Содержание

Введение

|  |  |
| --- | --- |
| 1 Назначение и цели разработки | 4 |
| 2 Разработка технического проекта на основе анализа требований | 5 |
| 2.1 Определение спецификаций программного обеспечения | 5 |
| 2.2 Проектирование модели данных и диаграммы классов | 11 |
| 2.3 Конструирование прототипа | 17 |
| 3 Реализация | 20 |
| 3.1 Обоснование выбора средств разработки | 20 |
| 3.2 Разработка базы данных в среде СУБД | 22 |
| 3.3 Разработка программных модулей | 24 |
| 4 Тестирование программных модулей | 31 |
| 4.1 Модульное тестирование | 31 |
| 4.2 Интеграционное тестирование | 34 |
| 5 Эксплуатационная документация | 35 |
| 5.1 Руководство пользователя | 35 |
| 5.2 Руководство системного программиста | 35 |
| Заключение | 36 |
| Список использованных источников | 38 |
| Приложение А Техническое задание. | 41 |
| Приложение Б Программный код | 49 |
| Приложение В Скрипт базы данных | 63 |
| Приложение Г Тест-кейсы | 75 |
| Приложение Д Руководство пользователя | 78 |
| Приложение Е Руководство системного программиста | 88 |

Введение

С развитием туризма и увеличением числа путешественников, возросла потребность в автоматизации процессов планирования туров и управления клиентскими записями. Туристические агентства сталкиваются с задачами оптимизации маршрутов, эффективного управления клиентскими данными и повышения качества обслуживания. Разработка программных модулей для автоматизации этих процессов становится ключевым инструментом для повышения конкурентоспособности и удовлетворенности клиентов. В целях автоматизации составления туров и записи клиентов на пешие и автомобильные туры создаётся программная система «Турагентство Турист».

Реализация данной системы позволит туристическим компаниям повысить эффективность процессов составления и организации туров, оптимизировать работу с клиентами, а также предоставить туристам более качественный сервис за счет автоматизации рутинных операций.

Актуальность данной темы обусловлена необходимостью повышения конкурентоспособности туристических компаний за счет сокращения трудозатрат и времени на составление туров, а также улучшения качества обслуживания клиентов. Внедрение современных информационных технологий в деятельность турагентств позволяет не только оптимизировать бизнес-процессы, но и предоставлять потребителям более качественные услуги.

1 Назначение и цели разработки

Заданием предусмотрена разработка программных модулей для автоматизации ключевых процессы туристических компаний, связанные с составлением туристических маршрутов, удобной записи на туры клиентам, а также предоставление актуальной информации о доступности туров и их ценах. Эти модули будут использоваться сотрудниками туристической организации для составления туров и клиентами данного приложения для просмотра туров и записи на них.

Целью разработки являются: повышение эффективности процессов составления и организации туристических маршрутов, создание удобного пользовательского интерфейса, а также автоматизация работы с клиентами.

2 Разработка технического проекта на основе анализа требований

2.1 Определение спецификаций программного обеспечения

Рассмотрим определение вариантов использования (прецедентов).

Система требуется прежде всего следующим заинтересованным лицам:

* посетитель;
* клиент;
* администратор.

Основные прецеденты для разрабатываемой системы следующие.

Прецеденты для посетителя:

* П1 – просмотр списка туров;
* П2 – просмотр информации о туре.

Прецеденты для клиента:

* П1 – Авторизация;
* П2 – Регистрация;
* П3 – Оформление тура;
* П4 – Добавление отзыва;
* П5 – Редактирование отзыва;
* П6 – Удаление отзыва.

Прецеденты для администратора:

* П1 – Вход в систему;
* П2 – Составление тура;
* П3 – Удаление тура;
* П4 – Редактирование тура;
* П5 – Добавление в тур места для посещения;
* П6 – Редактирование места для посещения;
* П7 – Удаление места для посещения;
* П8 – Создание места для посещения.

Диаграмма вариантов использования, созданная средствами MS Visio, представлена на рисунке 1.

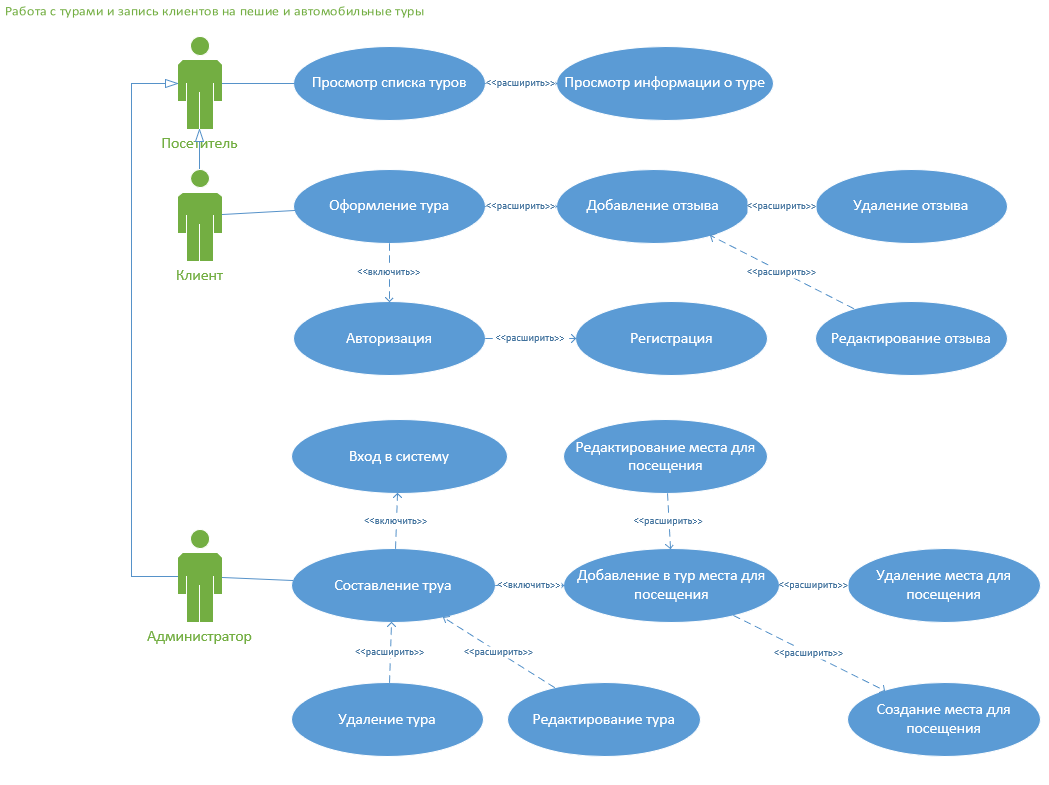


Рисунок 1 – Диаграмма вариантов использования программных модулей

Опишем подробно требования к реализации базового варианта использования, представленных на рисунке 1.

В таблице 1 представлено описание основного раздела сценария варианта использования (прецедента) «Оформление тура».

Таблица 1- Главный раздел сценария варианта использования «Оформление тура»

|  |  |
| --- | --- |
| Вариант использования | Запись клиента |
| Актеры | Клиент |
| Краткое описание | Клиент открывает список туров, выбирает нужный из них, переходит на страницу со всей информацией о туре и нажимает на кнопку оформление. После проверки корректности всех персональных данных клиент нажимает на кнопку оформить, после чего система автоматически обрабатывает заявку клиента и добавляет его в список оформленных на данный тур клиентов. |
| Цель | Добавить запись клиента |
| Тип | Базовый |
| Ссылки на другие варианты использования | Авторизация и добавление отзыва |

В таблице 2 описана последовательность действий, приводящая к успешному выполнению варианта использования «Оформление тура».

Таблица 2 - Сценарий успешного выполнения варианта использования «Оформление тура»

|  |  |
| --- | --- |
| Действия актеров | Отклик системы |
| 1. Клиент оформляет заявку на тур.  Исключение №1. Количество билетов равно 0. | 2. Система обрабатывает заявку клиента и проверяет наличие билетов у выбранного тура. |
| 3. Клиент оформляет заявку на тур.  Исключение №2. Статус тура не является “Запланирован”. | 4. Система обрабатывает заявку клиента и проверяет статус тура. |

В таблице 3 представлены сценарии обработки исключительных ситуаций для варианта использования «Добавление новых экземпляров книги». Каждая исключительная ситуация сопровождается описанием и предложением решений для её обработки.

Таблица 3 – Обработка исключительных ситуаций для варианта использования “Запись клиента”

|  |  |
| --- | --- |
| Действие актеров | Отклик системы |
| Исключение №1. Количество билетов равно 0. | |
|  | 2. Система сообщает о том, что билеты на данный тур закончились и отменяет заявку клиента. |
| Исключение №2. Статус тура не является “Запланирован”. | |
|  | 4. Система сообщает о том, что статус тура был изменён и удаляет заявку клиента |

Опишем алгоритм реализации описанного сценария для варианта использования «Оформление тура» с помощью диаграммы деятельности, представленной на рисунке 2. В данном алгоритме показаны все шаги, которые необходимо выполнить пользователю и системе для успешного оформления клиента на тур. Алгоритм начинается с выбора клиентом тура, после чего, клиент записывается на тур, далее система проверяет наличие билетов и статус тура. Завершается алгоритм сохранением в базу данных записи клиента и уведомлением его об успешной записи. Диаграмма деятельности наглядно демонстрирует взаимодействие между пользователем и системой на каждом этапе, обеспечивая понимание процесса и облегчающее его реализацию в программном коде.

