**Содержание**

[Введение 17](#_Toc104821473)

[1 Аналитический обзор существующих методов и средств решения поставленной задачи 19](#_Toc104821474)

[1.1 Пример решения аналогичной задачи 19](#_Toc104821475)

[1.2 Язык и среда программирования 21](#_Toc104821477)

[1.3 Основы графических интерфейсов 21](#_Toc104821478)

[1.4 Основы Web сервисов 24](#_Toc104821479)

[1.5 Основы проектирования базы данных 25](#_Toc104821480)

[1.6 Основы контейнеризации 27](#_Toc104821482)

[1.7 Постановка задачи 27](#_Toc104821483)

[2 АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 29](#_Toc104821484)

[2.1 Анализ задания на проектирование 29](#_Toc104821485)

[2.2 Доменные классы 33](#_Toc104821490)

[3 Разработка программного обеспечени 40](#_Toc104821538)

[3.1 Уровень доступа к данным 40](#_Toc104821539)

[3.2 WEB-сервис CallAPI 42](#_Toc104821549)

[3.3 WEB-сервис ServiceAPI 58](#_Toc104821607)

[3.4 Клиенты микросервисов 64](#_Toc104821609)

[3.5 Настольное приложение для диспетчеров 69](#_Toc104821623)

[3.6 WEB–приложение для бригад и персонала 73](#_Toc104821626)

[4 ТЕСТИРОВАНИЕ, ВЕРИФИКАЦИЯ И ВАЛИДАЦИЯ 75](#_Toc104821631)

[5 ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ДИПЛОМНОЙ РАБОТЫ 79](#_Toc104821632)

[5.1 Расчёт общей трудоемкости разработки программного обеспечения 79](#_Toc104821633)

[5.2 Расчет совокупных капитальных вложений в проект 83](#_Toc104821634)

[5.3 Расчёт затрат на разработку программного продукта 84](#_Toc104821635)

[5.4 Формирование цены при создании программного обеспечения 91](#_Toc104821636)

[5.5 Статистическая оценка экономической эффективности проекта 93](#_Toc104821637)

[6 ГОСУДАРСТВЕННАЯ ПОЛИТИКА В ОБЛАСТИ ОХРАНЫ ТРУДА 96](#_Toc104821638)

[Список использованных источников 103](#_Toc104821639)

[ПРИЛОЖЕНИЕ А 105](#_Toc104821640)

[ПРИЛОЖЕНИЕ Б 106](#_Toc104821641)

[ПРИЛОЖЕНИЕ В 107](#_Toc104821642)

[ПРИЛОЖЕНИЕ Г 108](#_Toc104821643)

[ПРИЛОЖЕНИЕ Д 109](#_Toc104821644)

# ВВЕДЕНИЕ

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

Дипломная работа

Уровень внедрения комплексных медицинских информационных систем в белорусском здравоохранении, по сдержанным оценкам, не превышает в настоящее время 30% от общей численности лечебно-профилактических организаций.

Наиболее актуальной и сложной до настоящего времени является проблема полной автоматизации процесса управления службой скорой медицинской помощи. Развитие данной службы невозможно без полной автоматизации процесса управления в условиях постоянного роста количества выполняемых вызовов.

Основой информационного обеспечения является система учета и отчетности, определяющая документальное обеспечение деятельности службы и взаимодействие ее структурных подразделений. Сегодня выбор медицинских информационных систем, с учетом их высокой стоимости, является одним из самых рискованных управленческих решений для руководителя любого уровня.

Существующая и по сей день ручная технология сбора и обработки оперативной информации не позволяет быстро и эффективно управлять выездными бригадами скорой медицинской помощи, оперативно проводить всесторонний анализ качественных и количественных показателей деятельности службы.

Так как работа скорой помощи очень важна, появляется необходимость оптимизировать ее работу для увеличения количества вызовов, которые она может выполнять за одну рабочую смену. Данная программа должна улучшить качество работы, скорость реагирования на вызовы, учет статистических и экономических данных.

Качество работы: в процессе обработки вызовов за счет заранее подготовленных данных, таких как улица, заболевание, наименование медикаментов, сложнее ошибиться.

Создание Web сервисов позволит унифицировать работу настольных и Web приложений. Это позволит переиспользовать функционал на разных платформах.

Скорость реагирования: чем быстрее будет передаваться вызов бригаде, тем больше вызовов можно сделать.

Учет статистических и экономических данных: благодаря базе данных, в которых хранятся вызова, проще рассчитывать расход медикаментов и количество топлива для машин скорой помощи.

База данных представляет собой определенный набор данных, которые, как правило, связаны объединяющим признаком либо свойством (или несколькими). Эти данные упорядочены, например, по алфавиту. Обилие различных данных, которые могут быть помещены в единую базу, ведет к множеству вариаций того, что может быть записано: личные данные пользователей, записи, даты, заказы и так далее.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

4

Дипломная работа

В первую очередь это удобно тем, что информацию можно быстро заносить в базу данных и так же быстро ее извлекать при необходимости. Немаловажной является и взаимосвязь информации в базе данных: изменение одной строчки может привести к значительным изменениям других строк. Работать с данными таким образом гораздо проще и быстрее, чем если бы изменения касались только одного места в базе данных.

Цель работы:

Создать приложение позволяющее облегчить и автоматизировать работу станций скорой медицинской помощи, а также собирать и анализировать статистические данные. Программа должна осуществлять передачу информации о вызовах между подстанциями и вести учет больных. Отслеживать работу бригад скорой помощи и вести учет вызовов, на которые реагировала бригада. Выводить статистические данные по разным критериям для пользователей с определенным доступом, для анализа работы станций.

Задача работы:

Приложение должно осуществлять прием вызовов, выполнять обработку их, хранить данные в базе и осуществлять поиск информации о вызове. Выводить информацию о больных и о вызовах сделанных каждой бригадой. Собирать и выводить статистические данные, позволять анализировать работу станций. Также в задачи работы включается создание Web приложения, которое позволит ускорить и облегчить работу сотрудников.

# **Аналитический обзор существующих методов и средств решения поставленной задачи**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

5

Дипломная работа

## **Пример решения аналогичной задачи**

Готовые программные решения для станций скорой медицинской помощи и мониторинга их работы являются узкоспециализированными, именно поэтому список возможных аналогов представлен несколькими системами. Также анализ существующих решений усложняется тем, что данные приложения не находятся в свободном доступе, даже для ознакомления.

Специализированная программная система ISIDA 103 обеспечивает управление деятельностью станции скорой медицинской помощи (ССМП) – как централизованной, так и децентрализованной, имеющей развитую сеть филиалов (подстанций).

Основные функции системы:

1. прием и регистрация вызовов от населения:

получение номера телефона входящего вызова;

определение повода к вызову скорой медицинской помощи, соответствующего ему профиля выездной бригады и номера подстанции, с которой будет осуществлен выезд;

определение места вызова с использованием электронной карты местности;

регистрация вызова.

1. диспетчеризация вызовов:

передача электронной карты вызова на соответствующую подстанцию, в соответствии с номером подстанции в карте вызова;

назначение выездных бригад на вызов, в том числе с использованием электронной карты местности;

передача электронных карт вызова выездным бригадам, в том числе с возможностью получения карты вызова на экран мобильного устройства (смартфона или планшетного компьютера);

закрытие карт вызова.

1. ввод данных о результатах вызовов:

ввод уточненных данных о пациенте;

ввод результата вызова и диагнозов по МКБ-10;

ввод данных о выполненных специфических мероприятиях при оказании медицинской помощи;

ввод данных об израсходованных медикаментах.

1. формирование оперативной отчетности;
2. формирование статистической отчетности на основании закрытых карт вызовов.

Кроме того, система снабжена развитыми поисковыми возможностями, набором настраиваемых отчётных срезов (признаки, по которым можно искать и производить группировку данных), системой адресной привязки зон обслуживания подстанций и поликлиник, возможностью формирования графиков смен выездных бригад и оперативного изменения профилей бригад.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

6

Дипломная работа

Функциональность системы может быть расширена за счет подключения дополнительных модулей и организации взаимодействия с информационными системами региона

В настоящее время, станции скорой медицинской помощи используют программу, позволяющая частично автоматизировать работу диспетчера скорой помощи.

На телефон диспетчера поступает вызов, звонящий называет фамилию пациента, его возраст, улицу, на которой находится больной, номер дома и номер квартиры, также называет контактные телефоны. Далее вызов передается диспетчеру конкретной подстанции, расположенной ближе других к этому адресу.

Вызов от диспетчера подстанции передается конкретной свободной бригаде. На месте вызова бригадой уточняется информация такая как: место прописки пациента, его ФИО, возраст и заболевания. По возвращении всея уточненная информация передается диспетчеру для обработки и занесения в базу.

Существующая программа позволяет производить поиск вызовов и пациентов в системе, включает в себя список возможных адресов, болезней, бригад, сотрудников, адреса поликлиник, результатов вызова. Данная система также обладает возможностью наблюдения за бригадами скорой помощи с помощью GPS трекеров.

Также есть возможность просмотра статистики по вызовам, по подстанциям, по бригадам.

Достоинствами такой системы являются:

1. скорость передачи вызова между подстанциями;
2. хранение данных о больных и сотрудниках в базе;
3. скорость реагирования на вызов.

Недостатками являются:

1. ручной ввод вызовов;
2. ручная обработка выполненных вызовов;
3. неполностью продуманный интерфейс программы.

## **Язык и среда программирования**

C# является объектно-ориентированным языком. Разработка современных приложений все больше тяготеет к созданию программных компонентов в форме автономных и самоописательных пакетов, реализующих отдельные функциональные возможности.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

7

Дипломная работа

Главная особенность таких компонентов в том, что они представляют собой модель программирования со свойствами, методами и событиями. У них есть атрибуты, предоставляющие декларативные сведения о компоненте. Они включают в себя собственную документацию. C# предоставляет языковые конструкции, непосредственно поддерживающие такую концепцию работы. Благодаря этому C# подходит для создания и применения программных компонентов.

Когда говорят C#, нередко имеют в виду технологии платформы .NET (Windows Forms, WPF, ASP.NET, Xamarin). И, наоборот, когда говорят .NET, нередко имеют в виду C#. Язык C# был создан специально для работы с фреймворком .NET, однако само понятие .NET несколько шире. Фреймворк .NET представляет мощную платформу для создания приложений. Можно выделить следующие ее основные черты:

1. поддержка нескольких языков. Основой платформы является общеязыковая среда исполнения Common Language Runtime (CLR), благодаря чему .NET поддерживает несколько языков: наряду с C# это также VB.NET, C++, F#, а также различные диалекты других языков, привязанные к .NET, например, Delphi.NET. При компиляции код на любом из этих языков компилируется в сборку на общем языке CIL (Common Intermediate Language) – своего рода ассемблер платформы .NET. Поэтому мы можем сделать отдельные модули одного приложения на отдельных языках;
2. кроссплатформенность. .NET является переносимой платформой (с некоторыми ограничениями). Последняя версия платформы на данный момент .NET Core поддерживается на большинстве современных ОС Windows, MacOS, Linux. Используя различные технологии на платформе .NET, можно разрабатывать приложения на языке C# для самых разных платформ – Windows, MacOS, Linux, Android, iOS, Tizen;
3. мощная библиотека классов. .NET представляет единую для всех поддерживаемых языков библиотеку классов.

## **Основы графических интерфейсов**

Windows Forms – [интерфейс программирования приложений](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9) (API), отвечающий за [графический интерфейс пользователя](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) и являющийся частью [Microsoft](https://ru.wikipedia.org/wiki/Microsoft) [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework). Данный интерфейс упрощает доступ к элементам интерфейса [Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows) за счет создания обёртки для существующего [Win32 API](https://ru.wikipedia.org/wiki/Windows_API) в [управляемом коде](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4). Причём управляемый код – классы, реализующие [API](https://ru.wikipedia.org/wiki/API) для Windows Forms, не зависят от языка разработки. То есть программист одинаково может использовать Windows Forms как при написании ПО на C#, С++, так и на VB.Net, J# и др.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

8

Дипломная работа

С одной стороны, Windows Forms рассматривается как замена более старой и сложной библиотеке [MFC](https://ru.wikipedia.org/wiki/Microsoft_Foundation_Classes), изначально написанной на языке [C++](https://ru.wikipedia.org/wiki/C%2B%2B). С другой стороны, WF не предлагает парадигму, сравнимую с [MVC](https://ru.wikipedia.org/wiki/Model-View-Controller). Для исправления этой ситуации и реализации данной функциональности в WF существуют сторонние библиотеки.

Windows Forms включает широкий набор элементов управления, которые можно добавлять на формы: текстовые поля, кнопки, раскрывающиеся списки, переключатели и даже Web-страницы. Список всех элементов управления, которые можно использовать в форме, представлены в разделе [Элементы управления для использования в формах Windows Forms](https://docs.microsoft.com/ru-ru/dotnet/framework/winforms/controls/controls-to-use-on-windows-forms). Если существующий элемент управления не удовлетворяет потребностям, в Windows Forms можно создать пользовательские элементы управления с помощью класса [UserControl](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.usercontrol).

В состав Windows Forms входят многофункциональные элементы пользовательского интерфейса, позволяющие воссоздавать возможности таких сложных приложений, как Microsoft Office. Используя элементы управления [ToolStrip](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.toolstrip) и [MenuStrip](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.menustrip), можно создавать панели инструментов и меню, содержащие текст и рисунки, подменю и другие элементы управления, такие как текстовые поля и поля со списками.

