**Министерство науки и высшего образования Российской Федерации**

**автономное образовательное учреждение высшего образования**

**«Санкт-Петербургский государственный политехнический университет Петра Великого»**

**(ФГАОУ ВО «СПбПУ»)**

**Институт среднего профессионального образования**

**КУРСОВАЯ РАБОТА**

Информационные системы и программирование

специальность

отделение Очное

По дисциплине: Технология разработки программного обеспечения

Тема: Страховая компания

Исполнитель: студент(ка) III курса группы 32919/11

Буров Сергей Витальевич

(фамилия, имя, отчество)

Руководитель \_\_\_преподаватель ИСПО\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ученная степень, звание)

Челищева Лилия Дмитриевна

\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

Курсовая работа допущен к защите

«11» марта 2024

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Защитил(а) курсовую работу с оценкой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«11» марта 2024

2024

**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

Институт среднего специального образования

УТВЕРЖДЕНО   
предметной комиссией  
«Информационные системы»  
  
ПредседательПЦК  
\_\_\_\_\_\_\_\_\_\_\_\_ Андреев В.А.

**ЗАДАНИЕна курсовое проектирование**

по модулю ПМ.02 «Осуществление интеграции программных модулей»

Специальность 09.02.07 «Информационные системы и программирование»

Студенту группы 32919/11 Буров Сергей Витальевич

Тема курсового проекта: «Страховая компания»

**Индивидуальное задание**

*\_\_Проанализировать предметную область, спроектировать систему (диаграммы вариантов использования, последовательности, активности, состояний), разработать систему, протестировать с помощью юнит-тестов и тест-кейсов, составить  
руководство оператора.*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Дата выдачи задания 04.09.2023 Зав. отделением \_\_\_\_\_\_\_\_\_Сухорукова О.А.

Срок окончания 11.04.2024 Преподаватель \_\_\_\_\_\_\_\_\_\_ Челищева Л. Д.

Студент \_\_\_\_\_\_\_\_\_\_\_ Буров С.В.

Содержание

[Введение 4](#_Toc163594830)

[1Общая часть 5](#_Toc163594831)

[1.1 Анализ предметной области 5](#_Toc163594832)

[1.1.1 Анализ проблемы 5](#_Toc163594833)

[1.1.2 Разработка карты навигации к информационной системе 10](#_Toc163594834)

[1.1.3 Разработка пользовательских историй 11](#_Toc163594835)

[1.1.4 Требования 12](#_Toc163594836)

[1.1.5Анализ методов решения 14](#_Toc163594838)

[1.1.6Обзор средств программирования 15](#_Toc163594839)

[1.1.6.1 Java 15](#_Toc163594840)

[1.1.6.2 Python 16](#_Toc163594841)

[1.1.6.3 С# 17](#_Toc163594842)

[2 Специальная часть 18](#_Toc163594843)

[2.1 Проектирование 18](#_Toc163594844)

[2.2 Разработка 21](#_Toc163594845)

[2.2.1 Разработка базы данных 21](#_Toc163594846)

[2.2.2 Разработка приложения 23](#_Toc163594847)

[2.3 Тестирование 24](#_Toc163594848)

[Заключение 25](#_Toc163594849)

[Список использованных источников 26](#_Toc163594850)

[Приложение А (обязательное) sql скрипт 27](#_Toc163594851)

[Приложение Б (обязательное) код программы 29](#_Toc163594852)

[Приложение В (обязательное) test case 42](#_Toc163594853)

[Приложение Г (справочное)интерфейс 45](#_Toc163594854)

# **ВВЕДЕНИЕ**

В современном мире страховые компании играют важную роль в обеспечении финансовой защиты для клиентов, предлагая широкий спектр услуг от страхования жизни до страхования имущества и ответственности. Однако, с увеличением объема операций и сложности клиентских запросов, страховые компании сталкиваются с вызовом оптимизации своих бизнес-процессов и повышения качества обслуживания.

В этом контексте разработка информационной системы для страховой компании становится критически важной задачей. Цель данной работы заключается в создании информационной системы, специально адаптированной для потребностей страховой компании, с акцентом на повышении эффективности ее деятельности и улучшении качества обслуживания клиентов.

Анализ потребностей страховой компании в области информационных технологий, проектирование структуры и функционала информационной системы, разработка приложения с основными функциями и последующее тестирование на соответствие заявленным требованиям — эти этапы представляют собой ключевые шаги в процессе разработки системы, направленной на оптимизацию бизнес-процессов и улучшение качества обслуживания клиентов страховой компании.

Разработка информационной системы для страховой компании представляет собой важный этап в процессе совершенствования бизнес-процессов и повышения конкурентоспособности компании в современном рыночном окружении.

**1. Общая часть**

**1.1 Анализ предметной области**

**1.1.1 Анализ проблемы**

В результате анализа рассматриваемой предметной области был выявлен список заинтересованных лиц:

* Директор страховой компании;
* Менеджер по обслуживаю клиентов.

После анализа заинтересованных лиц было произведено их анкетирование (результаты представлены в таблицах 1-2) для выявления их потребностей, а также общей актуальности проблемы.

Результат анкетирования страхового агента

**Анкета для директора страховой компании:**  
1. Имя: Владимир  
2. Ваша должность: Директор  
3. Какова основная цель вашей работы?

Основной целью моей работы являются обеспечение стабильности и успешного развития компании, предоставление клиентам надежной страховой защиты и максимизации финансовых результатов.

4. Какие основные проблемы вы видите в работе страховой компании?

Одной из основных проблем, с которой мы сталкиваемся, является неэффективное управление и неэффективное внедрение цифровых технологий, затрудняющее обработку данных клиентов.

5. Какие шаги предпринимаются для улучшения цифровой трансформации и оптимизации обработки данных клиентов в нашей страховой компании?

Мы будем внедрять современные информационные системы и автоматизировать процессы, так же мы проводим обучение персонала для эффективного использования новых технологий.

6. Какая информация вам необходима для эффективной работы в страховой компании?

Для эффективной работы в страховой компании мне необходима информация о условиях страхования, данных об услугах, а также информация всех филиалов.

**Анкета для менеджера по обсуживаю клиентов страховой компании:**  
1. Имя: Мария  
2. Ваша должность: Менеджер по обслуживанию клиентов  
3. Как часто клиенты пользуются услугами страховой компании?

Наши клиенты пользуются услугами страховой компании постоянно.

4. С какими типами страховых услуг к вам чаще всего обращаются клиенты?

Клиенты чаще всего обращаются за страховыми услугами на автострахование, медицинское страхование и страхование недвижимости.

5. Как программа управления Страховой компании будет влиять на вашу работу и удобство поиска необходимых материалов для удовлетворения потребностей клиентов?

Программа управления страховой компании будет значительно упрощать доступ к необходимой информации, улучшая ее эффективность в обработке данных клиентов и предоставлении необходимых услуг. Все это в совокупности будет содействовать быстрому реагированию на запросы клиентов и повышению удовлетворенности.

6. Какие функции или возможности вы считаете важными в создании программы страховой компании?

