Министерство образования Республики Беларусь

Учреждение образования «Минский государственный колледж электроники»

Отчёт

по учебной практике по разработке и сопровождению программного обеспечения

Разработчик: Никитина А. В.

Руководитель практики: Кулецкая Ю. Н.

Минск, 2019

СОДЕРЖАНИЕ

Введение3

1. Постановка задачи4

1.1 Описание предметной области4

1.2 Обзор существующих аналогов4

1.3 Информационная база задачи4

1.3.1 Входная и выходная информация4

1.4 Формулировка задачи 5

2. Проектирование задачи 6

2.1 Обоснование выбора средств для разработки 6

2.2 Разработка алгоритма решения7

2.3 Проектирование интерфейса 8

3. Реализация задач 10

3.1 Описание разработанных модулей 9

3.2 Описание реализованных методов и событий модулей 9

3.3 Компоненты Windows Forms 11

4. Тестирование 12

5. Применение16

5.1 Установка и системные требования приложения16

5.2 Демонстрационный пример работы16

Заключение20

Список использованных источников21

Приложение А Текст программы22

Приложение Б (Диаграмма вариантов использования)48

Приложение В (Диаграмма состояний)49

Приложение Г (Функциональная диаграмма)50

Приложение Д (Диаграмма деятельности)51

# Приложение Е (Диаграмма потоков данных)52

Приложение Ж (Диаграмма «сущность-связь»)53

# Приложение К (Диаграмма последовательности)54

Приложение Л (Диаграмма классов)55

Приложение М Техническое задание56

# 

# ВВЕДЕНИЕ

Целью практики является закрепление знаний, полученных в процессе изучения дисциплин «Конструирование программ и языки программирования» и «Основы алгоритмизации и программирования»; «Тестирование и отладка ПО»; «Технологий разработки ПО»; знакомство с современными средствами автоматизации процесса проектирования и разработки ПО; закрепление изученных методом и приёмов в области инженерного проектирования и разработки ПО; изучение современного состояния и развития CASE-средств и промышленных технологий проектирования ПО; решения задач на этапе сопровождения программного продукта.

Целью выполнения данной работы является создание программный модуля «Гостиница», с использованием технологии Windows Forms на языке программирования C#, которое поможет автоматизировать процессы в работе администратора гостиницы.

Программный модуль хранит сведения о наличии свободных мест и о проживающих в гостинице, кроме этого он предназначен для бронирования мест в гостинице и оформления проживающих в гостинице, благодаря ему обеспечивается возможность бронирования номеров, оформление гостей, выселение гостей из гостиницы, выборка данных необходимых для просмотра администратора, а также возможность добавления новых номеров и возможность отслеживать наличие свободных номеров в гостинце в заданный период времени.

Используя данный программный продукт, благодаря автоматизации процессов, администратор гостиницы сократит количество затрачиваемых им временных ресурсов.

Данному программному средству присущи следующие свойства:

– простота освоения;

– удобное представление информации;

– эргономичный интерфейс.

Данная пояснительная записка является полным описанием разработанного программного продукта. Она разделена на разделы и подразделы, и каждый раздел или подраздел посвящен определённому аспекту процесса разработки приложения.

# 1. Постановка задачи

# 1.1 Описание предметной области

Целью данной работы является разработка программы, предназначенной для автоматизации процесса работы администратора гостиницы. Программа должна предоставлять возможность добавления, изменения, поиска и фильтрации необходимых данных, кроме этого в программе должна присутствовать возможность сортировки и фильтрации информации.

# 1.2 Обзор существующих аналогов

На данный момент существует множество аналогов данного приложения, и данные аналоги установлены в разных гостиницах. Заказчику они не подходят из-за слишком большого функционала, большого размера и множества ненужных функций.

# 1.3 Информационная база задачи

Информационная база – это совокупность упорядоченной информации, которая используется для функционирования системы и делится на внешнюю и внутреннюю машинную базу.

Как и любая компьютерная программа, данная программа обрабатывает входные данные и выводит результат в виде выходных данных. Входные данные имеют структуру любой задачи – это описание некоторой ситуации (исходные данные) и формулировка вопроса, ответ на который должен быть на основе сложившейся ситуации. Разделение входных данных на две части дает возможность задавать вопросы по уже имеющейся ситуации, не описывая ее заново.

# 1.3.1 Входная и выходная информация

Входными данными для разработанного ПП являются:

* информация о клиенте;
* информация о проживании и брони номера;
* параметры фильтра и поисковых запросов;
* информация вводимая при внесении изменений в проживание;
* информация о новом номера.

Выходными данными разработанного ПП являются:

* информация отображаемая на экран по запросу пользователя;
* результаты поиска и фильтрации данных.

Постоянными данными является база данных.

# 1.4 Формулировка задачи

Создать приложение с использованием технологии Windows Forms на языке программирования C#. Программа должна предоставлять возможность добавления, изменения, поиска и фильтрации необходимых данных необходимых при работе администратора гостиницы.

Разрабатываемый ПП должен выполнять функции:

1. Бронирование номеров и оформление проживания.
2. Фильтрация файлов по разным критериям.
3. Поиск информации по выбранным критериям.

Разработать для проектируемого программного средства:

* техническое задание;
* функциональную диаграмму;
* диаграмму потоков данных;
* диаграмму сущность-связь;
* диаграмму вариантов использования;
* диаграмму деятельности;
* диаграмму последовательности;
* диаграмму состояний;
* диаграмму классов.

Произвести тестирование разработанного программного продукта. Для проведения тестирования на основании разработанных требований написать чек-лист и тест-кейсы.

Чек-лист и тест-кейсы должны быть разработаны, как для смоук- теста, так и для теста критического пути.

# 2. Проектирование задачи

# 2.1 Обоснование выбора средств для разработки

В качестве платформы была взята платформа NET, потому что платформа NET на текущий момент самая передовая и перспективная технология для разработки под Windows и не только. Объектно-ориентированный подход – платформа .NET изначально строится на принципах объектно-ориентированного программирования.

Мощный инструментарий – поставляемая вместе со средой библиотека базовых классов обладает достаточным функционалом для решения задач практически любой сложности.

Разделение кода – в .NET способ разделения кода между приложениями значительно отличается от предшествующих реализаций за счет использования сборок. Сборки обладают формальными средствами для управления версиями и допускают одновременное существование рядом нескольких различных версий сборок.

Поддержка языков высокого уровня – это свойство благоприятно сказывается на удобстве использования, быстроте написания и читабельности кода, что крайне важно для последующей поддержки программы.

Visual Studio – среда разработки, поставляемая вместе с .NET, предоставляет необходимый инструментарий для эффективного и быстрого создания приложений с графическим интерфейсом.

Язык программирования С# - появление технологии .NET повлекло за собой массовую реконструкцию некоторых языков программирования, стремящихся использовать те или иные возможности платформы, такие как С++ и альтернативу - язык, ориентированный специально Visual Basic. Microsoft решили предложить разработчикам альтернативу -язык, ориентированный специально .NET и создали С# и Java, что позволило программистам за достаточно короткое время изучить тонкости нового языка.

Несмотря на то, что С# и .NET предназначены в первую очередь для веб-разработки, их также активно применяют для создания приложений, которые должны устанавливаться на машине конечного пользователя, где и будет выполняться вся обработка данных. Разработку таких приложений обеспечивает библиотека Windows Forms, позволяющая проектировать графический интерфейс. Программа, описанная в данной работе, разработана именно с помощью библиотеки Windows Forms.

Обоснование выбора языка программирования:

* язык программирования С# претендует на подлинную объектную ориентированность;
* язык программирования С# призван реализовать компонентно-ориентированный подход к программированию, который способствует меньшей машинно-архитектурной зависимости результирующего программного кода, большей гибкости, переносимости и легкости повторного использования программ;
* принципиально важным отличием от предшественников является изначальная ориентация на безопасность кода.
* расширенная поддержка событийно-ориентированного программирования;
* язык программирования С# является «родным» для создания приложений в среде Microsoft .NET, поскольку наиболее тесно и эффективно интегрирован с ней.

Windows Forms — интерфейс программирования приложений (API), отвечающий за графический интерфейс пользователя и являющийся частью Microsoft .NET Framework. Данный интерфейс упрощает доступ к элементам интерфейса Microsoft Windows. Преимуществом является и то, что управляемый код — классы, реализующие 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.

# 2.2 Разработка алгоритма решения

Работа с программой будет начинаться с системы авторизации пользователей, программа имеет 2 тип пользователей: администратор программы и администратор гостинцы, каждый из пользователей имеет разграничение по предоставленным возможностям, так например администратор программы имеет возможность добавлять, удалять и изменять пользовательские данные, а администратор гостиницы имеет возможность отслеживать и вносить изменения в процессы проживания гостей и бронирования номеров. После прохождения авторизации пользователь администратор гостиницы перед собой видит главное окно программы.

В главном окне программы будут отображены основные функции программы для работы с процессами брони и проживания, после выбора необходимой функции или действия пользователь может перейти к работе.

При изменении отображаемых таблиц пользователь не теряет возможность выборки данных используя поисковую строку, так как она производит поиск по активной таблице из базы.

Для того, чтобы выполнить бронирование номера в главном окне программы во вкладке действия необходимо нажать на кнопку «Бронирование номера» после чего пользователь попадёт в окно бронирования и просмотрев необходимые данные нажатием на кнопку «Оформить бронирование» перейдёт к заполняемым формам. При выборе гостя пользователю необходимо указать код уже имеющегося гостя, либо же нажатием на кнопку добавить просмотреть осуществить добавление гостя в картотеку гостей, после заполнения всех данных и принятия заказа ему присваивается статус «Бронь».

Для того, чтобы оформить проживание в главном окне в группе действия «Оформить проживание» ввести код проживания, который необходимо привести в статус «Проживание».

Для того, чтобы закрыть проживание в главном окне в группе «Информация о» необходимо выбрать кнопку «Проживающих», после нажатия пользователь попадёт в окно отображающие всех проживающих гостей, и для выселения гостя, то есть закрытия проживания необходимо ввести его код проживания и статус примет статус проживания «Закрыто», кроме того перед этим возможно просмотреть гостей по фильтруя их по статусу проживания

Если нам нужно узнать информацию о свободных номерах, вам необходимо в главном окне в группе «Информация о» необходимо выбрать кнопку «Свободных номерах», прейдя в следующее окно возможно узнать о наличии свободных мест в определённый пользователем период времени, кроме того в этом окне также можно добавить новый номер в таблицу номеров.

В результате выполнения данных алгоритмов реализуются необходимые для решения поставленной задачи процессы.

# 2.3 Проектирование интерфейса

Программа будет состоять из 3 форм, включая главную. А именно:

* форма «Form1» - для авторизации пользователей;
* форма «Form2» - для работы администратора программы;
* форма «Form3» - главное окно администратора гостиницы;
* форма «Form4» - для оформления бронирования номера;
* форма «Form5» - для получения информации о проживающих в гостинице;
* форма «Form6» - для получения информации о свободных номерах.

Главная форма имеет такие компоненты как: DataGridView, BindingSource, TableAdapter используется для отображения таблиц и соеденения с базой данных. groupBox1 используется для отображения групп кнопок Button для выполнения необходимых функций. Кроме этого на форме присутствуют Textbox, которые необходимы для выполнения поиска или введения данных.

