**СОДЕРЖАНИЕ**

*Изм.*

*Лист*

*№ докум.*

*Подпись*

*Дата*

*Лист*

*У 3*

КП Т.295013.401

*Разраб.*

*Малиновская М.Д*

*Провер.*

*Розовская О.К*

*Т. Контр.*

*Н. Контр.*

*Утверд.*

*Лит.*

*Листов*

*КБП*

[Введение 3](#_Toc185963507)

[1 Описаниезадачи 4](#_Toc185963508)

[1.1 Анализ предметной области 4](#_Toc185963509)

[1.2 Постановка задачи 4](#_Toc185963510)

[2 Проектирование веб-приложения 6](#_Toc185963511)

[2.1 Проектирование модели 6](#_Toc185963512)

[2.2 Требования к веб-приложению 7](#_Toc185963513)

[2.3 Структура веб-приложения 8](#_Toc185963514)

[2.4 Проектирование макета веб-приложения 9](#_Toc185963515)

[2.5 Программно-технические средства, необходимые для разработки приложения 11](#_Toc185963516)

[2.6 Защита и сохранность данных 12](#_Toc185963517)

[3Реализация веб-приложения 15](#_Toc185963519)

[3.1 Описание разделов веб-приложения 15](#_Toc185963520)

[3.3 Разработка клиентской части приложения 17](#_Toc185963522)

[3.4 Описание используемых функций и процедур 18](#_Toc185963523)

[3.5 Функциональное тестирование 19](#_Toc185963524)

[4 Применение 25](#_Toc185963525)

[4.1 Назначение веб-приложения 25](#_Toc185963526)

[4.2 Программно-аппаратное обеспечение сервера и клиента 25](#_Toc185963527)

[Заключение 26](#_Toc185963528)

[Список информационных источников 27](#_Toc185963529)

[ПРИЛОЖЕНИЕ А 28](#_Toc185963530)

[ПРИЛОЖЕНИЕ Б 32](#_Toc185963531)

# Введение

Одной из тенденций современного общества является стремление к саморазвитию и получению новых навыков, в том числе в кулинарии. Популярность кулинарных мастер-классов растет, но часто запись на них организована неудобно, через телефонные звонки или сообщения в социальных сетях. Автоматизация этого процесса с помощью веб-приложения позволит решить ряд проблем как для организаторов, так и для участников.

Веб-приложение "Запись на кулинарные мастер-классы" предоставит удобный интерфейс для просмотра расписания, выбора интересующего мастер-класса, получения подробной информации о нем, а также быстрой и простой записи. Приложение будет актуальным, поскольку оно упрощает и делает прозрачным процесс записи на мастер-классы, что актуально в условиях растущего интереса к кулинарии.

# Описание задачи

## Анализ предметной области

Предметной областью поставленной задачи является организация и автоматизация процесса записи на кулинарные мастер-классы. Объектом решаемой задачи является разработка информационной системы (веб-приложения), которая обеспечит удобное взаимодействие между организаторами мастер-классов и потенциальными участниками, а также автоматизирует основные процессы, связанные с выбором, бронированием и управлением посещениями.

Кулинарные мастер-классы, проводятся профессиональными поварами и включают в себя обучение приготовлению различных блюд и техник. Организация таких мероприятий включает в себя:

• Планирование расписания: Определение тем, дат, времени и продолжительности мастер-классов.

• Управление информацией о мастер-классах: Создание подробных описаний курсов, используемых ингредиентах и стоимости.

• Коммуникация: Информирование участников о деталях мастер-класса.

Процесс взаимодействия с потенциальным участником начинается с просмотра расписания доступных мастер-классов. Пользователь может ознакомиться с подробным описанием каждого из них, включая темы, продолжительность, преподавателя и стоимость.

Для записи на мастер-класс пользователь должен пройти регистрацию, используя email и пароль, или авторизоваться, если учетная запись уже существует. В профиле предоставляется возможность просмотра информации о всех своих записях, возможность редактирования личной информации

Таким образом, веб-приложение обеспечивает:

• Удобный интерфейс для просмотра расписания и информации о мастер-классах.

• Возможность регистрации и авторизации для пользователей.

• Простую и быструю процедуру записи на мастер-классы.

• Возможность интеграции с платежными системами.

# Проектирование веб-приложения

Предметной областью базы данных является организация и управление кулинарными мастер-классами, а именно процесс записи на них.

Целью создания базы данных является автоматизация процесса записи на мастер-классы, управления расписанием и взаимодействием с пользователями.

Назначение проектируемой базы:

• Хранение информации о мастер-классах (тема, дата, время, описание, стоимость, преподаватель, количество мест).

• Хранение информации о пользователях (участниках и организаторах).

• Хранение информации о записях на мастер-классы.

• Обновление, добавление и удаление информации о мастер-классах и записях.

• Анализ посещаемости мастер-классов, популярности тем и преподавателей.

• Управление расписанием и доступностью мест.

