ВВЕДЕНИЕ

Мы живём в век информационных технологий, в век в котором техника может выполнять задачи, которые обременяют нас в повседневности. В настоящий момент существует масса мультимедийных систем, которые контролируют и упрощают наш повседневный быт. И при этом аждый день, все мы сталкиваемся с такими тривиальными задачами, как приготовление еды. Нужно знать, ЧТО ты хочешь приготовить, КАК и какие ПРОДУКТЫ необходимы. И при решении этой задачи мы сталкиваемся с проблемой упорядочения этих трех параметров: «ЧТО, КАК, Продукты?». А также актуальной задачей является экономия времени и средств.

Поэтому целесообразным инструментом решения поставленных задач, на мой взгляд, является приложение, подключенное к базе данных, представляющее собой «Книгу рецептов». Функциями этого приложения является: поиск рецептов блюд по широкому выбору критериев, просмотр детальной информации о приготовлении, составе продукта и так далее.

Целью курсовой работы является разработка приложения «Кулинарный справочник». Исходя из указанной цели, можно выделить частные задачи, поставленные в курсовой работе:

1. Проанализировать основные, наиболее интересуемые, критерии, которые интересуют человека, который ищет рецепты.
2. Изучить технологии: .NET,Windows Forms, C#, Visual Studio 2015,MS SQL.
3. Разработать структуру программы и базы данных, разработать функционал программы, соответствующий техническому заданию.
4. Разработать интерфейс и базу данных для приложения.

1 **Анализ предметной области**

1.1 Состояние вопроса

Для примера рассмотрим один из первых найденных в интернете кулинарных сайтов с рецептами, а также рассмотрим один книжный кулинарный справочник. В ходе рассмотрения будут анализированы недостатки и преимущества каждого из источников, на основании чего можно создавать технические требования приложения.  
 Для начала обратим внимание на книжный кулинарный справочник «Миллион рецептов праздничных блюд» [1], после чего исследуем сайт «Приятного аппетита» (http://priyatnogo-appetita.com) [2].

Таблица 1 – Анализ книжного кулинарного справочника «Миллион рецептов блюд»

|  |  |
| --- | --- |
| Достоинства | Недостатки |
| 1. «Всегда под рукой». Для использования этого ресурса нет необходимости в интернете либо каком-нибудь девайсе. 2. По мимо разбиения по категориям (салаты, закуски и т.д.) в книге присутствуют блюда к определенным праздникам, так же правила этикета, разнообразные способы украшения и подачи блюд. | 1.Не смотря на название книги, рецептов в книге отнюдь не миллион, а всего около семисот.  2. Рецепты в книге нельзя изменить, не испортив саму книгу. Так же плохи дела и с добавлением новых рецептов.  3. При том, что в книге рецептов не так уж и много, так и рецепты описаны очень минималистично (по 2 рецепта на 1 сторону страницы) и без всяких иллюстраций.  4. Поиск самих рецептов ограничен лишь только по названиям в содержании. |

Таблица 2 – Анализ Интернет-ресурса «Приятного аппетита».

|  |  |
| --- | --- |
| Достоинства | Недостатки |
| 1. Наличие иллюстраций и подробного описания. 2. Возможность оставлять комментарии к блюдам и выставлять оценки. 3. Возможность добавить рецепт (отправить заявку, которую должны будут подтвердить администраторы). | 1. Поиск возможен только по слову в названии. 2. Ресурс теряет свою работоспособность в случае отсутствия интернета.   . |

В ходе анализа были найдены достоинства и недостатки типичных кулинарных справочников. На основе данного анализа можно выделить основной функционал и требования, которые будет необходимо выполнить. Основным критерием является информативность рецептов, далее следует обратить внимание на удобный, быстрый поиск блюд. Для этого следует ввести необходимый функционал:

1. Поиск блюд, по категориям, названию, описанию и ингредиентам.
2. Возможность добавлять в рецепт фотографию.
3. Возможность в любой момент добавить новый рецепт или изменить существующий.

1.2 Актуальность и цель работы

Вопрос о поиске кулинарных блюд будет всегда актуален, так как связан с одной из основных и бытовых проблем, с которых человек связывается в повседневной жизни. Кулинария не перестает развиваться, так что и информационным ресурсам в этой сфере не стоит стоять на месте. При изучении состояния вопроса было уяснено, что текущие справочники и ресурсы, связанные с этим, имеют свои недостатки, поэтому целью моей курсовой будет создание приложения, которое позволит:

1. Искать блюдо для приготовления быстро, так как в приложении будет предусмотрена функция быстрого поиска, а также пользователю не обязательно будет регистрироваться или заходить на свой аккаунт.
2. Создавать закладки понравившихся и заинтересовавших блюд.
3. Искать блюда для приготовления подробно, с возможностью указывать необходимые бля блюда ингредиенты, категории и слова.
4. Делиться со всем миром своими кулинарными рецептами.

Целью курсовой работы является разработка приложения «Кулинарный справочник». Исходя из указанной цели, можно выделить частные задачи, поставленные в курсовой работе:

1. Выделить основные, наиболее интересуемые, моменты, которые интересуют человека, который ищет рецепты.
2. Изучить технологии: .NET, Windows Forms, C#, Visual Studio 2015, MS SQL.
3. Разработать структуру программы и базы данных, разработать функционал программы, соответствующий техническому заданию.
4. Разработать интерфейс и базу данных для приложения.

**2 Техническое задание**

2.1 Общие требования к продукту

Рисунок . Функциональная схема программы.

При запуске, в приложении, пользователь авторизирован, как гость. При этом он должен иметь возможность искать рецепты на таких же правах, как и зарегистрированный пользователь. Поиск должен быть двух типов: быстрый и детальный. Каждый вид поиска нужен в той или иной ситуации. Так же обязательно должна быть возможность просматривать подробную информации о способе приготовления блюда. При желании, пользователь должен иметь возможность зарегестрироваться, или войти в систему с помощью существующего аккаунта. Зарегистрированному пользователю добавляется возможность добавлять/удалять закладки всех блюд, а если пользователь отмечен, как модератор, то должна добавляться возможность редактирования, создания и удаления рецептов.

По мимо основного функционала, для всех категорий пользователей должна присутствовать справка, а также возможность смены аккаунта и настройки приложения.

2.2 Позиционирование продукта

2.2.1 Требование к пользовательским интерфейсам

Интерфейс программы должен быть не особо сложным, приятным на вид и учитывать существующие стандарты дизайна. Для облегчения взаимодействия программы с пользователем следует визуально разбить интерфейс на части, каждая из которых будет иметь свой функционал и предназначение.

Требования к интерфейсу приложения, представляемым пользователям:

Основной интерфейс (рис \*\*\*)

1. Главное окно
2. Блок, в котором будет отображаться список рецептов, подробный рецепт, редактирование рецепта, справочная информация.
3. Управляющая панель
4. Отображение состояния авторизации (имя пользователя, кнопка вода/выхода)
5. Панель с основными командами приложения (выход, настройки, вызов справки в блоке 2, создание рецепта, открытие закладок в блоке 2)
6. Кнопка запуска выбранного поиска
7. Панель настройки выбранного поиска
8. Блок поиска
9. Переключение на быстрый поиск
10. Переключение на детальный поиск

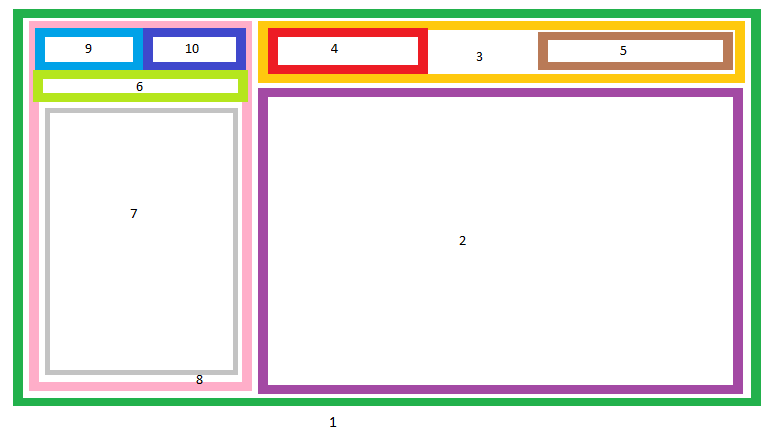
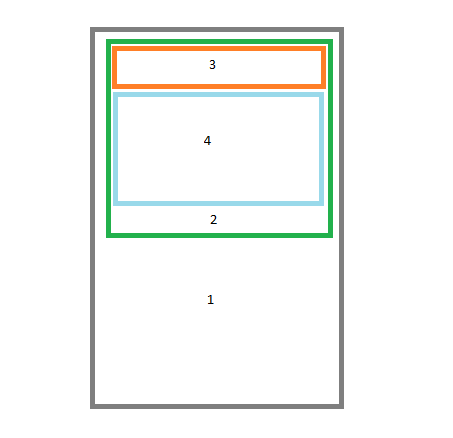