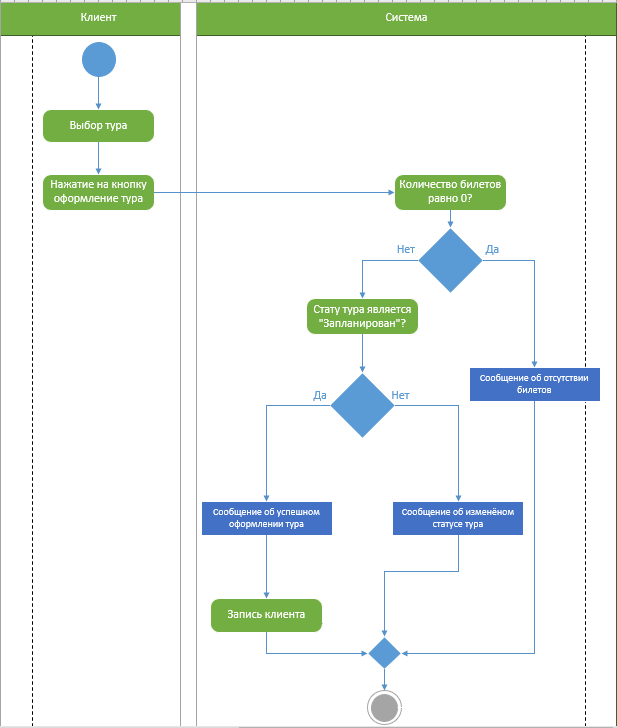


Рисунок 2 – Диаграмма деятельности для варианта использования «Оформление записи»

На следующем этапе опишем требования к реализации алгоритма с учетом динамики во времени на уровне сообщений. Для это была разработана диаграмма последовательности.

На рисунке 3 представлена диаграмма последовательности для варианта использования «Оформление записи», которая отражает требования к синхронизации операций при записи клиента на выбранный им тур.

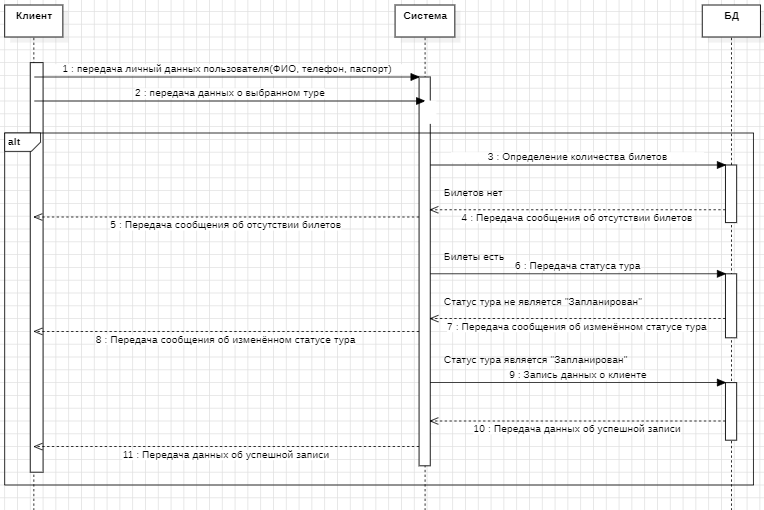


Рисунок 3 – Диаграмма последовательности для варианта использования «Оформление записи»

Приведенные выше диаграммы позволяют лучше узнать о специфике работы прецедента «Оформление записи». При разработке диаграмм использовался язык моделирования UML [7] и Visio 2022.

2.2 Проектирование модели данных и диаграммы классов

На основе анализа требований заказчика к функциям программной системы, требований к организации входных и выходных данных, а также с учетом спроектированных требований к реализации функций, была разработана модель данных системы. Модель данных представлена в виде ER-диаграммы, которая четко описывает требования к представлению логической структуры данных. На основе этой модели в дальнейшем будет разработана физическая структура данных для хранения во внешней памяти и программной обработки.

Для построения ER-диаграммы применялись средства среды MS Visio 2022. На рисунке 4 представлена полная ER-диаграмма для программных модулей системы «Турагентство Турист».

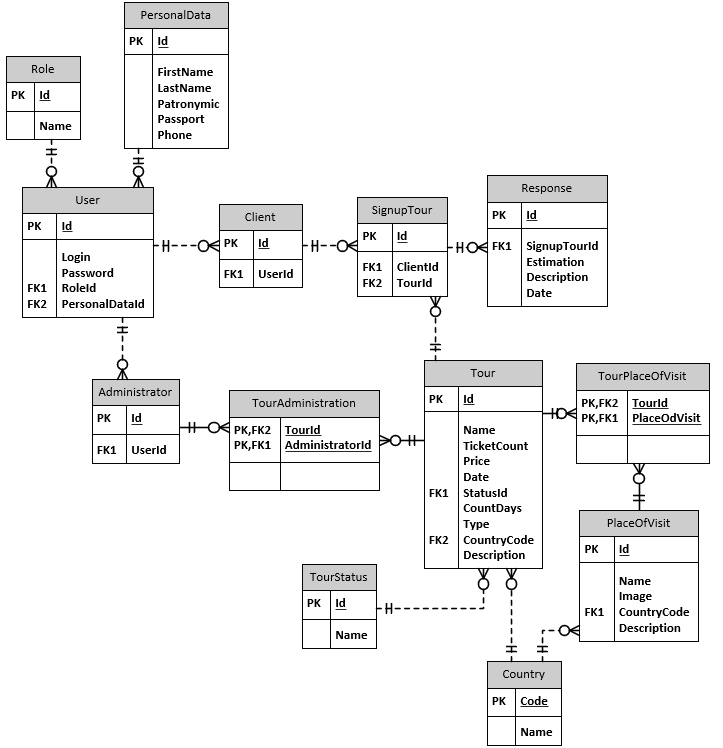


Рисунок 4 – ER-диаграмма для разрабатываемых программных модулей

В таблице 4 представлены сведения о сущностях и связях между сущностями ER-диаграммы из рисунка 4.

Таблица 4 – Характеристики сущностей и связей между сущностями   
ER-диаграммы для программных модулей системы «Турагентство Турист»

|  |  |  |  |
| --- | --- | --- | --- |
| Сущность | Назначение сущности | Ключ | Характеристики связей |
| Role | Содержит название ролей | Id (PK) | Role – User (1..n) |
| PersonalData | Содержит личную информацию пользователя | Id (PK) | PersonalData – User (1..n) |
| User | Содержит данные для входа в систему пользователя | Id (PK)  RoleId (FK)  PersonalData (FK) | User – Role (n..1)  User – PersonalData (n..1)  User – Administrator (1..n)  User – Client (1..n) |
| Administrator | Содержит данные об администраторе | Id (PK)  UserId (FK) | Administrator – User (n..1)  Administrator – TourAdministration (1..n) |
| TourAdministration | Предназначена для разбиения связи многие-ко-многим для сущностей Administrator и Tour | AdministratorId (PK, FK)  TourID (PK, FK) | TourAdministration – Administrator (n..1)  TourAdministration – Tour (n..1) |
| Client | Содержит данные о клиенте | Id (PK)  UserId (FK) | Client – User (n..1)  Client – SignupTour (1..n) |
| SignupTour | Содержит информация о записях клиента на тур | Id (PK)  ClientId (FK) TourId (FK) | SignupTour – Client (n..1)  SignupTour – Tour (n..1)  SignupTour – Response (1..n) |
| Response | Содержит информации о отзывах клиента о турах | Id (PK)  SignupTourId (FK) | Response – SignupTour (n..1) |

Продолжение таблицы 4

|  |  |  |  |
| --- | --- | --- | --- |
| Сущность | Назначение сущности | Ключ | Характеристики связей |
| Tour | Содержит информацию о туре | Id (PK)  StatusId (FK)  CountryCode (FK) | Tour – TourAdministration (1..n)  Tour – SignupTour (1..n)  Tour – TourPlaceOfVisit (1..n)  Tour – Country (n..1)  Tour – TourStatus (n..1) |
| TourStatus | Содержит информацию о статусе тура | Id (PK) | TourStatus – Tour (1..n) |
| Country | Содержит информацию о странах | Code (PK) | Country – Tour (1..n)  Country – PlaceOfVisit (1..n) |
| PlaceOfVisit | Содержит информацию о местах для посещения | Id (PK)  CountryCode (FK) | PlaceOfVisit – Country (n..1)  PlaceOfVisit – TourPlaceOfVisit (1..n) |
| TourPlaceOfVisit | Предназначена для разбиения связи многие-ко-многим для сущностей PlaceOfVisit и Tour | TourId (PK, FK) PlaceOfVisit  Id (PK, FK) | TourPlaceOfVsit – Tour (n..1)  TourPlaceOfVisit – PlaceOfVisit (n..1) |

На следующем этапе была спроектирована диаграмма классов, которая описывает требования к внутренней организации проектируемого программного обеспечения на уровне интерфейсов и классов, их кооперации, атрибутов и правил их взаимодействия.

На рисунке 5 представлена диаграмма классов для программных модулей система «Турагентство Турист».

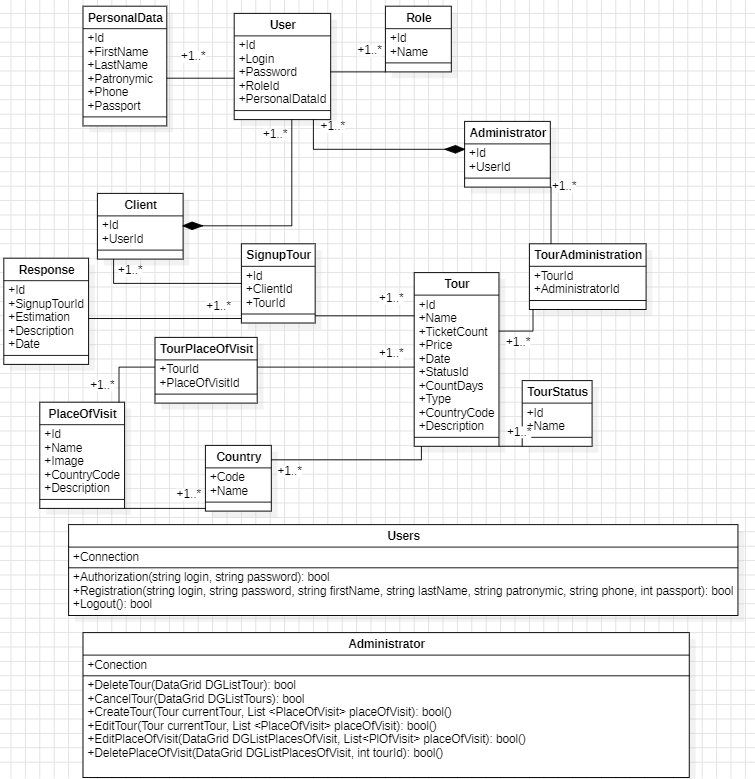


Рисунок 5 –Диаграмма классов для программных модулей системы «Турагентство Турист»

Класс User содержит данные о пользователях, такие как логин, пароль, роль пользователя и личные данные, а также метод для авторизации пользователя по вызванному методу из класса Users по логину и паролю.