Пользователи БД:

1. Организатор (Администратор):

• Обладает правами на добавление, редактирование и удаление информации о мастер-классах (создание новых мастер-классов, изменение расписания, цен, описаний и т.д.).

• Имеет доступ к информации о зарегистрированных участниках на каждый мастер-класс.

• Может просматривать и анализировать статистику посещаемости и популярности курсов.

• Имеет возможность отправлять уведомления участникам.

• Может управлять профилем преподавателя.

2. Участник (Клиент):

• Имеет возможность просматривать расписание доступных мастер-классов.

• Может просматривать подробную информацию о каждом мастер-классе (тема, описание, преподаватель, стоимость, количество мест).

• Имеет возможность зарегистрироваться в системе и записаться на выбранные мастер-классы.

• Может отменить свою запись на мастер-класс.

• Может просматривать историю своих записей и управлять своими уведомлениями.

• Имеет возможность оставить отзыв о посещенном мастер-классе (опционально).

Методология IDEF0

Для анализа функциональности системы можно использовать методологию IDEF0 (Integrated DEFinition). Она позволит описать функциональную структуру приложения, определить основные процессы и взаимодействия между ними. Например, можно выделить такие функции, как "Управление мастер-классами", "Регистрация пользователей", "Запись на мастер-класс", "Просмотр расписания".

Методология IDEF3

Для детального описания рабочих процессов подойдет методология IDEF3. Она поможет визуализировать последовательность действий при выполнении задач, например, процесс записи пользователя на мастер-класс: просмотр расписания -> выбор мастер-класса -> регистрация/авторизация -> запись -> подтверждение. Также можно описать процессы добавления нового мастер-класса, отмены записи и т.д.

Диаграммы потоков данных (DFD)

Диаграммы потоков данных (DFD) будут полезны для описания перемещения информации между пользователями, приложением и базой данных. DFD позволит показать поток данных при добавлении нового мастер-класса (ввод данных организатором -> сохранение в БД) или при записи пользователя на мастер-класс (выбор мастер-класса -> отправка запроса на запись -> сохранение в БД -> подтверждение).

## Проектирование модели

Главной целью проектирования моделей является отображение функциональной структуры объекта, то есть производимые ими действия и связи между этими действиями.

Диаграмма вариантов использования представляет собой визуальное описание основных действий и взаимодействий между актерами (пользователями) и системой (веб-приложением). Включает варианты использования, их названия, описание, связи между ними и актерами. Диаграмма вариантов использования представлена в графической части на листе 1.

Диаграмма деятельности является одной из UML-диаграмм и представляет собой графическое представление действий, состояния которых описаны на диаграмме состояний. Она используется для спецификации исполняемого поведения системы, представляя координированное последовательное и параллельное выполнение подчиненных элементов, таких как вложенные виды деятельности и отдельные действия, которые связаны между собой потоками, идущими от выходов одного узла к входам другого. Диаграмма деятельности представлена в графической части на листе 2.

Диаграмма классов показывает структуру классов и связи между ними в системе. Включает классы, их атрибуты и методы, а также связи между классами (ассоциация, наследование, зависимость, реализация, композиция и агрегация). Иллюстрирует модель данных и основные компоненты системы. Диаграмма классов представлена в графической части на листе 3.

Диаграмма последовательности иллюстрирует динамическое поведение проектируемой системы, показывая, как объекты взаимодействуют друг с другом через обмен сообщениями во времени для выполнения определенного сценария. Диаграмма фокусируется на порядке взаимодействий, предоставляя визуальное представление потока управления между различными компонентами системы. Диаграмма последовательности помогает понять, как система выполняет конкретные сценарии использования, выявляя последовательность взаимодействий и временные зависимости между ними. Это полезно для анализа и проектирования процессов, отладки и документирования сложных систем. Диаграмма последовательности представлена в графической части на листе 4.

Диаграмма структуры сайта представляет собой визуальное описание организации и взаимосвязей различных страниц и разделов веб-сайта. Включает в себя основные страницы сайта, их названия, иерархию, а также связи между страницами, такие как гиперссылки и навигационные элементы. Диаграмма структуры сайта помогает понять архитектуру сайта, навигационные пути и взаимосвязь контента, что упрощает разработку и улучшение пользовательского интерфейса. Диаграмма структуры сайта представлена в графической части на листе 5.

Диаграмма развертывания является одной из UML-диаграмм и представляет собой графическое отображение физического расположения различных компонентов системы на аппаратных узлах. Она включает узлы, представляющие собой аппаратные устройства или виртуальные машины, а также артефакты, которые размещены на этих узлах, такие как исполняемые файлы, библиотеки и базы данных. Диаграмма развертывания иллюстрирует, как программные компоненты распределены по различным аппаратным платформам, показывая их взаимодействие и сетевые связи. Это помогает в планировании инфраструктуры, анализе производительности и обеспечении безопасности системы. Диаграмма развертывания представлена в графической части на листе 6.

