Содержание

|  |  |
| --- | --- |
| Введение | 2 |
| 1 Назначение и цели разработки | 3 |
| 2 Разработка технического проекта на основе анализа требований | 4 |
| 2.1 Определение спецификаций программного обеспечения | 4 |
| 2.2 Проектирование модели данных и диаграммы классов | 11 |
| 2.3 Конструирование прототипа | 16 |
| 3 Реализация | 20 |
| 3.1 Обоснование выбора средств разработки | 20 |
| 3.2 Разработка базы данных в среде СУБД | 21 |
| 3.3 Описание программных модулей | 25 |
| 3.3.1 Модуль регистрации/авторизации | 26 |
| 3.3.2 Модуль добавления/изменения/удаления информации о врачах | 26 |
| 3.3.3 Модуль добавления/изменения/удаления информации о пациентах | 27 |
| 3.3.4 Модуль добавления/изменения/удаления информации о приемах | 27 |
| 3.3.5 Модуль аналитики | 28 |
| Список использованных источников | 22 |
| Приложение А Техническое задание. Требования к программным модулям | 25 |
| Приложение Б Программный код | 27 |
|  |  |

Введение

В настоящее время автоматизация работы медицинских учреждений становится все более востребованной. Особенно это актуально для частных клиник, которые вынуждены оперативно реагировать на потребности пациентов и эффективно управлять своей деятельностью. Одним из ключевых элементов успешной работы клиники является организация работы администратора, который должен контролировать все процессы в клинике и обеспечивать эффективную работу врачей и пациентов.

В связи с этим было принято решение начать разработку информационной системы «ZimAdmin». Эта система позволит значительно упростить и ускорить процессы работы администратора частной клиники «Zimamed», повысить качество обслуживания пациентов, а также обеспечить эффективное управление бизнес-процессами клиники.

1 Назначение и цели разработки

Назначение разработки информационной системы "ZimAdmin" заключается в создании программного продукта, который поможет оптимизировать и автоматизировать работу администратора частной клиники.

Функции системы:

– ведение базы данных пациентов с возможностью поиска по различным критериям (имя, фамилия, телефон и т.д.);

– регистрацию пациентов на прием к врачу;

– формирование расписания приема пациентов;

– отображение информации о текущих и предстоящих приемах пациентов;

– возможность внесения изменений в расписание приема;

– обработку заявок на вызов врача на дом;

– анализ нагрузки врачей;

– анализ количества записей на определённые процедуры;

– формирование отчетов по приемам и вызовам врача на дом.

Цели разработки включают в себя:

– улучшение качества организации работы администратора частной клиники, путем оптимизации процессов, связанных с записью на прием к врачу, ведением учета и расписания работы врачей, учета пациентов, а также выставления счетов и проведения финансовых операций;

– повышение эффективности работы персонала клиники, ускорение процесса обслуживания пациентов и улучшение их удовлетворенности услугами;

– упрощение процесса управления клиникой, путем автоматизации и централизации учета и анализа данных, ускорения принятия решений, снижения рисков ошибок и повышения уровня контроля;

2 Разработка технического проекта на основе анализа требований

2.1 Определение спецификаций программного обеспечения

Рассмотрим определение вариантов использования (прецедентов).

Система «ZimAdmin» в первую очередь требуется администратору частной клиники.

Основными участниками системы является сам администратор, который напрямую взаимодействует с системой, и пациент, который имеет возможность взаимодействовать с системой через администратора. Соответственно основными прецедентами будут являться следующие.

Прецеденты для администратора:

– аутентификация;

– регистрация;

– авторизация;

– просмотр информации о текущих и предстоящих приёмах;

– ведение базы данных пациентов;

– формирование расписания приёма пациентов;

– внесение изменений в расписание приёма;

– отмена приёма;

– редактирование приёма;

– регистрация пациента на приём к врачу;

– обработка заявок на вызов врача на дом.

Прецеденты для пациента:

– запись на приём;

– осуществление звонка на подачу заявки на вызов врача на дом.

Представим диаграмму вариантов использования средствами Visio 2019 professional, для системы «ZimAdmin» на рисунке 1.

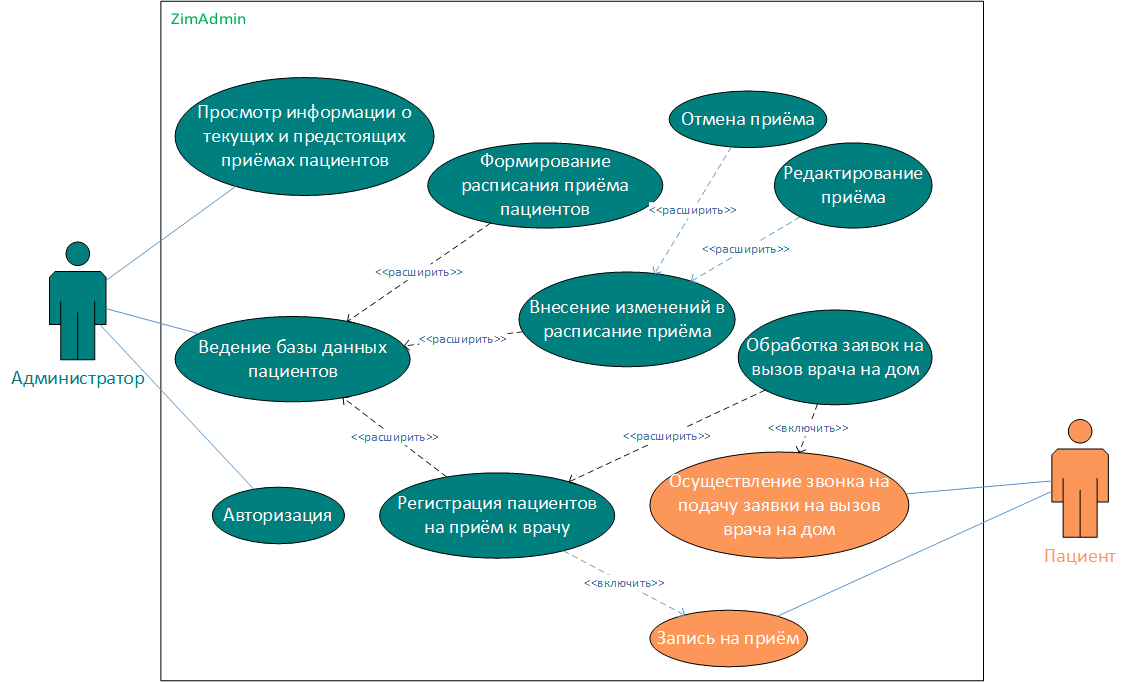


Рисунок 1 – Диаграмма вариантов использования программной системы «ZimAdmin»

Далее подробно опишем требования к реализации базовых вариантов использования, представленных на рисунке 1.

В таблице 1 представлено описание раздела сценария варианта использования «Регистрация пациентов на приём к врачу»

Таблица 1 – Раздел сценария варианта использования «Регистрация пациентов на приём к врачу»

|  |  |
| --- | --- |
| Вариант использования | Регистрация пациентов на приём к врачу |
| Актёры | Администратор, пациент |
| Краткое описание | Администратор авторизируется в системе, пациент осуществляет запись на приём, администратор регистрирует пациента на приём к врачу, а затем сообщает пациенту время и дату приёма. |
| Цель | Регистрация пациента на приём |
| Тип | Базовый |
| Ссылка на другие варианты использования | Включает в себя варианты использования: запить на приём, авторизация |

В таблице 2 описана последовательность действий, приводящая к успешному выполнению варианта использования «Регистрация пациентов на приём к врачу».

Таблица 2 – Сценарий успешного выполнения варианта использования «Регистрация пациентов на приём к врачу»

|  |  |
| --- | --- |
| Действия актёров | Отклик системы |
| 1. Администратор авторизируется в системе.  Исключение №1. Введённые данные не верные. | 2. Система сверяет данные в БД. |
| 3. Пациент записывается на приём  4. Администратор запрашивает у системы список врачей и их график | 5. Система выводит данные о врачах |
| 6. Администратор выбирает время и врача для записи пациента на приём.  7. Администратор регистрирует пациента на приём к врачу. | 8. Система записывает данные в БД. |
| 9. Администратор сообщает пациенту время и дату приёма. |  |

Далее опишем сценарий исключений.

В таблице 3 представлена обработка исключительной ситуации для варианта использования «Регистрация пациентов на приём к врачу».

Таблица 3 – Обработка исключительной ситуации для варианта использования «Регистрация пациентов на приём к врачу»

|  |  |
| --- | --- |
| Действия актёров | Отклик системы |
| Исключение №1. Введённые данные не верные. | |
|  | 2. Система требует повторного ввода данных администратора. |

На рисунке 2 описан алгоритм реализации описанных сценариев для варианта использования «Регистрация пациентов на приём к врачу» с помощью диаграммы деятельности».

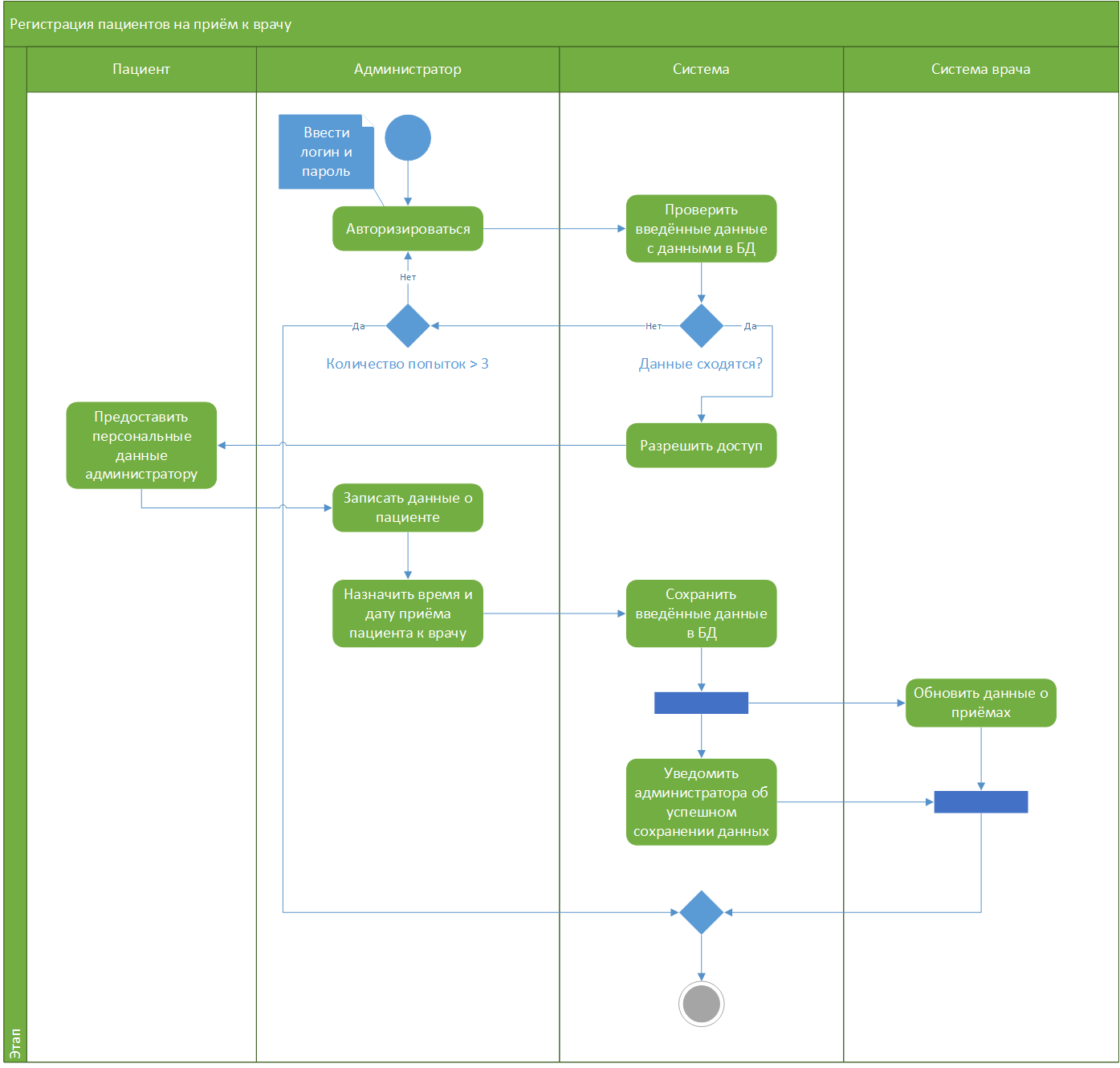


Рисунок 2 – Диаграмма деятельности для варианта использования «Регистрация пациентов на приём к врачу»

Следующим этапом является описание требования к реализации сценариев с учётом динамики во времени на уровне сообщений. Для демонстрации были разработаны диаграммы последовательности.