Во многих приложениях нужно отображать данные из базы данных, XML-файла, Web-службы XML или другого источника данных. Windows Forms предоставляет гибкий элемент управления с именем [DataGridView](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.datagridview) для отображения таких табличных данных в традиционном формате строк и столбцов так, что каждый фрагмент данных занимает свою собственную ячейку. С помощью [DataGridView](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.datagridview) можно, помимо прочего, настроить внешний вид отдельных ячеек, зафиксировать строки и столбцы на своем месте, а также обеспечить отображение сложных элементов управления внутри ячеек.

При использовании интеллектуальных клиентов Windows Forms можно легко подключаться к источникам данных по сети. Компонент [BindingSource](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.bindingsource) представляет соединение с источником данных и предоставляет методы для привязки данных к элементам управления, перехода к предыдущим и следующим записям, изменения записей и сохранения изменений в исходном источнике. Элемент управления [BindingNavigator](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.bindingnavigator) предоставляет простой интерфейс на основе компонента [BindingSource](https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.bindingsource) для перехода между записями.

Вы можете легко создавать элементы управления с привязкой к данным с помощью окна “Источники данных”. В нем приводятся имеющиеся в проекте источники данных, такие как базы данных, Web-службы и объекты. Создавать элементы управления с привязкой к данным можно путем перетаскивания объектов из этого окна в формы проекта. Также можно связывать существующие элементы управления с данными, перетаскивая объекты из окна «Источники данных» в существующие элементы управления.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

9

Дипломная работа

Технология WPF (Windows Presentation Foundation) является часть экосистемы платформы .NET и представляет собой подсистему для построения графических интерфейсов.

Если при создании традиционных приложений на основе WinForms за отрисовку элементов управления и графики отвечали такие части ОС Windows, как User32 и GDI+, то приложения WPF основаны на DirectX. В этом состоит ключевая особенность рендеринга графики в WPF: используя WPF, значительная часть работы по отрисовке графики, как простейших кнопочек, так и сложных 3D-моделей, ложиться на графический процессор на видеокарте, что также позволяет воспользоваться аппаратным ускорением графики.

Одной из важных особенностей является использование языка декларативной разметки интерфейса XAML, основанного на XML. XAML (Extensible Application Markup Language – расширяемый язык разметки приложений) представляет собой язык разметки, используемый для создания экземпляров объектов .NET. Хотя язык XAML – это технология, которая может быть применима ко многим различным предметным областям, его главное назначение – конструирование пользовательских интерфейсов WPF.

WPF позволяет разрабатывать приложения, используя как разметку, так и код программной части, что привычно для разработчиков на ASP.NET. Разметка XAML обычно используется для определения внешнего вида приложения, а управляемые языки программирования (код программной части) – для реализации его поведения. Такое разделение внешнего вида и поведения имеет ряд преимуществ:

1. затраты на разработку и обслуживание снижаются, так как разметка, определяющая внешний вид, не связана тесно с кодом, обуславливающим поведение;
2. повышается эффективность разработки, так как дизайнеры, занимающиеся внешним видом приложения, могут работать параллельно с разработчиками, реализующими поведение приложения.

Платформа WPF предоставляет широкий, гибкий и масштабируемый набор графических функций, который обладает перечисленными ниже преимуществами.

1. независимость графики от разрешения и устройства. Основной единицей измерения в графической системе WPF является аппаратно-независимый пиксель, размер которого составляет 1/96 дюйма вне зависимости от разрешения экрана. Это создает основу для, независимой от разрешения и аппаратной платформы, отрисовки. Каждый аппаратно-независимый пиксель автоматически масштабируется в соответствии с заданным в системе количеством точек на дюйм (DPI);
2. повышение точности. Система координат WPF основана на числах двойной точности с плавающей запятой, а не числах одинарной точности. Значения преобразования и прозрачности также выражаются числами двойной точности. Платформа WPF также поддерживает широкую цветовую палитру (scRGB) и имеет встроенную поддержку управления входными данными из разных цветовых схем;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

10

Дипломная работа

1. аппаратное ускорение. Система графики WPF использует возможности графического оборудования, чтобы снизить нагрузку на центральный процессор.

В результате, язык C# и принципы объектно-ориентированного программирования подходят для реализации программы, а технология WPF не только проста в использовании, но и обеспечит кроссплатформенность, что позволит запускать программу на практически любом компьютере.

## **Основы Web сервисов**

Web API представляет способ построения приложения ASP.NET, который специально заточен для работы в стиле REST.

Зачастую REST-стиль особенно удобен при создании всякого рода Single Page Application, которые нередко используют специальные javascript фреймворки типа Angular, React или Vue.js. По сути Web API представляет собой Web-службу, к которой могут обращаться другие приложения. Причем эти приложения могут представлять любую технологию и платформу – это могут быть Web-приложения, мобильные или десктопные клиенты.

REST (Representational state transfer) – это стиль архитектуры программного обеспечения для распределенных систем, таких как World Wide Web, который, как правило, используется для построения Web-служб. Термин REST был введен в 2000 году Роем Филдингом, одним из авторов HTTP протокола. Системы, поддерживающие REST, называются RESTful-системами. В общем случае REST является очень простым интерфейсом управления информацией без использования каких-то дополнительных внутренних прослоек. Каждая единица информации однозначно определяется глобальным идентификатором, таким как URL. Каждая URL в свою очередь имеет строго заданный формат.

Особенности архитектурного стиля:

1. каждая сущность должна иметь уникальный идентификатор – URI;
2. сущности должны быть связаны между собой;
3. для чтения и изменения данных должны использоваться;
4. стандартные методы;
5. должна быть поддержка нескольких типов ресурсов;
6. взаимодействие должно осуществляться без состояния.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

11

Дипломная работа

Стандартные методы таковы:

1. GET – получение данных без их изменения. Это наиболее популярный и легкий метод. Он только возвращает данные, а не изменяет их, поэтому на клиенте вам не нужно заботиться о том, что вы можете повредить данные;
2. POST – метод, подразумевающий вставку новых записей;
3. PUT – метод, подразумевающий изменение существующих записей;
4. PATCH – метод, подразумевающий изменение идентификатора существующих записей;
5. DELETE – метод, подразумевающий удаление записей.

## **Основы проектирования базы данных**

Проектирование базы данных для приложения осуществлялась с помощью MsSQL Server используя Sql management studio для создания БД.

Microsoft SQL Server – 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 (SQL) 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.

SQL Server – это основа платформы обработки данных Майкрософт, которая предоставляет надежную и устойчивую производительность (в том числе благодаря технологиям обработки данных в памяти) и помогает быстрее извлечь ценную информацию из любых данных, расположенных как в локальной среде, так и в облаке.

SQL Server характеризуется такими особенностями как:

1. производительность;
2. надежность и безопасность;
3. простота.

Центральным аспектом в MS SQL Server, как и в любой СУБД, является база данных. База данных представляет хранилище данных, организованных определенным способом. Нередко физически база данных представляет файл на жестком диске, хотя такое соответствие необязательно. Для хранения и администрирования баз данных применяются системы управления базами данных (database management system) или СУБД (DBMS). И как раз MS SQL Server является одной из такой СУБД.

Для организации баз данных MS SQL Server использует реляционную модель. Эта модель баз данных была разработана еще в 1970 году Эдгаром Коддом. А на сегодняшний день она фактически является стандартом для организации баз данных.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

12

Дипломная работа

Реляционная модель предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта.

Для идентификации каждой строки в рамках таблицы применяется первичный ключ (primary key). В качестве первичного ключа может выступать один или несколько столбцов. Используя первичный ключ, мы можем ссылаться на определенную строку в таблице. Соответственно две строки не могут иметь один и тот же первичный ключ.

Через ключи одна таблица может быть связана с другой, то есть между двумя таблицами могут быть организованы связи. А сама таблица может быть представлена в виде отношения («relation»).

Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос на языке SQL посредством специального API. СУБД должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения.

Изначально язык SQL был разработан в компании IBM для системы баз данных, которая называлась System/R. При этом сам язык назывался SEQUEL

(Structured English Query Language). Хотя в итоге ни база данных, ни сам язык не были впоследствии официально опубликованы, по традиции сам термин SQL нередко произносят как «сиквел».

В 1979 году компания Relational Software Inc. разработала первую систему управления баз данных, которая называлась Oracle и которая использовала язык SQL. В связи с успехом данного продукта компания была переименована в Oracle.

Впоследствии стали появляться другие системы баз данных, которые использовали SQL. В итоге в 1989 году Американский Национальный Институт Стандартов (ANSI) кодифицировал язык и опубликовал его первый стандарт. После этого стандарт периодически обновлялся и дополнялся. Последнее его обновление состоялось в 2011 году. Но несмотря на наличие стандарта нередко производители СУБД используют свои собственные реализации языка SQL, которые немного отличаются друг от друга.

Выделяются две разновидности языка SQL: PL-SQL и T-SQL. PL-SQL используется в таких СУБД как Oracle и MySQL. T-SQL (Transact-SQL) применяется в SQL Server.

Таким образом язык программирования C# позволит создать быстрое и удобное настольное приложение для работы в офисе, которое позволит использовать Web-сервисы для оптимизации и обобщения работы между несколькими частями системы. Web-сервис на базе WebAPI позволит создать Web приложение с помощью которого несколько программных решений могут иметь одинаковый функционал, что позволит переложить ответственность выполнения конкретной задачи на других участников процесса.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

13

Дипломная работа

База данных построенная с использованием реляционного подхода, позволит создать нормализированные сущности, что позволяет удобно хранить и обрабатывать данные. Работать с такой базой намного проще так как между всеми сущностями имеются различные связи, которые описывают взаимоотношения между ними.

## **Основы контейнеризации**

Docker – программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой [контейнеризации](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%82%D0%B5%D0%B9%D0%BD%D0%B5%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), контейнеризатор приложений. Позволяет «упаковать» приложение со всем его [окружением](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%BA%D1%80%D1%83%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) и зависимостями в контейнер, который может быть развёрнут на любой [Linux](https://ru.wikipedia.org/wiki/Linux)-системе с поддержкой [контрольных групп](https://ru.wikipedia.org/wiki/Cgroups) в [ядре](https://ru.wikipedia.org/wiki/%D0%AF%D0%B4%D1%80%D0%BE_Linux), а также предоставляет набор команд для управления этими контейнерами. Изначально использовал возможности [LXC](https://ru.wikipedia.org/wiki/LXC), с 2015 года начал использовать собственную библиотеку, абстрагирующую виртуализационные возможности ядра Linux – libcontainer. С появлением Open Container Initiative начался переход от монолитной к модульной архитектуре.

## **Постановка задачи**

Целью дипломной работы является спроектировать приложение позволяющее собирать и обрабатывать статистику межрегиональных станций скорой медицинской помощи.

Программа должна моделировать клиент-серверное приложение и должно быть построено на базе микросервисной архитектуры. Набор создаваемых приложений должен предоставлять собой распределенный узел работы. На стороне сервера должна содержаться вся нужна информация для принятия вызова, и его обработки, а именно:

1. название улиц;
2. название диагнозов;
3. название подстанций скорой помощи;
4. список типов людей, которые вызывают;
5. список типов бригад;
6. список мест обслуживания;
7. наименования лекарств и действий скорой помощи.

Настольное приложение должно осуществлять регистрацию вызова, передачу его бригадам, ввод бригад, учет статистических данных исходя из обработанных вызовов, учет хронических больных.

Web-приложение должно осуществлять регистрацию вызова, передачу его бригадам, также должно обладать функциональностью по обработке завершенного вызова и передаче вызова освободившейся бригаде.

При регистрации вызова выполняется запрос к микросервису на создание записи в базе данных. Просмотр этих вызовов доступен всем подстанциям. Когда освобождается бригада, ей передается вызов для обслуживания. Бригада, выполнив вызов и записав точную информацию о пациенте, передает все данные диспетчеру для редактирования и финальной обработки вызова, либо бригада обрабатывает его сама с помощью Web-приложения, после чего сможет взять следующий доступный вызов.

Обработанный вызов передается микросервису, где сохраняется в базу данных и может потом использоваться для учета статистических данных.

Ведение справочника хронических больных осуществляется с помощью введение в всей нужной информации, по обслуживанию пациента и регистрации вызова, в базу, что позволит быстро принять заявку и передать его бригаде.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

14

Дипломная работа

Просмотр всех вызовов, включая попутные, и подробной информации каждого вызова.

Просмотр вызовов выполненные каждой бригадой за смену.

# **АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

15

Дипломная работа

## **Анализ задания на проектирование**

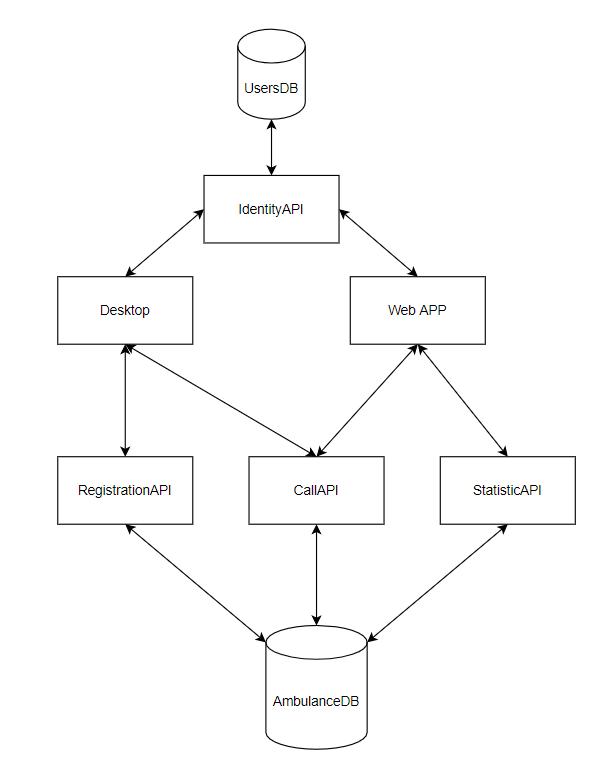


Рисунок 2.1 – Схема взаимодействия модулей

Согласно рисункам 2.1 и 2.2, настольное приложение обращается к CallAPI для реализации обработки вызовов, а именно регистрации вызова, обработка вызова, получение вызова или списка вызовов.