Класс Role содержит данные о текущей роли пользователя, а также идентификатор.

Класс PersonalData содержит личные данные пользователя, такие как ФИО, телефоне, паспорт.

Класс Administrator содержит данные администратора, такие как идентификатор и идентификатор пользователя.

Класс Client содержит данное клиента, такие как идентификатор и идентификатор пользователя.

Класс TourAdministrator содержит информация об администраторе, который создавал, изменял или удалял тур.

Класс SignupTour содержит информацию о заявках клиентов на туров, а именно идентификатор пользователя и тура.

Класс Response содержит информацию о отзывах клиентов, которые завершили тур, а именно: идентификатор клиента, оценка, описание и дату написания отзыва.

Класс Tour содержит информацию о туре, а именно: название, количество билетов, цену, дату, идентификатор статуса, количество дней, тип, идентификатор страны и описание.

Класс TourStatus содержит информацию о текущем статусе тура.

Класс Country содержит информация о вех странах, а именно: идентификатор и название страны.

Класс PlaceOfVisit содержит информация о местах для посещения, а именно: идентификатор, название, фотографию, идентификатор страны и описание.

Класс TourPlaceOfVisit содержит информацию о метах для посещения, которые привязаны к турам.

Класс Users содержит методы для авторизации, регистрации пользователей и их выхода из системы.

Класс Administrator содержит методы для создания, редактирования и отмены туров, а также методы для редактирования и удаления мест для посещения.

# 2.3 Конструирование прототипа

Для построения прототипа первоначально были составлены требования к макету приложения, для которого разрабатываются программные модули.

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

* разметка и дизайн (предпочтение отдается масштабируемой компоновке; должно присутствовать ограничение на минимальный размер окна; должна присутствовать возможность изменения размеров окна, где это необходимо; увеличение размеров окна должно увеличивать размер контентной части, например, таблицы с данными из БД);
* группировка элементов (в логические категории);
* использование соответствующих элементов управления (например, выпадающих списков для отображения подстановочных значений из базы данных);
* расположение и выравнивание элементов (метки, поля для ввода и т.д.);
* последовательный переход фокуса по элементам интерфейса (по нажатию клавиши TAB);
* общая компоновка логична, понятна и проста в использовании;
* последовательный пользовательский интерфейс, позволяющий перемещаться между существующими окнами или страницами в приложении (в том числе обратно, например, с помощью кнопки «Назад»);
* соответствующий заголовок на каждом окне приложения.

Основные требования руководства по стилю:

* все экранные формы пользовательского интерфейса должны иметь заголовок с логотипом, представленном на рисунке 6. Цвет, размер и пропорции логотипа не должны изменяться при использовании программы. Так же для приложения должна быть установлена иконка, соответствующая логотипу;
* тип шрифта – Calibri;
* цветовая схема предусматривает использование в качестве основного фона – белый цвет RGB (247, 247, 247), в качестве дополнительного – цвет RGB (79, 147, 239), для акцентирования внимания пользователя на целевое действие интерфейса – цвет RGB (30, 210, 114).
* в качестве навигации используется картинка, изображенная на рисунке 7 и использующую 3 цвета – RGB (247, 247, 247), RGB (188,188,188), RGB (79, 147, 239);

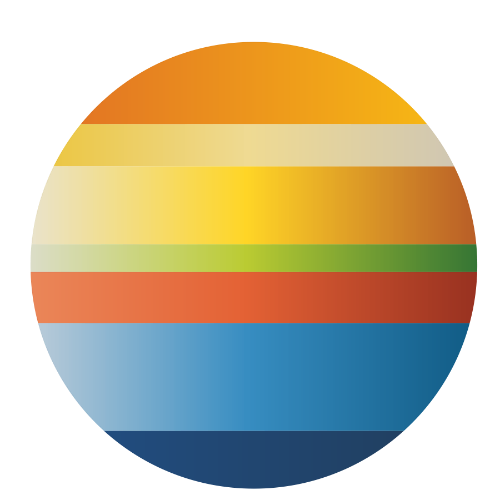


Рисунок 6 - Логотип приложения



Рисунок 7 – Иконка навигации

С учетом требований к макету и руководству по стилю, для обеспечения требуемых функций, был разработан прототип программы.

На рисунке 8 представлен прототип главного окна.

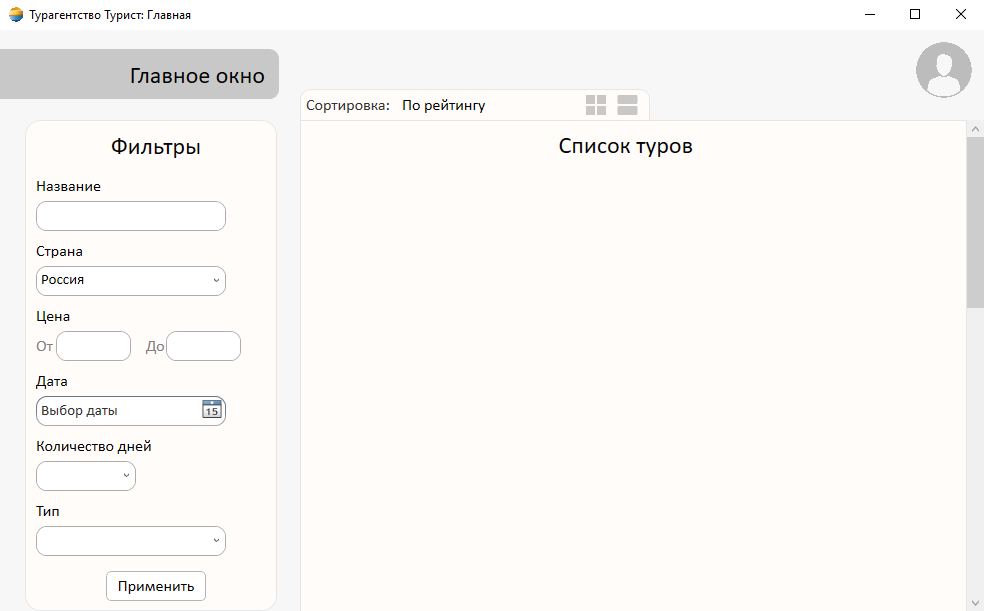


Рисунок 8 – Прототип главного окна

На рисунке 9 представлен прототип окна авторизации.

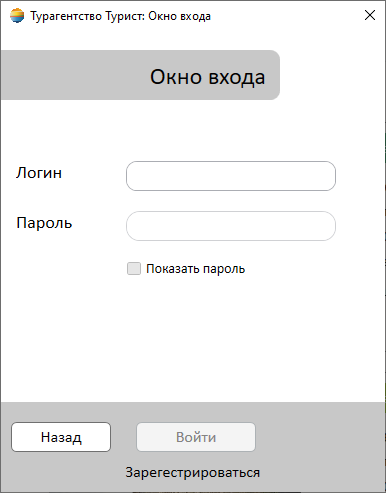


Рисунок 9 – Прототип окна авторизации

# 3 Реализация

# 3.1 Обоснование выбора средств разработки

В данном разделе мы рассмотрим основные критерии выбора среды для разработки модулей, а также системы для реализации базы данных.

Ниже приведена таблица 5, которая обоснует выбор средств для разработки программных модулей системы «Турагентство Турист».

Таблица 5 - Сравнение сред, языков программирования и систем для разработки приложений и реализации базы данных.

|  |  |  |
| --- | --- | --- |
| Категория | Название | Описание |
| Языки программирования для разработки приложений | C# | Объектно-ориентированный язык программирования, широко используемый для создания различных приложений, включая десктопные, веб и мобильные приложения. Поддерживает современную разработку, включая асинхронные операции, LINQ, и Entity Framework. |
|  | Java | Объектно-ориентированный язык программирования, используемый для разработки кросс-платформенных приложений. |
|  | C++ | Язык программирования, известный своей производительностью и эффективностью. Широко используется для системного и прикладного программирования. |

Продолжение таблицы 5

|  |  |  |
| --- | --- | --- |
| Категория | Название | Описание |
| СУБД (системы управления базами данных) | SQL Server | Реляционная СУБД, разработанная Microsoft. Обеспечивает высокую производительность, масштабируемость и надежность. Поддерживает интеграцию с различными инструментами. |
|  | MySQL | Открытая реляционная СУБД, популярная для веб-приложений. Обеспечивает хорошую производительность и простоту использования. |
|  | PostgreSQL | Открытая реляционная СУБД, известная своей расширяемостью и стандартами SQL. Поддерживает сложные запросы и большое количество данных. |
| Среда разработки | Visual Studio | Интегрированная среда разработки (IDE) от Microsoft, широко используемая для разработки на C#. Обеспечивает мощные инструменты для отладки, тестирования и управления версиями. |
|  | IntelliJ IDEA | Популярная IDE для разработки на Java, разработанная JetBrains. Поддерживает множество плагинов и инструментов для повышения производительности разработчиков. |

Для разработки приложения и реализации базы данных были выбраны следующие язык программирования C#, СУБД SQL Server и среда разработки Visual Studio.

Язык программирования C# идеально подходит для разработки десктопных приложений, особенно в сочетании с .NET Framework. C# предоставляет мощные возможности для разработки производительных и надежных приложений, что важно для библиотеки, где требуется стабильная работа с базой данных и пользовательским интерфейсом. Поддержка асинхронных операций, LINQ для запросов к данным и Entity Framework для работы с базами данных делает C# удобным и эффективным инструментом для разработки [16].