## Требования к веб-приложению

Для обеспечения удобства и эффективности использования веб-приложения "Запись на кулинарные мастер-классы" были определены следующие требования, учитывающие как потребности пользователей, так и особенности предметной области:

Требования к дизайну:

• Интуитивность и простота: Веб-приложение должно обладать простым и понятным интерфейсом, не перегруженным лишними элементами. Навигация должна быть логичной и очевидной, позволяя пользователю легко находить нужную информацию и выполнять необходимые действия. При проектировании необходимо избегать сложных и запутанных элементов, которые могут запутать пользователя.

• Эстетика и стиль: Дизайн должен быть современным и привлекательным, способствующим приятному восприятию контента. Цветовая палитра должна быть сдержанной и гармоничной, создавая атмосферу доверия и профессионализма. Следует избегать использования кричащих, контрастных цветов, которые могут утомлять глаза.

• Цветовая гамма: Предполагается использование трех-четырех основных цветов и их оттенков. Основными цветами могут быть, например, светлый (белый или кремовый) для фона, темный (темно-серый или синий) для текста и элементов интерфейса, и акцентный цвет (например, оранжевый или бирюзовый) для выделения важных элементов, кнопок и ссылок. Цветовая схема должна быть подобрана таким образом, чтобы обеспечить хорошую читабельность и не вызывать раздражения.

• Типографика: Шрифт должен быть разборчивым и легко читаемым на любых устройствах. Предпочтение отдается шрифтам без засечек, которые воспринимаются как современные и минималистичные. Важно, чтобы цвет шрифта контрастировал с фоном, обеспечивая оптимальную читаемость. Размер шрифта должен быть адаптивным и меняться в зависимости от размера экрана устройства.

Требования к отображению:

• Кроссбраузерность: Веб-приложение должно корректно отображаться и функционировать во всех популярных браузерах (например, Chrome, Firefox, Safari, Edge) последних версий, обеспечивая единообразный пользовательский опыт на разных платформах.

• Адаптивность: Веб-приложение должно быть адаптивным и корректно отображаться на устройствах с различными размерами экранов (компьютеры, ноутбуки, планшеты, смартфоны). Контент должен масштабироваться, сохраняя при этом свою читабельность и функциональность. Компоновка элементов должна меняться в зависимости от размера экрана, чтобы обеспечить удобство просмотра.

• Производительность: Веб-приложение должно быстро загружаться и работать без задержек. Следует избегать использования тяжелых элементов и оптимизировать код, чтобы обеспечить хорошую производительность даже на устройствах с ограниченными ресурсами.

Требования к контенту:

• Релевантность: Содержание сайта должно соответствовать тематике кулинарных мастер-классов и предоставлять пользователям необходимую информацию (расписание, описание, цены, преподаватели).

• Доступность и ясность: Информация должна быть представлена в четком, кратком и понятном виде. Следует избегать использования сложной терминологии и жаргона.

• Актуальность: Контент должен регулярно обновляться, чтобы предоставлять пользователям самую актуальную информацию о расписании и доступных мастер-классах.

Требования к управлению контентом:

• Удобство администрирования: Система управления контентом (CMS) должна быть интуитивно понятной и простой в использовании для администраторов. Интерфейс CMS должен быть логичным и удобным для навигации, позволяя администраторам легко добавлять, редактировать и удалять контент.

• Разграничение прав доступа: Доступ к CMS должен быть ограничен и предоставляться только авторизованным администраторам.

• Безопасность: Система должна быть защищена от несанкционированного доступа и атак.

Требования к компоновке:

• Шапка сайта: В верхней части страницы должна располагаться шапка с логотипом. Шапка должна оставаться видимой.

• Структура страницы: Содержание страницы должно быть структурированным и логично организованным. Важные элементы должны быть расположены на видном месте.

• Адаптивность: Компоновка страниц должна автоматически адаптироваться к ширине экрана пользователя.

Эти требования создают удобное, функциональное и привлекательное веб-приложение, которое будет удовлетворять потребностям как организаторов, так и участников кулинарных мастер-классов.

## Структура веб-приложения

Ментальная карта – это графическое представление информации, которое показывает связи между различными элементами веб-приложения. Ментальная карта начинается с центрального узла (главной страницы) и далее разветвляется на подузлы (подразделы). Структура веб-приложения приведена на рисунке 2.1. На рисунке изображена ментальная карта, котороая отображает страницы веб-приложения.

Прототип веб-приложения – Это ранняя версия, созданная для визуализации и тестирования основных функциональностей и интерфейса.



Рисунок 2.1 – Ментальная карта

## Проектирование макета веб-приложения

Целью веб-приложения "SKZ4LOVER" является предоставление пользователям удобного и простого способа поиска, просмотра и записи на доступный мастер-класс. Дизайн веб-приложения должен быть минималистичным и функциональным, чтобы пользователи могли легко находить интересующие мастер-классы.