Diagram

Description automatically generated

Изм.

Лист

№ докум.

Подпись

Дата

Лист

16

Дипломная работа

Рисунок 2.2 – Схема взаимодействия диспетчера с системой

Также обращение к RegistrationAPI позволяет регистрировать для работы сотрудников, бригады скорой помощи и данные нужные для работы приложения.

Web-приложение осуществляет похожую работу с вызовами как и настольное приложение, но также позволяет просматривать статистические данные по работе скорой помощи. Для работы обоих приложений, настольного и Web, требуется авторизоваться.

Diagram

Description automatically generated

Рисунок 2.3 – Схема взаимодействия фельдшера с Web-приложением

Согласно рисунку 2.3, авторизация осуществляется с помощью IdentityAPI, к которой обращаются оба приложения. Не авторизированные пользователи не могут работать с настольным и Web-приложениями.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

17

Дипломная работа

Diagram

Description automatically generated

Рисунок 2.4 – Действия диспетчера

В обязанности диспетчера скорой помощи входит регистрация вызовов, их передача, финальная обработка ввод бригад скорой помощи. Подробнее на рисунке 2.4.

В результате, данное приложение должно помочь отказаться от ручного труда, благодаря Web-решению и создания микросервисов. Заполнением дополнительной информации, согласно рисунку 2.5, будет заниматься сама бригада, которая выполняет конкретный вызов, данное заполнение будет происходить через браузер.

Diagram

Description automatically generated

Рисунок 2.5 – Заполнение дополнительной информации по вызову

### Вызов поступает на телефон станции, диспетчер записывает фамилию, имя, отчество, возраст пациента, его состояние, местоположение и кем приходится пациенту вызывающий. Все это заносится в поля формы и передается в список не выполненных вызовов. Подробнее на рисунке 2.6.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

18

Дипломная работа

Diagram

Description automatically generated

Рисунок 2.6 – Алгоритм регистрации вызова

В случае ошибочного заполнения, доступна функция очистки формы.

### После выполнения вызова, он поступает в список вызовов для обработки. Диспетчер должен заменить неточную информацию о вызове на информацию, уточненную у пациента. Подробнее на рисунке 2.7.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

19

Дипломная работа

Chart, diagram, box and whisker chart

Description automatically generated

Рисунок 2.7 – Алгоритм обработки вызова

## **Доменные классы**

Проанализировав предметную область, можно выделить следующие девять доменных классов:

1. (Пациент) Patients;
2. (Хронически больной) ChronicPatients;
3. (Диспетчер) Dispatcher;
4. (Доктор) Doktor;
5. (Фельдшер) MedicalAssistant;
6. (Санитар) Orderly;
7. (Водитель) Driver;
8. (Вызов) Call;
9. (Бригада) AmbulanceBrigade;
10. (Глобальные настройки) GlobalSettings.

Доменная область не подразумевает удаление записей.

Согласно рисунку 2.8, все сущности в схеме имеют отношение один ко многим между несколькими связанными сущностями, кроме сущности GlobalSettings так как эта таблица создана для хранения настроек.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

20

Дипломная работа

A screenshot of a computer

Description automatically generated with low confidence

Рисунок 2.8 – Диаграмма доменных классов

Сущность Patient.

Описывает сущность «Пациент», которого обслуживает бригада скорой помощи. Содержит поля, которые содержат всю нужную информацию о пациенте для обработки. Подробнее в таблице 2.1.

Таблица 2.1 – Сущность Patient.

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Содержание** | **Тип** | **Комментарий** |
| Id | ID | int | Идентификационный номер |
| FIO | Фамилия Имя Отчество | varchar | Фамилия Имя Отчество пациента |
| Age | Возраст | Float | Возраст пациента |
| Diagnosis | Диагноз | varchar | Диагноз поставленный бригадой |

Продолжение таблицы 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| Street | Улица адреса прописки | varchar | Улица на которой прописан пациент |
| HouseNumber | Номер дома адреса прописки | varchar | Номер дома в котором прописан |
| FlatNumber | Номер квартиры адреса прописки | varchar | Номер квартиры в которой прописан |
| Gender | Пол пациента | varchar | Пол пациента |

Изм.

Лист

№ докум.

Подпись

Дата

Лист

21

Дипломная работа

Сущность Dispatcher.

Описывает сущность «Диспетчер скорой помощи», который работает с настольной программой. Подробнее в таблице 2.2.

Таблица 2.2 – Сущность Dispatcher

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Содержание** | **Тип** | **Комментарий** |
| Id | ID | int | Идентификационный номер |
| Name | Имя | varchar | Имя сотрудника |
| Surname | Фамилия | varchar | Фамилия сотрудника |
| MiddleName | Отчество | varchar | Отчество сотрудника |

Сущность Doktor

Описывает сущность «Доктор выездной бригады» скорой помощи. Подробнее в таблице 2.3.

Таблица 2.3 – Сущность Doktor

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Содержание** | **Тип** | **Комментарий** |
| Id | ID | int | Идентификационный номер |
| Name | Имя | varchar | Имя сотрудника |

Продолжение таблицы 2.3

|  |  |  |  |
| --- | --- | --- | --- |
| Surname | Фамилия | varchar | Фамилия сотрудника |
| MiddleName | Отчество | varchar | Отчество сотрудника |

Изм.

Лист

№ докум.

Подпись

Дата

Лист

22

Дипломная работа

Сущность MedicalAssistant.

Описывает сущность «Фельдшер выездной бригады» скорой помощи. Подробнее в таблице 2.4.

Таблица 2.4 – Сущность MedicalAssistant

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Содержание** | **Тип** | **Комментарий** |
| Id | ID | int | Идентификационный номер |
| Name | Имя | varchar | Имя сотрудника |
| Surname | Фамилия | varchar | Фамилия сотрудника |
| MiddleName | Отчество | varchar | Отчество сотрудника |

Сущность Orderly.

Описывает сущность «Санитар выездной бригады» скорой помощи. Подробнее в таблице 2.5.

Таблица 2.5 – Сущность Orderly

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Содержание** | **Тип** | **Комментарий** |
| Id | ID | int | Идентификационный номер |
| Name | Имя | varchar | Имя сотрудника |
| Surname | Фамилия | varchar | Фамилия сотрудника |
| MiddleName | Отчество | varchar | Отчество сотрудника |

Сущность Driver.

Описывает сущность «Водитель машины выездной бригады» скорой помощи. Подробнее в таблице 2.6.

Таблица 2.6 – Сущность Driver

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Содержание** | **Тип** | **Комментарий** |
| Id | ID | int | Идентификационный номер |
| Name | Имя | varchar | Имя сотрудника |
| Surname | Фамилия | varchar | Фамилия сотрудника |
| MiddleName | Отчество | varchar | Отчество сотрудника |

Изм.

Лист

№ докум.

Подпись

Дата

Лист

23

Дипломная работа

Сущность Call.

Описывает сущность «Вызов», содержит в себе информацию о пациенте, о бригаде, которая выполняла вызов, о лечении и результате лечения. Подробнее в таблице 2.7.

Таблица 2.7 – Сущность Call

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Содержание** | **Тип** | **Комментарий** |
| Id | ID | int | Идентификационный номер |
| AmbulanceBrigadeId | Идентификатор бригады | Int | Идентификатор бригады |
| DispatcherId | Идентификатор диспетчера | Int | Идентификатор диспетчера принявшего вызов |
| MainCallNumber | Номер вызова с которым связан текущий | Int | Номер вызова с которым связан текущий |
| IsIncidential | Флаг показывающий тип вызова | Bit | Вызов может быть основным(false) или попутным(true) |
| CallNumber | Номер вызова | uint | Уникальный номер вызова |
| PatientId | Номер пациента | int | Номер пациента |
| Results | Результат | varchar | Результата вызова |
| DateTimeReception | Дата и время получения | DateTime | Дата и время получения вызова |
| TransferDateTime | Дата и время передачи | DateTime | Дата и время передачи вызова |
| ArrivalDateTime | Дата и время прибытия | DateTime | Дата и время прибытия вызова |
| DepartureDateTime | Дата и время отправления | DateTime | Дата и время выезда с вызова |

Продолжение таблицы 2.7

Изм.

Лист

№ докум.

Подпись

Дата

Лист

24

Дипломная работа

|  |  |  |  |
| --- | --- | --- | --- |
| ComeBackDateTime | Дата и время возвращения | DateTime | Дата и время возвращения на подстанцию |
| TransferringDispatcherId | Передающий диспетчер | int | Диспетчер передавший вызов |
| ProcessingDispatcher | Обрабатывающий диспетчер | varchar | Диспетчер обработавший вызов |
| KilometrageBefor | Километраж до | int | Километраж машины до вызова |
| KilometrageAfter | Километраж после | int | Километраж машины после вызова |
| Place | Место | varchar | Место обслуживания пациента |
| CallNotes | Заметки | varchar | Заметки к вызову |
| Treatment | Лечение | varchar | Лечение |
| CallType | Тип вызова | varchar | Тип вызова |
| IsProcessed | Флаг состояния обработки вызова | bit | Показывает обработан вызов или нет |

Сущность AmbulanceBrigade.

Данная сущность базы данных описывает бригаду скорой помощи которая занимается выполнением вызова. Подробнее в таблице 2.8.

Таблица 2.8 – Сущность AmbulanceBrigade

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Название** | **Содержание** | | **Тип** | | **Комментарий** | |
| Id | Уникальный идентификатор | | Int | | Уникальный идентификатор | |
| DoktorId | Идентификатор врача | | Int | | Идентификатор врача | |
| FirstMedicalAssistantId | Идентификатор фельдшера | | Int | | Идентификатор фельдшера | |
| SecondMedicalAssistantId | | Идентификатор фельдшера | | Int | | Идентификатор фельдшера |
| DriverId | | Идентификатор водителя | | Int | | Идентификатор водителя |
| OrderlyId | | Идентификатор санитара | | Int | | Идентификатор санитара |

Продолжение таблицы 2.8

Изм.

Лист

№ докум.

Подпись

Дата

Лист

25

Дипломная работа

|  |  |  |  |
| --- | --- | --- | --- |
| Number | Номер бригады | Int | Номер бригады |
| BrigadeType | Тип бригады | Byte | Тип бригады |
| DateTimeStart | Время и дата начала работы | Datetime | Время и дата начала работы |
| DateTimeEnd | Время и дата конца работы | Datetime | Время и дата конца работы |
| StationName | Название станции за которой закреплена бригада | varchar | Название станции за которой закреплена бригада |

# **Разработка программного обеспечени**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

## **Уровень доступа к данным**

Для доступа к базе данных используется подключаемая библиотека EntityFrameworkCore. Она построена с помощью таких паттернов как репозиторий и единица работы, что позволяет получить многофункциональный доступ к базе и строить сложные запросы в самом коде, вместо сложных запросов SQL.

Для того чтобы осуществлять базовые CRUD операции над сущностями создается класс контекста базы данных CallContext, который представлен на рисунке 3.1.

Graphical user interface, text

Description automatically generated

Рисунок 3.1 – Класс доступа к базе данных

Также, для получения данных создаются сущности соответствующие таблицам базы. В классе контекста данные таблица представляются в виде коллекций сущностей DbSet<T>, где T конкретная сущность доменной области. Список сущностей представлен на рисунке 3.2.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated with medium confidence

Рисунок 3.2 – Сущности базы данных

Для обеспечения целостности и согласованности базы, создается поставщик контекста базы, рисунок 3.3, который оборачивает любое обращение к базе в транзакцию, которая подтверждается при успешном завершении запроса или отменяет все внесенные изменения, что позволяет базе находиться в состоянии согласованности данных.

Text

Description automatically generated

Рисунок 3.3 – Поставщик контекста базы DatabaseProvider

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

## **WEB-сервис CallAPI**

По требованию REST спецификации приложения должны быть разработаны на основе клиент-серверной архитектуры. В роли серверов выступают WEB-сервисы CallAPI, ServiceAPI, StatisticsAPI.

Данные сервисы написаны с использованием паттерна проектирования Mediatr. Данный подход позволяет реализовать принцип единственной ответственности у контроллеров и уменьшить количество зависимостей на другие внутренние сервисы, которые используются для выполнения команды. Пример использования медиатора можно увидеть на рисунке 3.4.

Text

Description automatically generated

Рисунок 3.4 – Пример использования паттерна Mediatr

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Как видно из рисунка, передача команд на выполнение запросов происходит с помощью одного метода Send. Для этого передаваемая команда должна реализовывать интерфейс IRequest<T>, где T возвращаемый тип данных для соответсвующей команды, рисунок 3.5.

