**ОГЛАВЛЕНИЕ**

[**Введение 3**](#_Toc193844233)

[**ГЛАВА 1. ИНСТРУМЕНТЫ РАЗРАБОТКИ 6**](#_Toc193844234)

[**1.1 Общие понятия 6**](#_Toc193844235)

[**1.2 Понятие автоматизированной системы 8**](#_Toc193844236)

[**1.3 Выбор СУБД 10**](#_Toc193844237)

[**1.4 Выбор среды разработки 14**](#_Toc193844238)

[**1.5 Описание этапа разработки 16**](#_Toc193844239)

[**1.6 Планирование разработки 17**](#_Toc193844240)

[**1.7 Тестирование 19**](#_Toc193844241)

[**1.8 Критерии завершенности проекта 24**](#_Toc193844242)

[**ГЛАВА 2. РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА 25**](#_Toc193844243)

[**2.1 Проектирование и нормализация базы данных 25**](#_Toc193844244)

[**2.2 Разработка программного обеспечения для автоматизированной системы 26**](#_Toc193844245)

[**2.3 Тестирование созданной автоматизированной подсистемы 31**](#_Toc193844246)

# **Введение**

В условиях стремительного развития информационных технологий практически всех сфер деятельности, образовательные учреждения вынуждены адаптироваться к новым реалиям и совершенствовать процессы взаимодействия с клиентами. Учебные центры, как организаторы образовательных программ, сталкиваются с необходимостью оперативного и качественного приема заявок на обучение, что становится особенно актуальным при росте числа клиентов и расширении спектра предлагаемых услуг.

Традиционные методы обработки заявок – через телефонные звонки, бумажные формы или электронную почту – часто сопровождаются задержками, возможными системными ошибками и высокими трудовыми затратами. В таких условиях автоматизация процесса приема заявок представляется эффективным решением, позволяющим снизить нагрузку на персонал, упрощение отправки заявок в учебный центр для клиентов и повысить общий уровень сервиса.

Таким образом, разработка автоматизированной системы является актуальной задачей, способной оптимизировать работу образовательного учреждения, ускорить процесс обработки заявок и предоставить клиентам удобный и надежный способ подачи заявок на обучение.

Актуальность: В современных условиях образовательные центры нуждаются в оптимизации процессов, что позволяет сократить время обработки заявок и минимизировать ошибки традиционных методов приема.

Цель работы: Разработка автоматизированной системы для приема заявок на обучение, которая обеспечит оперативное взаимодействие с клиентами и повысит эффективность работы учебного центра.

Задачи работы:

1. Проанализировать существующие методы приема заявок и определить требования к системе.
2. Разработать архитектуру и проект системы.
3. Реализовать программный продукт с учетом современных стандартов.
4. Провести тестирование и оптимизацию системы.
5. Оценить эффективность внедрения автоматизированного решения.

Объект исследования: разработка автоматизированной системы для приема заявок на обучение клиентов учебного центра.

Предмет исследования: данной курсовой работы – автоматизированная система для приема заявок на обучение клиентов учебного центра

Методы исследования: теоретические, статистические, эмпирические.

Данный курсовой проект имеет следующую структуру:

1. Введение:

В данном разделе обосновывается актуальность выбранной темы, формулируется цель исследования, определяются конкретные задачи, а также обозначаются объект и предмет исследования.

1. Теоретическая глава:

Представлены общие данные о разрабатываемом приложении, проведен анализ требований к системам аналогичного типа, описаны характеристики автоматизации системы и сформирован план этапами и стадиям разработки.

1. Практическая часть:

Детально описываются этапы создания информационной системы для работников образовательного центра, включая:

* 1. проектирование и нормализацию базы данных,
  2. создание базы данных информационной системы,
  3. разработку пользовательского интерфейса,
  4. проведение тестирования разработанного решения.

4.Заключение:  
В данном разделе подводятся итоги проделанной работы, оценивается эффективность разработанной системы для приема заявок на обучение клиентов учебного центра.

Разработанная система показала свою эффективность в ходе тестирования, продемонстрировав стабильную работу, удобство использования и высокую производительность. Внедрение данной системы позволит улучшить качество предоставляемых услуг и повысить уровень обслуживание клиентов.

В процессе работы над курсовым проектом были изучены работы современных специалистов в области проектирования программных систем и программирования, а также общественные источники, что позволило обеспечить качество реализации проекта.

**Основные параметры работы:**

1. Количество страниц:

2.Количество литературных источников:

3.Количество рисунков:

# **ГЛАВА 1. ИНСТРУМЕНТЫ РАЗРАБОТКИ**

# **Общие понятия**

Автоматизированная система – это комплекс программных средств, предназначенных для упрощения обработки информации и выполнения рутинных задач с минимальным участием человека. Такие системы позволяют ускорить рабочие процессы и снизить вероятность ошибок.

В образовательных учреждениях автоматизированные системы для приема заявок обеспечивают быстрый и удобный способ регистрации клиентов, что способствует оптимизации работы учебного центра. Это позволяет эффективно управлять информацией и оперативно реагировать на запросы.

Применением автоматизированных решений улучшает качество обслуживания, повышает общую эффективность управления данными и облегчает работу сотрудников, что является важным фактором в условиях современного развития технологий.

Требование к Программному обеспечению:

Функциональные требования:

1. Регистрация и аутентификация пользователей (как для клиентов, так и для администраторов).
2. Форма для ввода заявок с необходимыми полями (ФИО, контактные данные, выбранный курс и т.д.).
3. Сохранение и хранение данных заявок в базе данных (Дата создания заявок, информация о клиенте).
4. Возможность редактирования, удаления и поиска заявок.
5. Автоматическая рассылка уведомлений о статусе заявки (подтверждение, изменение статуса, отклонение).
6. Генерация отчетов и аналитических данных по заявкам для администрации учебного центра.
7. Интеграция с внешними системами (например, с оплатой или почтовыми сервисами).

