iOS, Windows, веб-приложений и облачных приложений. VS создана компанией Microsoft и доступна для операционных систем Windows, Linux и macOS, и в то же время он совместим с несколькими языками программирования, такими как C++, C#, Visual Basic .NET, F#, Java, Python, Ruby и PHP, а также что среды веб-разработки, такие, как ASP.NET.[6].

VS – Это многофункциональная программа, которую можно использовать для многих аспектов разработки программного обеспечения. Помимо стандартного редактора и отладчика, предоставляемых большинством IDE, Visual Studio включает в себя компиляторы, инструменты завершения кода, графические дизайнеры и многие другие функции, упрощающие процесс разработки программного обеспечения.[10].

Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).[11].

Краткое описание пользовательского интерфейса Visual Studio представлено на рисунке 2.1.

****Рисунок 2.1 – Пользовательский интерфейс Visual Studio

### **2.1.3 Платформа разработки**

WPF - это сокращение от Windows Presetation Foundation. Это часть платформы разработки .NET. Но не просто часть, она самая большая из всех. Она охватывает многие концепции от пользовательских интерфейсов с XAML до шаблонов проектирования, таких как MVVM.[2].

С точки зрения программирования, это серия сборок и инструментов .NET Framework.[3]. Он предназначен для предоставления API (Интерфейс прикладного программирования) для создания богатых и сложных пользовательских интерфейсов для Windows. Он поддерживается от Windows XP до последней версии Windows.

А также WPF включает в себя новые функции, которые делают его предшественника, Windows Forms, устаревшим. Например, поддержка 3D, расширенная типография и PDF-подобные документы и многое другое.

Рассмотрим наиболее важные функции WPF:

1. декларативный графический интерфейс: WPF позволяет создавать пользовательские интерфейсы с использованием языка разметки под названием XAML;
2. динамический дизайн;
3. векторная графика: графики в WPF основаны на векторах. В отличие от графики на основе растровых изображений, векторную графику можно масштабировать без деформаций;
4. шаблоны: мы можем создавать повторно используемые элементы для графического интерфейса;
5. стили: стили в WPF очень похожи на каскадные таблицы стилей, CSS, для HTML. В этом случае стили более мощные, чем CSS. Мы можем обрабатывать любые графические функции, которые вы можете себе представить, поля, пространства, цвета, размеры и многое другое.[5].

### **2.1.4 Система управления базами данных**

Microsoft SQL Server - это система управления реляционными базами данных (СУБД), которая поддерживает широкий спектр приложений для обработки транзакций, бизнес-аналитики и аналитики в корпоративных вычислительных средах. Microsoft SQL Server является одной из трех ведущих технологий баз данных на рынке.[4].

Как и другие технологии СУБД, SQL Server в основном построен вокруг структуры таблиц на основе строк, которая соединяет связанные элементы данных в разных таблицах друг с другом, избегая необходимости избыточного хранения данных в нескольких местах в базе данных.[17]. Реляционная модель также обеспечивает ссылочную целостность и другие ограничения целостности для поддержания точности данных.

Основным компонентом Microsoft SQL Server является механизм базы данных SQL Server, который контролирует хранение, обработку и безопасность данных. Он включает в себя реляционный механизм, обрабатывающий все команды и запросы, и механизм хранения, управляющий файлами базы данных, таблицы, страницы, индексы, буферов данных и транзакций. Механизм базы данных также создает и выполняет хранимые процедуры, триггеры, представления и другие объекты базы данных.[13].

## **2.2 Выявление требований к программному обеспечению**

Для начала выявим все пользовательские требования для данного программного продукта, они представлены в таблице 2.1.

Таблица 2.1 – Пользовательские требования

|  |  |
| --- | --- |
| **№** | **Пользовательское требование** |
| 1 | Возможность регистрации пользователя |
| 2 | Возможность авторизации пользователя |
| 3 | Возможность восстановления пароля |
| 4 | Возможность просмотра личных данных |
| 5 | Возможность просмотра статистики |
| 6 | Возможность редактирования личных данных |
| 7 | Возможность просмотра теоретических занятий |
| 8 | Возможность прохождения практических занятий |
| 9 | Возможность просмотра словаря |
| 10 | Возможность редактирования словаря |

Далее необходимо выполнить сокращение этих требований:

Шаг.1:..Все требования, представленные выше, относятся к первоначальной цели разработки проекта, поэтому сокращение происходить не должно.

Шаг 2: Теперь следует распределить их по категориям, как показано в таблице 2.2.

Таблица 2.2 – Распределение требований по категориям

|  |  |
| --- | --- |
| **Категория** | **Список требований** |
| «должно быть включено» – если эти требования не будут включены, не будет достигнута цель | Возможность регистрации пользователя |
| Возможность авторизации пользователя |
| Возможность просмотра теоретических занятий |

Продолжение таблицы 2.2

|  |  |
| --- | --- |
| **Категория** | **Список требований** |
|  | Возможность прохождения практических занятий |
| Возможность просмотра словаря |
| Возможность редактирования словаря |
| «желательно включить» – если эти требования не будут включены, цели не будут реализованы в полном объеме | Возможность просмотра личных данных |
| Возможность редактирования личных данных |
| «хорошо бы включить» – это полезные требования, но они не способствуют достижению первоначальных целей | Возможность восстановления пароля |
| Возможность просмотра статистики |
| «нужно отклонить» – это требования, которые не соответствуют первоначальным целям | - |

Шаг 3: Приведем список требований в соответствие с возможностями: никакие из требований не нуждаются в удалении;

Шаг 4: Требования категории «должно быть включено» исключаться не будут, так как план проекта остался неизменным;

Шаг.5:.В таблице 2.3 представлен перечень оставшихся пользовательских требований, которая будет служить основой для контроля изменений. Данный перечень войдет в спецификацию требований к разрабатываемому ПО.

Таблица 2.3 – Пользовательские требования

|  |  |
| --- | --- |
| **№** | **Пользовательское требование** |
| 1 | Возможность регистрации пользователя |

Продолжение таблицы 2.3

|  |  |
| --- | --- |
| **№** | **Пользовательское требование** |
| 2 | Возможность авторизации пользователя |
| 3 | Возможность восстановления пароля |
| 4 | Возможность просмотра личных данных |
| 5 | Возможность просматривать статистику |
| 6 | Возможность редактирования личных данных |
| 7 | Возможность просмотра теоретических занятий |
| 8 | Возможность прохождения практических занятий |
| 9 | Возможность просмотра словаря |
| 10 | Возможность редактирования словаря |

Шаг 6: Отклоненные требования необходимо сохранить для будущих обзоров как нереализованные потребности.

Теперь мы должны выделить акторов, взаимодействующих с системой. В данной системе будет только два актора – гость и обучающийся.

Описание акторов представлено в таблице 2.4.

Таблица 2.4 – Описание акторов

|  |  |
| --- | --- |
| **Акторы** | **Краткое описание** |
| Гость | Неопределенный пользователь системы, желающий воспользоваться её возможностями |
| Обучающийся | Физическое лицо, осваивающее образовательную программу |

Далее необходимо описать варианты использования, выявленные выше. Они указаны в Таблице 2.5.

Таблица 2.5 – Выявленные варианты использования

|  |  |  |
| --- | --- | --- |
| **Основной**  **актор** | **Наименование** | **Формулировка** |
| Гость | Авторизация | Пользователь с помощью почты и пароля может войти в систему |
| Гость | Регистрация | Пользователь с помощью логина, почты и пароля может зарегистрироваться в системе |
| Обучающийся | Восстановления пароля | Пользователь может восстановить пароль, введя электронную почту, которую он указывал при регистрации |
| Обучающийся | Просмотр личных данных | Пользователь может просматривать свои личные данные в личном кабинете |
| Обучающийся | Редактирование личных данных | Пользователь может просматривать свои личные данные в личном кабинете |
| Обучающийся | Просмотр статистики | Пользователь может просматривать свою статистику |
| Обучающийся | Возможность просмотра теоретических занятий | Пользователь может просматривать теоретические занятия |
| Обучающийся | Возможность прохождения практических занятий | Пользователь имеет возможность проходить практические занятия |

Продолжение таблицы 2.5

|  |  |  |
| --- | --- | --- |
| **Основной**  **актор** | **Наименование** | **Формулировка** |
| Обучающийся | Возможность просмотра словаря | Пользователь просматривать свой словарь |
| Обучающийся | Возможность редактирования словаря | Пользователь может осуществлять ведение словаря |

Исходя из выявленных вариантов использования, была составлена диаграмма вариантов использования, изображённая на рисунке 2.2.

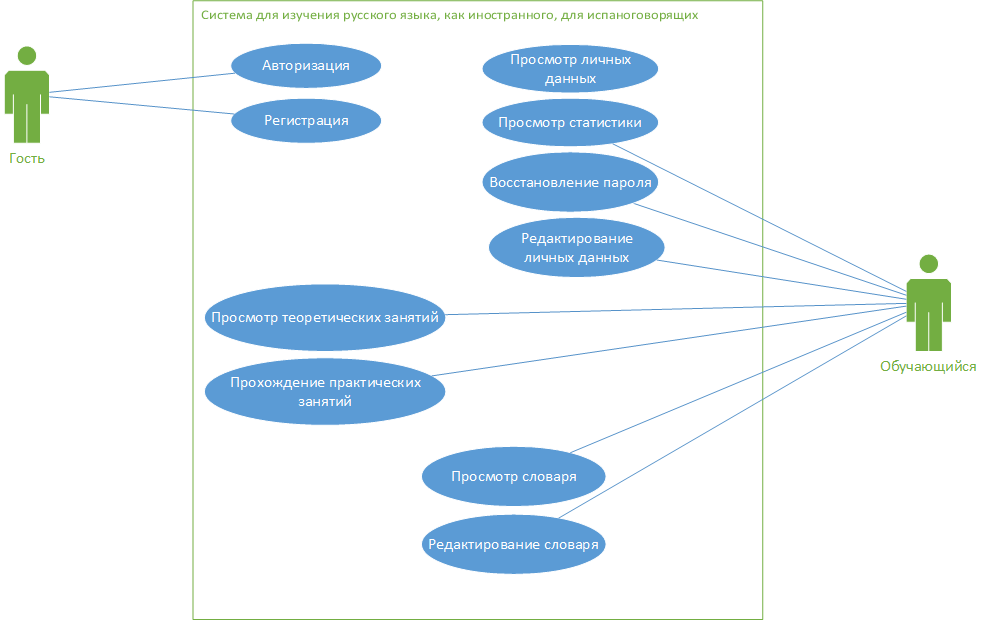


Рисунок 2.2 – Диаграмма вариантов использования

## **2.3 Проектирование диаграммы состояний**

Данная диаграмма необходима для того, чтобы описать, каким образом работают объекты, а также она показывает, как объект переходит из одного состояния в другое.

Диаграмма описывает все возможные состояния, в которые может попасть конкретный объект, и то, как состояние объекта изменяется в результате событий, наступающие для объекта.

Основными элементами являются:

1. состояние;
2. переход.

Диаграмма состояния для актора «Обучающийся» изображена на рисунке 2.3

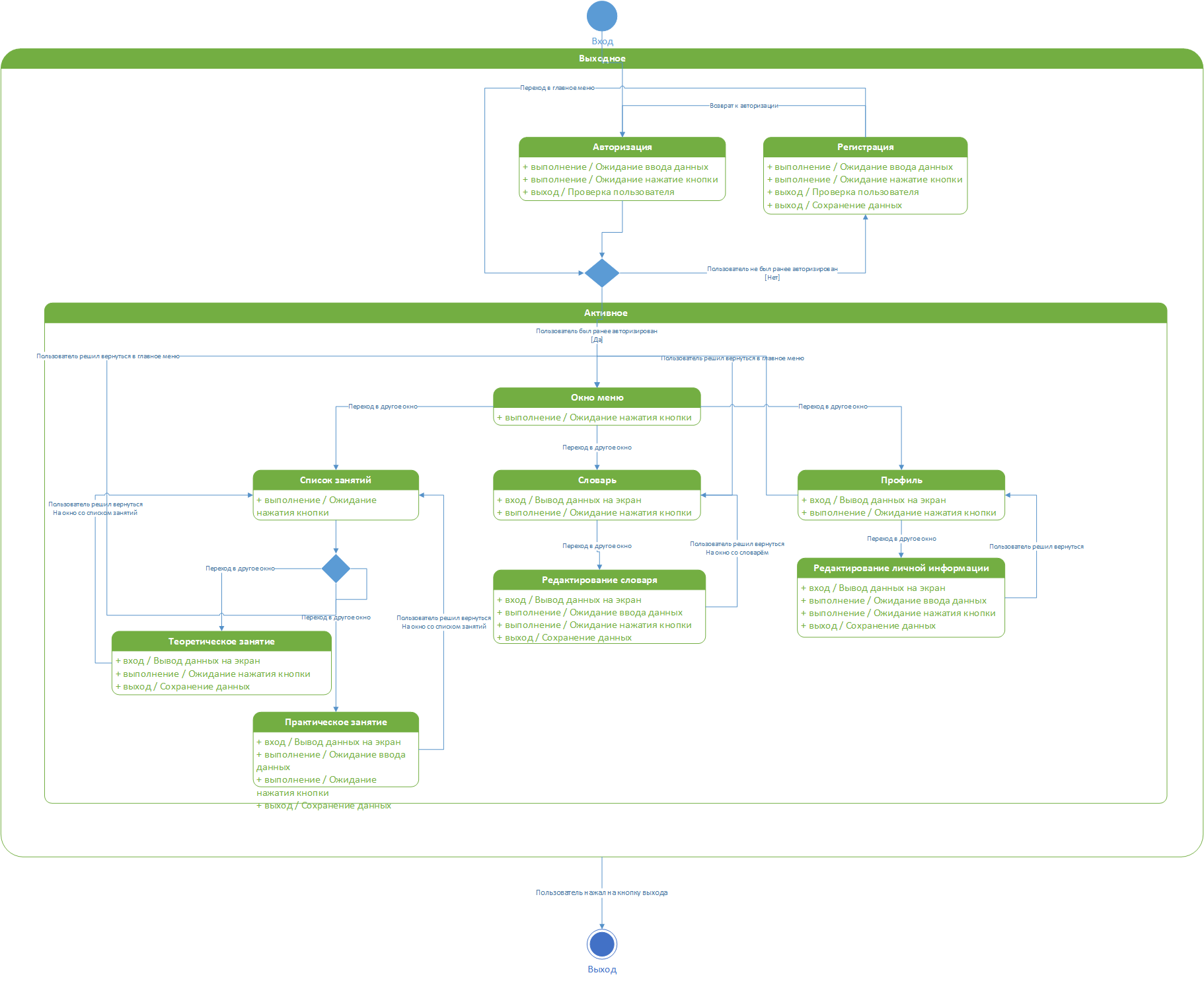


Рисунок 2.3 – Диаграмма состояний для актора «Обучающийся»

## **2.4 Проектирование диаграммы деятельности**

Диаграмма деятельности содержит в себе представление рабочих процессов поэтапных действий и действий с поддержкой выбора, итерации и параллелизма.

При построении этой диаграммы необходимо выполнить следующие шаги:

1. добавить дорожки для каждого субъекта;
2. добавить начальный узел;
3. расписать операции;
4. добавить конечный узел.

Данная диаграмма изображена на рисунке 2.4.

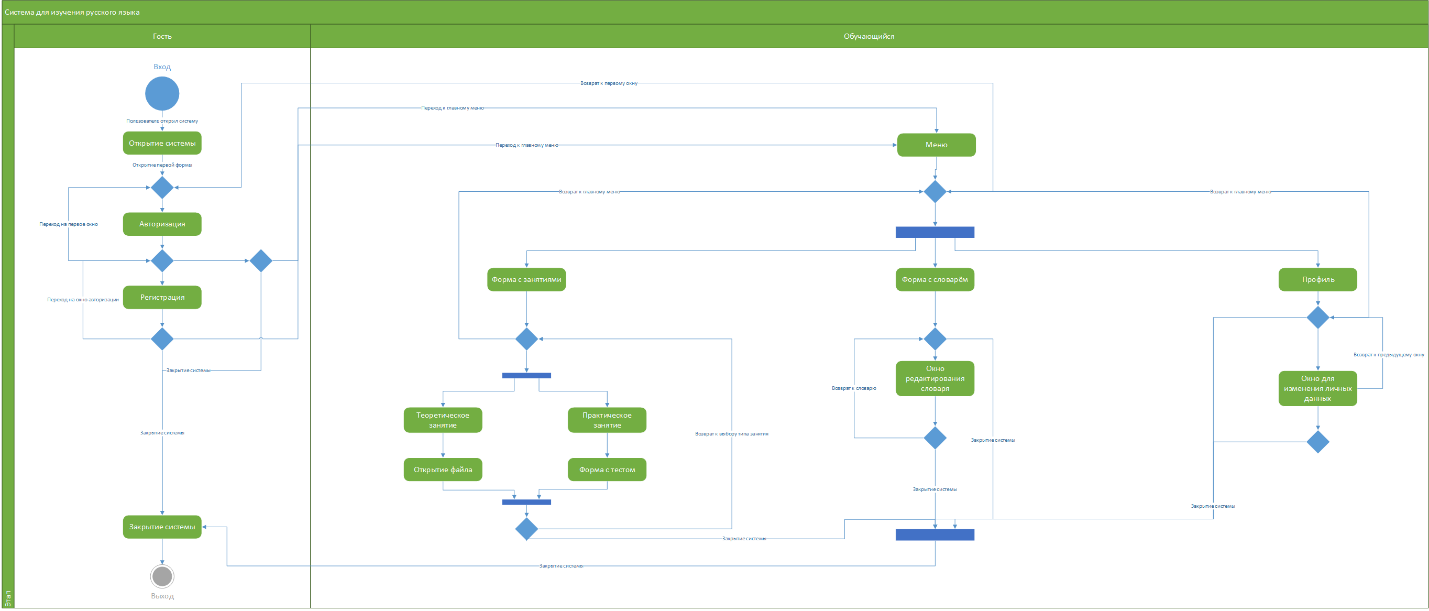


Рисунок 2.4 – Диаграмма деятельности

## **2.5 Проектирование диаграммы классов**

Диаграмма классов служит для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования.

Диаграмма классов может отражать, в частности, различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений.

Класс состоит из трёх частей:

1. название класса;
2. атрибуты;
3. операции.

Также существуют отношения между классами. Они подразделяются на следующие виды:

1. ассоциация;
2. обобщение;
3. зависимость.

Диаграмма классов представлена на рисунке 2.5.

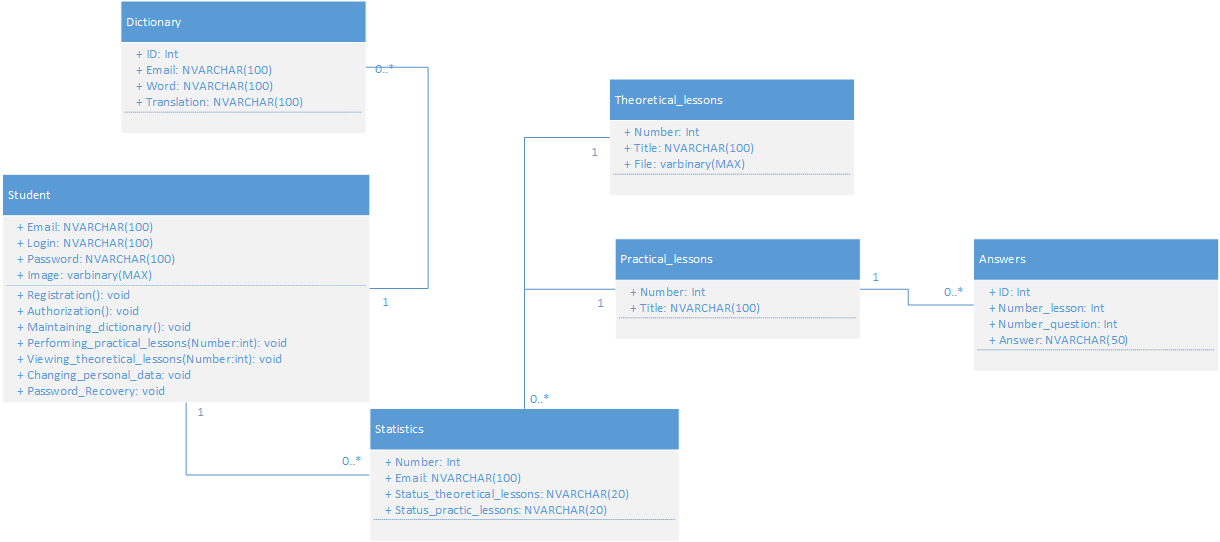


Рисунок 2.5 – Диаграмма классов

## **2.6 Проектирование диаграммы последовательностей**

Эта диаграмма описывает отношения объектов в различных условиях. Условия взаимодействия задаются сценарием, полученным на этапе разработки диаграмм вариантов использования.

Диаграмма последовательности содержит в себе: объекты, линии жизни, полосы активности, сообщения, условия, циклы.

Для построения этой диаграммы выполняются следующие шаги:

1. добавить объекты;
2. указать линии жизни;
3. добавить полосы активности на линиях жизни;
4. отобразить основные взаимодействия;
5. при необходимости добавить условия или циклы;
6. указать сообщения.

Для данного приложения были созданы диаграммы последовательностей для следующих прецедентов: «Авторизация», «Регистрация», «Просмотр занятий», «Восстановление пароля», «Редактирование личных данных». Эти диаграммы изображены на рисунках 2.6 – 2.10.

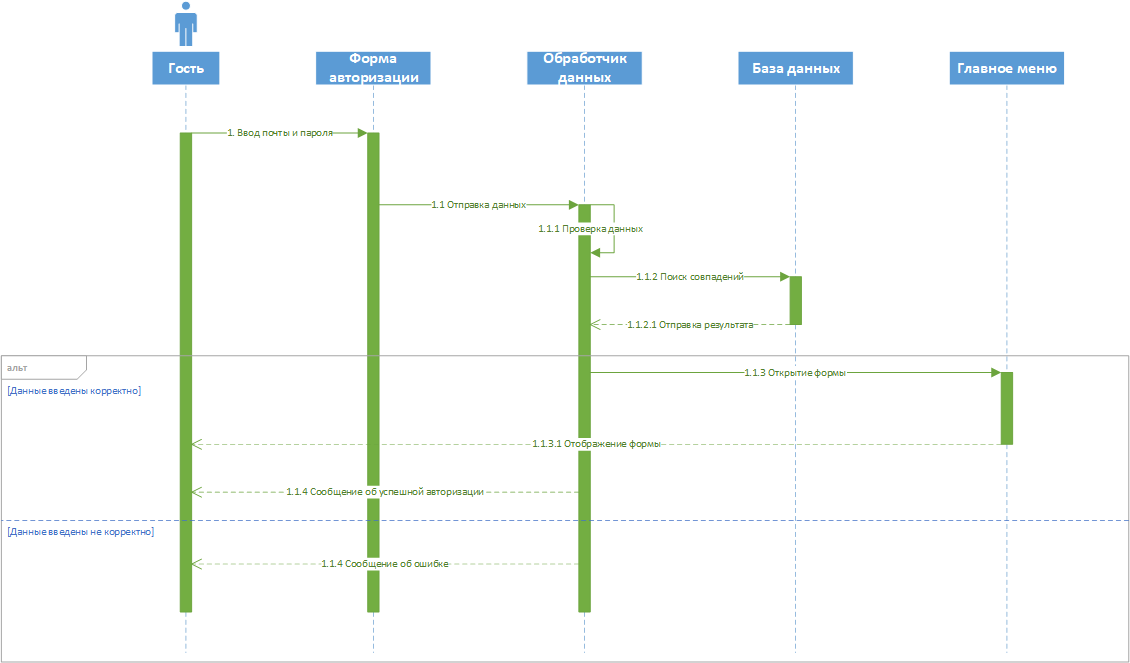


Рисунок 2.6 – Диаграмма последовательности для прецедента «Авторизация»

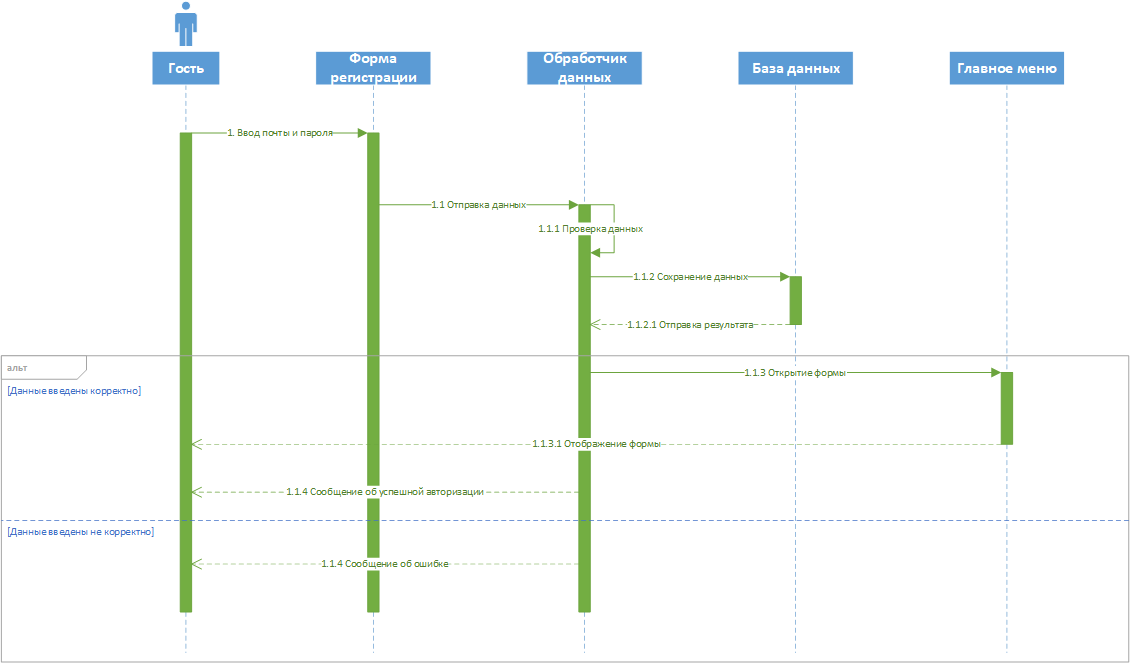


Рисунок 2.7 – Диаграмма последовательности для прецедента «Регистрация»

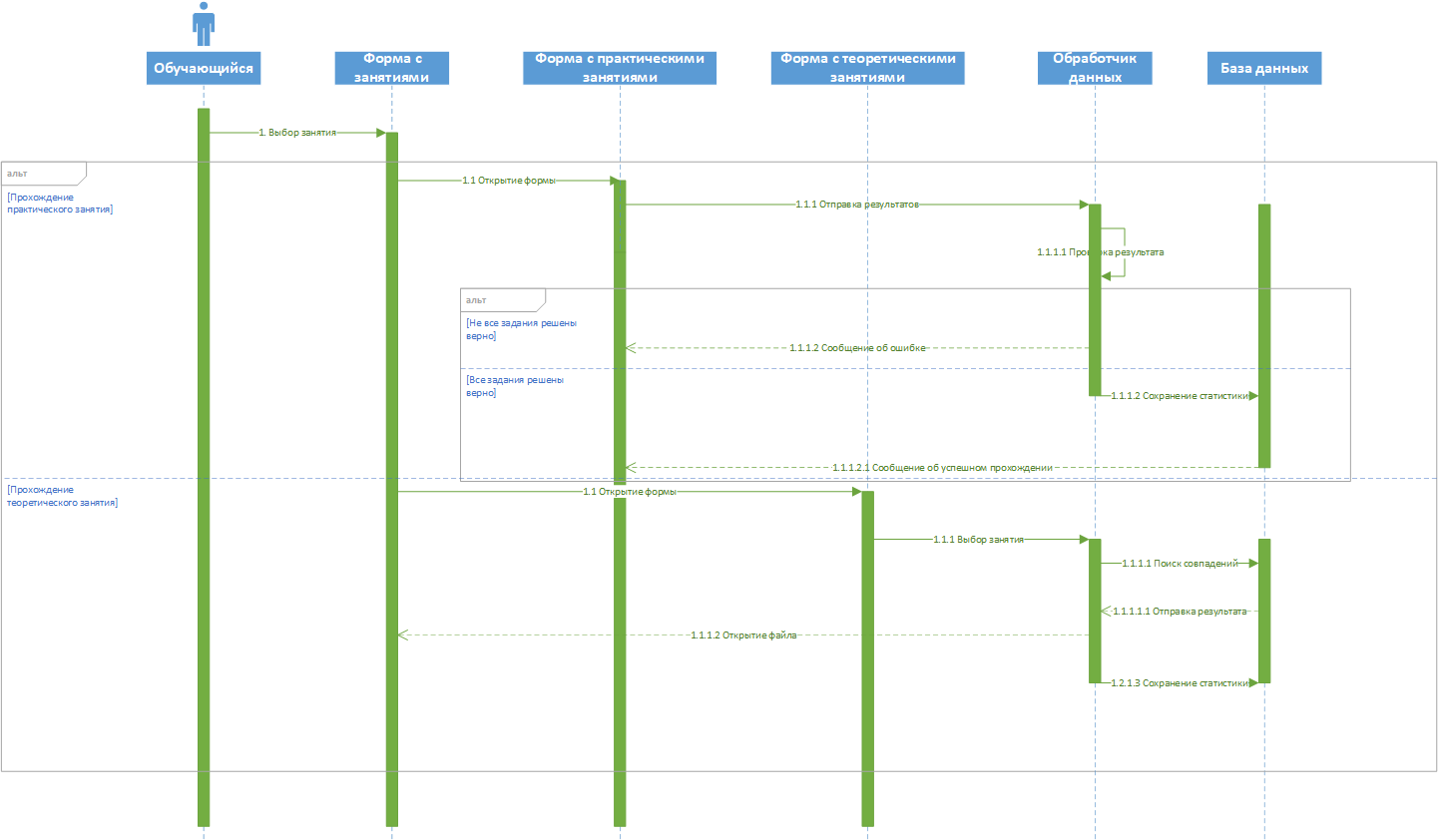


Рисунок 2.8 – Диаграмма последовательности для прецедента «Просмотр занятий»

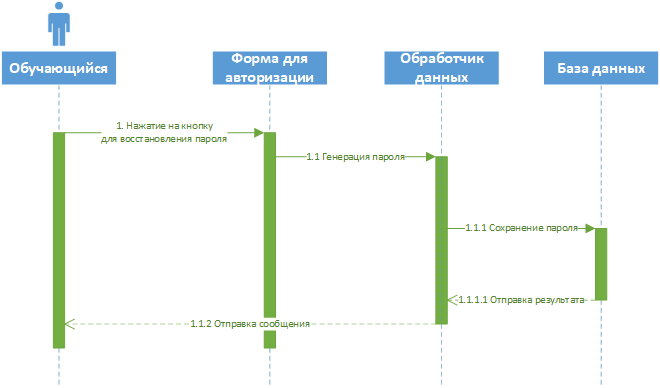


Рисунок 2.9 – Диаграмма последовательности для прецедента «Восстановление пароля»

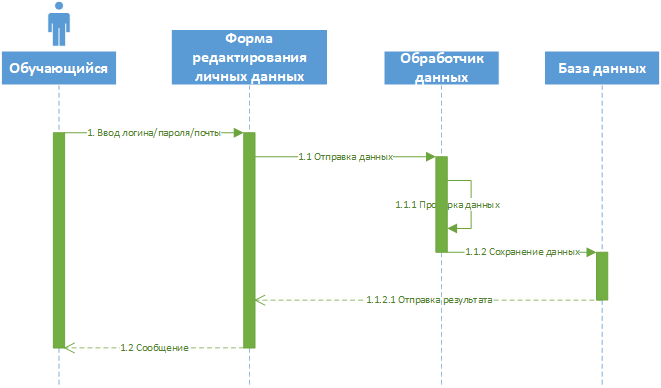


Рисунок 2.10 – Диаграмма последовательности для прецедента «Изменение личных данных»

## **2.7 Проектирование диаграммы сущность-связь**

Диаграмма сущность-связь предназначена для моделей данных и обеспечивает стандартный способ определения данных и отношений между ними.