На рисунке 4 представлена диаграмма последовательности для варианта использования «Регистрация пациентов на приём к врачу»

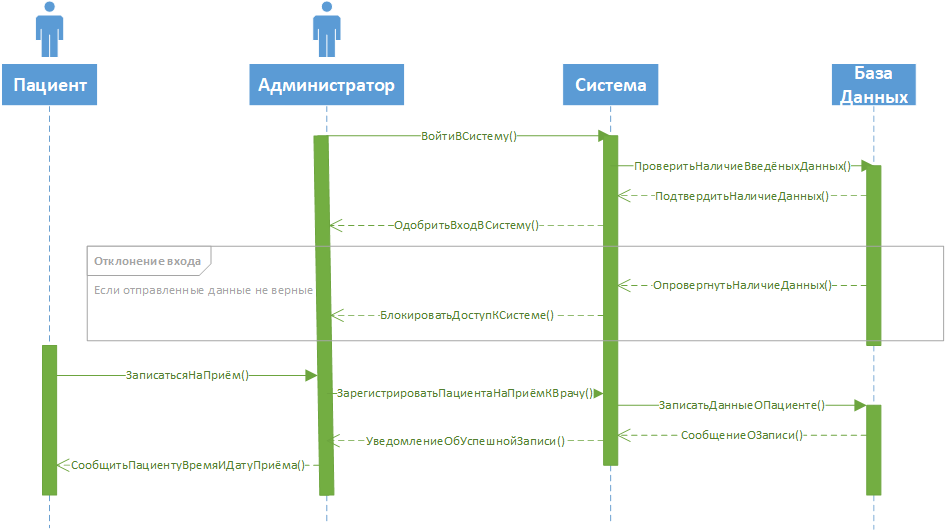


Рисунок 3 – Диаграмма последовательности для варианта использования «Регистрация пациента на приём к врачу»

Следом опишем ещё один наиболее важный вариант использования «Обработка заявок на вызов врача на дом».

В таблице 4 представлено описание сценария варианта использования «Обработка заявок на вызов врача на дом».

Таблица 4 – Сценария варианта использования «Обработка заявок на вызов врача на дом»

|  |  |
| --- | --- |
| Вариант использования | Обработка заявок на вызов врача на дом |
| Актёры | Администратор, пациент |
| Краткое описание | Пациент делает звонок для оставления заявки на вызов врача на дом, администратор обрабатывает заявку и назначает время и врача для вызова. |
| Цель | Обработка заявки вызова врача на дом |
| Тип | Базовый |
| Ссылка на другие варианты использования | Включает в себя варианты использования: регистрация пациента на приём, осуществление звонка на подачу заявления на вызов врача на дом. |

В таблице 5 описана последовательность действий, приводящая к успешному выполнению варианта использования «Обработка заявок на вызов врача на дом».

Таблица 5 – Сценарий успешного выполнения варианта использования «Обработка заявок на вызов врача на дом»

|  |  |
| --- | --- |
| Действия актёров | Отклик системы |
| 1. Пациент делает звонок для подачи заявки на вызов врача на дом | 2. Администратор обрабатывает заявку пациента. |
| 3. Администратор выбирает свободного врача для вызова.  Исключение №1. Свободного врача нет.  4. Администратор регистрирует пациента на приём. | 5. Система записывает данные в БД. |
| 6. Администратор сообщает пациенту время прибытия врача. |  |

Далее опишем сценарий исключений.

В таблице 6 представлена обработка исключительной ситуации для варианта использования «Обработка заявок на вызов врача на дом».

Таблица 6 – Обработка исключительной ситуации для варианта использования «Обработка заявок на вызов врача на дом»

|  |  |
| --- | --- |
| Действия актёров | Отклик системы |
| Исключение №1. Свободного врача нет. | |
| 5. Администратор вызывает бригаду скорой помощи. | 6. Система записывает данные в БД |

На рисунке 4 описан алгоритм реализации описанных сценариев для варианта использования «Обработка заявок на вызов врача на дом».

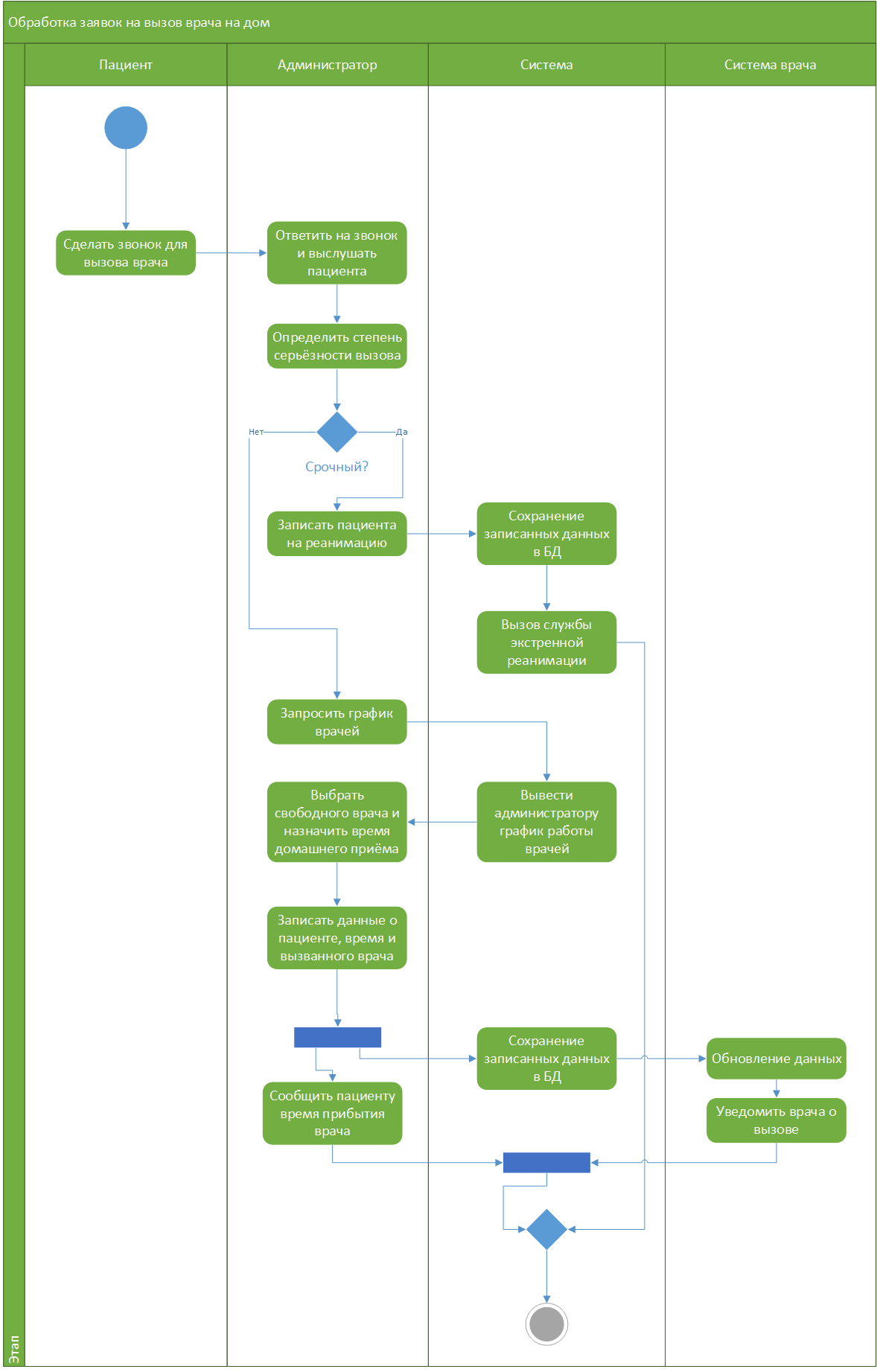


Рисунок 4 – Диаграмма деятельности для варианта использования «Обработка заявок на вызов врача на дом»

Так же опишем вариант использования «Обработка заявок на вызов врача на дом» на уровне сообщений.

На рисунке 5 представлена разработанная диаграмма последовательности для выбранного прецедента.

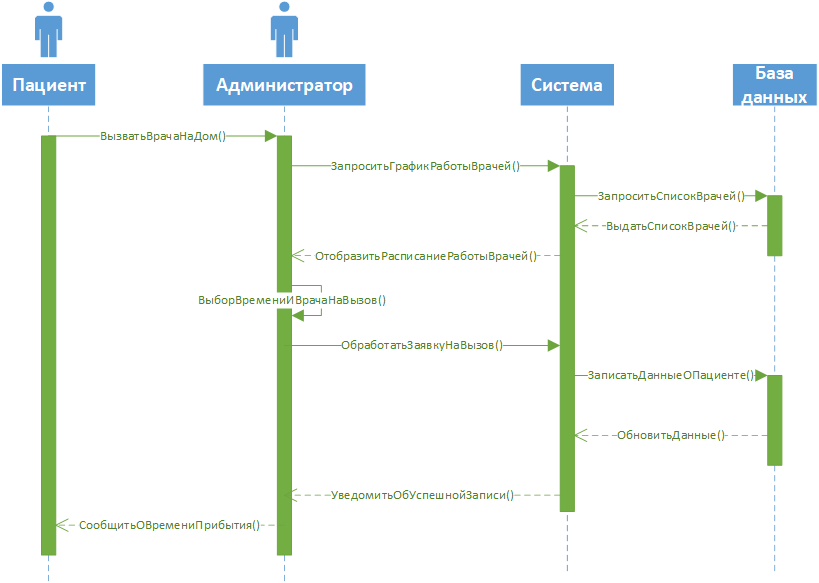


Рисунок 5 – Диаграмма последовательности для варианта использования «Обработка заявок на вызов врача на дом»

В диаграммах так же происходит выход за пределы разрабатываемой системы. Это сделано для того, чтобы продемонстрировать, как администратор может взаимодействовать с врачами и как происходит обновление данных в других связанных системах.

2.2 Проектирование модели данных и диаграммы классов

На основе составленного технического заданий была разработана модель данных системы и представлена в виде ER-диаграммы, представленная на рисунке 6, которая чётко описывает логическую структуру данных, и на основе которой, в дальнейшем, будет спроектирована полноценная БД.

Диаграмма была построена средством MS Visio 2019

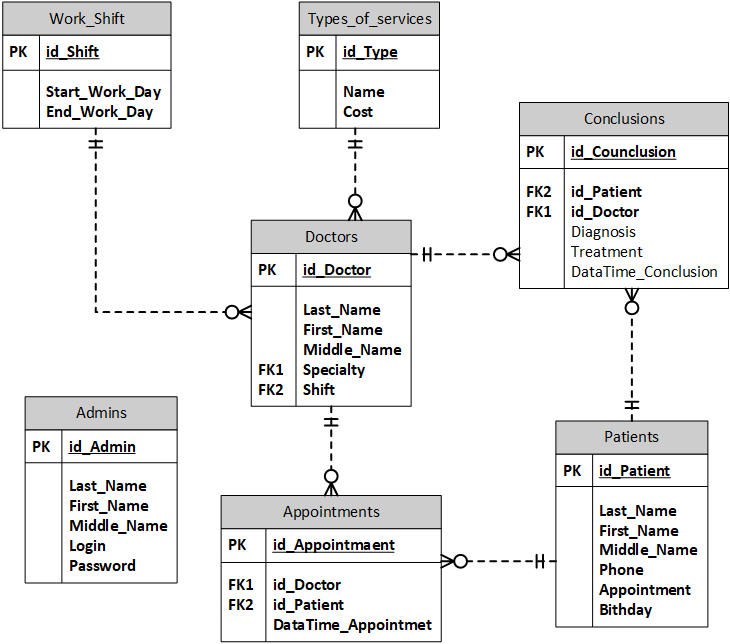


Рисунок 6 – ER-диаграмма для программной системы «ZimAdmin»

В таблице 7 представлены сведения о сущностях и их связях в ER-диаграмме.

Таблица 7 – Характеристики сущностей и связей в ER-диаграмме системы «ZimAdmin»

|  |  |  |  |
| --- | --- | --- | --- |
| Сущность | Назначение сущности | Ключ | Характеристики связей |
| Admins | Содержит данные об администраторе:  – last\_Name;  – first\_Name;  – middle\_Name;  – login;  – password. | Id\_Admin (PK); | – |
| Doctors | Содержит данные о докторе:  – last\_Name;  – first\_Name;  – middle\_Name;  – specialty;  – shift. | Id\_Doctor (PK);  Specialty (FK1);  Shift (FK2) | Types\_of\_services – Doctors (1..n);  Work\_Shift – Doctors (1..n);  Doctor – Conclusions (1..n);  Doctor – Appointments (1..n). |

Продолжение таблицы 7

|  |  |  |  |
| --- | --- | --- | --- |
| Work\_Shift | Содержит данные о расписании:  – start\_Work\_Day;  – end\_work\_Day. | Id\_Shift (PK) | Work\_Shift – Doctors (1..n) |
| Types\_of\_  services | Содержит данные об услуге:  – name;  – cost. | Id\_Type (PK) | Types\_of\_services – Doctors (1..n) |
| Patients | Содержит данные о пациенте:  – last\_Name;  – first\_Name;  – middle\_Name;  – phone  – appointment | Id\_Patient (PK) | Patient – Conclusions (1..n);  Patient – Appointments (1..n). |
| Conclusions | Содержит данные о заключении пациента:  – id\_Patient;  – id\_Doctor;  – diagnosis;  – treatment;  – dataTime\_Conclusion | Id\_  Conclusion (PK);  Id\_Doctor (FK1);  Id\_Patient (FK2) | Patient – Conclusions (1..n);  Doctor – Conclusions (1..n). |
| Appointments | Содержит данные о приёме:  – id\_Doctor;  – id\_Patient;  – dataTime\_Appointment. | Id\_  Appointment (PK);  Id\_Doctor (FK1);  Id\_Patient (FK2) | Patient – Appointments (1..n);  Doctor – Appointments (1..n). |

На следующем этапе была спроектирована диаграмма классов, описывающая требования к внутренней организации ПО на уровне интерфейсов и классов, их атрибутов и методов.