Важными функциями в программе управления страховой компании будут по моему мнению:

 Эффективная система урегулирования убытков: Быстрое и справедливое урегулирование убытков для клиентов и аналитику данных: Понимание рыночных трендов и потребностей клиентов через анализ данных.

7. Какая информация вам необходима для эффективного использования услуг страховой компании?

Для эффективного использования услуг страховой компании мне нужна полная информация о страховых продуктах, их условиях и тарифах.

**1.1.2 Разработка карты навигации к информационной системе**

Для обеспечения удобства использования и эффективной навигации пользователей к информационной системе была создана карта навигации (рисунок 1).

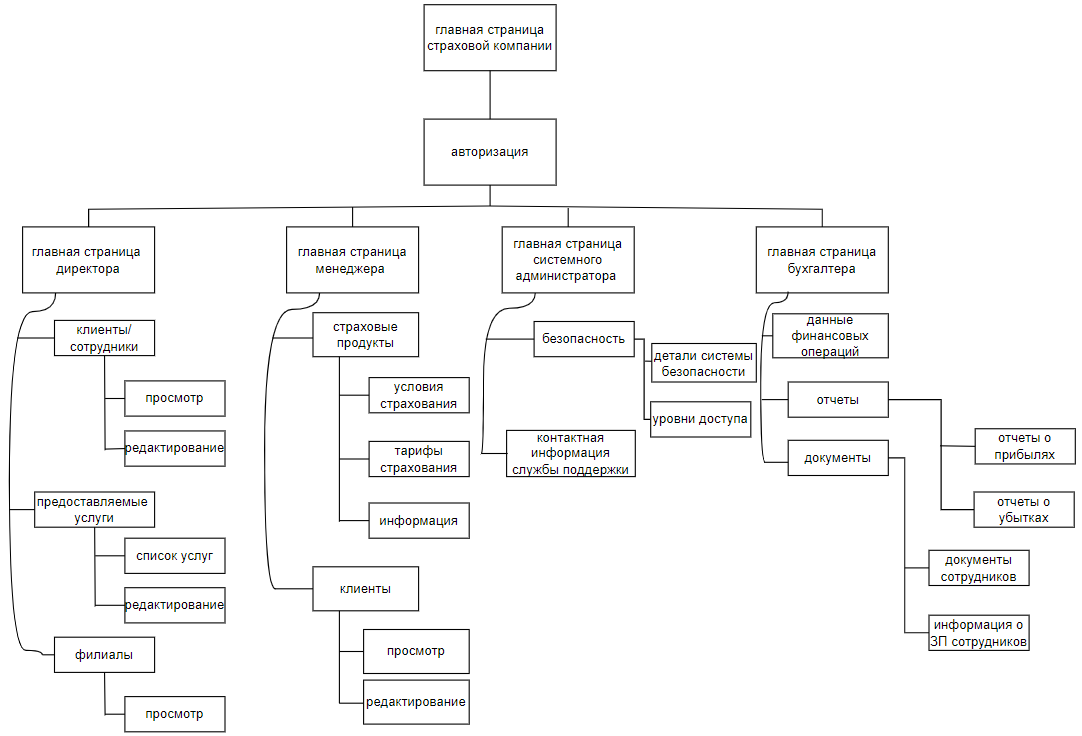


Рисунок 1 – карта навигации

**1.1.3 Разработка пользовательских историй**

Был разработан список пользовательских историй от различных типов пользователей:

Я как директор страховой компании, хочу получать сводные финансовые отчеты для того, чтобы принимать обоснованные стратегические решения и повысить финансовую стабильность

Я как директор страховой компании, хочу анализировать финансовые показатели для того, чтобы информированные решения по стратегии развития компании

Я как директор страховой компании, хочу получать ежедневные отчеты о выполнении ключевых показателей для того, чтобы мониторить производительность компании и принимать оперативные меры по улучшению результатов

Я как директор страховой компании, хочу иметь доступ к базе данных сотрудников и клиентов для того, чтобы анализировать динамику клиентской базы, а также эффективно управлять персоналом.

Я как директор страховой компании, хочу иметь доступ к информации всех филиалов для того, чтобы обеспечивать централизованное управление и контроль за работой всех филиалов компании, а так же анализировать и сравнивать производственные показатели.

Я как директор страховой компании, хочу иметь доступ к данным об услугах для того, чтобы проводить анализ ассортимента предлагаемых услуг компании, оценивать их популярность и востребованность на рынке.

Я как менеджер по обслуживанию клиентов, хочу быстро находить информацию о страховых услугах для того, чтобы обеспечивать клиентов актуальной и точной информацией для принятия решений

Я как менеджер по обслуживанию клиентов, хочу иметь доступ к базе данных клиентов для того, чтобы быстро находить информацию о клиентах и их страховых полисах.

Я как менеджер по обслуживанию клиентов, хочу иметь возможность создавать индивидуальные предложения для клиентов для того, чтобы предлагать клиентам персонализированные страховые пакеты.

**1.1.4 Требования**

**1.1.4.1 Функциональные требования**

* Система Регистрация и аутентификация пользователей.
* Просмотр и редактирование данных клиентов,
* Просмотр и редактирование сводных финансовых отчетов
* Анализ финансовых показателей для развития компании
* Просмотр и редактирование данных сотрудников и клиентов для четкого анализа динамики клиентской базы
* Просмотр и редактирование информации всех филиалов
* Просмотр и редактирование данных об услугах
* Просмотр и редактирование прав доступа пользователей к информационным ресурсам компании
* Просмотр и редактирование информации данных службы поддержки
* Бухгалтер должен иметь возможность управлять банковскими счетами компании, включая проведение платежей, контроль за поступлениями и списаниями.

**1.1.4.2 Нефункциональные требования**

* Для нормального функционирования данной информационной системы необходимы следующие технические средства:
* Система должна работать на MacOS, не ниже версии MacOS High Siearra
* Время входа пользователя в систему не должна превышать 2-х минут (вход в систему, аунтефикация, подрузка и загрузка всей системы)
* Система должна обеспечивать быстрый отклик на запросы пользователей, не должно превышать 1 секунды
* Система должна поддерживать определенное количество одновременных пользователей, а именно не менее 200 пользователей одновременно
* Система должна соблюдать требованиям безопасности и конфиденциальности пользователей в соответствии со стандартами ISO/IEC 27001: международный стандарт, определяющий требования к системам управления информационной безопасностью и ISO/IEC 27018: международный стандарт, устанавливающий рекомендации по защите личных данных в облачных услугах, а так же стандартам PCI, DSS - для обработки платежных данных и HIPPA для медицинской информации
* Система должна поддерживать увеличение объема данных в 3 раза без значительного снижения производительности
* Система должна обеспечивать обнаружение и предотвращение 95% известных видов вредоносных атак

**Анализ методов решения**

Реляционная база данных (РБД), размещенная на сервере, позволяет нескольким компьютерам (клиентам) одновременно подключаться к ней и взаимодействовать с данными. Это обеспечивает гибкость и удобство работы с информацией, так как различные пользователи могут одновременно обновлять данные и получать доступ к актуальной информации.

Преимущества реляционной базы данных включают:

- Совместная работа: возможность совместного доступа нескольких пользователей к данным упрощает обмен информацией и совместное решение задач.