Рисунок 2

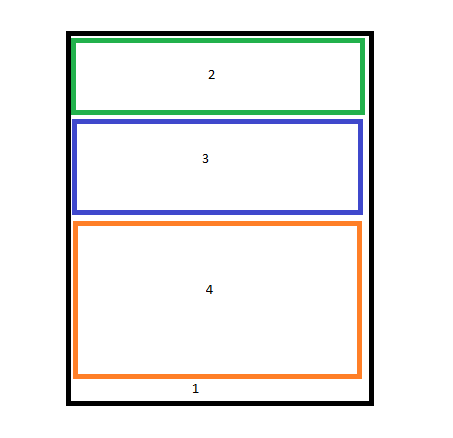
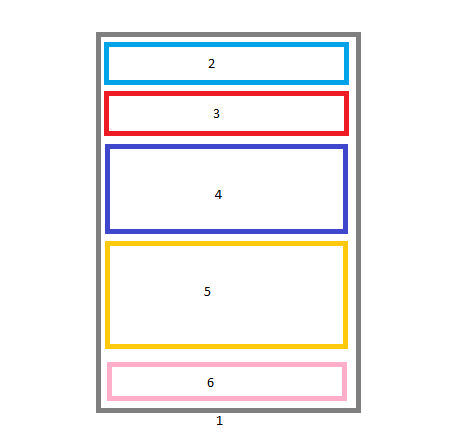
Данное окно является родительским, для всех окон. При запуске в главном блоке (блок 2) отображается справка по приложению, в блоке управления (блок 3) отображается имя «Гость» и все доступные ему операции, описанные в (\*\*\*). В блоке поиска по умолчанию выбран «Быстрый» поиск. В блоке быстрого поиска содержится столько дочерних панелей, сколько всего будет придумано вариантов разбиения на категории. Блок-схема блока поиска и его панелей описана далее:

1. Панель быстрого поиска
2. Панель категории
3. Название категории
4. Список элементов категории



При переключении на детальный поиск в блоке поиска блок быстрого поиска заменяется. Блок-схема панели дательного поиска выгляди следующим образом:

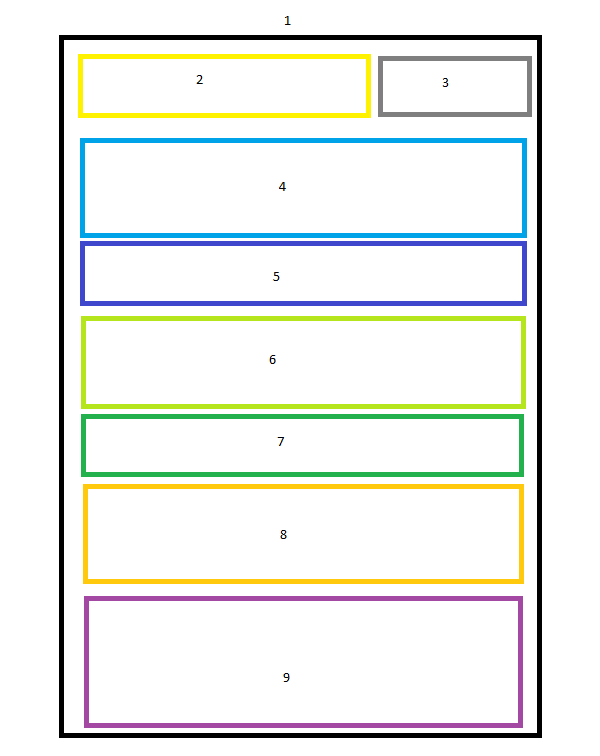
* Панель детального поиска
* Список ингредиентов для поиска
* Список элементов категорий для поиска
* Панель ингредиента в поиск
* Панель добавления элементов категорий
* Строка для ввода поискового слова

После выбора опций поиска и непосредственно запуска поиска в главном блоке открывается пустая панель просмотра списка рецептов и заполняется соответствующими рецептами. Каждый рецепт в этой панели можно изобразить следующей блок-схемой:

1. Панель минимального отображения рецепта
2. Название блюда
3. Ингредиенты
4. Часть описания приготовления

При желании, пользователь может нажать на любой из рецептов для детального просмотра. Блок-схемы детального просмотра блюда и редактирования блюда очень схожи. Отличаются они только дополнительными элементами в окне редактирования (4,6).

1. Панель, содержащая все элементы
2. Название блюда
3. Функциональная панель (содержит команды «сохранить», «назад», «редактировать» и т.д.)
4. Панель добавления подкатегории блюду
5. Список всех подкатегорий блюда
6. Добавление ингредиентов блюда
7. Список ингредиентов блюда
8. Редактирование или просмотр описания
9. Просмотр или изменение изображения результата



2.2.2 Требования к программным интерфейсам

Код программы должен быть понятным, содержать комментарии, не содержать повторяющегося кода. Создать класс для работы с базой данных.

(\*\*\*)

2.2.3 Требования к базе данных

База данных должна содержать данные о пользователях для успешного входа, полную информацию о блюдах, категориях, ингредиентах и закладках пользователя. Также потребуется обеспечить связи между таблицами. База данных должна хранить большие объемы данных и обеспечить удовлетворительную скорость работы. Для создания базы данных в данном курсовом проекте была выбрана СУБД MS SQL.

2.2.4 Требования к пользователям продукта

– Подключение к сети Интернет  
– Базовый навык работы с компьютером

2.3 Функции продукта

Основной функционал продукта:

1. Возможность быстрого поиска
2. Возможность гибкого поиска
3. Просмотр детальной и минимальной информации о рецепте
4. Настройка приложения
5. Получение справки
6. Добавление и редактирование блюд
7. Создание аккаунта и вход на существующие
8. Управление закладками (просмотр, удаление и добавление)

2.4 Сценарии использования продукта

Пользователь запускает приложение. Изначально пользователь авторизирован как гость. Открывается окно со справкой.

Основные сценарии

Сценарий 1: Пользователь нажимает на кнопку входа и появляется модальное окно входа в аккаунт.

Сценарий 2: Пользователь нажимает на кнопку быстрого поиска, в блоке поиска открывается окно с разными видами категорий для выбора.

Сценарий 3: Пользователь нажимает на кнопку детального поиска, в блоке поиска открывается окно детального поиска.

Сценарий 4: Пользователь нажимает на кнопку закладки (только для авторизированных пользователей), в главном блоке открывается окно для просмотра списка блюд, заполненное блюдами с закладок.

Сценарий 5: Пользователь нажимает на кнопку добавить блюдо (только для авторизированных пользователей, отмеченных как «доверенные»), в главном блоке открывается окно создания рецепта блюда.

Сценарий 6: Пользователь нажимает на кнопку справки, в главном блоке открывается окно с информацией об приложении и его использовании.

Сценарий 7: Пользователь нажимает на кнопку настроек и открывается модальный диалог настроек.

Сценарий 8: Пользователь нажимает на кнопку закрытия окна и осуществляется выход из приложения.

Сценарий 9: Пользователь нажимает на кнопку выхода с аккаунта (кнопка видима или включена только когда пользователь авторизирован) и в приложении становиться гостем.

Окно входа

Сценарий 1: пользователь вводит свои данные для входа.

Сценарий 1.1: пользователь успешно авторизуется и возвращается на главное окно, где в панели управления аккаунтом имя изменилось на имя пользователя, и все панели обновились согласно с правами доступа пользователя.

Сценарий 1.2: пользователь вводит данные неверно и нажимает кнопку входа. Появляется всплывающее окно с предупреждением о неверно набранном логине или пароле.

Сценарий 1.3: пользователь вводит данные верно, но база данных не отвечает, о чем пользователя уведомляет всплывающее через некоторое время окно.

Сценарий 2: пользователь нажимает кнопку регистрации и открывается окно регистрации.

Сценарий 3: кнопка отмены возвращает пользователя в главное окно, не изменивши при этом ничего.

Окно регистрации

Сценарий 1: пользователь вводит данные для регистрации.

Сценарий 1.1: данные не корректны - всплывает окно с предупреждением.

Сценарий 1.2: данные корректны, но база не отвечает - всплывает окно с предупреждением.

Сценарий 1.3: успешная регистрация, пользователь возвращается к окну входа.

Сценарий 2: отмена действия и возврат к окну входа.

Окно редактирования или создания блюда

Сценарий 1: пользователь вводит корректные данные и жмет кнопку сохранить, после чего перенаправляется в окно просмотра созданного блюда.

Сценарий 2: пользователь вводит некорректные данные.

Сценарий 2.1: неподходящее (пустое или существующее) имя. Пользователя предупреждают об этом.

Сценарий 2.2: несуществующий ингредиент. Если пользователь обладает правами, то пользователь может выбрать «занести ингредиент базу».

Окно отображения списка рецептов

Сценарий 1: пользователь нажимает на название блюда и переходит на странице детального просмотра.