На рисунке 7 представлена диаграмма классов для системы «ZimAdmin».

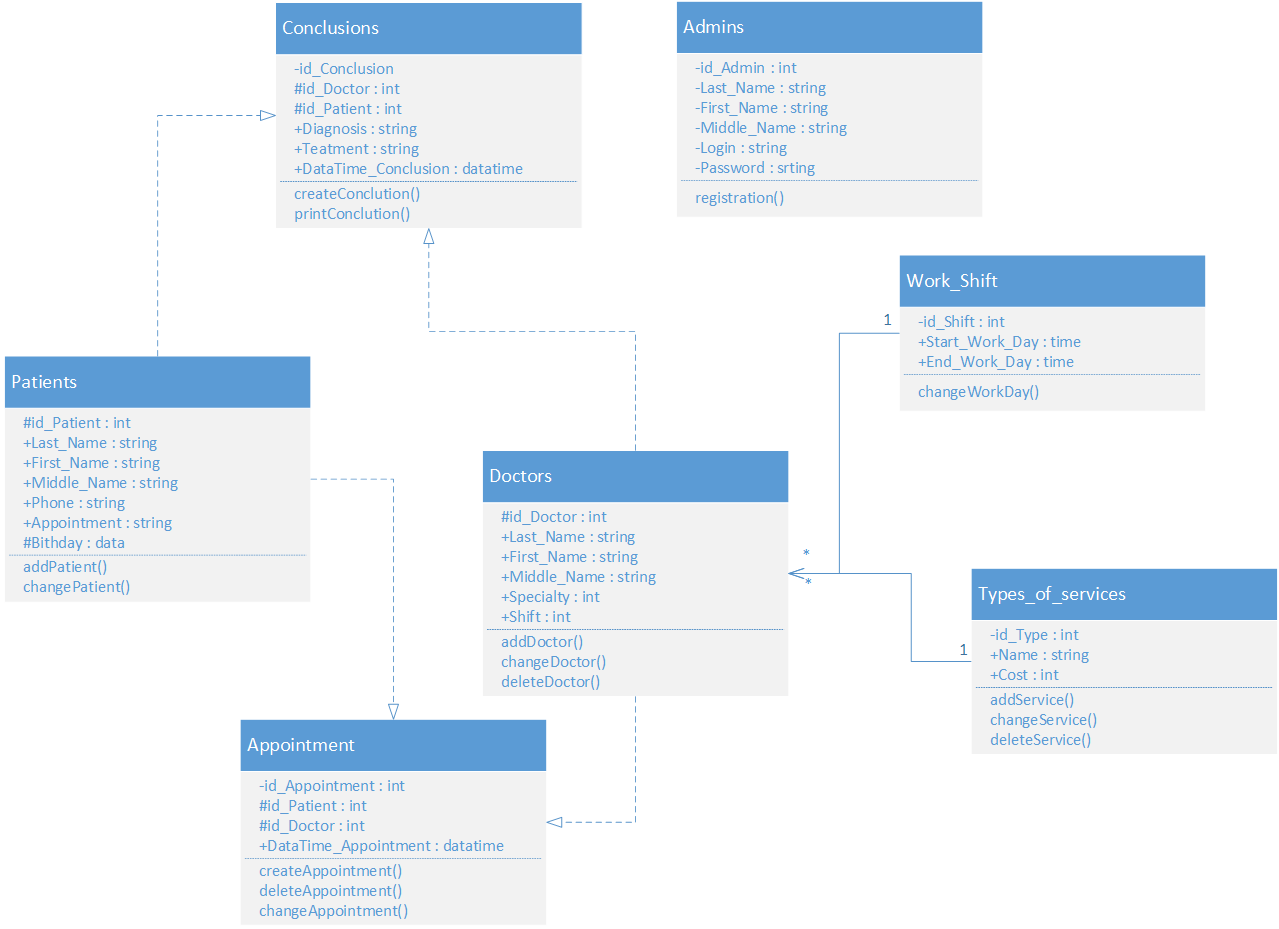


Рисунок 7 – Диаграмма классов системы «ZimAdmin»

Далее распишем несколько классов, их атрибуты, методы и связи.

Класс "Doctor" представляет собой модель врача и содержит следующие атрибуты:

– id\_Doctor: уникальный идентификатор врача;

– Last\_Name: фамилия врача;

– First\_Name: имя врача;

– Middle\_Name: отчество врача;

– Specialty: специализация врача;

– Shift: номер смены, в которой работает врач.

Также класс "Doctor" содержит следующие методы:

– addDoctor(): добавление нового врача в базу данных;

– deleteDoctor(): удаление врача из базы данных;

– changeDoctor(): изменение данных о враче в базе данных.

Класс "Patient" представляет информацию о пациентах в системе частной клиники. Каждый объект класса содержит следующие атрибуты:

– id\_Patient: уникальный идентификатор пациента;

– Last\_Name: фамилия пациента;

– First\_Name: имя пациента;

– Middle\_Name: отчество пациента;

– Phone: номер телефона пациента;

– Appointment: назначение пациента;

– Birthday: дата рождения пациента.

Класс "Patient" имеет следующие методы:

– addPatient(): метод для добавления нового пациента в базу данных системы;

– changePatient(): метод для изменения информации о существующем пациенте в базе данных системы.

Класс «Appointment» является связующим звеном между классами «Patient» и «Doctor». Он содержит информацию о записи пациента на прием к конкретному врачу в определенное время.

Атрибуты:

– id\_Appointment является уникальным идентификатором записи на прием;

– id\_Patient и id\_Doctor связывают запись на прием с конкретным пациентом и врачом соответственно;

– DataTime\_Appointment: хранит дату и время приема пациента к врачу.

Методы:

– createAppointment(): создает новую запись на прием и связывает её с определенным пациентом и врачом, а также указывает дату и время приема;

– deleteAppointment(): удаляет запись на прием;

– changeAppointment(): изменяет информацию о записи на прием, например, изменяет дату и время приема, или связывает запись на прием с другим пациентом или врачом.

2.3 Конструирование прототипа

Все компоненты приложения должны иметь единый согласованный внешний вид, соответствующий руководству по стилю и следующим требованиям:

– минимальные размеры окна должны быть равны 760px в высоту и 1420px в ширину;

– главные элементы навигации должны располагаться в одной боковой панели управления;

– для навигации по приложению должен быть использован элемент управления «Меню», для отображения данных из БД «DataGrid»;

– последовательный переход фокуса по элементам интерфейса (по нажатию клавиши TAB);

– общая компоновка логична, понятна и проста в использовании;

– последовательный пользовательский интерфейс, позволяющий перемещаться между существующими окнами или страницами в приложении;

Руководство по стилю:

– каждый элемент навигации должен иметь подпись и соответствующую по смыслу иконку.

– тип шрифта – Consolas;

– цветовая схема предусматривает использование в качестве основного фона – белый цвет RGB (255, 255, 225), в качестве дополнительного – цвет RGB (46, 51, 58), для акцентирования внимания пользователя на целевое действие интерфейса – цвет RGB (100, 149, 237).



Рисунок 8 – Логотип приложения

На рисунке 9 представлен прототип страницы профиля администратора.

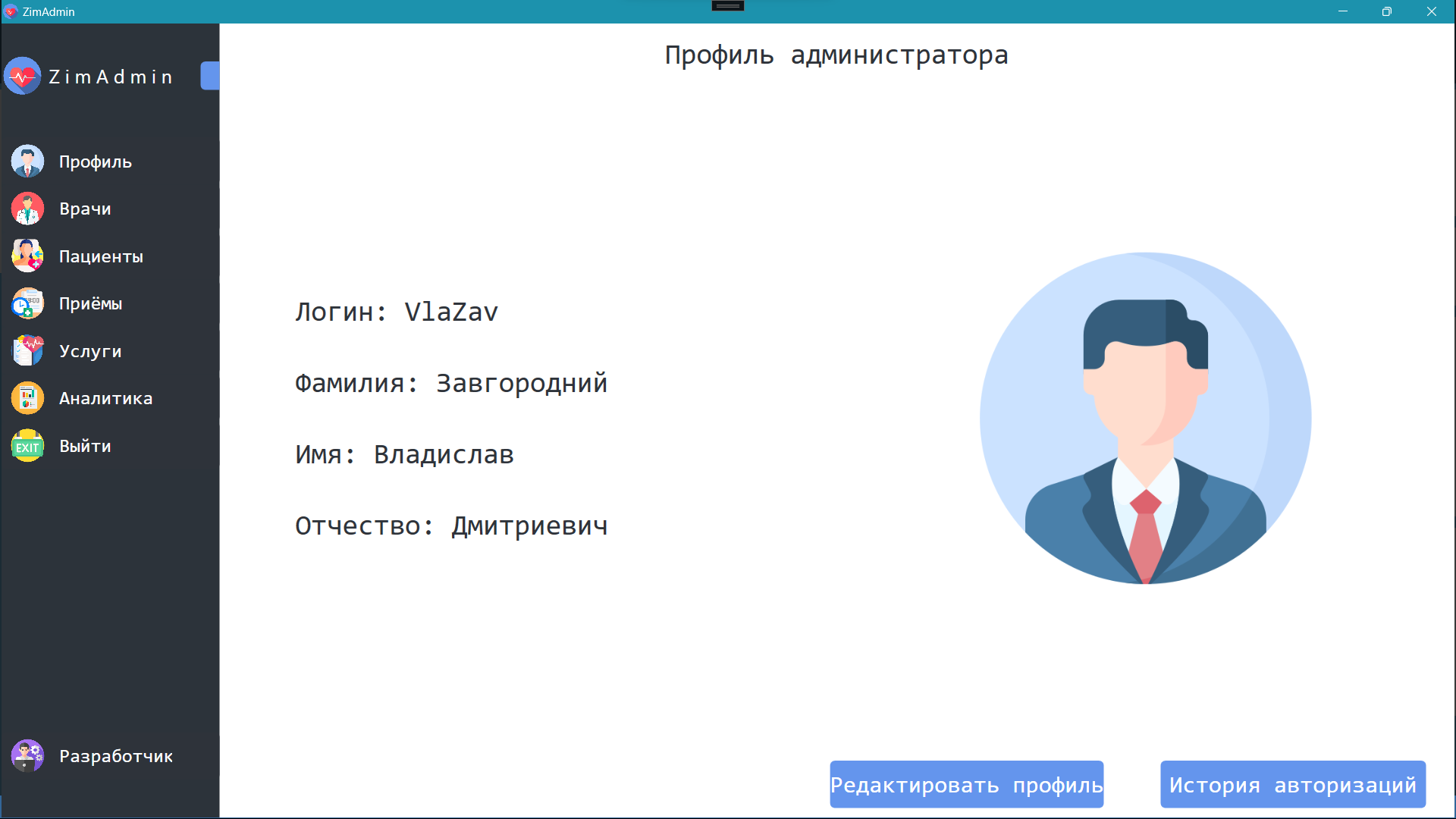


Рисунок 9 – Страница профиля администратора

На последующих рисунках будут представлены все основные страницы приложения.