- Управление данными: организация данных в виде таблиц обеспечивает понятную и удобную структуру для работы.

- Выполнение сложных запросов: поддержка языка SQL позволяет выполнять разнообразные и сложные запросы к данным.

Однако, реляционная база данных имеет и некоторые минусы:

- Зависимость от сервера: использование сервера для хранения и обработки данных может создать зависимость от его работоспособности и доступности.

- Возможные проблемы с производительностью: при большом количестве подключений или обработке больших объемов данных реляционная база данных может столкнуться с проблемами производительности.

**1.1.6 Обзор средств программирования**

С точки зрения возможности использования для решения поставленной задачи, наиболее популярными являются языки программирования Java, Python и C#. Для данного курсового проекта был выбран язык C#, который отлично подходит для разработки приложений с использованием платформы .NET. Язык C# предлагает широкий спектр возможностей для создания высокопроизводительных и масштабируемых приложений.

Для хранения данных выбрана СУБД Microsoft SQL Server. Это мощная и надежная система управления базами данных, обладающая широким набором функций и возможностей для работы с данными. Выбор данной СУБД обеспечивает эффективное хранение и обработку данных, что является важным аспектом для успешной реализации курсового проекта.

**1.1.6.1 Python**

Python — это высокоуровневый язык программирования общего назначения, который применяется в различных областях, включая разработку веб-приложений. Язык ориентирован на удобство разработчика и читаемость кода, что делает его популярным выбором для множества проектов.

Python поддерживает несколько парадигм программирования: структурную, объектно-ориентированную, функциональную, императивную и аспектно-ориентированную. Он обладает динамической типизацией, автоматическим управлением памятью, полной интроспекцией, механизмом обработки исключений, поддержкой многопоточных вычислений и удобными высокоуровневыми структурами данных.

Программный код на Python обычно организовывается в функции и классы, которые могут быть объединены в модули, а модули в свою очередь в пакеты. Python часто используется как интерпретируемый язык, но также может быть скомпилирован в байт-код Java и MSIL (в рамках платформы .NET).

Хотя Python также подходит для веб-разработки, в этой области он не всегда является оптимальным выбором с экономической точки зрения. Например, в веб-разработке Ruby часто оказывается эффективнее и экономически более выгодной, а компилируемые языки, такие как Go, Elixir и Java, могут обеспечить значительно лучшую производительность.

Однако, в областях анализа данных и машинного обучения Python является неоспоримым лидером благодаря богатому набору библиотек и инструментов, предназначенных для этих задач.

**1.1.6.2 Java**

Java — это язык программирования общего назначения, который относится к объектно-ориентированным языкам с сильной типизацией. Один из основных принципов, заложенных создателями Java, - это принцип WORA (write once, run anywhere), или "пиши один раз, запускай везде". Это означает, что приложение, написанное на Java, может быть запущено на любой платформе, где установлена среда исполнения Java (JRE, Java Runtime Environment).

Для достижения этой цели Java код компилируется в байт-код, который исполняется виртуальной машиной Java (JVM). JVM является частью среды исполнения Java и не зависит от платформы, что обеспечивает переносимость приложений на различные операционные системы.

Java широко используется для разработки веб-приложений. Популярные фреймворки, такие как Spring, Struts, JSP, используются для создания разнообразных веб-проектов, начиная от электронной коммерции до крупных порталов и правительственных ресурсов.

Также Java активно применяется в мобильной разработке для создания приложений под управлением операционной системы Android. Множество приложений для мобильных устройств разрабатывается на этом языке, включая клиентские приложения и приложения для образовательных целей.

**1.1.6.3 C#**

C# — это объектно-ориентированный язык программирования, разработанный командой инженеров Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота. Он относится к семье языков, вдохновленных языком C, и имеет синтаксис, схожий с Java и C++.

Этот язык программирования обладает рядом характеристик, делающих его мощным инструментом разработки:

* Статическая типизация;
* Полиморфизм;
* Перегрузка операторов;
* Поддержка делегатов, атрибутов, событий, обобщенных типов и анонимных функций.

C# наследует множество особенностей от других популярных языков программирования, таких как Delphi, Smalltalk и Java, избегая при этом некоторых их "проблемных" аспектов.

# Этот язык программирования предоставляет широкие возможности для разработки различных типов приложений, включая веб-приложения, игровые программы, мобильные приложения для платформ Android и iOS, а также программы для операционной системы Windows. Благодаря обширному набору инструментов и средств, C# является мощным и универсальным языком программирования.

# В C# реализована автоматическая сборка мусора, которая позволяет эффективно управлять памятью, освобождая ее от неиспользуемых объектов и ресурсов.

# Также язык обеспечивает удобные средства для обработки исключений, что позволяет легко выявлять и корректно обрабатывать ошибки в программном коде. Однако важно использовать эти возможности осмотрительно, чтобы избежать появления ошибок и непредвиденного поведения программы.

**2 Специальная часть**

**2.1 Проектирование**

На рисунке 2 показана диаграмма вариантов использования.

Изображение выглядит как текст, диаграмма, линия, Шрифт

Автоматически созданное описание

Рисунок 2 – Диаграмма вариантов использования

На рисунке 3 показана диаграмма последовательности.

Изображение выглядит как текст, диаграмма, Параллельный, линия

Автоматически созданное описание

Рисунок 3 – Диаграмма последовательности

На Рисунке 4 приведена диаграмма активности.

Изображение выглядит как диаграмма, текст, Технический чертеж, чек

Автоматически созданное описание

Рисунок 4 – Диаграмма активности

**2.2 Разработка**

**2.2.1 Разработка базы данных**

На основе проанализированных данных и была создана концептуальная модель, показанная на рисунке 5

**Изображение выглядит как диаграмма, снимок экрана, линия

Автоматически созданное описание**

Рисунок 5 – Концептуальная модель

Затем была разработана логическая модель и физическая модель, показанные на рисунках 6 и 7.

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Рисунок 6 – логическая модель

**Изображение выглядит как текст, снимок экрана, диаграмма, План

Автоматически созданное описание**

Рисунок 7 – Физическая модель

**2.2.2 Разработка приложения**

Перед разработкой приложения были сделаны примеры интерфейса некоторых окон.

Изображение выглядит как снимок экрана, текст, дизайн

Автоматически созданное описание

Рисунок 8 **—**Пример интерфейса окна авторизации

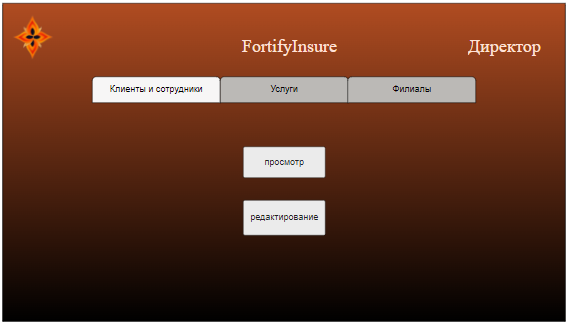


Рисунок 9 **—**Пример интерфейса окна директора

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок 10 — Пример интерфейса окна директора с выбором другой кнопки

Интерфейс приложения предоставлен в Приложении Г. Это приложение успешно интегрировано с ранее разработанной базой данных. Подробный листинг кода приложения представлен в Приложении Б.