СУБД SQL Server хорошо интегрируется с другими продуктами Microsoft, такими как Visual Studio и Entity Framework, что облегчает разработку и сопровождение приложения. SQL Server предоставляет мощные возможности для работы с большими объемами данных, обеспечивая высокую производительность и надежность [5].

Среда разработки Visual Studio предоставляет удобные инструменты для отладки, тестирования и управления версиями, что ускоряет процесс разработки и повышает качество кода. Visual Studio обеспечивает полную интеграцию с C# и SQL Server, что позволяет эффективно работать с базой данных и кодом приложения в одной среде. [6].

# 3.2 Разработка базы данных в среде СУБД

На основе ранее спроектированной ER-диаграммы, в среде SQL Server Management Studio была разработана база данных «TravelAgencyTourist», база хранится в файле TravelAgencyTourist.mdf. Скриншот диаграммы базы данных, реализованная средствами утилиты SQL Server Management Studio, представлен на рисунке 10.

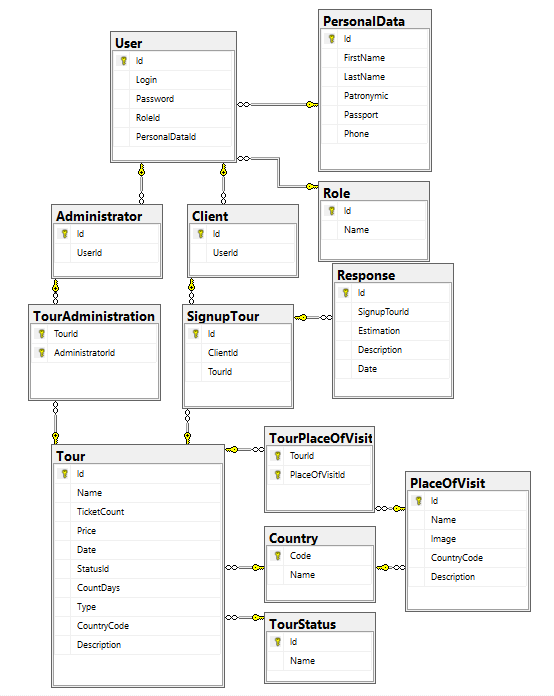


Рисунок 10 – Диаграмма базы данных TravelAgencyTourist

В таблице 6 представлено описание всех таблиц. SQL-скрипт представлен в приложении В.

Таблицы 6 – Описание таблиц базы данных TravelAgencyTourist

|  |  |
| --- | --- |
| Название таблицы | Описание |
| User | Хранит данные для о пользователях, такие как логин, пароль и роли. |
| PersonalData | Хранит персональную информацию пользователей, такие как ФИО, телефон и паспорт. |
| Role | Хранит данные о ролях пользователей. |
| Administrator | Хранит данные о администраторах. |
| Client | Хранит данные о клиентах. |
| TourAdministration | Хранит данные о администраторах, связанных с турами. |
| Tour | Хранит данные о турах, такие как название, количество билетов, цена, дата, статус, количество дней, тип, страна и описание. |
| SignupTour | Хранит данные о записях клиентов на туры. |
| Response | Хранит данные об оставленных клиентами отзывах на туры. |
| Country | Хранит данные о странах. |
| TourStatus | Хранит данные о статусе тура. |
| PlaceOfVisit | Хранит данные о местах для посещения в турах, такие как название места, фотография, страна и описание. |
| TourPlaceOfVisit | Хранит данные о местах для посещения, связанных с турами. |

Эта структура базы данных обеспечивает целостность и взаимосвязанность всех данных, необходимых для эффективного управления турагентством.

# 3.3 Разработка программных модулей

На рисунках 11-12 представлена структура приложения.

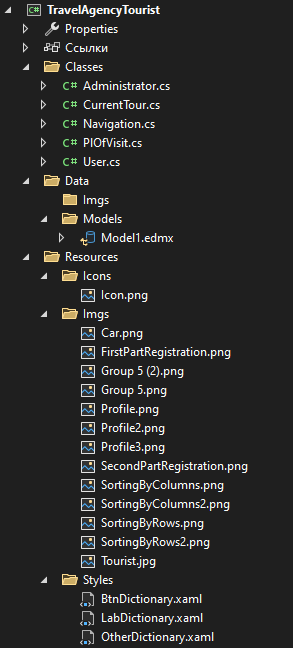


Рисунок 11 – Папки Classes, Data и Resources в структуре приложения

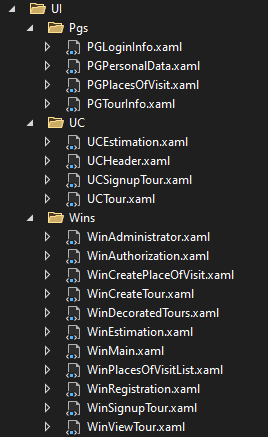


Рисунок 12 – Папка UI в структуре приложения

В таблице 7 представлено описание назначения всех модулей из проекта приложения.

Таблица 7 – Описание назначения модулей проекта

|  |  |
| --- | --- |
| Модуль | Описание |
| PGLoginInfo | Страница для ввода логина и пароля при регистрации нового пользователя. |
| PGPersonalData | Страница для ввода персональных данных при регистрации. |
| PGTourInfo | Страница для ввода или редактирования данных тура. |
| PGPlaceOfVisit | Основная страница, в которой можно убрать, создать, редактировать, а также выбрать место для посещения. |
| UCEstimation | UserControl для отображения оценки клиента о туре. |

Продолжение таблицы 7

|  |  |
| --- | --- |
| Модуль | Описание |
| UCHeader | UserControl для отображения заголовка окна. |
| UCSignupTour | UserControl для отображения туров, на которые записался пользователь |
| UCTour | UserControl для отображения туров на главном экране. |
| WinAdministrator | Основное окно администратора, для работы с турами. |
| WinAuthorization | Окно для авторизации пользователей. |
| WinCreatePlaceOfVisit | Окно для создания или редактирования места для посещения. |
| WinCreateTour | Окно, в котором содержаться PGTourInfo и PGPlaceOfVisit. |
| WinDecoratedTours | Окно для просмотра всех туров, на которые записан клиент. |
| WinEstimation | Окно для оставления отзыва о туре клиентом. |
| WinMain | Основное окно, в котором содержаться все запланированные туры. |
| WinPlaceOfVisitList | Окно, для добавления в тур уже существующих мест для посещения в выбранной стране. |
| WinRegistration | Окно, в котором содержаться PGLoginInfo и PGPersonalData. |
| WinSignupTour | Окно для записи клиента на тур. |
| WinViewTour | Окно, в которм содержится вся информация о туре. |

Рассмотрим основные разработанные макеты и обработчики. Скриншоты обработчиков и кода разметки представлены в приложении Б.

Макет страницы авторизации представлен на рисунке 13. Код разметки макета окна авторизации представлен на рисунках Б.1-Б.2. Код обработчика кнопки BtnLogon представлен на рисунке Б.3. Методы для показа и скрытия пароля Б.4.

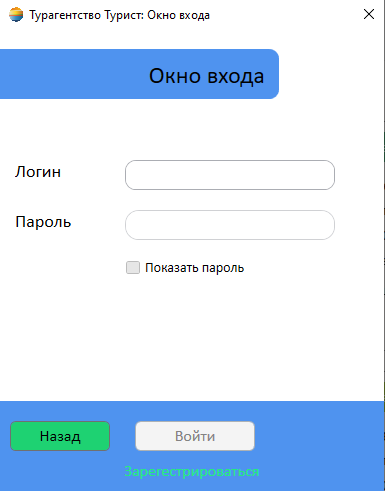


Рисунок 13 – Окно авторизации

Макет окна со списком тура, представлен на рисунке 14. Данное окно включает себя фильтрацию туров по названию, стране, цена, дате начала, количеству дней и типу. В окне присутствует сортировка по рейтингу, по убывающий цене и по возрастающей цене, а также группировка туров по одному и по двум столбцам. Код разметки макета окна со списком туров представлен на рисунках Б.5-Б.10. Код метода для загрузки туров с фильтрами представлен на рисунке Б.11. Метод для туров с группировкой по двум столбцам представлен на рисунке Б.12.