Нефункциональные требования:

1. Безопасность: Обеспечение защиты персональных данных клиентов и администраторов.
2. Производительность: обеспечение быстрого отклика системы при обработке запросов и высокой нагрузке, оптимизация работы базы данных.
3. Надежность: устойчивость к сбоям, наличие системы резервного копирования и восстановления данных.
4. Масштабируемость: возможность расширения функционала приложения.
5. Удобство использования: интуитивно понятный интерфейс, простота навигации и адаптивность для работы на различных устройствах и платформах.

Основанием для разработки программного обеспечения является индивидуальное задание к курсовому проекту, которое предусматривает создание системы для автоматизации системы для приема заявок на обучение клиентов учебного центра.

Что автоматизируется с помощью Программного обеспечения:

1.Учет заявок: автоматизация сбора, хранения и обработки данных о заявках на обучение.

2.Уведомление: автоматическое напоминание о важных событиях (например: изменение статуса заявки, подтверждение заявки)

3.Поиск и фильтрация данных: быстрый доступ к необходимой информации.

4.Аналитика и отчетность: анализ спроса на разные курсы обучения, отчеты для руководства учебного центра.

После анализа разных языков программирования будет выбран подходящий язык программирования. На этом языке будет разработана программа для автоматизации «приема заявок на обучения» с использованием СУБД для взаимодействия SQL.

Преимущества разработанной системы

1.Удобство использования: Возможность быстрого выбора курса и подачи заявки.

2.Оптимизация: Снижение нагрузки на сотрудников за счет автоматизации.

3.Автоматизация: Автоматическая обработка и сортировка заявок.

4. Безопасность данных: Встроенные меры безопасности для защиты конфиденциальной информации клиентов.

5.Экономия времени и ресурсов: Уменьшение затраты на бумажный документооборот и ручной ввод данных

6.Интеграция с внешними сервисами: Возможность подключение онлайн оплаты за обучение.

# **Понятие автоматизированной системы**

Автоматизированная система – это программно-аппаратный комплекс, разработанный для выполнения определенных задач с минимальным участием человека. Такие системы используются для хранения, обработки и управления данными, обеспечивая удобство и скорость выполнения операций.

Цель: оптимизация процесса, снижение времени обработки, снижение нагрузки на сотрудников.

Состав: интерфейс, база данных, система уведомлений.

Функции: прием и регистрация заявок на обучение, хранение данных, редактирование заявок, поиск данных в базе, уведомление клиентов о статусе их заявок.

Область применения: учебные центры и образовательные учреждения.

Свойства автоматизированной системы:

1. Автономность: минимизация необходимости вмешательства человека в процесс обработки заявок.
2. Надежность: обеспечение устойчивой работы без сбоев и потери данных.
3. Производительность – высокая скорость обработки заявок.
4. Безопасность: защита данных пользователей и предотвращение несанкционированного доступа.
5. Удобство использования: интуитивно понятный дизайн и легкость в освоении.

Этапы работы информационной системы представлены на схеме.



Рисунок 1 – схема работы информационной системы

На представленной схеме показан упрощенный процесс функционирования автоматизированной информационной системы, который можно разделить на ключевые этапы:

1. Сбор данных: система получает исходную информацию, включая автоматические датчики, сенсоры, другие информационные системы, а также ручной ввод пользователями.
2. Ввод данных: осуществляется пользователями с помощью клавиатуры, мыши, сканеров и других устройств.
3. Обработка данных: введенная информация подвергается различным операция, подготавливая данные для дальнейшего использования.
4. Хранение и обновление данных: полученная информация сохраняется в базе данных, где поддерживается её структурированное хранение.
5. Обновление: корректировка, добавление и изменение информации.
6. Поддержка: обеспечение безопасности, целостности данных, включая резервное копирование.
7. Поиск информации: пользователи или автоматизированные процессы могут запрашивать нужные данные в виде отчетов и поисковых запросов.
8. Формирование выходных данных –полученная информация преобразуется в удобный формат: отчеты, таблицы, графики.
9. Анализ данных: изучение полученной информации с целью выявления закономерностей, тенденций и аномалий, необходимых для принятия решений.
10. Принятие решений – на основе анализа данных принимаются управленческие или операционные решения, влияющие на дальнейшую работу системы.

В целом, схема отражает классический цикл обработки информации в автоматизированной системе: сбор, обработка, хранение, поиск, представление, анализ и принятие решений.

# **Выбор СУБД**

Разберем программное обеспечение-базу данных для хранения информации о заявках на обучение, а также для создания базы данных о клиентах на протяжении всего периода взаимодействия. Для проектирования этой системы проведём анализ разных программных решений и выберем наиболее подходящее.

Среди популярных систем управления базами данных (СУБД) для решения подобных задач выделяются следующие:

1. SQL – это встраиваемая реляционная база данных, созданная при помощи языка С. Она поддерживает большие наборы запросов и команд и отличается от многих СУБД, благодаря тому что функционирует без помощи специальных сервисных процессов, храня все данные в одном файле.

Преимущества:

1. Структура файла: преимущество в программе SQL является то, что вся база данных хранится в одном файле. Это упрощает переносимость, так как файл можно легко копировать между устройствами. Резервное копирование становится простым — достаточно сделать копию файла для сохранения всех данных.
2. Кроме того, отсутствие необходимости в отдельном сервере делает SQL удобным для разработки, использования в мобильных приложениях и встроенных системах.
3. Стандарты: SQL поддерживает большинство стандартных SQL операторов и конструкций, что позволяет разработчикам использовать знакомый синтаксис для работы с базами данных.
4. Удобство для разработки и тестирования: благодаря своей простоте, SQL идеально подходит для разработки и тестирования приложений, особенно когда требуется масштабирование и быстрая настройка. Эта база данных позволяет разработчикам сосредоточиться на создании функциональности без необходимости в сложной конфигурации, она обеспечивает высокую производительность и простоту в использовании.