Остальные формы программы содержат такие же компоненты, нов некоторых, кроме этого добавляются компоненты ComboBox – для отображения выпадающего списка, NumericUpDown – для выбора числового значения, DateTimePicker – выбора даты.

|  |  |  |
| --- | --- | --- |
| Главное окно | | |
| Бронирование номера | Информация о проживающих | Информация о свободных номерах |

Рисунок 2.3.1 – Предполагаемый интерфейс программы

# 3. Реализация задач

# 3.1 Описание разработанных модулей

В разработанной программе имеется 5 модулей.

Главный модуль «Program.cs» предназначен для запуска главного окна приложения.

Модуль «Form1.cs» предназначен для выполнения авторизации пользователей, через выполнение запросов к базе данных на проверку правильности введённых данных.

Модуль «Form2.cs» предназначен для выбора выполняемых администрирования учётных записей в системе.

Модуль «Form3.cs» предназначен для выбора выполняемых функций и модулей администратором гостиницы, так как модуль относиться к форме Главного окна администратора гостиницы.

Модуль «Form4.cs» предназначен для бронирования номеров гостиницы, а также добавления гостей в ходу оформления бронирования.

Модуль «Form5.cs» предназначен для отображения информации о проживающих в гостиницы по определённым критериям.

Модуль «Form6.cs» предназначен для получения информации о свободных номерах в гостинице на определённую дату.

# 3.2 Описание реализованных методов и событий модулей

Методы и события формы «Form3» - Главной формы администратора гостиницы:

private void Form3\_Load(object sender, EventArgs e) – загрузка при старте программы.

private void FrmMain\_MouseDown(object sender, MouseEventArgs e) и private void FrmMain\_MouseMove(object sender, MouseEventArgs e) – для перемещения окна программы по экрану пк;

private void Button3\_Click(object sender, EventArgs e) – для свертывания окна программы;

private void Button2\_Click(object sender, EventArgs e) – для закрытие программы;

private void Button1\_Click(object sender, EventArgs e) – для открытия формы бронирования;

private void Button10\_Click(object sender, EventArgs e) – для открытие формы для просмотра информации о проживающих в гостинице;

private void Button4\_Click(object sender, EventArgs e) – для оформления гостя;

private void TextBox2\_TextChanged(object sender, EventArgs e) – для поиска по таблицам;

private void Button11\_Click(object sender, EventArgs e), private void Button6\_Click(object sender, EventArgs e), private void Button9\_Click(object sender, EventArgs e), private void Button5\_Click(object sender, EventArgs e), private void Button3\_Click(object sender, EventArgs e), private void Button7\_Click(object sender, EventArgs e) – для выбора отображаемой таблицы на форме.

# 3.3 Компоненты Windows Forms

При разработке программы использовались следующие компоненты:

* DataGridView – компонент предназначен для отображения выбранных данных из таблиц базы данных;
* BindingSource – компонент предназначен для инкапсуляции источника данных, фильтрации и их сортировки;
* TableAdapter – Для обновления данных таблицы;
* TextBox – компонент предназначен для ввода информации;
* Button – компонент представляет собой элемент управления «Кнопка Windows»;
* В моем случае использовался для запуска процессов реализации методов;
* Label – компонент представляет собой стандартную метку Windows для создания текстовых пояснений;
* PictureBox– компонент предназначен для отображения картинок;
* ComboBox – компонент предназначен для отображения выпадающего списка;
* NumericUpDown – компонент предназначен для выбора числового значения;
* DateTimePicker – компонент предназначен для выбора и отображения даты;
* GroupBox – компонент предназначен для группировки компонентов.

# 4. Тестирование

Тестирование - ряд мероприятий, связанных с различного рода испытаниями объекта тестирования с целью установления соответствия или несоответствия его характеристик определенным требованиям и выявления дефектов. Дефектами, в свою очередь, могут быть как ошибки в работе, так и неприемлемое качество функционирования в определенных условиях эксплуатации.

Тестирование программного продукта проводят как на этапах его создания, так и на этапах технического сопровождения и дальнейшего развития. Виды тестирования программных продуктов и их краткое описание я постараюсь представить в этом материале.

Виды тестирования на этапе создания программного продукта классифицируют по преследуемым этим процессом целям. **Функциональное тестирование** - проверка корректного выполнения программой заложенных в нее функций. **Нефункциональное тестирование** - все прочие виды испытаний, такие как тестирование производительности в различных режимах эксплуатации, тестирование эргономики пользовательского интерфейса, тестирование отказоустойчивости и т.д.

Для проведения функционального тестирования персоналом отдела технического контроля разрабатывается документ программа и методика испытанийфункционала приложения (ПМИ). Документ ПМИ содержит перечень сценариев тестирования программного продукта (test cases) с подробным описанием шагов. Каждый шаг сценария тестирования характеризуется действиями пользователя (специалиста по тестированию) и ожидаемыми результатами – ответной реакции программы на эти действия. Программа и методика испытаний обязана имитировать эксплуатацию программного продукта в реальном режиме.

Нефункциональное тестирование, как следует из самого названия, в первую очередь оценивает второстепенные качества функционирования программного продукта. Второстепенные не в том смысле, что эти качества программного продукта менее важны, чем корректное выполнение им своих основных функций, а в том, что они никому не интересны, если продукт, в принципе, не выполняет то, что должен. Кому, например, интересны результаты замеров производительности подсистемы расчетов, если сами результаты этих расчетов не верны? Для кого имеет значение удобство пользовательского интерфейса программы, если его работа не приводит к ожидаемым от него результатам? Ну а насчет важности нефункционального тестирования можно сказать так: иногда его результаты имеют более серьезный вес в общей оценке работы программного продукта, чем результаты функционального тестирования. Например, для системы, работающей в реальном времени куда как более важно успевать взаимодействовать со всеми наблюдаемыми внешними объектами, чем корректно, с точки зрения функциональных требований, обрабатывать события от конкретного источника.

Таблица 4.1 - Чек-лист

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Модуль | Подмодуль/ экран | Описание теста | Ожидаемый результат | Test Case Passed/Failed |
| Приложение |  | Запуск приложения | Открытие приложения | Passed |
|  |  | Начало работы | Отображение данных в таблицах | Passed |
|  |  | Работа с основными функциями | Реализация работы с бронированием номера и оформлением гостей | Passed |
|  |  | Завершение работы | Закрытие программы | Passed |
|  |  | Сохранение результатов редактирования или добавления | Результат работы сохраняется автоматически | Passed |
| Администраторское окно | 1.Добавить учётную запись | Добавление учётной записи | Создаёться новая запись в таблице Users | Passed |
|  | 2. Изменить учётную запись | Изменение учётной записи | Изменение существующей учётной записи и сохранение изменений | Passed |
|  | 3.Удалить учётную запись | Удаленина учётная запись | Учётная запись удалена из системы | Passed |
|  | 4.Выбрать параметры поиска и фильтра | Выполнение поиска и фильтрации по выбранным данным | Поиск и фильтрация выполняеться успешно | Passed |

Таблица 4.2 – Test-case

|  |  |
| --- | --- |
| Запустить приложение | Приложение запустилось |
| Выполнить вход  1. Войти в учётною запись администратора гостиницы | 1. Вход выполнен |
| Выполнить вход  1.Войти в учётну запись администратора программы | 1. Вход выполнен |
| Просмотр информации.  1. Открыть любую таблицу  2. Выбрать параметры отображения | 1. Отображаются все столбцы и строки таблицы |
| Поиск данных в таблицк  1. Ввести условие поиска. | 1. Поисковые сиволы выделены введено.  2. В таблице отображается искомая информация. |
| Фильтрация  1. Выбрать параметр фильтра.  2. Нажать на кнопку фильрация. | 1. Выводятся отфильтрованные данные в таблице. |
| Оформить бронирование  1. Ввести все данные  2. Нажать на кнопку «Добавить/Просмотреть»  3. Доваить клиента  4. Принять заказ | 1. Создается и отображаеться только что созданная запись в таблице «Проживание» |
| Удалить учётную запись  1. Выделить учётную запись  2. Нажать «Удалить учётную запись». | 1. Запись выделена.  2. Учётная запись удалена |
| Создать учётную запись  1. Нажать на кнопку «Создать учётную запись».  2. Вывести всю необходимую информацию.  3. Внести изменения.  4. Нажать кнопку «ОК». | 1. Учётная запись создана.  2. Запись отображаеться в таблице |
| Добавить номер  1. Перейти в информацию о свободных номерах  2. Ввести всю информацию о номере.  3. Нажать на кнопку «Добавить». | 1. Номер создан.  2. Запись отображаеться в таблице. |
| Просмотреть информацию о свободных номерах  1. Ввести даты с и после.  2. нажать на кнопку «Подабрать». | 1. Выведены номера свободные в этом временном промежутке. |
| Закрыть проживание  1. Перейти в информацию о проживающих в гостинице  2. Ввести код проживание.  3.Нажать кнопку «Выселить ». | 1. Статус проживания изменен  2. Данные сохранены. |

# 5. Применение

# 5.1 Установка и системные требования приложения

Установка приложение не требуется, для его работоспособности необходимо поместить папку Hotel\_ на диск С и скопировать ярлык программы по пути Hotel\_\Hotel\_\bin\Debug\ Hotel\_.exe и поместить в удобное для пользователя место.

Программа предназначена для работы всех лиц, имеющих доступ к программному обеспечению. Программа разработана не для коммерческих целей и не для использования третьих лиц.

Минимальные системные требования:

* OC Windows XP/Vista/7/8/10;
* ОЗУ 1024 Мб;
* места на жестком диске 10 Мб;
* частота процессора 1 ГГц;
* .NET Framework должен быть 4.7.2.

# 5.2 Демонстрационный пример работы

При запуске программы пользователю необходимо пройти авторизацию, окно отображаемое при запуске отображено на рисунке 5.2.1.

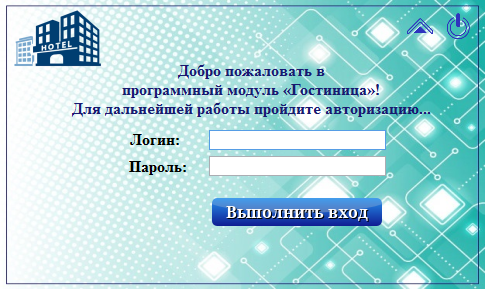


Рисунок 5.2.1 – Окно авторизации

В программе есть 2-а типа учётных записей: администратор программы и администратор гостиницы. Выполним вход в учётную запись администратора программы. Рассмотрим функциональные возможности данного типа учётной записи:

1. добавление учётной записи;
2. изменение учётной записи;
3. удаление учётной записи;
4. выбор параметра поиска;
5. проведение поиска;
6. фильтрация по выбранным параметрам;
7. просмотр таблицы пользователей.

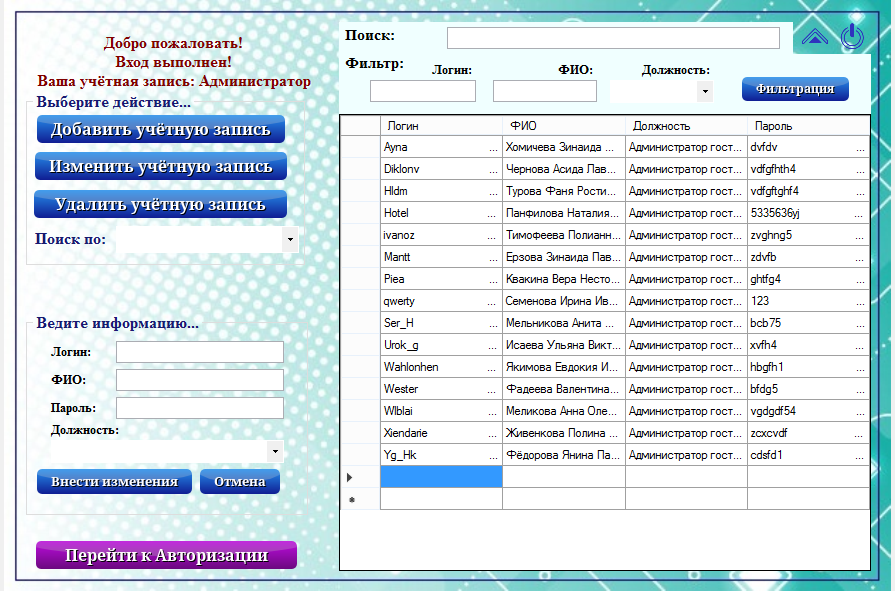


Рисунок 5.2.2 – Окно администратора программы

Выполним добавление и удаление учётной записи.