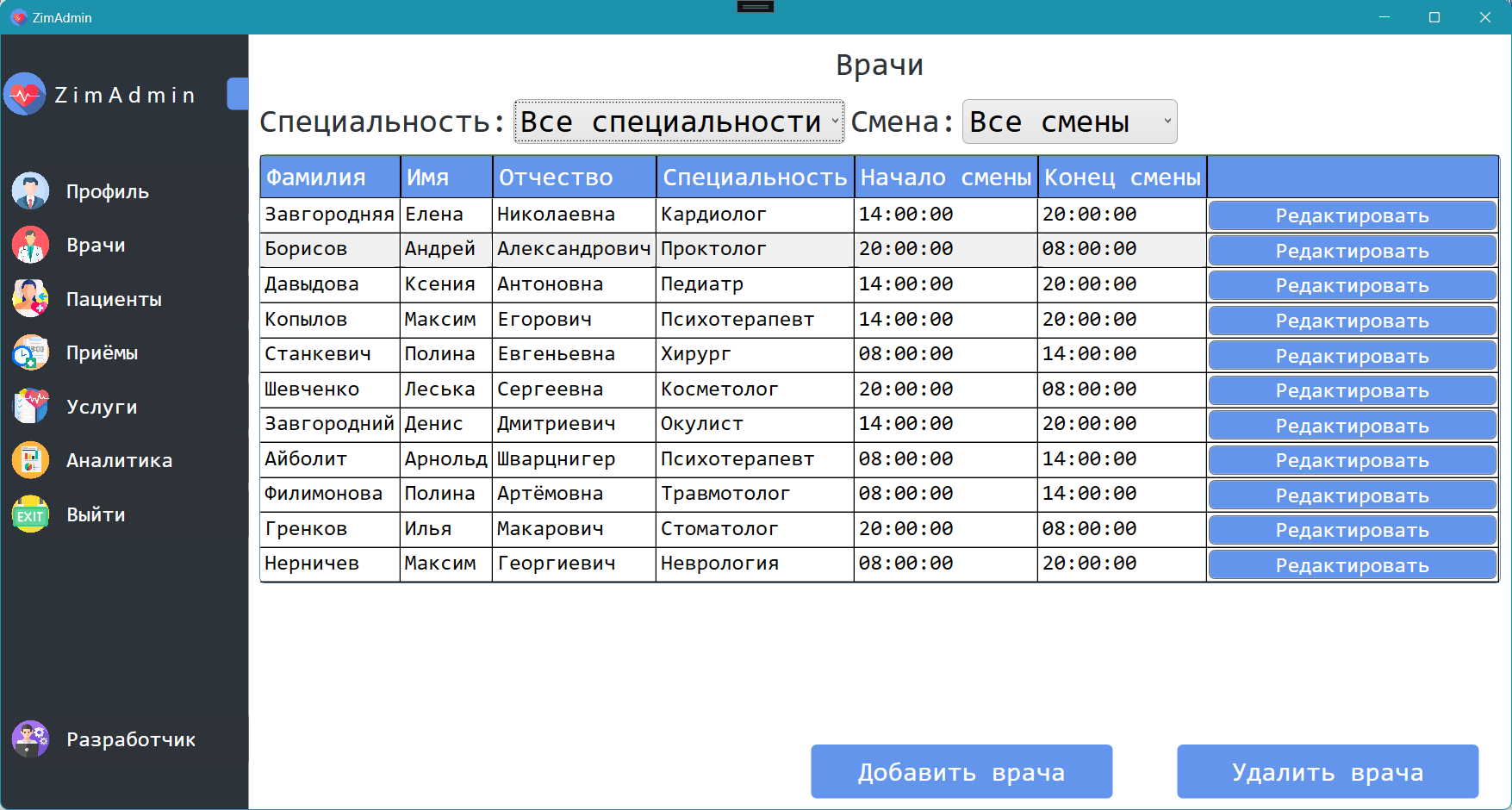


Рисунок 10 – Страница списка врачей

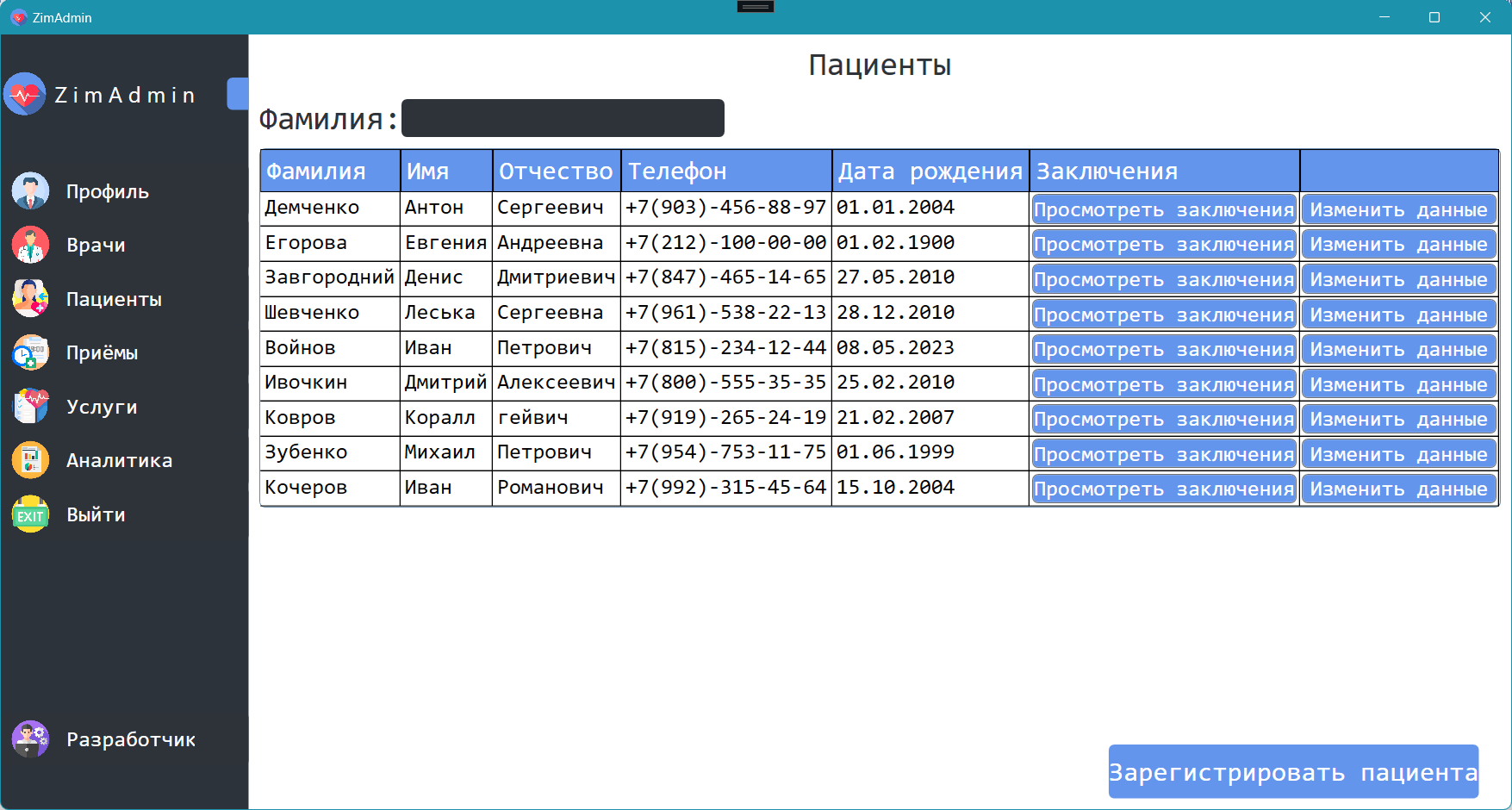


Рисунок 11 – Страница списка пациентов

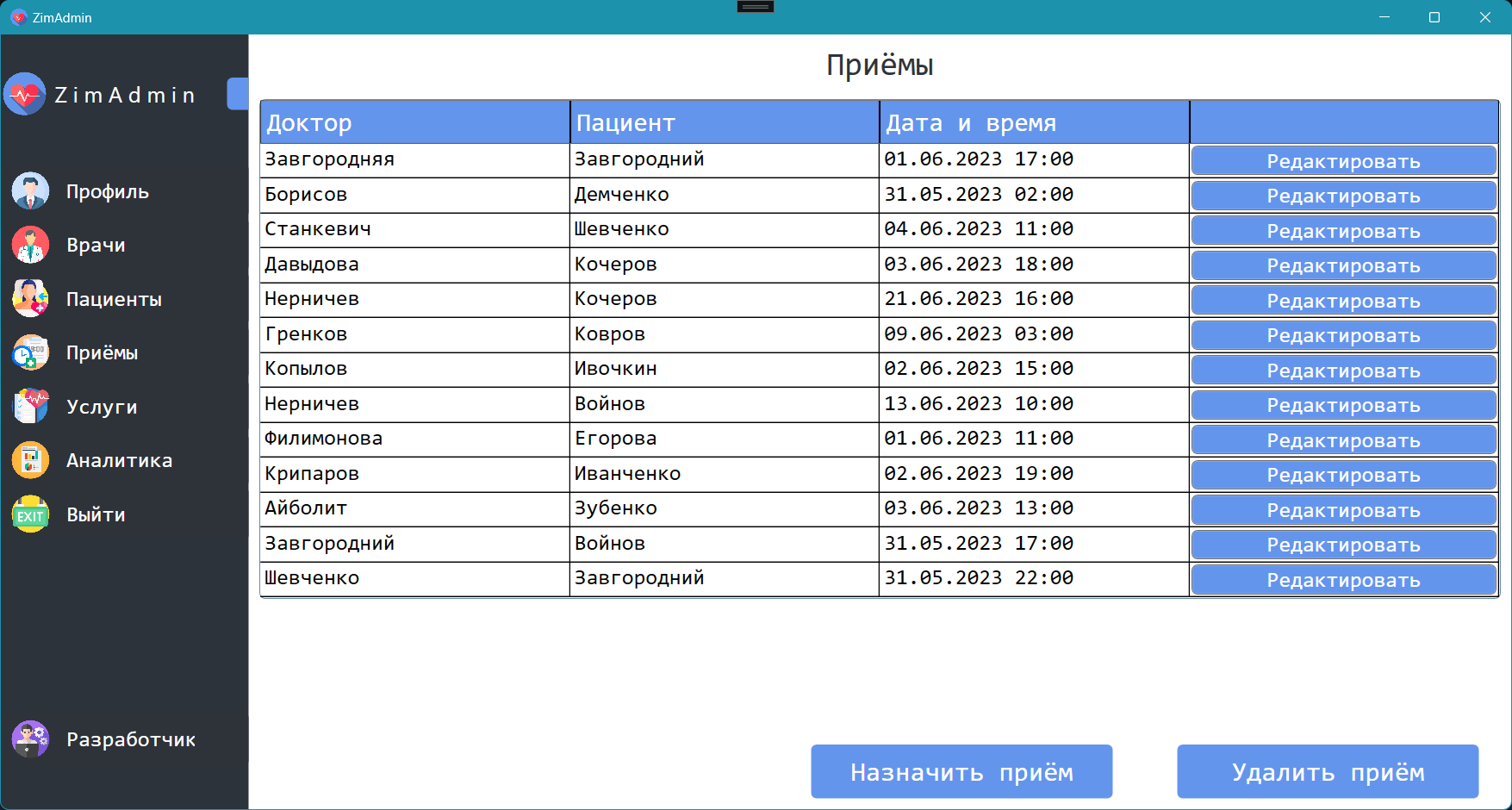


Рисунок 12 – Страница списка приёмов



Рисунок 13 – Страница списка услуг

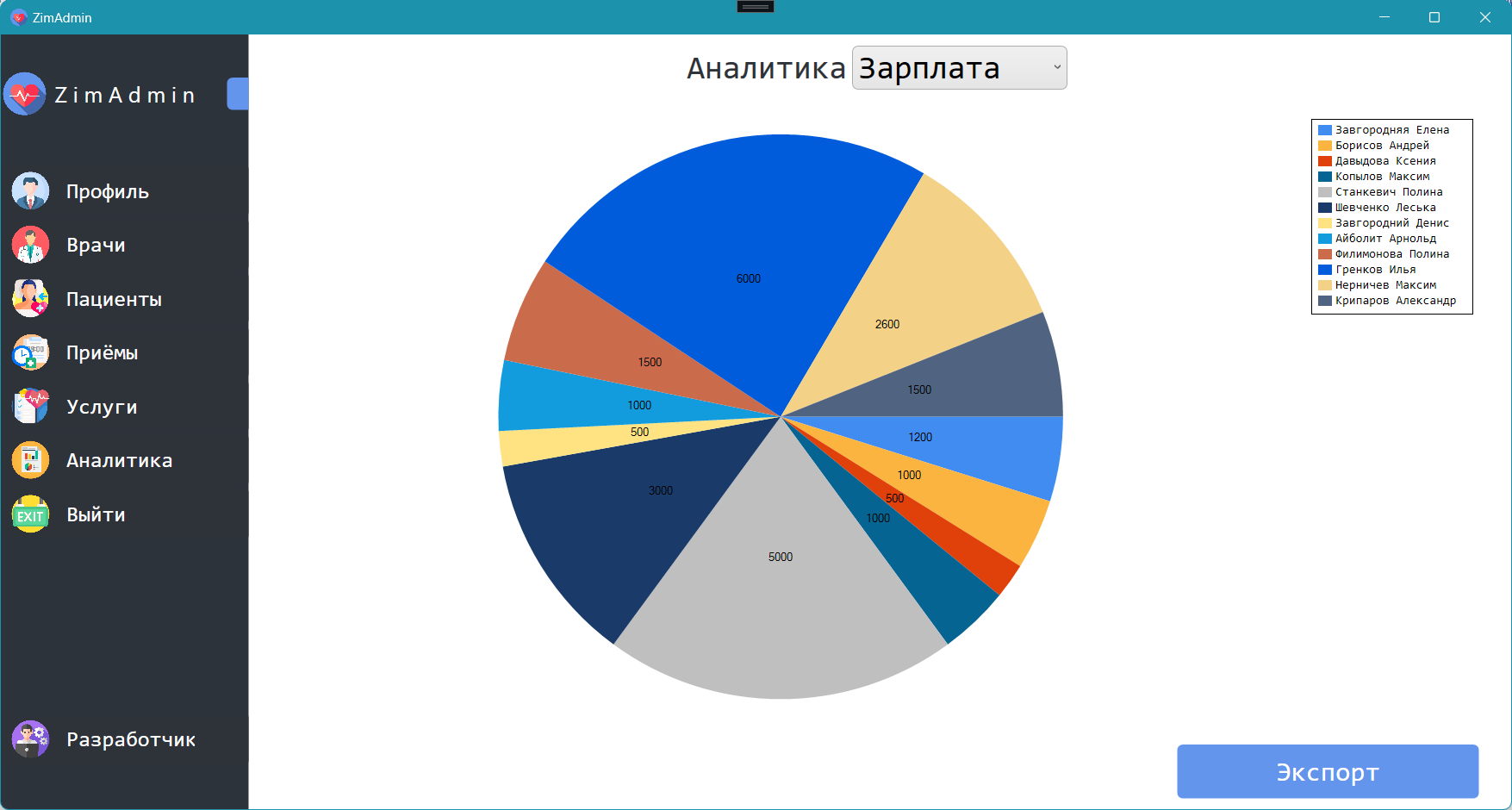


Рисунок 14 – Страница аналитики

3 Реализация

3.1 Обоснование выбора средств разработки

Для разработки приложения ZimAdmin, язык C# и технология WPF являются предпочтительными по следующим причинам:

– широкая поддержка и популярность: C# является одним из наиболее популярных языков программирования, особенно в контексте разработки приложений для платформы .NET. Он имеет обширную базу разработчиков и сообщества, что обеспечивает доступ к обширным ресурсам, библиотекам и фреймворкам. WPF, в свою очередь, является одной из основных технологий для разработки графических интерфейсов на платформе .NET, что обеспечивает большую поддержку и интеграцию с другими инструментами и ресурсами.