**2.3 Тестирование**

Для систематического и всестороннего тестирования приложения были разработаны тест-кейсы (см. Приложение В), каждый из которых содержит следующие основные аспекты:

* Название теста: короткое и информативное наименование, отражающее цель тестирования;
* Шаги тестирования: последовательность действий, необходимых для проведения теста, включая взаимодействие с приложением;
* Тестовые данные: входные данные или предусловия, которые должны быть настроены перед началом тестирования;
* Ожидаемый результат: описание ожидаемого поведения или результата в случае успешного завершения теста;
* Фактический результат: запись реального результата, полученного в ходе проведения теста;
* Статус: указание на успешное или неудачное завершение теста.

Каждый тест-кейс разработан с особой тщательностью с целью охвата различных аспектов функциональности и обеспечения надежности приложения.

Разнообразие тест-кейсов гарантирует полное покрытие возможных сценариев использования и выявление потенциальных проблем до выпуска приложения.

**ЗАКЛЮЧЕНИЕ**

В завершении данного курсового проекта следует подчеркнуть необходимость создания передовых информационных решений для эффективного управления бизнес-процессами в страховых компаниях. Разработанный прототип выступает важным доказательством потенциала автоматизации и оптимизации операций страховых компаний, обеспечивая более высокий уровень обслуживания клиентов и эффективное управление ресурсами.

Процесс создания прототипа позволил выявить ключевые требования и особенности, необходимые для успешной реализации информационной системы в этой сфере. Однако важно понимать, что созданный прототип — лишь первый шаг на пути к разработке полноценной информационной системы, способной в полной мере удовлетворить потребности как страховой компании, так и ее клиентов.

Дальнейшее развитие и совершенствование проекта требует дополнительного глубокого анализа потребностей пользователей, внедрения инновационных функциональных возможностей и активного взаимодействия с сотрудниками страховой компании и клиентами. Этот процесс является краеугольным камнем успешной адаптации информационной системы к динамично меняющимся требованиям рынка и обеспечению ее высокой конкурентоспособности.

Так же, следует подчеркнуть, что разработка информационной системы для страховой компании является инвестицией в будущее, способствующей повышению ее операционной эффективности и конкурентоспособности. Однако для достижения максимального потенциала этой инвестиции необходимо постоянное внимание к изменениям в отрасли, быстрая адаптация к новым технологиям и потребностям клиентов, а также систематическое развитие и совершенствование информационной системы в соответствии с этими изменениями.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Программирование на языке Java / [Электронный ресурс] // инфинITи : [сайт]. — URL: http://226school.ru/programmirovanie-na-yazyike-java.html (дата обращения: 09.04.2024).

Немного о Python / [Электронный ресурс] // Дисциплина "Язык программирования Python" : [сайт]. — URL: https://python-kipu.mya5.ru (дата обращения: 09.04.2024).

Работа с SQL Server в C# с помощью скриптинга / [Электронный ресурс] // : [сайт]. — URL: https://habr.com/ru/companies/otus/articles/711866/ (дата обращения: 09.04.2024).

Подключение в базу данных в Visual Studio URL: https://learn.microsoft.com/ru-ru/visualstudio/data-tools/add-new-connections?view=vs-2022/ (дата обращения: 07.04.2024).

Книги по C# для начинающих и практикующих специалистов URL: https://habr.com/ru/companies/ru\_mts/articles/757962/

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Sql скрипт**

**Таблицы**

SELECT TOP (1000) [ИД]

,[Имя]

,[Фамилия]

,[Должность]

,[Адрес]

,[ДатаРождения]

FROM [insuranceCompany].[dbo].[Клиенты]

SELECT TOP (1000) [ИД]

,[Имя]

,[Фамилия]

,[Должность]

,[Адрес]

,[ДатаРождения]

FROM [insuranceCompany].[dbo].[Сотрудники]

SELECT TOP (1000) [ИД]

,[Название]

,[Описание]

,[Тип]

,[Стоимость]

FROM [insuranceCompany].[dbo].[Услуги]

SELECT TOP (1000) [ИД]

,[Название]

,[Адрес]

,[Телефон]

,[ЭлектроннаяПочта]

FROM [insuranceCompany].[dbo].[Филиалы]

**Заполнение таблиц**

-- Добавление данных в таблицу "Клиенты"

INSERT INTO Клиенты (ИД, Имя, Фамилия, Должность, Адрес, ДатаРождения)

VALUES

(1, 'Иван', 'Иванов', 'Менеджер', 'ул. Пушкина, д.1, кв. 5', '1985-07-15'),

(2, 'Мария', 'Петрова', 'Бухгалтер', 'ул. Ленина, д.10, кв. 15', '1978-12-28'),

(3, 'Алексей', 'Сидоров', 'системный администратор', 'пр. Гагарина, д.20, кв. 3', '1990-03-05');

-- Добавление данных в таблицу "Услуги"

INSERT INTO Услуги (ИД, Название, Описание, Тип, Стоимость)

VALUES

(1, 'Автострахование', 'Страхование автомобилей от различных рисков', 'Автострахование', 5000.00),

(2, 'Медицинское страхование', 'Страхование здоровья и медицинских расходов', 'Медицинское страхование', 10000.00),

(3, 'Страхование недвижимости', 'Страхование жилой и коммерческой недвижимости', 'Недвижимость', 8000.00);

-- Добавление данных в таблицу "Филиалы"

INSERT INTO Филиалы (ИД, Название, Адрес, Телефон, ЭлектроннаяПочта)

VALUES

(1, 'Центральный офис', 'ул. Ленина, д.1', '+7 (123) 456-7890', 'info@example.com'),

(2, 'Филиал №1', 'пр. Гагарина, д.5', '+7 (234) 567-8901', 'filial1@example.com'),

(3, 'Филиал №2', 'ул. Победы, д.15', '+7 (345) 678-9012', 'filial2@example.com');

-- Добавление данных в таблицу "Сотрудники"

INSERT INTO Сотрудники (ИД, Имя, Фамилия, Должность, Адрес, ДатаРождения)

VALUES

(1, 'Петр', 'Смирнов', 'Директор', 'ул. Ленина, д.1, кв. 10', '1970-05-20'),

(2, 'Ольга', 'Иванова', 'Менеджер', 'ул. Гагарина, д.5, кв. 7', '1988-10-12'),

(3, 'Александр', 'Козлов', 'Специалист по страхованию', 'пр. Победы, д.15, кв. 22', '1983-04-30');

-- Добавление новой записи

INSERT INTO Клиенты (Имя, Фамилия, Должность, Адрес, ДатаРождения)

VALUES ('Новое имя', 'Новая фамилия', 'Новая должность', 'Новый адрес', '1990-01-01');

-- Изменение существующей записи

UPDATE Клиенты

SET Должность = 'Новая должность'

WHERE ИД = 1; -- ИД записи, которую вы хотите изменить

-- Удаление записи

DELETE FROM Клиенты

WHERE ИД = 1; -- ИД записи, которую вы хотите удалить

-- Добавление новой записи

INSERT INTO Сотрудники (Имя, Фамилия, Должность, Адрес, ДатаРождения)