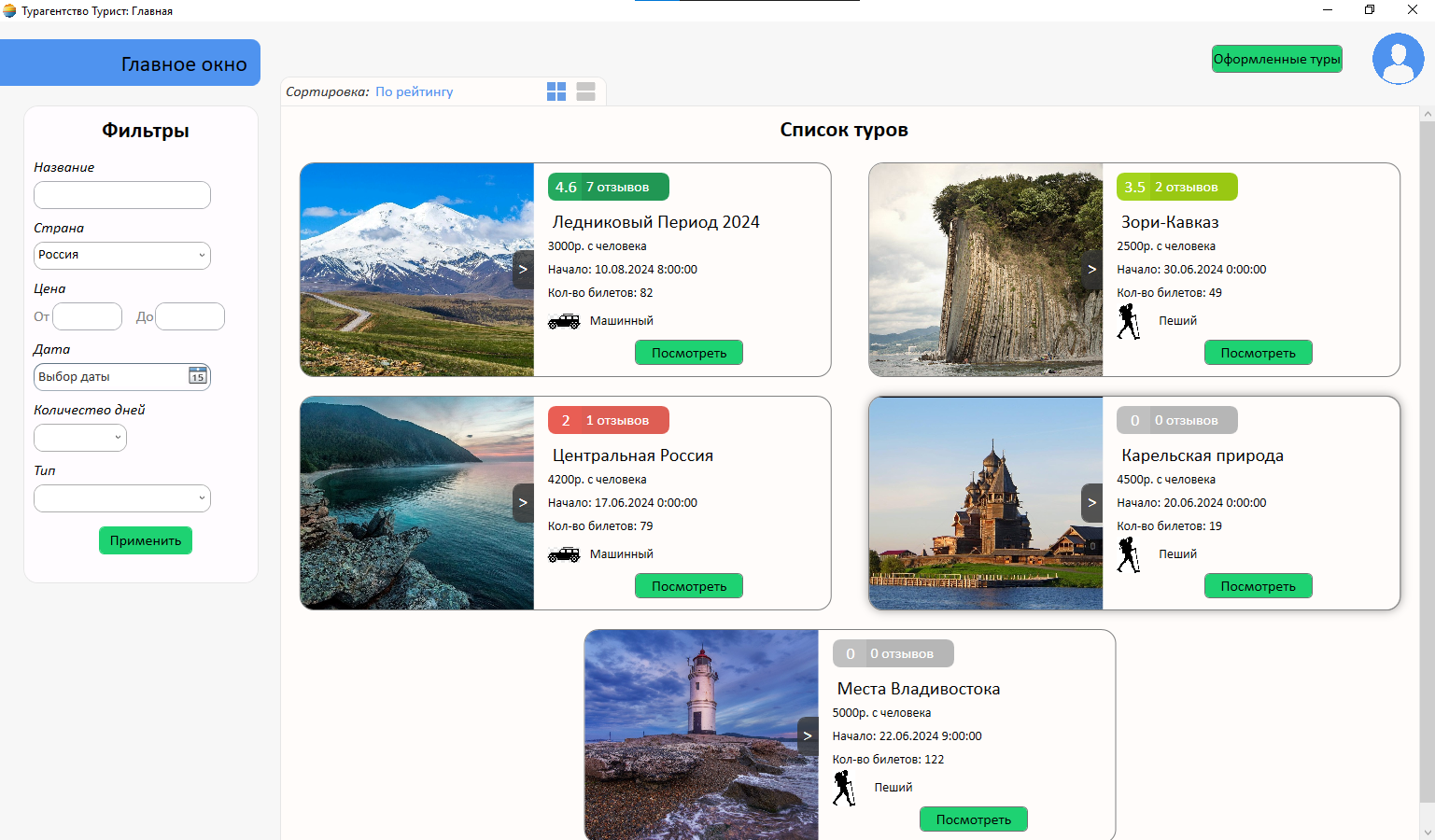


Рисунок 14 – Макет окна со списком туров с группировкой по двум столбцам

Макет окна создания места для посещения на рисунке 15. Данное окно включает в себя поле с названием места, его фотографией, а также описанием. Код разметки макета окна создания места для посещения представлен на рисунке Б.13-Б.14.

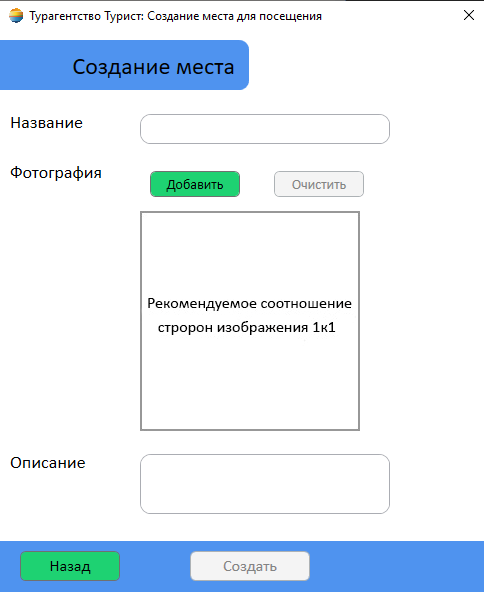


Рисунок 15 – Макет окна оформления записи

Метод, создающий и загружающий в базу данных место для посещения. Б.15. Метод, загружающий фотографию. Б.16.

# 4 Тестирование программных модулей

# 4.1 Модульное тестирование

Модульное тестирование – это процесс тестирования отдельных компонентов или модулей программы для проверки их корректной работы и изоляции возможных ошибок. Оно помогает убедиться, что каждый модуль функционирует правильно независимо от других частей программы [9].

Для реализации модульных тестов была использована среда разработки Visual Studio. В проекте был создан специальный проект для модульного тестирования, в котором были написаны и организованы тесты для проверки метода добавления новой информации о книге. Для создания модульных тестов использовались библиотеки Moq и NUnit. Код класса с модульными тестами представлен на рисунках 16-20. Результат выполнения модульных тестов предоставлен на рисунке 22.

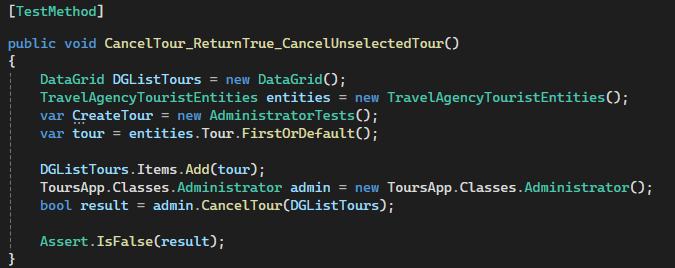


Рисунок 16 – Метод для тестирования отмены не выбранного тура из списка

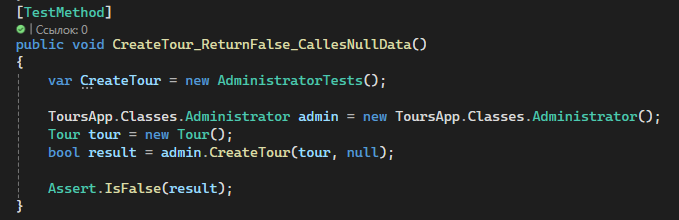


Рисунок 17 – Метод для тестирования создания тура, с нулевыми данными

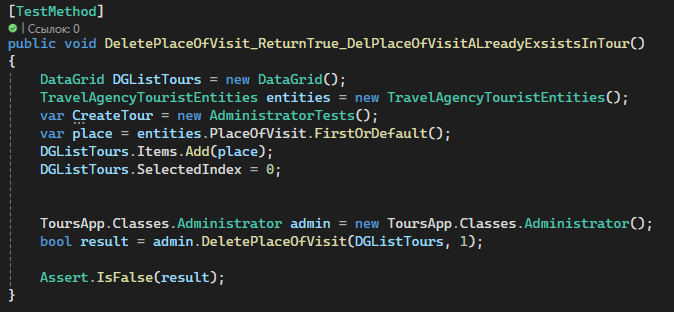


Рисунок 18 – Метод для тестирования удаления не выбранного из списка места для посещения

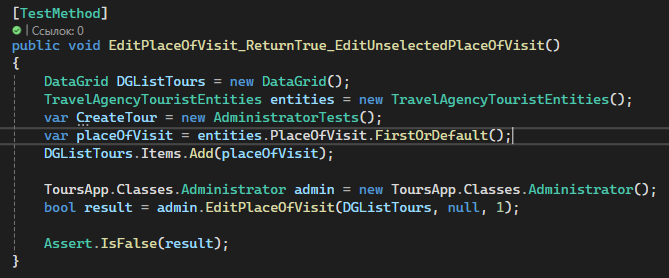


Рисунок 19 – Метод для тестирования редактирования не выбранного из списка места для посещения

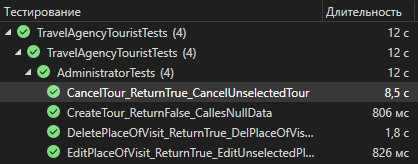


Рисунок 20 – Результаты тестов

Проведение модульного тестирование помогло убедиться, что ключевые функции окна записи клиентов работают корректно и что приложение обрабатывает различные сценарии, включая случаи с недостающими данными.

# 4.2 Интеграционное тестирование

Интеграционное тестирование — это тип тестирования программного обеспечения, который оценивает взаимодействие между модулями или компонентами системы. Его основная цель — выявить дефекты, возникающие при объединении модулей, и убедиться, что они работают вместе, как предполагалось. Это тестирование помогает обнаружить проблемы с интерфейсами, передачей данных и логикой взаимодействия. Интеграционное тестирование может проводиться инкрементно, когда модули интегрируются и тестируются поэтапно, что позволяет выявлять и исправлять ошибки на ранних стадиях. Важно убедиться, что все компоненты системы работают совместно и выполняют все функции и требования [15]. Тест-кейсы для интеграционного тестирования программных модулей системы «Турагентство Турист» представлены в приложении Д.

Общая информация о тестировании представлена в таблице 8.

Таблица 8 – Общая информация о тестировании

|  |  |
| --- | --- |
| Название проекта | TravelAgencyTourist |
| Номер версии | 1.0 |
| Имя тестировщика | Шанаурин Андрей |
| Даты тестирования | 15.06.2024 – 17.06.2024 |

В таблице Г.1 представлен тест-кейс №1, который предназначен для проверки создания нового тура.

В таблице Г.2 представлен тест-кейс №2, который предназначен для проверки записи клиента на тур.

В таблице Г.3 представлен тест-кейс №3, который предназначен для проверки добавления отзыва в выбранном туре.

# 5 Эксплуатационная документация

# 5.1 Руководство пользователя

Руководство пользователя представляет собой документ, предназначенный для описания назначения, условий применения, подготовки к работе и эксплуатации программных модулей системы. В нем подробно изложены все необходимые инструкции по использованию системы, включая описание операций и действия, которые должен предпринять пользователь в случае возникновения аварийных ситуаций. При составлении руководства пользователя были использованы рекомендации из методических указаний РД 50-34.698-90 [4] из комплекса ГОСТ 34. Документ представлен в приложении Д.

# 5.2 Руководство системного программиста

Руководство программиста является важным документом, описывающим внутреннюю структуру программного обеспечения, алгоритмы и методы, используемые при разработке системы. В этом руководстве приведены инструкции по установке, настройке и модификации программных модулей, а также рекомендации по их сопровождению и обновлению. При составлении руководства программиста были использованы рекомендации из методических указаний РД 50-34.698-90 [4] из комплекса ГОСТ 34 и представлен в приложении Е.

# Заключение

В результате проделанной работы в рамках курсового проектирования были разработаны программные модули системы, соответствующие требованиям ГОСТ 19.102-77 «Единая система программной документации (ЕСПД). Стадии разработки» [1]. Процесс разработки включал несколько ключевых этапов, каждый из которых был выполнен в соответствии с поставленными задачами и требованиями.