Добавление: Нажимаем на соответствующую кнопку «Добавить учётную запись», Вводим логин, пароль, ФИО, выбираем должность, после чего нажимаем на кнопку «Внести изменения». Результат – Запись добавлена.

Удаление: Выбираем учётную запись, которую необходимо удалить, нажимаем на соответствующую кнопку «Удалить учётную запись», запись удалена.

Перейдём к Авторизации и снова окажемся в окне авторизации. Рассмотрим учётную запись администратора гостиницы, при запуске окно отображено на рисунке 5.2.3.

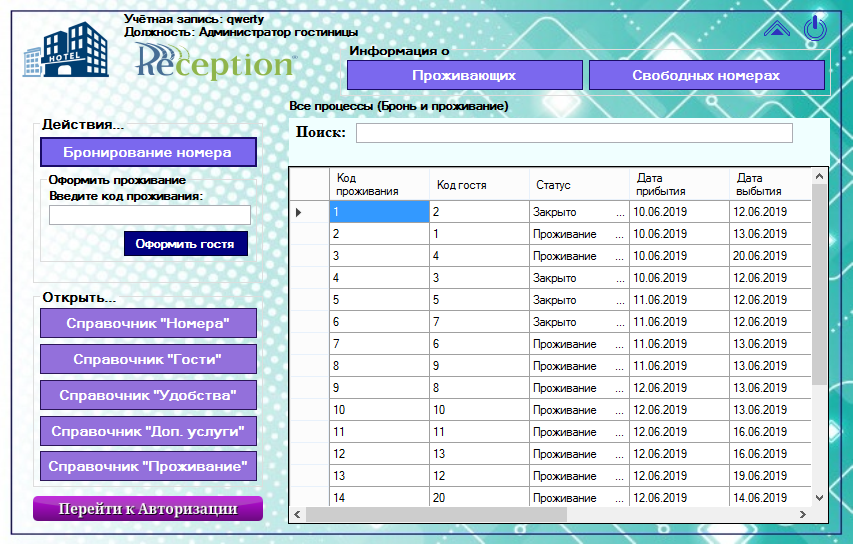


Рисунок 5.2.3 – Окно администратора гостиницы

Просмотрим справочники нажатием на кнопки группы открыть, после чего введём условия поиска, всё выполнено успешно см. Рисунок 5.2.4.

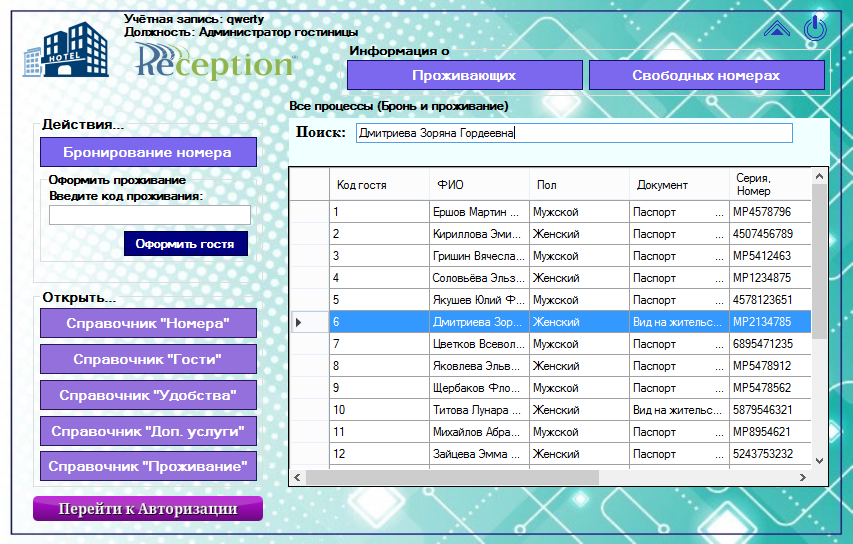


Рисунок 5.2.4 – Результат поиска

После чего оформим бронирование номера прейдём на форму бронирования номера используя кнопку «Бронирование номера» Далее нажав на кнопку «Оформить бронирование» переходим к заполнению данных, при выборе мы можем открывать справочники уточняя информацию, а также при добавлять гостя при их отсутствии в уже существующих, для добавление необходимо нажать на кнопку «Добавить/Просмотреть» и ввести всю информацию о госте, после заполнения данных нажимаем на кнопку «Принять заказ». Окно оформления бронирования представлено на рисунке 5.2.5.

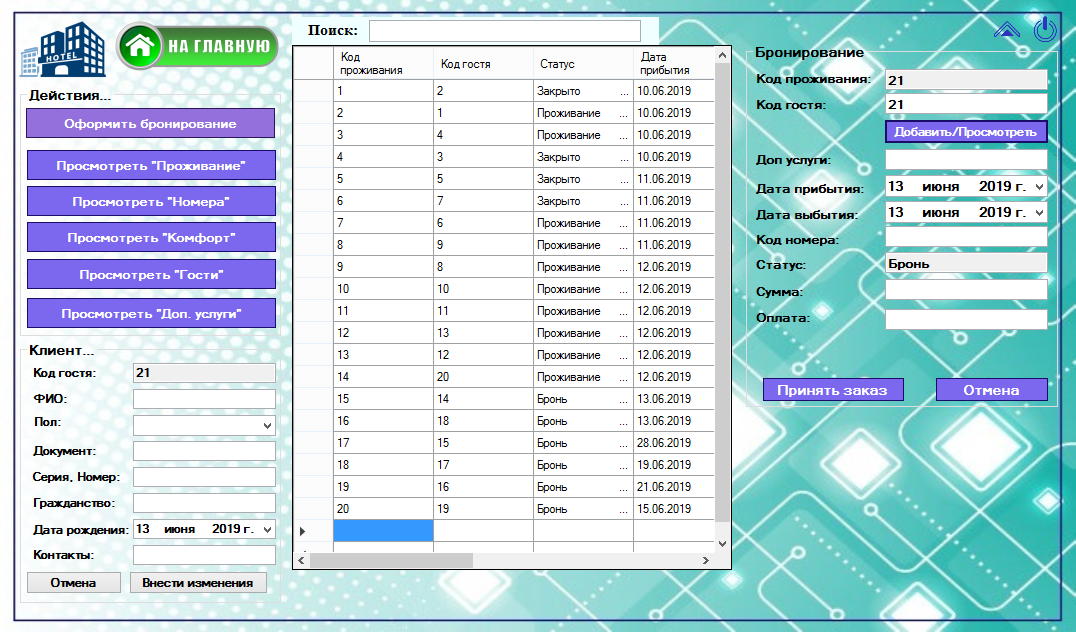


Рисунок 5.2.5 – Окно оформления бронирования

Далее нажатием на кнопку закрытия программы в правом верхнем углу.

# Заключение

В ходе прохождения практики были получены и закреплены знания в области дисциплин «Конструирование программ и языки программирования» и «Основы алгоритмизации и программирования»; «Тестирование и отладка ПО»; «Технология разработки ПО»;

Были освоены приемы проведения тестирования ПО, в частности разработки Чек-листа и Test-case.

В итоге была разработана программа, с понятным для пользователя интерфейсом, предназначен для бронирования мест в гостинице и оформления проживающих в гостинице, благодаря ему обеспечивается возможность бронирования номеров, оформление гостей, выселение гостей из гостиницы, выборка данных необходимых для просмотра администратора, а также возможность добавления новых номеров и возможность отслеживать наличие свободных номеров в гостинце в заданный период времени.

Используя данный программный продукт, благодаря автоматизации процессов, администратор гостиницы сократит количество затрачиваемых им временных ресурсов.

Данному программному средству присущи следующие свойства:

– простота освоения;

– удобное представление информации;

– эргономичный интерфейс.

В поставленные цели и функции были реализованы полноценно.

Список использованных источников

1. Методические указания по выполнению проектов Специальность: 2-40 01 01 «Программное обеспечение информационных технологий».
2. Албахари, Джозеф, Албахари, Бен – C# 6.0. Справочник. Полное описание языка – Вильямс, 2016 – 1040 с.
3. Зиборов, В. В. – Visual C# 2012 на примерах – БХВ-Петербург, 2013 – 475 с.
4. Дейтел, Пол, Дейтел, Харви – Как программировать на Visual C# 2012 – Спб.:Питер, 2015 – 858 с.
5. Хабрахабр [Электронный ресурс]. – Электронные данные. – Режим доступа: http://habrahabr.ru.
6. MSDN MicroSoft [Электронный ресурс]. – Электронные данные. – Режим доступа: http://msdn.microsoft.com.
7. Шилдт Г. – С#. Учебный курс.

# Приложение А (обязательное)

Листинг файла Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Hotel\_

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

Листинг файла Form1.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 Hotel\_

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

string login;

private int x = 0; private int y = 0;

private void textBox1\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

textBox2.Focus();

e.SuppressKeyPress = true;

}

}

private void textBox2\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

e.SuppressKeyPress = true;

Button1\_Click(sender, e);

}

}

private void FrmMain\_MouseDown(object sender, MouseEventArgs e)

{

x = e.X; y = e.Y;

}

private void FrmMain\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

Location = new Point(Location.X + (e.X - x), Location.Y + (e.Y - y));

}

}

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

{

if ((textBox1.Text == "Admin") && (textBox2.Text == "qwerty\_ad"))

{

this.Hide(); Form2 fr2 = new Form2(); fr2.Show();

}

else if ((textBox1.Text == "") || (textBox2.Text == ""))

{

MessageBox.Show("Ошибка! Введите данные для входа! ", "Сообщение", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + Application.StartupPath + @"\Hotel.mdf" + "'; Integrated Security = True";

string mySelectQuery = "SELECT Логин FROM [Users] WHERE Логин='" + textBox1.Text + "'AND Пароль='" + textBox2.Text + "' ";

SqlConnection myConnection = new SqlConnection(myConnectionString);

SqlCommand myCommand = new SqlCommand(mySelectQuery, myConnection);

try

{

myConnection.Open();

SqlDataReader myReader = myCommand.ExecuteReader();

if (myReader.HasRows == false) { MessageBox.Show("Ошибка! Вход не выполнен, проверьте введённые данные! ", "Сообщение", MessageBoxButtons.OK, MessageBoxIcon.Information); }

else

{

while (myReader.Read()) { login = myReader.GetString(0); }

this.Hide();

Form3 fr3 = new Form3(login);

fr3.Show();

}

}

catch

{

MessageBox.Show("Ошибка! Вход не выполнен, проверьте введённые данные! ", "Сообщение", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

finally

{

myConnection.Close();

}

}

}

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

{

textBox1.Focus();

textBox1.Select();

}

private void Button3\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void Button2\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void Form1\_Load\_1(object sender, EventArgs e)

{

}

}

}

Листинг файла Form2.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 Hotel\_