Недостатки:

* Отсутствие системы пользователей: нет возможности настройки прав доступа для разных пользователей.
* Ограниченная производительность: нет механизмов для увеличения производительности при работе с большими объемами данных.

1. MySQL – это реляционная СУБД с открытым исходным кодом, которая поддерживает табличные базы данных как с простой, так и со сложной структурой. Она отличается высокой скоростью обработки данных и простотой интерфейса.

Преимущества:

1. Простота: легко устанавливается и настраивается. Существует множество сторонних инструментов, упрощающих работу с базой данных.
2. Функциональность: поддерживает большую часть стандартных функций SQL.
3. Безопасность: встроенные механизмы безопасности обеспечивают защиту данных.
4. Масштабируемость: MySQL обладает высокой производительностью и способна эффективно обрабатывать большие объемы данных. Эта система управления базами данных (СУБД) идеально подходит для разработки масштабируемых приложений, которые могут расти и развиваться с течением времени.

Недостатки:

Ненадежность: Некоторые операции менее надежны по сравнению с другими СУБД.

1. Microsoft SQL Server – это СУБД, разработанная компанией Microsoft, которая поддерживает операционные системы Windows и Linux. Она характеризуется простым интерфейсом, высокой надежностью и совместимостью с другими продуктами Microsoft, такими как Excel и Access.

Преимущества:

1. Масштабируемость: подходит как для портативных устройств, так и для мощных серверов.
2. Высокая производительность: быстрое извлечение данных благодаря размеру страниц до 8 КБ.
3. Автоматизация: обычные задачи, такие как управление памятью и блокировками, автоматизированы.
4. Интеграция: поддерживает работу с другими продуктами Microsoft.

Недостатки:

Высокая стоимость: использование в коммерческих целях требует оплаты лицензии.

В результате анализа была выбрана СУБД MySQL по следующим причинам:

1. Отсутствие необходимости в администрировании: SQL не требует сложной настройки и управления.
2. Простота подключения: Легко интегрируется в приложения.
3. Высокая производительность: При достаточном объеме оперативной памяти обеспечивает быструю обработку данных.
4. Бесплатность: SQL распространяется бесплатно, что делает её экономически выгодным решением.

# **Выбор среды разработки**

Среда разработки (IDE – Integrated Development Environment) — это набор инструментов, который помогает программистам писать, тестировать и отлаживать код. Правильный выбор IDE влияет на удобство работы, скорость разработки и качество кода.

Среди популярных сред разработки выделяются следующие:

Visual Studio - это интегрированная среда разработки (IDE), созданная Microsoft. Она предназначена для создания различных программных обеспечений , включая десктопные и веб-приложения, мобильные приложения, игры и облачные сервисы.

Преимущества:

* Поддержка множества языков программирования:

Visual Studio поддерживает широкий спектр языков программирования:

C# (основной язык для .NET-разработки).

C++ (поддержка разработки на Windows и кроссплатформенной разработки).

Python, JavaScript и многие другие.

* Гибкость и расширяемость:

Visual Studio Marketplace предлагает тысячи расширений:

Расширения для Python, PHP, Java, Go.

Подключение сторонних инструментов (Postman, Redis, AWS, Azure).

* Инструменты для создания UI

Визуальные редакторы интерфейсов:

WinForms, WPF, MAUI для создания десктопных приложений.

ASP.NET, Blazor для веб-разработки.

Xamarin, .NET MAUI для кроссплатформенной мобильной разработки.

Недостатки:

* Высокая ресурсоёмкость

Visual Studio требует много оперативной памяти и процессорных ресурсов, особенно при работе с большими проектами.

При запуске больших решений может занимать 3-5 ГБ RAM.

Устаревшие компьютеры могут заметно тормозить.

Медленный старт IDE, особенно на слабых ПК.

* Сложность для новичков

Visual Studio имеет сложный интерфейс, с чем могут возникнуть сложности у новых пользователей.

Требует времени на освоение отладочных инструментов.

PyCharm — это мощная интегрированная среда разработки (IDE) для Python, созданная компанией JetBrains. Она широко используется для разработки веб-приложений, анализа данных и автоматизации процессов.

Преимущества:

* Умное автодополнение (IntelliSense):

PyCharm анализирует код в режиме реального времени и предлагает:

Автодополнение кода (методов, переменных, классов).

Подсказки по параметрам функций.

Предложения по исправлению ошибок.

* Удобный рефакторинг кода:

Автоматически переименовывать переменные и функции по всему проекту.

Оптимизировать импорт модулей.

Перемещать классы и функции между файлами без риска ошибок.

* Интеграция с Git и системами контроля версий:

Встроенная поддержка Git, GitHub, GitLab, Bitbucket, Mercurial.

Недостатки:

* Долгий запуск IDE:

PyCharm долго загружается, особенно на больших проектах.  
 При открытии множества файлов IDE может подвисать.

Высокая ресурсоёмкость: PyCharm требует много оперативной памяти и процессорных ресурсов.  
Может замедляться на слабых компьютерах. Рекомендуется не менее 8 ГБ RAM (лучше 16+).

* Некоторые плагины платные:

Не все расширения бесплатны (например, поддержка некоторых баз данных).

# **Описание этапа разработки**

Планирование разработки — это ключевой этап, который позволяет создать качественное, надежное и соответствующее требованиям программное обеспечение.

1. Анализ требований**:**

На данном этапе проводится анализ задач, которые должно решать программное обеспечение.

Основные требования:

1. Учёт и хранение данных о заявках (номер заявки, данные клиента, дата создания, статус заявки).
2. Управление данными (ФИО, контактные данные, выбранная специализация, результаты аттестации клиента).
3. Анализ данных на основе истории заявок (статистика по заявкам)
4. Формирование отчётов по выполненным и отработанным ранее заявкам.
5. Удобный интерфейс для использования, позволяющий быстро находить информацию, управлять заявками и отслеживать статусы.
6. Надёжное хранение данных с использованием SQL.
7. Высокая производительность и минимальное количество ошибок при обработке заявок.