VALUES ('Новое имя', 'Новая фамилия', 'Новая должность', 'Новый адрес', '1990-01-01');

-- Изменение существующей записи

UPDATE Сотрудники

SET Должность = 'Новая должность'

WHERE ИД = 1; -- ИД записи, которую вы хотите изменить

-- Удаление записи

DELETE FROM Сотрудники

WHERE ИД = 1; -- ИД записи, которую вы хотите удалить

-- Добавление новой записи

INSERT INTO Услуги (Название, Описание, Тип, Стоимость)

VALUES ('Новая услуга', 'Описание новой услуги', 'Тип услуги', 100.00);

-- Изменение существующей записи

UPDATE Услуги

SET Стоимость = 200.00

WHERE ИД = 1; -- ИД записи, которую вы хотите изменить

-- Удаление записи

DELETE FROM Услуги

WHERE ИД = 1; -- ИД записи, которую вы хотите удалить

**ПРИЛОЖЕНИЕ Б**

(обязательное)

**Код программы**

**Б.1 Form1.cs**

using System;

using System.Drawing;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

this.Load += Form1\_Load;

}

private void Form1\_Load(object sender, EventArgs e)

{

// Установка текста лейбла

label1.Text = "FORTIFY INSURE";

// Установка размера и шрифта лейбла

label1.Font = new Font("Arial", 20, FontStyle.Bold); // Настройте параметры размера и шрифта по вашему вкусу

// Установка текста-подсказки и цвета текста в полях ввода

SetPlaceholder(textBox2, "Login");

SetPlaceholder(textBox1, "Password");

// Установка символа для шифрования пароля

textBox1.PasswordChar = '\*';

}

private void SetPlaceholder(TextBox textBox, string placeholder)

{

textBox.Text = placeholder;

textBox.ForeColor = Color.Gray;

textBox.Enter += (s, args) =>

{

if (textBox.Text == placeholder)

{

textBox.Text = "";

textBox.ForeColor = Color.Black;

}

};

textBox.Leave += (s, args) =>

{

if (textBox.Text == "")

{

textBox.Text = placeholder;

textBox.ForeColor = Color.Gray;

}

};

}

private void button1\_Click\_1(object sender, EventArgs e)

{

{

string username = textBox2.Text;

string password = textBox1.Text;

// Проверяем логин и пароль

if (username == "director" && password == "password")

{

// Если логин и пароль верны, открываем главную форму (например, форму директора)

DirectorForm directorForm = new DirectorForm();

directorForm.Show();

this.Hide(); // Скрываем форму входа в систему

}

else

{

// Если логин и/или пароль неверны, выводим сообщение об ошибке

MessageBox.Show("Неверные данные, попробуйте еще раз", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void label1\_Click(object sender, EventArgs e)

{

}

}

}**Б.2 Form2.cs**

using System;

using System.Windows.Forms;

namespace InsuranceCompanyApp

{

public partial class LoginForm : Form

{

public LoginForm()

{

InitializeComponent();

}

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

{

string username = textBox2.Text;

string password = textBox1.Text;

// Проверяем логин и пароль

if (username == "director" && password == "password")

{

// Если логин и пароль верны, открываем главную форму (например, форму директора)

WindowsFormsApp1.DirectorForm directorForm = new WindowsFormsApp1.DirectorForm();

directorForm.Show();

this.Hide(); // Скрываем форму входа в систему

}

else

{

// Если логин и/или пароль неверны, выводим сообщение об ошибке

MessageBox.Show("Неверные данные", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void LoginForm\_Load(object sender, EventArgs e)

{

}

}

}**Б.3 LoginForm.cs**

namespace kurs

{

partial class Form3

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

this.components = new System.ComponentModel.Container();

this.dataGridView1 = new System.Windows.Forms.DataGridView();

this.label1 = new System.Windows.Forms.Label();

this.button3 = new System.Windows.Forms.Button();

this.button1 = new System.Windows.Forms.Button();

this.button2 = new System.Windows.Forms.Button();

this.kursDataSet = new kurs.kursDataSet();

this.contractsBindingSource = new System.Windows.Forms.BindingSource(this.components);

this.contractsTableAdapter = new kurs.kursDataSetTableAdapters.ContractsTableAdapter();

this.contractIDDataGridViewTextBoxColumn = new System.Windows.Forms.DataGridViewTextBoxColumn();

this.signingDateDataGridViewTextBoxColumn = new System.Windows.Forms.DataGridViewTextBoxColumn();

this.agentIDDataGridViewTextBoxColumn = new System.Windows.Forms.DataGridViewTextBoxColumn();

this.clientIDDataGridViewTextBoxColumn = new System.Windows.Forms.DataGridViewTextBoxColumn();

this.insuranceTypeIDDataGridViewTextBoxColumn = new System.Windows.Forms.DataGridViewTextBoxColumn();

this.employeeIDDataGridViewTextBoxColumn = new System.Windows.Forms.DataGridViewTextBoxColumn();

((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.kursDataSet)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.contractsBindingSource)).BeginInit();

this.SuspendLayout();

//

// dataGridView1

//

this.dataGridView1.AutoGenerateColumns = false;

this.dataGridView1.ColumnHeadersHeightSizeMode = System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;

this.dataGridView1.Columns.AddRange(new System.Windows.Forms.DataGridViewColumn[] {

this.contractIDDataGridViewTextBoxColumn,

this.signingDateDataGridViewTextBoxColumn,

this.agentIDDataGridViewTextBoxColumn,

this.clientIDDataGridViewTextBoxColumn,

this.insuranceTypeIDDataGridViewTextBoxColumn,

this.employeeIDDataGridViewTextBoxColumn});

this.dataGridView1.DataSource = this.contractsBindingSource;

this.dataGridView1.Location = new System.Drawing.Point(22, 77);

this.dataGridView1.Name = "dataGridView1";

this.dataGridView1.Size = new System.Drawing.Size(766, 335);

this.dataGridView1.TabIndex = 0;

//

// label1

//

this.label1.AutoSize = true;

this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.label1.Location = new System.Drawing.Point(236, 9);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(295, 55);

this.label1.TabIndex = 1;

this.label1.Text = "ДОГОВОРА";

//

// button3

//

this.button3.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(192)))), ((int)(((byte)(192)))), ((int)(((byte)(255)))));

this.button3.Location = new System.Drawing.Point(643, 19);

this.button3.Name = "button3";

this.button3.Size = new System.Drawing.Size(145, 20);

this.button3.TabIndex = 3;

this.button3.Text = "Назад";

this.button3.UseVisualStyleBackColor = false;

this.button3.Click += new System.EventHandler(this.button3\_Click);

//

// button1

//

this.button1.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(192)))), ((int)(((byte)(192)))), ((int)(((byte)(255)))));

this.button1.Font = new System.Drawing.Font("Microsoft Sans Serif", 14.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.button1.Location = new System.Drawing.Point(22, 435);

this.button1.Name = "button1";

this.button1.Size = new System.Drawing.Size(224, 72);

this.button1.TabIndex = 4;

this.button1.Text = "Создать договор";

this.button1.UseVisualStyleBackColor = false;

this.button1.Click += new System.EventHandler(this.button1\_Click);