\*\*\*( мб окно пользователя)

* 1. Функциональные требования

Таблица – Сценарий регистрации

|  |  |
| --- | --- |
| Сценарий использования | Регистрация |
| Приоритет | Важно/Желательно/Необязательно |
| Триггер | Нажатие на кнопку «Регистрация» |
| Предусловие | Открытие формы регистрации  Корректное заполнение обязательных полей формы |
| Основной сценарий | Пользователь уведомлен об успешной регистрации |
| Постусловие | Добавление нового пользователя в базу данных  Перенаправление пользователя в окно входа |
| Сценарий исключительных ситуаций | Пользователь не добавлен в базу, появляется окно с предупреждением. |

Таблица – Сценарий авторизации

|  |  |
| --- | --- |
| Сценарий использования | Вход в систему |
| Приоритет | Важно/Желательно/Необязательно |
| Триггер | Нажатие на кнопку «Вход» |
| Предусловие | Открытие формы входа  Корректное заполнение обязательных полей формы |
| Основной сценарий | Проверка пользователя в базе |
| Постусловие | Открывается главное окно приложения, пользователь авторизирован |
| Сценарий исключительных ситуаций | Пользователя нет в базе, появляется окно с предупреждением. |

Таблица – Сценарий создания блюда (рецепта блюда)

|  |  |
| --- | --- |
| Сценарий использования | Создание блюда |
| Приоритет | Важно/Желательно/Необязательно |
| Триггер | Нажатие на кнопку «Сохранить» на управляющей панели вкладки редактирования. |
| Предусловие | Открытое главное окно с помощью кнопки «добавить рецепт» |
| Основной сценарий | Добавление блюдо в базу |
| Постусловие | Перенаправление пользователя во вкладку просмотра рецепта. |
| Сценарий исключительных ситуаций | База не отвечает. Появляется окно с предупреждением. |

Таблица – Сценарий редактирования блюда

|  |  |
| --- | --- |
| Сценарий использования | Редактирование блюда |
| Приоритет | Важно/Желательно/Необязательно |
| Триггер | Нажатие на кнопку «Сохранить» на управляющей панели вкладки редактирования. |
| Предусловие | Открытие окна с помощью нажатия на клавишу редактирования блюда во вкладке просмотра блюда |
| Основной сценарий | Изменение блюда в базе |
| Постусловие | Перенаправление пользователя во вкладку просмотра рецепта. |
| Сценарий исключительных ситуаций | Запись была удалена из базы или база не отвечает. Появляется окно с предупреждением. |

Таблица – Сценарий создания закладки

|  |  |
| --- | --- |
| Сценарий использования | Создание закладки |
| Приоритет | Важно/Желательно/Необязательно |
| Триггер | Нажатие на кнопку добавления закладки на управляющей панели вкладки просмотра рецепта. |
| Предусловие | Открытое окно просмотра рецепта, авторизация пользователя |
| Основной сценарий | Добавление закладки в базу |
| Постусловие | Изменение кнопки на кнопку удаления. |
| Сценарий исключительных ситуаций | Запись была удалена из базы или база не отвечает. Появляется окно с предупреждением. |

**3 Реализация программного продукта**

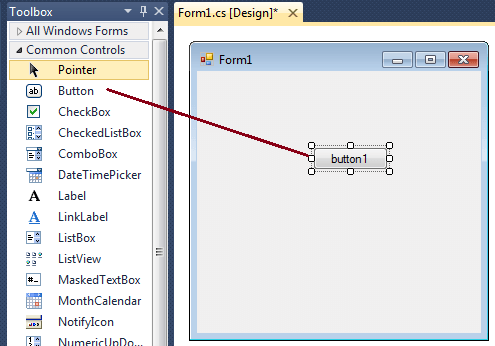
3.1 Обоснование средств разработки

3.1.1 Использование .NET Framework и Windows Form

Для создания GUI приложения было выбран Microsoft .NET Framework и его компонент Windows Forms. Windows Forms - новый стиль построения приложения на базе классов .NET Framework class library. Они имеют собственную модель программирования, которая более совершеннее, чем модели, основанные на Win32 API или MFC, и они выполняются в управляемой среде .NET Common Language Runtime (CLR). Это целая новая парадигма программирования, которая Она реализована на библиотеке классов .NET Framework class library и содержит более единую модель программирования, улучшенную защиту и более богатые возможности для написания полнофункциональных веб-приложений. И это только начало.

Windows Forms - одна из наиболее интересных возможностей Microsoft .NET. Если вы знакомы с MFC (или Windows API), то Windows Forms хорошее начало для работы с .NET Framework class library, потому что она позволяет писать традиционные GUI приложения с окнами, формами и т.п. вещами. Однажды, начав работать с Windows Forms можно быстро понять .NET Framework.

Главная выгода от написания Windows-приложений с использованием — Windows Forms - это то, что Windows Forms гомогенизируют (создают более однородную (гомогенную) структуру) программную модель и устраняют многие ошибки и противоречия от использования Windows API. Другая выгода от Windows Forms - вас используете тот же самый API, независимо от языка программирования, который вы выбрали. В прошлом, выбор языка программирования управлял выбором API. Если вы программировали в Visual Basic, вы использовали один API (реализованный на языке Visual Basic), в то время как программисты C использовали Win32 API, а программисты C++, вообще говоря, использовали MFC. MFC-программисту было трудно переключиться на Visual Basic и наоборот. Но теперь такого больше нет. Все приложения, которые используют Windows Forms, используют один API из .NET Framework class library. Знание одного API достаточно позволит программисту писать приложения фактически на любом языке, который он выберет.



Windows Forms никак не меньше, чем современная модель программирования для GUI приложений. В отличие от модели программирования Win32, в которой многое идет еще от Windows 1.0, новая модель была разработана с учетом всех современных требований.

3.1.2 Язык программирования С#

C# — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .NET Framework и впоследствии был стандартизирован как ECMA-334 иISO/IEC 23270.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы,итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Переняв многое от своих предшественников — языков C++, Pascal, Модула, Smalltalk и, в особенности, Java — С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличие от C++ не поддерживает множественное наследование классов (между тем допускается множественное наследование интерфейсов).

3.1.3 Visual Studio 2015

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. 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 Так, например, одним из используемых инструментов Visual Studio в данной курсовой работе является инструмент «Microsoft Visual Studio 2015 Installer Projects». Данный инструмент позволит создавать установочный паект для данного приложения за пару минут.

3.2 Концептуальное проектирование БД

Назначение концептуальной модели состоит в том, чтобы представить формализованную информацию о предметной области таким образом, чтобы она было достаточно емкой для оценки глубины и корректности проработки проекта базы данных.

Были выделены следующие сущности, необходимые для решения задачи:

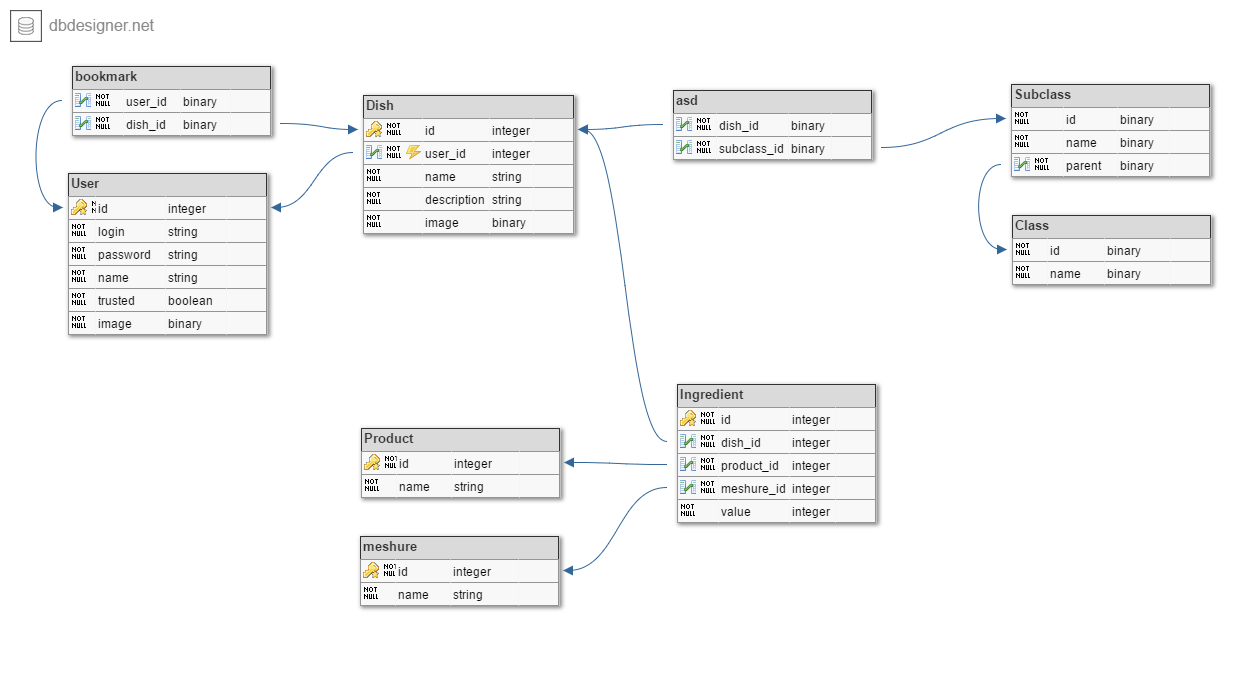
1. Пользователь
2. Блюдо
3. Ингредиент
4. Продукт
5. Категория
6. Элемент категории
7. Закладка

Основными сущностями БД, как можно заметить, являются блюдо (рецепт) и пользователь. При детальном рассмотрении ТЗ и требуемого функционала следует, что следует ввести еще роли. Т.е. каждый пользователь должен обладать ограниченными правами на доступ. На схеме ниже представлена концептуальная схема базы данных, которая описывает сущности-объекты, необходимые для решения поставленной задачи в данной курсовой работе.



3.1.2 Физическая модель БД

Физическое проектирование базы данных подразумевает процесс подготовки переноса реализации базы данных на определенную СУБД. На изображении ниже представлена физическая схема базы данных для курсового проекта. Для построения данной схемы было выбрано веб-приложение DbDesigner.