2. Определение спецификаций**:**

Для разработки программы используются следующие средства и технологии:

* 1. Язык программирования: C#.
  2. База данных: SQL (для хранения и управления данными).
  3. Фреймворк: Windows Forms (WinForms) или WPF для создания десктопного приложения.
  4. Среда разработки: Visual Studio.
  5. Система контроля версий: Git (например, через GitHub или GitLab).

# **Планирование разработки**

Планирование разработки — это ключевой этап в создании программного обеспечения, который помогает определить цели проекта, его архитектуру, сроки выполнения и ресурсы.

Этапы разработки:

* + - 1. Анализ требований:

На данном этапе проводится исследование задач, которые должна решать система.

Функциональные требования:

Регистрация и аутентификация пользователей (клиентов и администраторов).

Форма для ввода заявок (ФИО, контактные данные, выбранный курс).  
Хранение данных в базе SQL (информация о заявках, клиентах).  
Редактирование, удаление и поиск заявок.  
Автоматические уведомления о статусе заявки.  
Генерация отчётов и аналитических данных по заявкам.

Нефункциональные требования:

Безопасность: защита персональных данных.  
Производительность: быстрая обработка запросов.  
Надёжность: устойчивость к сбоям, система резервного копирования.  
Масштабируемость: возможность расширения функционала.  
Удобство: интуитивно понятный интерфейс.

* + - 1. Выбор технологий:

На основе требований выбираются язык программирования, база данных, фреймворк и среда разработки.

* Язык программирования: C#

Мощный язык для разработки десктопных приложений, удобная работа с GUI.

* Фреймворк: Windows Forms (WinForms) или WPF
* СУБД: SQL (для хранения и управления данными)

Надёжная база данных для хранения заявок и информации о клиентах.

* Среда разработки: Visual Studio

Профессиональная среда разработки с поддержкой отладки.

* Система контроля версий: Git (GitHub, GitLab)

Удобная система для хранения и версионирования кода.

* + - 1. Проектирование системы:

Создание архитектуры приложения и структуры базы данных.

Архитектура системы:

* Клиентская часть (пользовательский интерфейс).
* Серверная часть (логика обработки данных).
* База данных (хранение информации).
  + - 1. Реализация:

Разработка включает в себя:

* Создание базы данных (SQL).
* Разработка пользовательского интерфейса (C#, WinForms/WPF).
* Реализация бизнес-логики (работа с заявками, уведомления, отчёты).
* Интеграция с внешними сервисами (например: почтовые уведомления, оплата)
  + - 1. Тестирование системы:

Критерии успешного тестирования:

* Функциональность – система выполняет все заявленные функции.
* Безотказность – система не падает при обработке данных.
* Производительность – быстрая работа с базой данных.
* Безопасность – защита данных клиентов.

# **Тестирование**

Тестирование – один из ключевых этапов разработки, который позволяет выявить ошибки, проверить соответствие требованиям и убедиться в стабильности работы системы. В данном проекте тестирование направлено на проверку работоспособности системы для приёма заявок на обучение в учебный центр.

Цели тестирования:

* + Убедиться, что система работает без ошибок.
  + Проверить, что все функции соответствуют функциональным требованиям.
  + Оценить производительность системы при разных нагрузках.
  + Проверить защищённость данных (персональные данные клиентов, заявки).
  + Определить устойчивость к сбоям и отказам.

Виды тестирования:

* 1. Функциональное тестирование:

Цель: проверить, что каждая функция системы работает в соответствии с требуемой спецификацией и соответствует заявленным требованиям.

Что тестируется:

* Регистрация и аутентификация: проверка правильности ввода логина/пароля, обработка ошибок при неверных данных, восстановление пароля.
* Создание заявки: ввод данных (ФИО, контакты, выбранный курс), обязательность заполнения всех необходимых полей.
* Редактирование и удаление заявок: корректность обновления данных, ограничения доступа для разных ролей (например, администратор/ обычный пользователь).
* Поиск и фильтрация: корректный поиск заявок по различным критериям (дата, статус, клиент) и отображение результатов.
* Уведомления: проверка, что уведомления (например, подтверждение заявки, изменение статуса) отправляются корректно.
* Генерация отчетов: проверка формирования аналитических данных и статистики по заявкам.

Методы:

* Ручное тестирование: тестировщик выполняет шаги, прописанные в тест-кейсах, и фиксирует обнаруженные отклонения.
* Автоматизированное функциональное тестирование: написание скриптов, которые последовательно проверяют работу каждой функциональной единицы.
  1. Юнит-тестирование (тестирование отдельных модулей):

Цель: проверить корректность работы отдельных модулей или компонентов системы (функций, методов, классов) в изоляции от других частей приложения.

Что тестируется:

* Бизнес-логика: методы для обработки заявок, изменения их статуса, расчёты для генерации отчетов.
* Работа с базой данных: операции вставки, обновления, удаления и выборки данных.
* Вспомогательные функции: валидация данных, формирование уведомлений и обработка исключений.

Методы и инструменты:

* Использование фреймворков для юнит-тестирования (например, NUnit для C#).
* Создание тестовых наборов (test suites) для проверки каждой функции с различными входными данными.
* Имитация (mocking) зависимостей для проверки работы модулей в изоляции.
  1. Интеграционное тестирование:

Цель: проверить корректное взаимодействие между отдельными модулями системы, когда они объединены в единое целое.

Что тестируется:

* Связь между интерфейсом и серверной частью: корректная передача данных с пользовательского интерфейса в базу данных и обратно.
* Последовательность бизнес-процессов: выполнение полного сценария, от подачи заявки до изменения её статуса.