При запуске веб-приложения на экране пользователя отображается главное окно с мастер-классами . На главной странице представлены: список доступных мастер классов, вход и регистрация, а также профиль. Незарегистрированные пользователи могут искать и просматривать мастер классы, однако для добавления товаров в избранное и запись, необходимо выполнить авторизацию. Макет главной страницы представлен на рисунке 2.1.

Рисунок 2.1 - Макет страницы «Главная»

При переходе в глвную на экране пользователя отображаются карточки мастер-классов, содержащие название, а также кнопка перехода в профиль. Макет страницы каталога представлен на рисунке 2.4.



Рисунок 2.3 - Макет страницы «Главная»

На странице профиля пользователь может изменить информацию о себе, нажав на кнопку «редактировать профиль». А также может зайти в избранное и посмотреть, что там находится. Макет страницы профиля представлен на рисунке 2.4.

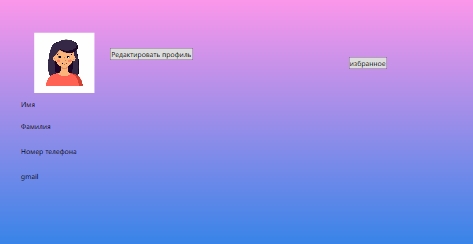


Рисунок 2.4 - Макет страницы «Профиль»

## Программно-технические средства, необходимые для разработки приложения

В ходе выполнения проекта для разработки программного средства были использованы следующие программные инструменты разработки:

* операционная система Windows 11;
* среда разработки Microsoft Visual Studio code;
* языки программирования C#;

Windows – операционная система, разработанная корпорацией Microsoft. Эта операционная система позволяет пользователю взаимодействовать с файлами, Интернетом, запускать различные программы, смотреть фильмы, слушать музыку, просматривать фото.

Основным инструментом разработки является среда разработки MS Visual Studio Code, которая имеет поддержку разработки очень большого количества языков, имеет удобный и приятный интерфейс.

C# — это современный, объектно-ориентированный язык программирования, разработанный компанией Microsoft. Он используется для создания разнообразных приложений, от десктопных программ и веб-сервисов до мобильных игр и приложений. C# является основным языком для разработки на платформе .NET.

## Защита и сохранность данных

База данных веб-приложения кулинарных мастер-классов хранит личную пользовательскую информацию, информацию о мастер-классах. Следовательно, веб-приложение обязано защищать эту информацию от несанкционированного доступа. Для этого применялось хеширование паролей пользователей, даже если кто-то сможет получить данные, предаваемые от сервера клиенту или от клиента к серверу, то у него будут только захешированные данные, из которых невозможно будет получить исходные данные.

Кроме этого, присутствует разделение пользователей по ролям, следовательно, у обычного пользователя не будет доступа к функциям добавления, удаления и редактирования информации.

Пользователи имеют возможность просматривать мастер-классы, добавлять их в избранное и записываться.

Такой подход гарантирует защиту данных и обеспечивает безопасное использование записи для клиентов.

# Реализация веб-приложения

## Описание разделов веб-приложения

При запуске веб-приложения интернет-магазина музыкальных инструментов и аксессуаров открывается приветственная страница. На этой странице пользователи могут войти в аккаунт, зарегистрироваться или ознакомиться с информацией о магазине и его товарах. Авторизация не является обязательной, и пользователи могут свободно перемещаться по сайту без входа в аккаунт.

В верхней части страницы находится шапка. Слева расположена боковая панель, обеспечивающая удобную навигацию по сайту.

В шапке размещены следующие элементы: логотип магазина, строка поиска, кнопки для перехода в избранное и корзину. При вводе символов в строку поиска автоматически отображается список товаров, в названиях которых содержатся введённые символы.

Нажатие на кнопку «Избранное» открывает страницу с товарами, которые пользователь добавил в список понравившихся.

Нажатие на кнопку «Корзина» открывает страницу с ранее добавленными товарами и предварительной информацией о заказе.

На верхней панели размещены следующие разделы:

* «Главная»;
* «Вход»;
* «Регистрация»;
* «Профиль»;
* «Редактирование профиля»;
* «Описание мастер-классов»;
* «Избранное»;
* «Оформление записи».

Функционал разделов верхней панели:

«Главная»: при нажатии пользователь перенаправляется на главную страницу сайта, где отображаются доступные мастер-классы. же можно перейти на страницу профиля.

«Вход»: при нажатии на кнопку, мы перейдем на страницу входа, где пользователь должен будет ввести некоторые данные.

«Регистрация»: при нажатии на кнопку, мы перейдем на страницу входа, где пользователь должен будет ввести некоторые данные.

«Профиль»: в профиле мы увидим кнопки «редактировать профиль», после которой, вся информация на странице «профиль» сохраняется, а также кнопка «избранное», в которое мы можем увидеть, добавленные нами мастер-классы.

«Редактирование профиля»: при нажатии кнопки, пользователь должен будет ввести все свои новые данные, либо старые. Они все сохранятся, при переходи на другие страницы.

«Описание мастер-классов»: его пользователь может увидеть, при выборе любого мастер-класса.