Для нашей ИС данная диаграмма отображена на рисунке 2.11.

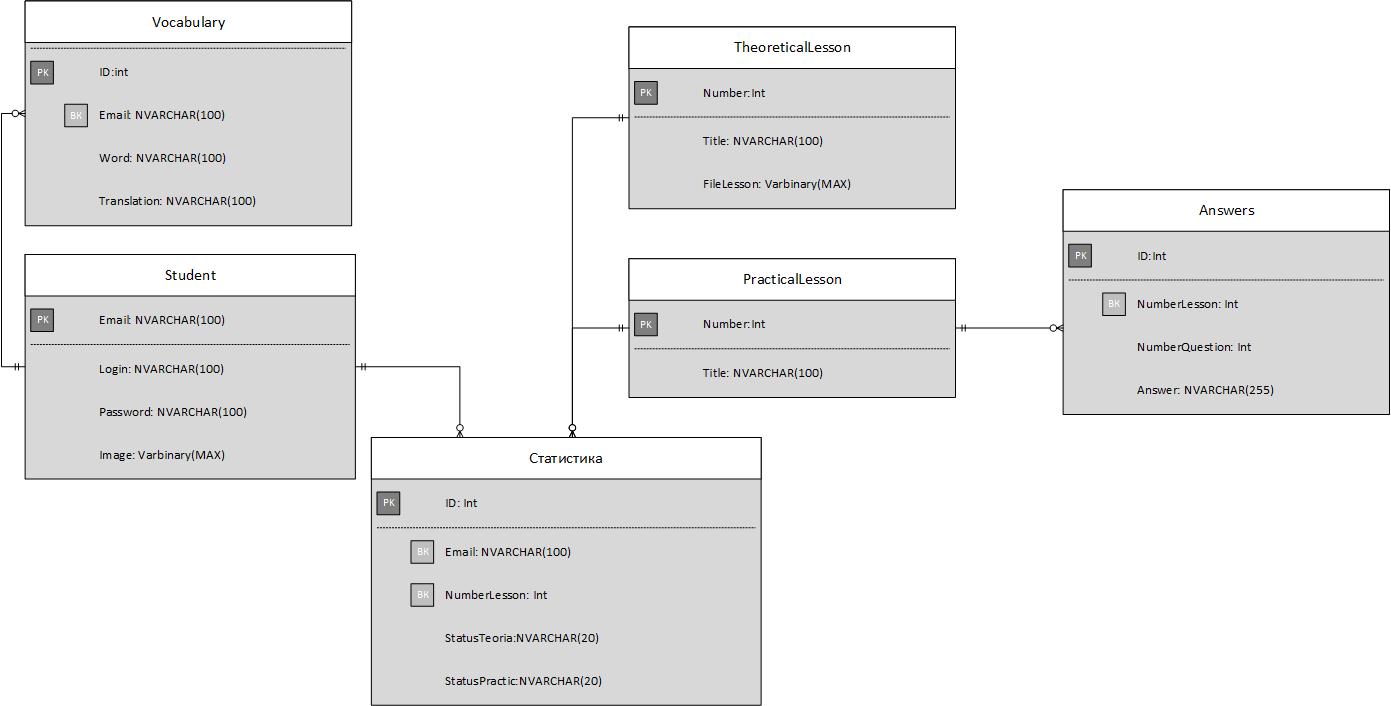


Рисунок 2.11 – Диаграмма сущность связь

## **2.8 Разработка пользовательского интерфейса**

Первый экран – приветствие. На нём находится описание приложения, а также ссылка на telegram, где можно задать свои вопросы. Данный экран показан на рисунке 2.12.

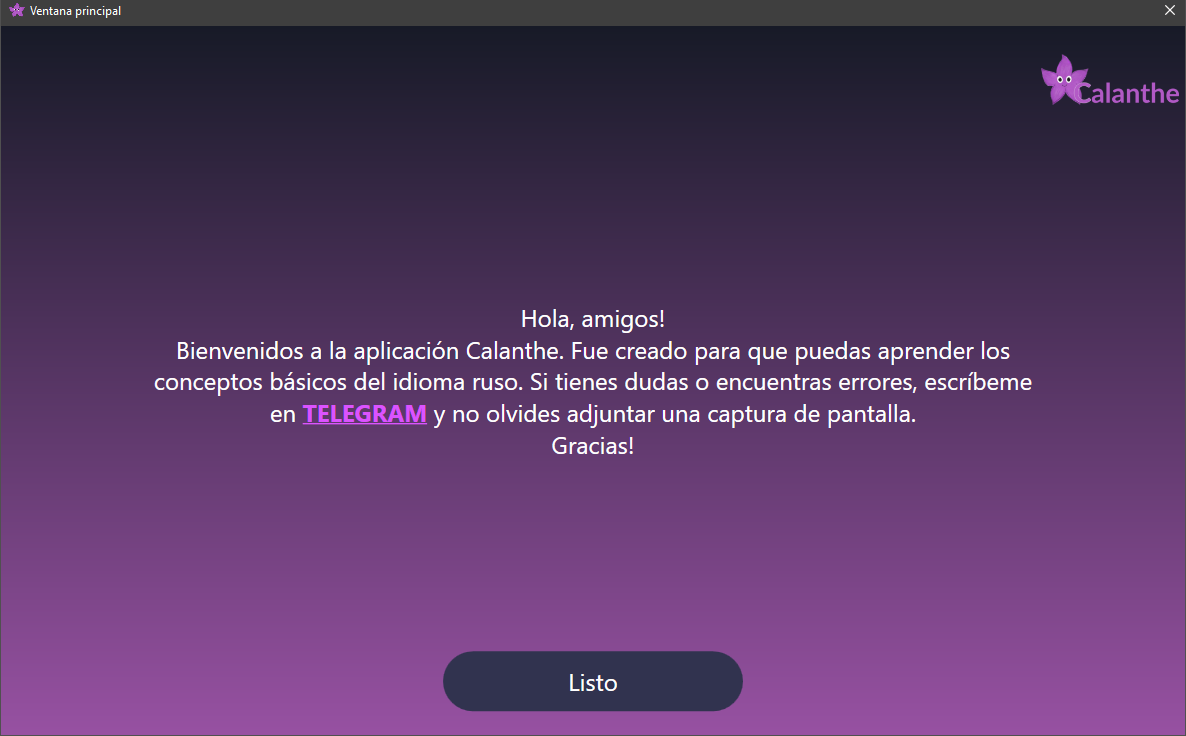


Рисунок 2.12 – Окно приветствия

Нажав на кнопку с названием «Listo», мы переходим в окно авторизации, которое изображено на рисунке 2.13. Чтобы войти в аккаунт нам необходимо ввести электронную почту и пароль. Если пользователь забыл свой пароль, то он может восстановить его, введя электронную почту и нажав на «Recuperar contrasena». Восстановление пароля возможно только, если компьютер подключён к интернету, в противном случае программа выдаст ошибку. Поле пароля имеет два состояния, его можно скрывать и отображать снова, нажав на кнопку с иконкой глаза.

Также пользователь может вернуться в предыдущее окно или перейти в окно регистрации.

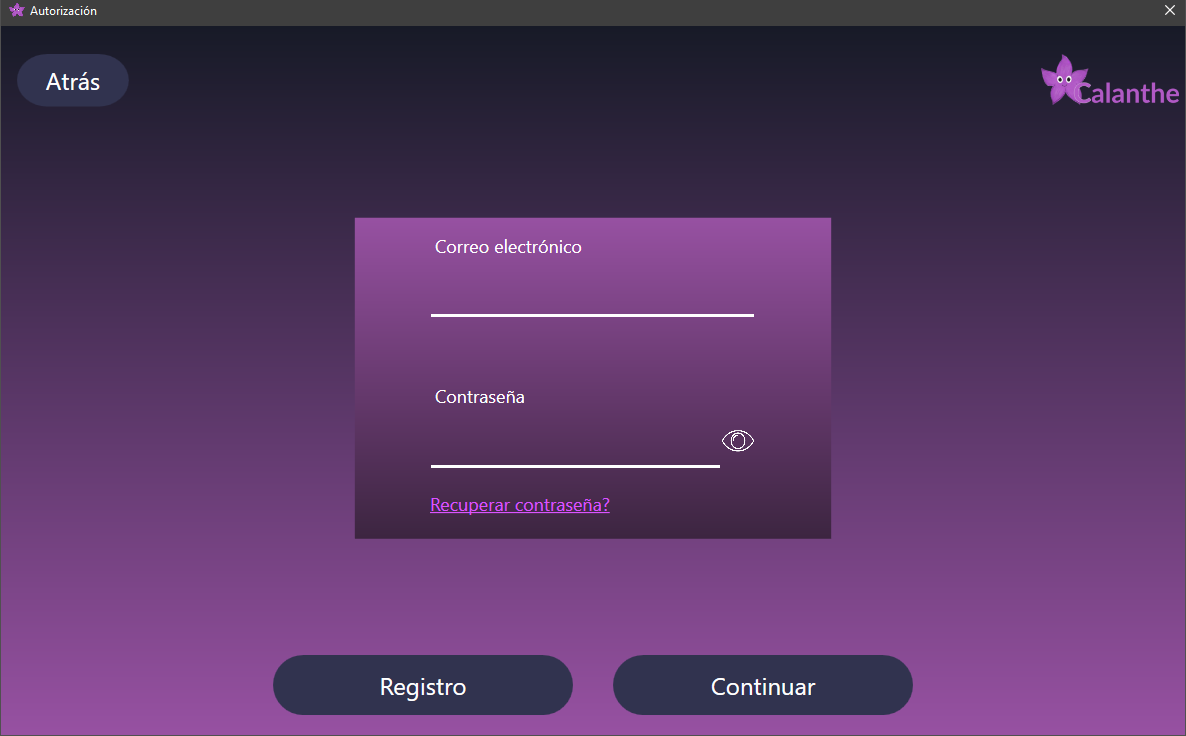


Рисунок 2.13 – Окно авторизации

Окно регистрации изображено на рисунке 2.14. Для регистрации необходимо придумать логин, пароль и ввести электронную почту. Пользователь имеет право вернуться в окно авторизации, а также, при необходимости, скрыть свой пароль.

В данном окне входные данные должны быть следующего типа:

1. логин может состоять из любых символов;
2. почта должна иметь вид: \_\_\_\_\_ @ \_\_ . \_\_;
3. пароль должен содержать в себе более 8 символов.

На рисунке 2.15 представлено окно меню, которое встречает пользователя, когда он входит в систему.

Меню состоит из трёх кнопок:

1. кнопка для перехода в окно профиля;
2. кнопка для перехода в окно с занятиями;
3. кнопка для перехода в окно со словарём.

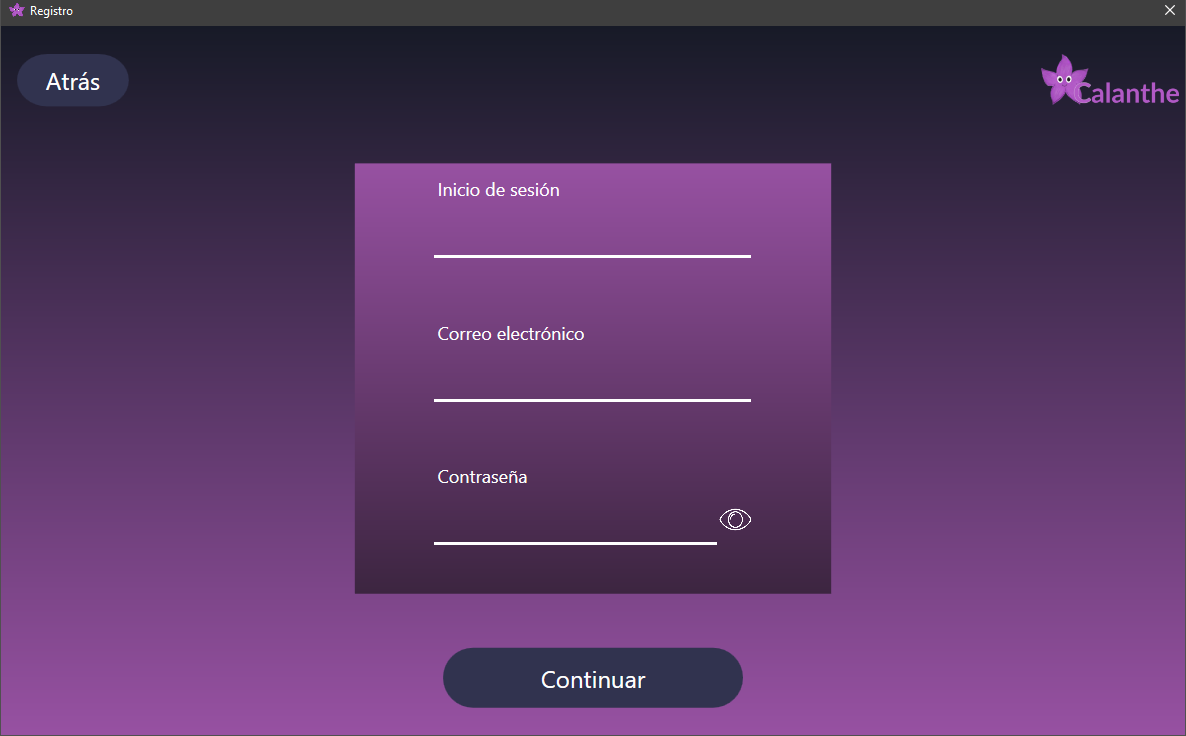
А так же можно вернуться назад.

Рисунок 2.14 – Окно регистрации

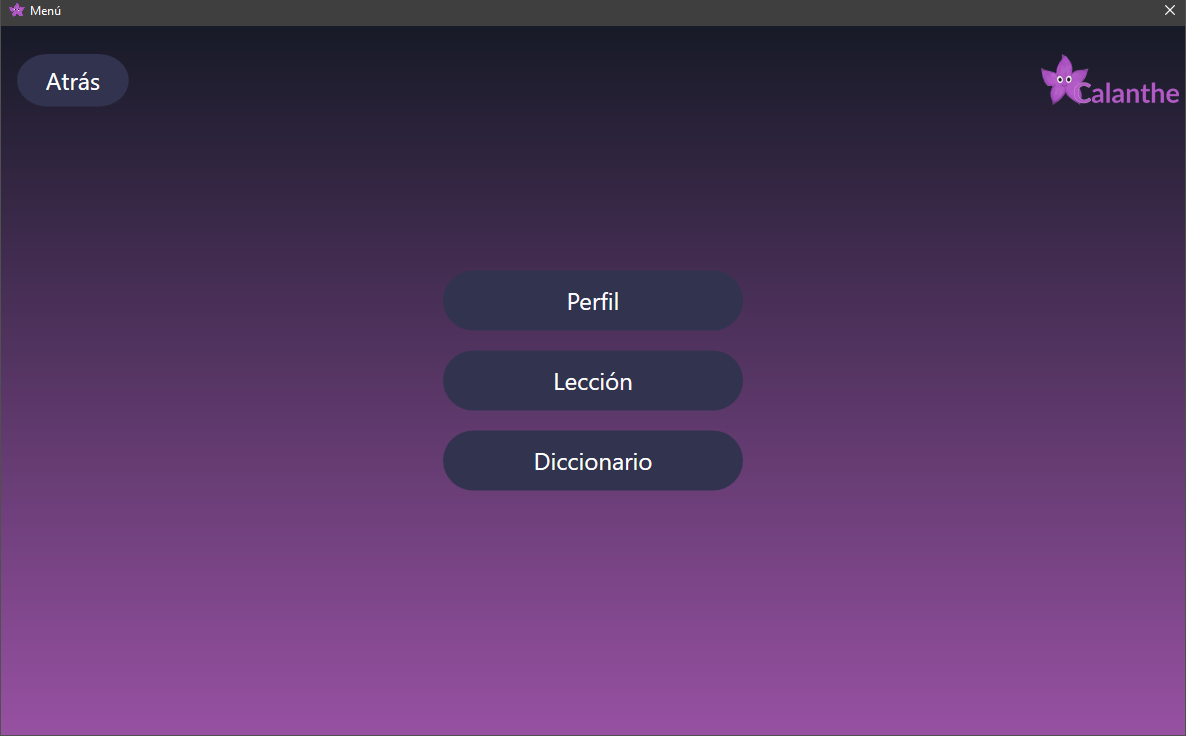


Рисунок 2.15 – Окно меню

На рисунке 2.16 отображено окно профиля обучающегося. На нём можно увидеть его фотографию, статистику прохождения занятий, кнопку возврата и кнопку для перехода в окно редактирования профиля.

Статистика формируется следующим образом: если теоретическое занятие было пройдено и тест был выполнен без ошибок, то один блок будет загораться более ярким цветов, в случае, если практическое или теоретическое занятие не будет пройдено, то блок загораться не будет.



Рисунок 2.16 – Окно профиля

В окне редактирования профиля обучающегося, которое представлено на рисунке 2.17, открывается следующий функционал: возможность изменять логин, изменять и скрывать пароль, менять или удалять фото профиля. Также доступна кнопка для возврата и кнопка выхода из аккаунта.

В программе доступно 16 занятий. Каждое занятие подразделяется на практическое и теоретическое. Теоретическое занятие сохраняется на компьютер в виде pdf - файла, а практическое занятие создано в форме теста. Их список показан на рисунке 2.18.



Рисунок 2.17 – Окно редактирования профиля

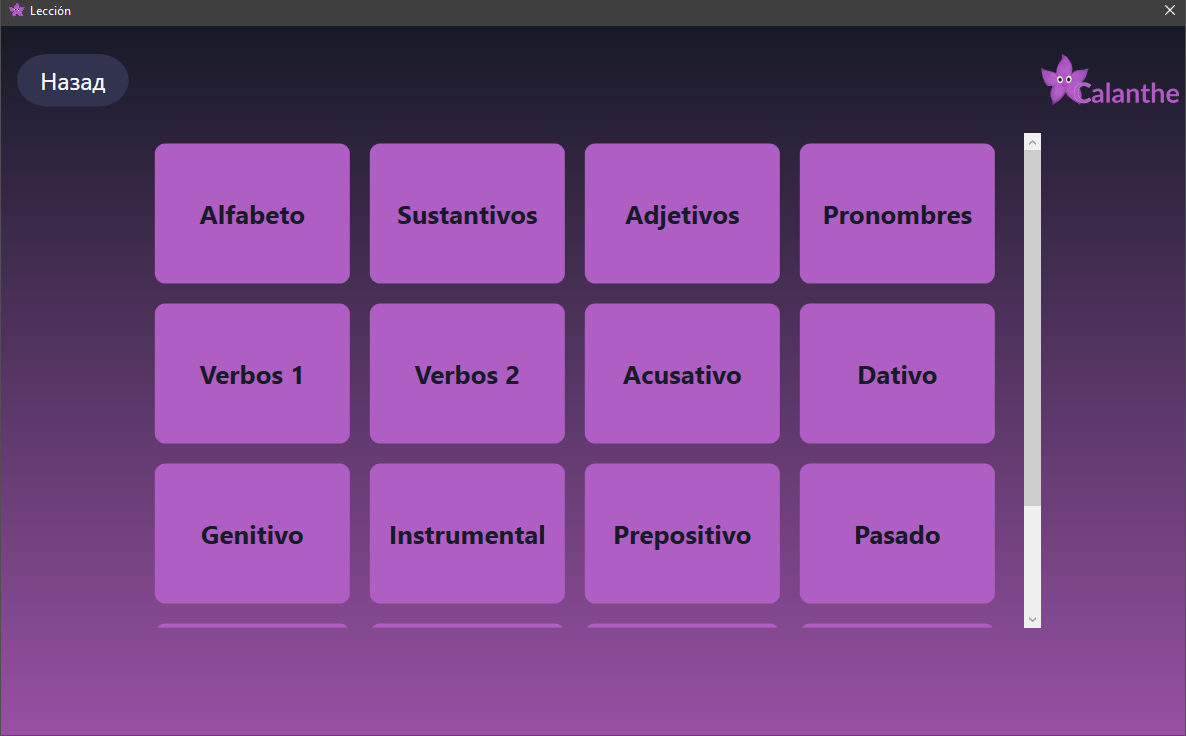


Рисунок 2.18 – Окно списка занятий

Если нажать на занятие, то откроется меню, которое показано на рисунке 2.19. Система предложит выбрать тип занятия. А также, в данном окне расположена кнопка для перехода в окно со словарём и возврата.

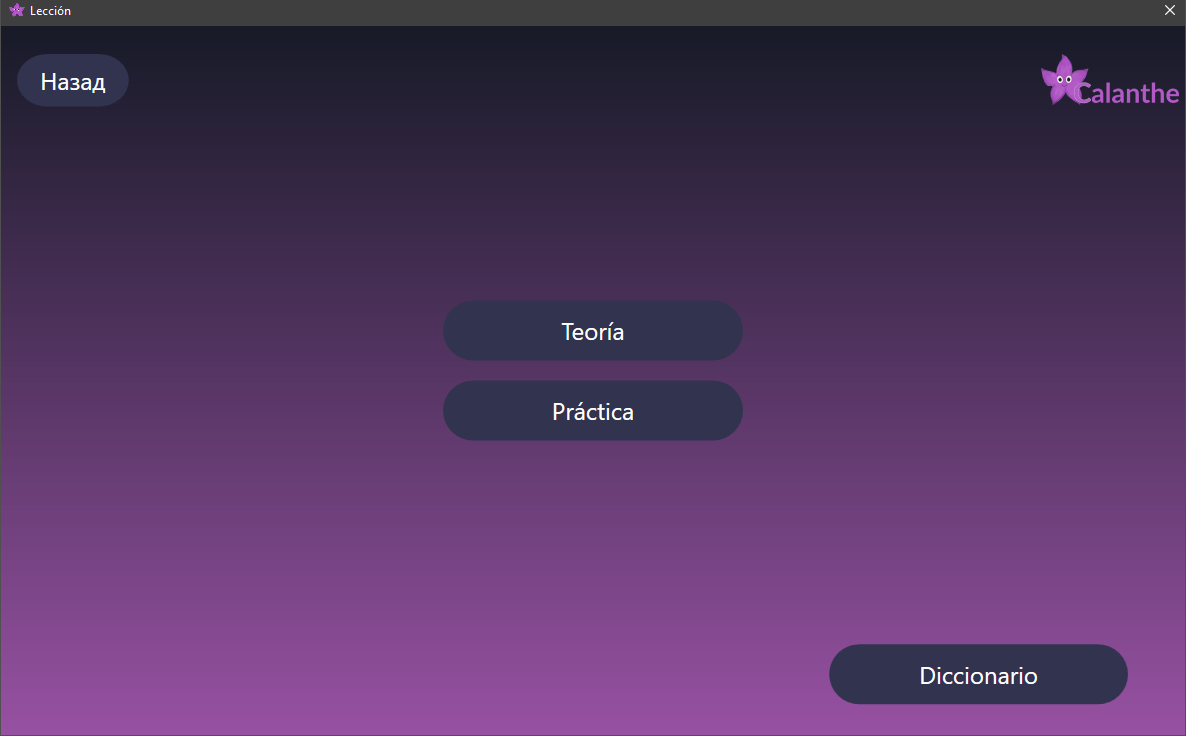


Рисунок 2.19 – Окно меню для выбора типа занятия

Выбрав практическое занятие, открывается тест, как показано на рисунке 2.20, пройдя который, система зафиксирует результат, если тест был решён правильно.

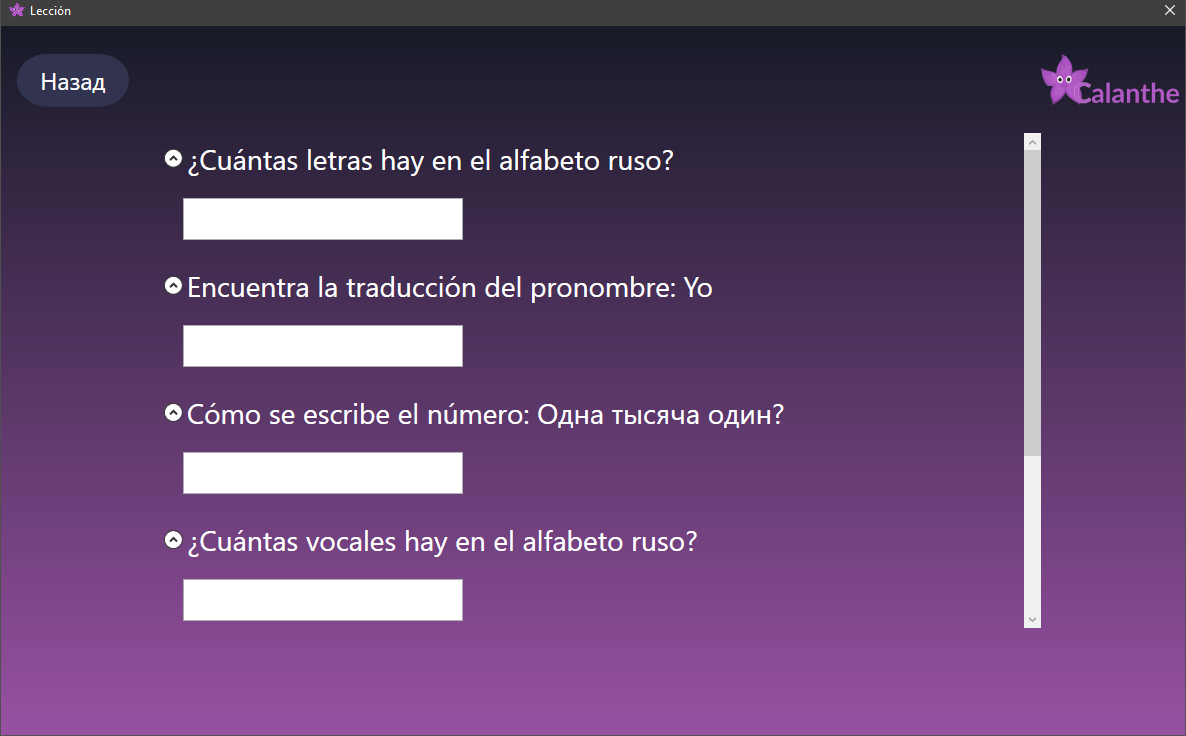


Рисунок 2.20 – Практическое занятие

Для того, чтобы воспользоваться теоретическим занятием, нужно выбрать папку, куда сохранится файл. Изображение окна представлено на рисунке 2.21. После сохранения, файл автоматически откроется в формате pdf.

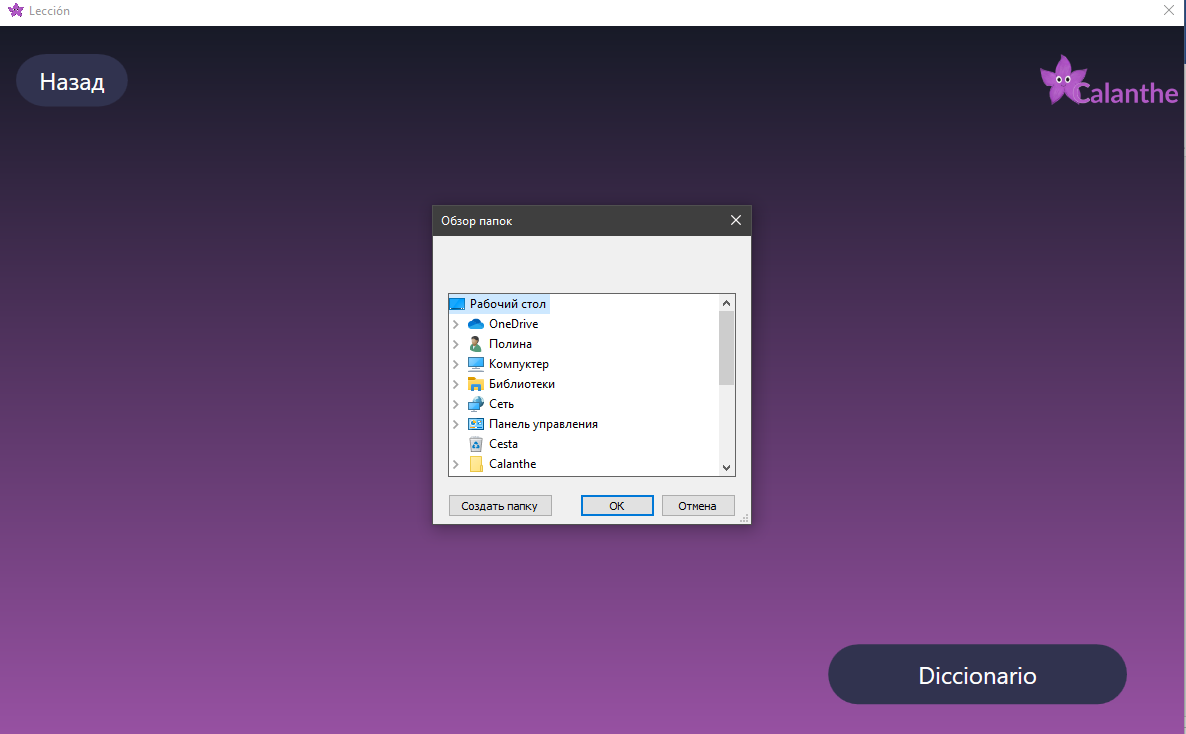


Рисунок 2.21 – Сохранение теоретического занятия

Внешний вид теоретического занятия показан на рисунке 2.22.