Методы:

* + Проведение сценарного тестирования, когда тестировщик имитирует реальные рабочие процессы.
  + Автоматизация тестов, проверяющих обмен данными между компонентами.
  1. Нагрузочное тестирование:

Цель: оценить, как система ведет себя под высокой нагрузкой, и определить её предельные возможности.

Что тестируется:

* + Обработка большого объема данных:

создание и обработка большого числа заявок за короткий период.

* + Время отклика:

измерение скорости выполнения операций (запись, чтение, обновление данных).

* + Производительность базы данных:

способность базы данных справляться с параллельными запросами.

Методы и инструменты: мониторинг системных ресурсов (CPU, RAM, сеть) во время проведения тестирования.

* 1. Тестирование безопасности:

Цель: обеспечить защиту системы от внешних угроз и убедиться, что данные пользователей защищены от несанкционированного доступа.

Что тестируется:

* + Аутентификация и авторизация: проверка, что доступ к функционалу системы ограничен в соответствии с ролями пользователей.
  + Защита данных: шифрование паролей, использование безопасных протоколов для передачи данных.
  + Контроль доступа: правильная настройка прав для различных типов пользователей (администратор/клиент).

Методы:

* + Использование специализированных программных средств для сканирования уязвимостей.
  + Анализ логов системы на предмет попыток несанкционированного доступа.

Итог: каждый из видов тестирования охватывает различные аспекты работы системы: от проверки отдельных функций до оценки общей производительности и безопасности. Такой комплексный подход позволяет:

* + Обнаружить ошибки на ранних этапах разработки.
  + Убедиться, что система соответствует заявленным требованиям.
  + Гарантировать надежность и устойчивость работы продукта.

Эффективное тестирование обеспечивает качественный продукт, который будет стабильно работать при реальных нагрузках и защищен от внешних угроз, что является критически важным для автоматизированной системы приёма заявок в учебном центре.

# **Критерии завершенности проекта**

Проект считается завершённым, когда он соответствует всем требованиям и демонстрирует безотказную работоспособность. Это означает, что разработанное программное обеспечение полностью удовлетворяет заявленным функциональным и нефункциональным требованиям, а также стабильно работает в условиях реального использования. Основными критериями завершенности являются:

1. Соответствие требованиям

* Функциональные требования: все функции, заявленные в техническом задании (регистрация, обработка заявок, уведомления, генерация отчетов и т.д.), реализованы и работают корректно.
* Нефункциональные требования: продукт удовлетворяет требованиям по безопасности, производительности, надежности, удобству использования и масштабируемости.

1. Безотказная работоспособность:

* Тест-кейсы: каждый сценарий использования проверен с помощью заранее разработанных тест-кейсов.
* Юнит-тесты:

Все ключевые модули системы протестированы с использованием юнит-тестов.

# **ГЛАВА 2. РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА**

# **2.1 Проектирование и нормализация базы данных**

База данных — это набор организованной информации, хранящейся на компьютере определенным структурированным способом. Она позволяет легко получать доступ к данным, управлять ими и обновлять их. База данных работает как хранилище, где информация упорядочена для быстрого поиска и использования. Данные хранятся в виде таблиц, состоящих из строк и столбцов, или в других форматах, в зависимости от типа базы данных.

Разработка базы данных была выполнена в системе управления реляционными базами данных MySQL. База данных хранит всю необходимую информацию для работы системы, включая данные о пользователях, заявках и курсах. Ниже приведено описание таблиц, их назначение и связь между ними.

Таблицы базы данных:

Таблица clients создана для работы с данными клиентов

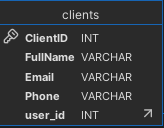


Рисунок 2 – Таблица clients .

Таблица users создана для хранения аккаунтов

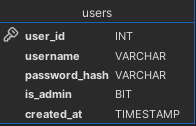


Рисунок 3 – Таблица users.

Таблица applications создана для хранения информации о заявках

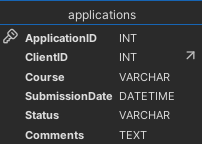


Рисунок 4 – Таблица applications.

# **2.2 Разработка программного обеспечения для автоматизированной системы**

Пользовательский интерфейс автоматизированной системы для приема заявок на обучение клиентов учебного центра разработан и реализует современный дизайн с темной цветовой схемой. Интерфейс характеризуется интуитивной понятностью и эргономичностью.

**Общие элементы интерфейса**

Цветовая схема:

1. Основной фон: темно-серый (#1E1E1E)
2. Вторичный фон: чуть светлее основного (#2D2D30)
3. Акцентный цвет: синий (#007ACC)
4. Текст: белый или светло-серый

Компоненты интерфейса:

1. Кнопки: прямоугольные с закругленными углами, с эффектами наведения
2. Текстовые поля: с выделенной нижней границей
3. Выпадающие списки (ComboBox): с темным фоном и светлым текстом
4. Таблицы данных (DataGrid): с чередующимися темными строками для удобства чтения

**Основные функциональные модули системы:**

Модуль авторизации и регистрации:

1. Авторизация существующих пользователей с разделением по ролям
2. Регистрация новых клиентов
3. Восстановление доступа

Административный модуль:

1. Управление пользователями
2. Управление курсами (добавление, редактирование, деактивация)
3. Обработка заявок (просмотр, одобрение, отклонение)

Клиентский модуль:

1. Просмотр доступных курсов
2. Подача заявок на обучение
3. Отслеживание статуса заявок

**Пользовательский интерфейс:**

**Окно авторизации**

Окно авторизации предоставляет поля для ввода логина и пароля, а также возможность перехода к окну регистрации. Предусмотрена валидация вводимых данных и отображение информативных сообщений об ошибках.

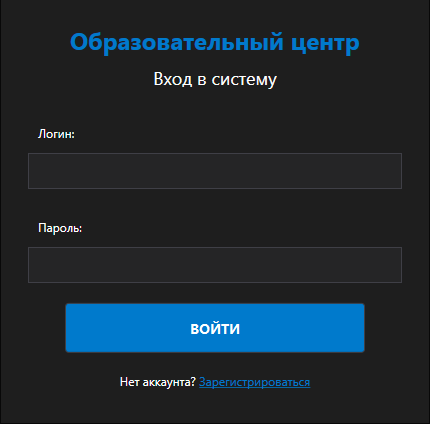


Рисунок 5 – Окно авторизации.