– Продвинутые возможности графического интерфейса: WPF обладает мощными возможностями для создания современных и интерактивных графических интерфейсов пользователя. Он поддерживает векторную графику, анимации, стилизацию, мультимедиа и другие функциональные возможности, которые могут значительно улучшить пользовательский опыт.

– Кросс-платформенность: C# и WPF позволяют создавать кросс-платформенные приложения с использованием платформы .NET Core. Это обеспечивает возможность разработки приложения, которое может работать на различных операционных системах, включая Windows, macOS и Linux.

Сравнивая с другими языками программирования и технологиями, C# и WPF обладают рядом преимуществ:

– C# обладает простым и понятным синтаксисом, что делает его относительно легким для изучения и использования.

– C# и WPF полностью интегрированы в экосистему .NET, что обеспечивает доступ к широкому спектру инструментов, библиотек и фреймворков.

– C# имеет мощную систему типов и средства обработки исключений, что способствует разработке надежных и безопасных приложений.

– WPF обладает большими возможностями для создания пользовательского интерфейса, включая стилизацию, анимацию и поддержку мультимедиа.

– WPF обеспечивает легкое связывание данных и реактивное обновление интерфейса при изменении данных.

– WPF обладает хорошей производительностью и возможностями оптимизации, что важно для разработки ресурсоемких приложений.

В целом, выбор языка C# и технологии WPF для разработки приложения ZimAdmin обеспечивает высокую производительность, мощные возможности для создания интерфейса и интеграцию с широким спектром инструментов и ресурсов, что делает их оптимальным выбором.

Далее, в таблице 8, представлена сравнительная характеристика языка программирования C# с другими популярными языками программирования.

Таблица 8 – Сравнительная характеристика языка C# с другими языками программирования.

|  |  |  |  |
| --- | --- | --- | --- |
| **Критерий** | **C#** | **Java** | **C++** |
| Тип языка | Объектно-ориентированный, событийно-ориентированный | Объектно-ориентированный | Мультипарадигменный |
| Платформа | .NET Framework | Java Virtual Machine (JVM) | Зависит от компилятора и операционной системы |
| Графический интерфейс | Windows Presentation Foundation (WPF) | JavaFX, Swing | Qt, MFC, WinAPI |
| Управление памятью | Автоматическое сборка мусора | Автоматическое сборка мусора | Возможны различные подходы (ручное, автоматическое, умные указатели) |
| Безопасность типов | Сильная типизация | Сильная типизация | Слабая или сильная типизация (зависит от использования) |
| Популярность | Широко используется для разработки приложений под Windows | Широко используется для разработки кросс-платформенных приложений | Широко используется для системного и производительного программирования |

3.2 Разработка базы данных в среде СУБД

Для разработки базы данных была выбрана базы данных SQL Server и СУБД SSMS из-за высокой надёжности и стабильности, расширенные возможностей, такие как хранимые процедуры, триггеры, представления и шифрование данных, а так же из-за хорошей масштабируемости и высокой производительности.

Для таблиц БД было выделено 8 сущностей:

– администраторы;

– врачи;

– пациенты;

– заключения;

– приёмы;

– типы услуг;

– рабочие смены;

– история авторизаций.

Далее будут представлены сущности с полями и типами данных.

Таблица "Admins" представляет собой хранение информации о администраторах в системе. Она содержит следующие поля:

– Id\_Admin: INT (PK).

– Last\_Name: NVARCHAR(50);

– First\_Name: NVARCHAR(50);

– Middle\_Name: NVARCHAR(50);

– Login: NVARCHAR(50);

– Password: NVARCHAR(20).

Таблица "Admins" позволяет хранить информацию о каждом администраторе, включая его персональные данные и данные для авторизации. Эти данные могут быть использованы для аутентификации администратора и управления его доступом к функциональности системы.

Таблица "Appointments" представляет собой хранение информации о назначениях (встречах) между врачами и пациентами. Она содержит следующие поля:

– Id\_Appointment: INT (PK);

– Id\_Doctor: INT (FK);

– Id\_Patient: INT (FK);

– DateTime\_Appointment: DATETIME.

Таблица "Appointments" позволяет отслеживать назначения между врачами и пациентами. Каждая запись в таблице представляет конкретное назначение, указывающее на определенного врача и пациента, а также дату и время назначения.

Таблица "Conclusions" предназначена для хранения информации о заключениях в медицинских записях. Она содержит следующие поля:

– Id\_Conclusion: Уникальный идентификатор заключения (первичный ключ).

– Id\_Patient: Идентификатор пациента, связанный с данным заключением (внешний ключ).

– Id\_Doctor: INT (PK);

– Complaint: NVARCHAR(MAX);

– Life\_History: NVARCHAR(MAX);

– Objective\_Status: NVARCHAR(MAX);

– Diagnosis: NVARCHAR(MAX);

– Treatment: NVARCHAR(MAX);

– DateTime\_Conclusion: DATETIME.

Таблица "Conclusions" позволяет хранить информацию о состоянии пациента, проблемах, выявленных врачом, поставленном диагнозе и рекомендациях по лечению. Эти данные играют важную роль в ведении медицинских записей, позволяют мониторить состояние пациентов и предоставлять необходимую информацию врачам для принятия решений о дальнейшем лечении и уходе за пациентом.

Таблица "Doctors" предназначена для хранения информации о врачах в частной клинике. Она содержит следующие поля:

– Id\_Doctor: INT (PK);

– Last\_Name: NVARCHAR(50);

– First\_Name: NVARCHAR(50);

– Middle\_Name: NVARCHAR(50);

– Specialty: INT (FK);

– Shift: INT (FK).

Таблица "Doctors" позволяет хранить информацию о врачах, их персональных данных, специальностях и графике работы. Эти данные необходимы для организации работы в клинике, назначения приемов пациентам и управления расписанием врачей.

Таблица "Patients" предназначена для хранения информации о пациентах в частной клинике. Она содержит следующие поля:

– Id\_Patient: INT (PK);

– Last\_Name: NVARCHAR(50);

– First\_Name: NVARCHAR(50);

– Middle\_Name: NVARCHAR(50);

– Phone: Номер NVARCHAR(17);

– Birthday: DATETIME;

Таблица "Patients" позволяет сохранять данные о пациентах, их персональных данных, контактной информации и дате рождения. Эти данные необходимы для идентификации пациентов, связи с ними и ведения медицинской истории.

Таблица "Types\_of\_services" предназначена для хранения информации о типах услуг, предоставляемых в частной клинике. Она содержит следующие поля:

– Id\_Type: INT (PK);

– Name: NVARCHAR(50);

– Cost: INT.

Таблица "Types\_of\_services" позволяет сохранять информацию о различных видов услуг, которые предоставляет клиника. Каждый тип услуги имеет свое уникальное наименование и стоимость. Эти данные позволяют клинике управлять списком доступных услуг, определять их цены и предоставлять информацию пациентам о доступных вариантах лечения и стоимости.

Таблица "Work\_shift" предназначена для хранения информации о сменах работы сотрудников в частной клинике. Она содержит следующие поля:

– Id\_Shift: INT (PK);

– Start\_Work\_Day: TIME(7);

– End\_Work\_Day: TIME(7);

– Number: NVARCHAR(10).

Таблица "Work\_shift" позволяет управлять расписанием работы сотрудников клиники. Каждая смена имеет свое уникальное идентификатор и определенное время начала и окончания рабочего дня.

Таблица "Authorization\_history" предназначена для хранения информации об истории авторизации администраторов в системе. Она содержит следующие поля:

– Id\_Auth: INT (PK);

– Id\_Admin: INT (FK);

– Date\_Auth: DATETIME.

Таблица "Authorization\_history" позволяет отслеживать и анализировать историю авторизаций администраторов в системе.

3.3 Описание программных модулей

Приложение состоит из нескольких ключевых программных модулей. Модули не должны зависеть друг от друга, т.е. если один из модулей (к примеру, добавление/удаление/изменение информации о врачах), ложится и не функционирует, остальные должны работать в штатном режиме и стабильно выполнять свои функции.

На рисунке 15 показана структура проекта.

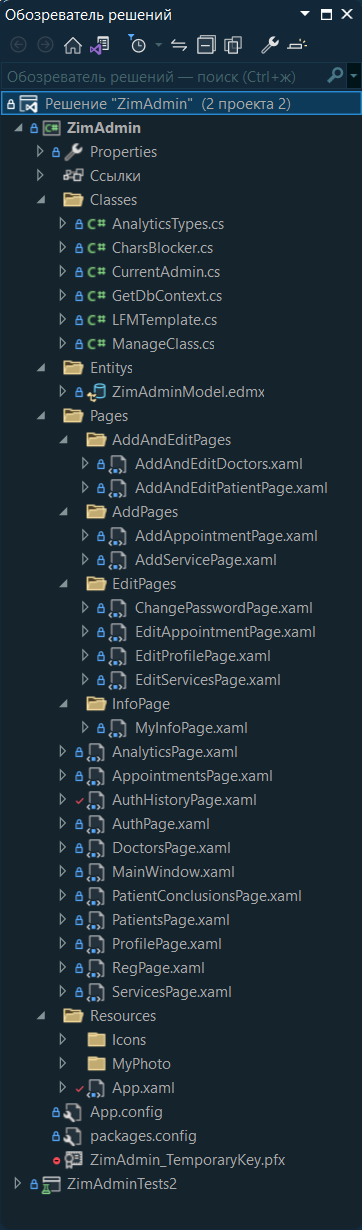


Рисунок 15 – Структура проекта

Далее приведена таблица с описанием программных модулей.

Таблица 9 – Описание программных модулей

3.3.1 Модуль регистрации/авторизации

Описание: Этот модуль отвечает за функциональность регистрации новых пользователей и авторизацию уже зарегистрированных пользователей в системе.

Функции:

– регистрация новых пользователей путем ввода необходимых регистрационных данных, таких как фамилия, имя, отчества, логин и пароль;

– проверка уникальности логина пользователя для предотвращения создания дублирующихся учетных записей;

– хранение информации о зарегистрированных пользователях в базе данных;

– авторизация пользователей путем проверки введенных данных и сравнения их с данными в базе данных;

– предоставление доступа к функциональности системы после успешной авторизации.

3.3.2 Модуль добавления/изменения/удаления информации о врачах

Описание: Этот модуль отвечает за управление информацией о врачах в системе.

Функции:

– добавление новых врачей в систему путем ввода соответствующих данных, таких как фамилия, имя, отчество, специальность и рабочий график;

– изменение данных существующих врачей, позволяющее обновлять информацию, например, при изменении графика работы или других персональных данных;

– удаление врачей из системы при необходимости;

– хранение информации о врачах в базе данных.

3.3.3 Модуль добавления/изменения/удаления информации о пациентах

Описание: Этот модуль отвечает за управление информацией о пациентах в системе.

Функции:

– Добавление новых пациентов в систему путем ввода соответствующих данных, таких как фамилия, имя, отчество, контактные данные и дата рождения.

– Изменение данных существующих пациентов, позволяющее обновлять информацию, например, при изменении контактных данных или других персональных данных.

– Удаление пациентов из системы при необходимости.

– Проверка наличия зависимостей, таких как приемы, связанные с конкретным пациентом, и предотвращение удаления пациента с активными зависимостями.

– Хранение информации о пациентах в базе данных.

3.3.4 Модуль добавления/изменения/удаления информации о приемах

Описание: Этот модуль отвечает за управление информацией о приемах пациентов к врачам.

Функции:

– Добавление новых приемов путем выбора врача, пациента и указания даты и времени приема.