Окно с выводом словаря показано на рисунке 2.23. Система предоставляет пользователю следующие возможности: поиск необходимого слова, добавление, удаление, редактирования слов, а так же в данном окне расположены кнопки для возврата к меню и перехода в окно с занятиями.

Реализация окон для добавления, удаления и редактирования слов представлена на рисунках 2.24 – 2.26.

Чтобы добавить слово, необходимо ввести само слово и его перевод. При удалении слова, Вы перейдёте в окно, в котором система отобразит его, чтобы пользователь убедился в правильности выбора. При редактировании, система так же отобразит слово и перевод, чтобы обучающему было легче изменить его.

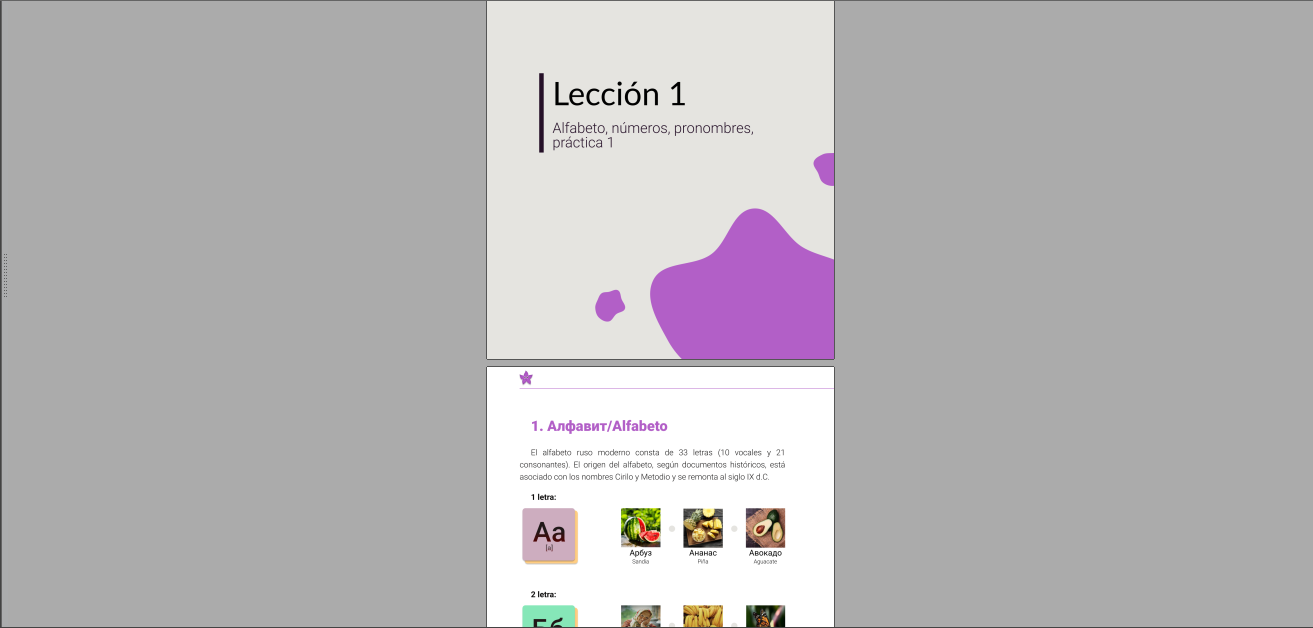


Рисунок 2.22 – Внешний вид занятия

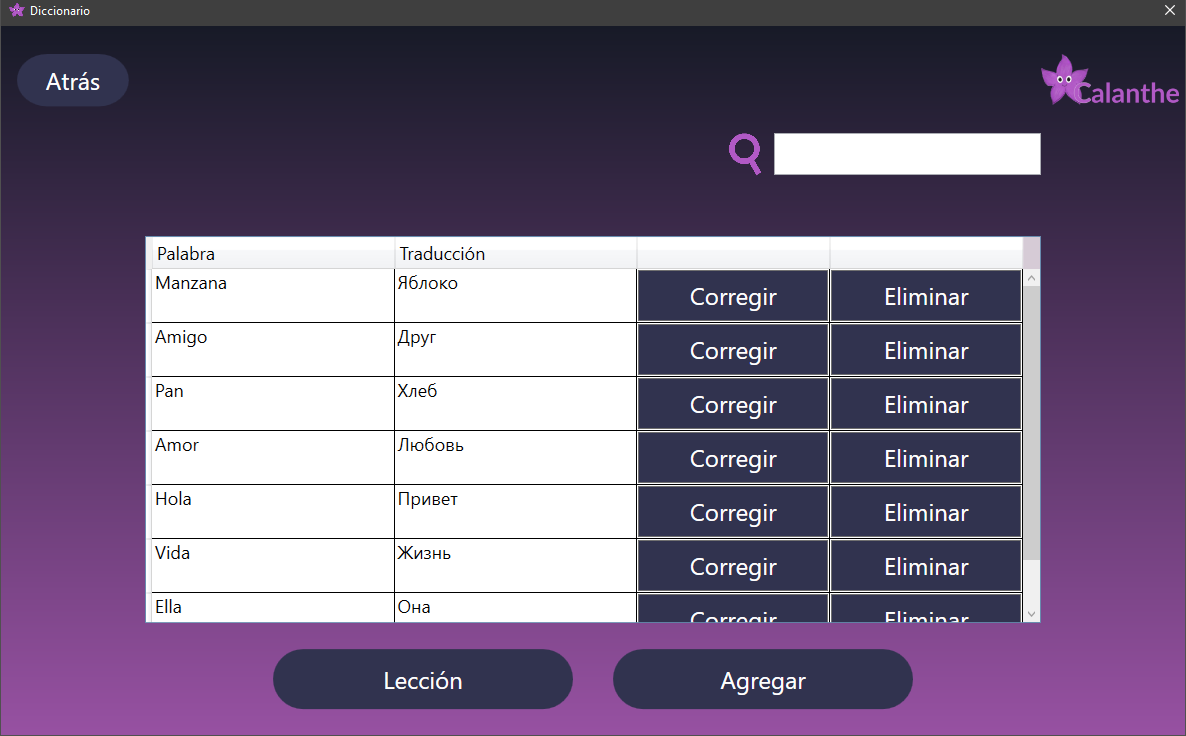


Рисунок 2.23 – Окно со словарём

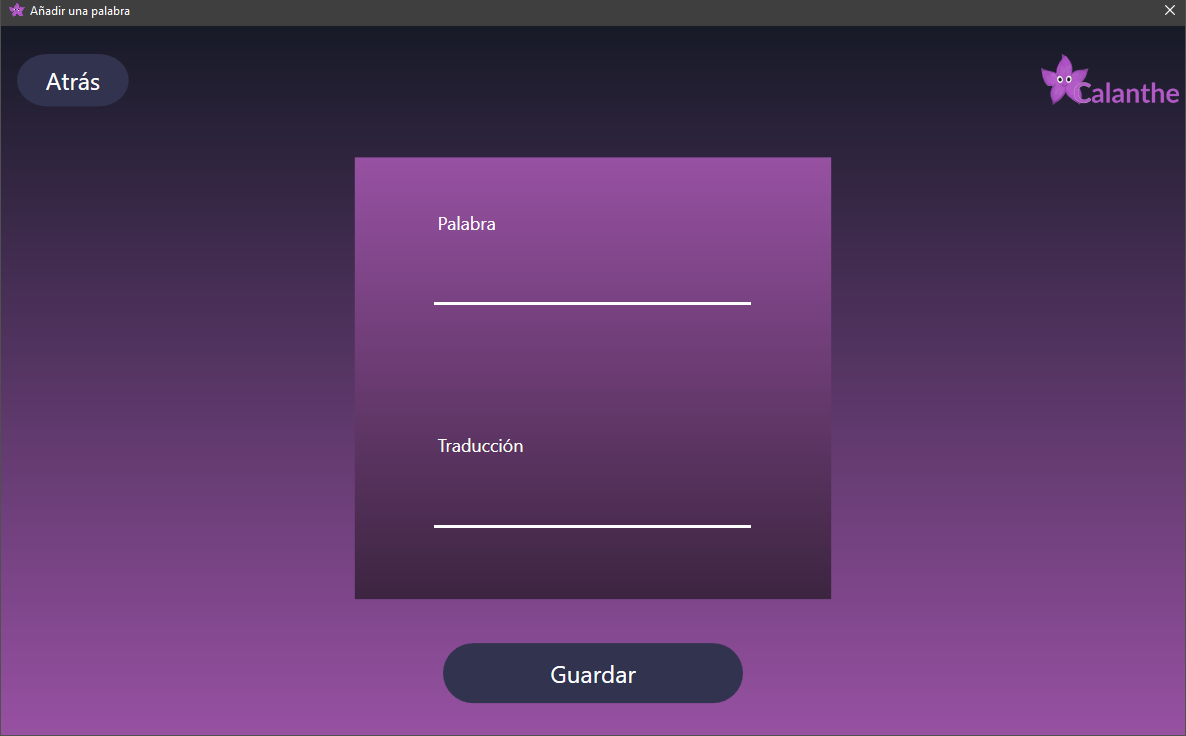


Рисунок 2.24 – Окно для добавления слова

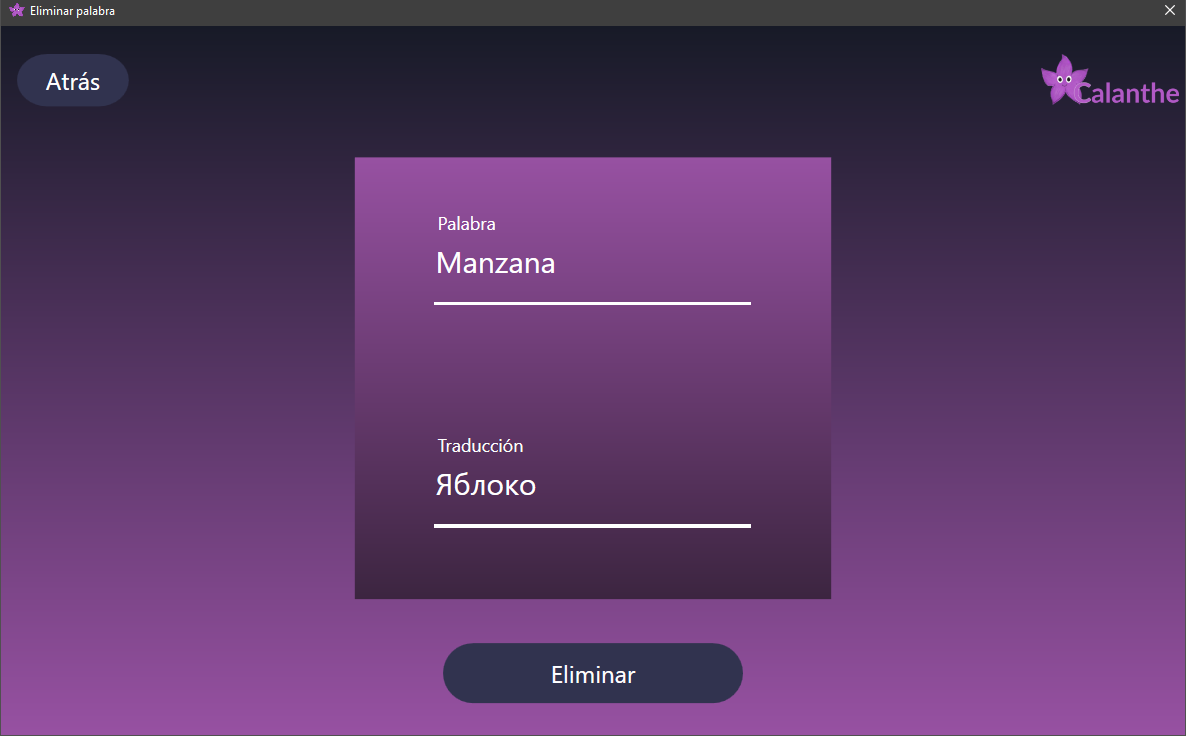


Рисунок 2.25 – Окно для удаления слова

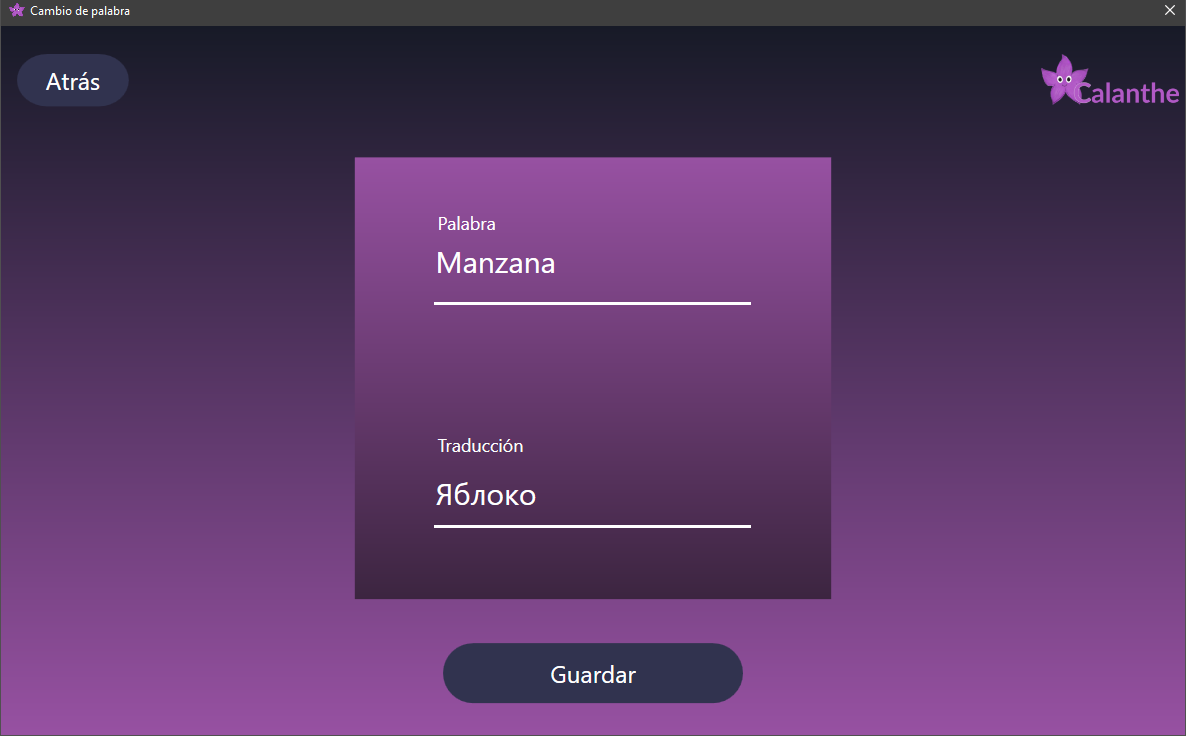


Рисунок 2.26 – Окно для редактирования слова

# ГЛАВА 3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

В данной главе описывается процесс создания схемы базы данных и определение ограничений целостности.

Опираясь на диаграмму сущность – связь, схема которой находится в главе 2 пункте 2.7, создадим таблицы в SQL server.

Начнём с таблицы «Answers», которая представлена на рисунке 3.1. В данной таблице будут следующие столбцы:

1. ID: ИД, тип данных – int;
2. NumberLesson: номер занятия, тип данных – int;
3. NumberQuestion: номер вопроса, тип данных – int;
4. Answer: ответ, тип данных – nvarchar(50).

Содержание этой таблицы показано на рисунке 3.2.

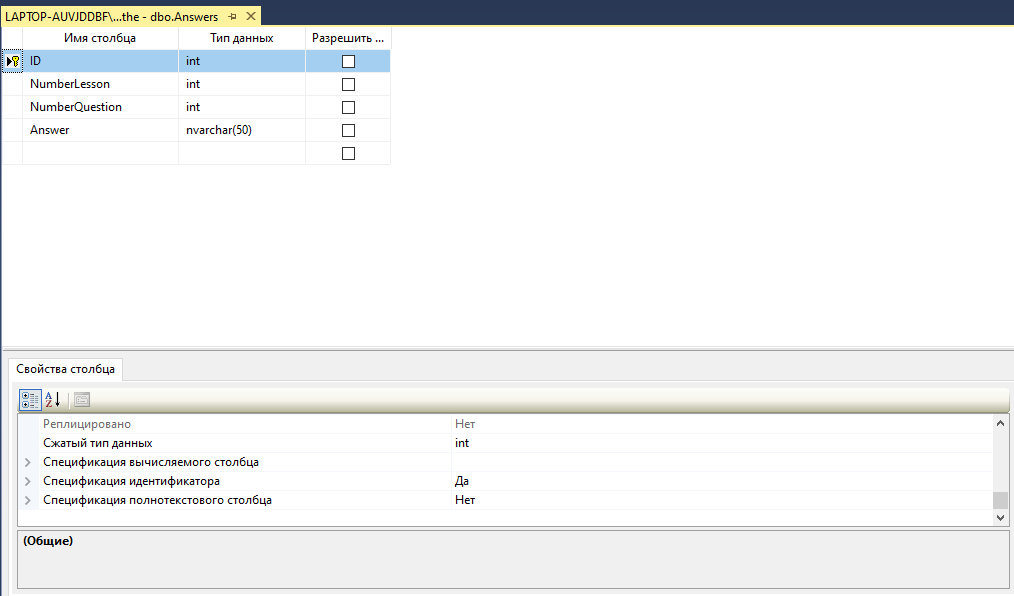
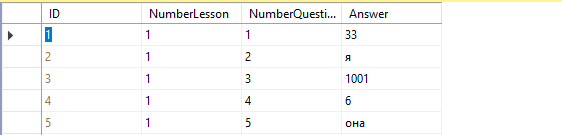


Рисунок 3.1 – Дизайнер таблицы «Answers» Рисунок 3.2 – Содержание таблицы «Answers»

Далее идёт таблица на рисунке 3.3 – «PracticalLesson», а её содержание представлено на рисунке 3.4. Для этой таблицы создадим столбцы:

1. Number: номер, тип данных – int;
2. Title: название, тип данных – nvarchar(100).

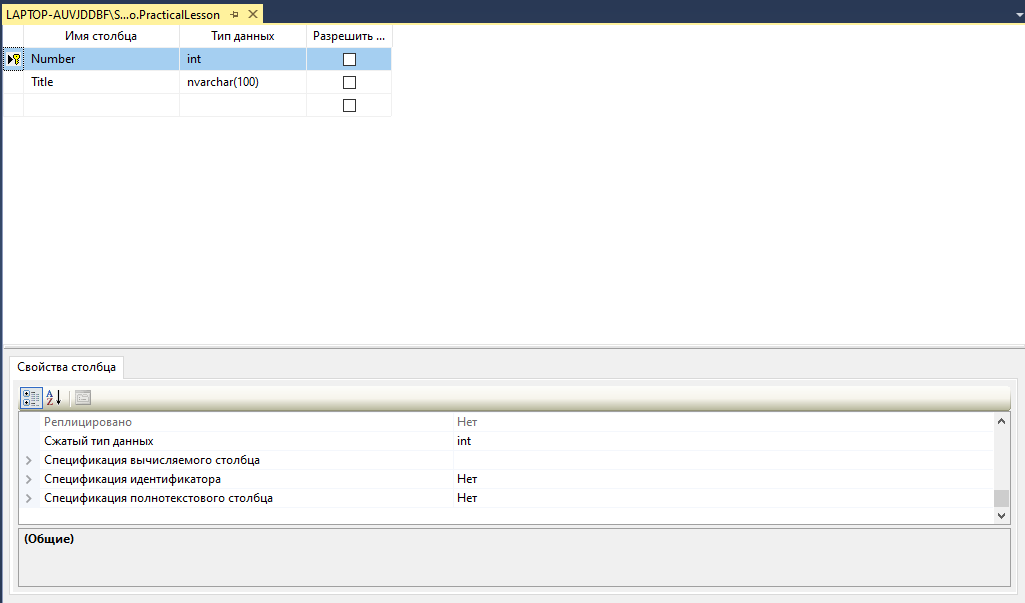


Рисунок 3.3 – Дизайнер таблицы «PracticalLesson»



Рисунок 3.4 – Содержание таблицы «PracticalLesson»

Таблица «Statistics» на рисунке 3.5 содержит в себе:

1. ID: номер, тип данных – int;
2. Email: почта, тип данных – nvarchar(100);
3. NumberLesson: номер занятия, тип данных – int;
4. StatusTeoria: статус теоретического занятия, тип данных –nvarchar(20);
5. StatusPractic: статус практического занятия, тип данных – nvarchar(20).

Содержание данной таблицы представлено на рисунке 3.6.

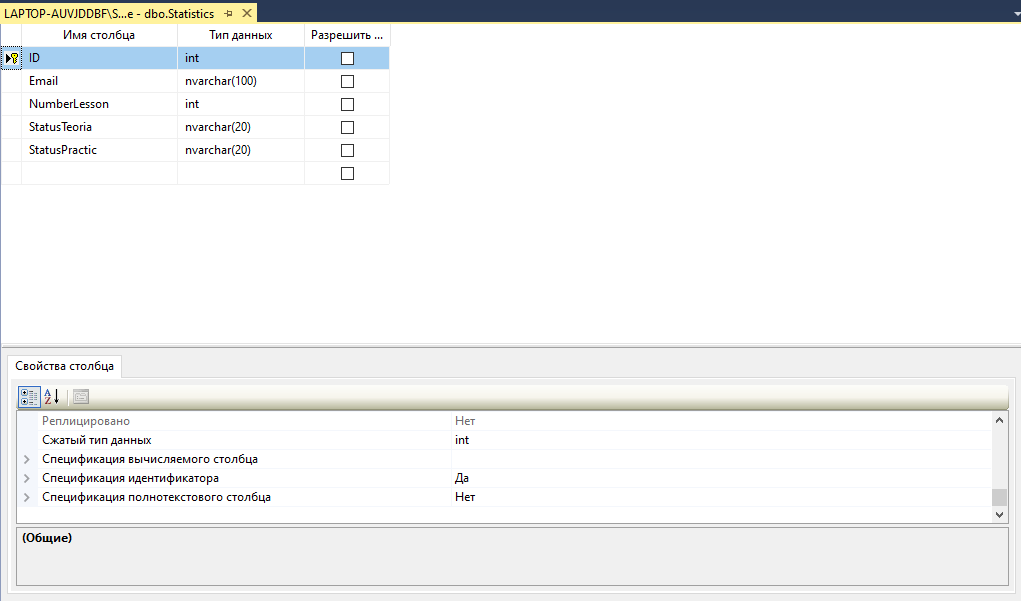


Рисунок 3.5 – Дизайнер таблицы «Statistics»

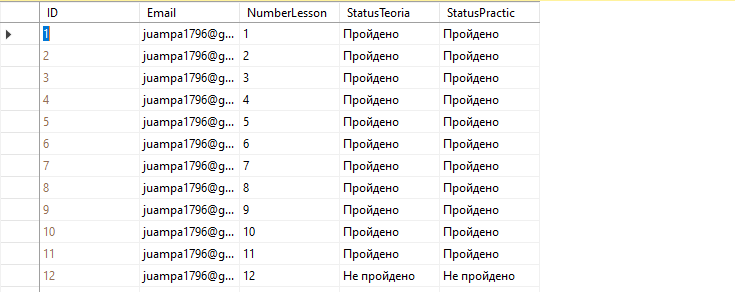


Рисунок 3.6 – Содержание таблицы «Statistics»

Следующая таблица – «Student», представленная на рисунке 3.7, состоит из следующих столбцов:

1. Email: почта, тип данных – nvarchar(100);
2. Login: логин, тип данных – nvarchar(100);
3. Password: пароль, тип данных – nvarchar(100);
4. Image: фото профиля, тип данных – varbinary(MAX).

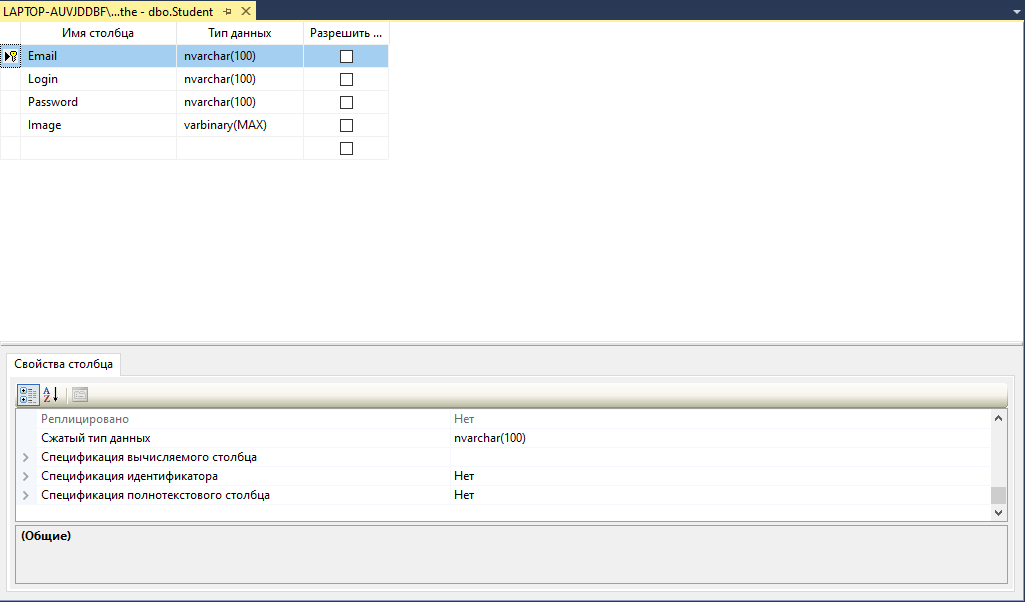


Рисунок 3.7 – Дизайнер таблицы «Student»

Содержание этой таблицы можно увидеть на рисунке 3.8.

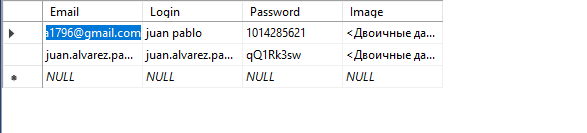


Рисунок 3.8 – Содержание таблицы «Student»

В следующей таблица – «TheoreticalLesson», которая находится на рисунке 3.9 – состоит из трёх столбцов:

1. Number: номер занятия, тип данных – int;
2. Title: название, тип данных – nvarchar(100);
3. FileLesson: файл занятия, тип данных – varbinary(MAX).

Её содержание показано на рисунке 3.10.

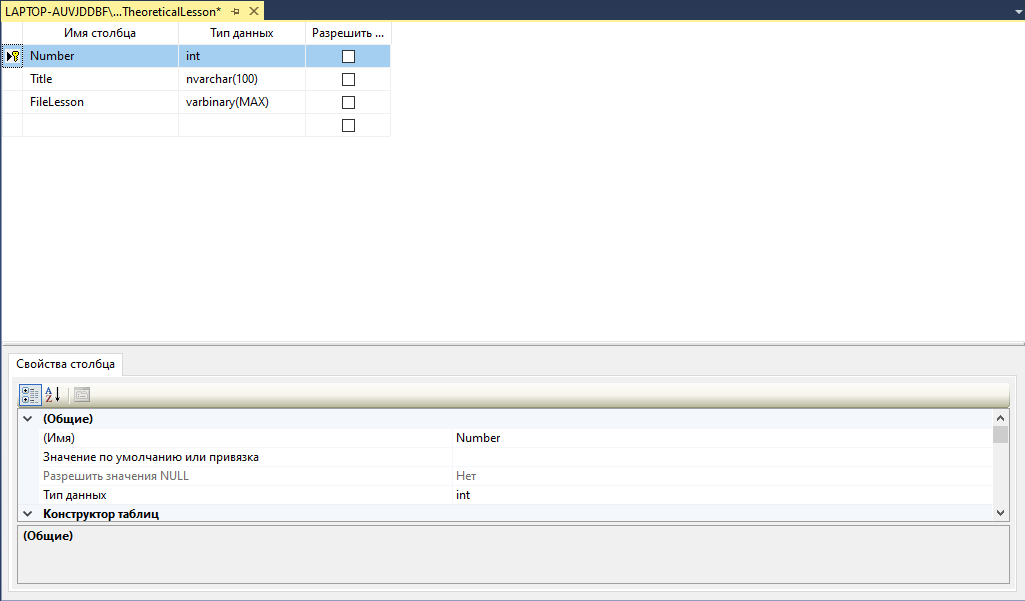


Рисунок 3.9 – Таблица «TheoreticalLesson»



Рисунок 3.10 – Содержание таблицы «TheoreticalLesson»

И последняя таблица называется «Vocabulary», которая изображена на рисунке 3.11, содержит в себе:

1. ID: номер занятия, тип данных – int;
2. Email: название, тип данных – nvarchar(100);
3. Word: файл занятия, тип данных – nvarchar(100);
4. Translation: файл занятия, тип данных – nvarchar(100).