//

// button2

//

this.button2.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(192)))), ((int)(((byte)(192)))), ((int)(((byte)(255)))));

this.button2.Font = new System.Drawing.Font("Microsoft Sans Serif", 14.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

this.button2.Location = new System.Drawing.Point(551, 435);

this.button2.Name = "button2";

this.button2.Size = new System.Drawing.Size(224, 72);

this.button2.TabIndex = 5;

this.button2.Text = "Удалить договор";

this.button2.UseVisualStyleBackColor = false;

this.button2.Click += new System.EventHandler(this.button2\_Click);

//

// kursDataSet

//

this.kursDataSet.DataSetName = "kursDataSet";

this.kursDataSet.SchemaSerializationMode = System.Data.SchemaSerializationMode.IncludeSchema;

//

// contractsBindingSource

//

this.contractsBindingSource.DataMember = "Contracts";

this.contractsBindingSource.DataSource = this.kursDataSet;

//

// contractsTableAdapter

//

this.contractsTableAdapter.ClearBeforeFill = true;

//

// contractIDDataGridViewTextBoxColumn

//

this.contractIDDataGridViewTextBoxColumn.DataPropertyName = "ContractID";

this.contractIDDataGridViewTextBoxColumn.HeaderText = "ContractID";

this.contractIDDataGridViewTextBoxColumn.Name = "contractIDDataGridViewTextBoxColumn";

this.contractIDDataGridViewTextBoxColumn.ReadOnly = true;

//

// signingDateDataGridViewTextBoxColumn

//

this.signingDateDataGridViewTextBoxColumn.DataPropertyName = "SigningDate";

this.signingDateDataGridViewTextBoxColumn.HeaderText = "SigningDate";

this.signingDateDataGridViewTextBoxColumn.Name = "signingDateDataGridViewTextBoxColumn";

//

// agentIDDataGridViewTextBoxColumn

//

this.agentIDDataGridViewTextBoxColumn.DataPropertyName = "AgentID";

this.agentIDDataGridViewTextBoxColumn.HeaderText = "AgentID";

this.agentIDDataGridViewTextBoxColumn.Name = "agentIDDataGridViewTextBoxColumn";

//

// clientIDDataGridViewTextBoxColumn

//

this.clientIDDataGridViewTextBoxColumn.DataPropertyName = "ClientID";

this.clientIDDataGridViewTextBoxColumn.HeaderText = "ClientID";

this.clientIDDataGridViewTextBoxColumn.Name = "clientIDDataGridViewTextBoxColumn";

//

// insuranceTypeIDDataGridViewTextBoxColumn

//

this.insuranceTypeIDDataGridViewTextBoxColumn.DataPropertyName = "InsuranceTypeID";

this.insuranceTypeIDDataGridViewTextBoxColumn.HeaderText = "InsuranceTypeID";

this.insuranceTypeIDDataGridViewTextBoxColumn.Name = "insuranceTypeIDDataGridViewTextBoxColumn";

//

// employeeIDDataGridViewTextBoxColumn

//

this.employeeIDDataGridViewTextBoxColumn.DataPropertyName = "EmployeeID";

this.employeeIDDataGridViewTextBoxColumn.HeaderText = "EmployeeID";

this.employeeIDDataGridViewTextBoxColumn.Name = "employeeIDDataGridViewTextBoxColumn";

//

// Form3

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(128)))), ((int)(((byte)(128)))), ((int)(((byte)(255)))));

this.ClientSize = new System.Drawing.Size(800, 519);

this.Controls.Add(this.button2);

this.Controls.Add(this.button1);

this.Controls.Add(this.button3);

this.Controls.Add(this.label1);

this.Controls.Add(this.dataGridView1);

this.Name = "Form3";

this.Text = "Договора";

this.Load += new System.EventHandler(this.Form3\_Load);

((System.ComponentModel.ISupportInitialize)(this.dataGridView1)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.kursDataSet)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.contractsBindingSource)).EndInit();

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.DataGridView dataGridView1;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.Button button3;

private System.Windows.Forms.Button button1;

private System.Windows.Forms.Button button2;

private kursDataSet kursDataSet;

private System.Windows.Forms.BindingSource contractsBindingSource;

private kursDataSetTableAdapters.ContractsTableAdapter contractsTableAdapter;

private System.Windows.Forms.DataGridViewTextBoxColumn contractIDDataGridViewTextBoxColumn;

private System.Windows.Forms.DataGridViewTextBoxColumn signingDateDataGridViewTextBoxColumn;

private System.Windows.Forms.DataGridViewTextBoxColumn agentIDDataGridViewTextBoxColumn;

private System.Windows.Forms.DataGridViewTextBoxColumn clientIDDataGridViewTextBoxColumn;

private System.Windows.Forms.DataGridViewTextBoxColumn insuranceTypeIDDataGridViewTextBoxColumn;

private System.Windows.Forms.DataGridViewTextBoxColumn employeeIDDataGridViewTextBoxColumn;

}

}

**Б.4 Form4.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class DirectorForm : Form

{

public DirectorForm()

{

InitializeComponent();

btnView.Visible = false;

btnEdit.Visible = false;

btnView2.Visible = false;

btnView3.Visible = false;

}

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

{

btnView.Visible = true;

btnEdit.Visible = true;

if (btnView2.Visible)

{

btnView2.Visible = false;

}

if (btnView3.Visible)

{

btnView3.Visible = false;

}

}

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

{

// Действия при нажатии на кнопку "Просмотр"

}

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

{

// Действия при нажатии на кнопку "Редактирование"

}

private void button2\_Click\_1(object sender, EventArgs e)

{

btnView2.Visible = true;

if (btnView.Visible || btnEdit.Visible)

{

btnView.Visible = false;

btnEdit.Visible = false;

}

// Скрываем кнопки "Просмотр филиалов" и "Редактирование филиалов", если они были видимыми

if (btnView3.Visible)

{

btnView3.Visible = false;

}

}

private void btnView2\_Click(object sender, EventArgs e)

{

TalbleFormSERVICES tableForm = new TalbleFormSERVICES();

tableForm.Show();

}

private void btnEdit2\_Click(object sender, EventArgs e)

{

}

private void button3\_Click(object sender, EventArgs e)

{

btnView3.Visible = true;

if (btnView.Visible || btnEdit.Visible)

{

btnView.Visible = false;

btnEdit.Visible = false;

}

// Скрываем кнопки "Просмотр услуг" и "Редактирование услуг", если они были видимыми

if (btnView2.Visible)

{

btnView2.Visible = false;

}

}

private void btnView3\_Click(object sender, EventArgs e)

{

TableFormFILIALS tableForm = new TableFormFILIALS();

tableForm.Show();

}

private void DirectorForm\_Load(object sender, EventArgs e)

{

}

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

{

TableFormCLIENTS\_PERSONALS tableForm = new TableFormCLIENTS\_PERSONALS();

tableForm.Show();

}

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

{

TableFormClients\_Personals2 tableForm = new TableFormClients\_Personals2();

tableForm.Show();