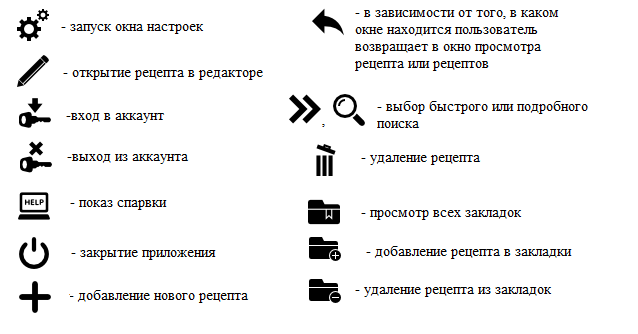
****

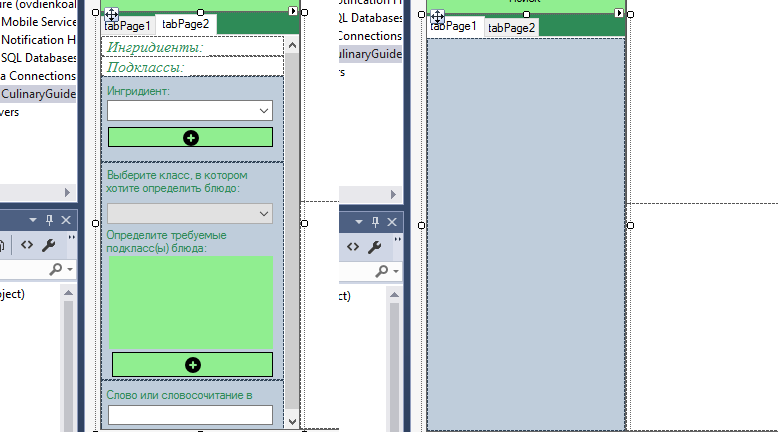
**4 Разработка электронного кулинарного справочника**

* 1. Разработка пользовательского интерфейса

Интерфейс приложения – важная особенность приложения, так как это связь между пользователем и программой. Основные характерные черты пользовательского интерфейса данного приложения:

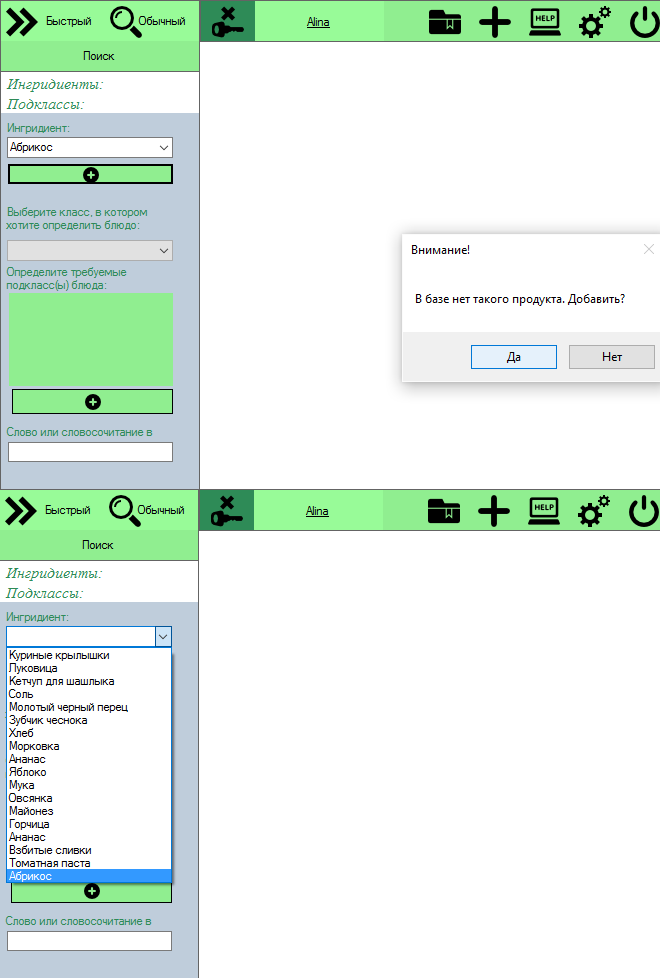
* В Windows Forms интерфейс приложения строится на стандартных элементах Windows, что упрощает понимание пользователем того, как с ними взаимодействовать.
* В дизайне приложения был использован популярный на данный момент стиль Flat.
* Основным цветом приложения выбран – зеленый, а в особенности: sea green, panel green и light green.
* Способ организации графического интерфейса – MDI (*multiple document interface).*

В соответствии вышесказанным, при запуске приложения пользователем открывается главное окно в нем открыта вкладка со справкой, а слева в блоке поиска открыт быстрый поиск (рис \*\*\*). Как можно увидеть, почти все кнопки оформлены в виде черно белых изображений, каждое из которых характеризует операцию, с которой эта кнопка или переключатель связанны. На рис \*\*\* изображены иконки всех кнопок и описание их функций. 

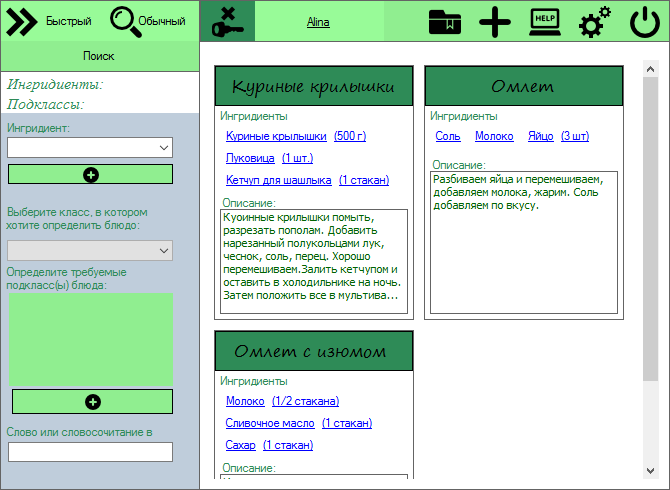
В соответствии с ТЗ главное окно состоит из трех логических частей: панель управления приложением и аккаунтом, панель поиска и главная панель. Главная панель и панель поиска реализованы с помощью элемента tab control. Данная особенность позволяет проектировать пользовательский интерфейс не программно (не создавая дочернее окно или меня свойство Visible каждый раз, при клике), а в визуальном редакторе Visual Studio (рис \*\*\*), также избавляет от проверки повторного нажатия клавиши выбора меню, что уберегает от перезагрузки формы, которое пользователь увидит, как «мигание». Блок поиска состоит из двух кнопок переключения, которые контролируют вкладки поискового tab control и кнопки запуска поиска, а главный tab control состоит из четырех страниц, все из которых будут рассмотрены далее. 

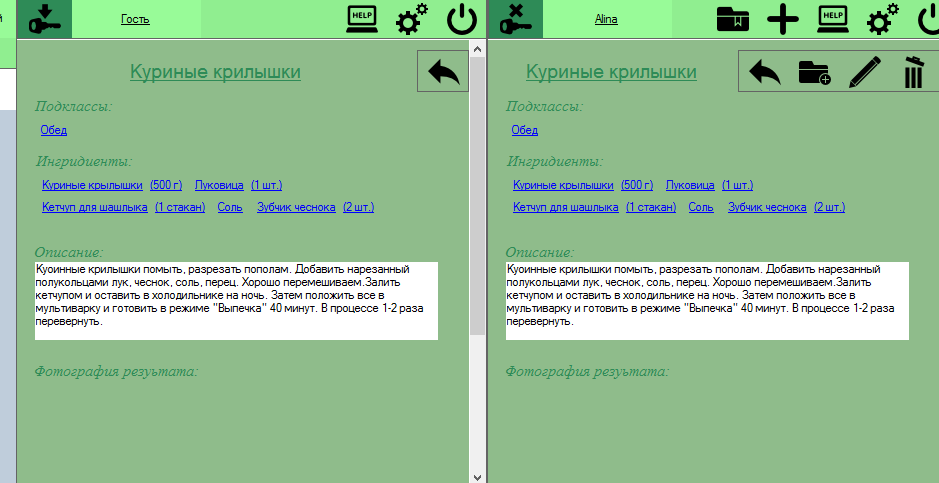
Начальная страница поискового tab control изначально пуста и заполняется лишь при старте приложения (рис \*\*\*). Как можно заметить заполняется эта страница пользовательскими элементами. Данный пользовательский элемент был разработан специально для отображения списка элементов категории, поэтому состоит из верхушки (название категории) и панели с переключателями check box (элементы категории), стилизованными под приложение.

Вторая страница поискового tab control имеет два списка list view, один для списка ингредиентов, второй для списка элементов категории. Каждый элемент списка – пользовательский элемент, который уничтожается при нажатии на него. Данная функция важна, так как это простой способ удалять элементы из списка, а не пересоздавать список каждый раз. После списков расположена панель с combo box для выбора продукта. Список продуктов будет загружаться из базы, и при этом пользователь может ввести свой продукт. При нажатии на клавишу выбранный продукт добавляется в список ингредиентов. Если продукта нет в базе, то, обладая должными правами, пользователь может добавить продукт в базу (рис \*\*\*). Далее идет панель для добавления элементов категории с помощью combo box и уже знакомым пользовательским элементом для просмотра списка элементов категории. Работает это так, что, когда пользователь выбирает категорию из combo box в пользовательский элемент загружается список из элементов этой категории, далее пользователь может выделить некоторые из них и нажать на кнопку добавления, после чего выделенные элементы категории появятся в списке элементов категорий. И в самом низу панели находится text box, введенное туда слово обязательно должно содержаться в описании или названии блюда.