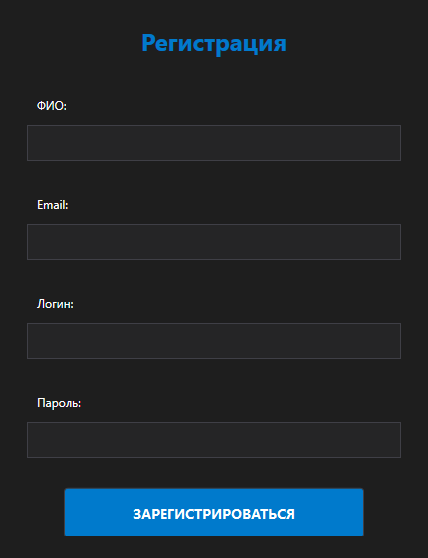


Рисунок 6 – Окно регистрации.

**Административная панель**

Административная панель содержит несколько вкладок для управления различными аспектами системы:

1. Управление пользователями
2. Управление курсами
3. Обработка заявок
4. Формирование отчетов

Для каждой вкладки предусмотрен набор элементов управления, соответствующих функциональности.

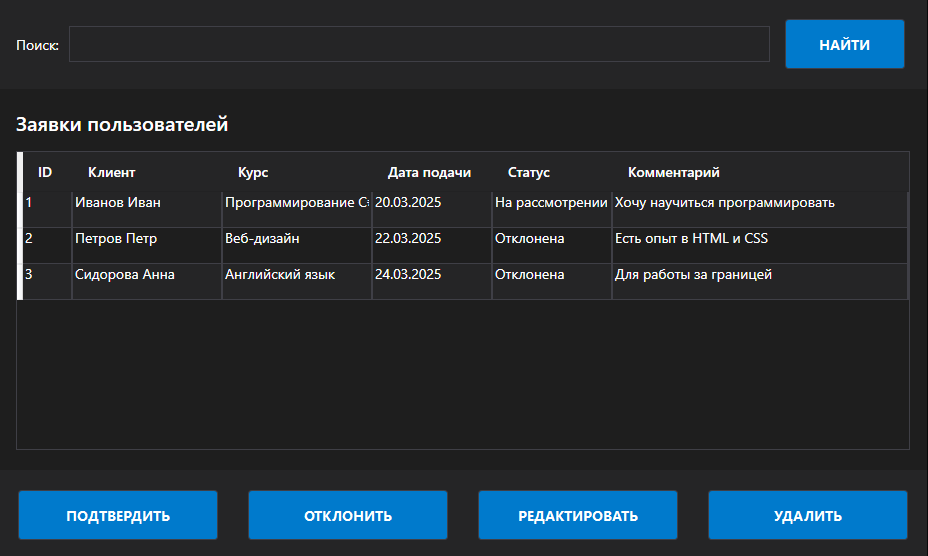


Рисунок 7 – Окно панели администратора.

**Клиентский интерфейс**

Интерфейс клиента включает следующие основные экраны:

1. Личный кабинет с информацией о пользователе
2. Просмотр доступных курсов
3. Форма подачи заявки
4. История заявок с возможностью отслеживания их статуса

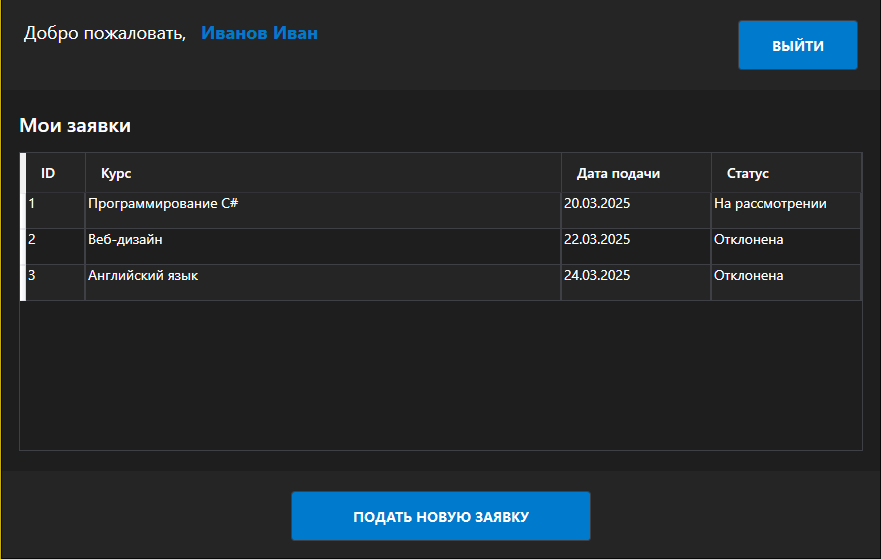
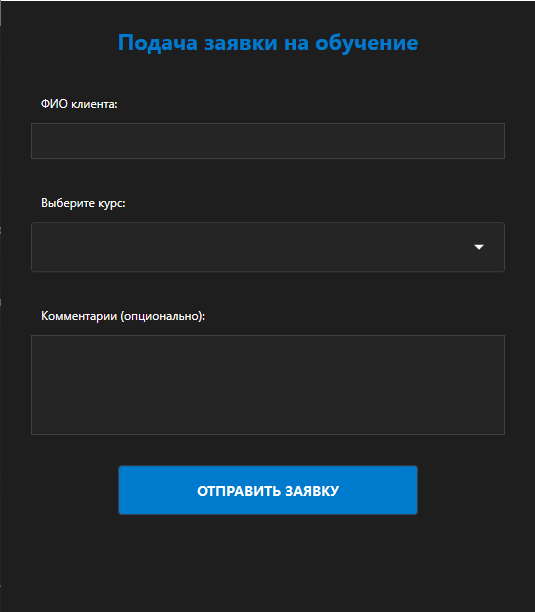


Рисунок 8 – Окно клиента.