Text

Description automatically generated

Рисунок 3.5 – Пример команды реализующей интерфейс

Также для обработки команды должен быть написан обработчик этой команды, под названием Handler. Данный обработчик реализовывает интерфейс IRequestHandler<TRequest, TResponce>, где TRequest это соответствующая команда для выполнения, а TResponce тип данных который будет возвращать данный обработчик и который совпадает с запрашиваемым типом данных в реализованном интерфейсе у команды, рисунок 3.6.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.6 – Пример обработчика команды

В роли клиентов, то есть потребителей данных сервисов, выступают настольное приложение Ambulance.Desktop и WEB-приложение Ambulance.Web.

Для данных потребителей предоставляются клиенты соответствующих WEB-сервисов, для вызова соответствующих точек доступа к данным. Соответствующие клиенты можно упаковать в подключаемую библиотеку и подключать в разные приложения с помощью управления NuGet пакетами.

Каждый набор приложений, состоящий из Ambulance.Desktop и Ambulance.WEB, представляет собой тенант со своим уникальным названием. Данное название передается с каждым запросов в адресной строке в качестве запрашиваемого параметра. Имя тенанта используется WEB-сервисами для получения адреса соответствующей базы данных.

Данный WEB-сервис позволяет работать с сущностями вызов, бригада скорой помощи и пациент, CallEntity, AmbulanceBrigadeEntity и PatientEntity, соответственно. Доступ к данным осуществляется через контроллеры, рисунок 3.7.

Text

Description automatically generated

Рисунок 3.7 – Структура WEB-сервиса CallAPI

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Контроллеры микросервисов написаны с помощью паттерна Mediatr. Для каждой команды написаны соответсвующие обработчики. Для работы с бригадами скорой помощи используется контроллер BrigadeController, рисунок 3.8.

Text

Description automatically generated

Рисунок 3.5 – Список методов BrigadeController

Метод Get, с параметром int, принимает идентификатор бригады и возвращает данные по ней в модели AmbulanceBrigade, рисунок 3.9, которая включает в себя номер бригады, фамилия, имя и отчество доктора, обоих фельдшеров, санитара и водителя, также возвращается тип бригады и название станции за которой она закреплена.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.9 – Модель AmbulanceBrigade

Метод GetBrigades возвращает коллекцию состоящая из моделей, представленной на рисунке 3.9.

Метод GetBrigadesMonitoringInfo возвращает коллекцию сущностей которые описывают состояние бригад скорой помощи на данный момент. Эта модель содержит в себе номер бригады, ее тип, улицу на которой находится бригада в данный момент, номер дома, уникальный идентификатор и номер вызова, дата и время последнего обновления состояния бригады и название станции к которой она привязана, модель представлена на рисунке 3.10.

Text

Description automatically generated

Рисунок 3.10 – Модель BrigadeMonitoringInfo

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Метод GetCalls, с параметром int, принимает уникальный идентификатор бригады и возвращает коллекцию моделей описывающие вызова которые обслужила конкретная бригада. Данные модели включают в себя номер вызова фамилию, имя и отчество пациента, улицу пациента, номер его дома и квартиры, возраст пациента, первичная причина вызова в виде предварительного диагноза, кто вызывает, дата и время получения вызова, дата и время передачи вызова конкретной бригаде, идентификаторы диспетчера принявший вызов и диспетчера передавший вызов, заметки к вызову и тип вызова, модель представлена на рисунке 3.11.

Graphical user interface, text

Description automatically generated

Рисунок 3.11 – Модель CallOfficeInfo

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Метод CreateBrigade, принимающий в параметрах модель запроса на создание сущности бригады, которая представлена на рисунке 3.12 и содержит в себе идентификаторы доктора, обоих фельдшеров, санитара, водителя, номер бригады, ее тип и название станции к которой привязана бригада, возвращает уникальный идентификатор созданной бригады скорой помощи.

Text

Description automatically generated

Рисунок 3.12 – Модель CreateBrigadeRequest

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Метод DeleteBrigade, принимает в параметрах модель для удаления, а именно изменения статуса бригады скорой помощи, которая представлена на рисунке 3.13. Данная модель включает в себя идентификатор бригады и время удаления бригады.

Text

Description automatically generated

Рисунок 3.13 – Модель DeleteBrigadeRequest

Метод AssignCallToBrigade, принимает в параметрах модель которая содержит данные для передачи вызова на выполнение бригаде, модель представлена на рисунке 3.14 и включает в себя идентификатор бригады и вызова, Дата и время передачи вызова бригаде и идентификатор диспетчера, который передал вызов.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

A screenshot of a computer

Description automatically generated with medium confidence

Рисунок 3.14 – Модель AssignCallRequest

Метод ReleaseBrigade, принимает в параметрах идентификатор бригады и возвращает статус код успешно выполненной операции. Этот метод используется для завершения выполнения вызова бригадой и перевод статсуса вызова в ожидающий обработки.

Метод ReturnCall, также как и метод ReleaseBrigade, принимает идентификатор бригады и возвращает статсу код успешно выполненной операции. Метод переводит вызов в статус ожидающий передачи бригаде.

Список команд и их обработчиков можно увидеть на рисунке 3.15. Все обработчики созданы на основе паттерна Mediatr и каждый обрабатывают свою уникальную команду.

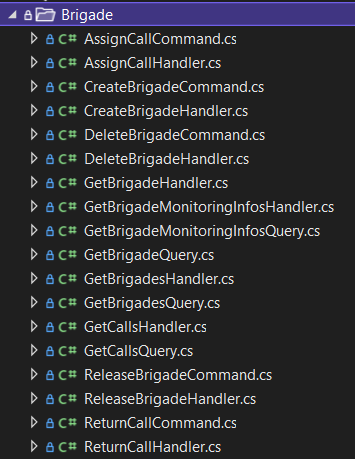


Рисунок 3.15 – Примеры команд и их обработчиков для BrigadeController

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Для работы с вызовами используется контроллер CallController, который представлен на рисунке 3.16.

Text

Description automatically generated

Рисунок 3.16 – Список методов контроллера CallController

Метод Get, принимающий в параметрах int, используется для получения вызова по его уникальному идентификатору, содержит номер вызова, фамилию, имя и отчество пациента, улицу пациента, номер его дома и квартиры, возраст пациента, первичная причина вызова в виде предварительного диагноза, кто вызывает, дата и время получения вызова, дата и время передачи вызова конкретной бригаде, идентификаторы диспетчера принявший вызов и диспетчера передавший вызов, заметки к вызову и тип вызова, модель представлена на рисунке 3.11.

Метод GetAccepted, возвращает коллекцию вызовов принятых для обслуживания но еще не переданные бригаде, данные вызовы имеют статус Accepted. Модель из которой состоит коллекция представлена на рисунке 3.11.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Метод GetPending, возвращает коллекцию обслуженных вызовов, данные вызовы имеют статус PendingForProcessing. Данная модель включает в себя полную информацию о вызове и представлена на рисунке 3.17 и 3.18.

Graphical user interface, text, application

Description automatically generated

Рисунок 3.17 – Модель CallFullOfficeInfo

Text

Description automatically generated

Рисунок 3.18 – Продолжение модели CallFullOfficeInfo

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Метод Create, принимает в параметрах модель для создания вызова, включающую в себя поля фамилия, имя, отчество, улица, номер дома и квартиры пациента, его возраст, идентификатор вызывающего, диспетчера принявшего вызов и диагноза, дата и время принятия, заметки по вызову, и тип вызова. Модель представлена на рисунке 3.19.

Text

Description automatically generated

Рисунок 3.19 – Модель CreateCallRequest

Метод ProcessCall, принимает в параметрах модель содержащую в себя полные данных о вызове для его финальной обработки. Данные модели представлены на рисунке 3.20 и 3.21.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.20 – Модель ProcessCallRequest

A screenshot of a computer

Description automatically generated with medium confidence

Рисунок 3.21 – Продолжение модели ProcessCallRequest

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Список команд и их обработчиков можно увидеть на рисунке 3.22. Данные обработчики также построены с использованием паттерна Mediatr.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.22 – Список команд и обработчиков для CallController

Для работы с пациентами используется PatientController, рисунок 3.23.

Text

Description automatically generated

Рисунок 3.23 – Список методов контроллера PatienController

Метод GetPatien, принимает в параметрах идентификатор пациента и возвращает информацию о нем, а именно фамилию, имя, отчество, возраст пациента, идентификатор улицы, номер дома и квартиры пациента, а также его пол. Модель представлена на рисунке 3.24.

Text

Description automatically generated

Рисунок 3.24 – Модель Patient

Метод Update, принимает в параметрах модель содержащую данные о пациенте для обновления. Данная модель представлена на рисунке 3.25.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.25 – Модель UpdatePatienRequest

Cписок команд и обработчиков представлен на рисунке 3.26.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.26 – Список команд и обработчиков для PatienController

Для возможности масштабирования и для того, чтобы потребителями данного сервиса могли быть приложения установленные в разных городах или регионах создана логика позволяющая получить адрес базы данных соответствующего города или региона. Так как каждый город или регион имеет свой уникальный идентификатор под названием Tenant, создана соответствующая логика, представленная на рисунке 3.27, позволяющая получить из строки запроса параметр с названием Tenant. Данный класс получает адрес базы данных, соответствующий полученному идентификатору приложения, и передает ее в контекст базы данных для установки соединения.

Text

Description automatically generated

Рисунок 3.27 – Получение строки подключения к базе

## **WEB-сервис ServiceAPI**

Данный WEB-сервис используется для получения и обработки сервисной информации такой как получения списков болезней, адресов, вызывающих людей, мест вызова, результатов, лекарств. Также, данный сервис используется для получения списка сотрудников, а именно список докторов, фельдшеров, санитаров, водителей и для получения данных диспетчера.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Для получения данных диспетчера используется контроллер DispatcherController, рисунок 3.28.

Graphical user interface, text, application

Description automatically generated

Рисунок 3.28 – Методы контроллера DispatcherController

Метод GetDispatcher принимает идентификатор диспетчера в параметрах и возвращает модель с полной информацией о диспетчере. Эта информация включает в себя фамилию, имя, отчество диспетчера и его идентификатор. Возвращаемая модель представлена на рисунке 3.29.

Text

Description automatically generated

Рисунок 3.29 – Модель Dispatcher

Медиатр данного контроллера использует команды и их обработчики указанные на рисунке 3.30.

Text

Description automatically generated

Рисунок 3.30 – Команды и обработчики DispatcherController

Для получения данных о сотрудниках используется контроллер EmployeeController, показанный на рисунке 3.31.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.31 – Методы контроллера EmployeeController

Метод GetDoktors, возвращает список моделей докторов, которые включают в себе информацию о сотруднике. Информация состоит из уникального идентификатора, фамилии, имени и отчества. Модель представлена на рисунке 3.32.

Text

Description automatically generated

Рисунок 3.32 – Модель сотрудника скорой помощи

Метод GetMedicalAssistants, возвращает список моделей фельдшеров, которые включают в себе информацию о сотруднике. Информация состоит из уникального идентификатора, фамилии, имени и отчества. Модель представлена на рисунке 3.32.

Метод GetOrderlies, возвращает список моделей санитаров, которые включают в себе информацию о сотруднике. Информация состоит из уникального идентификатора, фамилии, имени и отчества. Модель представлена на рисунке 3.32.

Метод GetDrivers, возвращает список моделей водителей, которые включают в себе информацию о сотруднике. Информация состоит из уникального идентификатора, фамилии, имени и отчества. Модель представлена на рисунке 3.32.

Медиатр данного контроллера использует команды и их обработчики указанные на рисунке 3.33.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Graphical user interface, text, chat or text message

Description automatically generated

Рисунок 3.33 – Команды и их обработчики для EmployeeController

Для получения сервисной информации, такой как список вызывающих, диагнозов, улиц, мест вызова, результатов вызова и лекарств используется контроллер ServiceController, представленный на рисунке 3.34.

Text

Description automatically generated

Рисунок 3.34 – Методы контроллера ServiceController

Метод GetCallers, возвращает список моделей вызывающих, тоесть тип людей, которые сообщают о вызове. Модель содержит в себе уникальный идентификатор и название, подробнее на рисунке 3.35.

Text

Description automatically generated

Рисунок 3.35 – Модель Caller

Метод GetDiagnosis, возвращает список моделей диагнозов. Модель содержит в себе уникальный идентификатор и название, подробнее на рисунке 3.36.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.36 – Модель Diagnosis

Метод GetStreets, возвращает список моделей улиц. Модель содержит в себе уникальный идентификатор и название, подробнее на рисунке 3.37.

Text

Description automatically generated

Рисунок 3.37 – Модель Street

Метод GetPlaces, возвращает список моделей мест вызова. Модель содержит в себе уникальный идентификатор и название, подробнее на рисунке 3.38.

Text

Description automatically generated

Рисунок 3.38 – Модель Place

Метод GetResults, возвращает список моделей результатов вызова. Модель содержит в себе уникальный идентификатор и название, подробнее на рисунке 3.39.

Text

Description automatically generated

Рисунок 3.39 – Модель Result

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Метод GetDrugs, возвращает список моделей лекарств. Модель содержит в себе уникальный идентификатор и название и дозу лекарства, подробнее на рисунке 3.40.

A screenshot of a computer

Description automatically generated with medium confidence

Рисунок 3.40 – Модель Drug

Медиатр данного контроллера использует команды и их обработчики указанные на рисунке 3.41.

Text

Description automatically generated

Рисунок 3.42 – Команды и их обработчики контроллера ServiceController

## **Клиенты микросервисов**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Клиенты для микросервисов создаются с целью обеспечения удобного и понятного API для разработки приложения-потребителя соответствующего микросервиса. Каждый клиент можно превратить в подключаемую библиотеку и поставлять отдельно, что позволяет получить доступ к соответствующему WEB-сервису независимо от другого.