}

}

}

**Б.5 DirectorForm.cs**

namespace kurs  
{  
partial class Form5  
{  
/// <summary>  
/// Required designer variable.  
/// </summary>  
private System.ComponentModel.IContainer components = null;  
  
/// <summary>  
/// Clean up any resources being used.  
/// </summary>  
/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>  
protected override void Dispose(bool disposing)  
{  
if (disposing && (components != null))  
{  
components.Dispose();  
}  
base.Dispose(disposing);  
}  
  
#region Windows Form Designer generated code  
  
/// <summary>  
/// Required method for Designer support - do not modify  
/// the contents of this method with the code editor.  
/// </summary>  
private void InitializeComponent()  
{  
this.label1 = new System.Windows.Forms.Label();  
this.label2 = new System.Windows.Forms.Label();  
this.label3 = new System.Windows.Forms.Label();  
this.label4 = new System.Windows.Forms.Label();  
this.label5 = new System.Windows.Forms.Label();  
this.textBox1 = new System.Windows.Forms.TextBox();  
this.textBox2 = new System.Windows.Forms.TextBox();  
this.textBox3 = new System.Windows.Forms.TextBox();  
this.textBox4 = new System.Windows.Forms.TextBox();  
this.button1 = new System.Windows.Forms.Button();  
this.button2 = new System.Windows.Forms.Button();  
this.SuspendLayout();  
//   
// label1  
//   
this.label1.AutoSize = true;  
this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 27.75F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));  
this.label1.Location = new System.Drawing.Point(56, 9);  
this.label1.Name = "label1";  
this.label1.Size = new System.Drawing.Size(362, 42);  
this.label1.TabIndex = 0;  
this.label1.Text = "Создание договора";  
//   
// label2  
//   
this.label2.AutoSize = true;  
this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif", 14.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));  
this.label2.Location = new System.Drawing.Point(48, 103);  
this.label2.Name = "label2";  
this.label2.Size = new System.Drawing.Size(77, 24);  
this.label2.TabIndex = 1;  
this.label2.Text = "AgentID";  
//   
// label3  
//   
this.label3.AutoSize = true;  
this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif", 14.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));  
this.label3.Location = new System.Drawing.Point(48, 141);  
this.label3.Name = "label3";  
this.label3.Size = new System.Drawing.Size(74, 24);  
this.label3.TabIndex = 2;  
this.label3.Text = "ClientID";  
//   
// label4  
//   
this.label4.AutoSize = true;  
this.label4.Font = new System.Drawing.Font("Microsoft Sans Serif", 14.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));  
this.label4.Location = new System.Drawing.Point(48, 180);  
this.label4.Name = "label4";  
this.label4.Size = new System.Drawing.Size(153, 24);  
this.label4.TabIndex = 3;  
this.label4.Text = "InsuranceTypeID";  
//   
// label5  
//   
this.label5.AutoSize = true;  
this.label5.Font = new System.Drawing.Font("Microsoft Sans Serif", 14.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

**Б.6 TableFormCLIENTS\_PERSONALS.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class TableFormCLIENTS\_PERSONALS : Form

{

public TableFormCLIENTS\_PERSONALS()

{

InitializeComponent();

}

private void TableFormCLIENTS\_PERSONALS\_Load(object sender, EventArgs e)

{

FillDataGridView();

}

private void FillDataGridView()

{

string connectionString = "Server=DESKTOP-1RP44O8;Database=insuranceCompany;Integrated Security=true;";

string sqlQuery = "SELECT \* FROM Клиенты";

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand selectCommand = new SqlCommand(sqlQuery, connection);

SqlDataAdapter adapter = new SqlDataAdapter(sqlQuery, connection);

adapter.SelectCommand = selectCommand;

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

dataGridView1.DataSource = dataTable;

}

}

private void SaveChangesToDatabase()

{

string connectionString = "Server=DESKTOP-1RP44O8;Database=insuranceCompany;Integrated Security=true;";

string sqlQuery = "SELECT \* FROM Клиенты";

using (SqlConnection connection = new SqlConnection(connectionString))

{

// Создание адаптера данных для обновления данных в базе данных

SqlDataAdapter adapter = new SqlDataAdapter(sqlQuery,connection);

// Создание команды для обновления данных в базе данных

SqlCommandBuilder builder = new SqlCommandBuilder(adapter);

// Обновление данных в базе данных

adapter.Update((DataTable)dataGridView1.DataSource);

}

}

private void TableFormCLIENTS\_PERSONALS\_FormClosing\_1(object sender, FormClosingEventArgs e)

{

// Сохранение изменений в базе данных перед закрытием формы

SaveChangesToDatabase();

}

}

**Б.7 TableFormClietns\_Personals2.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class TableFormClients\_Personals2 : Form

{

public TableFormClients\_Personals2()

{

InitializeComponent();

}

private void TableFormClients\_Personals2\_Load(object sender, EventArgs e)

{

FillDataGridView();

}

private void FillDataGridView()

{

string connectionString = "Server=DESKTOP-1RP44O8;Database=insuranceCompany;Integrated Security=true;";

string sqlQuery = "SELECT \* FROM Сотрудники";

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand selectCommand = new SqlCommand(sqlQuery, connection);

SqlDataAdapter adapter = new SqlDataAdapter(sqlQuery, connection);

adapter.SelectCommand = selectCommand;

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

dataGridView1.DataSource = dataTable;

}

}

private void SaveChangesToDatabase()

{

string connectionString = "Server=DESKTOP-1RP44O8;Database=insuranceCompany;Integrated Security=true;";

string sqlQuery = "SELECT \* FROM Сотрудники";

using (SqlConnection connection = new SqlConnection(connectionString))

{

// Создание адаптера данных для обновления данных в базе данных

SqlDataAdapter adapter = new SqlDataAdapter(sqlQuery, connection);

// Создание команды для обновления данных в базе данных

SqlCommandBuilder builder = new SqlCommandBuilder(adapter);

// Обновление данных в базе данных

adapter.Update((DataTable)dataGridView1.DataSource);

}

}

private void TableFormClients\_Personals2\_FormClosing(object sender, FormClosingEventArgs e)

{

SaveChangesToDatabase();

}

}

}

}

**Б.8 TableFormSERVICES.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class TalbleFormSERVICES : Form

{

public TalbleFormSERVICES()

{

InitializeComponent();

}

private void TalbleFormSERVICES\_Load(object sender, EventArgs e)

{

FillDataGridView();

}

private void FillDataGridView()

{

string connectionString = "Server=DESKTOP-1RP44O8;Database=insuranceCompany;Integrated Security=true;";

string sqlQuery = "SELECT \* FROM Услуги";

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand selectCommand = new SqlCommand(sqlQuery, connection);

SqlDataAdapter adapter = new SqlDataAdapter(sqlQuery, connection);

adapter.SelectCommand = selectCommand;

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

dataGridView1.DataSource = dataTable;

}

}

private void SaveChangesToDatabase()

{

string connectionString = "Server=DESKTOP-1RP44O8;Database=insuranceCompany;Integrated Security=true;";

string sqlQuery = "SELECT \* FROM Услуги";

using (SqlConnection connection = new SqlConnection(connectionString))