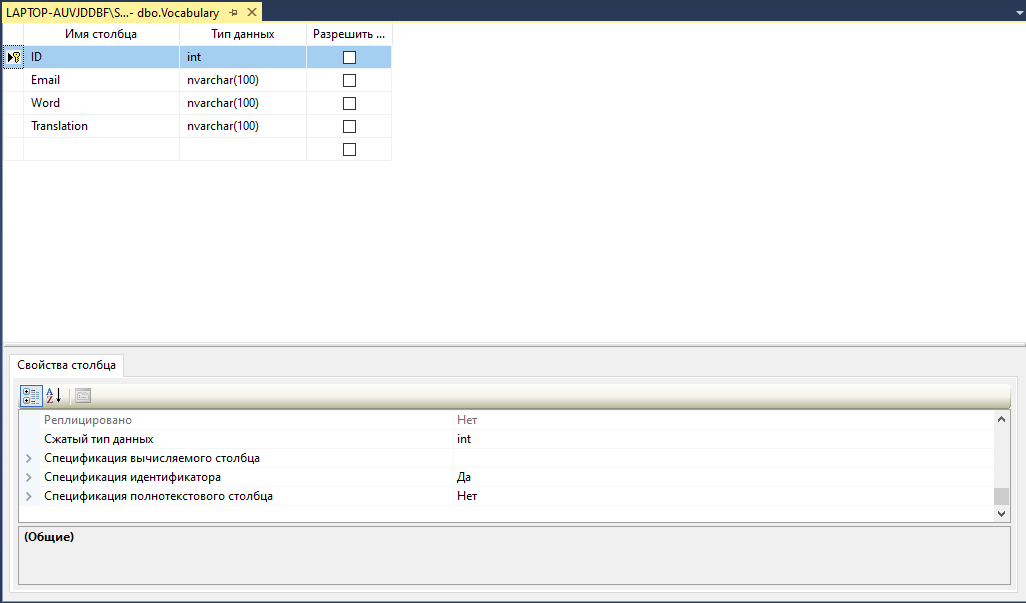


Рисунок 3.11 – Таблица «Vocabulary»

В её содержание входят данные, представленные на рисунке 3.12.



Рисунок 3.12 – Содержание таблицы «Vocabulary»

Собрав все таблицы и расставив связи, у нас, в итоге, получилась схема базы данных, показанная на рисунке 3.13.

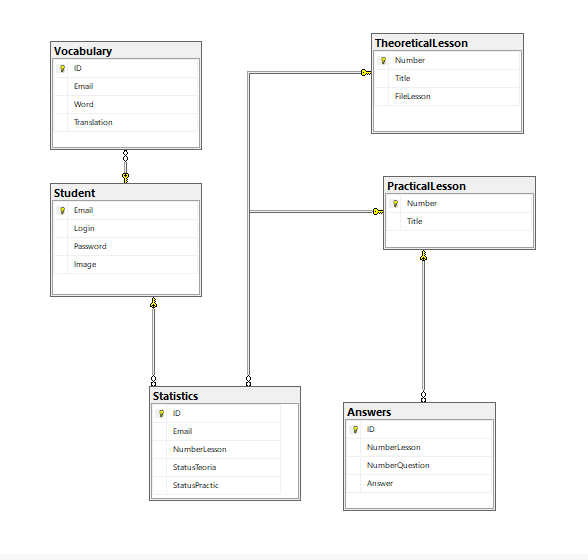


Рисунок 3.13 – Схема базы данных

# ГЛАВА 4 ПОЛЬЗОВАТЕЛЬСКАЯ ДОКУМЕНТАЦИЯ

Пользовательская документация представляет собой руководство пользователя и руководство по установке ПО, описывающее каждую функцию программы, а также шаги, которые нужно выполнить для использования этой функции.

## **4.1 Руководство по установке программного обеспечения**

Для того, что установить программу «Calanthe» необходимо запустить исполняемый файл Calanthe.exe, его иконка изображена на рисунке 4.1.

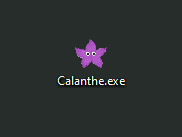


Рисунок 4.1 – Исполняемый файл Calanthe.exe

Запустив его, мы принимаем условия соглашение и нажимаем «Продолжить».

Далее открывается окно, которое представлено на рисунке 4.2. Оно позволяет пользователю выбирать папку, куда сохранится наша программа.

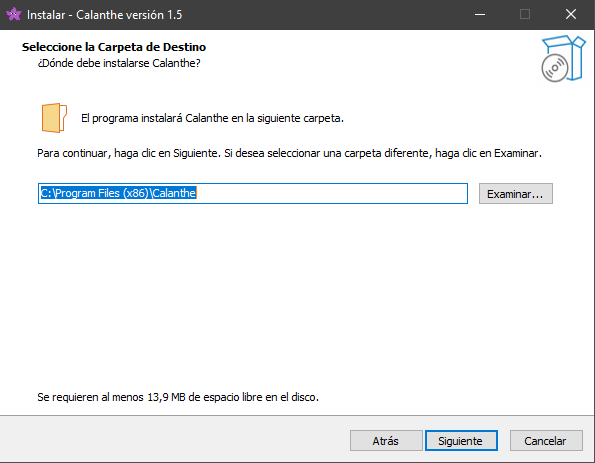


Рисунок 4.2 – Выбор папки для сохранения

Следующее, что нужно сделать – это решить, нужно ли создавать ярлык на рабочем столе. После того, как решение было принято, нужно нажать на «Продолжить» , как показано на рисунке 4.3.

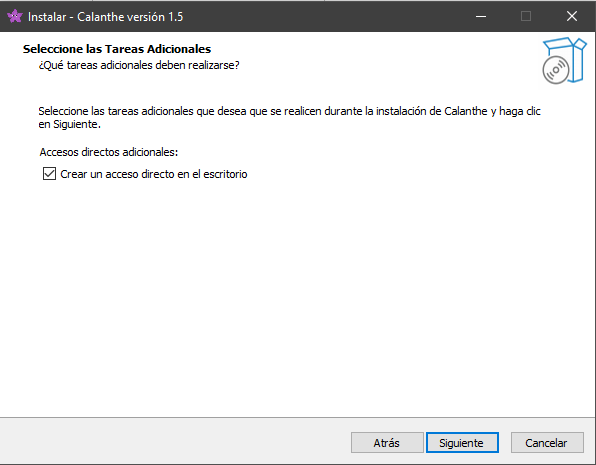


Рисунок 4.3 – Дополнительно

На рисунке 4.4 изображено окно, на котором нам нужно нажать на кнопку «Установить».

И последнее, что нам нужно сделать – подождать, пока программа установится на Ваш компьютер.

Если ранее Вы установили флажок, чтобы программа запустилась автоматически, то после установки откроется окно приветствия, как показано на рисунке 4.5.

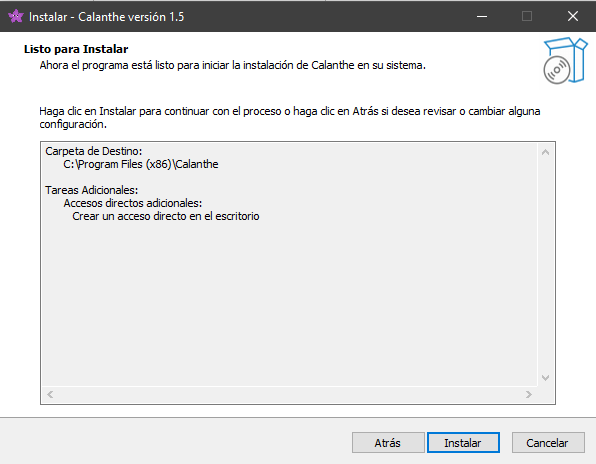


Рисунок 4.4 – Подтверждение установки

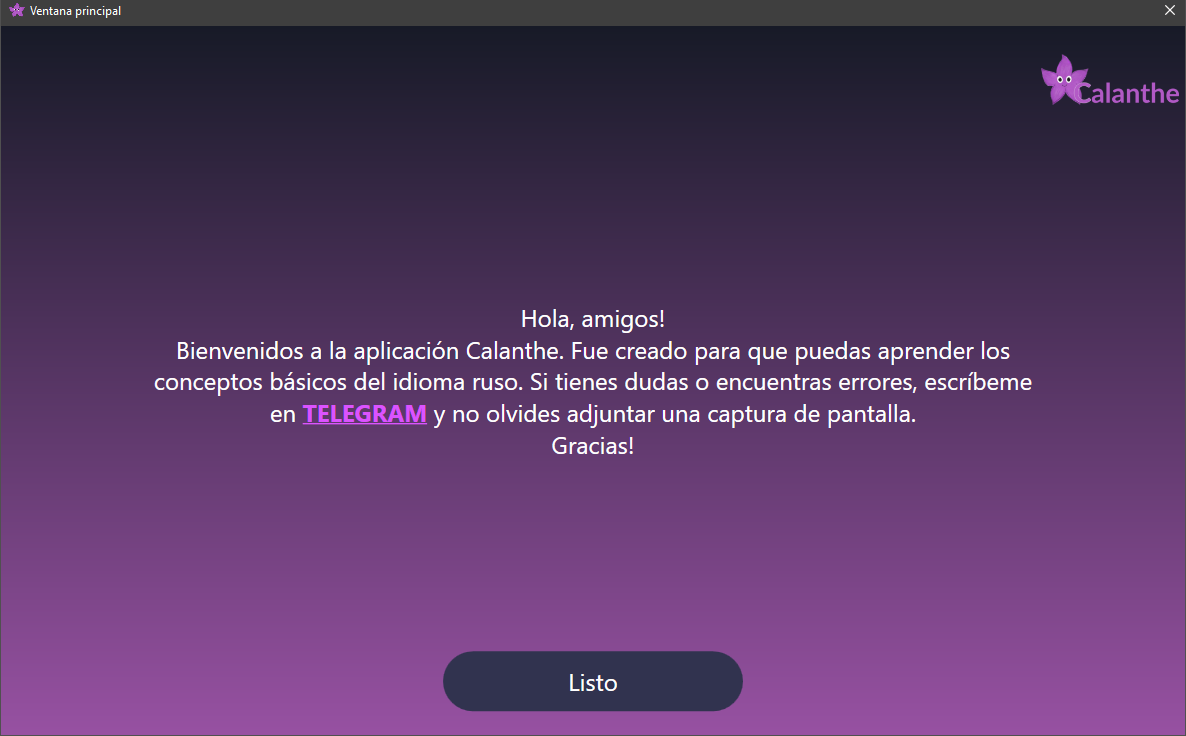


Рисунок 4.5– Окно приветствия

## **Руководство пользователя**

Настоящий документ является руководством пользователя по эксплуатации автоматизированной информационной системы «Calanthe».

Данное приложение создано с целью изучения русского языка, как иностранного, для испаноговорящих. Оно предназначено для образовательных учреждений, предоставляющих подготовительные курсы для иностранцев.

Основными возможностями ИС являются:

1. регистрация;
2. авторизация;
3. просмотр теоретических занятий;
4. просмотр практических занятий;
5. редактирование профиля;
6. просмотр статистики:
7. ведение словаря.

Все пользователи должны иметь навыки работы с операционными системами семейства Microsoft Windows (7/8/8.1/10/11).

Основным предметом автоматизации являются просмотр теоретических занятий, выполнение практических занятий и редактирование словаря.

Работа пользователей возможна при выполнении следующих требований к рабочему месту:

* [требования к программному обеспечению](#page7);
* [требования к техническому обеспечению.](#page8)

Перед началом работы с ИС «Calanthe» необходимо ознакомиться с Руководством по установке, описанным в предыдущем пункте.

А теперь начнём знакомство с системой. Первый экран – приветствие. На нём находится описание приложения, а так же ссылка на telegram, где можно задать свои вопросы. Данный экран показан на рисунке 4.6.

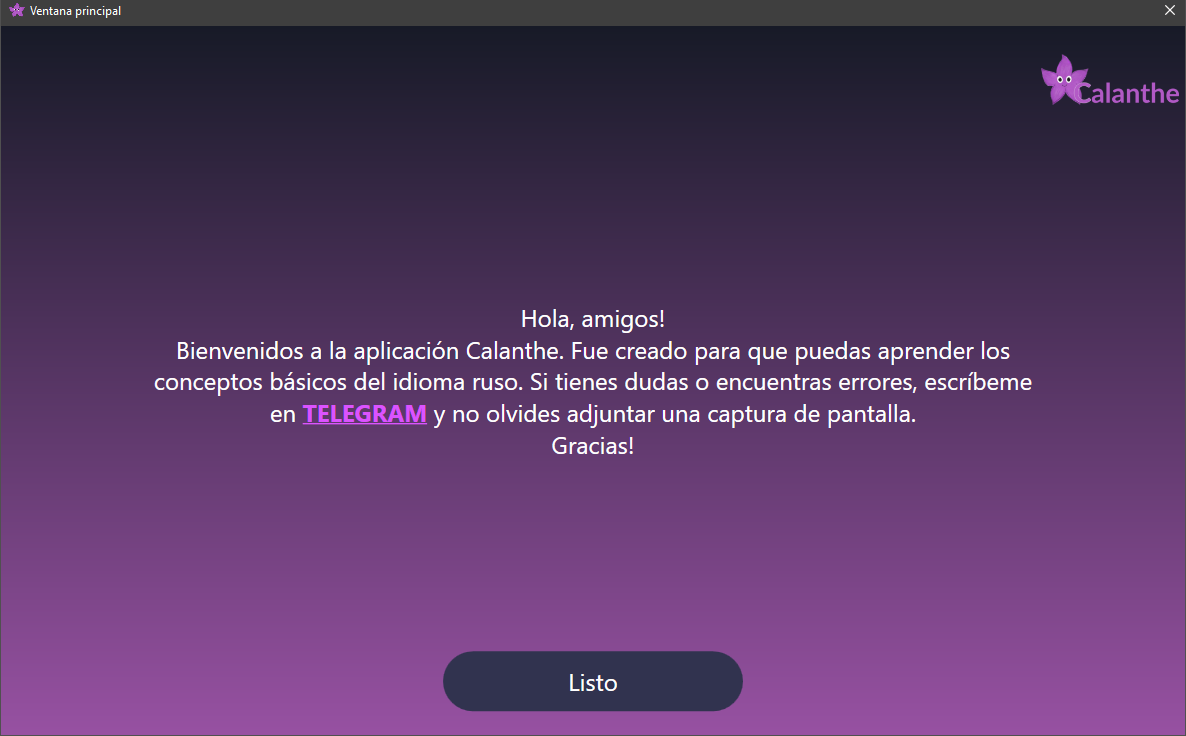


Рисунок 4.6 – Окно приветствия

Нажав на кнопку с названием «Listo», мы переходим в окно авторизации, которое изображено на рисунке 4.7. Чтобы войти в аккаунт нам необходимо ввести электронную почту и пароль. Если пользователь забыл свой пароль, то он может восстановить его, введя электронную почту и нажав на «Recuperar contrasena». Также пользователь может вернуться в предыдущее окно или перейти в окно регистрации.

Окно регистрации изображено на рисунке 4.8. Для регистрации необходимо придумать логин, пароль и ввести электронную почту. Пользователь имеет право вернуться в окно авторизации, а также при необходимости скрыть свой пароль.

На рисунке 4.9 представлено окно меню, которое встречает пользователя, когда он входит в систему. Меню состоит из трёх кнопок:

1. кнопка для перехода в окно профиля;
2. кнопка для перехода в окно с занятиями;
3. кнопка для перехода в окно со словарём.

А так же можно вернуться назад.

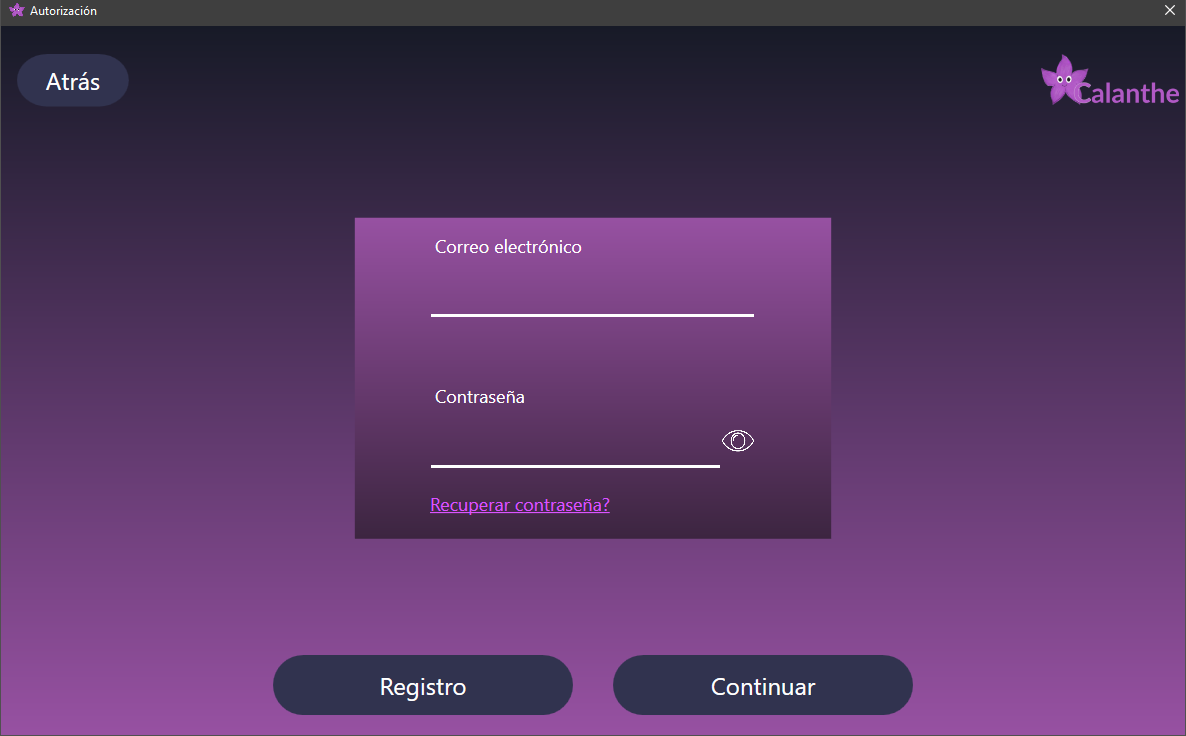
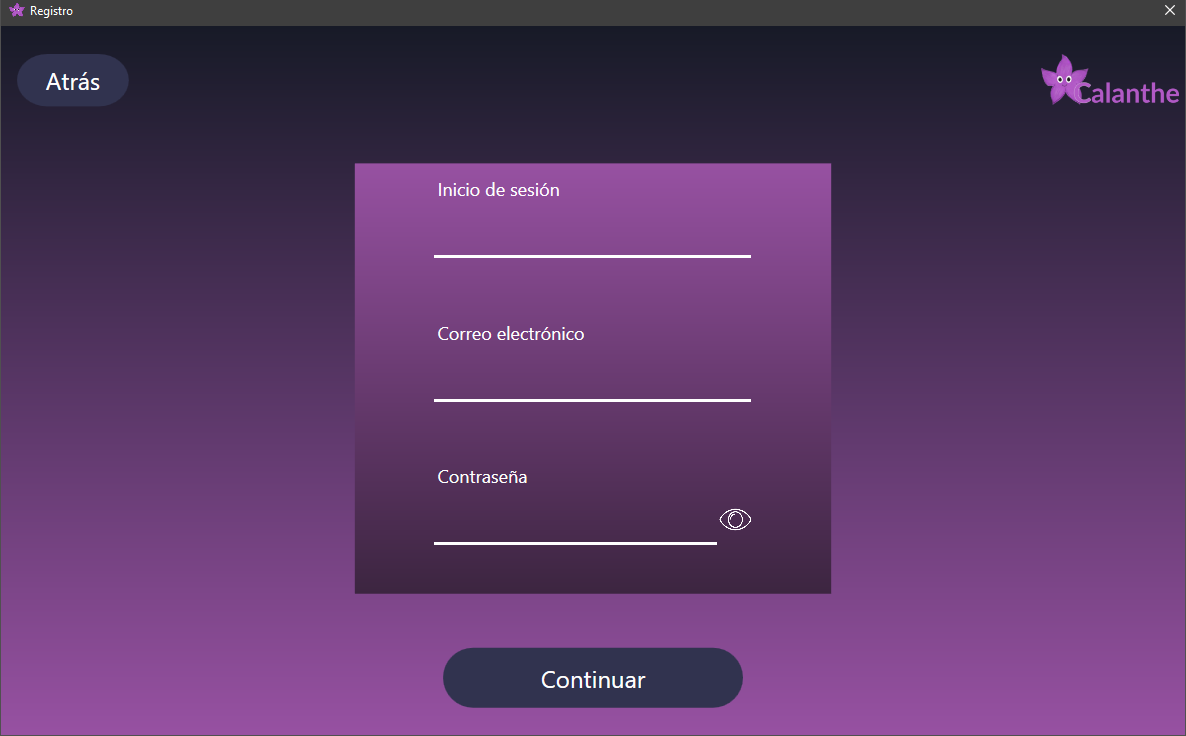


Рисунок 4.7 – Окно авторизации

Рисунок 4.8 – Окно регистрации

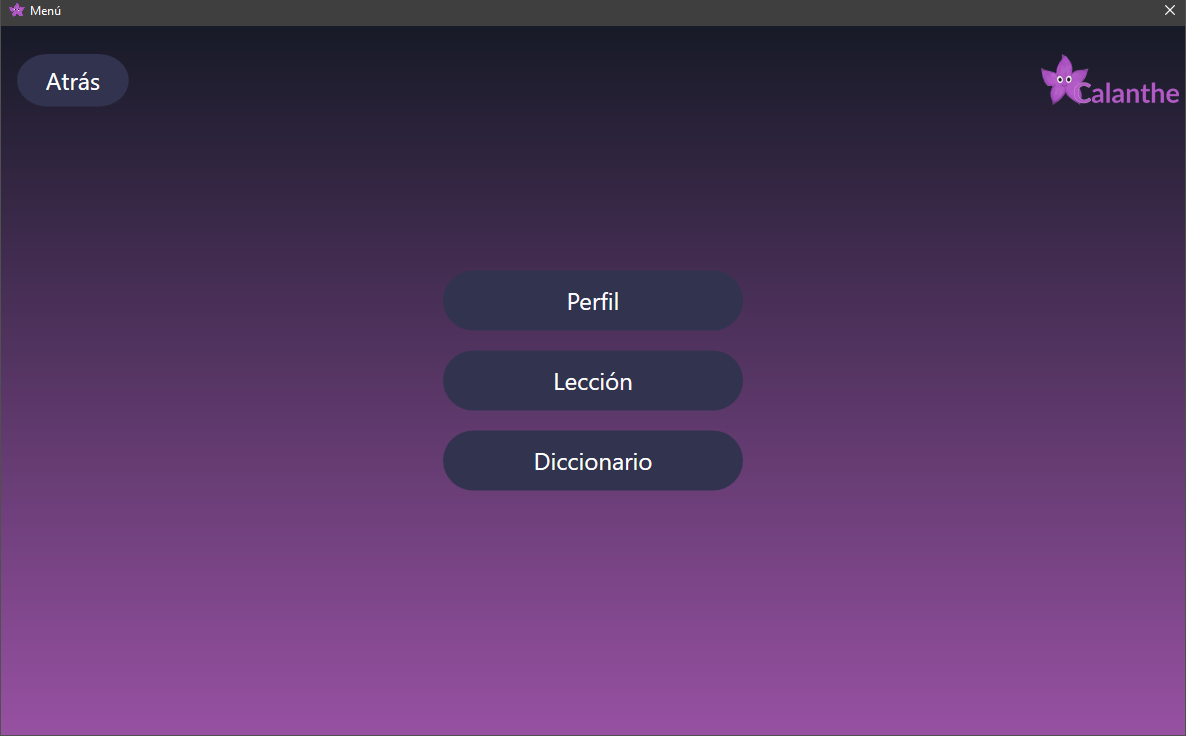


Рисунок 4.9 – Окно меню

На рисунке 4.10 отображено окно профиля обучающегося. На нём можно увидеть его фотографию, статистику прохождения занятий, кнопку возврата и кнопку для перехода в окно редактирования профиля.

Статистика формируется следующим образом: если теоретическое занятие было пройдено и тест был выполнен без ошибок, то один блок будет загораться более ярким цветов, в случае, если практическое или теоретическое занятие не будет пройдено, то блок загораться не будет.

В окне редактирования профиля обучающегося, которое представлено на рисунке 4.11, открывается следующий функционал: возможность изменять логин, изменять и скрывать пароль, менять или удалять фото профиля. Также доступна кнопка для возврата и кнопка выхода из аккаунта.

В программе доступно 16 занятий. Каждое занятие подразделяется на практическое и теоретическое. Теоретическое занятие сохраняется на компьютер в виде pdf - файла, а практическое занятие создано в форме теста. Их список показан на рисунке 4.12.



Рисунок 4.10 – Окно профиля



Рисунок 4.11 – Окно редактирования профиля

Если нажать на занятие, то откроется меню, которое показано на рисунке 4.13. Система предложит выбрать тип занятия. А так же, в данном окне расположена кнопка для перехода в окно со словарём и возврата.

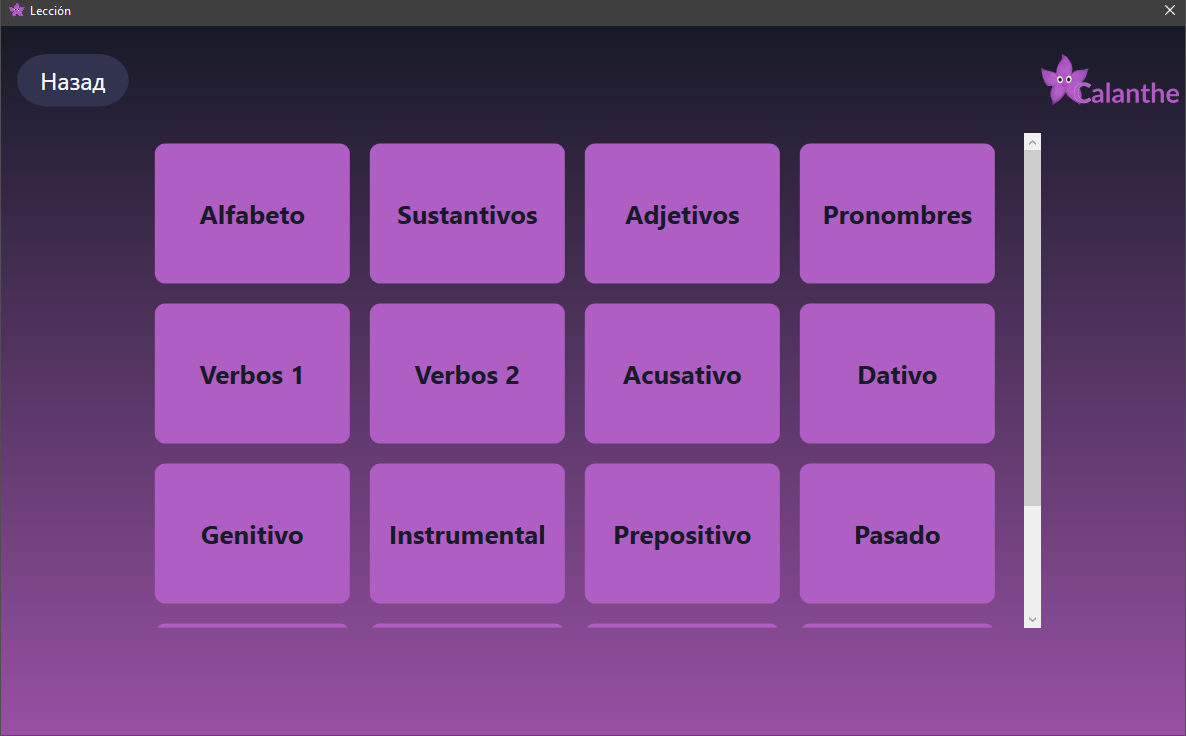


Рисунок 4.12 – Окно списка занятий

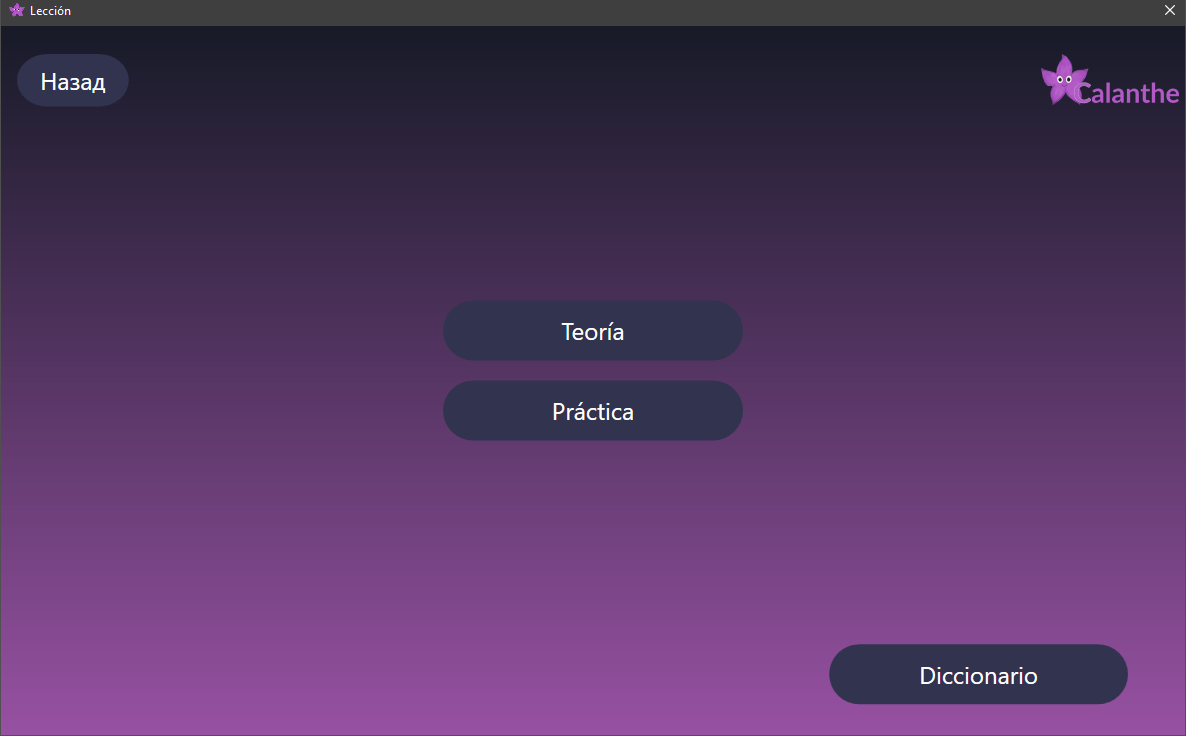


Рисунок 4.13 – Окно меню для выбора типа занятия

Выбрав практическое занятие, открывается тест, как показано на рисунке 4.14, пройдя который, система зафиксирует результат, если тест был решён правильно.