– Изменение данных существующих приемов, позволяющее изменять дату и время приема.

– Удаление приемов из системы при необходимости.

– Хранение информации о приемах в базе данных.

– Предоставление возможности просмотра расписания приемов врачей и информации о назначенных приемах для пациентов.

3.3.5 Модуль аналитики

Модуль аналитики является важной частью приложения для частной клиники и предназначен для обработки и анализа данных, собранных в рамках клинической деятельности. Он предоставляет пользователю возможность получать ценную информацию и статистику, которая помогает принимать информированные решения и оптимизировать работу клиники. Модуль аналитики обеспечивает клинике полезный инструмент для мониторинга и анализа её деятельности, что позволяет повысить эффективность, качество услуг и принимать обоснованные управленческие решения.

4 Тестирование программных модулей

На данном этапе, в целях тестирования функционирования отдельных модулей были написаны 4 модульных теста, которые проверяют работоспособность обработчика ввода обязательных полей (имени, фамилии, отчества). На рисунке 15 представлен программный код модульных тестов.



Рисунок 16 – Программный код модульных тестов

Ниже, на рисунке 16, приведён обозреватель тестов, который показывает успешность прохождения всех разработанных тестов.

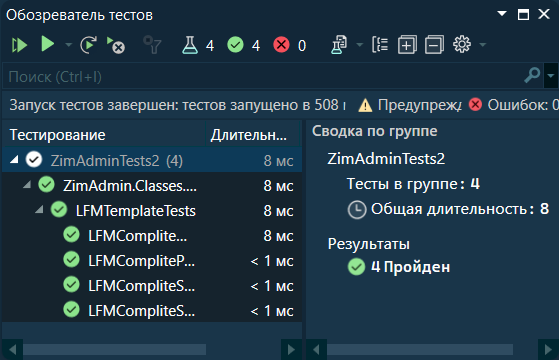


Рисунок 17 – Результат выполнения модульных тестов

5 Эксплуатационная документация

Установка и настройка системы:

– запустите установочный файл системы ZimAdmin и следуйте инструкциям мастера установки для установки программы на ваш компьютер;

– после установки системы запустите приложение.

Авторизация в системе:

– для авторизации в системе введите ваш логин и пароль в соответствующие поля;

– в случае отсутствия аккаунта в системе необходимо зарегистрироваться, для этого на форме авторизации нажмите «Зарегистрироваться», далее введите необходимые данные (фамилия, имя, отчество, логин, пароль) и нажмите «Зарегистрироваться». Далее следуйте первому пункту.

Работа с личным кабинетом:

– для отображения вашего профиля используйте соответствующий раздел в главном меню системы «Профиль»;

– для изменения личных данных нажмите «Редактировать профиль» измените необходимые вам данные и нажмите «Сохранить изменения»;

– для изменения пароля аккаунта на странице редактирования нажмите «Изменить пароль», введите старый пароль и новый пароль в соответствующие поля и нажмите «Изменить»;

Работа с данными пациентов:

– для ведения базы данных пациентов используйте соответствующий раздел в главном меню системы «Пациенты»;

– для добавления нового пациента нажмите кнопку "Добавить пациента" и введите необходимую информацию о пациенте (имя, фамилия, отчество, телефон и дату рождения).

– для поиска пациента воспользуйтесь функцией поиска, указав фамилию пациента;

– для просмотра заключений каждого пациента нажмите на «Просмотреть заключение» рядом с нужным пациентом;

– для вывода заключения на печать, нажмите на «Печать», предварительно выбрав необходимое заключение;

– для сохранения заключения в формате .docx и .pdf воспользуйтесь кнопкой «Сохранить заключение», предварительно выбрав необходимое заключение.

Регистрация пациентов на прием к врачу:

– для регистрации пациента на прием к врачу перейдите в раздел «Приёмы» и нажмите «Назначить приём»;

– выберите нужную дату, врача и пациента, а затем нажмите кнопку «Сохранить»;

– для поиска приёма пациента воспользуйтесь функцией поиска, указав фамилию пациента;

– после регистрации пациента его прием будет отображаться в расписании приемов.

Работа с данными врачей:

– для просмотра данных о врачах клиники воспользуйтесь соответствующим разделом в главном меню (Врачи);

– для редактирования информации о врачах, нажмите на кнопку «Редактировать» рядом с соответствующим врачом, измените необходимые данные и нажмите на «Сохранить». Изменённые данные отобразятся в списке врачей;

– для добавления нового врача в базу, нажмите на кнопку «Добавить врача», на открывшейся странице внесите данные о новом враче (фамилия, имя, отчество, специальность и график работы), нажмите кнопку «Сохранить». Новые врач отобразится в списке врачей;

– для удаления врача из базы данных выберете из списка врача, которого нужно удалить нажмите «Удалить врача» и подтвердите своё решение;

– для поиска врача воспользуйтесь функцией фильтрации, выбрав рабочую смену и специальность.

Работа с данными услуг:

– для просмотра данных об услугах воспользуйтесь соответствующим разделом в главном меню;

– для изменения информации об услуге нажмите на кнопку «Редактировать» рядом с необходимой услугой, измените стоимость и нажмите на кнопку «Сохранить». Сохранённые изменения отобразятся в списке услуг;

– для добавления новой услуги нажмите на кнопку «Добавить услугу», введите название и стоимость услуги и нажмите «Сохранить». Добавленная услуга отобразится в списке услуг.

Аналитика:

– для просмотра аналитики воспользуйтесь соответствующим разделом в главном меню;

– для просмотра другой аналитики воспользуйтесь выпадающим списком, который находится над графиками;

– для экспорта данных в таблицы Excel выберете аналитику зарплат и нажмите кнопку «Экспорт».

Заключение

В ходе разработки информационной системы "ZimAdmin" для частной клиники "Zimamed" было достигнуто основное назначение курсового проекта - создание эффективной и удобной системы, способной упростить и ускорить процессы работы администратора, повысить качество обслуживания пациентов и обеспечить эффективное управление бизнес-процессами клиники.

В процессе разработки системы были сформулированы цели проекта, которые были достигнуты успешно.

Анализ требований позволил определить основные функциональные и нефункциональные требования к системе. В ходе реализации были применены современные методы и технологии программирования, что обеспечило надежность, безопасность и удобство использования системы.

Для обеспечения качества разработки системы было проведено тестирование программных модулей. Это позволило выявить и исправить ошибки, а также убедиться в правильной работе системы в различных сценариях использования.

В результате разработки была создана информационная система "ZimAdmin", которая позволяет администратору клиники эффективно управлять процессами и ресурсами клиники. Система включает функции по записи и учету пациентов, планированию расписания врачей, управлению медицинскими картами и другими важными операциями. Она обладает простым и интуитивно понятным интерфейсом, что облегчает работу администратора и повышает общую эффективность работы клиники.

Для успешной эксплуатации системы была создана соответствующая эксплуатационная документация, которая включает описание функций системы, инструкции по установке и настройке, руководство пользователя и другую необходимую информацию. Это позволит персоналу клиники быстро овладеть системой и использовать её в повседневной работе.

В целом, разработка информационной системы "ZimAdmin" была успешной и полностью соответствует поставленным целям и требованиям. Она представляет собой мощный инструмент для автоматизации работы администратора частной клиники, способствует улучшению качества обслуживания пациентов и эффективному управлению бизнес-процессами. Разработанная система готова к использованию, что позволяет клинике "Zimamed" быть на передовом месте в сфере медицинской автоматизации.

Список используемой литературы:

– METANIT. Сайт о программировании Руководство по WPF – <https://metanit.com/sharp/wpf/>;

– WorldSkills Russia. Сайт национальной сборной по программированию. Программные решения для бизнеса – <https://nationalteam.worldskills.ru/skills/programmnye-resheniya-dlya-biznesa/>;

– CyberForum. Форум программистов. C#. Программирование с использованием WPF, UWP и Silverlight – <https://www.cyberforum.ru/wpf-silverlight/>;

– ГОСТ 2.105 – 2019. Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам (Издание с Изменением N 1) = Unified system for design documentation. General requirements for textual documents: национальный стандарт Российской Федерации;

– ГОСТ 19.201-78. Единая система программной документации (ЕСПД). Техническое задание. Требования к содержанию и оформлению (с Изменением N 1) = Unified system for program documentation. Technical specifications for development. Requirements to contents and form of presentation: межгосударственный стандарт;

– Visual Studio 2022. Интегрированная среда разработки Visual Studio является творческой стартовой площадкой, которую можно использовать для редактирования, отладки и сборки кода, а также для публикации приложения. В дополнение к стандартному редактору и отладчику, предоставляемых большинством интегрированных сред разработки, Visual Studio включает компиляторы, средства завершения кода, графические конструкторы и многие другие функции для улучшения процесса разработки программного обеспечения. – <https://visualstudio.microsoft.com/ru/>;

– SQL Server Management Studio 2019. Утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server. Утилита включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера – <https://www.microsoft.com/ru-ru/sql-server/sql-server-2019>;

– SQL Server 2022. Система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов - Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка – <https://www.microsoft.com/ru-ru/sql-server?rtc=1>.

Приложение А

Техническое задание. Требования к программным модулям

1 Введение

* 1. Наименование программы

Наименование программы – «ZimAdmin».

* 1. Краткая характеристика области применения.

Частные клиники являются учреждениями, которые оказывают платные медицинские услуги населению. Они работают в сфере здравоохранения и предоставляют различные виды медицинских услуг: от диагностики заболеваний до проведения хирургических операций. Кроме того, в частных клиниках могут оказываться услуги по физиотерапии, массажу, психотерапии, косметологии и другим видам медицинской помощи.

Для частных клиник важна автоматизация рабочих процессов, так как это позволяет увеличить эффективность работы и повысить качество медицинских услуг, которые они предоставляют.

2 Основания для разработки

Основанием для разработки является Договор, подписанный 11.04.2023 Директором частной клиники «Зимамед» Петровым Константином Сергеевичем, именуемым в дальнейшем Заказчиком и Завгородним Владиславом Дмитриевичем, именуемым в дальнейшем Исполнителем.

Согласно Договору, Исполнитель обязан разработать и установить систему «ZimAdmin» на оборудовании Заказчика не позднее 15.06.2023, предоставить исходные коды и документацию к разработанной системе не позднее 20.06.2023.

Наименование темы разработки – «Разработка автоматизированного рабочего места администратора».

Условное обозначение темы разработки (шифр темы) – «ZimAdmin»

3 Назначение разработки

3.1 Функциональное назначение системы

Функциональное назначение системы заключается в автоматизации процессов, связанных с приёмом и обработкой информации о пациентах и работе администратора клиники.

3.2 Эксплуатационное назначение системы

Эксплуатационное назначение системы заключается в обеспечении эффективной работы администратора клиники и улучшении обслуживания пациентов.

Система обеспечит:

– улучшение организации работы администратора частной клиники;

– повышение эффективности работы персонала клиники;

– упрощение процесса управления клиникой.

4 Требования к системе

4.1 Функциональные характеристики

Администратор должен выполнять следующие функции в системе:

– создавать, редактировать и удалять информацию о пациентах в базе данных, а также быстро найти нужного пациента по его фамилии;

– регистрировать пациента на прием к врачу, указав необходимые данные, такие как имя, фамилия, дата рождения, контактная информация и причина обращения к врачу;

– создавать расписание приема пациентов для каждого врача в клинике. Расписание должно быть отображено по дням, а также назначать время приема пациентов и привязывать их к определенному врачу;

– просматривать информацию о текущих и предстоящих приемах пациентов на основе расписания. Администратор должен видеть данные о пациентах, времени приема, враче, к которому они назначены;

– вносить изменения в расписание приема, если возникают какие-либо неожиданные обстоятельства, такие как заболевание врача или непредвиденная задержка пациента;

– анализировать нагрузку врачей в клинике: просматривать статистику по количеству врачей, работающих в определённую смену;

– анализировать популярность процедур, а именно сколько дохода принесла каждая услуга;

– формировать отчеты по заключениям. Отчеты должны включать информацию о дате и времени, а также о пациентах и врачах, участвующих в приеме.

4.2 Требования к надежности

Для обеспечения надёжности работы с данными и их хранения система должна соответствовать следующим требованиям надёжности:

– система должна обеспечивать сохранность данных пользователя. при возникновении сбоев в работе системы, данные пользователя не должны быть потеряны;

– система должна работать без перебоев, круглосуточно;

– система должна обладать защитой от несанкционированного доступа и обеспечивать конфиденциальность информации;

4.3 Условия эксплуатации

Для нормального функционирования системы необходимо соблюдать следующие требования по эксплуатации системы:

– система должна работать на персональных компьютерах с операционной системой Windows 10 или более поздней версией;

– сервер должен иметь не менее 1тб на жестком диске для хранения базы данных;

– система должна иметь возможность подключения к локальной сети, чтобы обмениваться данными с другими системами в клинике;

– для обеспечения качественного функционирования системы, необходимо регулярно проводить техническое обслуживание оборудования.