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

}

bool d = false, iz = false; int znach = 0;

private void Form2\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "hotelDataSet.Users". При необходимости она может быть перемещена или удалена.

this.usersTableAdapter.Fill(this.hotelDataSet.Users);

}

private int x = 0; private int y = 0;

private void FrmMain\_MouseDown(object sender, MouseEventArgs e)

{

x = e.X; y = e.Y;

}

private void FrmMain\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

Location = new Point(Location.X + (e.X - x), Location.Y + (e.Y - y));

}

}

private void GroupBox1\_Enter(object sender, EventArgs e)

{

}

private void Button8\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void Button9\_Click(object sender, EventArgs e)

{

Application.Exit();

}

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

{

this.Validate();

this.usersBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.hotelDataSet);

}

bool b = false; bool dob = false;

private void Button3\_Click(object sender, EventArgs e)

{

if (b == false)

{

usersBindingSource.AddNew();

логинTextBox.ReadOnly = false;

groupBox1.Visible = true; d = true;

label8.Visible = false;

b = true; dob = true;

}

}

private void Button5\_Click(object sender, EventArgs e)

{

label8.Visible = false;

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection connect = new SqlConnection(myConnectionString);

string sql = "Delete from [Users] WHERE Логин=@Логин;";

SqlCommand cmd\_SQL = new SqlCommand(sql, connect);

cmd\_SQL.Parameters.AddWithValue("@Логин", usersDataGridView.Rows[usersDataGridView.CurrentRow.Index].Cells[0].Value.ToString());

try

{

connect.Open();

int n = cmd\_SQL.ExecuteNonQuery();

label8.Text = "Запись успешно Удалена!";

label8.Visible = true;

}

catch (SqlException ex)

{

label8.Text = "Ошибка! Запись не удалена";

label8.Visible = true;

}

finally

{

connect.Close();

}

usersTableAdapter.Update(hotelDataSet.Users);

this.usersTableAdapter.Fill(this.hotelDataSet.Users);

}

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

{

b = false;

if (логинTextBox.Text != "")

{

if (d == true)

{

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection connect = new SqlConnection(myConnectionString);

string sql = "INSERT INTO [Users] (Логин, ФИО, Должность, Пароль) VALUES (@Логин, @ФИО, @Должность,@Пароль );";

SqlCommand cmd\_SQL = new SqlCommand(sql, connect);

cmd\_SQL.Parameters.AddWithValue("@Логин", логинTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@ФИО", фИОTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Должность", должностьComboBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Пароль", парольTextBox.Text);

try

{

connect.Open();

int n = cmd\_SQL.ExecuteNonQuery();

label8.Text = "Запись успешно добавлена!";

label8.Visible = true;

}

catch (SqlException ex)

{

label8.Text = "Ошибка! Запись не добавлена введите данные корректно! ";

label8.Visible = true;

}

finally

{

connect.Close();

this.usersTableAdapter.Fill(this.hotelDataSet.Users);

}

}

if (iz == true)

{

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection connect = new SqlConnection(myConnectionString);

string sql = "Update [Users] set ФИО=@ФИО, Должность=@Должность, Пароль=@Пароль WHERE Логин=@Логин;";

SqlCommand cmd\_SQL = new SqlCommand(sql, connect);

cmd\_SQL.Parameters.AddWithValue("@Логин", логинTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@ФИО", фИОTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Должность", должностьComboBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Пароль", парольTextBox.Text);

try

{

connect.Open();

int n = cmd\_SQL.ExecuteNonQuery();

label8.Text = "Запись успешно измененина!";

label8.Visible = true;

}

catch (SqlException ex)

{

label8.Text = "Ошибка! Запись не измененина";

label8.Visible = true;

}

finally

{

connect.Close();

this.usersTableAdapter.Fill(this.hotelDataSet.Users);

}

}

логинTextBox.ReadOnly = false;

groupBox1.Visible = false;

Validate();

usersBindingSource.EndEdit();

usersTableAdapter.Update(hotelDataSet.Users);

}

else label8.Text = "Поле логина не может быть пустым!";

}

private void Button7\_Click(object sender, EventArgs e)

{

if (dob != true)

{

b = false;

label8.Visible = false;

usersDataGridView.Rows[usersDataGridView.RowCount - 2].Cells[0].Value = " ";

usersBindingSource.RemoveAt(usersDataGridView.RowCount - 2);

Validate();

usersBindingSource.EndEdit();

usersTableAdapter.Update(hotelDataSet.Users);

groupBox1.Visible = false;

}

groupBox1.Visible = false;

}

private void Button2\_Click(object sender, EventArgs e)

{

Form1 frm = new Form1();

frm.Show(); this.Close();

}

private void ComboBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

label8.Visible = false;

switch (comboBox3.SelectedIndex)

{

case 0:

znach = 0;

break;

case 1:

znach = 1;

break;

case 2:

znach = 2;

break;

case 3:

znach = 3;

break;

default:

znach = 0;

break;

}

}

private void TextBox1\_TextChanged(object sender, EventArgs e)

{

if ((znach == 0) && (textBox1.Text != ""))

{

for (int i = 0; i < usersDataGridView.RowCount; i++)

{

usersDataGridView.Rows[i].Selected = false;

for (int j = 0; j < usersDataGridView.ColumnCount; j++)

if (usersDataGridView.Rows[i].Cells[j].Value != null)

if (usersDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox1.Text))

{

usersDataGridView.Rows[i].Selected = true;

}

}

}

else if (textBox1.Text != "") Search(znach);

else if (textBox1.Text == "") usersDataGridView.ClearSelection();

}

private void Search(int znach)

{

for (int i = 0; i < usersDataGridView.RowCount; i++)

{

usersDataGridView.Rows[i].Selected = false;

int j = znach - 1;

if (usersDataGridView.Rows[i].Cells[j].Value != null)

if (usersDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox1.Text))

{

usersDataGridView.Rows[i].Selected = true;

if (textBox1.Text == "") usersDataGridView.ClearSelection();

}

}

}

private void Button6\_Click(object sender, EventArgs e)

{

List<string> filterParts = new List<string>();

if (textBox2.Text != "")

filterParts.Add("Логин = '" + textBox2.Text + "'");

if (textBox3.Text != "")

filterParts.Add("ФИО = '" + textBox3.Text + "'");

if (comboBox2.SelectedIndex >= 0)

filterParts.Add("Должность = '" + comboBox2.Text + "'");

string filter = string.Join(" AND ", filterParts);

usersBindingSource.Filter = filter;

}

private void ComboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

switch (comboBox2.SelectedIndex)

{

case 0:

comboBox2.Text = "Администратор гостиницы";

break;

default:

comboBox2.Text = "";

break;

}

}

private void ДолжностьComboBox\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void Button4\_Click(object sender, EventArgs e)

{

label8.Visible = false;

логинTextBox.ReadOnly = true;

должностьComboBox.Text = usersDataGridView.Rows[usersDataGridView.CurrentRow.Index].Cells[2].Value.ToString();

groupBox1.Visible = true;

iz = true;

}

}

}

Листинг файла Form3.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 Hotel\_

{

public partial class Form3 : Form

{

int znach = 0;

string login;

public Form3(string logi)

{

InitializeComponent();

login = logi;

label2.Text = "Учётная запись: " + login + "\nДолжность: Администратор гостиницы"; }

private int x = 0; private int y = 0;

private void FrmMain\_MouseDown(object sender, MouseEventArgs e)

{

x = e.X; y = e.Y;

}

private void FrmMain\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

Location = new Point(Location.X + (e.X - x), Location.Y + (e.Y - y));

}

}

private void Form3\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "hotelDataSet.Guests". При необходимости она может быть перемещена или удалена.

this.guestsTableAdapter.Fill(this.hotelDataSet.Guests);

// TODO: данная строка кода позволяет загрузить данные в таблицу "hotelDataSet.Services". При необходимости она может быть перемещена или удалена.

this.servicesTableAdapter.Fill(this.hotelDataSet.Services);

// TODO: данная строка кода позволяет загрузить данные в таблицу "hotelDataSet.Room". При необходимости она может быть перемещена или удалена.

this.roomTableAdapter.Fill(this.hotelDataSet.Room);

// TODO: данная строка кода позволяет загрузить данные в таблицу "hotelDataSet.Comfort". При необходимости она может быть перемещена или удалена.

this.comfortTableAdapter.Fill(this.hotelDataSet.Comfort);

this.accommodationTableAdapter.Fill(this.hotelDataSet.Accommodation);

this.guestsTableAdapter.Fill(this.hotelDataSet.Guests);

this.roomTableAdapter.Fill(this.hotelDataSet.Room);

this.servicesTableAdapter.Fill(this.hotelDataSet.Services);

this.comfortTableAdapter.Fill(this.hotelDataSet.Comfort);

}

private void Button3\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void Button2\_Click(object sender, EventArgs e)

{

Application.Exit();

}

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

{

Form4 frm4 = new Form4(login); this.Close();

frm4.Show();

}

private void Button10\_Click(object sender, EventArgs e)

{

Form5 fr = new Form5(login);this.Close();

fr.Show();

}

private void Button4\_Click(object sender, EventArgs e)

{

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection connect = new SqlConnection(myConnectionString);

string sql = "Update [Accommodation] set [Статус]=@Статус WHERE [Код проживания]=@Код";

SqlCommand cmd\_SQL = new SqlCommand(sql, connect);

cmd\_SQL.Parameters.AddWithValue("@Код", textBox2.Text);

cmd\_SQL.Parameters.AddWithValue("@Статус", "Проживание");

try

{

connect.Open();

int n = cmd\_SQL.ExecuteNonQuery();

}

catch (SqlException ex)

{

}

finally

{

connect.Close();

this.accommodationTableAdapter.Fill(this.hotelDataSet.Accommodation);

}

}

private void TextBox2\_TextChanged(object sender, EventArgs e)

{

if ((znach == 0) && (textBox2.Text != ""))

{

for (int i = 0; i < accommodationDataGridView.RowCount; i++)

{

accommodationDataGridView.Rows[i].Selected = false;

for (int j = 0; j < accommodationDataGridView.ColumnCount; j++)

if (accommodationDataGridView.Rows[i].Cells[j].Value != null)

if (accommodationDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox2.Text))

{

accommodationDataGridView.Rows[i].Selected = true;

}

}

}

if ((znach == 1) && (textBox2.Text != ""))

{

for (int i = 0; i < roomDataGridView.RowCount; i++)

{

roomDataGridView.Rows[i].Selected = false;

for (int j = 0; j < roomDataGridView.ColumnCount; j++)

if (roomDataGridView.Rows[i].Cells[j].Value != null)

if (roomDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox2.Text))

{

roomDataGridView.Rows[i].Selected = true;

}

}

}

if ((znach == 2) && (textBox2.Text != ""))

{

for (int i = 0; i < comfortDataGridView.RowCount; i++)

{

comfortDataGridView.Rows[i].Selected = false;

for (int j = 0; j < comfortDataGridView.ColumnCount; j++)

if (comfortDataGridView.Rows[i].Cells[j].Value != null)

if (comfortDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox2.Text))

{

comfortDataGridView.Rows[i].Selected = true;

}

}

}

if ((znach == 3) && (textBox2.Text != ""))

{

for (int i = 0; i < guestsDataGridView.RowCount; i++)

{

guestsDataGridView.Rows[i].Selected = false;

for (int j = 0; j < guestsDataGridView.ColumnCount; j++)

if (guestsDataGridView.Rows[i].Cells[j].Value != null)

if (guestsDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox2.Text))