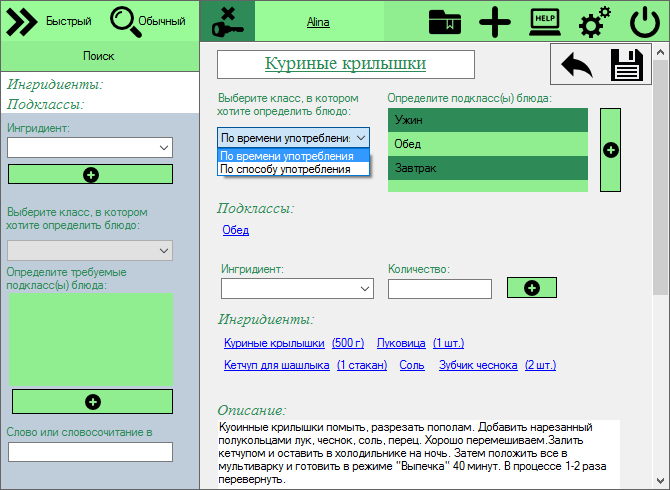


Первая страница главного tab control – это панель для просмотра списка рецептов. Рецепты на этой панели представлены в виде минимизированного окна (рис \*\*\*). Оно содержит название блюда, ингредиенты (показывает первые 6) и описание, которое вмещается в свободное пространство, отведенное под элемент.



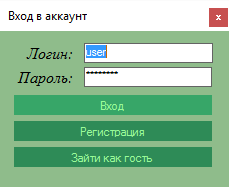
При нажатии на название текста для пользователя в главном tab control открывается страница просмотра блюда (рис \*\*\*). На данной странице отображаются все атрибуты блюда: название, ингредиенты, описание, картинку и все категории, к которым блюдо принадлежит. По мимо атрибутов блюда, на данной странице содержится меню управления с кнопками: «назад», «добавить/удалить закладку», «редактирование» и «удаление». Данные элементы доступны только тогда, когда пользователь имеет права на их использование (рис \*\*\*). Например, пользователь не увидит кнопки «удалить», если он не доверенный или же в один момент времени можно увидеть только одну из двух кнопок редактирования закладок добавление/удаление. Чтобы увидеть следующую страницу главного tab control нужно нажать на кнопку редактирования в вышеописанной странице либо нажать на кнопку «добавить» на главной управляющей панели.

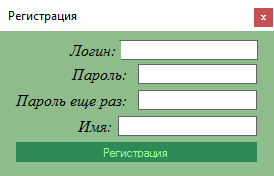
Страница редактирования (рис \*\*\*) очень схожа со страницей просмотра, только на данной странице текстовые поля редактируемые, а также добавлены две панели для добавления ингредиентов и элементов категории. Действие этих панелей аналогично действию панелей в детальном поиске. Меню управления на данной странице состоит лишь из двух элементов: «назад» и «сохранить». Оставшаяся страница – справка, ее можно увидеть на рис \*\*\*. На ней содержится инструкция по использованию, а также информация о приложении.



Главную панель управления приложением можно разбить на две чести: левую и правую. Справа находятся управляющие кнопки, функционал которых описан ранее (рис \*\*\*). Некоторые из них скрываются, если у пользователя недостаточно прав. Слева находится две кнопки и текстовое поле. Если пользователь не авторизирован, то в текстовом поле написано «Гость» и видна кнопка «зайти в аккаунт», иначе в текстовом поле находится имя пользователя, а рядом кнопка «выйти из аккаунта».

При нажатии на кнопку «войти в аккаунт» для пользователя открывается модальное диалоговое окно (рис \*\*\*), в котором он сможет ввести свои логин и пароль, вернуться в главное меню или открыть окно регистрации (рис \*\*\*). При открытии окна регистрации окно входа в систему не закрывается, а скрывается, тогда как закрытии окна регистрации окно входа снова становится видимым.





4.2 Разработка логики приложения

После создания каркаса приложения его стоит наполнить функционалом. Это легко осуществить с помощью привязки определенных методов к элементам интерфейса. Но перед началом создания логики, в приложение стоит ввести основные объекты, которые в дальнейшем будут часто использованы. Это позволит улучшить переносимость приложения, а также ряд других факторов, таких, например, как читабельность кода.

Из всех объектов следует выделить:

* Пользователь
* Рецепт приготовления
* Ингредиент
* Продукт
* Категория блюда
* Элемент категории
* Класс с данными для поиска

Все данные объекты больше схожи на структуры и имеют лишь поля атрибутов и конструктор, но при этом некоторые из них могу содержат другие, как это видно на рис \*\*\*.

В соответствии с интерфейсом приложения, функциональные требования из ТЗ были разбиты на следующие методы:

Таблица 8 – Методы класса MainForm

|  |  |
| --- | --- |
| Название метода | Описание |
| RefreshLocalData | Обновляет из базы локальные переменные: список всех продуктов в базе, информацию о пользователе, классах, блюдах. Используется почти во всех методах, где осуществляется работа с данными базы. |
| FillPanel(List<DishClass>) | Заполняет страницу просмотра списка блюд, принимая в качестве параметра этот самый список. Вызывается из метода начала поиска. |
| OpenDish | Открывает страницу просмотра рецепта и заполняет данными из локальной переменной dish. В данную переменную заносится информация о рецепте при нажатии на название рецепта в просмотре списка рецептов. |
| EditDish | Открываетя при редактировании или создании блюда. При редактировании поля панели заполняются данными из локальной переменной dish, а при создании остаются пустыми. |
| StartSearchBtn\_Click | В зависимости от выбранной страницы поиска запускает быстрый или детальный поиск. Заполняет класс SearchData данными о поиске, отсеивает неподходящие блюда и выводит подходящие с помощью метода FillPanel |
| SaveDishBtn\_Click | Создает или изменяет рецепт блюда на основе элементов из страницы редактирования рецепта. |
| editAddIngBtn\_Click,  addIngToDSBtn\_Click | Методы для добавления ингредиентов, которые позволяют добавлять в базу новые продукты. |
| loginBtn\_Click | Открывает модальное окно логирования. |
| logoutBtn\_Click | Выход из текущего аккаунта пользователя. |

Таблица 9 – Методы форм и пользовательских элементов

|  |  |  |
| --- | --- | --- |
| Название формы или пользовательского элемента | Название метода | Описание |
| loginForm | loginBtn\_Click | Проверяет наличие в базе пользователя по логину и паролю. |
| registrationForm | registrationBtn\_Click | Составляет объект User на основе текстовых полей формы и регистрирует пользователя в базе. |
| MinDishForm | MinDishForm(MainForm, DishClass) | Заполняет элементы формы на основе полученного в форму класса рецепта. |
| MinDishForm | nameLinkBtn\_Click | Открывает в главном окне блюдо, которое описанное в данной форме. |
| DishClassLBUC | SetList(DishCategoryClass) | Заполняет пользовательский элемент элементами категории, переданной в параметрах. |
| DishClassLBUC | List<SubcategoryClass> GetSelectedList | Возвращает список элементов категории, выделенных в пользовательском элементе. |

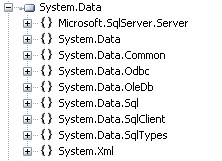
* 1. Проектирование взаимодействия с БД

Учитывая потребности приложения было бы оптимально созданы класс для работы с базой данных. Данный класс позволит обращаться к базе не из методов форм приложения, а через вызов его методов. Это убережет от повторения кода, а также позволит отделить логику работы с данными от логики приложения.

Для работы с базой данных в .NET Framework есть замечательная библиотека ADO.NET. Библиотеки ADO.NET можно применять тремя концептуально различными способами: в подключенном режиме, в автономном режиме и с помощью технологии Entity Framework.

В рамках курсовой работы был использован подключенный уровень работы с БД. При использовании подключенного уровня (connected layer), кодовая база явно подключается к соответствующему хранилищу данных и отключается от него. При таком способе использования ADO.NET обычно происходит взаимодействие с хранилищем данных с помощью объектов подключения, объектов команд и объектов чтения данных.

С точки зрения программиста, тело ADO.NET составляет базовая сборка с именем System.Data.dll. В этом двоичном файле находится значительное количество пространств имен, многие из которых представляют типы конкретного поставщика данных ADO.NET (рис \*\*\*).



Для работы MS SQL в ADO.NET содержит System.Data.SqlClient, это пространство имен содержит классы, используемые для подключения к базе данных Microsoft SQL Server, в том числе SqlDbCommand, SqlDbConnection и SqlDbDataAdapter. Эти классы оптимизированы для использования интерфейса TDS к SQL Server.

Таблица 10 - Основные объекты поставщиков данных ADO.NET.

|  |  |  |  |
| --- | --- | --- | --- |
| Тип объекта | Базовый класс | Соответствующие интерфейсы | Назначение |
| Connection | DbConnection | IDbConnection | Позволяет подключаться к хранилищу данных и отключаться от него. Кроме того, объекты подключения обеспечивают доступ к соответствующим объектам транзакций |
| Command | DbCommand | IDbCommand | Представляет SQL-запрос или хранимую процедуру. Кроме того, объекты команд предоставляют доступ к объекту чтения данных конкретного поставщика данных |
| DataReader | DbDataReader | IDataReader, IDataRecord | Предоставляет доступ к данным только для чтения в прямом направлении с помощью курсора на стороне сервера |
| DataAdapter | DbDataAdapter | IDataAdapter, IDbDataAdapter | Пересылает наборы данных из хранилища данных к вызывающему процессу и обратно. Адаптеры данных содержат подключение и набор из четырех внутренних объектов команд для выборки, вставки, изменения и удаления информации в хранилище данных |
| Parameter | DbParameter | IDataParameter, IDbDataParameter | Представляет именованный параметр в параметризованном запросе |
| Transaction | DbTransaction | IDbTransaction | Инкапсулирует транзакцию в базе данных |

Конкретные имена этих основных классов различаются у различных поставщиков (например, SqlConnection, OracleConnection, OdbcConnection и MySqlConnection), но все эти объекты порождены от одного и того же базового класса (в случае объектов подключения это DbConnection), который реализует идентичные интерфейсы (вроде IDbConnection).