4.4 Требования к составу и параметрам технических средств

Для работы с разрабатываемой системой необходимо следующее оборудование:

– компьютеры: для каждого рабочего места администратора клиники должен быть назначен отдельный компьютер с конфигурацией не менее, чем Intel Core i5, 8 гб оперативной памяти, 500 ГБ жестким диском;

– мониторы: для каждого компьютера необходим монитор с диагональю не менее 19 дюймов и разрешением не менее 1280x1024 пикселей;

– принтер: для печати документов и форм необходим принтер с поддержкой печати на лазерной технологии.

4.5 Требования к информационной и программной совместимости

– операционная система: система должна быть совместима с операционной системой Windows 10 или более поздней версии;

– база данных: система должна быть совместима с СУБД SSMS 2016 или более поздней версии;

– программное обеспечение: система должна быть совместима с приложениями Microsoft Office 2016 или более поздней версии, и другими программными продуктами, используемыми в клинике. так же программный код системы должен быть написан на языке программировании c# с использованием технологии WPF.

4.6 Требования к маркировке и упаковке

Программное изделие передается по физическому носителю (флешки). Специальных требований к маркировке не предъявляется.

4.7 Требования к транспортированию и хранению

Специальных требований не предъявляется.

4.8 Специальные требования

Программа должна обеспечивать взаимодействие с пользователем посредством графического пользовательского интерфейса, разработанного согласно рекомендациям компании-производителя операционной системы.

5 Требования к программной документации

Программная документация должна содержать в себе:

– техническое задание на разработку информационной системы;

– руководство пользователя.

Все документы должны быть представлены в удобочитаемом и доступном формате, а также должны содержать достаточную информацию, необходимую для понимания и использования системы.

6. Технико-экономические показатели

Предполагаемые технико-экономические показатели

– сметная стоимость разработки системы – 5 000 000 руб;

– срок разработки системы - 3 месяца;

– срок окупаемости проекта - 7 месяцев;

– прогнозируемый годовой доход от использования системы - 10 000 000 руб;

– прогнозируемый годовой расход на обслуживание и поддержку системы - 1 000 000 руб;

– предполагаемая прибыль от использования системы - 9 000 000 руб;

– рентабельность проекта - 180%;

– предполагаемое количество пользователей системы - 1 человек;

– стоимость обновления и развития системы - 1 000 000 руб. в год;

7 Стадии и этапы разработки

7.1 Формирование требований к системе

– определение целей и задач проекта;

– анализ предметной области и требований к системе;

– разработка технического задания на создание системы;

– определение необходимых ресурсов и оценка экономической целесообразности проекта.

7.2 Проектирование

– разработка архитектуры и дизайна системы;

– определение функциональных требований к системе;

– выбор технологий и инструментов разработки;

– разработка программной документации.

7.3 Реализация

Реализация будет проходить в нескольких этапах:

– реализация функциональных требований и тестирование их работоспособности;

– разработка интерфейсов пользовательского взаимодействия;

– разработка системы защиты информации.

7.4 Тестирование

Система должна пройти 2 вида тестирования:

– проведение модульного тестированию;

– проведение пользовательского тестирования.

7.5 Ввод в действие

Для ввода системы в эксплуатацию необходимо выполнить следующие действия:

– установка и настройка системы на рабочих местах клиники;

– обучение пользователей системы.

7.6 Сопровождение

Для нормального функционирования системы, во время сопровождение необходимо выполнять следующие действия:

– поддержка и сопровождение работы системы;

– мониторинг и анализ производительности и стабильности работы системы;

– разработка обновлений и улучшений системы.

8 Порядок контроля и приемки

Приемосдаточные испытания программы должны проводиться согласно разработанной исполнителем и согласованной заказчиком «Программы и методики испытаний».

Ход проведения приемо-сдаточных испытаний заказчик и исполнитель документируют в протоколе испытаний.

На основании протокола испытаний исполнитель совместно с заказчиком подписывают акт приемки-сдачи программы в эксплуатацию.

Приложение Б

Программный код

Программный код представлен в контексте текстового редактора MS Visual Studio Professional 2022.

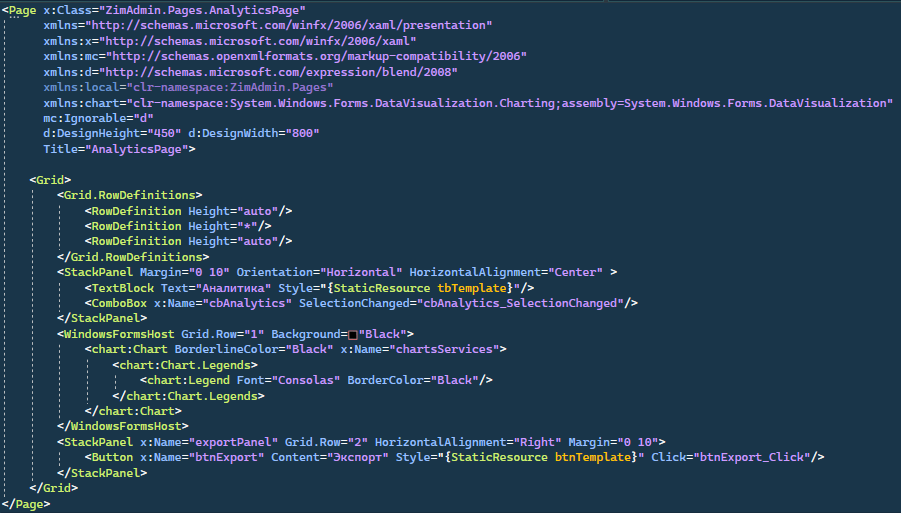


Рисунок Б.1 – Программный код разметки страницы «Аналитики»

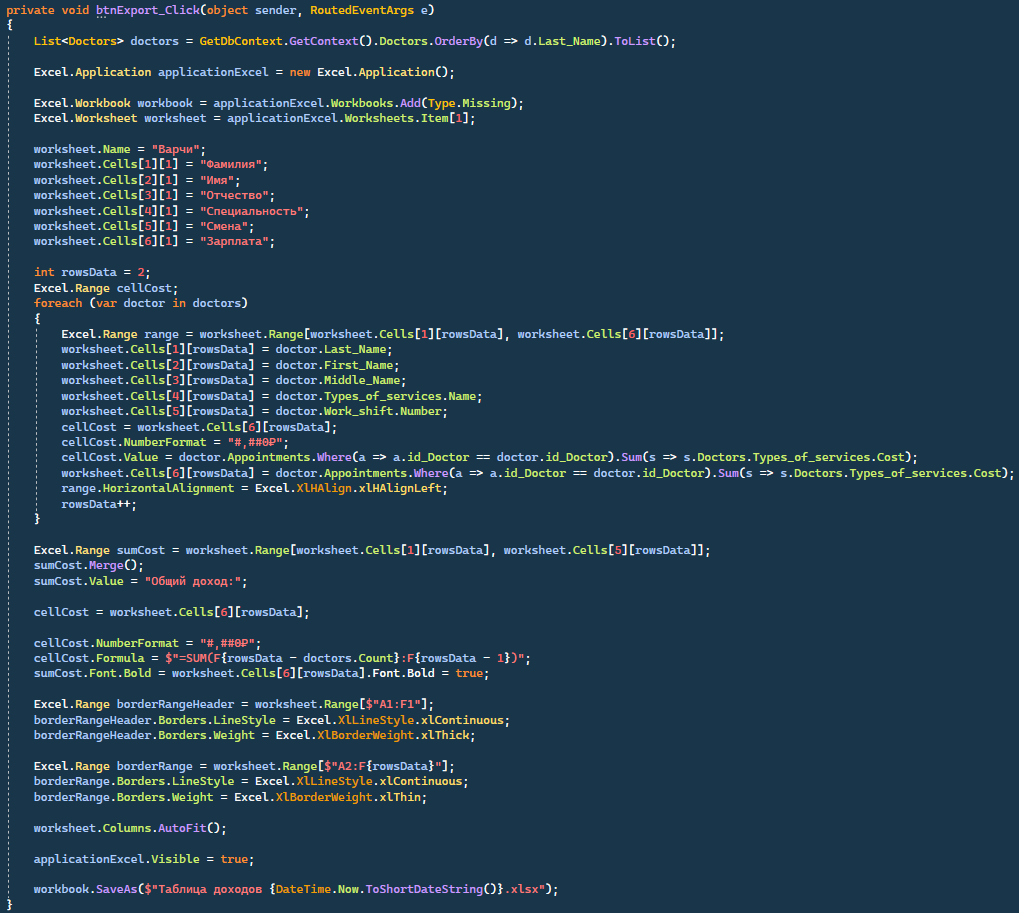


Рисунок Б.2 – Программный код обработчика кнопки «Экспорт» данных в таблицы Excel на странице «Аналитика».

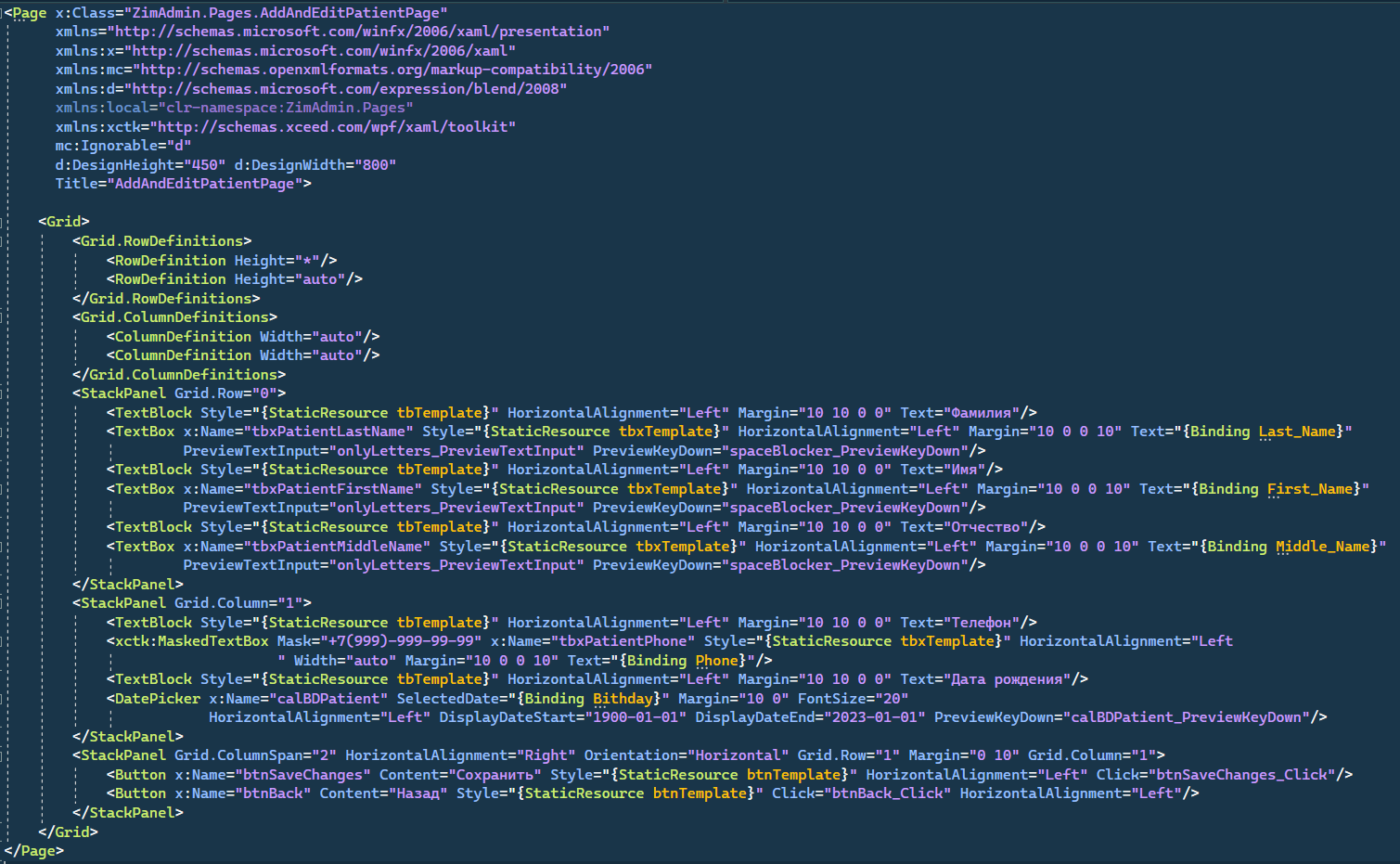


Рисунок Б.3 – Программный код разметки страницы «Добавить и изменить пациента»