В клиенте, который позволяет работать с WEB-сервисом CallAPI, создан класс-клиент, который содержит классы-ресурсы, с помощью который осуществляется доступ к соответствующим методам сервиса, рисунок 3.43.

Text

Description automatically generated

Рисунок 3.43 – Класс-клиент для CalAPI

Для каждого контроллера написан интерфейс который описывает адрес метода микросервиса, тип данных и как они будут передаваться. Для контроллера CallController написан ICallEndpoint, для контроллера BrigadeController написан IBrigadeEndpoint, для контроллера PatientController написан IPatientEndpoint. Интерфейсы представлены на рисунках 3.44, 3.45 и 3.46 соответственно.

Text

Description automatically generated

Рисунок 3.44 – Методы доступа CallController

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.45 – Методы доступа BrigadeController

Text

Description automatically generated

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Рисунок 3.46 – Методы доступа PatientController

Все вышеперечисленные интерфейсы при вызове методы, передают уникальный идентификатор приложения в строке запроса, рисунок 3.47, с помощью которого микросервисы будут получать соответствующую строку подключения к базе данных.

A screenshot of a computer

Description automatically generated with medium confidence

Рисунок 3.47 – Базовый контракт для интерфейсов доступа.

Также соответсвующий класс-клиент, для получения данных, создан для ServiceAPI, рисунок 3.48.

Text

Description automatically generated

Рисунок 3.48 – Класс-клиент для ServiceAPI

Для получения доступа к методам контроллеров созданы интерфейсы, опиывающие адрес и модель запроса к каждому методу контроллеров ServiceController, DispatcherController, EmployeeController, рисунки 3.49, 3.50, 3.51 соответственно.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.49 – Методы доступа ServiceController

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Text

Description automatically generated

Рисунок 3.50 – Методы доступа DispatcherController

A screenshot of a computer

Description automatically generated with medium confidence

Рисунок 3.51 – Методы доступа EmployeeController

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

## **Настольное приложение для диспетчеров**

При запуске приложения диспетчера встречает главное меню, с помощью которого можно ввести вызов, авторизоваться как диспетчер в системе, просмотреть принятые вызова и список бригад, зарегистрировать бригаду или обработать вызов. Главное меню представлено на рисунке 3.52.

Graphical user interface

Description automatically generated

Рисунок 3.52 – Главное окно приложения

Для авторизации выбирается соответствующая кнопка и производиться ввод уникального идентификатора диспетчера, рисунок 3.53. После авторизации становятся доступны все пункты главного окна приложения. В противном случае, пользователь не сможет работать с приложением.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Graphical user interface, application

Description automatically generated

Рисунок 3.53 – Авторизация диспетчера

Окно мониторинга работы скорой помощи, рисунок 3.54, позволяет просматривать принятые вызова, подробные данные о них и список бригад которые доступны или заняты обслуживанием вызова. Данное окно позволяет управлять работой бригады, а именно, передавать вызов бригаде, отменять вызов, завершать вызов и просматривать все вызова совершенные бригадой за смену, рисунок 3.55, просматривать данные о вызове на котором находится бригада, рисунок 3.56.

Graphical user interface

Description automatically generated

Рисунок 3.54 – Окно мониторинга работы бригад.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Table

Description automatically generated

Рисунок 3.55 – Список вызовов выполненные бригадой

Graphical user interface, text, application

Description automatically generated

Рисунок 3.56 – Данные вызова выполняемого бригадой

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

С главного окна можно получить доступ к окну регистрации бригад, рисунок 3.57. На нем нужно ввести ФИО доктора бригады, двух фельдшеров, санитара и водителя бригады. Также через это же окно происходит завершение работы бригады в конце рабочей смены.

Graphical user interface, application

Description automatically generated

Рисунок 3.57 – Регистрация бригад скорой помощи

Окно обработки вызовов позволяет выбрать вызов ожидающий обработки, ввести уточненные данные пациента и заполнить список лекарств которыми производилось лечение, рисунок 3.58.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Graphical user interface, application

Description automatically generated

Рисунок 3.58 – Окно обработки вызовов

## **WEB–приложение для бригад и персонала**

WEB-приложение должно позволить избавить бригады от бумажной работы. Также приложение позволяет просматривать вызовы и статистику по ним. Для просмотра вызовов выбирается соответствующий пункт меню и вводятся данные в поля для фильтрации, рисунок 3.59. Осуществлять поиск вызовов можно по номеру вызова, фамилии пациента, диагнозу, периоду времени в которое был принят вызов, улица, номер дома, номер квартиры пациента и результату. Для удобства просмотра реализована пагинация.

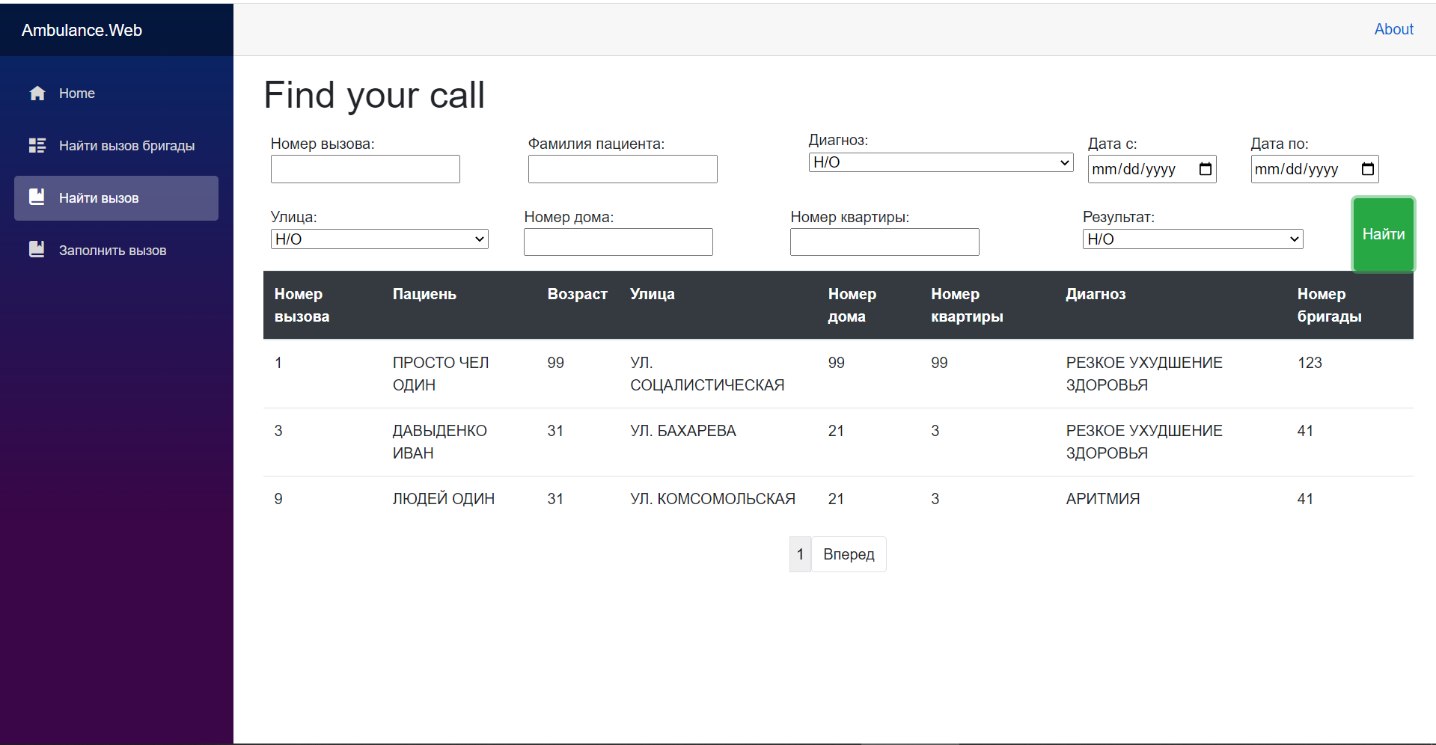


Рисунок 3.59 – Просмотр обработанных вызовов

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Для сотрудников бригады реализован пункт меню в котором можно занести всю дополнительную подробную информацию по вызову который обслуживает бригада. Данный пункт меню содержит информацию о состоянии пациента на момент вызова, при транспортировке, после транспортировки, состояние его органов и самочувствия. Для удобства выводится краткая информацию по вызову в верхней части экрана. Окно представлено на рисунке 3.60.

A screenshot of a computer

Description automatically generated

Рисунок 3.60 – Пример заполнения дополнительной информации по вызову

# **ТЕСТИРОВАНИЕ, ВЕРИФИКАЦИЯ И ВАЛИДАЦИЯ**

Тестирование программного продукта является важной частью разработка программного обеспечения. Программа которая не функционирует согласно требованиям или функционирует но основной процесс, ради которого разрабатывалось приложение, не работает, не может конкурировать и быть востребованным на рынке.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Для проверки работоспособности программы создаются тест-кейсы с описанием шагов по воспроизведению теста и описанием ожидаемого поведения программы на данные шаги. Особенно важно создать тест-кейсы для критических путей программы, то есть для тех функций которые несут в себе основной смысл разрабатываемого продукта.

Верификация – это процесс оценки того, насколько система (программа, устройство) по итогам некоторого этапа ее разработки соответствует условиям, заданным в начале этапа.

Валидация – процесс оценки того, насколько система (программа, устройство) соответствует требованиям по ее назначению.

Основное правило разрабатываемого настольного приложения – с приложением должен работать диспетчер. То есть для работы с приложением нужно авторизоваться как диспетчер, в противном случае пункты меню не будут доступны. Тест-кейсы описаны в таблице 4.1 и 4.2.

Таблица 4.1 – Негативные тест-кейсы для главного окна Desktop приложения.

|  |  |  |
| --- | --- | --- |
| **Описание** | **Шаги воспроизведения** | **Ожидаемый результат** |
| Попытка ввести вызов | * загрузка Desktop приложения; * заполнение полей вызова; * нажатие на кнопку «Принять». | Появляется новое окно с сообщением «Вы не зарегистрировались как диспетчер» |
| Попытка открытия окна с принятыми вызовами | * загрузка Desktop приложения; * заполнение полей вызова; * нажатие на кнопку «Передача вызова». |
| Попытка открытия окна обработки вызовов | * загрузка Desktop приложения; * заполнение полей |

Продолжение таблицы 4.1

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

|  |  |  |
| --- | --- | --- |
|  | * вызова; * нажатие на кнопку «Обработка». |  |
| Попытка открытия окна для регистрации бригад | * загрузка Desktop приложения; * заполнение полей вызова; * нажатие на кнопку «Данные». |

На рисунке 4.1 представлен результат выполнения тест-кейсов описанных в таблице 4.1.

Graphical user interface

Description automatically generated

Рисунок 4.1 – Сообщение для неавторизованного пользователя

Таблица 4.2 – Положительные тест-кейсы для главного окна Desktop приложения

|  |  |  |
| --- | --- | --- |
| **Описание** | **Шаги воспроизведения** | **Ожидаемый результат** |
| Попытка ввести вызов | * загрузка Desktop приложения; * авторизация диспетчером; | Появляется новое окно с надписью о принятии вызова и его номером. |

Продолжение таблицы 4.2

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

|  |  |  |
| --- | --- | --- |
|  | * заполнение полей вызова; * нажатие на кнопку «Принять». |  |
| Попытка открытия окна с принятыми вызовами | * загрузка Desktop приложения; * авторизация диспетчером; * заполнение полей вызова; * нажатие на кнопку «Передача вызова». | Появляется новое окно с принятыми вызовами и списком бригад |
| Попытка открытия окна обработки вызовов | * загрузка Desktop приложения; * авторизация диспетчером; * заполнение полей вызова; * нажатие на кнопку «Обработка». | Появляется окно для обработки вызова со списком вызовов ожидающие обработки |
| Попытка открытия окна для регистрации бригад | * загрузка Desktop приложения; * авторизация диспетчером; * заполнение полей вызова; * нажатие на кнопку «Данные». | Появляется новое окно для ввода и удаления бригад. |

Пример авторизации и результат одного из тест-кейсов представлены на рисунках 4.2 и 4.3 соответственно.

Graphical user interface

Description automatically generated

Рисунок 4.2 – Авторизация диспетчера

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

A screenshot of a computer

Description automatically generated

Рисунок 4.3 – Открытие окна регистрации бригад авторизованным пользователем

# ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ДИПЛОМНОЙ РАБОТЫ

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

## Расчёт общей трудоемкости разработки программного обеспечения

Общий объём трудоёмкости разработки программного комплекса по автоматизации планирования и организации собеседований () определяется исходя из количества и объёма функций, реализуемых программой, по каталогу функций ПО в соответствии с таблицей 1.1 приложения 1 источника [6] по формуле (5.1):

, (5.1)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | объем отдельной функции ПО; |
|  |  | – | общее число функций. |

Анализируя разработанную программу, уточнённый объём ПО () определяем по формуле (5.2):

, (5.2)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | уточнённый объём отдельной функции ПО в строках исходного кода (LOC); |

Сравнение исходного и уточнённых объёмов строк исходного кода представлены в таблице Б.1 приложения Б.

Разработанное в ходе выполнения дипломной работы приложение относится к третьей категории сложности.

На основании принятого к расчёту (уточнённого) объёма () и категории сложности ПО определяется нормативная трудоемкость ПО () выполняемых работ, которая приведена в таблице 5.1.