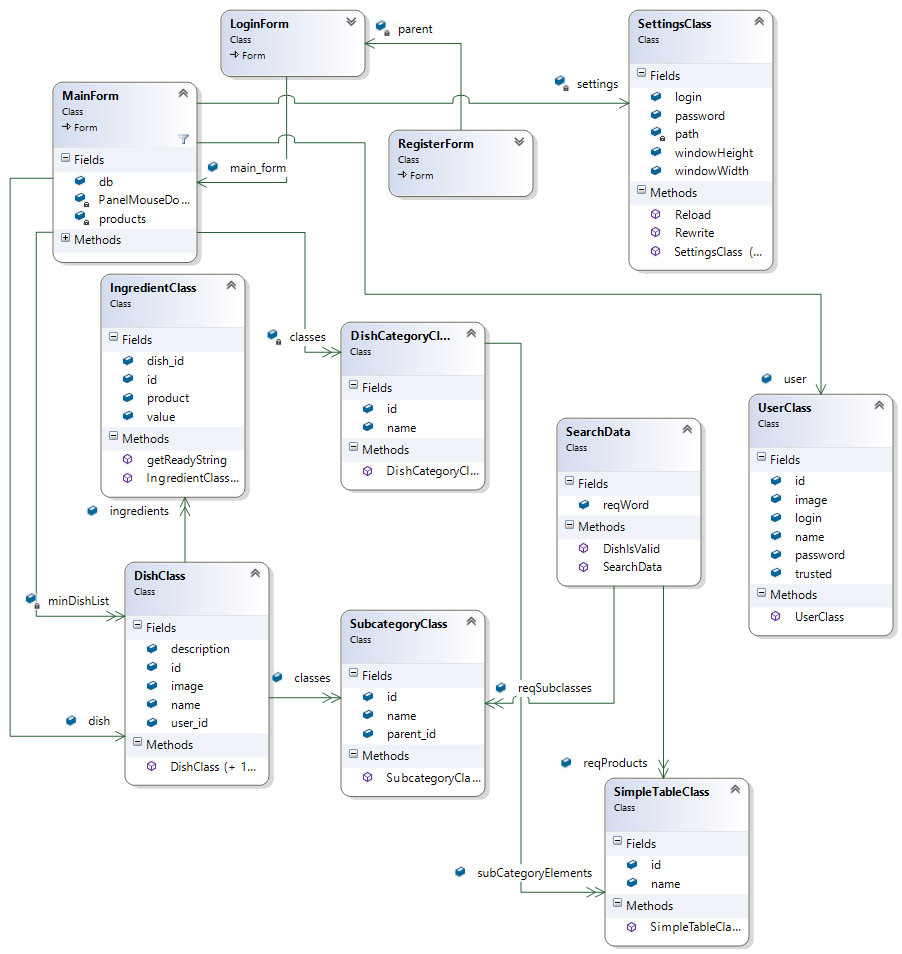
Разработанный класс для работы с БД содержит множество методов, которые по функциональному назначению можно разбить на несколько типов:

Таблица 11 – Методы класса DBWorker

|  |  |  |
| --- | --- | --- |
| Тип функции | Экземпляры функций | Функционал данного типа функций |
| Для работы с соединением | Сonnect,Close,IsOpen,  RefreshConnect. | Все методы, необходимые для управления соединением. |
| Для возвращения объектов | GetSubCategoryById(int),  GetUserByLoginPassword(string,string),  GetSimpleClassById(int),  GetProductIdByName(string) и т.д. | Методы, возвращающие один объект по какому-либо параметру, характеризующему таблицу в базе. |
| Для возвращения списков объектов | GetAllDishClasses(),  GetAllSubCategoryByCategoryId(int),  GetSubCategoryById(int),  GetAllDishIdFromUserBookmarks(int) и т.д. | Методы, возвращающие списки объектов. |
| Для удаления объектов | DeleteAllDishIngredientsByDishId(int),  DeleteAllDishSubclassesByDihId(int),  DeleteTableById(int),  DeleteBookmark(int, int). | Удаление объектов. Класс DeleteTableById – универсальный для всех таблиц, имеющих атрибут id. |
| Для создания объектов | AddSubclass(int, int),  AddIngredient(IngredientClass),  AddUser(UserClass),  AddBookmark(int, int) и т.д. | Добавление экземпляров таблиц в базу данных. В большинстве случаем эти методы принимают класс того объекта, который представлять данную таблицу. |
| Для изменения объектов | UpdateDish(Dish) | Обновление существующего экземпляра таблицы. |
| Для проверки наличия существования объекта | UserExist(string) |  |

**5 Описание программного продукта.**

5.1 Описание объектов и их взаимодействия.



На рис \*\*\* изображены основные классы и их взаимодействие. Как можно заметить – связь есть, как и между элементарными классами, представляющими объекты БД, так и между классами формы приложения, которые буферизируют классы из базы данных, позволяя переносить эти объекты между методами. Это один из способов связи в проекте. Так же видно, что

Так же в приложении есть класс для работы с базой данных, связи этого класса с формами приложения изображены на рис \*\*\*.



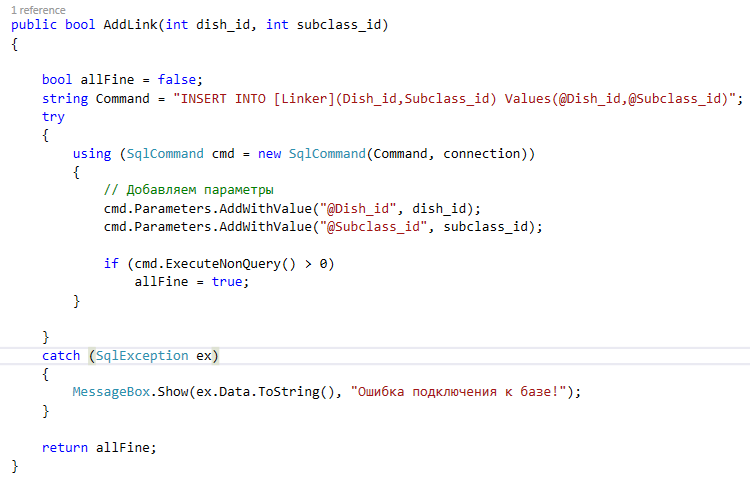
**5.2 Описание Sql-запросов**

Все sql-запросы приложения содержаться в методах класса DBWorker и строятся данные запросы с помощью стандартного метода string.Format(string, agrs\*) Некоторые из них можно увидеть в таб \*\*\*.

Таблица 12 – sql-запросы, существующие в методах класса DBWorker

|  |  |
| --- | --- |
| Метод | Выражение для создания Sql-запроса |
| UserExist(string login) | string.Format("SELECT \* FROM [User] WHERE Login ='{0}'", login) |
| GetDishById(int id) | string.Format("SELECT \* FROM [Dish] WHERE Id ='{0}'", id) |
| GetDishCatigoriesByDishId(int id) | string.Format("SELECT \* FROM [Linker] where Dish\_id = '{0}'",id) |
| GetAllMinDishs() | string.Format("SELECT \* FROM [Dish]"); |
| UpdateDish(DishClass dish) | string.Format("UPDATE [Dish] set Name=@Name,Description=@Description, Image=@Image where id='{0}'", dish.id.ToString()) |
| AddDish(DishClass dish) | "INSERT INTO [Dish](Description,Name,User\_id,Image) Values(@Description,@Name,@User\_id,@Image)" |
| DeleteAllDishIngredientsByDishId(int id) | string.Format("Delete from [Ingredient] where Dish\_id = '{0}'", id) |
| DeleteBookmark(int user\_id, int dish\_id) | string.Format("Delete from [Bookmark] where User\_id = '{0}' and Dish\_id = '{1}'", user\_id,dish\_id) |

Созданные команды выполняются с помощью класса SqlCommand. Команда вместе с открытым подключением посылаются в конструктор этого класса, а затем выполняется с помощью одного из следующих методов: ExecuteReader или ExecuteNonQuery. Первый возвращает объект, с помощью которого производится считывание данных из БД, а второй возвращает количество элементов БД, затронутых выполнимой командой. При выполнении операций INSERT или UPDATE следует так же задать параметры, передаваемые в базу (рис \*\*\*). Пример работы c ADO.NET (рис \*\*\*):



**6 Тестирование и внедрение**

6.1Тестирование качества ПО

Назначение, область применения и требования к надежности программного продукта:

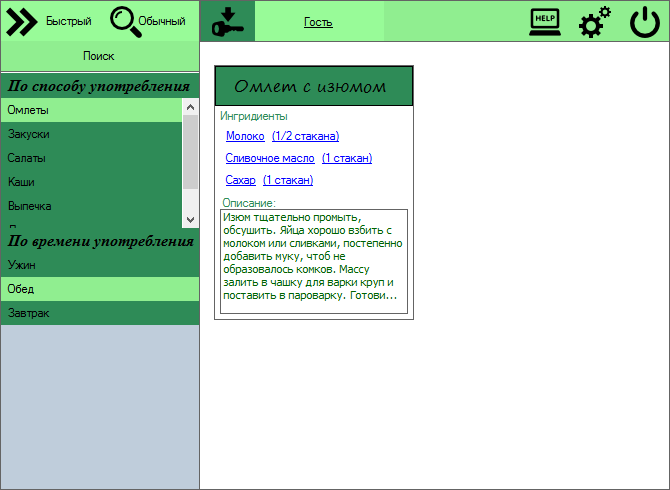
* Назначение: предоставление справки о рецептах
* Область применения: поиск информации
* Требования к надежности: Скорость и стабильность работы