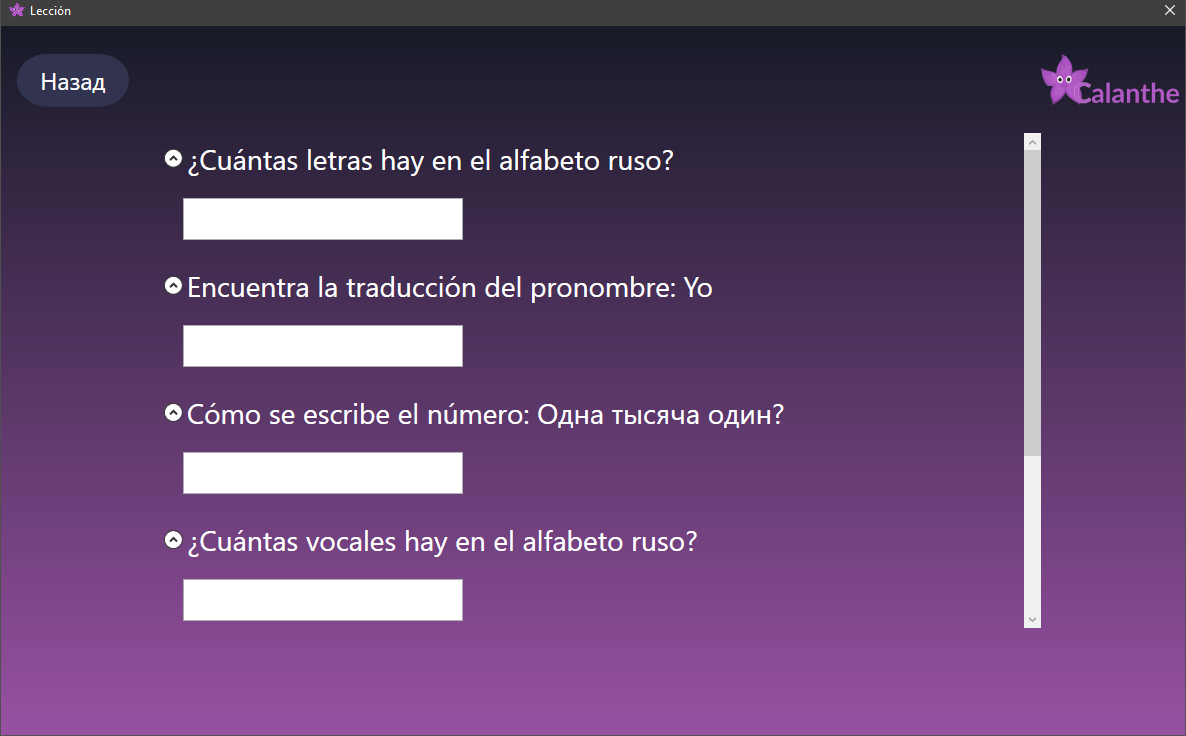


Рисунок 4.14 – Практическое занятие

Для того, чтобы воспользоваться теоретическим занятием, нужно выбрать папку, куда сохранится файл. Изображение окна представлено на рисунке 4.15. После сохранения, файл автоматически откроется в формате .pdf.

Внешний вид теоретического занятия показан на рисунке 4.16.

Окно с выводом словаря показано на рисунке 4.17. Система предоставляет пользователю следующие возможности: поиск необходимого слова, добавление, удаление, редактирования слов, а так же в данном окне расположены кнопки для возврата к меню и перехода в окно с занятиями.

Реализация окон для добавления, удаления и редактирования слов представлена на рисунках 4.18 – 4.20.

Для того, чтобы добавить слово, необходимо ввести само слово и его перевод. При удалении слова, Вы перейдёте в окно, в котором система отобразит его, чтобы пользователь убедился в правильности выбора. При редактировании, система так же отобразит слово и перевод, чтобы обучающему было легче изменить его.

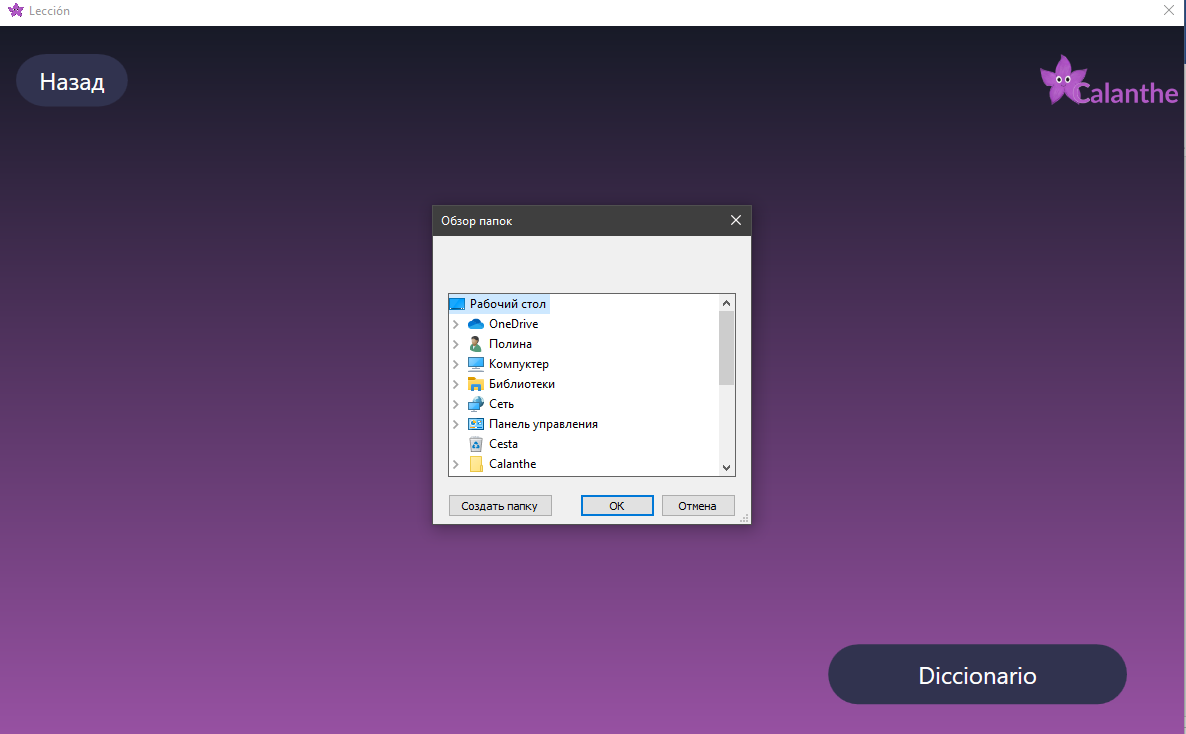


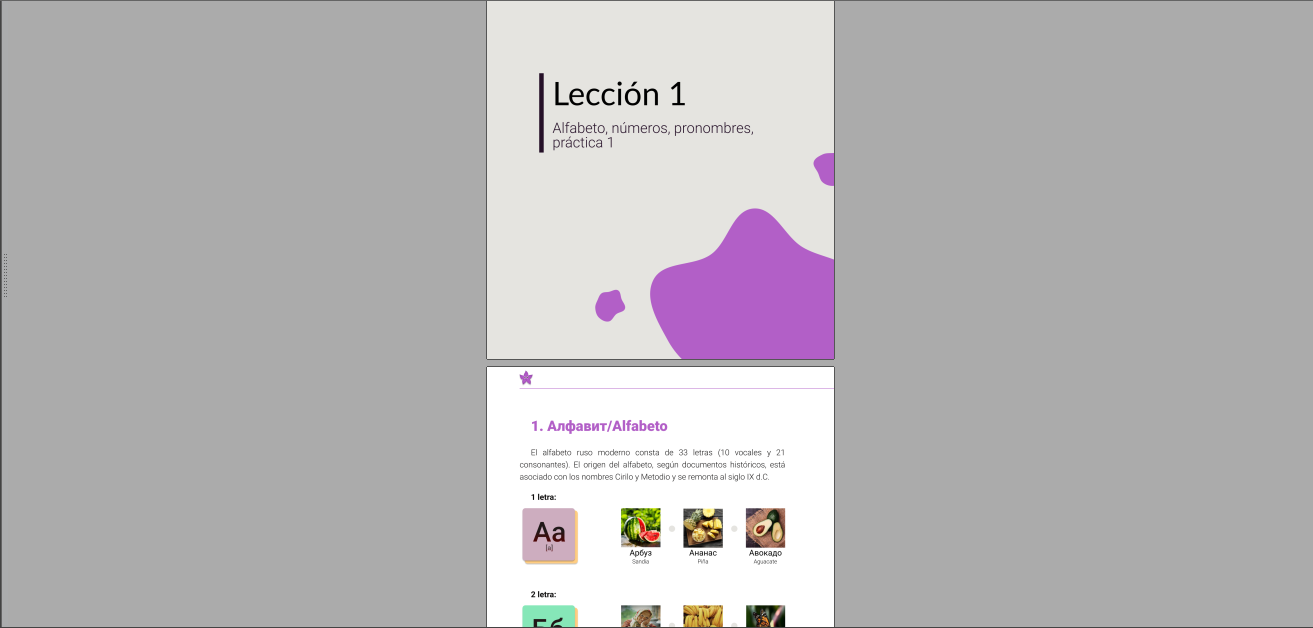
Рисунок 4.15 – Сохранение теоретического занятия

Рисунок 4.16 – Внешний вид занятия

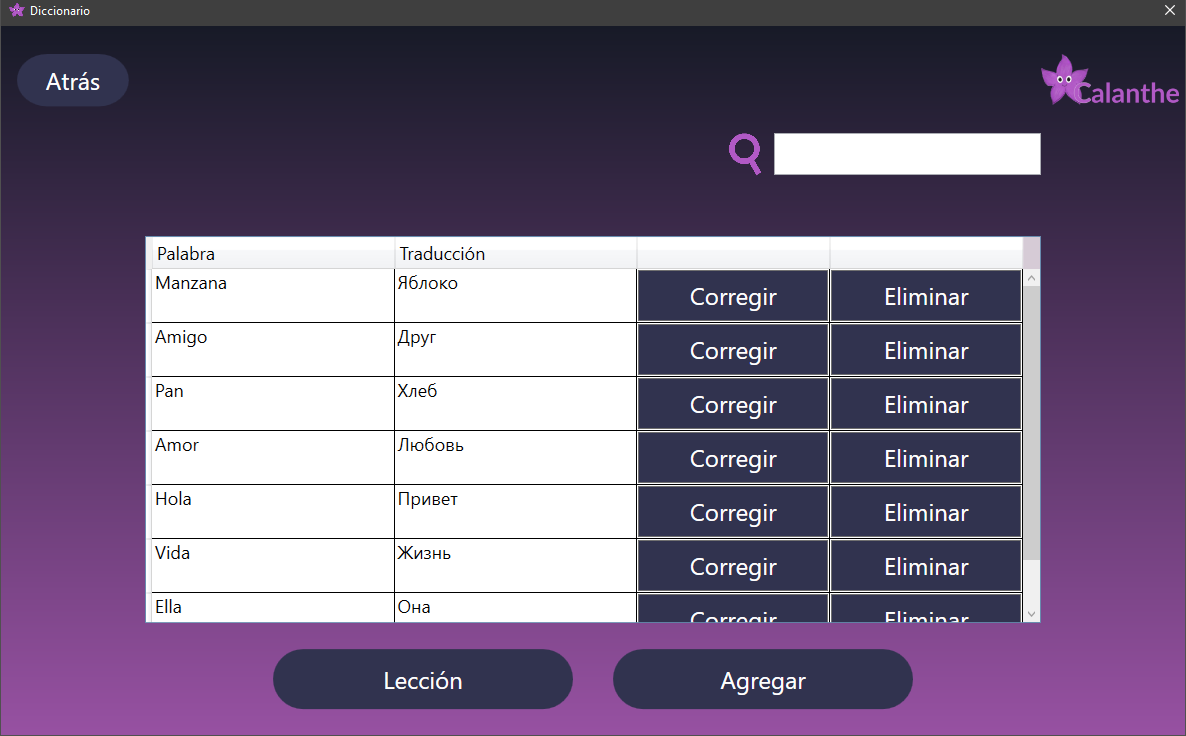


Рисунок 4.17 – Окно со словарём

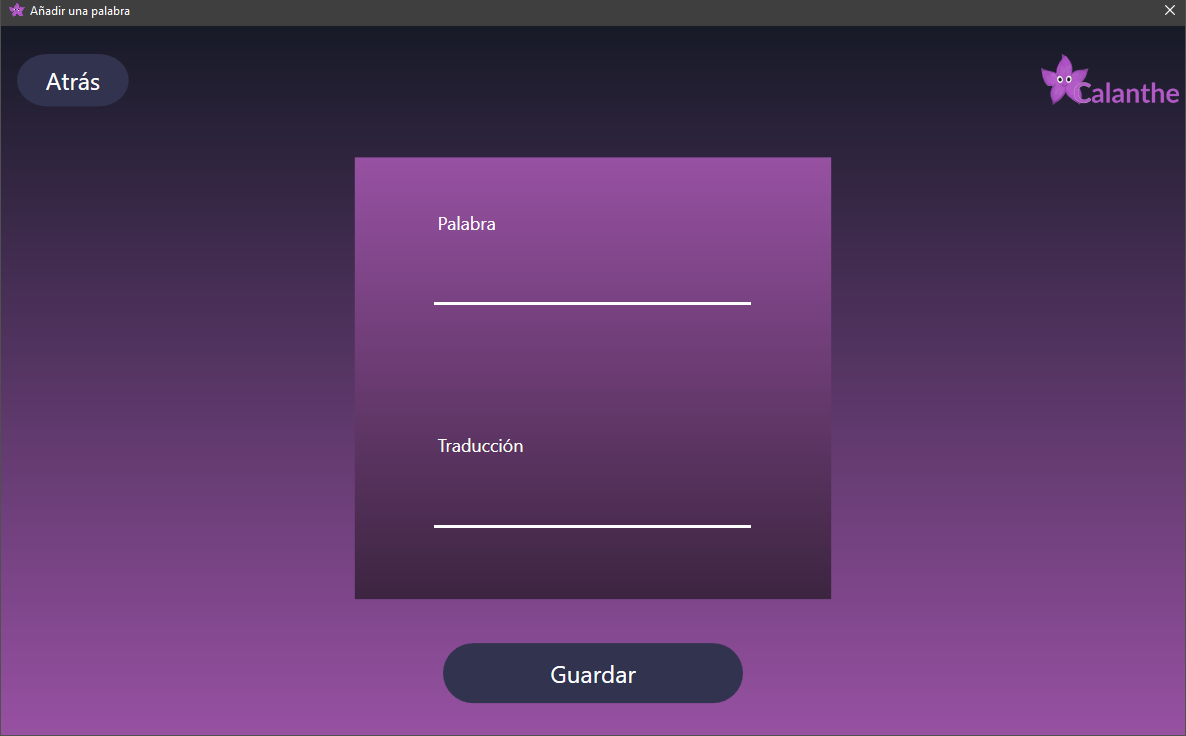


Рисунок 4.18 – Окно для добавления слова

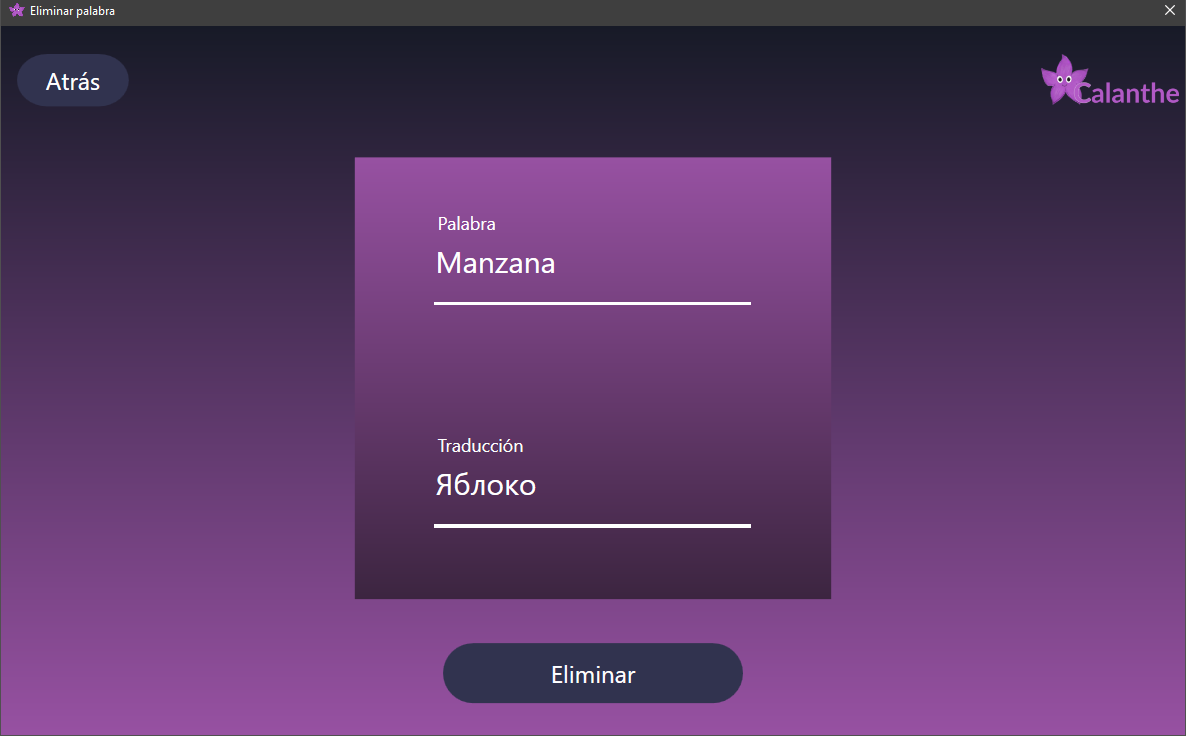


Рисунок 4.19 – Окно для удаления слова

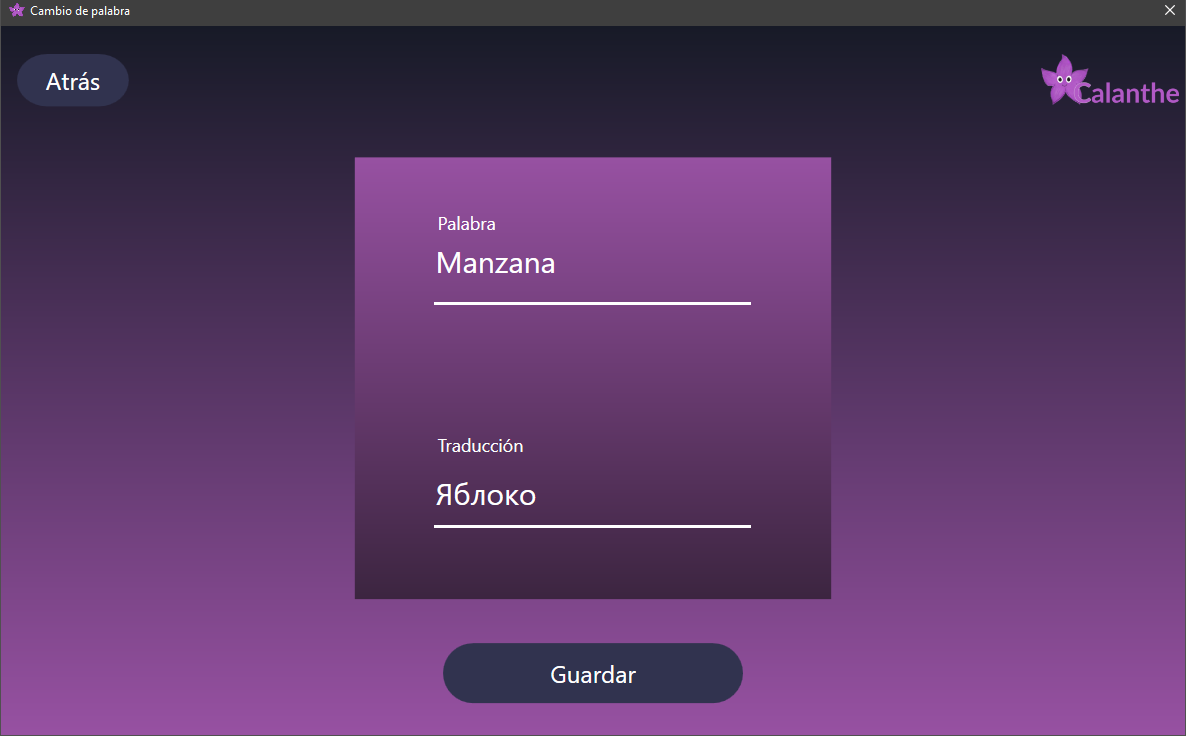


Рисунок 4.20 – Окно для редактирования слова

# ЗАКЛЮЧЕНИЕ

В результате дипломной работы было разработано десктоп – приложение «Calanthe». Данная ИС обладает всеми функциями, описанными в главе 2, пункте 2.2.

А также были выполнены следующие задачи:

1. выявлены пользовательские требования к ИС;
2. выявлены функциональные требования к ИС;
3. установлены границы системы; роли, взаимодействующие с системой; варианты использования системы и отношения между ролями и прецедентами;
4. описано поведение объектов системы и то, как это поведение изменяется в зависимости от внутренних и внешних событий; определено, как поток управления переходит от одной деятельности к другой;
5. определен набор классов, реализующих поведение, описанное в прецедентах;
6. выполнено проектирование базы данных;
7. выполнена физическая реализация базы данных с использованием СУБД Microsoft SQL Server;
8. выполнена программная реализация информационной системы с использованием технологии WPF;
9. разработаны следующие руководства: руководство пользователя, руководство программиста, руководство по установке.

Подводя итоги, можно сказать, что все поставленные задачи решены, а цель достигнута.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1..Новые возможности C# 9.0 // Microsoft URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/whats-.. (дата обращения: 12.05.2022).

2..Основа WPF // Professor web URL: https://professorweb.ru/my/WPF/base\_WPF/level1/base\_W.. (дата обращения: 10.04.2022).

3..Полное руководство по WPF // Учебник по WPF URL: https://www.wpf-tutorial.com/ (дата обращения: 23.04.2022).

4. Стасышин, В. М. Базы данных: технологии доступа : учеб. пособие для СПО / В. М. Стасышин, Т. Л. Стасышина. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2018. — 164 с.

5. Руководство по классическим приложениям (WPF .NET) // Microsoft URL: https://docs.microsoft.com/ru-ru/dotnet/desktop/wpf/o.. (дата обращения: 11.05.2022).

6. Что нового в Visual Studio 2022? // Cpab URL: https://cpab.ru/chto-novogo-v-visual-studio-2022-clou.. (дата обращения: 18.04.2022).

7. Язык программирования C#: краткая история, возможности и перспективы // timeweb>cloud URL: https://timeweb.com/ru/community/articles/chto-takoe-.. (дата обращения: 05.03.2022).

8. Entity Framework: как быстрее написать код для работы с базой данных // Skillbox URL: https://skillbox.ru/media/code/entity\_framework/ (дата обращения: 01.05.2022).

9. Кудрина, Е. В. Основы алгоритмизации и программирования на языке c# : учеб. пособие для СПО / Е. В. Кудрина, М. В. Огнева. — М. : Издательство Юрайт, 2019. — 322 с.

10..Microsoft Visual Studio // Microsoft URL: https://visualstudio.microsoft.com/ru/ (дата обращения: 12.02.2022).

11..Visual..Studio..2022…//…ХАБР…URL:..https://habr.com/ru/company /microsoft/blog/553442/ (дата обращения: 25.01.2022).

12. Сайт о программировании [Электронный ресурс] – Режим доступа: https://metanit.com/, свободный (дата обращения: 20.02.2022).

13. Стружкин, Н. П.  Базы данных: проектирование: учебник для среднего профессионального образования / Н. П. Стружкин, В. В. Годин. — Москва: Издательство Юрайт, 2019. — 477 с. — (Профессиональное образование). — ISBN 978-5-534-11635-9. — Текст: электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/445776 (дата обращения: 20.02.2022). Режим доступа: для авториз. пользователей.

14. Ашарина, И.В. Основы программирования на языках С и С++: Курс лекций для высших учебных заведений / И.В. Ашарина. — М.: Гор. линия-Телеком, 2018. — 208 c.

15. Гниденко, И. Г. Технология разработки программного обеспечения : учеб. пособие для СПО / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — М. : Издательство Юрайт, 2017. — 235 с.

16. Трофимов, В. В. Основы алгоритмизации и программирования : учебник для СПО / В. В. Трофимов, Т. А. Павловская ; под ред. В. В. Трофимова. — М. : Издательство Юрайт, 2019. — 137 с.

17. Маркин, А. В. Программирование на sql в 2 ч. Часть 2 : учебник и практикум для бакалавриата и магистратуры / А. В. Маркин. — М. : Издательство Юрайт, 2019. — 292 с.

18. Зыков, С. В. Программирование. Объектно-ориентированный подход : учебник и практикум для академического бакалавриата / С. В. Зыков. — М. : Издательство Юрайт, 2019. — 155 с.

19. Кауфман, В.Ш. Языки программирования. Концепции и принципы / В.Ш. Кауфман. — М.: ДМК, 2017. — 464 c.

20. Страуструп, Б. Язык программирования C++: Специальное издание / Б. Страуструп; Пер. с англ. Н.Н. Мартынов. — М.: БИНОМ, 2017. — 1136 c.

# ПРИЛОЖЕНИЕ А

**Список сокращений**

БД - база данных

ГОСТ - государственный стандарт

ИС - информационная система

ПО - программное обеспечение

СУБД - система управления базами данных

ЯП – язык программирования

C# - C Sharp

MVVM - model-View-ViewModel

SQL - structured Query Language – Язык структурированных запросов

VS - visual studio – среда разработки

WPF – windows Presentation Foundation

# ПРИЛОЖЕНИЕ Б

**Словарь данных**

****

Рисунок Б.1 – Словарь данных

# ПРИЛОЖЕНИЕ В

**Контрольный пример базы данных**

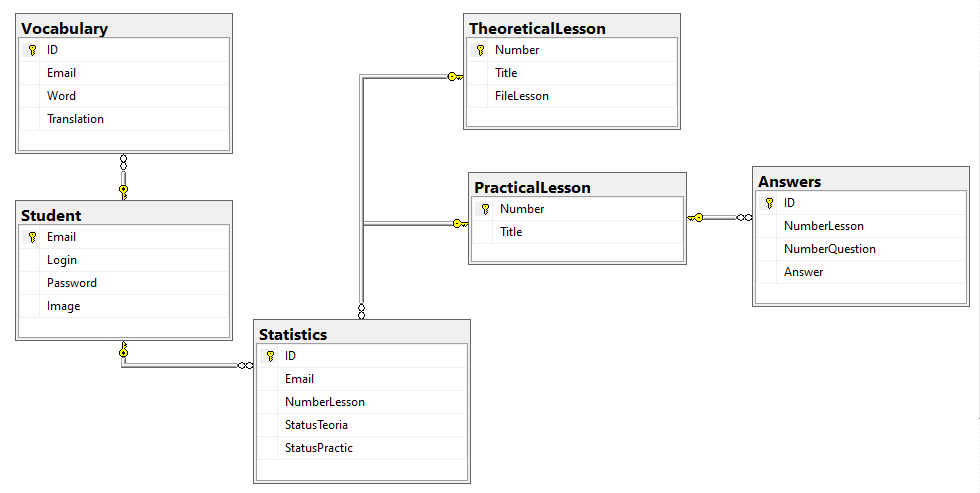
****

Рисунок В.1 – Схема базы данных

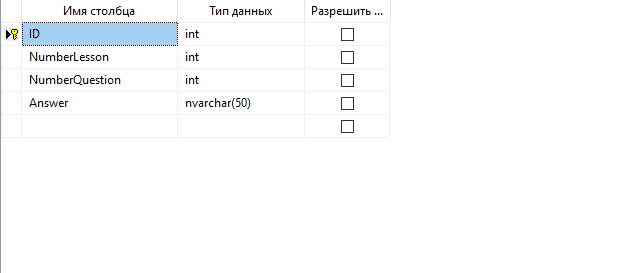
****

Рисунок В.2 – Дизайнер таблицы «Answers»

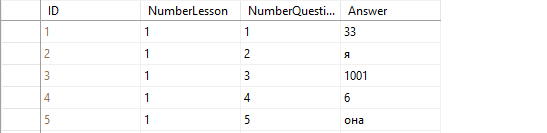
****

Рисунок В.3 – Содержание таблицы «Answers»

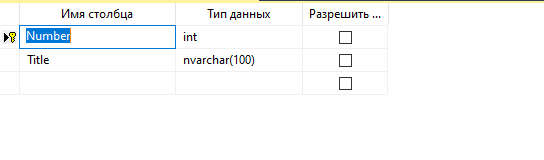
****

Рисунок В.4 – Дизайнер таблицы «PracticalLesson»

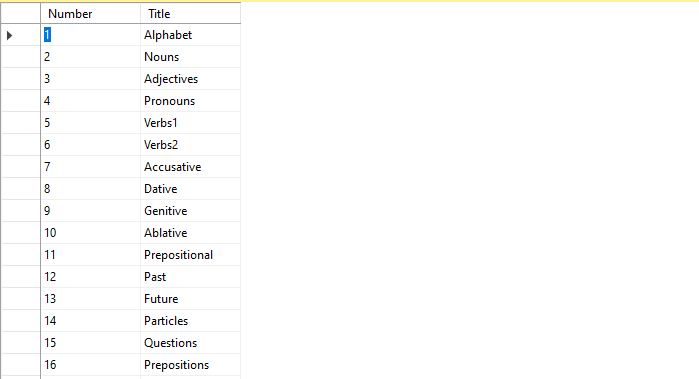
****

Рисунок В.5 – Содержание таблицы «PracticalLesson»

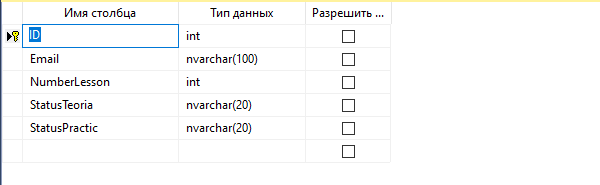
****

Рисунок В.6 – Дизайнер таблицы «Statistics»