**Функционал приложения:**

**Для клиентов:**

* Подача новых заявок на выбранные курсы обучения
* Просмотр статуса поданных заявок (на рассмотрении, подтверждена, отклонена)
* Отслеживание истории заявок с датами и комментариями

  
Рисунок 9 – Окно подачи заявок

**Для администраторов:**

* Просмотр всех поступивших заявок в системе
* Фильтрация и поиск заявок по различным критериям (ФИО клиента, курс, статус, дата)
* Сортировка заявок по любому из параметров
* Редактирование заявок с возможностью изменения курса и добавления комментариев
* Изменение статуса заявок (одобрение/отклонение)

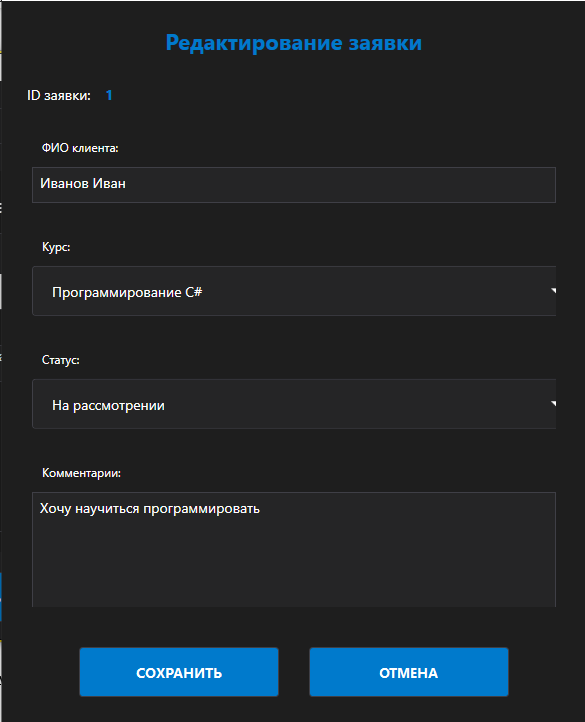


Рисунок 10 – Окно редактирования заявки

# **2.3 Тестирование созданной автоматизированной подсистемы**

После завершения этапа разработки программного продукта особое внимание было уделено тестированию созданной автоматизированной подсистемы. Цель данного этапа – подтверждение корректности работы всех функциональных компонентов, обеспечение стабильности системы при реальных нагрузках, а также выявление и устранение возможных ошибок до этапа внедрения.

**Цели и задачи тестирования:**

Основными целями тестирования стали:

* Проверка соответствия реализованных функций требованиям технического задания;
* Оценка стабильности работы системы при стандартных и экстремальных нагрузках;
* Выявление ошибок в работе модулей и их интеграции;
* Обеспечение безопасности данных и корректного функционирования механизмов защиты;
* Повышение удобства использования подсистемы за счет выявления проблем в пользовательском интерфейсе.

**Организация тестирования:**

Тестирование проводилось в несколько этапов, что позволило охватить все ключевые аспекты работы системы:

* Функциональное тестирование: проверялась корректность работы отдельных модулей – регистрация и аутентификация пользователей, создание, редактирование и удаление заявок, поиск и фильтрация данных, а также генерация отчетов.
* Интеграционное тестирование: особое внимание уделялось взаимодействию между компонентами системы: передача данных между пользовательским интерфейсом, серверной логикой и базой данных, а также корректная работа уведомлений и внешних API.
* Нагрузочное тестирование: с помощью специализированных инструментов оценивалась производительность системы при одновременной работе большого количества пользователей и обработке массового объема заявок.
* Тестирование безопасности: проводились проверки на устойчивость к типичным угрозам (SQL-инъекции, XSS, CSRF) и оценивалась корректность работы механизмов защиты персональных данных.

**Методика проведения тестов:**

Для повышения объективности и полноты тестирования были разработаны подробные тест-кейсы, отражающие реальные сценарии использования системы. Процесс тестирования включал как ручное, так и автоматизированное тестирование:

* Ручное тестирование выполнялось пошагово проверялось выполнение основных сценариев работы, фиксируя обнаруженные несоответствия.
* Автоматизированное тестирование проводилось с использованием фреймворков для юнит-тестирования, что позволило оперативно выявлять ошибки в отдельных модулях и обеспечить повторяемость тестовых сценариев.

**Результаты тестирования и их анализ:**

Проведенные тесты показали, что:

* Все заявленные функции работают корректно и в соответствии с требованиями;
* Система демонстрирует высокую стабильность даже при максимальной нагрузке, а время отклика остается в пределах допустимых норм;
* Механизмы аутентификации и защиты данных успешно предотвращают попытки несанкционированного доступа;
* Пользовательский интерфейс является интуитивно понятным, что обеспечивает комфортное взаимодействие для пользователей.

Обнаруженные в процессе тестирования недостатки были оперативно исправлены, что позволило повысить качество программного продукта и обеспечить его готовность к эксплуатации в реальных условиях.

**Выводы:**

Комплексное тестирование подтверждает, что разработанная автоматизированная подсистема соответствует всем заявленным функциональным и нефункциональным требованиям. Полученные результаты являются показателем высокого качества продукта, что обеспечивает его надежное и эффективное использование для автоматизации приема заявок на обучение.

**Заключение**

В результате проделанной работы была разработана автоматизированная система для приема заявок на обучение, которая успешно решает поставленные задачи по оптимизации и ускорению обработки информации. Проект прошел все этапы – от анализа требований и проектирования базы данных до реализации пользовательского интерфейса и комплексного тестирования, что подтверждает его работоспособность и надежность.