Таблица 5.1 – Нормативная трудоёмкость на разработку ПО ()

|  |  |  |
| --- | --- | --- |
| **Уточнённый объем,** | **3-я категория сложности ПО** | **Номер нормы** |
| 4000 | 162 | 43 |

Дополнительные затраты труда, связанные с повышением сложности разрабатываемого ПО, учитываются посредством коэффициента повышения сложности ПО (). Кс рассчитывается по формуле (5.3):

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

*,* (5.3)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | коэффициент, соответствующий степени повышения сложности; |
|  |  | – | количество учитываемых характеристик. |

Таким образом:

Новизна разработанного ПО определяется путем экспертной оценки данных, полученных при сравнении характеристик разрабатываемого ПО с имеющимися аналогами. Влияние фактора новизны на трудоёмкость учитывается путем умножения нормативной трудоёмкости на соответствующий коэффициент, учитывающий новизну ПО (). Разработанная программа обладает категорией новизны В, а значение .

Современные технологии разработки компьютерных программ предусматривают широкое использование коробочных продуктов (пакетов, модулей, объектов). Степень использования в разрабатываемом ПО стандартных модулей определяется их удельным весом в общем объеме ПО.

В данном программном комплексе используется до 20% стандартных модулей, что соответствует значению коэффициента .

Приложение разработано на языке *C#*, что соответствует коэффициенту функционирования в локальных сетях, учитывающему средства разработки ПО, .

Значения коэффициентов удельных весов трудоёмкости стадий разработки ПО в общей трудоемкости ПО определяются с учётом установленной категории новизны ПО согласно таблице 2.5 [6].

При этом сумма значений коэффициентов удельных весов всех стадий в общей трудоёмкости равна единице. Значения коэффициентов приведены в таблице 5.2.

Таблица 5.2 – Значения коэффициентов удельных весов трудоёмкости стадий разработки ПО в общей трудоёмкости ПО

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Категория новизны ПО** | **Без применения CASE-технологии** | | | | |
| **Стадии разработки ПО** | | | | |
| ТЗ | ЭП | ТП | РП | ВН |
| **Значения коэффициентов** | | | | |

Продолжение таблицы 5.2

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Категория новизны ПО** | **Значения коэффициентов** | | | | |
|  |  |  |  |  |
| В | 0,08 | 0,19 | 0,28 | 0,34 | 0,11 |

Нормативная трудоёмкость ПО () выполняемых работ по стадиям разработки корректируется с учетом коэффициентов: повышения сложности ПО (), учитывающих новизну ПО (), учитывающих степень использования стандартных модулей (), средства разработки ПО () и определяются по формулам:

* для стадии ТЗ по формуле (5.4):

(5.4)

* для стадии ЭП по формуле (5.5):

(5.5)

* для стадии ТП по формуле (5.6):

(5.6)

* для стадии РП по формуле (5.7):

(5.7)

* для стадии ВН по формуле (5.8):

(5.8)

Коэффициенты вводятся на всех стадиях разработки, а коэффициент вводится только на стадии РП.

Таким образом:

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Общая трудоёмкость разработки ПО () определяется суммированием нормативной (скорректированной) трудоёмкости ПО по стадиям разработки по формуле (5.9):

, (5.9)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | нормативная (скорректированная) трудоёмкость разработки ПО на *i*-й стадии (чел/дней); |
|  |  | – | количество стадий разработки. |

Таким образом:

Распределение скорректированной трудоемкости ПО по стадиям вычисляется с помощью: распределения нормативной трудоемкости, коэффициента , коэффициента учитывающий степень использования стандартных модулей , коэффициента учитывающего новизну  и коэффициента повышения сложности :

* для стадии ТЗ по формуле (5.10):

, (5.10)

* для стадии ЭП по формуле (5.11):

, (5.11)

* для стадии ТП по формуле (5.12):

, (5.12)

* для стадии РП по формуле (5.13):

, (5.13)

* для стадии ВН по формуле (5.14):

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

, (5.14)

Таким образом:

Общая скорректированная трудоёмкость разработки ПО () определяется суммированием (скорректированной) трудоёмкости ПО по стадиям разработки по формуле (5.15):

, (5.15)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | нормативная (скорректированная) трудоёмкость разработки ПО на *i*-й стадии (чел/дней); |
|  |  | – | количество стадий разработки. |

Таким образом:

Результаты расчётов по определению нормативной и скорректированной трудоёмкости ПО по стадиям разработки и общую трудоёмкость разработки ПО () представлены в таблице В.1 приложения В.

## Расчет совокупных капитальных вложений в проект

В общем виде совокупность капитальных вложений в проект может быть рассчитан по формуле (5.16):

, (5.16)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | стоимость устанавливаемого оборудования, руб.; |
|  |  | – | недоамортизированная часть стоимости демонтируемого оборудования, руб.; |
|  |  | – | ликвидационная стоимость (выручка от продажи) демонтируемого оборудования, руб.; |
|  |  | – | стоимость приобретенных программных продуктов, руб. |

Таким образом:

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

руб.

Так как равен стоимости устанавливаемого компьютера, стоимость которого равна 2590 руб. Остальные коэффициенты равны нулю так как оборудование не демонтировалось и программные продукты не преобретались.

## Расчёт затрат на разработку программного продукта

В состав затрат на разработку системы по автоматизации учёта лабораторных испытаний промышленного предприятия входят следующие статьи расходов:

* затраты труда на создание программного продукта (затраты по основной, дополнительной заработной плате и соответствующие отчисления) ();
* затраты на изготовление эталонного экземпляра ();
* затраты на технологию (затраты на приобретение и освоение программных средств, используемых при разработке программного продукта; затраты на ПО, используемое как эталон) ();
* затраты на машинное время (расходы на содержание и эксплуатацию технических средств разработки, эксплуатации и сопровождения) ();
* затраты на материалы (информационные носители) ();
* затраты на энергию, на использование каналов связи (для отдельных видов);
* общепроизводственные расходы (затраты на управленческий персонал, на содержание помещений) ();
* непроизводственные (коммерческие) расходы (затраты, связанные с рекламой, поиском заказчиков, поставками конкретных экземпляров) ().

В таблице Г.1 приложения Г приведены значения основных параметров, необходимых для расчёта затрат на разработку программного продукта.

Суммарные затраты на разработку ПО () определяются по формуле (5.17):

, (5.17)

Расходы на оплату труда разработчиков с отчислениями определяются по формуле (5.18):

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

, (5.18)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | основная заработная плата разработчиков, руб.; |
|  |  | – | дополнительная заработная плата разработчиков, руб.; |
|  |  | – | сумма отчислений от заработной платы (социальные нужды, страхование от несчастных случаев), руб. |

Основная заработная плата разработчиков рассчитывается по формуле (5.19):

, (5.19)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | средняя часовая тарифная ставка; |
|  |  | – | общая трудоемкость разработки, чел-час; |
|  |  | – | коэффициент, учитывающий доплаты стимулирующего характера. |

Средняя часовая тарифная ставка определяется по формуле (5.20):

, (5.20)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | часовая тарифная ставка разработчика *i* – й категории; |
|  | *ni* | – | количество разработчиков *i*-й категории. |

Часовая тарифная ставка разработчика *i*-й категории определяется по формуле (5.21):

, (5.21)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | базовая ставка; |
|  | *k* | – | тарифный коэффициент. |

Таким образом:

.

.

Дополнительная заработная плата определяется по формуле (5.22):

, (5.22)

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | норматив отчислений на дополнительную заработную плату разработчиков. |

Таким образом:

Отчисления от основной и дополнительной заработной платы (отчисления на социальные нужды и обязательное страхование) рассчитываются по формуле (5.23):

, (5.23)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | процент отчислений на социальные нужды и обязательное страхование от суммы основной и дополнительной заработной платы (). |

Затраты машинного времени определяются по формуле (5.24):

, (5.24)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | стоимость 1 часа машинного времени (руб./ч.); |
|  |  | – | коэффициент мультипрограммности, показывающий распределение времени работы ЭВМ в зависимости от количества пользователей ЭВМ; |
|  | =1 | – |  |
|  |  | – | машинное время ЭВМ, необходимое для разработки и отладки проекта (ч.). |

Стоимость машино-часа определяется по формуле (5.25):

, (5.25)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | затраты на заработную плату обслуживающего персонала с учетом всех отчислений, (руб. в год); |
|  |  | – | стоимость аренды помещения под размещение вычислительной техники, (руб. в год); |
|  |  | – | амортизационные отчисления за год, (руб. в год) |
|  |  | – | затраты на электроэнергию, (руб. в год); |
|  |  | – | затраты на материалы, необходимые для обеспечения нормальной работы ПЭВМ (вспомогательные), (руб. в год); |
|  |  | – | затраты на текущий и профилактический ремонт ЭВМ (руб. в год); |
|  |  | – | прочие затраты, связанные с эксплуатацией ПЭВМ (руб. в год); |
|  |  | ­ | действительный фонд времени работы ЭВМ (час/год). |

Все статьи затрат формируются в расчете на единицу ПЭВМ.

Затраты на заработную плату обслуживающего персонала () определяются по формуле (5.26):

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

, (5.26)

,

,

,

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | основная заработная плата обслуживающего персонала, руб.; |
|  |  | – | дополнительная заработная плата обслуживающего персонала, руб.; |
|  |  | – | сумма отчислений от заработной платы (социальные нужды, страхование от несчастных случаев), руб.; |
|  |  | – | количество обслуживаемых ПЭВМ, шт.; |
|  |  | – | месячная тарифная ставка *i*-го работника, руб.; |
|  |  | – | численность обслуживающего персонала, чел.; |
|  |  | – | процент дополнительной заработной платы обслуживающего персонала от основной; |
|  |  | – | процент отчислений на социальные нужды и обязательное страхование от суммы основной и дополнительной заработной платы. |

Тарифная ставка 7-го разряда обслуживающего персонала:

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Годовые затраты на аренду помещения () определяются по формуле (5.27):

, (5.27)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | средняя годовая ставка арендных платежей, руб./м2; |
|  |  | – | площадь помещения, м2; |
|  |  | – | количество ПЭВМ, шт. |

Сумма годовых амортизационных отчислений () определяется по формуле (5.28):

, (5.28)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| где | – | затраты на приобретение (стоимость) единицы ПЭВМ, руб.; | | |
|  | – | коэффициент, характеризующий дополнительные затраты, связанные с доставкой, монтажом и наладкой оборудования, от ; | | |
|  |  | | – | балансовая стоимость ЭВМ, руб.; |
|  | – | норма амортизации, %. | | |

Стоимость электроэнергии, потребляемой за год, (ЗЭП) определяется по формуле (5.29):

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

, (5.29)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| где | М | – | паспортная мощность ПЭВМ, (кВт), М = 0,06 кВт; | |
|  |  | – | стоимость одного кВт-часа электроэнергии, руб.; | |
|  |  | – | общее количество дней в году; | |
|  | – | | | число выходных и праздничных дней; |
|  |  | – | продолжительность 1 смены; | |
|  |  | – | коэффициент сменности; | |
|  |  | – | коэффициент, учитывающий потери рабочего времени, связанные с профилактикой и ремонтом ЭВМ; | |
|  |  | – | Действительный годовой фонд времени работы ПЭВМ. | |

Затраты на материалы (), необходимые для обеспечения нормальной работы ПЭВМ составляют около 1% от балансовой стоимости ЭВМ и определяются формулой (5.30):

, (5.30)

|  |  |  |
| --- | --- | --- |
| где | – | затраты на приобретение (стоимость) ЭВМ, руб.; |
|  | – | коэффициент, характеризующий дополнительные затраты, связанные с доставкой, монтажом и наладкой оборудования, от ; |
|  | – | коэффициент, характеризующий затраты на вспомогательные материалы (). |

Затраты на текущий и профилактический ремонт () принимаются равными 7% от балансовой стоимости ЭВМ и рассчитываются по формуле (5.31):

, (5.31)

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | коэффициент, характеризующий затраты на текущий и профилактический ремонт (). |

Прочие затраты, связанные с эксплуатацией ЭВМ (), состоят из амортизационных отчислений на здания, стоимости услуг сторонних организаций, составляют 5 % от балансовой стоимости и рассчитываются по формуле (5.32):

, (5.32)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | коэффициент, характеризующий размет прочих затрат, связанных с эксплуатацией ЭВМ (). |

Для расчета машинного времени ЭВМ ( в часах), необходимого для разработки и отладки проекта, следует использовать формулу (5.33), а для расчета затраты машинного времени формулу (5.34):

, (5.33)

(5.34)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | срок реализации стадии «Рабочий проект» (РП), 30 дней; |
|  |  | – | срок реализации стадии «Ввод в действие» (ВП), 14 дня; |
|  |  | – | продолжительность рабочей смены, (ч.), ; |
|  |  | – | количество рабочих смен, . |

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

При написании дипломной работы были использованы среда разработки *Visual Studio 2019* и локальная СУБД *SQL Server 2019*, поэтому затраты на технологию () и изготовление эталонного экземпляра () и затраты на материалы () будут нулевыми.

Общепроизводственные затраты рассчитываются по формуле (5.35):

, (5.35)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | норматив общепроизводственных затрат. |

Непроизводственные затраты рассчитываются по формуле (5.36):

, (5.36)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | норматив непроизводственных затрат. |

Итого получаем суммарные затраты на разработку:

Результаты расчетов приведены в таблице Д.1 приложения Д.