На начальном этапе были определены основные задачи, представленные перед разработчиком: создание программных модулей для автоматизации управления турагентством, интеграция системы с базой данных, разработка пользовательского интерфейса и реализация функционала для администратора, пользователя и клиента системы.

Этап технического проекта включал сбор и анализ требований, создание архитектуры системы, моделирование данных и разработку диаграмм вариантов использования. В ходе реализации была осуществлена разработка пользовательского интерфейса с использованием WPF, интеграция с базой данных SQL Server и реализация модулей для работы с турами, а также оформлением записей на туры.

Процесс тестирования включал проверку основных функций системы.

Тестирование различных сценариев использования системы позволило выявить и устранить возможные дефекты, обеспечив высокое качество конечного продукта.

В завершение, предложены направления для дальнейшего развития системы: В будущем можно расширить функциональные возможности системы, добавить создание дополнительных услуг в туре для администратора и возможное включение этих услуг клиентом при оформлении тура, возможность создания тура по нескольким странам, возможность указания вида места для посещения при его создании (природная достопримечательность, храм, пещера, отель, заправка и т.д.), улучшение пользовательского интерфейса и интеграцию с другими системами.

Был разработан список использованных источников при разработке программных модулей в соответствии с ГОСТ 7.0.100 [2].

Список использованных источников

1. ГОСТ 2.105 – 2019. Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам (Издание с Изменением N 1) = Unified system for design documentation. General requirements for textual documents: национальный стандарт Российской Федерации: издание официальное: утвержден и введен в действие Приказом Федерального агентства по техническому регулированию и метрологии от 29 апреля 2019 г. № 175-ст: введен впервые: дата введения 2021-02-01 / Разработан Федеральным государственным унитарным предприятием «Российский научно-технический центр информации по стандартизации, метрологии и оценке соответствия» (ФГУП «СТАНДАРТИНФОРМ»). – Москва.: Стандартинформ, 2021. – 35 с. –Текст непосредственный.

2. ГОСТ 7.0.100 – 2018. Система стандартов по информации, библиотечному и издательскому делу (СИБИД). Библиографическая запись. Библиографическое описание. Общие требования и правила составления (с Поправкой) = System of standards on information, librarianship and publishing. Bibliographic record. Bibliographic description. General requirements and rules: национальный стандарт Российской Федерации: издание официальное: утвержден и введен в действие Приказом Федерального агентства по техническому регулированию и метрологии от 3 декабря 2018 года № 1050-ст: введен впервые: дата введения 2019-07-01 / Разработан Федеральным государственным унитарным предприятием "Информационное телеграфное агентство России (ИТАР-ТАСС)", филиал "Российская книжная палата", Федеральным государственным бюджетным учреждением "Российская государственная библиотека", Федеральным государственным бюджетным учреждением "Российская национальная библиотека". – Москва: Стандартинформ, 2018. – 128 с. – Текст непосредственный.

3. ГОСТ 19.201-78. Единая система программной документации (ЕСПД). Техническое задание. Требования к содержанию и оформлению (с Изменением N 1) = Unified system for program documentation. Technical specifications for development. Requirements to contents and form of presentation: межгосударственный стандарт: издание официальное: утвержден и введен в действие Постановлением Государственного комитета СССР по стандартам от 18 декабря 1978 г. № 3351: введен впервые: дата введения 1980-01-01. – Москва: Стандартинформ, 2010. – 4 с. – Текст непосредственный.

4. РД 50-34.698-90. Методические указания. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы. Автоматизированные системы. Требования к содержанию документов: руководящий документ по стандартизации: издание официальное: утверждены и введены в действие Постановлением Государственного комитета СССР по управлению качеством продукции и стандартами от 27 декабря 1990 г. № 3380: дата введения 1992-01-01 / Разработан Министерством электротехнической 38 ККЭП 09.02.07 129 ПЗ Лист Изм. Лист № докум. Подпись Дата промышленности и приборостроения СССР. – Москва.: ИПК Издательство стандартов, 2002 г. – 27 с. –Текст непосредственный.

5. METANIT.COM. Сайт о программировании. Полное руководство по языку программирования С# 9.0 и платформе .NET 5: [сайт] – Текст: электронный. – URL: https://metanit.com/sharp/tutorial (дата обращения 16.05.2024).

6. Хабр. Microsoft SQL Server — Система управления реляционными базами данных: [сайт] – Текст: электронный. – URL: https://habr.com/ru/hubs/mssql/articles/ (дата обращения 05.06.2024).

7. Полезные возможности среды Visual Studio. [сайт] – Текст: электронный. – URL: https://habr.com/ru/sandbox/130042/ (дата обращения 05.06.2024).

8. Семакин, И.Г. Основы алгоритмизации и программирования: учебник для студ. учреждений среднего профессионального образования / И.Г. Семакин, А.П. Шестаков. – 2-е изд., стер. – М.: Издательский центр «Академия», 2018. – 304 с. – (Профессиональное образование). – ISBN 978-5-4468-6228-3. – Текст: непосредственный. (дата обращения 15.05.2024).

9. Федорова, Г.Н. Основы проектирования баз данных: учебник для студ. учреждений среднего профессионального образования / Г.Н. Федорова. – 2-е изд., стер. – М.: Издательский центр «Академия», 2018. – 224 с. – (Профессиональное образование). – ISBN 978-5-4468-6170-5. – Текст: непосредственный. (дата обращения 04.05.2024).

10. Федорова, Г.Н. Разработка модулей программного обеспечения компьютерных систем: учебник для студ. учреждений среднего профессионального образования / Г.Н. Федорова. – М.: Издательский центр «Академия», 2016. – 336 с. – (Профессиональное образование). – ISBN 978-5-4468-1585-2. – Текст: непосредственный. (дата обращения 12.05.2024)

11. Российская Федерация. Министерство образования и науки. Об утверждении федерального государственного стандарта среднего профессиональ¬ного образования по специальности 09.02.07 Информационные системы и программирование: Приказ Министерства образования науки Российской Федерации от 09.12.2016 № 1547: редакция от 16.01.2017: зарегистрировано в Минюсте России 26.12.2016 № 44936. – Текст: электронный // СПС «Гарант» [сайт] – URL: https://www.garant.ru/products/ipo/prime/doc/71477324/#review (дата обращения 01.05.2024).

12. Тестирование ПО. Ulearn.me: [сайт] – Текст. Видеоматериалы: электронные. – URL: https://ulearn.me/Course/Testing/Vvedenie\_5656d8a3-1269-4834-bdfe-8fbc1f1c8f30 (дата обращения 25.05.2024).

13. METANIT.COM. Сайт о программировании. Полное руководство по языку программирования С# 9.0 и платформе .NET 5: [сайт] – Текст: электронный. – URL: https://metanit.com/sharp/tutorial (дата обращения 20.05.2024).

14. UML-диаграммы классов. Программирование. [сайт] – Текст: электронный. – URL: https://prog-cpp.ru/uml-classes (дата обращения 05.06.2024).

Приложение А

Техническое задание. Требования к программным модулям

1 Введение

1.1 Наименование программы

Наименование программной системы «Турагентство Турист».

Наименование разрабатываемых программных модулей системы «Турагентство Турист»: «Работа с турами» и «Запись клиентов».

1.2 Краткая характеристика области применения

Программная система «Турагентство Турист» предназначена для использования в компании ООО «Nature Tours», занимающийся составлением туров и записи клиентов на пешие и автомобильные туры. Модуль «Работа с турами» будет использоваться сотрудниками организации, с ролью администратор, а также посетителями системы «Турист» и пользователями, которые имеют роль клиент. В зависимости от роли, функционал модуля может отличаться. Модуль «Запись клиентов» будет использоваться пользователями, которые имеют роль клиент.

2 Основания для разработки

Основанием для разработки является Договор 14 от 30.08.2021. Договор утвержден директором ООО «Nature Tours» Головином Максимом Владиславовичем, именуемым в дальнейшем Заказчиком, и Шанауриным Андреем Владиславовичем (самозанятый), именуемым в дальнейшем исполнителем, 30.08.2024.

Согласно Договору, Исполнитель обязан разработать и установить программную систему «Турист» на оборудовании Заказчика не позднее 29.04.2024, предоставить исходные коды и документацию к разработанной системе не позднее 25.05.2024.

Наименование темы разработки – «Разработка программных модулей тестовой системы «Турагентство Турист».

Условное обозначение темы разработки (шифр темы) – «ТурРПС -01».

3 Назначение разработки

3.1 Функциональное назначение

При работе в системе «Турагентство Турист» должны быть реализованы две роли: администратор и клиент.

Модуль «Работа с турами» должен обеспечивать возможность сотрудникам организации «Nature Tours», которые имеют роль администратор, создавать, изменять и отменять туры, создавать новые места для посещения, которые можно добавить к туру, или выбирать уже существующие. Так же этот модуль должен обеспечивать возможность администраторам, клиентам и посетителям системы «Турист» просматривать список туров и информацию о выбранном туре.

Модуль «Запись клиентов» должен обеспечивать возможность клиентам самостоятельно оформлять подходящий тур из списка всех туров, а также оставлять отзыв о проведённом туре. Клиента, который уже оформил и оплатил тур, может скачать билет тура в формате pdf.

3.2 Эксплуатационное назначение

Программная система будет использоваться на рабочих местах сотрудников компании «Nature Tours», представляющих собой персональные компьютеры с операционной системой Windows.

4 Требования к программе

4.1 Требования к функциональным характеристикам

4.1.1 Требования к составу выполняемых функций с входными и выходными данными

Требования к входным и выходным данным функций модулей «Работа с турами» и «Запись клиентов» предоставлены в таблицах – А1 - А3.