{

guestsDataGridView.Rows[i].Selected = true;

}

}

}

if ((znach == 4) && (textBox2.Text != ""))

{

for (int i = 0; i < servicesDataGridView.RowCount; i++)

{

servicesDataGridView.Rows[i].Selected = false;

for (int j = 0; j < servicesDataGridView.ColumnCount; j++)

if (servicesDataGridView.Rows[i].Cells[j].Value != null)

if (servicesDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox2.Text))

{

servicesDataGridView.Rows[i].Selected = true;

}

}

}

if (textBox2.Text == "")

{

accommodationDataGridView.ClearSelection();

guestsDataGridView.ClearSelection();

roomDataGridView.ClearSelection();

comfortDataGridView.ClearSelection();

servicesDataGridView.ClearSelection();

}

}

private void Button11\_Click(object sender, EventArgs e)

{

Form6 fr = new Form6(login);this.Close();

fr.Show();

}

private void Button6\_Click(object sender, EventArgs e)

{

znach = 1;

accommodationDataGridView.Visible = false;

guestsDataGridView.Visible = false;

roomDataGridView.Visible = true;

comfortDataGridView.Visible = false;

servicesDataGridView.Visible = false;

}

private void Button9\_Click(object sender, EventArgs e)

{

znach = 0;

accommodationDataGridView.Visible = true;

guestsDataGridView.Visible = false;

roomDataGridView.Visible = false;

comfortDataGridView.Visible = false;

servicesDataGridView.Visible = false;

}

private void Button5\_Click(object sender, EventArgs e)

{

znach = 3;

accommodationDataGridView.Visible = false;

guestsDataGridView.Visible = true;

roomDataGridView.Visible = false;

comfortDataGridView.Visible = false;

servicesDataGridView.Visible = false;

}

private void Button7\_Click(object sender, EventArgs e)

{

znach = 2;

accommodationDataGridView.Visible = false;

guestsDataGridView.Visible = false;

roomDataGridView.Visible = false;

comfortDataGridView.Visible = true;

servicesDataGridView.Visible = false;

}

private void Button8\_Click(object sender, EventArgs e)

{

znach = 4;

accommodationDataGridView.Visible = false;

guestsDataGridView.Visible = false;

roomDataGridView.Visible = false;

comfortDataGridView.Visible = false;

servicesDataGridView.Visible = true;

}

private void Button12\_Click(object sender, EventArgs e)

{

Form1 frm = new Form1();

frm.Show(); this.Close();

}

}

}

Листинг файла 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 Hotel\_

{

public partial class Form4 : Form

{

int znach = 0;

string login;

public Form4(string logi)

{

login = logi;

InitializeComponent();

}

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

{

this.Validate();

this.accommodationBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.hotelDataSet);

}

private int x = 0; private int y = 0;

private void FrmMain\_MouseDown(object sender, MouseEventArgs e)

{

x = e.X; y = e.Y;

}

private void FrmMain\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

Location = new Point(Location.X + (e.X - x), Location.Y + (e.Y - y));

}

}

private void Form4\_Load(object sender, EventArgs e)

{

this.accommodationTableAdapter.Fill(this.hotelDataSet.Accommodation);

this.guestsTableAdapter.Fill(this.hotelDataSet.Guests);

this.roomTableAdapter.Fill(this.hotelDataSet.Room);

this.servicesTableAdapter.Fill(this.hotelDataSet.Services);

this.comfortTableAdapter.Fill(this.hotelDataSet.Comfort);

}

static bool Search()

{

bool b = false;

return b;

}

private void Button8\_Click(object sender, EventArgs e)

{

bool b = false;

groupBox3.Visible = true;

if (код\_гостяTextBox.Text != "")

{

for (int i = 0; i < guestsDataGridView.RowCount; i++)

{

guestsDataGridView.Rows[i].Selected = false;

int j = 0;

if (guestsDataGridView.Rows[i].Cells[j].Value != null)

if ((guestsDataGridView.Rows[i].Cells[j].Value.ToString().Contains(код\_гостяTextBox.Text)) && (b != true))

{

guestsDataGridView.CurrentCell = guestsDataGridView.Rows[i].Cells[j];

groupBox3.Visible = true; b = true;

button13.Visible = false; button9.Visible = false;

}

}

}

if (b == false)

{

guestsBindingSource.AddNew();

код\_гостяTextBox1.ReadOnly = false;

код\_гостяTextBox1.Text = Convert.ToString(guestsDataGridView.RowCount - 1);

код\_гостяTextBox.Text = код\_гостяTextBox1.Text;

код\_гостяTextBox1.ReadOnly = true;

groupBox3.Visible = true;

}

}

string pol = "-";

private void Button9\_Click(object sender, EventArgs e)

{

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection connect = new SqlConnection(myConnectionString);

string sql = "INSERT INTO [Guests] ([Код гостя], [ФИО], [Пол], [Документ], [Серия, Номер], [Гражданство], [Дата рождения], [Контакты]) VALUES (@Код, @ФИО, @Пол,@Документ,@Серия,@Гражданство,@Дата,@Контакты);";

SqlCommand cmd\_SQL = new SqlCommand(sql, connect);

cmd\_SQL.Parameters.AddWithValue("@Код", код\_гостяTextBox1.Text);

cmd\_SQL.Parameters.AddWithValue("@ФИО", фИОTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Пол", pol);

cmd\_SQL.Parameters.AddWithValue("@Документ", документTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Серия", серия\_\_НомерTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Гражданство", гражданствоTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Дата", дата\_рожденияDateTimePicker.Value);

cmd\_SQL.Parameters.AddWithValue("@Контакты", контактыTextBox.Text);

try

{

connect.Open();

int n = cmd\_SQL.ExecuteNonQuery();

}

catch (SqlException ex)

{

}

finally

{

connect.Close();

}

Validate();

this.guestsTableAdapter.Fill(this.hotelDataSet.Guests);

groupBox3.Visible = false;

}

private void ПолComboBox\_SelectedIndexChanged(object sender, EventArgs e)

{

switch (полComboBox.SelectedIndex)

{

case 0:

pol = "Мужской";

break;

case 1:

pol = "Женский";

break;

default:

pol = "-";

break;

}

}

private void Button13\_Click(object sender, EventArgs e)

{

guestsBindingSource.RemoveAt(guestsDataGridView.RowCount - 2);

Validate();

guestsBindingSource.EndEdit();

guestsTableAdapter.Update(hotelDataSet.Guests);

groupBox3.Visible = false;

}

private void Код\_номераTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && number != 44 && number != 46)

{

e.Handled = true;

}

}

private void СуммаTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && number != 44 && number != 46)

{

e.Handled = true;

}

}

private void ОплатаTextBox\_KeyPress(object sender, KeyPressEventArgs e)

{

char number = e.KeyChar;

if ((e.KeyChar <= 47 || e.KeyChar >= 58) && number != 8 && number != 44 && number != 46)

{

e.Handled = true;

}

}

private void Button10\_Click(object sender, EventArgs e)

{

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection connect = new SqlConnection(myConnectionString); string sql = "INSERT INTO [Accommodation] ([Код проживания], [Код гостя], [Дата прибытия], [Дата выбытия], [Код номера], [Доп услуги], [Сумма], [Оплата], [Статус]) VALUES (@Кодп, @Кодг, @Датап,@Датав, @Кодн, @Доп,@Сумма,@Оплата, @Статус);";

SqlCommand cmd\_SQL = new SqlCommand(sql, connect);

код\_проживанияTextBox.ReadOnly = false;

string kod = код\_проживанияTextBox.Text;

код\_проживанияTextBox.ReadOnly = true;

cmd\_SQL.Parameters.AddWithValue("@Кодп", Convert.ToInt32(kod));

cmd\_SQL.Parameters.AddWithValue("@Кодг", Convert.ToInt32(код\_гостяTextBox.Text));

cmd\_SQL.Parameters.AddWithValue("@Датап", дата\_прибытияDateTimePicker.Value);

cmd\_SQL.Parameters.AddWithValue("@Датав", дата\_выбытияDateTimePicker.Value);

cmd\_SQL.Parameters.AddWithValue("@Кодн", Convert.ToInt32(код\_номераTextBox.Text));

cmd\_SQL.Parameters.AddWithValue("@Доп", доп\_услугиTextBox.Text);

cmd\_SQL.Parameters.AddWithValue("@Сумма", Convert.ToDouble(суммаTextBox.Text));

cmd\_SQL.Parameters.AddWithValue("@Оплата", Convert.ToDouble(оплатаTextBox.Text));

cmd\_SQL.Parameters.AddWithValue("@Статус", статусTextBox.Text);

try

{

connect.Open();

int n = cmd\_SQL.ExecuteNonQuery();

this.accommodationTableAdapter.Fill(this.hotelDataSet.Accommodation);

this.Width = 747;

this.Height = 632;

button14.Visible = true;

button15.Visible = true;

}

catch (SqlException ex)

{

}

finally

{

connect.Close();

}

}

private void Button11\_Click(object sender, EventArgs e)

{

this.Width = 747;

this.Height = 632;

button14.Visible = true;

button15.Visible = true;

this.accommodationTableAdapter.Fill(this.hotelDataSet.Accommodation);

}

private void Button4\_Click(object sender, EventArgs e)

{

znach = 0;

accommodationDataGridView.Visible = true;

guestsDataGridView.Visible = false;

roomDataGridView.Visible = false;

comfortDataGridView.Visible = false;

servicesDataGridView.Visible = false;

}

private void Button5\_Click(object sender, EventArgs e)

{

znach = 1;

accommodationDataGridView.Visible = false;

guestsDataGridView.Visible = false;

roomDataGridView.Visible = true;

comfortDataGridView.Visible = false;

servicesDataGridView.Visible = false;

}

private void Button6\_Click(object sender, EventArgs e)

{

znach = 2;

accommodationDataGridView.Visible = false;

guestsDataGridView.Visible = false;

roomDataGridView.Visible = false;

comfortDataGridView.Visible = true;

servicesDataGridView.Visible = false;

}

private void Button7\_Click(object sender, EventArgs e)

{

znach = 3;

accommodationDataGridView.Visible = false;

guestsDataGridView.Visible = true;

roomDataGridView.Visible = false;

comfortDataGridView.Visible = false;

servicesDataGridView.Visible = false;

}

private void Button12\_Click(object sender, EventArgs e)

{

znach = 4;

accommodationDataGridView.Visible = false;

guestsDataGridView.Visible = false;

roomDataGridView.Visible = false;

comfortDataGridView.Visible = false;

servicesDataGridView.Visible = true;

}

private void GroupBox2\_Enter(object sender, EventArgs e)

{

}

private void Button3\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void Button2\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void Button16\_Click(object sender, EventArgs e)

{

this.Close(); Form3 fr = new Form3(login);

fr.Show();

}

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

{ this.Width =1076;

this.Height =632;

button14.Visible = false;

button15.Visible = false;

accommodationBindingSource.AddNew();

дата\_прибытияDateTimePicker.MinDate = DateTime.Today.Date;

дата\_выбытияDateTimePicker.MinDate = дата\_прибытияDateTimePicker.Value;

статусTextBox.Text = "Бронь";

статусTextBox.ReadOnly = true;

код\_проживанияTextBox.ReadOnly = false;

код\_проживанияTextBox.Text = Convert.ToString(accommodationDataGridView.RowCount - 1);

код\_проживанияTextBox.ReadOnly = true;

groupBox1.Visible = true;

}

private void Дата\_выбытияDateTimePicker\_ValueChanged(object sender, EventArgs e)