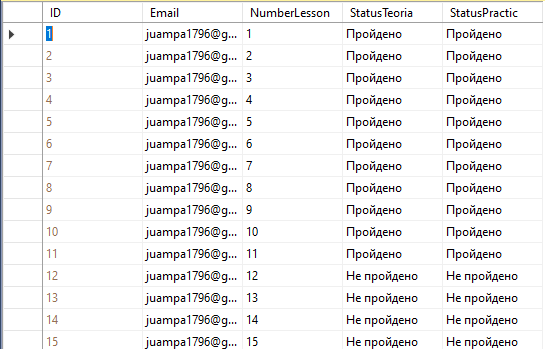
****

Рисунок В.7 – Содержание таблицы «Statistics»

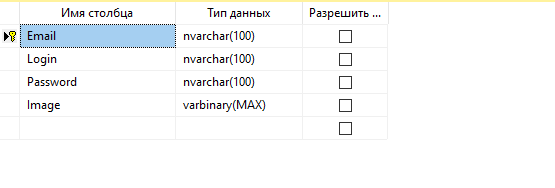
****

Рисунок В.8 – Дизайнер таблицы «Student»

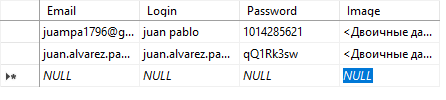
****

Рисунок В.9 – Содержание таблицы «Student»

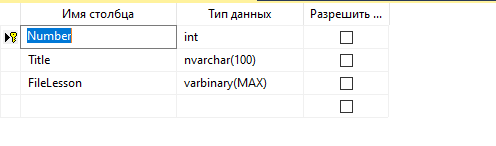
****

Рисунок В.10 – Дизайнер таблицы «TheoreticalLesson»

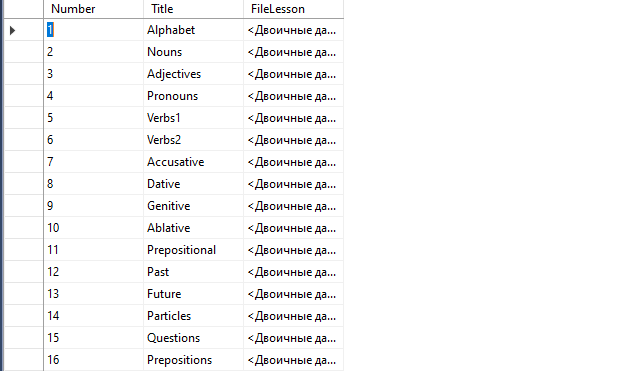
****

Рисунок В.11 – Содержание таблицы «TheoreticalLesson»

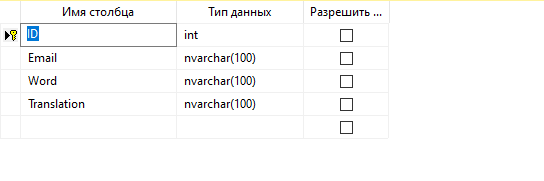
****

Рисунок В.12 – Дизайнер таблицы «Vocabulary»



Рисунок В.13 – Содержание таблицы «Vocabulary»

# ПРИЛОЖЕНИЕ Г

**Листинг программы**

Окно «Autorization»:

namespace Calanthe

{

public partial class Autorization : Window

{

Student \_user = new Student();

CalantheEntities db = new CalantheEntities();

public Autorization()

{

InitializeComponent();

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

MainWindow \_win = new MainWindow();

this.Close();

\_win.Show();

}

private void registration\_b\_Click\_1(object sender, RoutedEventArgs e)

{

Registration \_win = new Registration();

this.Close();

\_win.Show();

}

private void Continue\_b\_Click\_1(object sender, RoutedEventArgs e)

{

int n = 0;

if (Mail.Text == "" || Password.Password.ToString() == "") MessageBox.Show("Introduzca los datos!");

else

{

foreach (var user in db.Student)

{

if (Mail.Text == user.Email && Password.Password.ToString() == user.Password)

{

n = 1;

break;

}

}

if (n == 1)

{

Menu \_win = new Menu(Mail.Text);

this.Close();

\_win.Show();

MessageBox.Show("Bienvenido!");

}

else MessageBox.Show("Este usuario no existe!");

}

}

private void Hyperlink\_Click(object sender, RoutedEventArgs e)

{

try

{

int n = 0;

if (Mail.Text == "") MessageBox.Show("Ingrese su correo electrónico y luego haga clic aquí de nuevo!");

else

{

foreach (var user in db.Student)

{

if (Mail.Text == user.Email)

{

\_user = user;

n = 1;

break;

}

}

if (n == 1)

{

var chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";

var stringChars = new char[8];

var random = new Random();

string finalString = "";

for (int i = 0; i < stringChars.Length; i++)

{

stringChars[i] = chars[random.Next(chars.Length)];

finalString += stringChars[i];

}

MailAddress from = new MailAddress("Polina\_alekseevna\_valova@mail.ru", "Calanthe - Ruso Idioma");

MailAddress to = new MailAddress(Mail.Text);

MailMessage m = new MailMessage(from, to);

m.Subject = "Recuperación de contraseña";

m.Body = "Su nueva contraseña:" + finalString;

m.IsBodyHtml = true;

SmtpClient smtp = new SmtpClient("smtp.mail.ru", 25);

smtp.Credentials = new NetworkCredential("Polina\_alekseevna\_valova@mail.ru", "D3vTLNZE6CBLNB4dKAEG");

smtp.EnableSsl = true;

smtp.Send(m);

MessageBox.Show("Contraseña enviada a Su correo electrónico! Después de recibirlo, inicie sesión con él!");

\_user.Password = finalString;

db.SaveChanges();

}

else MessageBox.Show("El usuario con dicho correo electrónico no existe!");

}

}

catch (Exception ex)

{

MessageBox.Show("No hay conexión a Internet o este correo electrónico no existe!");

}

}

private void Password\_PasswordChanged(object sender, RoutedEventArgs e)

{

Password\_tb.Text = Password.Password;

}

private void Password\_tb\_TextChanged(object sender, TextChangedEventArgs e)

{

Password.Password = Password\_tb.Text;

}

private void eye\_Click(object sender, RoutedEventArgs e)

{

if (Password.Visibility == Visibility.Visible)

{

Password.Visibility = Visibility.Collapsed;

Password\_tb.Visibility = Visibility.Visible;

Password\_tb.Text = Password.Password;

}

else

{

Password.Visibility = Visibility.Visible;

Password\_tb.Visibility = Visibility.Collapsed;

Password.Password = Password\_tb.Text; }}}}

Окно «AddWords»:

namespace Calanthe

{

public partial class AddWords : Window

{

string mail;

private CalantheEntities \_context = new CalantheEntities();

int id;

string Email;

public AddWords(string mail)

{

InitializeComponent();

this.mail = mail;

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Dictionary \_win = new Dictionary(mail);

\_win.Show();

this.Close();

}

private void Save\_b\_Click(object sender, RoutedEventArgs e)

{

Vocabulary \_vocabulary = new Vocabulary();

if (Word.Text == "" || Translite.Text == "") MessageBox.Show("Introduzca los datos!");

else

{

\_vocabulary.Email = mail;

\_vocabulary.Word = Word.Text.Trim();

\_vocabulary.Translation = Translite.Text.Trim();

\_context.Vocabulary.Add(\_vocabulary);

\_context.SaveChanges();

Dictionary \_win = new Dictionary(mail);

\_win.Show();

this.Close();

MessageBox.Show("Guardado!"); }}}}

Окно «DelWords»:

namespace Calanthe

{

public partial class DelWords : Window

{

string mail;

private CalantheEntities \_context;

private Vocabulary \_vocabulary;

private Dictionary dictionary;

int id;

public DelWords(string mail, CalantheEntities context, object o, Dictionary \_dictionary)

{

InitializeComponent();

this.mail = mail;

\_vocabulary = (o as Button).DataContext as Vocabulary;

\_context = context;

dictionary = \_dictionary;

Word.Text = \_vocabulary.Word;

Translite.Text = \_vocabulary.Translation;

id = \_vocabulary.ID;

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Dictionary \_win = new Dictionary(mail);

this.Close();

\_win.Show();

}

private void Del\_b\_Click\_1(object sender, RoutedEventArgs e)

{

foreach (var user in \_context.Vocabulary)

{

if (id == user.ID)

{

\_vocabulary = user;

\_context.Vocabulary.Remove(\_vocabulary);

break;

}

}

\_context.SaveChanges();

MessageBox.Show("Guardado!");

Dictionary \_win = new Dictionary(mail);

\_win.Show();

this.Close(); }}}

Окно «Dictionary»:

namespace Calanthe

{

public partial class Dictionary : Window

{

string mail;

CalantheEntities \_context = new CalantheEntities();

public Dictionary(string mail)

{

InitializeComponent();

this.mail = mail;

dbWords.ItemsSource = \_context.Vocabulary.ToList();

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Menu \_win = new Menu(mail);

this.Close();

\_win.Show();

}

private void Add\_b\_Click(object sender, RoutedEventArgs e)

{

AddWords \_win = new AddWords(mail);

this.Close();

\_win.Show();

}

private void Back\_lessons\_Click(object sender, RoutedEventArgs e)

{

Lessons \_win = new Lessons(mail);

this.Close();

\_win.Show();

}

private void edit\_b\_Click\_1(object sender, RoutedEventArgs e)

{

EditWords \_win = new EditWords(mail, \_context, sender, this);

\_win.Show();

this.Close();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

var current\_item = CalantheEntities.GetContext().Vocabulary.ToList();

current\_item = current\_item.Where(p => p.Email.ToString().ToLower().Contains(mail.ToLower())).ToList();

CalantheEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload());

dbWords.ItemsSource = current\_item;

}

private void del\_b\_Click(object sender, RoutedEventArgs e)

{

DelWords \_win = new DelWords(mail, \_context, sender, this);

\_win.Show();

this.Close();

}

private void tbResearch\_TextChanged(object sender, TextChangedEventArgs e)

{

if (tbResearch.Text == string.Empty)

{

var current\_item1 = CalantheEntities.GetContext().Vocabulary.ToList();

current\_item1 = current\_item1.Where(p => p.Email.ToString().ToLower().Contains(mail.ToLower())).ToList();

dbWords.ItemsSource = current\_item1;

}

else

{

var current\_item = CalantheEntities.GetContext().Vocabulary.ToList();

current\_item = current\_item.Where(p => p.Word.ToString().ToLower().Contains(tbResearch.Text.ToLower())).ToList();

dbWords.ItemsSource = current\_item;}}}}

Окно «EditWords»:

namespace Calanthe

{

public partial class EditWords : Window

{

string mail;

private CalantheEntities \_context;

private Vocabulary \_vocabulary;

private Dictionary dictionary;

int id;

public EditWords(string mail, CalantheEntities context, object o, Dictionary \_dictionary)

{

InitializeComponent();

this.mail = mail;

\_vocabulary = (o as Button).DataContext as Vocabulary;

\_context = context;

dictionary = \_dictionary;

Word.Text = \_vocabulary.Word;

Translite.Text = \_vocabulary.Translation;

id = \_vocabulary.ID;

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Dictionary \_win = new Dictionary(mail);

this.Close();

\_win.Show();

}

private void Save\_b\_Click(object sender, RoutedEventArgs e)

{

foreach (var user in \_context.Vocabulary)

{

if (id == user.ID)

{

\_vocabulary = user;

\_vocabulary.Word = Word.Text;

\_vocabulary.Translation = Translite.Text;

break;

}

}

\_context.SaveChanges();

MessageBox.Show("Guardado!");

Dictionary \_win = new Dictionary(mail);

\_win.Show();

this.Close();}}}

Окно «Lessons»:

namespace Calanthe

{

public partial class Lessons : Window

{

string mail;

CalantheEntities db = new CalantheEntities();

TheoreticalLesson theoreticalLesson = new TheoreticalLesson();

Statistics statistics = new Statistics();

Answers answers = new Answers();

string filename = null;

string msg;

string Answer;

String count = "";

public Lessons(string mail)

{

InitializeComponent();

this.mail = mail;

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Menu \_win = new Menu(mail);

this.Close();

\_win.Show();

}

private void Dictionary\_b\_Click(object sender, RoutedEventArgs e)

{

Dictionary \_win = new Dictionary(mail);

this.Close();

\_win.Show();

}

private void ViewLesson(object sender, RoutedEventArgs e)

{

msg = ((Button)sender).Name;

PanelMenu.Visibility = Visibility.Visible;

Dictionary\_b.Visibility = Visibility.Visible;

Dictionary\_b.Visibility = Visibility.Visible;

Back\_b2.Visibility = Visibility.Visible;

Back\_b.Visibility = Visibility.Hidden;

PanelLessons.Visibility = Visibility.Hidden;

}

private void Back\_b2\_Click(object sender, RoutedEventArgs e)

{

PanelMenu.Visibility = Visibility.Hidden;

Dictionary\_b.Visibility = Visibility.Hidden;

Dictionary\_b.Visibility = Visibility.Hidden;

Back\_b2.Visibility = Visibility.Hidden;

Back\_b.Visibility = Visibility.Visible;

PanelLessons.Visibility = Visibility.Visible;

}

private void Back\_b3\_Click(object sender, RoutedEventArgs e)

{

PeacticAlphabet.Visibility = Visibility.Hidden;

PeacticNouns.Visibility = Visibility.Hidden;

PeacticAdjectives.Visibility = Visibility.Hidden;

PeacticPronouns.Visibility = Visibility.Hidden;

PeacticVerbs1.Visibility = Visibility.Hidden;

PeacticVerbs2.Visibility = Visibility.Hidden;

PeacticAccusative.Visibility = Visibility.Hidden;

PeacticDative.Visibility = Visibility.Hidden;

PeacticGenitive.Visibility = Visibility.Hidden;

PeacticAblative.Visibility = Visibility.Hidden;

PeacticPrepositional.Visibility = Visibility.Hidden;

PeacticPast.Visibility = Visibility.Hidden;

PeacticFuture.Visibility = Visibility.Hidden;

PeacticParticles.Visibility = Visibility.Hidden;

PeacticQuestions.Visibility = Visibility.Hidden;

PeacticPrepositions.Visibility = Visibility.Hidden;

Back\_b3.Visibility = Visibility.Hidden;

Back\_b2.Visibility = Visibility.Visible;

PanelMenu.Visibility = Visibility.Visible;

Dictionary\_b.Visibility = Visibility.Visible;

}

private void Teoria\_b\_Click(object sender, RoutedEventArgs e)

{

int n = 0;

Statistics statistics = new Statistics();

try

{

foreach (var item in db.TheoreticalLesson)

{

if (msg == item.Title)

{

theoreticalLesson = item;

MessageBox.Show("Por favor, seleccione la carpeta para guardar el archivo!");

using (var dialog = new System.Windows.Forms.FolderBrowserDialog())

{

System.Windows.Forms.DialogResult result = dialog.ShowDialog();

filename = dialog.SelectedPath + @"\" + msg + ".pdf";

}

File.WriteAllBytes(filename, theoreticalLesson.FileLesson);

var process = Process.Start(filename);

foreach (var i in db.Statistics)

{

if (item.Number == i.NumberLesson && i.Email == mail)

{

i.StatusTeoria = "Пройдено";

break;

}

}

}

}

db.SaveChanges();

}

catch (Exception)

{

}

}

private void Practic\_b\_Click(object sender, RoutedEventArgs e)

{

Back\_b3.Visibility = Visibility.Visible;

Back\_b.Visibility = Visibility.Hidden;

PanelLessons.Visibility = Visibility.Hidden;

Dictionary\_b.Visibility = Visibility.Hidden;

PanelMenu.Visibility = Visibility.Hidden;

MessageBox.Show("Por favor escriba todas las respuestas en letra minúscula!");

switch (msg)

{

case "Alphabet":

PeacticAlphabet.Visibility = Visibility.Visible;

var statAlphabet = db.Answers.Where(p => p.NumberLesson == 1);

foreach (Answers p in statAlphabet)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Nouns":

PeacticNouns.Visibility = Visibility.Visible;

var statNouns = db.Answers.Where(p => p.NumberLesson == 2);

foreach (Answers p in statNouns)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Adjectives":

PeacticAdjectives.Visibility = Visibility.Visible;

var statAdjectives = db.Answers.Where(p => p.NumberLesson == 3);

foreach (Answers p in statAdjectives)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Pronouns":

PeacticPronouns.Visibility = Visibility.Visible;

var statPronouns = db.Answers.Where(p => p.NumberLesson == 4);

foreach (Answers p in statPronouns)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Verbs1":

PeacticVerbs1.Visibility = Visibility.Visible;

var statVerbs1 = db.Answers.Where(p => p.NumberLesson == 5);

foreach (Answers p in statVerbs1)

{

for (int i = 0; i < 4; i++)

{

count += p.Answer;

break;

}

}

break;

case "Verbs2":

PeacticVerbs2.Visibility = Visibility.Visible;

var statVerbs2 = db.Answers.Where(p => p.NumberLesson == 6);

foreach (Answers p in statVerbs2)

{

for (int i = 0; i < 4; i++)

{

count += p.Answer;

break;

}

}

MessageBox.Show(count);

break;

case "Accusative":

PeacticAccusative.Visibility = Visibility.Visible;

var statAccusative = db.Answers.Where(p => p.NumberLesson == 7);

foreach (Answers p in statAccusative)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Dative":

PeacticDative.Visibility = Visibility.Visible;

var statDative = db.Answers.Where(p => p.NumberLesson == 8);

foreach (Answers p in statDative)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Genitive":

PeacticGenitive.Visibility = Visibility.Visible;

var statGenitive = db.Answers.Where(p => p.NumberLesson == 9);

foreach (Answers p in statGenitive)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Ablative":

PeacticAblative.Visibility = Visibility.Visible;

var statAblative = db.Answers.Where(p => p.NumberLesson == 10);

foreach (Answers p in statAblative)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Prepositional":

PeacticPrepositional.Visibility = Visibility.Visible;

var statPrepositional = db.Answers.Where(p => p.NumberLesson == 11);

foreach (Answers p in statPrepositional)

{

for (int i = 0; i < 4; i++)

{

count += p.Answer;

break;

}

}

break;

case "Past":

PeacticPast.Visibility = Visibility.Visible;

var statPast = db.Answers.Where(p => p.NumberLesson == 12);

foreach (Answers p in statPast)

{

for (int i = 0; i < 4; i++)

{

count += p.Answer;

break;

}

}

break;

case "Future":

PeacticFuture.Visibility = Visibility.Visible;

var statFuture = db.Answers.Where(p => p.NumberLesson == 13);

foreach (Answers p in statFuture)

{

for (int i = 0; i < 4; i++)

{

count += p.Answer;

break;

}

}

break;

case "Particles":

PeacticParticles.Visibility = Visibility.Visible;

var statParticles = db.Answers.Where(p => p.NumberLesson == 14);

foreach (Answers p in statParticles)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Questions":

PeacticQuestions.Visibility = Visibility.Visible;

var statQuestions = db.Answers.Where(p => p.NumberLesson == 15);

foreach (Answers p in statQuestions)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

case "Prepositions":

PeacticPrepositions.Visibility = Visibility.Visible;

var statPrepositions = db.Answers.Where(p => p.NumberLesson == 16);

foreach (Answers p in statPrepositions)

{

for (int i = 0; i < 5; i++)

{

count += p.Answer;

break;

}

}

break;

}

}

void ChangeStatusPractic()

{

Statistics statistics = new Statistics();

foreach (var item in db.TheoreticalLesson)

{

if (msg == item.Title)

{

foreach (var i in db.Statistics)

{

if (item.Number == i.NumberLesson && i.Email == mail)

{

i.StatusPractic = "Пройдено";

break;

}

}

}

}

db.SaveChanges();

}

private void SavePeacticAlphabet\_Click(object sender, RoutedEventArgs e)

{

if (PeacticAlphabet1.Text == "" || PeacticAlphabet2.Text == "" || PeacticAlphabet3.Text == "" || PeacticAlphabet4.Text == "" || PeacticAlphabet5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticAlphabet1.Text + PeacticAlphabet2.Text + PeacticAlphabet3.Text + PeacticAlphabet4.Text + PeacticAlphabet5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticAlphabet1.Text = string.Empty;

PeacticAlphabet2.Text = string.Empty;

PeacticAlphabet3.Text = string.Empty;

PeacticAlphabet4.Text = string.Empty;

PeacticAlphabet5.Text = string.Empty;

}

}

}

private void SavePeacticNouns\_Click(object sender, RoutedEventArgs e)

{

if (PeacticNouns1.Text == "" || PeacticNouns2.Text == "" || PeacticNouns3.Text == "" || PeacticNouns4.Text == "" || PeacticNouns5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticNouns1.Text + PeacticNouns2.Text + PeacticNouns3.Text + PeacticNouns4.Text + PeacticNouns5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticNouns1.Text = string.Empty;

PeacticNouns2.Text = string.Empty;

PeacticNouns3.Text = string.Empty;

PeacticNouns4.Text = string.Empty;

PeacticNouns5.Text = string.Empty;

}

}

}

private void SavePeacticAdjectives\_Click(object sender, RoutedEventArgs e)

{

if (PeacticAdjectives1.Text == "" || PeacticAdjectives2.Text == "" || PeacticAdjectives3.Text == "" || PeacticAdjectives4.Text == "" || PeacticAdjectives5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticAdjectives1.Text + PeacticAdjectives2.Text + PeacticAdjectives3.Text + PeacticAdjectives4.Text + PeacticAdjectives5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticAdjectives1.Text = string.Empty;

PeacticAdjectives2.Text = string.Empty;

PeacticAdjectives3.Text = string.Empty;

PeacticAdjectives4.Text = string.Empty;

PeacticAdjectives5.Text = string.Empty;

}

}

}

private void SavePeacticPronouns\_Click(object sender, RoutedEventArgs e)

{

if (PeacticPronouns1.Text == "" || PeacticPronouns2.Text == "" || PeacticPronouns3.Text == "" || PeacticPronouns4.Text == "" || PeacticPronouns5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticPronouns1.Text + PeacticPronouns2.Text + PeacticPronouns3.Text + PeacticPronouns4.Text + PeacticPronouns5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticPronouns1.Text = string.Empty;

PeacticPronouns2.Text = string.Empty;

PeacticPronouns3.Text = string.Empty;

PeacticPronouns4.Text = string.Empty;

PeacticPronouns5.Text = string.Empty;

}

}

}

private void SavePeacticVerbs1\_Click(object sender, RoutedEventArgs e)

{

if (PeacticVerbs11.Text == "" || PeacticVerbs12.Text == "" || PeacticVerbs13.Text == "" || PeacticVerbs14.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticVerbs11.Text + PeacticVerbs12.Text + PeacticVerbs13.Text + PeacticVerbs14.Text ;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticVerbs11.Text = string.Empty;

PeacticVerbs12.Text = string.Empty;

PeacticVerbs13.Text = string.Empty;

PeacticVerbs14.Text = string.Empty;

}

}

}

private void SavePeacticVerbs2\_Click(object sender, RoutedEventArgs e)

{

if (PeacticVerbs21.Text == "" || PeacticVerbs22.Text == "" || PeacticVerbs23.Text == "" || PeacticVerbs24.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticVerbs21.Text + PeacticVerbs22.Text + PeacticVerbs23.Text + PeacticVerbs24.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticVerbs21.Text = string.Empty;

PeacticVerbs22.Text = string.Empty;

PeacticVerbs23.Text = string.Empty;

PeacticVerbs24.Text = string.Empty;

}

}

}

private void SavePeacticAccusative\_Click(object sender, RoutedEventArgs e)

{

if (PeacticAccusative1.Text == "" || PeacticAccusative2.Text == "" || PeacticAccusative3.Text == "" || PeacticAccusative4.Text == "" || PeacticAccusative5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticAccusative1.Text + PeacticAccusative2.Text + PeacticAccusative3.Text + PeacticAccusative4.Text + PeacticAccusative5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticAccusative1.Text = string.Empty;

PeacticAccusative2.Text = string.Empty;

PeacticAccusative3.Text = string.Empty;

PeacticAccusative4.Text = string.Empty;

PeacticAccusative5.Text = string.Empty;

}

}

}

private void SavePeacticDative\_Click(object sender, RoutedEventArgs e)

{

if (PeacticDative1.Text == "" || PeacticDative2.Text == "" || PeacticDative3.Text == "" || PeacticDative4.Text == "" || PeacticDative5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticDative1.Text + PeacticDative2.Text + PeacticDative3.Text + PeacticDative4.Text + PeacticDative5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticDative1.Text = string.Empty;

PeacticDative2.Text = string.Empty;

PeacticDative3.Text = string.Empty;

PeacticDative4.Text = string.Empty;

PeacticDative5.Text = string.Empty;

}

}

}

private void SavePeacticGenitive\_Click(object sender, RoutedEventArgs e)

{

if (PeacticGenitive1.Text == "" || PeacticGenitive2.Text == "" || PeacticGenitive3.Text == "" || PeacticGenitive4.Text == "" || PeacticGenitive5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticGenitive1.Text + PeacticGenitive2.Text + PeacticGenitive3.Text + PeacticGenitive4.Text + PeacticGenitive5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticGenitive1.Text = string.Empty;

PeacticGenitive2.Text = string.Empty;

PeacticGenitive3.Text = string.Empty;

PeacticGenitive4.Text = string.Empty;

PeacticGenitive5.Text = string.Empty;

}

}

}

private void SavePeacticAblative\_Click(object sender, RoutedEventArgs e)

{

if (PeacticAblative1.Text == "" || PeacticAblative2.Text == "" || PeacticAblative3.Text == "" || PeacticAblative4.Text == "" || PeacticAblative5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticAblative1.Text + PeacticAblative2.Text + PeacticAblative3.Text + PeacticAblative4.Text + PeacticAblative5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticAblative1.Text = string.Empty;

PeacticAblative2.Text = string.Empty;

PeacticAblative3.Text = string.Empty;

PeacticAblative4.Text = string.Empty;

PeacticAblative5.Text = string.Empty;

}

}

}

private void SavePeacticPrepositional\_Click(object sender, RoutedEventArgs e)

{

if (PeacticPrepositional1.Text == "" || PeacticPrepositional2.Text == "" || PeacticPrepositional3.Text == "" || PeacticPrepositional4.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticPrepositional1.Text + PeacticPrepositional2.Text + PeacticPrepositional3.Text + PeacticPrepositional4.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticPrepositional1.Text = string.Empty;

PeacticPrepositional2.Text = string.Empty;

PeacticPrepositional3.Text = string.Empty;

PeacticPrepositional4.Text = string.Empty;

}

}

}

private void SavePeacticPast\_Click(object sender, RoutedEventArgs e)

{

if (PeacticPast1.Text == "" || PeacticPast2.Text == "" || PeacticPast3.Text == "" || PeacticPast4.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticPast1.Text + PeacticPast2.Text + PeacticPast3.Text + PeacticPast4.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticPast1.Text = string.Empty;

PeacticPast2.Text = string.Empty;

PeacticPast3.Text = string.Empty;

PeacticPast4.Text = string.Empty;

}

}

}

private void SavePeacticFuture\_Click(object sender, RoutedEventArgs e)

{

if (PeacticFuture1.Text == "" || PeacticFuture2.Text == "" || PeacticFuture3.Text == "" || PeacticFuture4.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticFuture1.Text + PeacticFuture2.Text + PeacticFuture3.Text + PeacticFuture4.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticFuture1.Text = string.Empty;

PeacticFuture2.Text = string.Empty;

PeacticFuture3.Text = string.Empty;

PeacticFuture4.Text = string.Empty;

}

}

}

private void SavePeacticParticles\_Click(object sender, RoutedEventArgs e)

{

if (PeacticParticles1.Text == "" || PeacticParticles2.Text == "" || PeacticParticles3.Text == "" || PeacticParticles4.Text == "" || PeacticParticles5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticParticles1.Text + PeacticParticles2.Text + PeacticParticles3.Text + PeacticParticles4.Text + PeacticParticles5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticParticles1.Text = string.Empty;

PeacticParticles2.Text = string.Empty;

PeacticParticles3.Text = string.Empty;

PeacticParticles4.Text = string.Empty;

PeacticParticles5.Text = string.Empty;

}

}

}

private void SavePeacticQuestions\_Click(object sender, RoutedEventArgs e)

{

if (PeacticQuestions1.Text == "" || PeacticQuestions2.Text == "" || PeacticQuestions3.Text == "" || PeacticQuestions4.Text == "" || PeacticQuestions5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticQuestions1.Text + PeacticQuestions2.Text + PeacticQuestions3.Text + PeacticQuestions4.Text + PeacticQuestions5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticQuestions1.Text = string.Empty;

PeacticQuestions2.Text = string.Empty;

PeacticQuestions3.Text = string.Empty;

PeacticQuestions4.Text = string.Empty;

PeacticQuestions5.Text = string.Empty;

}

}

}

private void SavePeacticPrepositions\_Click(object sender, RoutedEventArgs e)

{

if (PeacticPrepositions1.Text == "" || PeacticPrepositions2.Text == "" || PeacticPrepositions3.Text == "" || PeacticPrepositions4.Text == "" || PeacticPrepositions5.Text == "") MessageBox.Show("No todos los campos están llenos!");

else

{

string answer = PeacticPrepositions1.Text + PeacticPrepositions2.Text + PeacticPrepositions3.Text + PeacticPrepositions4.Text + PeacticPrepositions5.Text;

if (answer == count)

{

MessageBox.Show("Prueba aprobada!");

ChangeStatusPractic();

}

else

{

MessageBox.Show("Prueba no aprobada!");

PeacticPrepositions1.Text = string.Empty;

PeacticPrepositions2.Text = string.Empty;

PeacticPrepositions3.Text = string.Empty;

PeacticPrepositions4.Text = string.Empty;

PeacticPrepositions5.Text = string.Empty;

}

}

Окно «MainWindow»:

namespace Calanthe

{

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Hyperlink\_Click(object sender, RoutedEventArgs e)

{

Process.Start("https://t.me/calanthe\_er");

}

private void Continue\_b\_Click(object sender, RoutedEventArgs e)

{

Autorization \_win = new Autorization();

this.Close();

\_win.Show(); }}}

Окно «Menu»:

namespace Calanthe

{

public partial class Menu : Window

{

string mail;

public Menu(string mail)

{

InitializeComponent();

this.mail = mail;

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Autorization \_win = new Autorization();

this.Close();

\_win.Show();

}

private void Lessons\_b\_Click(object sender, RoutedEventArgs e)