«Избранное»: открывает страницу, в которую мы собственно добавили 1 или больше мастер-классов. Если избранное пусто, он нам напишет «избранное пусто».

«Оформление записи»: оно осуществляется через кнопку «заказать» либо при выборе одного мастер-класса, либо уже в избранном, при 1 и больше мастер-классах

## Разработка клиентской части приложения

Логическая структура клиентской части приложения на C# (в контексте WPF) определяет организацию и взаимодействие компонентов, отвечающих за пользовательский интерфейс и его логику. Она включает в себя:

• Главное окно (MainWindow): На верхнем уровне находится главное окно приложения (MainWindow или его аналог), которое управляет общим потоком приложения и является контейнером для других окон и элементов управления. Оно отвечает за начальную загрузку, навигацию между окнами и общую координацию работы приложения.

• Окна (Windows): Ниже находятся отдельные окна (Window2, Window3 и другие), каждое из которых представляет собой отдельный экран или функциональную область приложения. Окна могут быть модальными (блокирующими работу с главным окном) или немодальными.

• Модули (UserControls/ViewModels): Внутри окон располагаются логические модули, часто представленные пользовательскими элементами управления (UserControls). Эти модули отвечают за конкретные функциональности, например, отображение списка избранного (Window3), оформление заказа (заказать\_груз) или аутентификацию.

• Обработчики событий: В коде C# определены обработчики событий, которые реагируют на действия пользователя (нажатие кнопок, ввод текста и т.д.) и выполняют соответствующие действия (открывают новые окна, обновляют данные на экране и т.д.). В наших примерах это методы Button\_Click, Button\_Click\_2 и UpdateLabel.

• Логика работы: Бизнес-логика приложения (например, работа с данными, взаимодействие с серверной частью) также может быть расположена в клиентской части, хотя часто её выносят в отдельные классы или слои (особенно при использовании архитектурных паттернов). В наших примерах это простая логика по отображению избранного и переходе между окнами.

Физическая структура клиентской части на C# (WPF):

Физическая структура клиентской части описывает организацию файлов и ресурсов, необходимых для работы приложения. Она обычно включает:

• Файлы C# (\*.cs): Содержат код классов, определяющих логику окон, пользовательских элементов управления, обработчиков событий и других компонентов.

• Файлы XAML (\*.xaml): Содержат разметку пользовательского интерфейса, описывающую расположение и внешний вид окон и элементов управления.

• Файлы XAML.CS (\*.xaml.cs): Содержат код, связанный с XAML-разметкой, обычно обработчики событий и код-бихайнд.

• Ресурсы:

\* Изображения (\*.png, \*.jpg и др.): Используются для отображения иконок, логотипов и других графических элементов.

\* Шрифты (\*.ttf и др.): Используются для определения внешнего вида текста.

\* Другие ресурсы: Звуки, видео и другие медиа-файлы, которые могут понадобиться приложению.

## Описание используемых функций и процедур

1. Функции/методы, связанные с окнами (Windows):

• MainWindow() (Конструктор главного окна):

• Назначение: Инициализирует главное окно приложения при его запуске.

• Действие: Создает экземпляр главного окна, настраивает его начальное состояние, загружает XAML-разметку.

• Window2() / Window3() (Конструкторы других окон):

• Назначение: Инициализируют соответствующие окна приложения при их создании.

• Действие: Создают экземпляры окон, настраивают их начальное состояние, загружают XAML-разметку, и обычно вызывают метод UpdateLabel (в Window3) для первичного заполнения данных.

• Show() (Метод открытия окна):

• Назначение: Отображает окно на экране.

• Действие: Делает окно видимым для пользователя, запуская его процесс. В коде использовался для открытия окон MainWindow и заказать\_груз.

• Close() (Метод закрытия окна):

\* Назначение: Закрывает окно на экране.

\* Действие: Закрывает текущее окно, убирая его с экрана. В коде использовался для закрытия окна Window3 при переходе на главное окно.

2. Функции/методы, связанные с обработкой событий (Event Handlers):

• Button\_Click(object sender, RoutedEventArgs e) (Обработчик события нажатия кнопки "Заказать груз"):

• Назначение: Выполняет действия, когда пользователь нажимает на кнопку "Заказать груз".

• Действия:

\* Проверяет, не пуст ли список избранного.

\* Если список не пуст, создаёт и открывает окно заказать\_груз (если не пуст) или выдаёт сообщение об ошибке (если пуст).

• Button\_Click\_2(object sender, RoutedEventArgs e) (Обработчик события нажатия кнопки "На главную"):

• Назначение: Выполняет действия, когда пользователь нажимает на кнопку "На главную".

• Действия:

\* Создает новый экземпляр MainWindow

\* Отображает главное окно (mainWindow.Show()).

\* Закрывает текущее окно (this.Close()).

3. Функции/методы, связанные с отображением данных:

• UpdateLabel() (Метод обновления содержимого Label):

• Назначение: Обновляет содержимое Label на форме Window3 (в нашем примере) на основе списка избранного.