{

дата\_выбытияDateTimePicker.MinDate = дата\_прибытияDateTimePicker.Value;

}

private void Дата\_прибытияDateTimePicker\_ValueChanged(object sender, EventArgs e)

{

дата\_выбытияDateTimePicker.MinDate = дата\_прибытияDateTimePicker.Value;

}

private void Button16\_Click\_1(object sender, EventArgs e)

{

this.Close(); Form3 fr = new Form3(login);

fr.Show();

}

private void TextBox1\_TextChanged(object sender, EventArgs e)

{

if ((znach == 0) && (textBox1.Text != ""))

{

for (int i = 0; i < accommodationDataGridView.RowCount; i++)

{

accommodationDataGridView.Rows[i].Selected = false;

for (int j = 0; j < accommodationDataGridView.ColumnCount; j++)

if (accommodationDataGridView.Rows[i].Cells[j].Value != null)

if (accommodationDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox1.Text))

{

accommodationDataGridView.Rows[i].Selected = true;

}

}

}

else if ((znach == 1) && (textBox1.Text != ""))

{

for (int i = 0; i < roomDataGridView.RowCount; i++)

{

roomDataGridView.Rows[i].Selected = false;

for (int j = 0; j < roomDataGridView.ColumnCount; j++)

if (roomDataGridView.Rows[i].Cells[j].Value != null)

if (roomDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox1.Text))

{

roomDataGridView.Rows[i].Selected = true;

}

}

}

else if ((znach == 2) && (textBox1.Text != ""))

{

for (int i = 0; i < comfortDataGridView.RowCount; i++)

{

comfortDataGridView.Rows[i].Selected = false;

for (int j = 0; j < comfortDataGridView.ColumnCount; j++)

if (comfortDataGridView.Rows[i].Cells[j].Value != null)

if (comfortDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox1.Text))

{

comfortDataGridView.Rows[i].Selected = true;

}

}

}

else if ((znach == 3) && (textBox1.Text != ""))

{

for (int i = 0; i < guestsDataGridView.RowCount; i++)

{

guestsDataGridView.Rows[i].Selected = false;

for (int j = 0; j < guestsDataGridView.ColumnCount; j++)

if (guestsDataGridView.Rows[i].Cells[j].Value != null)

if (guestsDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox1.Text))

{

guestsDataGridView.Rows[i].Selected = true;

}

}

}

else if ((znach == 4) && (textBox1.Text != ""))

{

for (int i = 0; i < servicesDataGridView.RowCount; i++)

{

servicesDataGridView.Rows[i].Selected = false;

for (int j = 0; j < servicesDataGridView.ColumnCount; j++)

if (servicesDataGridView.Rows[i].Cells[j].Value != null)

if (servicesDataGridView.Rows[i].Cells[j].Value.ToString().Contains(textBox1.Text))

{

servicesDataGridView.Rows[i].Selected = true;

}

}

}

else if (textBox1.Text == "")

{

accommodationDataGridView.ClearSelection();

guestsDataGridView.ClearSelection();

roomDataGridView.ClearSelection();

comfortDataGridView.ClearSelection();

servicesDataGridView.ClearSelection();

}

}

private void GroupBox2\_Enter\_1(object sender, EventArgs e)

{

}

}

}

Листинг файла Form5.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 Hotel\_

{

public partial class Form5 : Form

{

string login;

public Form5(string logi)

{

login = logi;

InitializeComponent(); LoadData();

}

private int x = 0; private int y = 0;

private void FrmMain\_MouseDown(object sender, MouseEventArgs e)

{

x = e.X; y = e.Y;

}

private void FrmMain\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

Location = new Point(Location.X + (e.X - x), Location.Y + (e.Y - y));

}

}

private void LoadData()

{

dataGridView1.Rows.Clear();

string path = Application.StartupPath + @"\Hotel.mdf";

string connectString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection myConnection = new SqlConnection(connectString);

myConnection.Open();

string query = "SELECT Accommodation.[Код гостя], Accommodation.[Код проживания], Guests.[ФИО], Accommodation.[Статус], Accommodation.[Код номера], Guests.[Контакты], Accommodation.[Дата прибытия], Accommodation.[Дата выбытия] FROM Accommodation, Guests WHERE Accommodation.[Код гостя] = Guests.[Код гостя] AND Accommodation.[Статус] LIKE N'П%' ";

SqlCommand command = new SqlCommand(query, myConnection);

SqlDataReader reader = command.ExecuteReader();

List<string[]> data = new List<string[]>();

while (reader.Read())

{

data.Add(new string[8]);

data[data.Count - 1][0] = reader[0].ToString();

data[data.Count - 1][1] = reader[1].ToString();

data[data.Count - 1][2] = reader[2].ToString();

data[data.Count - 1][3] = reader[3].ToString();

data[data.Count - 1][4] = reader[4].ToString();

data[data.Count - 1][5] = reader[5].ToString();

data[data.Count - 1][6] = reader[6].ToString();

data[data.Count - 1][7] = reader[7].ToString();

}

reader.Close();

myConnection.Close();

foreach (string[] s in data)

dataGridView1.Rows.Add(s);

}

private void Form5\_Load(object sender, EventArgs e)

{

}

private void LoadDataF()

{

dataGridView1.Rows.Clear();

string path = Application.StartupPath + @"\Hotel.mdf";

string connectString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection myConnection = new SqlConnection(connectString);

myConnection.Open();

string query = "SELECT Accommodation.[Код гостя], Accommodation.[Код проживания], Guests.[ФИО], Accommodation.[Статус], Accommodation.[Код номера], Guests.[Контакты], Accommodation.[Дата прибытия], Accommodation.[Дата выбытия] FROM Accommodation, Guests WHERE Accommodation.[Код гостя] = Guests.[Код гостя] AND Accommodation.[Статус] LIKE N'П%' AND Accommodation.[Дата выбытия] = CONVERT (date, SYSDATETIME())";

SqlCommand command = new SqlCommand(query, myConnection);

SqlDataReader reader = command.ExecuteReader();

List<string[]> data = new List<string[]>();

while (reader.Read())

{

data.Add(new string[8]);

data[data.Count - 1][0] = reader[0].ToString();

data[data.Count - 1][1] = reader[1].ToString();

data[data.Count - 1][2] = reader[2].ToString();

data[data.Count - 1][3] = reader[3].ToString();

data[data.Count - 1][4] = reader[4].ToString();

data[data.Count - 1][5] = reader[5].ToString();

data[data.Count - 1][6] = reader[6].ToString();

data[data.Count - 1][7] = reader[7].ToString();

}

reader.Close();

myConnection.Close();

foreach (string[] s in data)

dataGridView1.Rows.Add(s);

}

private void LoadDataС()

{

dataGridView1.Rows.Clear();

string path = Application.StartupPath + @"\Hotel.mdf";

string connectString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection myConnection = new SqlConnection(connectString);

myConnection.Open();

string query = "SELECT Accommodation.[Код гостя], Accommodation.[Код проживания], Guests.[ФИО], Accommodation.[Статус], Accommodation.[Код номера], Guests.[Контакты], Accommodation.[Дата прибытия], Accommodation.[Дата выбытия] FROM Accommodation, Guests WHERE Accommodation.[Код гостя] = Guests.[Код гостя] AND Accommodation.[Статус] LIKE N'Б%'";

SqlCommand command = new SqlCommand(query, myConnection);

SqlDataReader reader = command.ExecuteReader();

List<string[]> data = new List<string[]>();

while (reader.Read())

{

data.Add(new string[8]);

data[data.Count - 1][0] = reader[0].ToString();

data[data.Count - 1][1] = reader[1].ToString();

data[data.Count - 1][2] = reader[2].ToString();

data[data.Count - 1][3] = reader[3].ToString();

data[data.Count - 1][4] = reader[4].ToString();

data[data.Count - 1][5] = reader[5].ToString();

data[data.Count - 1][6] = reader[6].ToString();

data[data.Count - 1][7] = reader[7].ToString();

}

reader.Close();

myConnection.Close();

foreach (string[] s in data)

dataGridView1.Rows.Add(s);

}

private void Button14\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void Button15\_Click(object sender, EventArgs e)

{

Application.Exit();

}

int vib = 0;

private void ComboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

switch (comboBox2.SelectedIndex)

{

case 0:

vib = 0;

break;

case 1:

vib = 1;

break;

case 2:

vib = 2;

break;

default:

vib = 0;

break;

}

}

private void Button6\_Click(object sender, EventArgs e)

{

if (vib == 0) { LoadData(); }

else if (vib == 1) { LoadDataF(); }

else if (vib == 2) { LoadDataС(); }

}

private void TextBox1\_TextChanged(object sender, EventArgs e)

{

if (textBox1.Text != "")

{

for (int i = 0; i < dataGridView1.RowCount; i++)

{

dataGridView1.Rows[i].Selected = false;

for (int j = 0; j < dataGridView1.ColumnCount; j++)

if (dataGridView1.Rows[i].Cells[j].Value != null)

if (dataGridView1.Rows[i].Cells[j].Value.ToString().Contains(textBox1.Text))

{

dataGridView1.Rows[i].Selected = true;

}

}

}

else if (textBox1.Text == "") dataGridView1.ClearSelection();

}

private void Label7\_Click(object sender, EventArgs e)

{

}

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

{

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection connect = new SqlConnection(myConnectionString);

string sql = "Update [Accommodation] set [Статус]=@Статус WHERE [Код проживания]=@Код";

SqlCommand cmd\_SQL = new SqlCommand(sql, connect);

cmd\_SQL.Parameters.AddWithValue("@Код", textBox2.Text);

cmd\_SQL.Parameters.AddWithValue("@Статус", "Закрыто");

try

{

connect.Open();

int n = cmd\_SQL.ExecuteNonQuery();

LoadData();

}

catch (SqlException ex)

{

}

finally

{

connect.Close();

}

}

private void Button16\_Click(object sender, EventArgs e)

{

this.Close(); Form3 fr = new Form3(login);

fr.Show();

}

}

}

Листинг файла Form6.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 Hotel\_

{

public partial class Form6 : Form

{ string login;

public Form6(string logi)

{

InitializeComponent();

login = logi;

LoadData();

}

private void LoadDataF()

{

dataGridView1.Rows.Clear();

string path = Application.StartupPath + @"\Hotel.mdf";

string connectString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection myConnection = new SqlConnection(connectString);

myConnection.Open();

string query = "SELECT \* FROM Room WHERE Room.[Код номера] NOT IN (SELECT Accommodation.[Код номера] FROM Accommodation WHERE Accommodation.[Дата прибытия] BETWEEN '" + dateTimePicker1.Value.ToString("yyyy-MM-dd") + "' AND '" + dateTimePicker2.Value.ToString("yyyy-MM-dd") +"' ) ";

//SELECT \*FROM Call WHERE phone\_number NOT IN(SELECT phone\_number FROM Phone\_book)

//CONVERT (date, SYSDATETIME()).