Таблица А.1 – Требования к функциям модуля «Работа с турами» для сотрудников организации, которые имеют роль администратор

|  |  |  |
| --- | --- | --- |
| Наименование функции | Входные данные | Выходные данные |
| Составление тура | Название, количество билетов, цена, дата и время, количество дней, тип тура, статус, страна, описание | Сообщение об успешном или отрицательном результате составления тура |
| Удаление тура | Выбранный тур | Сообщение об успешном или отрицательном результате удаления тура |
| Создание и редактирование места для посещения | Название места для посещения, фотография, страна описание | Сообщение об успешном или отрицательном результате составления места для посещения |

Продолжение таблицы А.1

|  |  |  |
| --- | --- | --- |
| Удаление места для посещения | Выбранное место для посещения | Сообщение об успешном или отрицательном результате удаления места для посещения |
| Добавление места для посещения в тур | Выбранный тур, выбранное место для посещения | Добавленное в тур место для посещения |

Таблица А.2 – Требования к функциям модуля «Работа с турами» для сотрудников организации, которые имеют роль администратор, а также посетителей и клиентов системы «Турист»

|  |  |  |
| --- | --- | --- |
| Наименование функции | Входные данные | Выходные данные |
| Просмотр списка туров | Список туров | Список туров, которые имеют статус “Запланирован” |
| Просмотр информации тура | Выбранный тур | Название, количество билетов, цена, дата и время, количество дней, тип тура, статус, страна, описание, места для посещения данного тура |

Таблица А.3 – Требования к функциям модуля «Запись клиентов»

|  |  |  |
| --- | --- | --- |
| Наименование функции | Входные данные | Выходные данные |
| Оформить тур | Данные клиента, данные тура | Запись клиента |
| Создание и редактирование отзыва о туре | Данные клиента, оценка, описание | Сформированный отзыв |
| Удаление отзыва о туре | Выбранный отзыв | Сообщение об успешном или отрицательном результате удаления отзыва |
| Скачать билет | Данные клиента, данные тура | Билет в формате pdf |

4.2 Требования к надежности и безопасности

Система должна функционировать в многопользовательском режиме, поэтому каждый пользователь должен иметь свой пароль доступа в систему. Необходимо разграничить пользовательские права, т.е. не дать клиентам возможности редактировать туры и места для посещения. Одновременно в системе может находиться несколько десятков пользователей. Система должна обеспечить одновременный доступ к одним и тем же данным в базе данных, для чего необходимо применение архитектуры клиент–сервер. Кроме того, в системе должно быть предусмотрено резервное копирование и восстановление данных, а также самовосстановление системы после сбоев в операционной системе или отключения электропитания.

4.3 Требования к составу и параметрам технических средств

Состав технических средств предусматривает наличие компьютеров – рабочих станций и компьютеров -серверов.

Компьютер-рабочая станция (для клиента, администратора) включает:

* процессор x86 с тактовой частотой, не менее 1 ГГц;
* оперативную память объемом, не менее 1 Гб;
* видеокарту, монитор, мышь, клавиатура.

Два компьютера – сервера (для функционирования СУБД) (основной и резервный), включают в себя:

* процессор x86 с тактовой частотой, не менее 1 ГГц;
* оперативную память объемом, не менее 2 Гб;
* видеокарту, монитор, мышь.

4.4 Требования к информационной и программной совместимости

На серверах должны быть установлены ОС Windows Server 2019, сервер БД Microsoft SQL Server Express 2019, SQL Server Management Studio 18.8.

На компьютерах-рабочих станциях должна быть установлена операционная система ОС Microsoft Windows 10 Pro (Edu). Все формируемые отчеты должны иметь возможность экспортирования в редактор электронных таблиц Microsoft Office 2019 или 365 (Excel).

5 Требования к программной документации

В состав программной документации обязательно должно входить «Руководство пользователя», в котором должны быть описаны все инструкции для администратора, клиента и посетителя.

Приложение Б

(обязательное)

Программный код

Программный код представлен в контексте текстового редактора Microsoft Visual Studio Professional 2022 (64-разрядная версия) – Preview Версия 17.7.0 Preview 2.0

<Window x:Class="ToursApp.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:uc="clr-namespace:ToursApp.UI.UC"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:ToursApp"

mc:Ignorable="d"

Title="Турагентство Турист: Окно входа"

WindowStartupLocation="CenterScreen"

ResizeMode="NoResize"

Height="500"

Width="400"

KeyUp="Window\_KeyUp"

Icon="/Resources/Icons/Icon.png">

<Grid>

<uc:UCHeader x:Name="UCHeader"

Title="Окно входа"

Margin="0,20,0,0"/>

<Grid Margin="15,130,0,200">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="110px"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="50px"/>

<RowDefinition Height="50px"/>

<RowDefinition Height="50px"/>

</Grid.RowDefinitions>

<Label x:Name="LabLogin"

Content="Логин"

Grid.Column="0"

Grid.Row="0"/>

<TextBox x:Name="TBoxLogin"

Grid.Column="1"

Grid.Row="0"

GotFocus="TextBox\_GotFocus"

LostFocus="TBoxLogin\_LostFocus"

TextChanged="TBoxLogin\_TextChanged"/>

<Label x:Name="LabPassword" Content="Пароль"

Grid.Column="0"

Grid.Row="1" />

<TextBox x:Name="TBoxPassword"

Grid.Column="1"

Grid.Row="1"

Visibility="Hidden"

LostFocus="PBoxPassword\_LostFocus"

GotFocus="PBoxPassword\_GotFocus"/>

Рисунок Б.1 – Программный код разметки окна авторизации

<PasswordBox x:Name="PBoxPassword"

Grid.Column="1"

Grid.Row="1"

GotFocus="PBoxPassword\_GotFocus"

LostFocus="PBoxPassword\_LostFocus"/>

<CheckBox x:Name="CBoxCheckPassword"

Grid.Column="1"

Grid.Row="2"

Content="Показать пароль"

FontSize="14"

Checked="CheckBox\_Checked"

Unchecked="CheckBox\_Unchecked"/>

</Grid>

<Grid x:Name="GridFooter"

VerticalAlignment="Bottom"

Background="#FF4F93EF">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="110"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="60"/>

<RowDefinition Height="30"/>

</Grid.RowDefinitions>

<Button x:Name="BtnMain"

Style="{DynamicResource BtnLogonStyle}"

Grid.Column="0"

Grid.Row="0"

Content="Назад"

Margin="10,20,0,0"

Width="100"

Click="BtnMain\_Click"/>

<Button x:Name="BtnLogon"

Style="{DynamicResource BtnLogonStyle}"

Margin="25,20,0,0"

Grid.Column="1"

Grid.Row="0"

Content="Войти"

Width="120"

Click="BtnLogon\_Click"/>

<Button x:Name="BtnNotRegistered"

Style="{StaticResource BtnNotRegisteredStyle}"

Grid.ColumnSpan="2"

MouseEnter="BtnNotRegistered\_MouseEnter"

MouseLeave="BtnNotRegistered\_MouseLeave"

Click="BtnNotRegistered\_Click"

Margin="0,55,0,0"

VerticalAlignment="Top"

HorizontalAlignment="Center"

Grid.RowSpan="2"/>

</Grid>

</Grid>

</Window>

Рисунок Б.2 – Программный код разметки окна авторизации

private void BtnLogon\_Click(object sender, RoutedEventArgs e) {

if (TBoxPassword.Visibility == Visibility.Visible) {

PBoxPassword.Password = TBoxPassword.Text; }

else if (PBoxPassword.Visibility == Visibility.Visible) {

TBoxPassword.Text = PBoxPassword.Password; }

bool result = user.Authorization(TBoxLogin.Text, TBoxPassword.Text);

if (result == true) {

App.Current.Resources["Attempt"] = "";

window.Close(); }

else if (result == false) {

if (attempt != 1) {

attempt--;

App.Current.Resources["Attempt"] = attempt.ToString();

MessageBox.Show($"Введён неверный логин или пароль!\nОсталось попыток: {attempt}", "Неверные данные", MessageBoxButton.OK, MessageBoxImage.Warning); }

else {

MessageBox.Show($"Превышен лимит попыток!\nОкно заблокировано на 10 секунд!", "Неверные данные", MessageBoxButton.OK, MessageBoxImage.Warning);

BtnLogon.IsEnabled = false;

BtnNotRegistered.IsEnabled = false;

BtnMain.IsEnabled = false;

TBoxLogin.IsEnabled = false;

TBoxPassword.IsEnabled = false;

PBoxPassword.IsEnabled = false;

CBoxCheckPassword.IsEnabled = false;

BtnLogon.Content = time;

App.Current.Resources["Attempt"] = "";

attempt = 3;

timer.Tick += new EventHandler(Timer\_Tick);

timer.Interval = new TimeSpan(0, 0, 1);

timer.Start();

}

}

}

Рисунок Б.3 – Программный код обработчика для кнопки «Войти» в WinAuthorization «Авторизация»

private void CheckBox\_Checked(object sender, RoutedEventArgs e)

{

TBoxPassword.Text = PBoxPassword.Password;

PBoxPassword.Visibility = Visibility.Hidden;

TBoxPassword.Visibility = Visibility.Visible;

}

private void CheckBox\_Unchecked(object sender, RoutedEventArgs e)

{

PBoxPassword.Password = TBoxPassword.Text;

PBoxPassword.Visibility = Visibility.Visible;

TBoxPassword.Visibility = Visibility.Hidden;

}

Рисунок Б.4 – Программный код методов для показа и скрытия пароля

<Window x:Class="ToursApp.UI.Winds.WinMain"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:uc="clr-namespace:ToursApp.UI.UC"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:ToursApp.UI.Winds"

mc:Ignorable="d"

WindowStartupLocation="CenterScreen"

Style="{StaticResource WinMainStyle}"

Activated="Window\_Activated"

Loaded="Window\_Loaded"

StateChanged="Window\_StateChanged"

SizeChanged="Window\_SizeChanged">

<Grid x:Name="Grid"

Background="#F7F7F7">

<uc:UCHeader Title="Главное окно"

Margin="0,20,0,0"/>

<Border x:Name="BorderSorting"

Style="{StaticResource MainBorderStyle}"

CornerRadius="10,10,0,0"

Width="350"

Height="32"