• Действия:

\* Проверяет, есть ли элементы в списке Window2.Избранное.

\* Если список не пуст, форматирует его элементы в строку через запятую и устанавливает её в качестве содержимого myLabel.

\* Если список пуст, устанавливает текст "Избранное пусто" в качестве содержимого myLabel.

## Функциональное тестирование

Функциональное тестирование – это тестирование функций приложения на соответствие требованиям и проводится для выявления неполадок и недочетов программы на этапе ее сдачи в эксплуатацию.

В таблицах 3.1 - 3.4 представлены тест-кейсы для проведения функционального тестирования.

Предварительное условие: открыть сайт, быть не авторизированным и нажать на кнопку «Профиль».

В таблице 3.1 представлен тест-кейс для тестирования функции автризации/регистрации.

Таблица 3.1 – Тест кейс функций «Авторизация/регистрация пользователя»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Описание | Шаги выполнения | Результат |
| 1 | Проверка возможности регистрации нового пользователя | 1. Ввести логин 2. Ввести пароль 3. Повторить пароль 4. Нажать на кнопку «Регистрация» | Ожидаемый результат: Пользователь получил доступ к программе |
| Фактический результат: Совпадает с ожидаемым |
| Продолжение таблицы 3.1 | | | |
| 3 | Проверка успешного входа с корректными данными | 1. Ввести в поле «Пароль» пароль учетной записи 2. Нажать кнопку «Вход» | Ожидаемый результат: Пользователь получил доступ к программе |
| Фактический результат: Совпадает с ожидаемым |
| 4 | Тестирование корректной обработки неверных данных при попытке входа (неверный пароль) | 1. Ввести в поле «Логин» логин учетной записи 2. Ввести в поле «Пароль» пароль учетной записи 3. Нажать кнопку «Вход» | Ожидаемый результат: Получено сообщение об ошибке ввода |
| Фактический результат: Совпадает с ожидаемым |

Предварительное условие: открыть сайт, быть не авторизированным и нажать на кнопку «Профиль» или на кнопку «Авторизация».

В таблице 3.2 представлен тест-кейс для тестирования функции просмотра страницы товаров.

Таблица 3.2 – Тест кейс функций «Страница мастер-классов»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Описание | Шаги выполнения | Результат |
| 1 | Проверка корректного отображения информации о мастер-классах | 1. Просмотреть мастер-классов | Ожидаемый результат: Показывается список мастер-классов |
| Фактический результат: Совпадает с ожидаемым |
| 2 | Проверка возможности добавить товар в избранное. | 1. Нажать на мастер-класс 2. Нажать на кнопку «Добавить в избранное» 3. Проверить избранное | Ожидаемый результат: Товар добавлен картину |
| Фактический результат: Совпадает с ожидаемым |

Предварительное условие: открыть сайт, перейти на главную и выбрать категорию мастер-классов.

В таблице 3.3 представлен тест-кейс для тестирования функции добавления товара в корзину.

Таблица 3.3 – Тест кейс функций «Корзина»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Описание | Шаги выполнения | Результат |
| 1 | Проверка мастер-классов в корзине. | 1. Проверить, есть ли добавленный мастер-класс | Фактический результат: Совпадает с ожидаемым |
| 2 | Проверка заказа | 1. Нажать на кнопку «Заказать» | Фактический результат: Совпадает с ожидаемым |

Предварительное условие: открыть сайт, быть авторизованным, перейти на страницу избранное, иметь в корзине хотя бы 1 товар.

В таблице 3.6 представлен тест-кейс для тестирования функции профиль.

Таблица 3.6 – Тест кейс функции «Профиль»

|  |  |  |  |
| --- | --- | --- | --- |
| № | Описание | Шаги выполнения | Результат |
| 1 | Проверка отображения личной информации (имя, email). | 1. Открыть профиль 2. Нажать «редактировать профиль» 3. Изменить данные 4. Нажать кнопку «сохранить» | Ожидаемый результат: Данные отображаются корректно |
| Фактический результат: Совпадает с ожидаемым |

Предварительное условие: открыть сайт, быть авторизованным, нажать на кнопку «Профиль».

По результатам тестирования можно с уверенностью заявить, что программа готова к последующей работе с пользователями.

# Применение

## Назначение веб-приложения

Основное назначение данного веб-приложения – предоставить пользователям онлайн-платформу для записи на кулинарные мастер-классы и управления этими записями. Оно стремится упростить процесс выбора, регистрации и участия в кулинарных мероприятиях.

## Программно-аппаратное обеспечение сервера и клиента

Требования разработанного веб-приложения к аппаратным и программным ресурсам клиента следующие:

* любой популярный браузер, например: Microsoft Edge, Google Chrome, Mozilla Firefox Opera или иной;
* оперативная память 20 Гб;
* процессор с частотой от 2,4 ГГц.

Требования к аппаратным и программным ресурсам сервера выше, так как ему придется одновременно обрабатывать запросы от большого количества клиентов:

* операционная система Microsoft Windows или Linux;
* процессор AMD Ryzen 5 5600 или аналогичный;
* оперативная память 20 Гб.