SqlCommand command = new SqlCommand(query, myConnection);

SqlDataReader reader = command.ExecuteReader();

List<string[]> data = new List<string[]>();

while (reader.Read())

{

data.Add(new string[5]);

data[data.Count - 1][0] = reader[0].ToString();

data[data.Count - 1][1] = reader[1].ToString();

data[data.Count - 1][2] = reader[2].ToString();

data[data.Count - 1][3] = reader[3].ToString();

data[data.Count - 1][4] = reader[4].ToString();

}

reader.Close();

myConnection.Close();

foreach (string[] s in data)

dataGridView1.Rows.Add(s);

}

private void LoadData()

{

dataGridView1.Rows.Clear();

string path = Application.StartupPath + @"\Hotel.mdf";

string connectString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection myConnection = new SqlConnection(connectString);

myConnection.Open();

string query = "SELECT \* FROM Room";

SqlCommand command = new SqlCommand(query, myConnection);

SqlDataReader reader = command.ExecuteReader();

List<string[]> data = new List<string[]>();

while (reader.Read())

{

data.Add(new string[5]);

data[data.Count - 1][0] = reader[0].ToString();

data[data.Count - 1][1] = reader[1].ToString();

data[data.Count - 1][2] = reader[2].ToString();

data[data.Count - 1][3] = reader[3].ToString();

data[data.Count - 1][4] = reader[4].ToString();

}

reader.Close();

myConnection.Close();

foreach (string[] s in data)

dataGridView1.Rows.Add(s);

}

private int x = 0; private int y = 0;

private void FrmMain\_MouseDown(object sender, MouseEventArgs e)

{

x = e.X; y = e.Y;

}

private void FrmMain\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

Location = new Point(Location.X + (e.X - x), Location.Y + (e.Y - y));

}

}

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

{

this.Validate();

this.roomBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.hotelDataSet);

}

private void Form6\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "hotelDataSet.Room". При необходимости она может быть перемещена или удалена.

this.roomTableAdapter.Fill(this.hotelDataSet.Room);

dateTimePicker1.MinDate= DateTime.Today.Date;

dateTimePicker2.MinDate = DateTime.Today.Date;

}

private void DateTimePicker1\_ValueChanged(object sender, EventArgs e)

{

dateTimePicker2.MinDate = dateTimePicker1.Value;

}

private void DateTimePicker2\_ValueChanged(object sender, EventArgs e)

{

dateTimePicker2.MinDate = dateTimePicker1.Value;

}

private void Button2\_Click(object sender, EventArgs e)

{

LoadDataF();

}

private void Button3\_Click(object sender, EventArgs e)

{

LoadData();

}

private void Button16\_Click(object sender, EventArgs e)

{

this.Close(); Form3 fr = new Form3(login);

fr.Show();

}

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

{

string path = Application.StartupPath + @"\Hotel.mdf";

string myConnectionString = @"Data Source = (LocalDB)\MSSQLLocalDB; AttachDbFilename = '" + path + "'; Integrated Security = True";

SqlConnection connect = new SqlConnection(myConnectionString); string sql = "INSERT INTO [Room] ([Код номера], [Вместимость], [Класс], [Стоимость], [Тип удобства]) VALUES (@Код, @Вместимость, @Класс,@Стоимость, @Тип);";

SqlCommand cmd\_SQL = new SqlCommand(sql, connect);

cmd\_SQL.Parameters.AddWithValue("@Код", textBox2.Text);

cmd\_SQL.Parameters.AddWithValue("@Вместимость", numericUpDown1.Value);

cmd\_SQL.Parameters.AddWithValue("@Класс", ti);

cmd\_SQL.Parameters.AddWithValue("@Стоимость", textBox3.Text);

cmd\_SQL.Parameters.AddWithValue("@Тип", textBox4.Text);

try

{

connect.Open();

int n = cmd\_SQL.ExecuteNonQuery();

LoadData();

}

catch (SqlException ex)

{

}

finally

{

connect.Close();

}

}

string ti;

private void ComboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

switch (comboBox1.SelectedIndex)

{

case 0:

ti = "Double";

break;

case 1:

ti = "Quadriple";

break;

case 2:

ti = "Twin";

break;

case 3:

ti = "Triple";

break;

case 4:

ti = "Duplex";

break;

case 5:

ti = "Apartment";

break;

case 6:

ti = "Sibgle";

break;

case 7:

ti = "Executive";

break;

default:

ti = comboBox1.Text;

break;

}

}

private void Button14\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void Button15\_Click(object sender, EventArgs e)

{

Application.Exit();

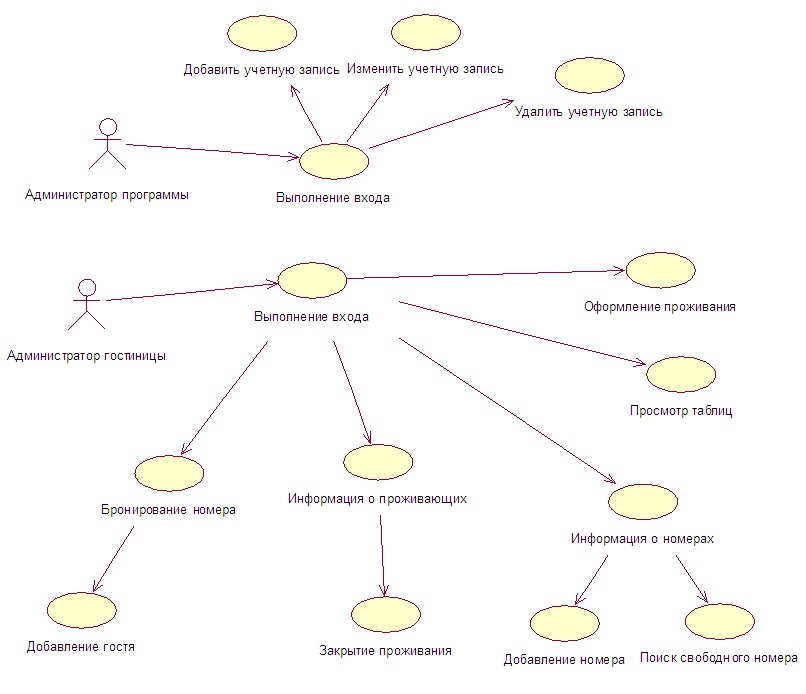
}

}

}

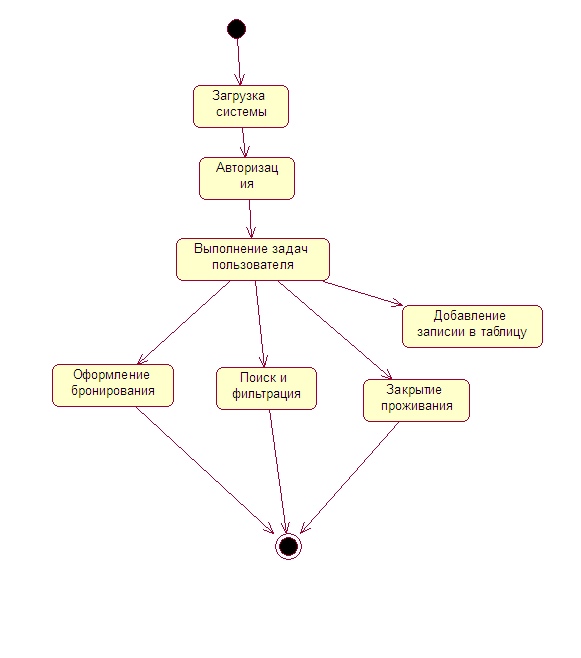
# Приложение Б (обязательное)