## Формирование цены при создании программного обеспечения

Оптовая цена ПО () определяется следующей формулой (5.37):

, (5.37)

,

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | себестоимость ПО, руб.; |
|  |  | – | прибыль от реализации ПО, руб.; |
|  |  | – | уровень рентабельности ПО, % (). |

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Прогнозируемая отпускная цена ПО с НДС рассчитывается по формуле (5.38):

, (5.38)

Налог на добавленную стоимость () рассчитывается по формуле (5.39):

, (5.39)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | ставка налога на добавленную стоимость, %, |

Результаты расчетов формирования цены на разработку программы приведены в таблице 5.3.

Таблица 5.3 – Расчет формирования цены на разработку программы

|  |  |
| --- | --- |
| **Наименование статьи расходов** | **Значение** |
| 1 | 2 |
| Суммарные затраты на разработку ПО | 19319,43 |
| Полная себестоимость | 19319,43 |
| Прибыль от реализации ПО | 5795,82 |
| Отпускная цена ПО без НДС | 25115,25 |
| Налог на добавленную стоимость | 5023,05 |
| Отпускная цена ПО с НДС | 30138,30 |

## Статистическая оценка экономической эффективности проекта

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Данный продукт не является новым на рынке, но предоставляет удобный интерфейс и расширенный функционал. Стоимость аналогичных мониторинговых систем () равна 65000 руб. Эффект может быть рассчитан по формуле (5.40):

Э(П) = – , (5.40)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | текущие и инвестиционные затраты по новому проекту, руб. |

Э(П) = 65000 – 30138,30 = 34861,70 руб.

Рентабельность затрат (З) или инвестиций (И) на новую информационную технологию, программный продукт рассчитывается по формуле (5.41):

, (5.41)

180,44%

Простой срок окупаемости затрат может быть расчитан по следующей формуле (5.42):

, (5.42)

лет

Годовой экономический эффект может быть рассчитан по следующей формуле (5.43):

, (5.43)

|  |  |  |  |
| --- | --- | --- | --- |
| где |  | – | рентабельность затрат (инвестиций) базового варианта, руб. |

На основании выполненных расчетов была сформирована таблица технико-экономических показателей проекта в таблице 5.1.

Таблица 5.4 – Технико-экономические показатели проекта

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Наименование показателя** | | **Единица измерения** | | **Проектный вариант** | |
| **Показатели конкурентоспособности** | | | | | | |
| 1.1 | Интегральный коэффициент конкурентоспособности | | – | | – | |
| 1.2 | Коэффициент эквивалентности | | – | | – | |
| 1.3 | Коэффициент соответствия нормативам | | – | | – | |
| 1.4 | Коэффициент цены потребления | | – | | – | |
| **Показатели затрат на разработку** | | | | | | |
| 2 | Общая трудоемкость разработки ПО | | чел.-дн. | | 134 | |
| 3 | Капитальные вложения в проекте | | руб. | | 2590 | |
| 4 | Затраты на разработку программы | | руб. | | 19319,42 | |
| 4.1 | Затраты на оплату труда разработчиков | | руб. | | 9135,22 | |
| 4.2 | Затраты машинного времени | | руб. | | 4264,23 | |
| 4.3 | Затраты на изготовление эталонного экземпляра | | руб. | | 0 | |
| 4.4 | Затраты на технологию | | руб. | | 0 | |
| 4.5 | Затраты на материалы | | руб. | | 29,01 | |
| 4.6 | Общепроизводственные затраты | | руб. | | 456,76 | |
| 4.7 | Непроизводственные (коммерческие) затраты | | руб. | | 456,76 | |
| **Показатели стоимости** | | | | | | |
| 5 | | Число снимаемых копий ПП | | шт. | | 0 |
| 6 | | Отпускная цена ПП с НДС | | руб. | | 30138,30 |
| 7 | | Розничная цена ПП | | руб. | |  |
| **Показатели эксплуатационных затрат** | | | | | | |
| 8 | | Затраты на внедрение и адаптацию ПП | | руб. | | 0 |
| 9 | | Затраты на эксплуатацию ПП | | руб. | |  |
| **Показатели экономической эффективности** | | | | | | |
| 10 | | Рентабельность затрат | | % | | 180,44 |
| 11 | | Простой срок окупаемости проекта | | лет | | 0,55 |
| 12 | | Годовой экономический эффект | | руб. | | 29065,87 |

Продолжение таблицы 5.4

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

|  |  |  |  |
| --- | --- | --- | --- |
| 13 | Приведенные затраты | руб. | – |
| 14 | Чистый дисконтированный доход | руб. | – |
| 15 | Внутренняя норма доходности | % | – |
| 16 | Индекс рентабельности (доходности) | % | – |
| 17 | Динамический срок окупаемости | Лет | – |
| **Показатели частной экономической эффективности** | | | |
| 18 | Экономический эффект от производства нового ПО | руб. | – |
| 19 | Экономический эффект от использования нового ПО | руб. | – |
| 20 | Экономия затрат на оплату машинного времени | руб. | – |
| 21 | Экономия затрат на оплату труда специалистов | руб. | – |
| 22 | Экономия затрат на материалы | руб. | – |

После оценки технико-экономических показателей проектного программного обеспечения можно сделать вывод о том, что реализация проекта является обоснованной и экономически целесообразной при полной себестоимости продукта в 19319,43 руб. и отпускной цене ПО в размере 30138,30 руб. срок окупаемости составит 0,55 лет. Так как срок окупаемости меньше года, то не требуется рассчитывать динамическую оценку эффективности проекта.

# ГОСУДАРСТВЕННАЯ ПОЛИТИКА В ОБЛАСТИ ОХРАНЫ ТРУДА

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

В Республике Беларусь вопросам безопасности труда придается безусловно приоритетное значение. Право работника на безопасный труд – одна из важнейших гарантий в области труда, установленных Конституцией Республики Беларусь: «человек, его права, свободы и гарантии их реализации, являются высшей ценностью и целью общества и государства» [7].

Отличительная черта политики в области охраны труда в Республике Беларусь – это активное влияние государства на процессы в сфере охраны труда через доступные ему механизмы и процедуры. Это, прежде всего, разработка и принятие законодательства, которое устанавливает обязательные требования и гарантии, определяет компетенцию в области охраны труда всех заинтересованных сторон [9].

На сегодняшний день в республике создана развитая законодательная база в области охраны труда, которая учитывает имеющийся положительный международный опыт и многолетнюю практику регулирования этих вопросов на национальном уровне [9].

Основным нормативным документом в области охраны труда является Закон Республики Беларусь «Об охране труда», который был принят в 2008 году [11].

На основании данного закона создана целостная система управления охраной труда, определяющая роль и задачи каждого из участников процесса обеспечения безопасных условий труда на всех уровнях: от республиканского уровня до конкретных организаций, должностных лиц и работников.

Она позволяет комплексно решать вопросы обеспечения безопасности на рабочих местах, создавая необходимые условия для реализации права на работу в безопасных условиях труда, а также на защиту работником своих прав и получение соответствующих компенсаций при наступлении несчастного случая на производстве или профессионального заболевания [9].

Весь комплекс мероприятий в области охраны труда, от принятия законодательства и разработки систем управления охраной труда до государственного надзора и общественного контроля, направлен на решение одной главной задачи – сокращение производственного травматизма. В Республике Беларусь в этом вопросе удалось достичь положительных результатов.

В целях обеспечения прав граждан на безопасный труд в Республике Беларусь с 1993 года введен институт государственного надзора за соблюдением законодательства об охране труда [9].

В настоящее время его осуществляет специально созданный надзорный орган – Департамент государственной инспекции труда Министерства труда и социальной защиты Республики Беларусь [9].

В решении вопросов охраны труда активно участвуют и социальные партнеры. Закон Республики Беларусь «О профессиональных союзах» наделил профсоюзы правами по защите прав работников на безопасные условия труда в рамках общественного контроля.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Меры по обеспечению безопасного труда являются неотъемлемой частью общей системы управления организацией. В этой работе учитываются особенности конкретного предприятия, те риски, с которыми связаны осуществляемые на нем производственные процессы.

Учесть данные особенности и обеспечить комплексную работу по управлению рисками в области охраны труда призвана система управления охраной труда. Закон Республики Беларусь «Об охране труда» обязывает нанимателя разрабатывать, внедрять и поддерживать функционирование системы управления охраной труда [9].

В Республике Беларусь разработана Концепция государственного управления охраной труда, которая определяет, что цель государственной политики в области охраны труда – сохранение жизни и здоровья граждан в процессе трудовой деятельности. Основными принципами государственной политики в области охраны труда являются:

1. приоритет жизни и здоровья работников по отношению к результатам производственной деятельности;
2. обеспечение гарантий права работников на охрану труда;
3. установление обязанностей всех субъектов правоотношений в области охраны труда, полной ответственности работодателей за обеспечение здоровых и безопасных условий труда;
4. совершенствование правоотношений и управления в этой сфере, включая внедрение экономического механизма обеспечения охраны труда [12].

Основными направлениями государственной политики в области охраны труда являются:

1. приоритет сохранения жизни и здоровья работающих;
2. ответственность работодателя за создание здоровых и безопасных условий труда;
3. комплексное решение задач охраны труда на основе республиканских, отраслевых и территориальных целевых программ по улучшению условий и охраны труда с учетом направлений экономической и социальной политики, достижений в области науки и техники;
4. социальная защита работающих, возмещение вреда лицам, потерпевшим при несчастных случаях на производстве и (или) получившим профессиональные заболевания;
5. установление единых требований по охране труда для всех работодателей;
6. использование экономических методов управления охраной труда, участие государства в финансировании мероприятий по улучшению условий и охраны труда;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

1. информирование граждан, обучение работающих по вопросам охраны труда;
2. взаимодействие республиканских органов государственного управления и иных государственных организаций, подчиненных Правительству Республики Беларусь, местных исполнительных и распорядительных органов, органов, уполномоченных на осуществление контроля (надзора), профессиональных союзов, работодателей; сотрудничество между работодателями и работающими;
3. использование международного опыта организации работы по улучшению условий и повышению безопасности труда [12].

Государственное управление в области охраны труда осуществляют Президент Республики Беларусь, Правительство Республики Беларусь, республиканские органы государственного управления и иные государственные организации, подчиненные Правительству Республики Беларусь, местные исполнительные и распорядительные органы в пределах своей компетенции [12].

Президент Республики Беларусь определяет единую государственную политику в области охраны труда и осуществляет иные полномочия в этой области в соответствии с Конституцией Республики Беларусь, Законом «Об охране труда» и иными законодательными актами [12].

Согласно Директивы Президента Республики Беларусь от 11.03.2004 г №1 «О мерах по укреплению общественной безопасности и дисциплины» руководители государственных органов, иных организаций независимо от форм собственности должны обеспечивать:

1. здоровые и безопасные условия труда, промышленную, пожарную, ядерную и радиационную безопасность, безопасность движения и эксплуатации транспорта;
2. систематический контроль физического состояния работников, занятых на работах с вредными и (или) опасными условиями труда или повышенной опасностью, путем проведения освидетельствований и (или) медицинских осмотров для исключения чрезвычайных происшествий и производственного травматизма;
3. обеспечивать безусловное привлечение работников к дисциплинарной ответственности вплоть до увольнения за появление на работе в состоянии алкогольного, наркотического или токсического опьянения, а также распитие спиртных напитков, употребление наркотических средств, психотропных веществ, их аналогов, токсических веществ в рабочее время или по месту работы, а также за нарушение требований по охране труда, повлекшее увечье или смерть других работников [10].

Правительство Республики Беларусь обеспечивает проведение единой государственной политики в области охраны труда, организует разработку республиканских целевых программ по улучшению условий и охраны труда, осуществляет иные полномочия в этой области в соответствии с Конституцией Республики Беларусь, Законом «Об охране труда», иными законами и актами Президента Республики Беларусь.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Республиканские органы государственного управления и иные государственные организации, подчиненные Правительству Республики Беларусь, осуществляют:

1. государственное управление охраной труда на отраслевом уровне;
2. разработку и принятие отраслевых правил по охране труда, типовых инструкций по охране труда, других нормативных правовых актов, в том числе технических нормативных правовых актов, содержащих требования по охране труда, по согласованию с Министерством труда и социальной защиты, другими республиканскими органами государственного управления, осуществляющими регулирование и управление в соответствующих сферах деятельности;
3. разработку и реализацию отраслевых целевых программ по улучшению условий и охраны труда;
4. анализ результатов аттестации рабочих мест по условиям труда, паспортизации санитарно-технического состояния условий и охраны труда, причин производственного травматизма и профессиональной заболеваемости, разработку и реализацию мер по их профилактике; контроль за соответствием требованиям законодательства об охране труда деятельности организациями (ведомственный контроль);
5. организацию обучения и проверки знаний по вопросам охраны труда руководителей и специалистов;
6. организацию и координацию проведения научно-исследовательских работ по вопросам условий и охраны труда;
7. информационное обеспечение организаций по вопросам охраны труда;
8. пропаганду и распространение передового опыта в области охраны труда в организациях;
9. международное сотрудничество по вопросам охраны труда;
10. иные полномочия в области охраны труда, предусмотренные законодательством [12].