# Заключение

В рамках курсового проекта, посвящённого разработке веб-приложения для магазина музыкальных инструментов и аксессуаров «SKZ4LOVER», было создано функциональное веб-приложение.

Основные функции веб-приложения включают:

* добавление товаров в избранное;
* оформление записи;
* сохранение товаров в избранное;

Веб-приложение имеет интуитивно понятный графический интерфейс, благодаря которому даже пользователи с минимальными навыками работы на компьютере могут легко им пользоваться.

Программа полностью реализована в соответствии с поставленными требованиями, успешно отлажена и протестирована. Все поставленные задачи выполнены.

В процессе разработки веб-приложения были применены и углублены знания изученного материала. Также были отработаны навыки применения методов надёжного программирования и повышения эффективности разработки программного обеспечения.

# Список информационных источников

1. Багласова, Т.Г. Методические указания по оформлению курсовых и дипломных проектов / Т.Г. Багласова, К.О. Якимович. – Минск: КБП, 2013. – 29 c.
2. Бондарь, А.Г. Microsoft SQL Server 2012 / А.Г. Бондарь. – СПб. : БХВ-Петербург, 2013. – 608 с.
3. Капитал в 21 веке / Томас Пикетти ; пер. с англ. Е. Артеменковой. – Москва : Ад Маргинем Пресс, 2015. – 656 с.
4. Мацяшек, Лешек А. Анализ и проектирование информационных систем с помощью UML 2.0 / Лешек А. Мацяшек. – 3-е изд. – М. : ООО «И.Д. Вильямс», 2008. – 816 с.
5. Митин, А.И. Работа с базами данных Microsoft SQL Server: сценарии практических занятий / А. И. Митин. – М. : Директ-Медиа, 2020. – 142 с
6. "Охрана труда в Республике Беларусь: Правовые и организационные аспекты" / Авторы: И.А. Калинкович, С.В. Шкрабут, В.С. Левченко и др.; Минск: Государственное научное учреждение "Институт экономики НАН Беларуси". 2019 г. - 236 с.
7. Общие требования к тестовым документам : ГОСТ 2.105-95. – Введ. 01.01.1996. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 1995. – 84 с.
8. Программа и методика испытаний. Требования к содержанию, оформлению и контролю качества : ГОСТ 19.301-2000. – Введ. 01.09.2001. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 14 с.
9. Текст программы. Требования к содержанию, оформлению и контролю качества: ГОСТ 19.401-2000. – Введ. 01.09.2001. – Минск : Межгос. совет по стандартизации, метрологии и сертификации, 2000. – 16 с.
10. Обзор обновлений и новых функций Windows 10 [Электронный ресурс]. – Microsoft, 2022 – Режим доступа: https://support.microsoft.com/ru-ru/windows#WindowsVersion=Windows\_10. – Дата доступа: 26.04.2023.
11. Обзор C# [Электронный ресурс]. – Microsoft, 2023 – Режим доступа: https://learn.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/. – Дата доступа: 05.05.2023

# ПРИЛОЖЕНИЕ А

**(обязательное)**

**Текст программных модулей**

Главное окно

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace проект

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Button1\_Click(object sender, RoutedEventArgs e)

{

// Создаем экземпляр Window1

Window1 window1 = new Window1();

// Отображаем окно Window1

window1.Show();

}

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

{

// Проверяем, были ли нажаты кнопки Button\_Click\_2 или Button\_Click\_3

Window2 window2 = new Window2();

// Отображаем окно Window2

window2.Show();

}

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

{

профиль профиль = new профиль();

// Отображаем окно Window1

профиль.Show();

}

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

вход form = new вход();

form.Show();

}

private void Button\_Click\_3(object sender, RoutedEventArgs e)

{

регистрация form = new регистрация();

form.Show();

}

}

}

Редактировать профиль

namespace проект

{

public partial class редактировать\_профиль : Form

{

public редактировать\_профиль()

{

InitializeComponent();

}

// Событие изменения текста в TextBox для имени

private void textBox1\_TextChanged(object sender, EventArgs e)

{

// Просто сохраняем значение, валидацию можно сделать позже, если это потребуется

// Например, можно проверять длину или наличие недопустимых символов

}

// Событие изменения текста в TextBox для фамилии

private void textBox2\_TextChanged(object sender, EventArgs e)

{

// Аналогично textBox1, пока просто сохраняем

}

// Событие изменения текста в TextBox для телефона

private void textBox3\_TextChanged(object sender, EventArgs e)

{

// Проверяем формат номера телефона при каждом изменении

ValidatePhoneNumber();

}

// Событие изменения текста в TextBox для почты

private void textBox4\_TextChanged(object sender, EventArgs e)

{

// Проверяем формат почты при каждом изменении

ValidateEmail();

}

// Событие нажатия на кнопку "Сохранить"

private void button1\_Click(object sender, EventArgs e)