Диаграмма вариантов использования



# Приложение В (обязательное)

Диаграмма состояний



# Приложение Г (обязательное)

Функциональная диаграмма

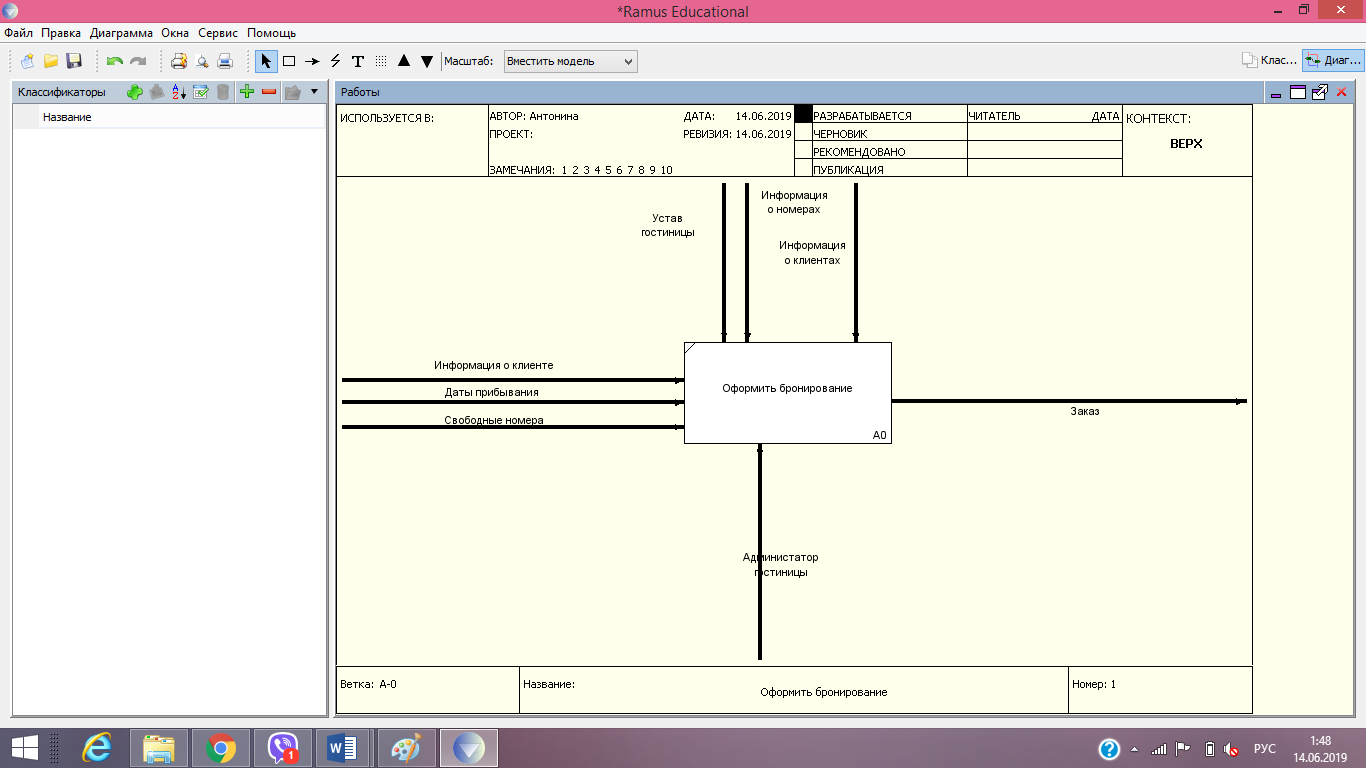


Рисунок Г1 – Контекстная диаграмма

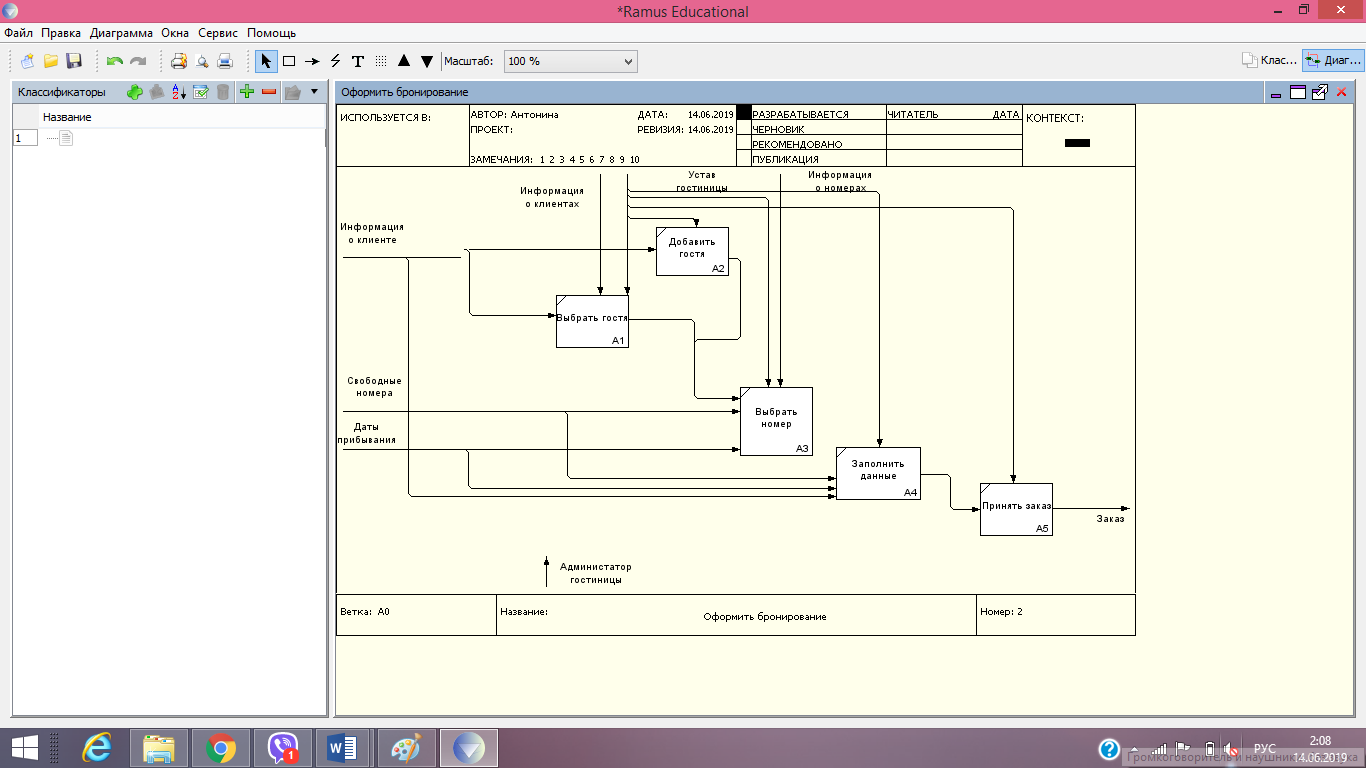
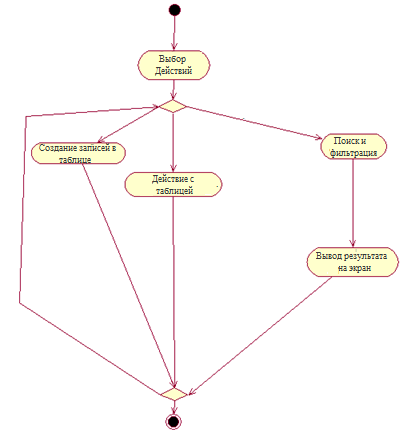


Рисунок Г2 - Декомпозиция первого уровня функциональной модели

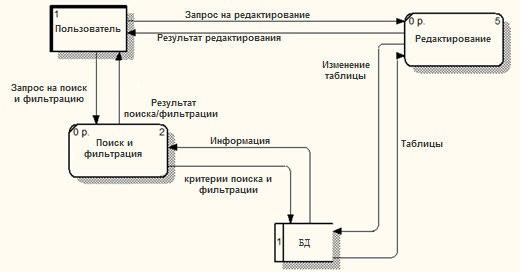
# Приложение Д (обязательное)

Диаграмма деятельности



# Приложение Е (обязательное)

# Диаграмма потоков данных



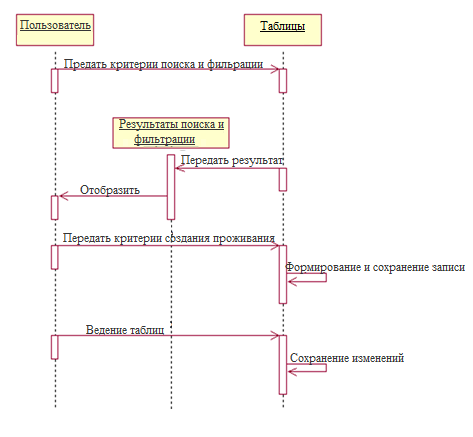
# Приложение Ж (обязательное)

Диаграмма «сущность-связь»



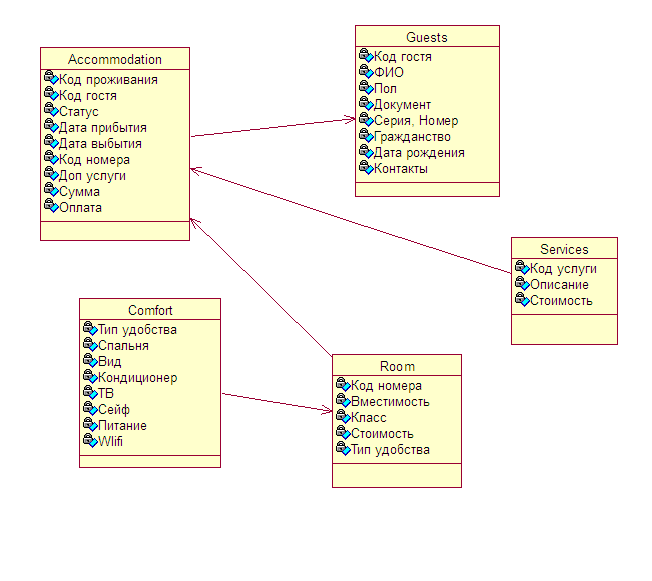
# Приложение К (обязательное)

# Диаграмма последовательности



# Приложение Л (обязательное)

Диаграмма классов



# Приложение М

Техническое задание

# Общие сведения

# Формулировка задания

Разработать программный продукт (ПП), предназначенной для автоматизации процесса работы администратора гостиницы.

ПП должен обеспечивать бронирования мест в гостинице и оформления проживающих в гостинице.

## Категории пользователей

ПП должен быть рассчитан на работу всех лиц, имеющих доступ к программному обеспечению.

## Наименование организации заказчика

Заказчиком на разработку данного ПП является УО «Минский государственный колледж электроники». (220000 г. Минск, ул. Казинца, 91).

## Основания для проведения работ

Основанием для разработки данного ПП является договор с заказчиком.

# Описание предметной области

## Описание процессов

В Республике Беларусь, функционирует большое количество гостиниц и очень важен процесс автоматизации процессов бронирования и оформления проживания гостей в гостинице.

ПП должен осуществлять:

* обработку запросов на поиск информации по следующим таблицам: гости, проживание, номера, удобства, доп. услуги;
* предоставлять отфильтрованную информацию пользователям.
* осуществлять бронирование номеров
* осуществлять оформление проживания.

## Состав данных и алгоритмы обработки информации

Входными данными для разработанного ПП являются:

* информация о клиенте;
* информация о проживании и брони номера;
* параметры фильтра и поисковых запросов;
* информация вводимая при внесении изменений в проживание;
* информация о новом номера.

Выходными данными разработанного ПП являются:

* информация отображаемая на экран по запросу пользователя;
* результаты поиска и фильтрации данных.

Постоянными данными является база данных.

## Недостатки существующих проектных решений

Множество аналогов данного приложения, и данные аналоги установлены в разных гостиницах. Заказчику они не подходят из-за слишком большого функционала, большого размера и множества ненужных функций.

## Текущий уровень автоматизации

На данный момент заказчик не располагает каким-либо ПП, обеспечивающим полноценную автоматизацию всей его деятельности.

# Требования к разработке

## Информационная модель

Информационная модель представлена на рисунке 3.1.1

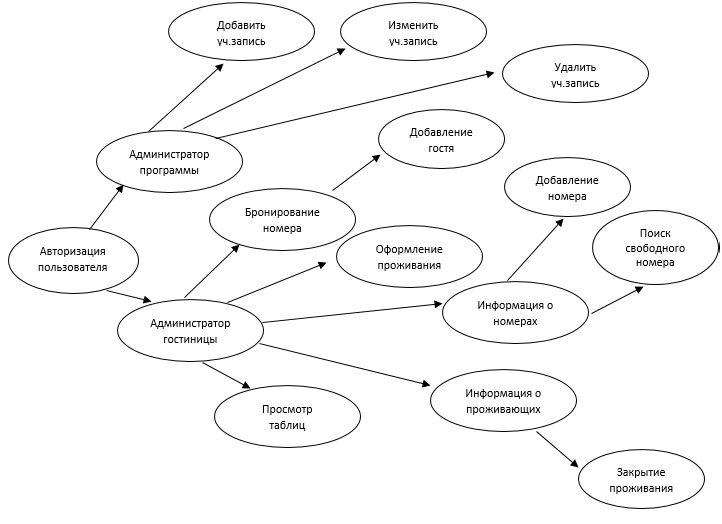


Рисунок 3.1.1 Информационная модель

## Структура меню

Структура меню представлена интерфейсов пользователя, а также группами кнопок в программе.

## Функциональные требования

1) Бронирование номеров

Функция предназначена для:

-Создания, изменения статуса проживания, заполнения соответствующих таблиц.

2) Фильтрация отображаемых данных.

- Предназначена для фильтрации отображаемых таблицей данных.

3) Поиск.

-Поиск по заданным таблицам.

## Требования к информационному обеспечению

Разрабатываемый ПП должен:

- сохранять данные, вводимые пользователем;

- предоставлять удобный доступ к файлам пользователя;

- позволять вносить данные;

## Требования к пользовательскому интерфейсу

Разрабатываемый ПП должен быть обеспечен упрощенной формой интерфейса:

- глубина вложенности окон должна быть не более четырёх;

- содержать краткие описания элементов меню;

- должен быть понятным и простым в использовании.

Интерфейс должен содержать в себе все необходимые функции для работы заказчика. При необходимости интерфейс изменяется только разработчиком.

## Требования к алгоритмам

Главным требованием к разрабатываемым алгоритмам является то, что они должны легко модифицироваться. В течение сроков, установленных в договоре между заказчиком и разработчиком в ПП могут быть внесены изменения, не требующие глобальной переработки алгоритмов.

## Прочие требования

Разрабатываемый ПП должен функционировать в ОС Win 7, 8.1, 10.

# Порядок контроля и обеспечения качества

## Экспертиза

Не требуется разработка макета на стадии «Технический проект». Требуется проведение экспертизы при участии представителя Заказчика.

## Тестирование

Тестирование должно проводиться разработчиком в соответствии с установленными правилами. Присутствие представителя заказчика необязательно.

Тестирование проводится разработчиком по предложенной схеме заказчика:

1) создается БД небольшого объема по всем таблицам и пунктам меню;

2) в течение срока, указанного в договоре, разработчик проводит тестирование ПП и передает заказчику готовый программный продукт, который будет сопровождаться разработчиком в течении сроков, предусмотренных договором.

## Опытная эксплуатация

Опытная эксплуатация проводится заказчиком, при контроле разработчика, в соответствии с договором.

# Требования к документированию

## Требования к справочной системе

Не требуется наличие справочной системы

## Требования к документации пользователя

Разрабатываемый ПП должен сопровождаться полным пакетом конструкторской и эксплуатационной документации, перечисленной в договоре.

Разработанный ПП сопровождается полным пакетом документов, предусмотренным ГОСТ 7685 – 2000, а именно:

1) лицензионное право использования разработанного ПП;

2) документация по эксплуатации;

3) талон на последующее обслуживание ПП;

4) документы, подтверждающие монопольное использование разработанного ПП. Размещено на Allbest.ru