Таблица - Анализ надежности программного продукта

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № Теста | Данные для теста  (вводимая информация) | Время функционирования программы | Результат (успешно\отказ) | Результат |
|  | Запуск приложения | 1-2 секунды | успешно | Открытие главной формы приложения |
|  | Попытка добавить несуществующий продукт в поиск без прав доверенного пользователя | Менее 1 секунды | успешно | Предупреждение о том, что элемента нет в базе |
|  | Попытка добавить несуществующий продукт в поиск с правами доверенного пользователя | Менее 1 секунды | успешно | Предложение добавить элемент в базу |
|  | Нажатие на кнопку выхода из аккаунта | Меньше секунды | успешно | Очищение локальной переменной пользователя, обновление элементов интерфейса |
|  | Нажатие на кнопку входа | Меньше секунды | успешно | Вызов окна авторизации |
|  | Ввод корректных данных для входа | Меньше секунды | успешно | Авторизация в приложении |
|  | Ввод некорректных данных для входа | Моментально | успешно | Предупреждение о неверном вводе данных |
|  | Ввод корректных данных для регистрации | Меньше секунды | успешно | Регистрация в базе, предупреждение перенаправление пользователя к окну входа. |
|  | Попытка создать блюдо, с уже существующим названием | Меньше секунды | успешно | Предупреждение об не оригинальности названия |

Как можно заметить из таблицы \*\*\* - приложение работает быстро и надежно. В надожность осуществлена с помощью динамичных панелей, которые исчезают, когда у пользователя нет прав на их использования и т.д. Данный шаг уменьшает количество возможных ошибок в приложении, а так же упрощает интерфейс пользователя.

1. Руководство пользователя
   1. Использование продукта



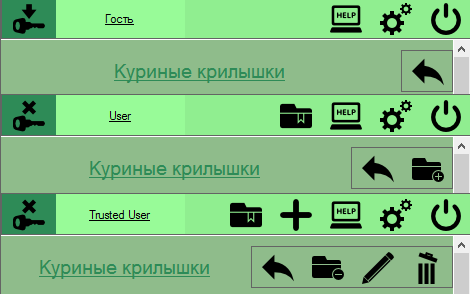
Данное приложение является электронным кулинарным справочником. По мимо просмотра и поиска блюд, данный кулинарный справочник позволяет персонализировать данные операции, а также делиться своими рецептами с другими пользователями. Так же, справочник позволяет использовать сразу два вида поиска: быстрый и детальный. Для уточнения действия какого-либо элемента управления – наведите на него курсором и подождите пол секунды. Далее данные функции будут рассмотрены более детально.

Пользователи и доступ

В данном приложении есть три типа пользователей:

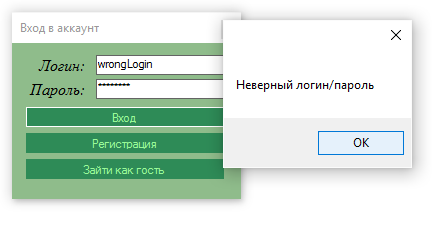
1. Гость. Гость может искать и изучать рецепты.
2. Зарегистрированный пользователь. Он может сохранять блюда в закладки.
3. Доверенный пользователь. Данный пользователь может создавать и редактировать свои рецепты.

Каждый последующий имеет и возможности всех предыдущих.

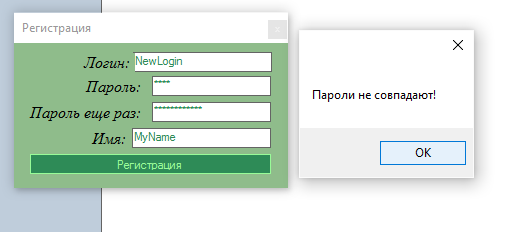


Вход на существующий аккаунт

Для входа в аккаунт надо нажать на кнопку входа и верно ввести в поля логин и пароль свои данные. При неверном вводе вас уведомят об ошибке. Если по у вас нет аккаунта или по какой-то причине не можете зайти на свой, то можете зарегистрировать новый аккаунт.



Регистрация



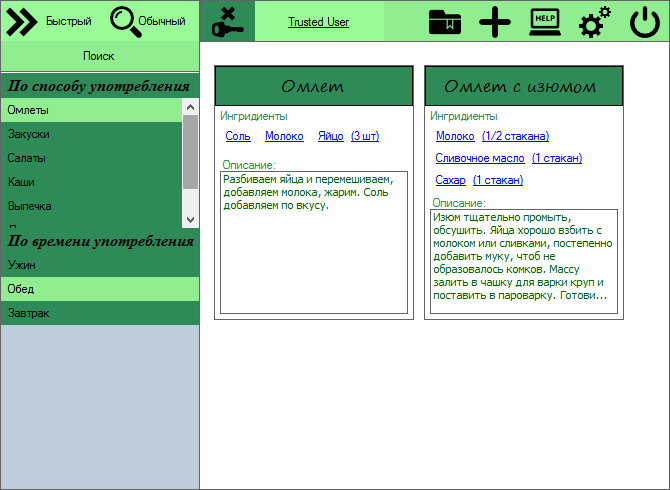
Для регистрации следует перейти из главного меню в меню входа в аккаунт, а в нем нажать на кнопку «Регистрация», после чего вам будет предложено зарегистрироваться. Для успешной регистрации стоит:

* Ввести уникальный логин.
* Заполнить все поля.
* Убедится, что пароль длиннее 8 символов.
* Убедится, что пароли в элементах «Пароль» и «Пароль еще раз» совпадают.

После нажатия клавиши при успешной регистрации вы будете перенаправлены назад в окно входа в аккаунт, либо будете предупреждены о проблемах в приложении.

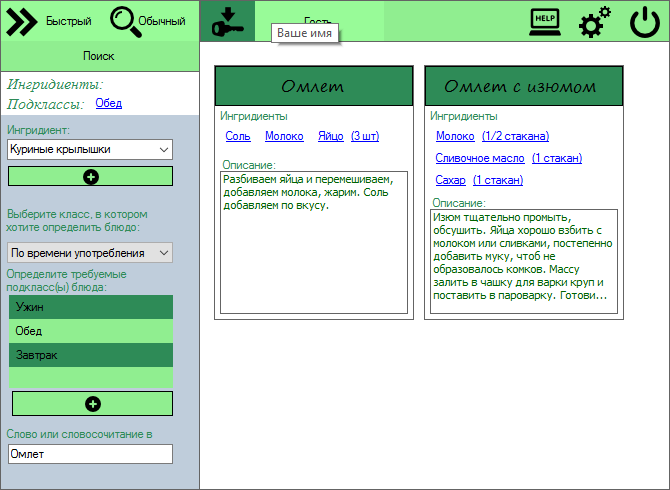
Быстрый поиск блюд:

1. Откройте вкладку быстрого поиска
2. Выберите интересующие категории блюд
3. Нажмите поиск

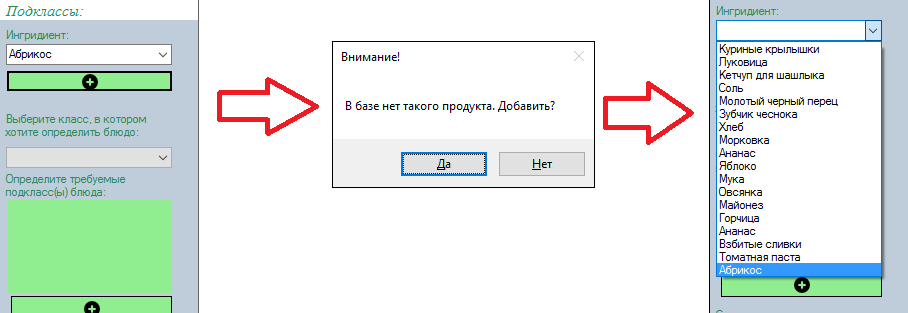


Детальный поиск блюд:

1. Откройте вкладку детального поиска
2. Выберите нужные вам критерии поиска
3. Нажмите на кнопку поиска



Если вам нужно найти рецепты с упоминаниями некоторого слова или словосочетания, то введите его в текстовом элементе внизу формы. При желании отсеять блюда по категориям – воспользуйтесь панелью добавления категорий: выберите интересующую вас классификацию блюд, щелчком мыши выделите нужные категории после чего нажмите на кнопку добавления категорий. Когда нужно учитывать и наличие определенных продуктов, тогда следует добавить продукты для поиска с помощью элемента добавления продукта. Введите название продукта. Если он есть в базе, то нужный вариант можно будет выбрать и с помощью авто заполнения. Если продукта в базе нет, то при наличии специальных прав, вы можете добавить продукт в базу, но в этом нет смысла, так как если продукта в базе нет, то и рецептов с ним тоже.

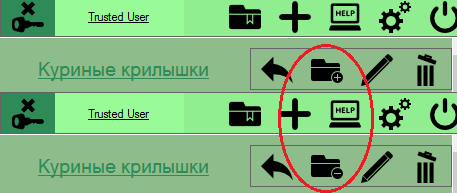


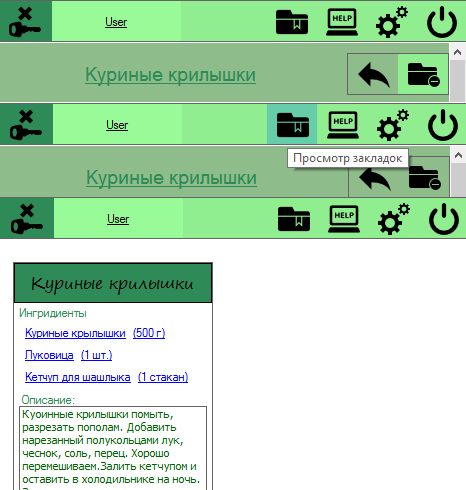
Просмотр списка рецептов

На данную вкладку главного окна можно попасть из закладок или при поиске. Рецепт в минимальном представлении содержит название, ингредиенты и часть описания, которая поместилась в элементе. При нажатии на название блюда, данное открывается в окне просмотра блюд.