{

Lessons \_win = new Lessons(mail);

this.Close();

\_win.Show();

}

private void Dictionary\_b\_Click(object sender, RoutedEventArgs e)

{

Dictionary \_win = new Dictionary(mail);

this.Close();

\_win.Show();

}

private void Profile\_b\_Click(object sender, RoutedEventArgs e)

{

Profil \_win = new Profil(mail);

this.Close();

\_win.Show(); }}}

Окно «EditProfil»:

namespace Calanthe

{

public partial class EditProfil : Window

{

string mail;

string password;

String filename;

Student user = new Student();

CalantheEntities db = new CalantheEntities();

public EditProfil(string mail)

{

InitializeComponent();

this.mail = mail;

Mail.Text = mail;

foreach (var user in db.Student)

{

if (Mail.Text == user.Email)

{

imageEllipse.Fill = new ImageBrush(LoadImage(user.Image));

Login.Text = user.Login;

Password.Text = user.Password;

break;

}

}

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Profil \_win = new Profil(mail);

this.Close();

\_win.Show();

}

private void GoOut\_b\_Click(object sender, RoutedEventArgs e)

{

Login.Text = null;

Mail.Text = null;

Password.Text = null;

Autorization \_win = new Autorization();

this.Close();

\_win.Show();

}

private void Save\_b\_Click(object sender, RoutedEventArgs e)

{

foreach (var user in db.Student)

{

if (Mail.Text == user.Email)

{

user.Login = Login.Text;

user.Password = Password.Text;

break;

}

}

db.SaveChanges();

MessageBox.Show("Guardado!");

}

private static BitmapImage LoadImage(byte[] imageData)

{

if (imageData == null || imageData.Length == 0) return null;

var image = new BitmapImage();

using (var mem = new MemoryStream(imageData))

{

mem.Position = 0;

image.BeginInit();

image.CreateOptions = BitmapCreateOptions.PreservePixelFormat;

image.CacheOption = BitmapCacheOption.OnLoad;

image.UriSource = null;

image.StreamSource = mem;

image.EndInit();

}

image.Freeze();

return image;

}

private void NewFoto\_Click(object sender, RoutedEventArgs e)

{

byte[] buffer;

Microsoft.Win32.OpenFileDialog dlg = new Microsoft.Win32.OpenFileDialog();

dlg.FileName = "Document";

dlg.DefaultExt = ".docx";

dlg.Filter = "Text documents (.png; .jpg) | \*.png;\*.jpg;";

Nullable<bool> result = dlg.ShowDialog();

if (result == true)

{

filename = dlg.FileName;

buffer = File.ReadAllBytes(filename);

try

{

Student student = new Student();

foreach (var user in db.Student)

{

if (Mail.Text == user.Email)

{

user.Image = buffer;

imageEllipse.Fill = new ImageBrush(LoadImage(user.Image));

break;

}

}

db.SaveChanges();

}

catch (Exception)

{

MessageBox.Show("Error!");

}

}

}

private void Del\_Click(object sender, RoutedEventArgs e)

{

int n = 0;

byte[] buffer;

buffer = System.IO.File.ReadAllBytes(@"ziro\_foto.png");

Student student = new Student();

foreach (var user in db.Student)

{

if (Mail.Text == user.Email)

{

user.Image = buffer;

imageEllipse.Fill = new ImageBrush(LoadImage(user.Image));

break;

}

}

db.SaveChanges();

}

private void Password\_pb\_PasswordChanged(object sender, RoutedEventArgs e)

{

Password.Text = Password\_pb.Password;

}

private void Password\_TextChanged(object sender, TextChangedEventArgs e)

{

Password\_pb.Password = Password.Text;

}

private void eye\_Click(object sender, RoutedEventArgs e)

{

if (Password\_pb.Visibility == Visibility.Visible)

{

Password\_pb.Visibility = Visibility.Collapsed;

Password.Visibility = Visibility.Visible;

Password.Text = Password\_pb.Password;

}

else

{

Password\_pb.Visibility = Visibility.Visible;

Password.Visibility = Visibility.Collapsed;

Password\_pb.Password = Password.Text; }}}}

Окно «Profil»:

namespace Calanthe

{

public partial class Profil : Window

{

string mail;

Student user = new Student();

CalantheEntities db = new CalantheEntities();

Statistics statistics = new Statistics();

string filename = null;

int count = 0;

public Profil(string mail)

{

InitializeComponent();

this.mail = mail;

foreach (var user in db.Student)

{

if (mail == user.Email)

{

imageEllipse.Fill = new ImageBrush(LoadImage(user.Image));

break;

}

}

var stat = db.Statistics.Where(p => p.Email == mail && p.StatusPractic == "Пройдено" && p.StatusTeoria == "Пройдено");

foreach(Statistics p in stat)

{

for (int i = 0; i < 16; i++)

{

count += 1;

break;

}

}

switch (count)

{

case 1:

rcg1.Fill = Brushes.Blue;

break;

case 2:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

break;

case 3:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

break;

case 4:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

break;

case 5:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

break;

case 6:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

break;

case 7:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

break;

case 8:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

break;

case 9:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

rcg9.Fill = Brushes.Fuchsia;

break;

case 10:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

rcg9.Fill = Brushes.Fuchsia;

rcg10.Fill = Brushes.Fuchsia;

break;

case 11:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

rcg9.Fill = Brushes.Fuchsia;

rcg10.Fill = Brushes.Fuchsia;

rcg11.Fill = Brushes.Fuchsia;

break;

case 12:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

rcg9.Fill = Brushes.Fuchsia;

rcg10.Fill = Brushes.Fuchsia;

rcg11.Fill = Brushes.Fuchsia;

rcg12.Fill = Brushes.Fuchsia;

break;

case 13:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

rcg9.Fill = Brushes.Fuchsia;

rcg10.Fill = Brushes.Fuchsia;

rcg11.Fill = Brushes.Fuchsia;

rcg12.Fill = Brushes.Fuchsia;

rcg13.Fill = Brushes.GreenYellow;

break;

case 14:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

rcg9.Fill = Brushes.Fuchsia;

rcg10.Fill = Brushes.Fuchsia;

rcg11.Fill = Brushes.Fuchsia;

rcg12.Fill = Brushes.Fuchsia;

rcg13.Fill = Brushes.GreenYellow;

rcg14.Fill = Brushes.GreenYellow;

break;

case 15:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

rcg9.Fill = Brushes.Fuchsia;

rcg10.Fill = Brushes.Fuchsia;

rcg11.Fill = Brushes.Fuchsia;

rcg12.Fill = Brushes.Fuchsia;

rcg13.Fill = Brushes.GreenYellow;

rcg14.Fill = Brushes.GreenYellow;

rcg15.Fill = Brushes.GreenYellow;

break;

case 16:

rcg1.Fill = Brushes.Blue;

rcg2.Fill = Brushes.Blue;

rcg3.Fill = Brushes.Blue;

rcg4.Fill = Brushes.Blue;

rcg5.Fill = Brushes.Goldenrod;

rcg6.Fill = Brushes.Goldenrod;

rcg7.Fill = Brushes.Goldenrod;

rcg8.Fill = Brushes.Goldenrod;

rcg9.Fill = Brushes.Fuchsia;

rcg10.Fill = Brushes.Fuchsia;

rcg11.Fill = Brushes.Fuchsia;

rcg12.Fill = Brushes.Fuchsia;

rcg13.Fill = Brushes.GreenYellow;

rcg14.Fill = Brushes.GreenYellow;

rcg15.Fill = Brushes.GreenYellow;

rcg16.Fill = Brushes.GreenYellow;

break;

}

}

private void Continue\_b\_Click(object sender, RoutedEventArgs e)

{

EditProfil \_win = new EditProfil(mail);

this.Close();

\_win.Show();

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Menu \_win = new Menu(mail);

this.Close();

\_win.Show();

}

private static BitmapImage LoadImage(byte[] imageData)

{

if (imageData == null || imageData.Length == 0) return null;

var image = new BitmapImage();

using (var mem = new MemoryStream(imageData))

{

mem.Position = 0;

image.BeginInit();

image.CreateOptions = BitmapCreateOptions.PreservePixelFormat;

image.CacheOption = BitmapCacheOption.OnLoad;

image.UriSource = null;

image.StreamSource = mem;

image.EndInit();

}

image.Freeze();

return image;

} } }

Окно «Registration»:

namespace Calanthe

{

public partial class Registration : Window

{

Student user = new Student();

CalantheEntities db = new CalantheEntities();

public Registration()

{

InitializeComponent();

}

private void Back\_b\_Click(object sender, RoutedEventArgs e)

{

Autorization \_win = new Autorization();

this.Close();

\_win.Show();

}

private void Continue\_b\_Click(object sender, RoutedEventArgs e)

{

int n = 0;

byte[] buffer;

buffer = System.IO.File.ReadAllBytes(@"ziro\_foto.png");

if (Password.Text == "" || Login.Text == "" || Mail.Text == "") MessageBox.Show("No ha introducido todos los datos!");

else

{

foreach (var item in db.Student)

{

if (item.Email == Mail.Text)

{

user = item;

n = 1; }}

if (n == 0)

{

string[] dataLogin = Mail.Text.Split('@');

if (dataLogin.Length == 2)

{

string[] data2Login = dataLogin[1].Split('.');

if (data2Login.Length == 2)

{

if (Password.Text.Length >= 8)

{

user.Email = Mail.Text.Trim();

user.Login = Login.Text.Trim();

user.Password = Password.Text.Trim();

user.Image = buffer;

db.Student.Add(user);

db.SaveChanges();

Statistics statistic;

for (int i = 0; i < 16; i++)

{

statistic = new Statistics(Mail.Text, i + 1, "Не пройдено", "Не пройдено");

db.Statistics.Add(statistic);

}

db.SaveChanges();

Autorization \_win = new Autorization();

this.Close();

\_win.Show();

}

else MessageBox.Show("La contraseña debe ser de más de 8 caracteres!");

}

else MessageBox.Show("El correo electrónico debe estar en formato: .. @ .. . ..!");

}

else MessageBox.Show("El correo electrónico debe estar en formato: .. @ .. . ..!");

}

if(n == 1) MessageBox.Show("Este correo ya ha sido usado!");

}

}

private void Password\_pb\_PasswordChanged(object sender, RoutedEventArgs e)

{

Password.Text = Password\_pb.Password;

}

private void Password\_TextChanged(object sender, TextChangedEventArgs e)

{

Password\_pb.Password = Password.Text;

}

private void eye\_Click(object sender, RoutedEventArgs e)

{

if (Password\_pb.Visibility == Visibility.Visible)

{

Password\_pb.Visibility = Visibility.Collapsed;

Password.Visibility = Visibility.Visible;

Password.Text = Password\_pb.Password;

}

else

{

Password\_pb.Visibility = Visibility.Visible;

Password.Visibility = Visibility.Collapsed;