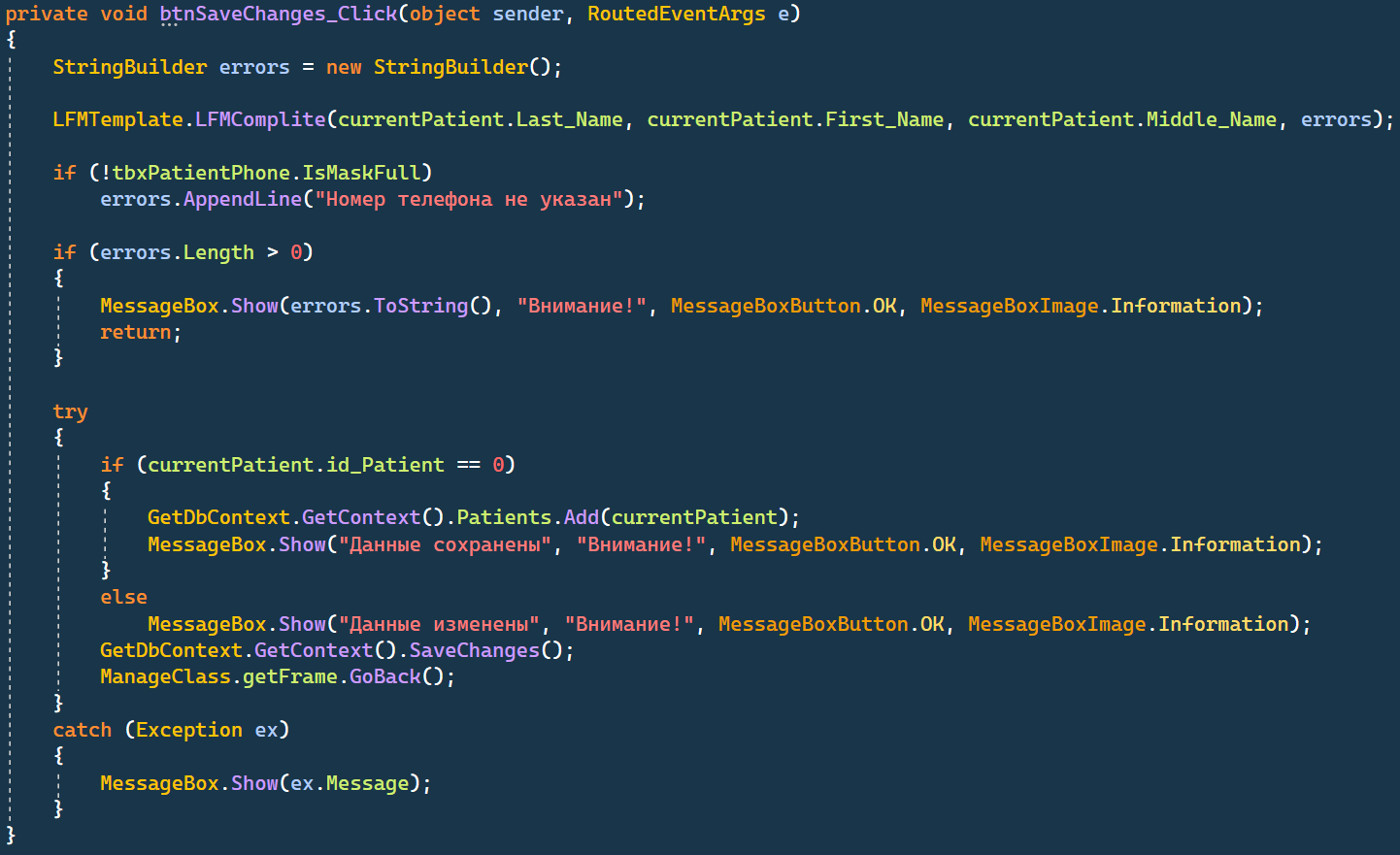


Рисунок Б.4 Программный код для кнопки «Сохранить изменения» на странице «Добавить и изменить пациента»

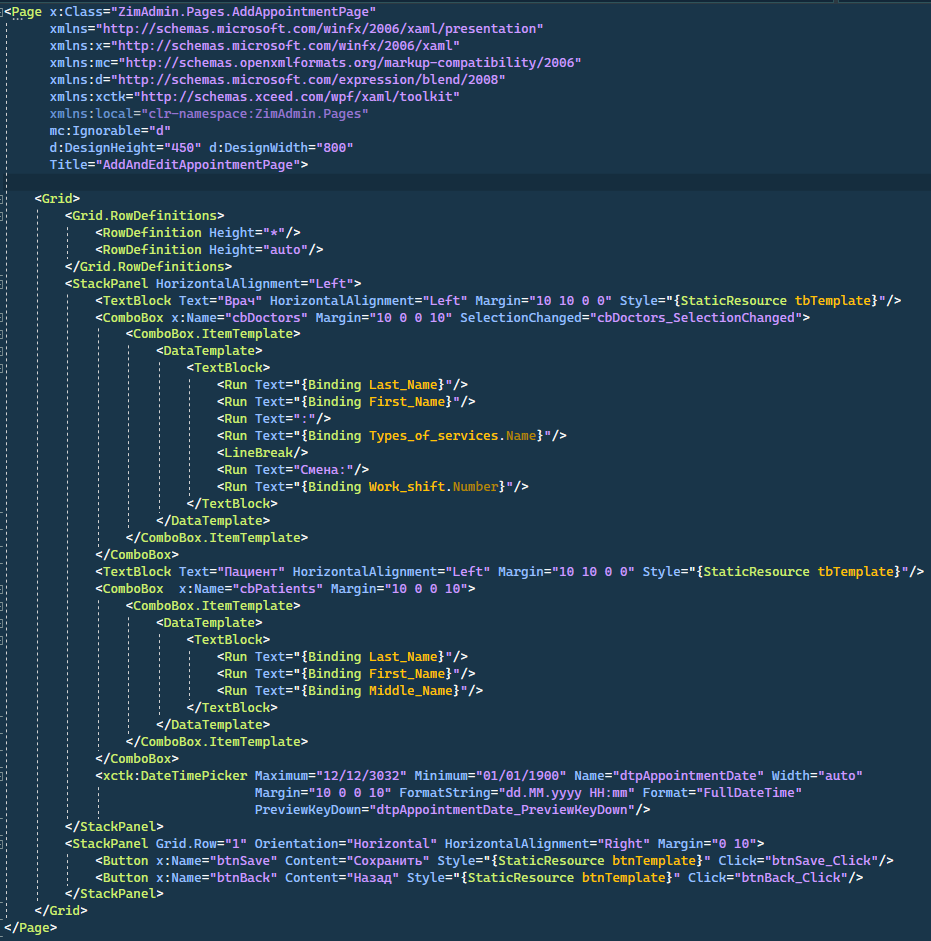


Рисунок Б.5 – Программный код разметки страницы «Добавление приёма»

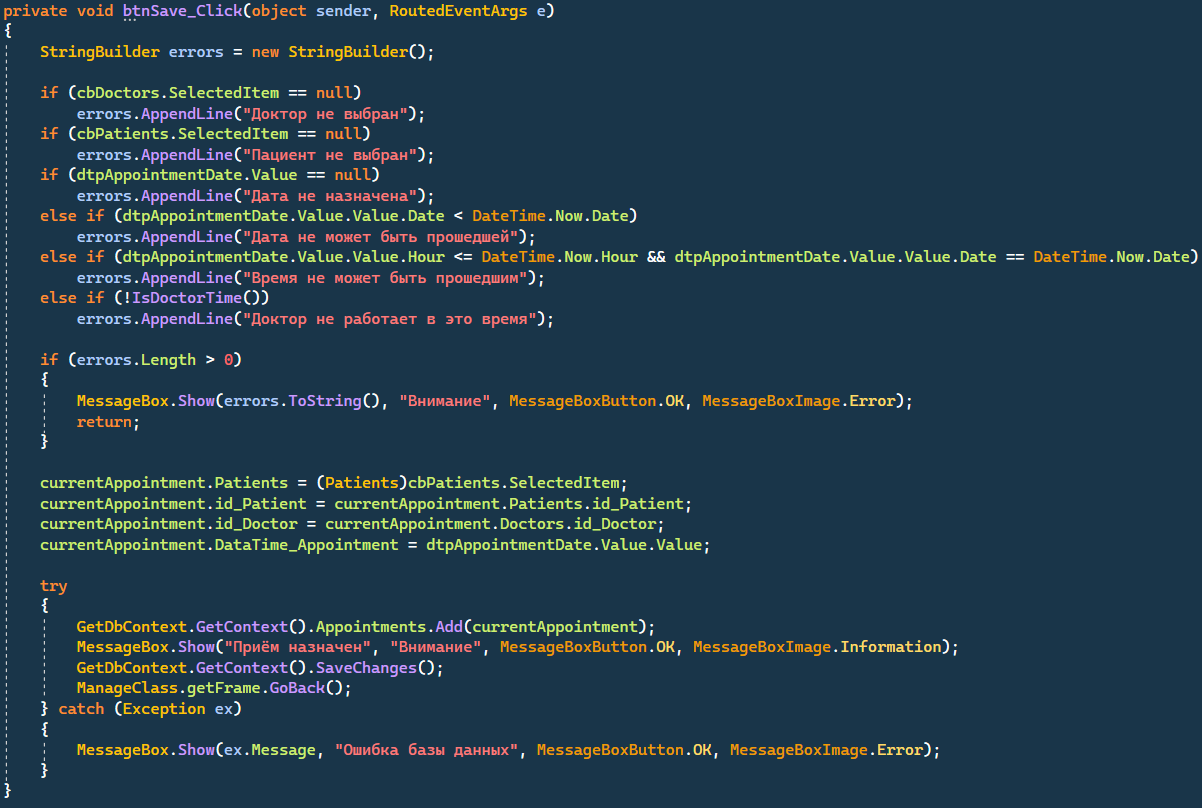


Рисунок Б.6 – Программный код обработчика кнопки «Добавить приём»



Рисунок Б.7 – Программный код страницы «Заключения пациента»

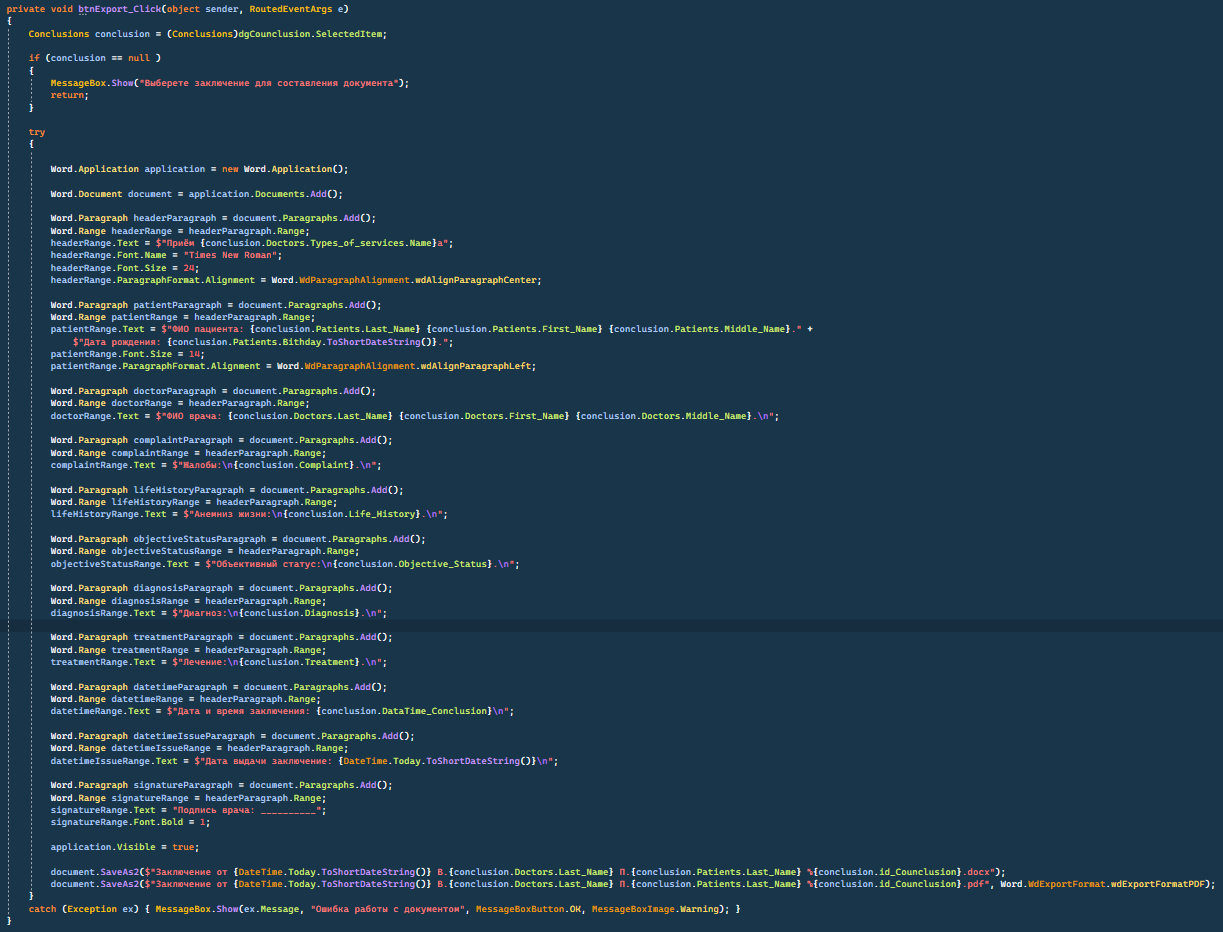


Рисунок Б.8 – Программный код обработчика кнопки «Экспорт» на странице «Заключение пациента»