**Основные результаты проекта**

* **Полное соответствие функциональным требованиям:** Реализованы все ключевые функции, включая регистрацию и аутентификацию пользователей, создание, редактирование и удаление заявок, а также поиск и фильтрацию данных.
* **Повышение производительности и надежности:**
* Система демонстрирует высокую скорость обработки данных, стабильную работу при различных нагрузках и устойчивость к ошибкам, что особенно важно для образовательного учреждения.
* **Безопасность данных:**
* Внедрены механизмы защиты персональной информации пользователей, что позволяет минимизировать риски несанкционированного доступа и утечки данных.
* **Удобство использования:**
* Интуитивно понятный и эргономичный интерфейс обеспечивает комфортное взаимодействие как для конечных пользователей, так и для администраторов, что способствует повышению качества обслуживания клиентов.
* **Комплексное тестирование:**
* Проведенные функциональные, интеграционные, нагрузочные и тесты безопасности подтвердили стабильную работу системы и позволили своевременно выявить и устранить недостатки.

**Планы по доработке системы:**

* **Расширение функциональности:**
* Планируется добавить дополнительные модули, такие как аналитика по динамике заявок, прогнозирование нагрузки и автоматическая генерация расширенных отчетов, что позволит получить более глубокую информацию о работе центра.
* **Интеграция с внешними сервисами:**
* Внедрение дополнительных интеграций с популярными платежными системами и внешними сервисами для улучшения автоматизации процессов и повышения оперативности работы системы.
* **Улучшение пользовательского интерфейса:**
* На основе обратной связи пользователей будут внесены изменения в дизайн и функциональные элементы интерфейса, что позволит сделать работу системы еще более удобной и интуитивно понятной.
* **Оптимизация производительности:**
* Планируется провести дополнительные работы по оптимизации кода и базы данных для повышения масштабируемости системы и обеспечения быстрого отклика при увеличении объема данных и количества пользователей.
* **Повышение уровня безопасности:**
* В рамках дальнейших доработок будет проведен дополнительный аудит безопасности с целью выявления потенциальных уязвимостей и внедрения новых механизмов защиты данных.

Таким образом, реализованный проект является надежной основой для автоматизации процесса приема заявок на обучение, а предусмотренные планы по доработке позволят адаптировать систему к будущим требованиям и расширить ее функциональные возможности для удовлетворения растущих потребностей образовательного центра.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Антонова, А. В. Компьютерное тестированиесовременная форма контроля учебных достижений учащихся в процессе обучения истории в школе / А. В. Антонова // В сборнике: Инновации и традиции педагогической науки2019. Сборник трудов ХIХ Международной научнопрактической конференции / под ред. Т. А. Макаренко, С. В. Паниной. — 2019. — С. 2327. — Текст : электронный // elibrary.ru [сайт]. — URL: https://www.elibrary.ru/download/elibrary\_37326820\_65482003.pdf (дата обращения: 04.02.2023).

2. Болье, А. Изучаем SQL. Генерация, выборка и обработка данных / А. Болье. — М.: Издательство ДиалектикаВильямс, 2020. — 423 с. — (Профессиональное образование). — Текст: электронный // ЭБС МГТУ им. Баумана [сайт]. — URL: https://elearning.bmstu.ru/iu6/pluginfile.php/18674/mod\_resource/content/1/Изучаем\_SQL\_Генерация%2C\_выборка\_и\_обработка\_данных%2C\_3\_изд.pdf (дата обращения: 01.02.2023).

3. Заельская, С. А. История отечества с древнейших времён и до наших дней в тестах и заданиях: электронное учебнометодическое пособие для бакалавров заочного отделения / С. А. Заельская. — Оренбург, 2019. — Текст : электронный // elibrary.ru [сайт]. — URL: https://lib.rucont.ru/efd/702462/info (дата обращения: 04.02.2023).

4. Зарапин, Р. В. Образовательный потенциал использования онлайнтестирования при изучении истории / Р. В. Зарапин // В сборнике: Актуальные вопросы гуманитарных наук: теория, методика, практика. Сборник научных статей IХ Всероссийской научнопрактической конференции с международным участием. — Москва, 2022. — С. 239244. — Текст : электронный // elibrary.ru [сайт]. — URL: https://www.elibrary.ru/download/elibrary\_50163440\_95910012.pdf (дата обращения: 04.02.2023).

5. Маркин, А. В. Программирование на SQL в 2 ч. Часть 1 : учебник и практикум для вузов / А. В. Маркин. — 3е изд., перераб. и доп. — М. : Издательство Юрайт, 2023. — 429 с. — (Высшее образование). — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/509818 (дата обращения: 04.02.2023).

6. Советов, Б. Я. Базы данных : учебник для вузов / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской. — 3е изд., перераб. и доп. — М. : Издательство Юрайт, 2023. — 420 с. — (Высшее образование). — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/510752 (дата обращения: 04.02.2023).

7. Стасышин, В. М. Базы данных: технологии доступа : учебное пособие для вузов / В. М. Стасышин, Т. Л. Стасышина. — 2е изд., испр. и доп. — М.: Издательство Юрайт, 2023. — 164 с. — (Высшее образование). — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/514252 (дата обращения: 04.02.2023).

8. MYSQL.COM — [Электронный ресурс] / Электрон. текстовые дан. 2022. — Режим доступа: https://www.mysql.com/, свободный. (дата обращения: 01.02.2023).

12. Васильев, А. Н. Программирование на Python в примерах и задачах / А. Н. Васильев. — 1е изд. — Москва: Эксмо, 2022. — 616 с.

13. Мэтиз, Э. Изучаем Python: программирование игр, визуализация данных, вебприложения / Э. Мэтиз. — 3е изд. — Питер: Издательский дом «Питер», 2022. — 560 с.

14. Любанович, Б. Простой Python. Современный стиль программирования / Б. Любанович. — 2е изд. — Питер: Издательский дом «Питер», 2025. — 592 с.