{

// Создание адаптера данных для обновления данных в базе данных

SqlDataAdapter adapter = new SqlDataAdapter(sqlQuery, connection);

// Создание команды для обновления данных в базе данных

SqlCommandBuilder builder = new SqlCommandBuilder(adapter);

// Обновление данных в базе данных

adapter.Update((DataTable)dataGridView1.DataSource);

}

}

private void TalbleFormSERVICES\_FormClosing(object sender, FormClosingEventArgs e)

{

SaveChangesToDatabase();

}

}

}

**Б.9 TableFormFILIALS.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class TableFormFILIALS : Form

{

public TableFormFILIALS()

{

InitializeComponent();

}

private void TableFormFILIALS\_Load(object sender, EventArgs e)

{

FillDataGridView();

}

private void FillDataGridView()

{

string connectionString = "Server=DESKTOP-1RP44O8;Database=insuranceCompany;Integrated Security=true;";

string sqlQuery = "SELECT \* FROM Филиалы";

using (SqlConnection connection = new SqlConnection(connectionString))

{

SqlCommand selectCommand = new SqlCommand(sqlQuery, connection);

SqlDataAdapter adapter = new SqlDataAdapter(sqlQuery, connection);

adapter.SelectCommand = selectCommand;

DataTable dataTable = new DataTable();

adapter.Fill(dataTable);

dataGridView1.DataSource = dataTable;

}

}

private void SaveChangesToDatabase()

{

string connectionString = "Server=DESKTOP-1RP44O8;Database=insuranceCompany;Integrated Security=true;";

string sqlQuery = "SELECT \* FROM Филиалы";

using (SqlConnection connection = new SqlConnection(connectionString))

{

// Создание адаптера данных для обновления данных в базе данных

SqlDataAdapter adapter = new SqlDataAdapter(sqlQuery, connection);

// Создание команды для обновления данных в базе данных

SqlCommandBuilder builder = new SqlCommandBuilder(adapter);

// Обновление данных в базе данных

adapter.Update((DataTable)dataGridView1.DataSource);

}

}

private void TableFormFILIALS\_FormClosing(object sender, FormClosingEventArgs e)

{

SaveChangesToDatabase();

}

}

}

**ПРИЛОЖЕНИЕ В**

(обязательное)

**TestCase**

Таблица В.1 – Результат ручного тестирования

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Title** | **Test Steps** | **Test Data** | **Expected Result** | **Actual Result** | **Status** |
| 1. Запуск программы | а) Дважды нажать на файл  б) Дождаться запуска главного меню |  | Программа запущена, на экране отображено главное меню. | Программа запущена, на экране отображено главное меню. | Passed |
| 1. Закрытие программы | а) Запустить программу  б) Начать игру, нажав на старт  в) Нажать на крестик в углу формы программы |  | После закрытия, программа работает корректно. | После закрытия программа работает корректно. | Passed |
| 1. Проверка возможности входа в систему при введении верных логина и пароля. | а) Ввести корректные данные.  б) Нажать «авторизоваться». | Логин: direct  Пароль: 15998753457989623148 | Окно входа закрывается. Открывается окно главного меню. | Окно входа закрывается. Открывается окно главного меню. | Passed |
| 1. Проверка возможности входа в систему при введении неверных логина или пароля. | а) Ввести некорректные данные.  б) Нажать «авторизоваться». | Логин: direct1  Пароль: 1599875345 | Выводится сообщение «Неверный логин/пароль». | Выводится сообщение «Неверный логин/пароль». | Passed |
| 1. Проверка возможности просмотра Клиентов и сотрудников | а) После авторизации выбрать «Клиенты и сотрудники»  б) Нажать «Просмотр». | Клиенты/сотрудники: Сотрудник ФИО: Медиков Мурад Валюжный  Дата рождения: 12.03.1999  Должность: Системный Администратор  Адрес: Москва, маршала Мерецкова, д3, корп1,кв12  Телефон: +79520554328 | Окно открывается со всей информацией | Окно открывается со всей информацией | Passed |
| 6) Проверка возможности редактирования со внесением корректных данных | а) После авторизации выбрать «Клиенты и сотрудники»  б) Нажать «Редактирование». | Клиенты/сотрудники: Сотрудник ФИО: Иванов Иван Иванович  Дата рождения: 12.03.1993  Должность: Бухгалтер  Адрес старый: Москва, проспект Маршала жукова, д14, корп1,кв123 Адрес новый: Москва, проспект Маршала жукова, д17, корп1,кв12  Телефон: +799688566213 | Данные обновляются с выводом окна «данные сохранены» | Данные обновляются с выводом окна «данные сохранены» | Passed |
| 1. Проверка возможности редактирования со внесением некорректных данных | а) После авторизации выбрать «Клиенты и сотрудники»  б) Нажать «Редактирование». | Клиенты/сотрудники: Сотрудник ФИО: Иванов Иван Иванович  Дата рождения: 12.03.1993  Должность: Бухгалтер  Адрес старый: Москва, проспект Маршала жукова, д14, корп1,кв123 Адрес новый: Москва, проспект Маршала жукова, д17 , корп1 строение два ,кв12  Телефон: +799688566213 | Вывод сообщения: «введены некорректные данные, проверьте информацию повторно» | Вывод сообщения: «введены некорректные данные, проверьте информацию повторно» | Passed |
| 1. Проверка возможности просмотра списка услуг | а) После авторизации выбрать «услуги»  б) Нажать «список услуг» | Наименование: 1) Оформление ОСАГО  2) Страхование Жизни  3) Страхование недвижимости  4) Страхование домашнего питомца | Окно открывается со всей информацией | Окно открывается со всей информацией | Passed |
| 1. Проверка возможности редактирования услуг со внесением корректных данных | а) После авторизации выбрать «услуги»  б) Нажать «редактирование» | Наименование старых услуг:  1) Оформление ОСАГО  2) Страхование Жизни  3) Страхование недвижимости  4) Страхование домашнего питомца  наименование новых услуг: 1) Оформление ОСАГО  2) Страхование Жизни  3) Страхование недвижимости  4) Страхование несчастных случаев | Данные обновляются с выводом окна «данные сохранены» | Данные обновляются с выводом окна «данные сохранены» | Passed |
| 1. Проверка возможности редактирования услуг со внесением некорректных данных | а) После авторизации выбрать «услуги»  б) Нажать «редактирование» | Наименование старых услуг:  1) Оформление ОСАГО  2) Страхование Жизни  3) Страхование недвижимости  4) Страхование домашнего питомца  наименование новых услуг: 1)оформление ОСАГО  2) Страхование Жизни  3) Страхование недвижимости  4) Страхование несчастных случаев | Вывод сообщения: «введены некорректные данные, проверьте изменения повторно» | Вывод сообщения: «введены некорректные данные, проверьте изменения повторно» | Passed |

**ПРИЛОЖЕНИЕ Г**

(справочное)

**Интерфейс**

**Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, дизайн

Автоматически созданное описание**

Рисунок Г.1 **—**Интерфейс окна авторизации

**Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание**

Рисунок Г.2 **—**Интерфейс окна договора и клиенты