Работа с закладками

Работа с закладками очень проста и понятна. Для этого существует всего три кнопки: добавления, удаления и просмотра. При нажатии на первую блюдо добавляется в список закладок, при нажатии на удаление – удаляется. Для того, чтоб просмотреть все свои закладки нужно нажать на кнопку закладок, расположенную на главной панели управления.

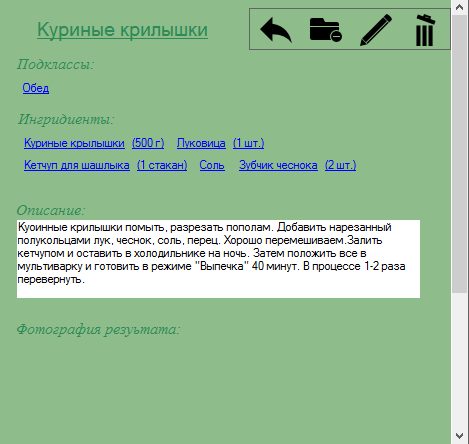




Просмотр рецепта

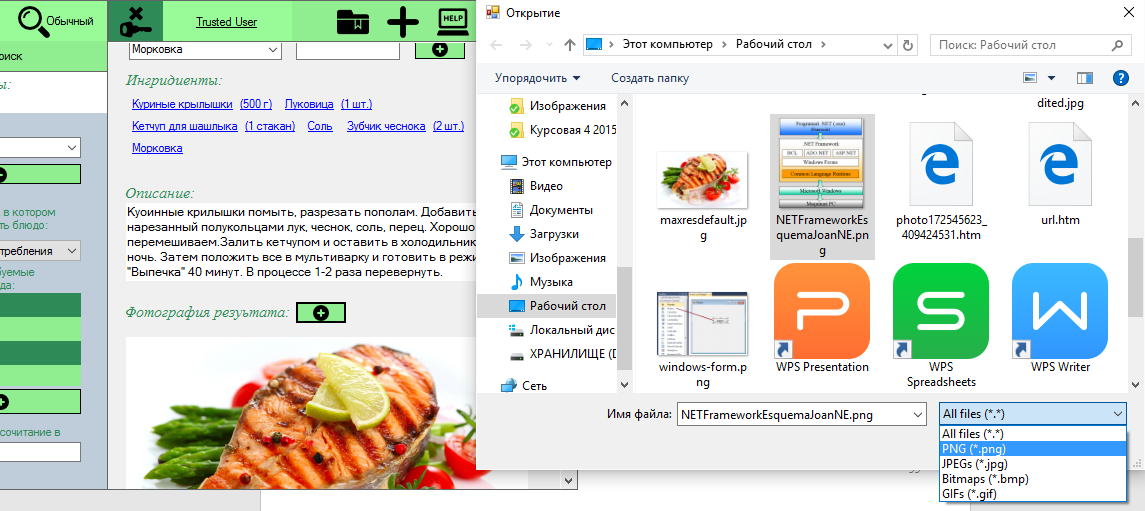
В данном окне видна вся информация о рецепте, а на панели управления расположены следующие кнопки:

* «Назад». Возвращает в предыдущее окно.
* «Добавить в закладки» / «Удалить закладку». В конкретный момент времени отображается или одни из кнопок, или ни одной.
* «Редактировать». Открывает окно редактирования данного рецепта.
* «Удалить». Удаляет рецепт.

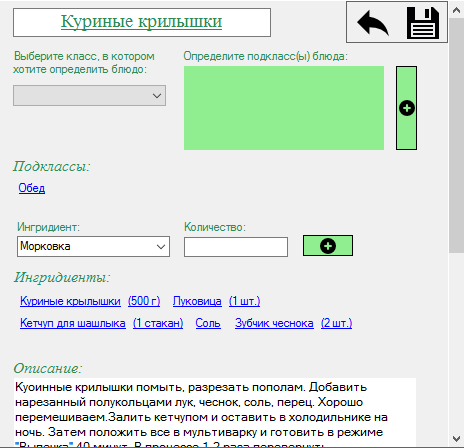


Редактирование и создание рецепта

Очень схожое с окном просмотра блюда, только содержит дополнительные кнопки и редактируемые элементы. Процесс добавления категорий аналогичен тому, что в детальном поиске, а новый процесс добавления ингредиента отличается от добавления продукта тем, что ингредиент состоит из продукта и значения (словесное описание количества, например, «2 яйца», «3 ложки сахара», «щепотка приправы» и т.д.). Так же есть кнопка выбора фотографии блюда, которая вызывает файловый диалог для выбора фото формата .png, но вы можете выбрать фото с другим расширением, выбрав тип документа \*\*\* на «Аll files».



В этом окне видны только две кнопки: возврата и сохранения. Кнопка возврата возвращает к просмотру редактируемого блюда или на страницу помощи, если блюдо не редактировалось, а создавалось. Кнопка «Сохранить» действует в зависимости от того, как вы попали в окно редактирования. Если вы попали в это окно при создавали блюда, то нажатие на вышеупомянутую кнопку создаст новое блюдо, иначе изменит тот рецепт, с которого вы перешли в редактирование.

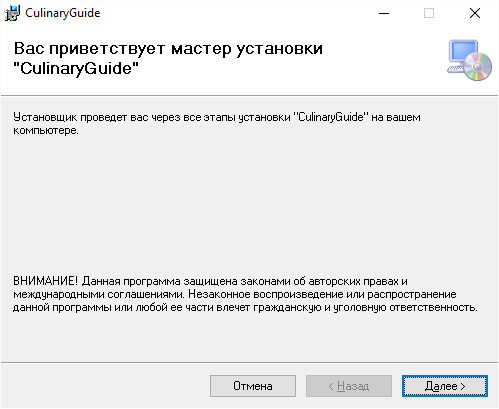


Настройки

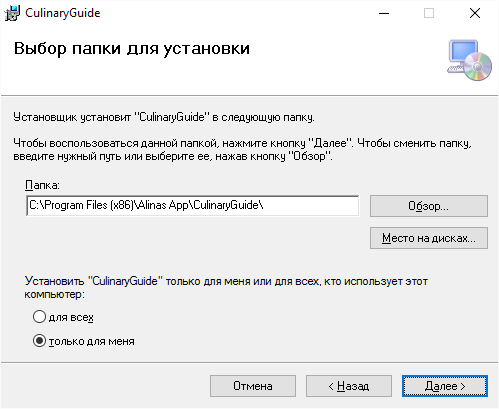
Если вас не устраивает расширение главного окна приложения, то вы можете нажать на кнопку настроек, появится модальное окно настроек. После выбора разрешения нажмите «Ок», после чего размер приложения изменится.

* 1. Инструкция по установке

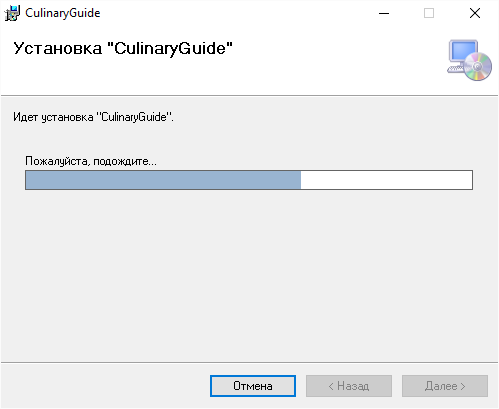
1. Запустите файл CulinaryGuideInstaller.msi от имени администратора, нажмите «Далее» в появившемся окне диалога (рис \*\*\*).



1. Выберите папку для установки и пользователя, для которого установите программу (рис \*\*\*) и нажмите «далее».



1. Нажмите «далее» в подтверждении установки.
2. Дождитесь установки программы и подтвердите завершение установки (рис \*\*\*).



* 1. Системные и программные средства

Необходимые минимальные системные требования:

* Процессор – Pentium 4, Celeron и выше
* Оперативная память – 1 GB и выше
* Видеокарта с памятью 256 М и выше
* Свободно место на диске минимум 500 kb

Необходимые программные требования:

* Операционная система Windows XP/ Windows7/8/10
* .NET framework
* \*\*\*

**ЗАКЛЮЧЕНИЕ**

В результате работы над курсовой работой выполнены все поставленные задачи. Разработано рабочее приложение «Кулинарный справочник». Был изучен ряд технологий для создания проекта: .NET, Windows Forms, C#, Visual Studio 2015, MS SQL. Также было написано обоснование выбора данных технологий в курсовом проекте. Функционал приложения был утвержден согласно разработанной структуре программы и базы данных. На основе разработанной функциональной схемы проекта составлен программный продукт, обладающий интерфейсной частью и базой данных для хранения данных о пользователях, блюдах, категорий блюд, закладках и продуктах.

На данный момент при работе с приложением доступны следующие возможности: регистрация, авторизация пользователей, просмотр блюд, возможность изменения существующих рецептов и создания новых так же есть возможность сохранения блюд в закладки, а также два вида поиска блюд.

Дальнейшее развитие программы связано с расширением ее возможностей, улучшением уровня безопасности, подключением к удаленной БД. Тестирование программы показало ее работоспособность.