Министерство труда и социальной защиты Республики Беларусь помимо указанных выше полномочий:

1. организует взаимодействие по вопросам охраны труда республиканских органов государственного управления и иных государственных организаций, подчиненных Правительству Республики Беларусь, местных исполнительных и распорядительных органов, профсоюзов;
2. разрабатывает предложения об основных направлениях и приоритетах государственной политики в области охраны труда, нормативные правовые акты по условиям и охране труда;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

1. осуществляет разработку республиканских целевых программ по улучшению условий и охраны труда;
2. утверждает самостоятельно или совместно с республиканскими органами государственного управления межотраслевые правила по охране труда, типовые инструкции по охране труда, другие нормативные правовые акты, содержащие требования по охране труда;
3. разрабатывает и утверждает типовые нормы бесплатной выдачи работникам средств индивидуальной защиты, перечень средств индивидуальной защиты, нормативные правовые акты по вопросам планирования и разработки мероприятий по охране труда, проведения контроля за соблюдением законодательства об охране труда в организации, проведения паспортизации санитарно-технического состояния условий и охраны труда;
4. организует осуществление надзора за соблюдением законодательства об охране труда;
5. организует проведение государственных экспертиз условий труда и качества проведения аттестации рабочих мест по условиям труда, осуществление надзора за соблюдением законодательства о труде по вопросам предоставления компенсаций работникам за работу с вредными и (или) опасными условиями труда;
6. осуществляет мониторинг состояния условий и охраны труда, анализ нарушений законодательства об охране труда, причин производственного травматизма и вносит предложения по их предупреждению;
7. организует создание и актуализацию банка данных результатов аттестации рабочих мест по условиям труда;
8. осуществляет аккредитацию юридических лиц (индивидуальных предпринимателей) на оказание услуг в области охраны труда;
9. ведет государственную статистику по вопросам условий и охраны труда;
10. осуществляет организационно-техническое обеспечение деятельности республиканской комиссии для проверки знаний руководителей и членов комиссий республиканских органов государственного управления и иных государственных организаций по вопросам охраны труда;
11. обеспечивает иные полномочия в области охраны труда [12].

Местные исполнительные и распорядительные органы осуществляют:

1. государственное управление охраной труда на территориальном уровне;
2. разработку и реализацию территориальных целевых программ по улучшению условий и охраны труда;
3. надзор за соблюдением законодательства об охране труда, законодательства о труде по вопросам предоставления компенсаций работникам за работу с вредными и (или) опасными условиями труда в организациях и ведомственный контроль;

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

1. организацию обучения и проверки знаний по вопросам охраны труда руководителей и специалистов организаций, расположенных на подведомственной им территории и иных государственных организаций, подчиненных Правительству Республики Беларусь;
2. анализ причин производственного травматизма и профессиональной заболеваемости в организациях, разработку и реализацию мер по их профилактике;
3. информационное обеспечение организаций по вопросам охраны труда;
4. пропаганду и распространение передового опыта в области охраны труда в организациях;
5. участие в разработке проектов нормативных правовых актов, в том числе технических нормативных правовых актов, содержащих требования по охране труда, в международном сотрудничестве по вопросам охраны труда;
6. иные полномочия в области охраны труда [12].

Важным этапом в реализации государственной политики в области охраны труда было принятие Трудового кодекса Республики Беларусь, вступившего в силу с 1 января 2000 года.

Нормами Кодекса установлены права, обязанности и ответственность в области охраны труда нанимателей и работников, а также функции государства в этой сфере. Установлено, в частности, что для реализации права работника на охрану труда государство обеспечивает организацию охраны труда, осуществление государственного надзора и контроля за соблюдением законодательства по охране труда, ответственность за нарушение законодательства [8].

Большое значение имеют нормы трудового кодекса, касающиеся служб охраны труда нанимателей, и, в частности, базовых нормативов этих служб, а также утвержденные Министерством труда и социальной защиты Республики Беларусь нормативы численности специалистов по охране труда на предприятиях. Трудовым кодексом введена также норма, устанавливающая право работника на отказ от выполнения порученной работы в случае возникновения непосредственной опасности для жизни и здоровья его и окружающих до устранения этой опасности, а также при непредоставлении ему средств индивидуальной защиты, непосредственно обеспечивающих безопасность труда [8].

Сегодня мы можем констатировать, что в Республике Беларусь сформирована национальная нормативная правовая база, регулирующая данную сферу социально-трудовых отношений. Установлены и реализуются организационные схемы управления в этой области.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

Организована переподготовка лиц, имеющих высшее образование, с присвоением квалификации «специалист по охране труда» в:

1. Республиканском институте повышения квалификации и переподготовки работников Министерства труда и социальной защиты Республики Беларусь;
2. Институте повышения квалификации и переподготовки руководителей и специалистов промышленности «Кадры индустрии».

Приведенные учебные заведения, являются примерами из обширного перечня мест, где можно пройти такую переподготовку.

Создана система проверки знаний по вопросам охраны труда на всех уровнях управления.

Налажена работа местных исполнительных и распорядительных органов по реализации государственной политики по охране труда на территориальном уровне.

Во многих регионах внедрены системы управления охраной труда, реализуются целевые программы по улучшению условий и охраны труда, функционируют межведомственные комиссии по охране труда, проводятся смотры-конкурсы, Дни охраны труда, созданы учебные центры по обучению, повышению квалификации и проверке знаний работников по этим вопросам.

Вопросы состояния охраны труда, производственного травматизма, соблюдения законодательства о труде находятся на постоянном контроле Правительства, систематически, глубоко и всесторонне рассматриваются на его заседаниях, принимаются управленческие решения по совершенствованию работы, повышению безопасности и улучшению условий труда на производстве.

# **Список использованных источников**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

Дипломная работа

1. Фриман, Адам Pro ASP.NET Core MVC 2 / Адам Фриман. - Лондон: Apress. 2017. – 1013 с
2. Натан, Адам WPF 4 Подробное руководство / Адам Натан. - Санкт-Петербург: Символ-Плюс. 2019. – 878 с
3. Рихтер, Джефри CLR via C#: Программирование на платформе Microsoft .NET Framework 4.5 на языке C# / Джефри Рихтер. - Санкт-Петербург: Питер, 2013. – 893 с
4. Обзор Windows Forms.: Официальная документация. – Электрон. данные. – Режим доступа: https://docs.microsoft.com/ru- ru/dotnet/framework/winforms/windows-forms-overview. – Дата доступа: 05.04.2022.
5. Общие сведения о WPF: Официальная документация. – Электрон. данные. – Режим доступа: https://docs.microsoft.com/ru- ru/dotnet/framework/wpf/introduction-to-wpf. – Дата доступа: 05.04.2022.
6. Кожевников, Е.А. Расчет экономической эффективности разработки программных продуктов: метод. указания по подготовке организац.-экон. раздела дипломных работ для студентов специальности 1-40 01 02 «Информационные системы и технологии (по направлениям)» днев. формы обучения / Е. А. Кожевников, Н. В. Ермалинская. – Гомель: ГГТУ им. П. О. Сухого, 2012. – 64 с.
7. Конституция Республики Беларусь 1994 г. (с изменениями и дополнениями, принятыми на республиканских референдумах 24 ноября 1996 г., [17 октября 2004 г.](https://etalonline.by/webnpa/text.asp?RN=P00400002) и [27 февраля 2022г.](https://etalonline.by/webnpa/text.asp?RN=P02200001)) // Национальный центр правовой информации Республики Беларусь – Режим доступа: https://etalonline.by/document/?regnum=V19402875 – Дата доступа: 05.05.2022.
8. Трудовой кодекс Республики Беларусь от 26 июля 1999 г.№ 296-З (по состоянию на 29.05.2021) // Национальный центр правовой информации Республики Беларусь – Режим доступа: https://etalonline.by/document/?regnum=HK9900296– Дата доступа: 05.05.2022.
9. Президент Республики Беларусь – Режим доступа: https://president.gov.by/ru/belarus/social/zashhita-naselenija/ohrana-truda – Дата доступа: 06.05.2022.
10. «О мерах по укреплению общественной безопасности и дисциплины» Директива № 1 от 11 марта 2004 г. (в редакции Указа № 420 от 12 октября 2015 г.) – Режим доступа: https://president.gov.by/ru/documents/direktiva-1-ot-11-marta-2004-g-1397 – Дата доступа: 06.05.2022.
11. Закон Республики Беларусь 23 июня 2008 г. № 356-З Об охране труда (в редакции Закона от 18 декабря 2019 г. № 274-З) – Режим доступа: http://myadel.gov.by/uploads/files/11-06-2020-2.pdf – Дата доступа: 06.05.2022.
12. Лазаренков А.М., Фасевич Ю.Н. Курс лекций по дисциплине «Охрана труда»: электронное учебное пособие / А.М. Лазаренков, Ю.Н. Фасевич – Минск: БНТУ, 2019. – Режим доступа: https://rep.bntu.by/bitstream/handle/data/53479/Kurs\_lekcij\_po\_discipline\_Ohrana\_truda.pdf?sequence=1&isAllowed=y – Дата доступа: 06.05.2022.

Изм.

Лист

№ докум.

Подпись

Дата

Лист

1

Дипломная работа

# ПРИЛОЖЕНИЕ А

# ПРИЛОЖЕНИЕ Б

(справочное)

**Каталог функций программного обеспечения**

Таблица Б.1

|  |  |  |  |
| --- | --- | --- | --- |
| **Код функций** | **Наименование (содержание) функций** | **Объем функции строк исходного кода (LOC)** | |
| **Покаталогу ()** | **уточненный ()** |
| 101 | Организация ввода информации | 130 | 30 |
| 107 | Организация ввода-вывода информации в интерактивном режиме | 280 | 50 |
| 203 | Обработка наборов и записей базы данных | 2370 | 600 |
| 206 | Манипулирование данными | 7860 | 1100 |
| 207 | Организация поиска и поиск в базе данных | 4720 | 1400 |
| 403 | Формирование служебных таблиц | 1140 | 370 |
| 506 | Обработка ошибочных сбойных ситуаций | 1540 | 350 |
| 707 | Графический вывод результата | 420 | 100 |
|  | Итого*:* | 18460 | 4000 |

# ПРИЛОЖЕНИЕ В

(справочное)

**Расчет общей трудоемкости разработки ПО**

Таблица В.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Показатели** | **Стадии разработки** | | | | | **Итого** |
| **ТЗ** | **ЭП** | **ТП** | **РП** | **ВН** |  |
| Общий объем ПО (), кол-во строк LOC | – | – | – | – | – | 18460 |
| Общий уточненный объем ПО (), кол-во строк LOC | – | – | – | – | – | 4000 |
| Категория сложности разрабатываемого ПО | – | – | – | – | – | 3 |
| Нормативная трудоемкость разработки ПО (), чел./дн. | – | – | – | – | – | 162 |
| Коэффициент повышения сложности ПО () | 1,13 | 1,13 | 1,13 | 1,13 | 1,13 | – |
| Коэффициент, учитывающий новизну ПО () | 0,63 | 0,63 | 0,63 | 0,63 | 0,63 | – |
| Коэффициент, учитывающий степень использования стандартных модулей () | – | – | – | 0,9 | – | 0,9 |
| Коэффициент, учитывающий средства разработки ПО() | 1,2 | 1,2 | 1,2 | 1,2 | 1,2 | – |
| Коэффициенты удельных весов трудоемкости стадий разработки ПО (, , , , ) | 0,08 | 0,19 | 0,28 | 0,34 | 0,11 | 1,0 |
| Распределение нормативной трудоемкости ПО по стадиям, чел./дн. | 13 | 31 | 45 | 55 | 18 | 162 |
| Распределение скорректированной (с учетом , , , ) трудоемкости ПО по стадиям, чел./дн. | 11,07 | 26,29 | 38,75 | 42,35 | 15,22 | 133,69 |
| Общая трудоемкость разработки ПО (), чел./дн. | – | – | – | – | – | 134 |

# ПРИЛОЖЕНИЕ Г

(справочное)

**Производственные затраты на разработку ПО**

Таблица Г.1

|  |  |  |
| --- | --- | --- |
| **Параметр** | **Единица измерения** | **Значение** |
| Базовая ставка | руб. | 457 |
| Разряд разработчика | – | 1 |
| Тарифный коэффициент 7-го разряда | – | 1,47 |
| Коэффициент | – | 1 |
| Норматив отчислений на доп. зарплату разработчиков () | % | 15 |
| Численность обслуживающего персонала | чел. |  |
| Разряд обслуживающего персонала | – |  |
| Базовая ставка 5-го разряда | – |  |
| Коэффициент | – |  |
| Средняя годовая ставка арендных платежей () | руб./м2 | 16,9 |
| Площадь помещения () | м2 | 10 |
| Количество ПЭВМ () | шт. | 1 |
| Затраты на приобретение единицы ПЭВМ | руб. | 2590 |
| Стоимость одного кВт-часа электроэнергии () | руб. | 0,29567 |
| Затраты на технологию () | руб. | – |

# ПРИЛОЖЕНИЕ Д

(справочное)

**Расчет суммарных затрат на разработку ПО**

Таблица Д.1

|  |  |
| --- | --- |
| **Статья затрат** | **Итого** |
|
| Затраты на оплату труда разработчиков (), руб. | 9135,22 |
| Затраты машинного времени (), руб. | 4264,23 |
| Стоимость машино-часа, руб/ч | 9,26 |
| Сумма годовых амортизационных отчислений, руб. | 580,16 |
| Действительный годовой фонд времени работы ПЭВМ, дн. | 1672,8 |
| Затраты на текущий и профилактический ремонт, руб. | 203,06 |
| Прочие затраты, связанные с эксплуатацией ЭВМ, руб. | 145,04 |
| Машинное время ЭВМ, ч. | 460,5 |
| Затраты на изготовление эталонного экземпляра (), руб. | 0 |
| Затраты на технологию (), руб. | 0 |
| Общепроизводственные затраты () | 456,76 |
| Непроизводственные (коммерческие) затраты () | 456,76 |
| Затраты на материалы (), руб. | 29,01 |
| Суммарные затраты на разработку ПО () | 19319,42 |