Password\_pb.Password = Password.Text;}}}}

# ПРИЛОЖЕНИЕ Д

**Презентация**

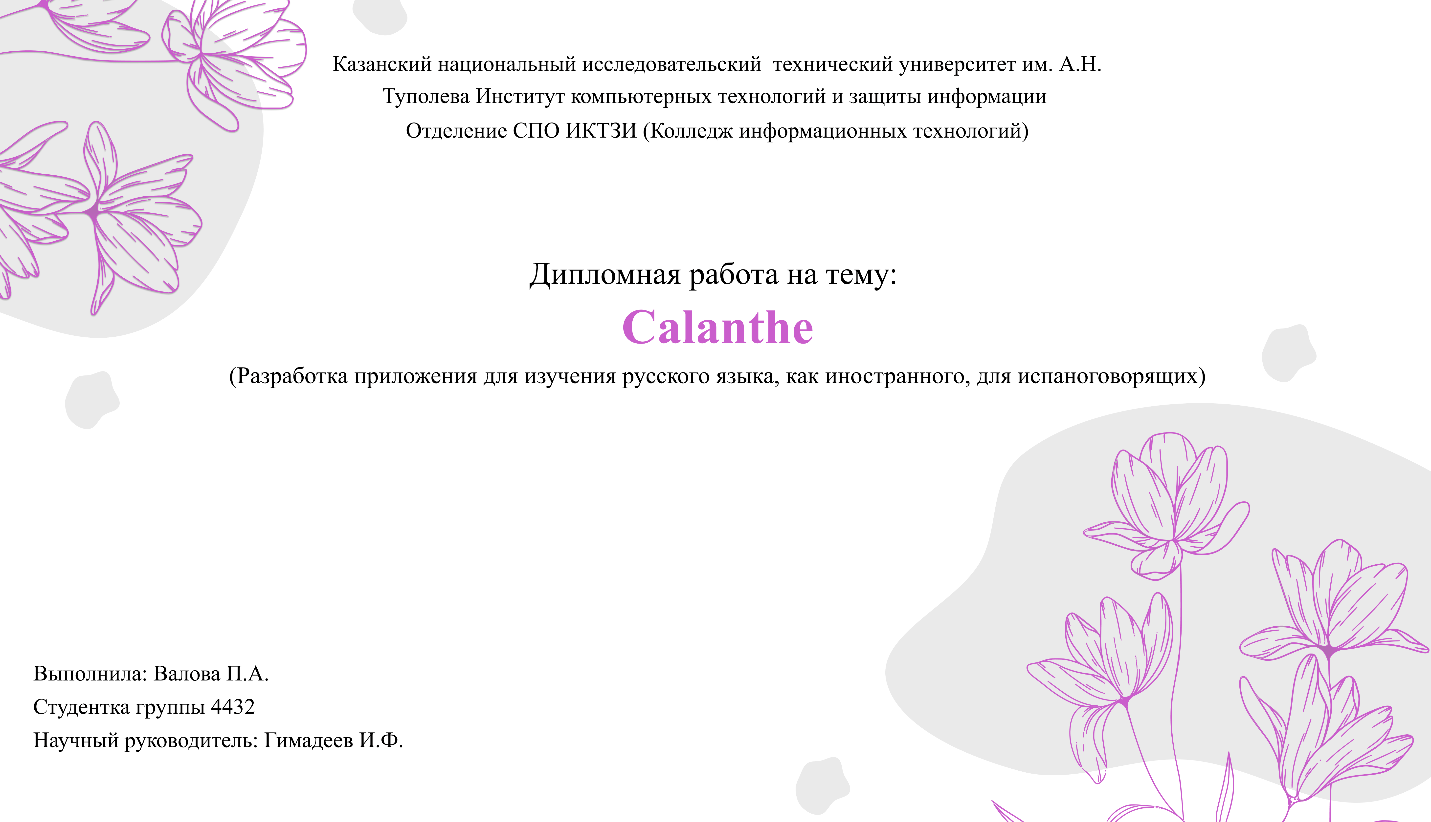


Рисунок Д.1 – Слайд №1

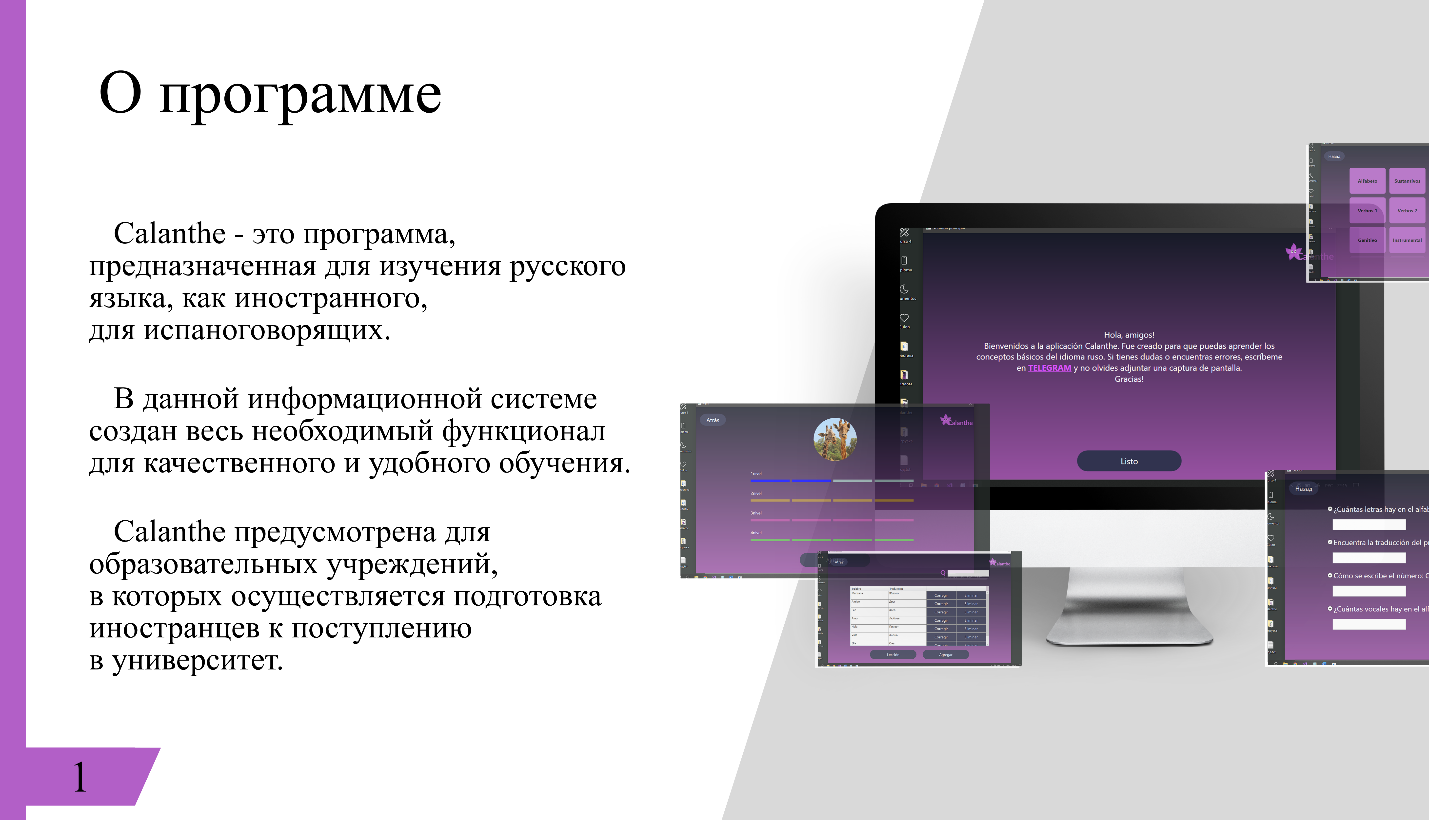


Рисунок Д.2 – Слайд №2



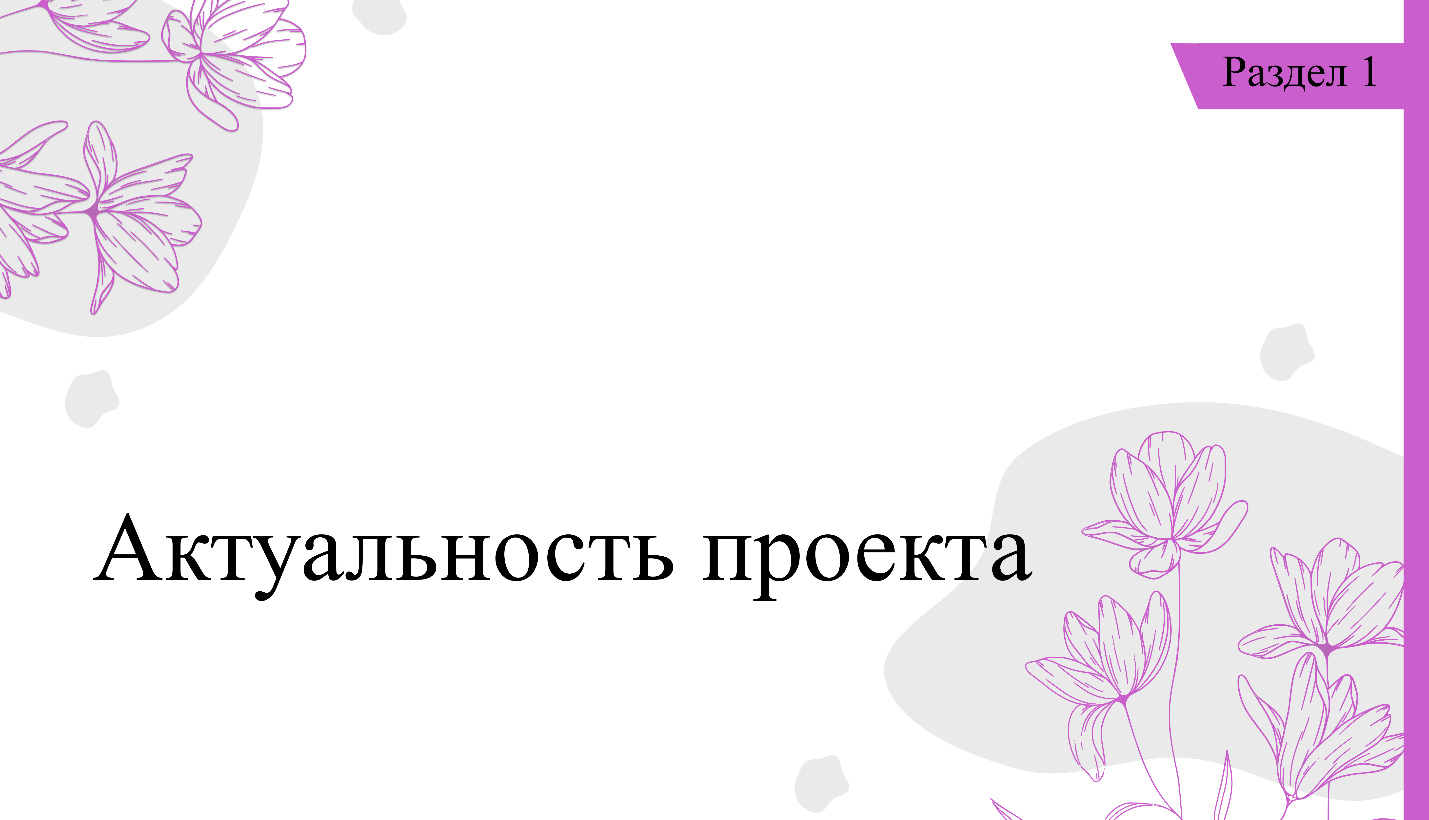
Рисунок Д.3 – Слайд №3

Рисунок Д.4 – Слайд №4



Рисунок Д.5 – Слайд №5



Рисунок Д.6 – Слайд №6



Рисунок Д.7 – Слайд №7

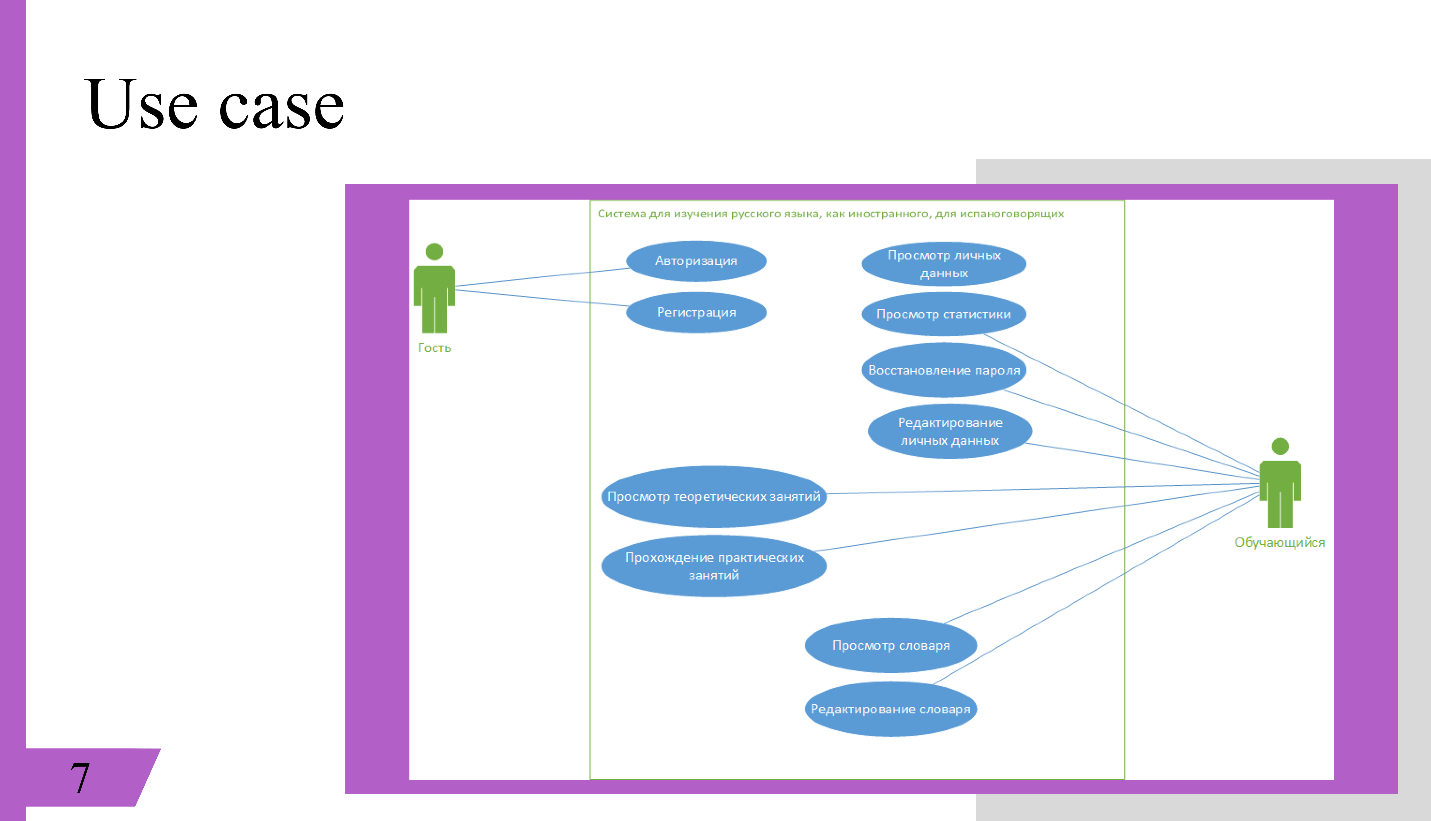


Рисунок Д.8 – Слайд №8

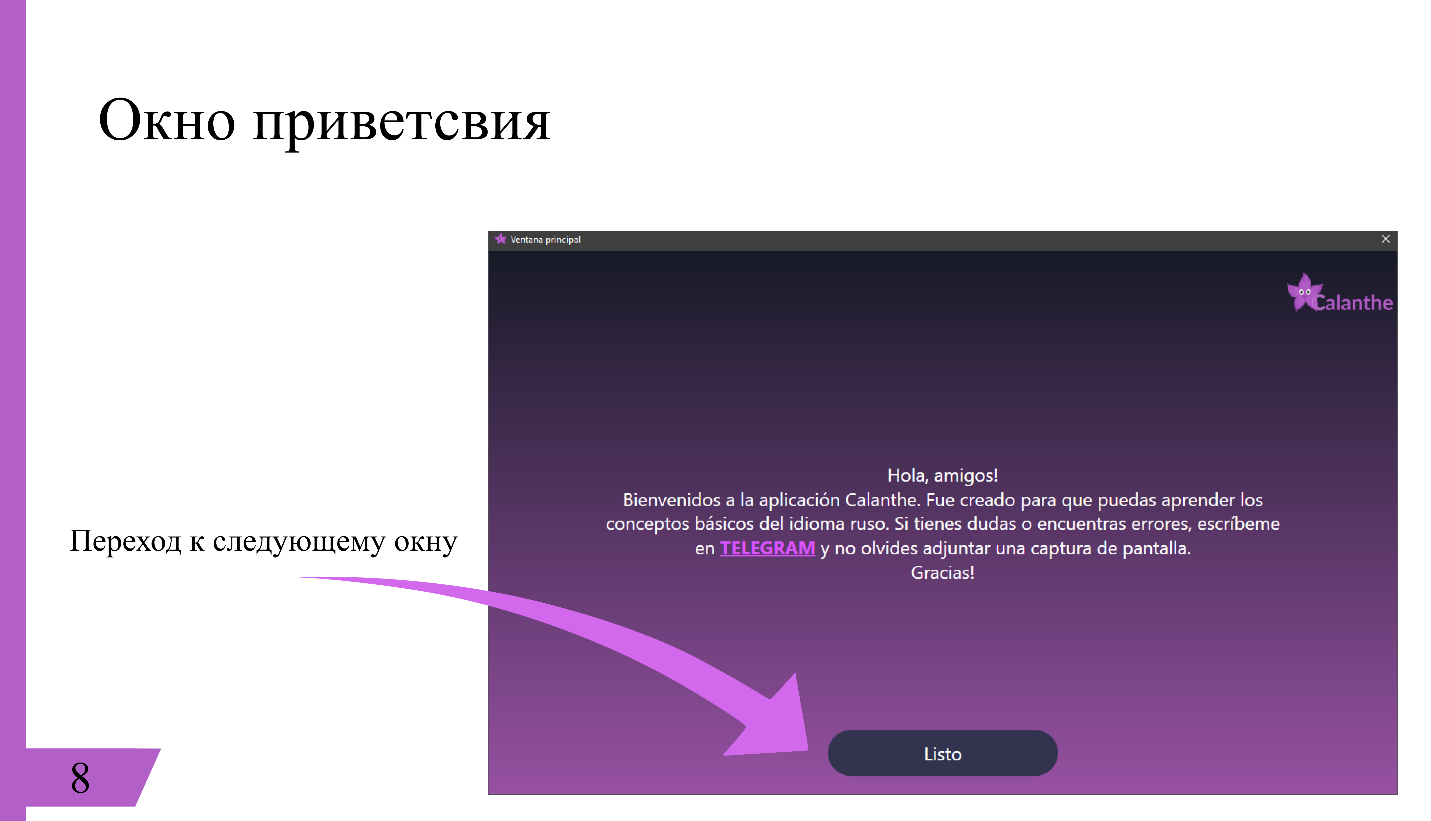


Рисунок Д.9 – Слайд №9

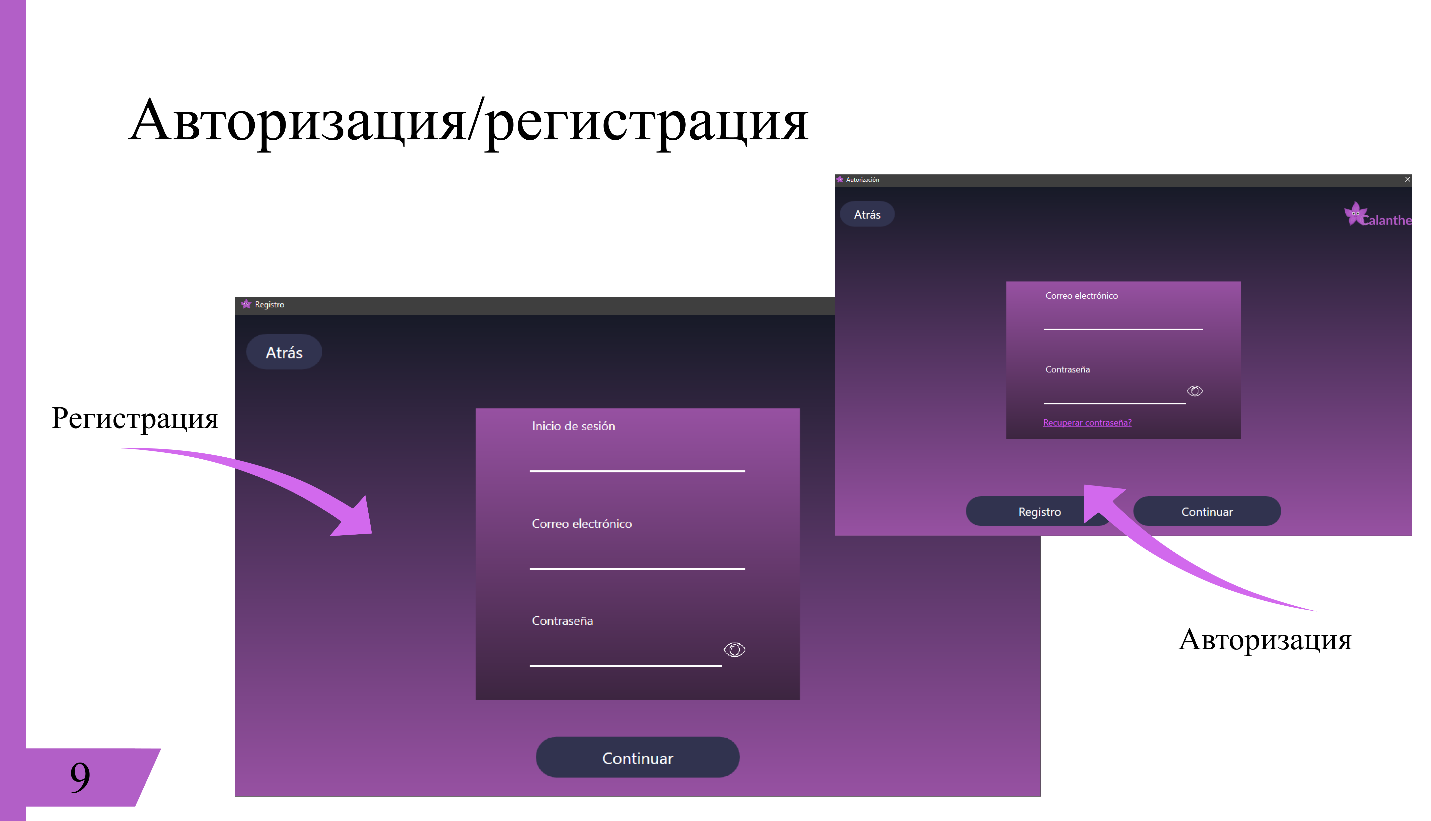


Рисунок Д.10 – Слайд №10

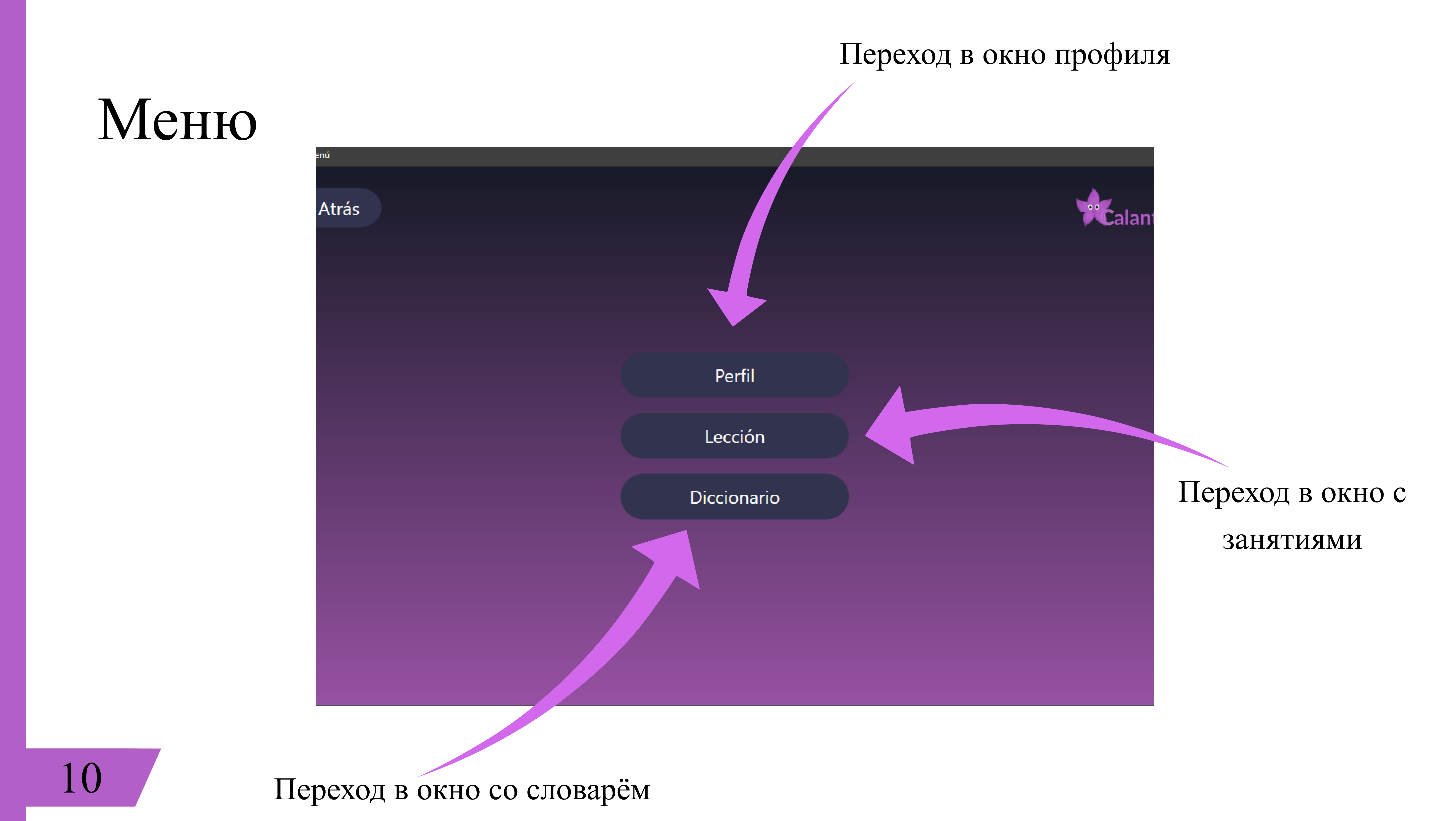


Рисунок Д.11 – Слайд №11

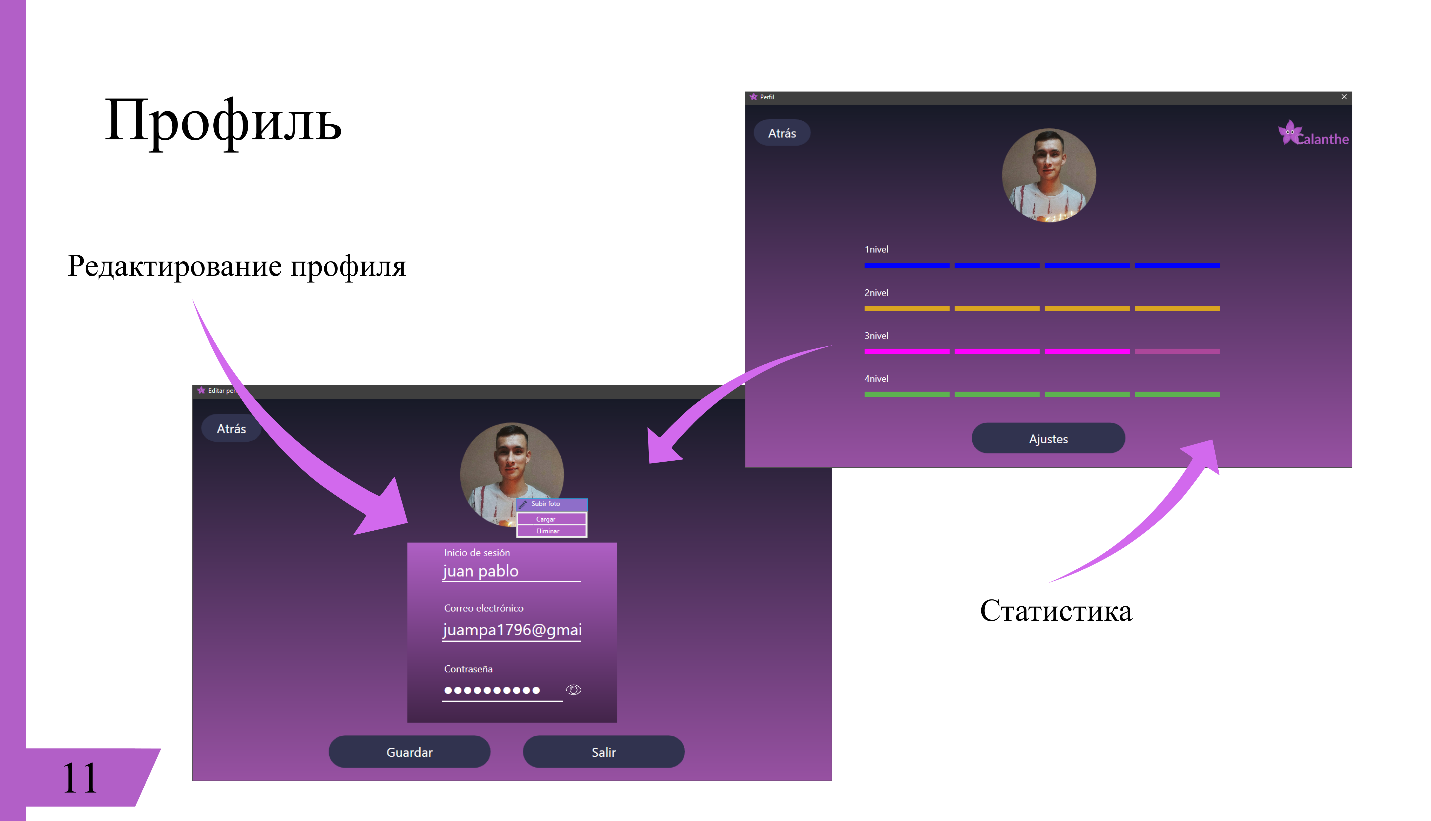


Рисунок Д.12 – Слайд №12

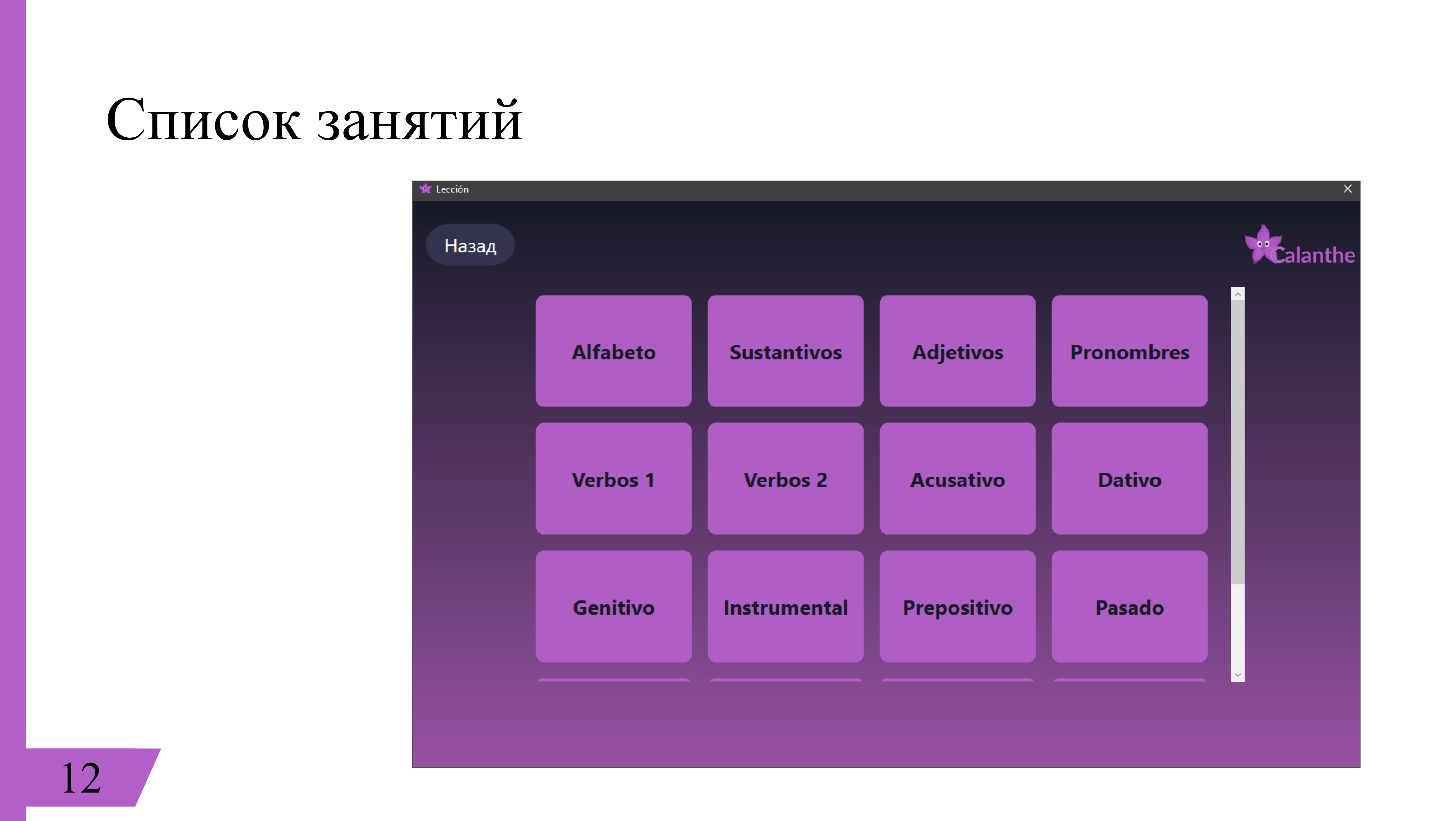


Рисунок Д.13 – Слайд №13

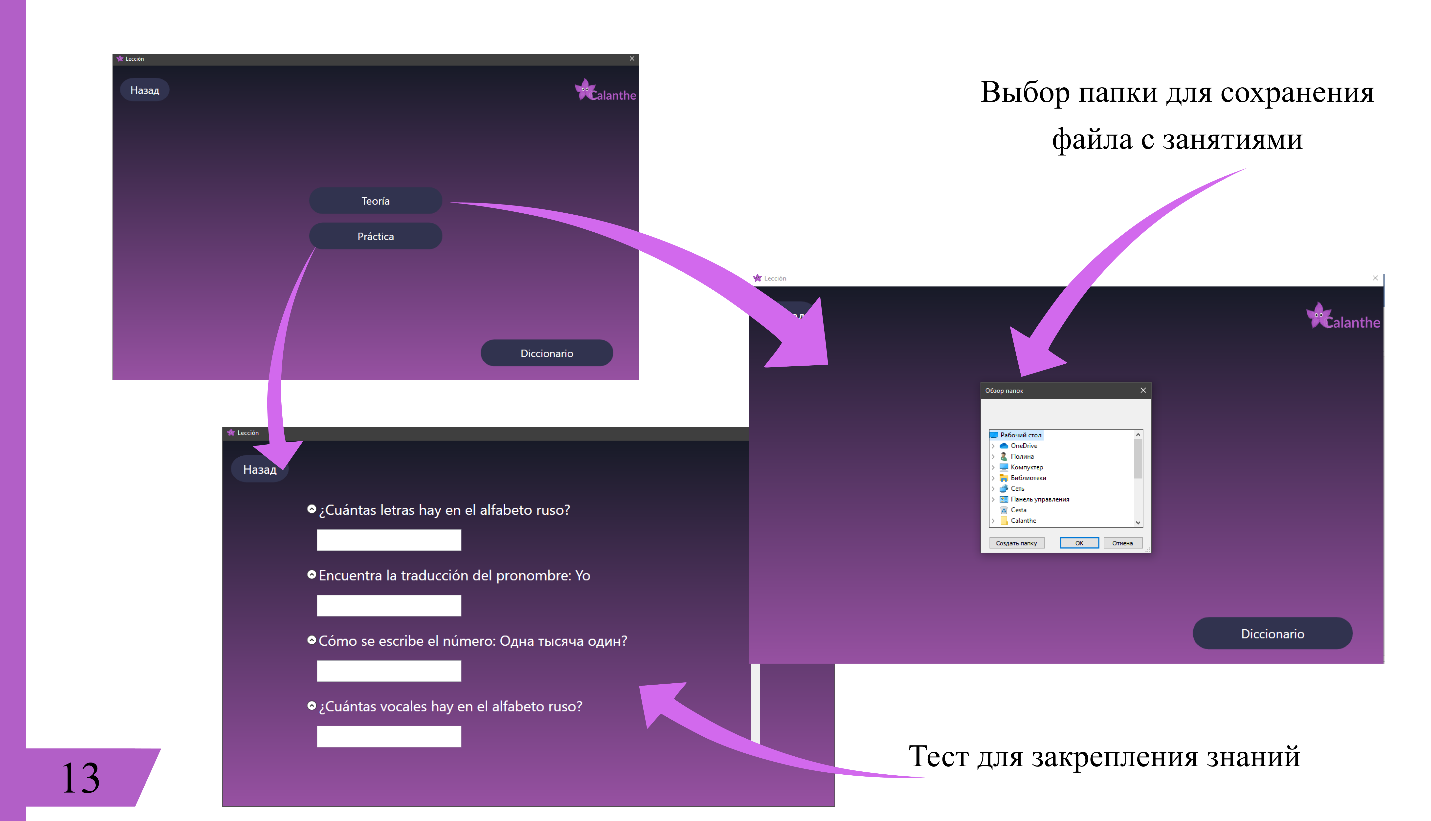


Рисунок Д.14 – Слайд №14

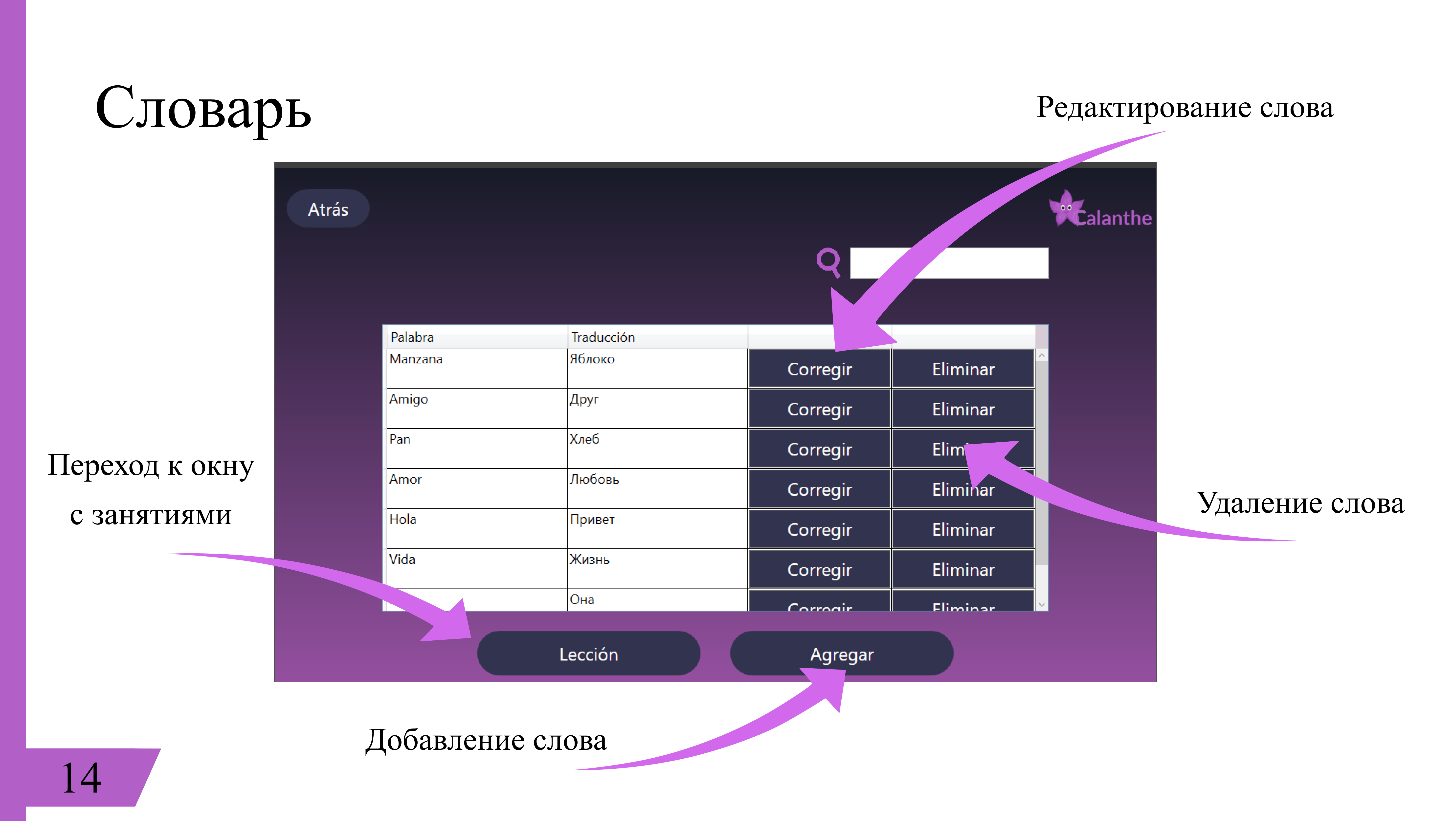


Рисунок Д.15 – Слайд №15

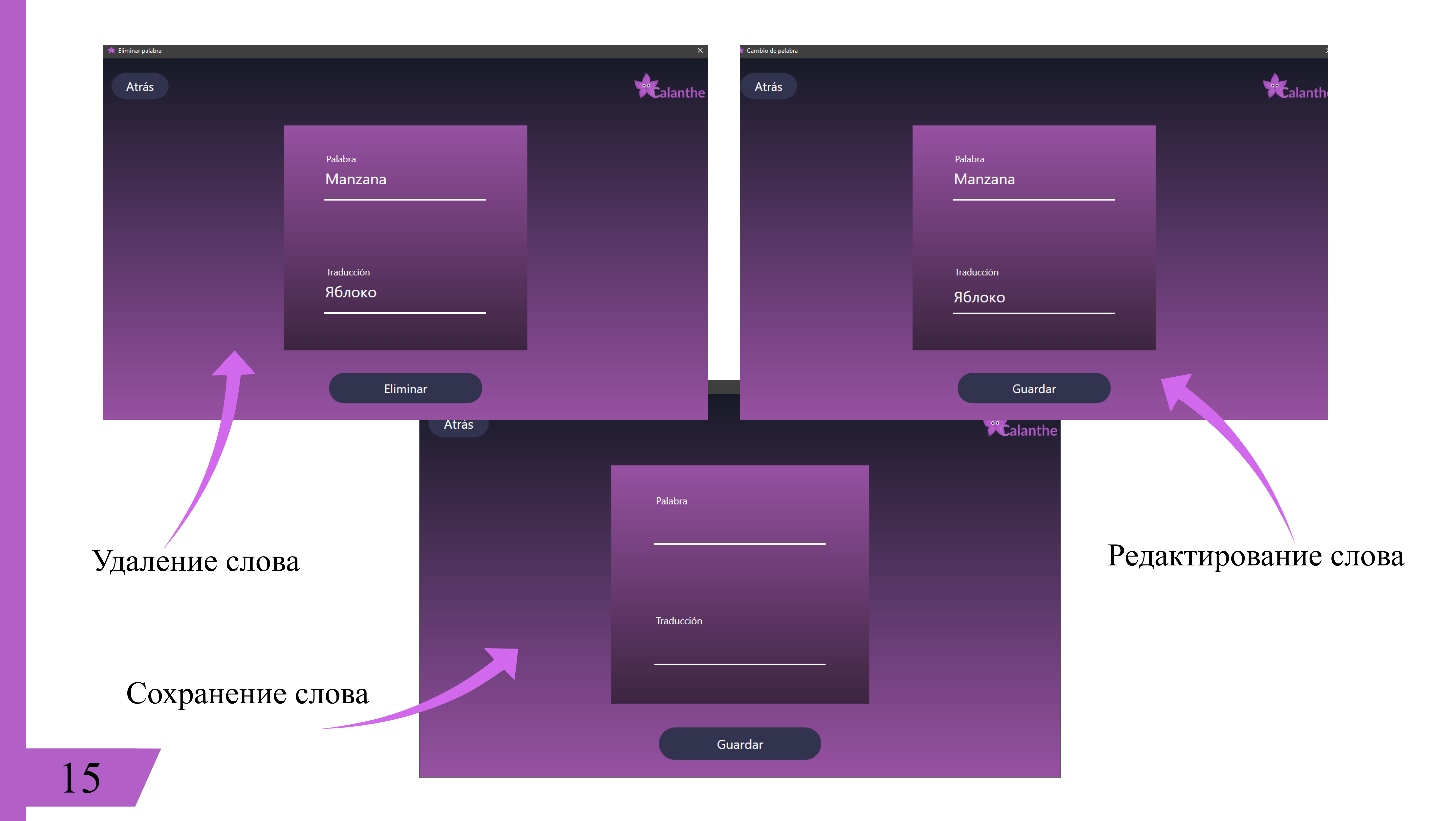


Рисунок Д.16 – Слайд №16



Рисунок Д.17 – Слайд №17

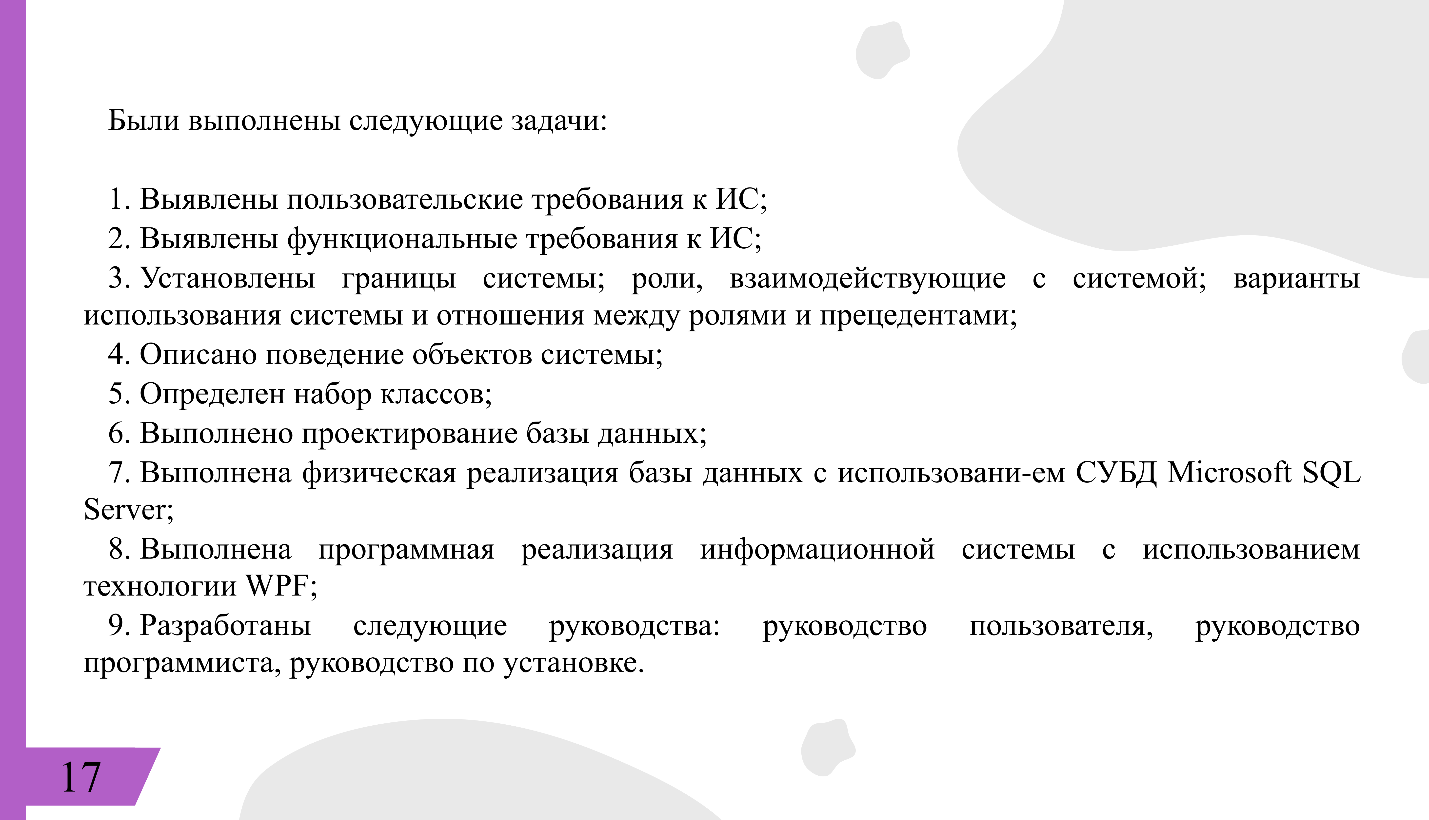


Рисунок Д.18 – Слайд №18



Рисунок Д.19 – Слайд №19