{

// Перед сохранением проверяем все поля еще раз

if (!ValidateAllFields())

{

MessageBox.Show("Пожалуйста, исправьте ошибки в полях.", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

return; // Если есть ошибки, не сохраняем и выходим из метода

}

// Если все поля валидны, можно сохранять данные

string имя = textBox1.Text;

string фамилия = textBox2.Text;

string номерТелефона = textBox3.Text;

string электроннаяПочта = textBox4.Text;

// Создаем экземпляр формы "Профиль"

профиль profileForm = new профиль();

// Обновляем данные на форме "Профиль"

profileForm.UpdateProfile(имя, фамилия, номерТелефона, электроннаяПочта);

// Показываем форму "Профиль"

profileForm.Show();

// Закрываем текущую форму

this.Close();

}

// Метод для проверки формата номера телефона

private void ValidatePhoneNumber()

{

string phoneNumber = textBox3.Text;

// Используем регулярное выражение для проверки, начинается ли номер с +82 и содержит 10 цифр после

if (Regex.IsMatch(phoneNumber, @"^\+82\d{10}$"))

{

// Если номер телефона валиден, то убираем ошибку.

errorProvider1.SetError(textBox3, string.Empty);

}

else

{

// Если номер телефона не валиден, то показываем ошибку.

errorProvider1.SetError(textBox3, "Неверный формат номера телефона. Пример: +821234567890");

}

}

// Метод для проверки формата электронной почты

private void ValidateEmail()

{

string email = textBox4.Text;

// Используем регулярное выражение для проверки формата электронной почты

if (Regex.IsMatch(email, @"^[a-zA-Z0-9.\_%+-]+@gmail\.com$"))

{

// Если почта валидна, то убираем ошибку.

errorProvider1.SetError(textBox4, string.Empty);

}

else

{

// Если почта не валидна, то показываем ошибку.

errorProvider1.SetError(textBox4, "Неверный формат электронной почты. Пример: example@gmail.com");

}

}

// Метод для проверки всех полей перед сохранением

private bool ValidateAllFields()

{

ValidatePhoneNumber();

ValidateEmail();

// Проверяем, есть ли ошибки в errorProvider

return string.IsNullOrEmpty(errorProvider1.GetError(textBox1)) &&

string.IsNullOrEmpty(errorProvider1.GetError(textBox2)) &&

string.IsNullOrEmpty(errorProvider1.GetError(textBox3)) &&

string.IsNullOrEmpty(errorProvider1.GetError(textBox4));

}

private void редактировать\_профиль\_Load(object sender, EventArgs e)

{

}

}

}

Избранное

namespace проект

{

/// <summary>

/// Логика взаимодействия для Window3.xaml

/// </summary>

public partial class Window3 : Window

{

public Window3()

{

InitializeComponent();

UpdateLabel(); // Обновляем label при загрузке

}

private void UpdateLabel()

{

if (Window2.Избранное.Any())

{

// Выводим список в Label

myLabel.Content = string.Join(", ", Window2.Избранное);

}

else

{

myLabel.Content = "Избранное пусто";

}

}

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

// Создаем экземпляр главного окна (MainWindow)

MainWindow mainWindow = new MainWindow();

// Отображаем главное окно

mainWindow.Show();

// Закрываем текущее окно (Window3)

this.Close();

}

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

{

if (Window2.Избранное.Any())

{

заказать\_груз заказатьгруз = new заказать\_груз();

// Отображаем окно заказать\_груз

заказатьгруз.Show();

}

else

{

// Сообщаем пользователю об ошибке

MessageBox.Show("Заказывать нечего, так как избранное пусто.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

}

# ПРИЛОЖЕНИЕ Б

**(справочное)**

**Результаты работы приложения**

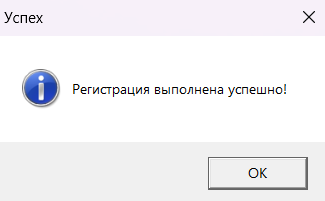


Рисунок Б.1 – Результат работы тест-кейса «Регистрация».

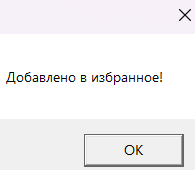


Рисунок Б.2 – Результат работы тест-кейса «Добавление мастер-класса в избранное».

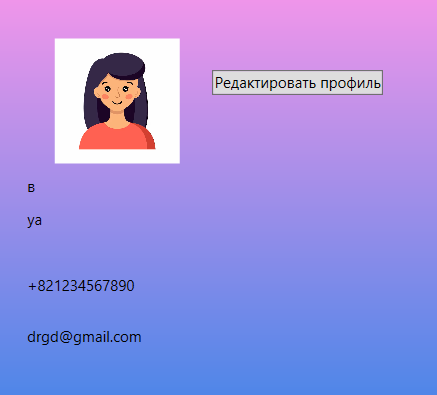


Рисунок Б.5 – Результат работы тест-кейса «Редактировать профиль».