Margin="300,59,0,0">

<Grid x:Name="GridSorting"

Margin="5,5,0,0">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="95"/>

<ColumnDefinition Width="185"/>

<ColumnDefinition/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Label x:Name="LabSorting"

Content="Сортировка:"

FontSize="16"

Foreground="#FF323232"

Grid.Column="0"

FontStyle="Italic"/>

<Button x:Name="BtnSorting"

Style="{DynamicResource BtnNotRegisteredStyle}"

Margin="0"

HorizontalAlignment="Left"

Foreground="#FF4F93EF"

Content="По рейтингу"

HorizontalContentAlignment="Left"

Grid.Column="1"

Background="{x:Null}"

BorderBrush="{x:Null}"

Click="BtnSorting\_Click" />

Рисунок Б.5 – Программный код разметки макета главного окна

<Image x:Name="ImgSortingByColumns" Width="20"

Grid.Column="2"

Height="20"

HorizontalAlignment="Left"

Opacity="0.6"

VerticalAlignment="Top"

Source="/Resources/Imgs/SortingByColumns2.png"

MouseUp="ImgSortingByColumns\_MouseUp"/>

<Image x:Name="ImgSortingByRows"

Width="20"

Grid.Column="3"

Height="20"

Cursor="Hand"

HorizontalAlignment="Left"

Opacity="0.6"

VerticalAlignment="Top"

Source="/Resources/Imgs/SortingByRows.png"

MouseUp="ImgSortingByRows\_MouseUp"/>

</Grid>

</Border>

<Border Style="{StaticResource MainBorderStyle}"

Margin="25,90,0,0"

Background="#FFFFFCFF">

<Grid x:Name="GFilters"

Width="250"

Height="500"

VerticalAlignment="Top"

HorizontalAlignment="Left"

Margin="10,10,0,0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition Height="30"/>

<RowDefinition Height="35"/>

<RowDefinition Height="30"/>

<RowDefinition Height="35"/>

<RowDefinition Height="30"/>

<RowDefinition Height="35"/>

<RowDefinition Height="30"/>

<RowDefinition Height="35"/>

<RowDefinition Height="30"/>

<RowDefinition Height="35"/>

<RowDefinition Height="30"/>

<RowDefinition Height="45"/>

<RowDefinition Height="30"/>

</Grid.RowDefinitions>

<Label x:Name="LabFilters"

Grid.Row="0"

Content="Фильтры"

FontSize="24"

FontWeight="Bold"

HorizontalAlignment="Center"/>

<Label x:Name="LabName"

Grid.Row="1"

Content="Название"

Margin="0,5,0,0"

FontSize="16"

FontStyle="Italic"/>

<TextBox x:Name="TBoxName"

Grid.Row="2"

Width="190"

Padding="3,3,0,0"

Style="{DynamicResource TextBoxFilterStyle}"

FontSize="15" />

Рисунок Б.6 – Программный код разметки макета главного окна

<Label x:Name="LabCountry"

Grid.Row="3"

Content="Страна"

Margin="0,5,0,0"

FontSize="16"

FontStyle="Italic"/>

<ComboBox x:Name="CBoxCountry"

Template="{DynamicResource ComboBoxTemplate}"

FontSize="15"

Padding="5,4,0,0"

ItemsSource="{Binding}"

Grid.Row="4"

Width="190"/>

<Label x:Name="LabPrice"

Grid.Row="5"

Content="Цена"

Margin="0,5,0,0"

FontSize="16"

FontStyle="Italic"/>

<Label x:Name="LabMinPrice"

Foreground="Gray"

Grid.Row="6"

Content="От"

Margin="0,5,0,0"

FontSize="16"/>

<TextBox x:Name="TBoxMinPrice"

PreviewTextInput="NumberValidationTextBox"

Grid.Row="6"

Width="75"

Padding="3,3,0,0"

Style="{DynamicResource TextBoxFilterStyle}"

FontSize="15"

Margin="20,0,0,0" />

<Label x:Name="LabMaxPrice"

Foreground="Gray"

Grid.Row="6"

Content="До"

Margin="110,5,0,0"

FontSize="16"/>

<TextBox x:Name="TBoxMaxPrice"

PreviewTextInput="NumberValidationTextBox"

Grid.Row="6"

Width="75"

Margin="130,0,0,0"

Padding="3,3,0,0"

Style="{DynamicResource TextBoxFilterStyle}"

FontSize="15" />

<Label x:Name="LabDate"

Grid.Row="7"

Content="Дата"

Margin="0,5,0,0"

FontSize="16"

FontStyle="Italic"/>

<DatePicker x:Name="DPTourDate"

Padding="3,3,0,0"

Template="{DynamicResource DatePickerTemplate}"

IsTabStop="True"

Grid.Row="8"

Width="190"

FontSize="15">

<DatePicker.Resources>

<Style TargetType="DatePickerTextBox">

<Setter Property="IsReadOnly"

Value="True"/>

</Style>

</DatePicker.Resources>

</DatePicker>

Рисунок Б.7 – Программный код разметки макета главного окна

<Label x:Name="LabCountDays" Grid.Row="9"

Content="Количество дней"

Margin="0,5,0,0"

FontSize="16"

FontStyle="Italic"/>

<ComboBox x:Name="CBoxCountDays"

Template="{DynamicResource ComboBoxTemplate}"

FontSize="15"

Padding="5,4,0,0"

Grid.Row="10"

Width="100"/>

<Label x:Name="LabType"

Grid.Row="11"

Content="Тип"

Margin="0,5,0,0"

FontSize="16"

FontStyle="Italic"/>

<ComboBox x:Name="CBoxType"

FontSize="15"

Padding="5,4,0,0"

Template="{DynamicResource ComboBoxTemplate}"

Grid.Row="12"

Width="190"/>

<Button x:Name="BtnSearch"

Style="{DynamicResource BtnLogonStyle}"

Content="Применить"

Grid.Row="13"

VerticalAlignment="Top"

Width="100"

Click="BtnSearch\_Click"

BorderBrush="#FF1ED272"

HorizontalAlignment="Center"/>

</Grid>

</Border>

<ScrollViewer Margin="300,90,0,0">

<Border Style="{StaticResource MainBorderStyle}"

CornerRadius="0"

HorizontalAlignment="Stretch"

VerticalAlignment="Stretch">

<Grid x:Name="ListViewProductCard"/>

</Border>

</ScrollViewer>

<Button x:Name="BtnCreateTour"

Content="Работа с турами"

Visibility="Hidden"

Style="{DynamicResource BtnLogonStyle}"

Margin="0,25,100,0"

Width="115"

Click="BtnCreateTour\_Click"

HorizontalAlignment="Right"

Cursor=""/>

<Button x:Name="BtnToursDecorated"

Content="Оформленные туры"

Style="{DynamicResource BtnLogonStyle}"

Margin="0,25,100,0"

HorizontalAlignment="Right"

Visibility="Hidden"

Click="BtnToursDecorated\_Click" />

Click="BtnToursDecorated\_Click" />

<Label x:Name="LabMainTourInfo" Visibility="Hidden"

Content="Туры не найдены"

Margin="275,170,0,0"

FontWeight="Bold"

HorizontalAlignment="Center"/>

<Label x:Name="LabTourInfo"

Visibility="Hidden"

Content="Мы не нашли вариантов по вашим критериям поиска. Попробуйте их изменить."

FontSize="14"

Margin="275,200,0,0"

HorizontalAlignment="Center"/>

Рисунок Б.8 – Программный код разметки макета главного окна

<Image x:Name="ImgProfile"

HorizontalAlignment="Right"

Height="60"

Margin="0,10,10,0"

VerticalAlignment="Top"

Width="60"

Source="/Resources/Imgs/Profile2.png"

MouseEnter="ImgProfile\_MouseEnter"

MouseLeave="ImgProfile\_MouseLeave"/>

<Border x:Name="BorderUserInfo"

Visibility="Hidden"

Style="{StaticResource MainBorderStyle}"

CornerRadius="10"

BorderBrush="#FF4F93EF"

Width="200"

Height="120"

HorizontalAlignment="Right"

Margin="0,70,10,0">

<Grid Margin="10,5,0,0">

<Grid.ColumnDefinitions>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="25"/>

<RowDefinition Height="25"/>

<RowDefinition Height="25"/>

<RowDefinition/>

</Grid.RowDefinitions>

<Label x:Name="LabUserFullName"

Padding="0"

FontSize="16"

Grid.Row="0"/>

<Label x:Name="LabUserPhone"

FontSize="16"

Padding="0"

Grid.Row="1"/>

<Label x:Name="LabUserRole"

Padding="0"

Foreground="Gray"

FontSize="14"

Grid.Row="2"/>

<Button x:Name="BtnLogout"

Content="Выйти"

Grid.Row="3"

Style="{DynamicResource BtnLogonStyle}"

Margin="40,5,0,0"

Width="100"

Height="28"

BorderBrush="#FF1ED272"

VerticalAlignment="Top"

Click="BtnLogout\_Click"/>

</Grid>

</Border>

Рисунок Б.9 – Программный код разметки макета главного окна

<Border x:Name="BorderSetSorting"

Style="{StaticResource MainBorderStyle}"

Visibility="Hidden"

Background="#FFFFFFFA"

BorderBrush="#FFD2D2D2"

Width="180"

Height="130"

HorizontalAlignment="Left"

VerticalAlignment="Top"

Margin="350,90,0,0">

<Grid x:Name="GridSetSorting"

Margin="10">

<Grid.RowDefinitions>

<RowDefinition/>

<RowDefinition/>

<RowDefinition/>

</Grid.RowDefinitions>

<RadioButton x:Name="RBReating"

Style="{DynamicResource RBSortingStyle}"

Content="По рейтингу"

Grid.Row="0"

Checked="RBReating\_Checked"/>

<RadioButton x:Name="RBMinCost"

Style="{DynamicResource RBSortingStyle}"

Content="По возрастанию цены"

Grid.Row="1"

Checked="RBMinCost\_Checked"/>

<RadioButton x:Name="RBMaxCost"

Style="{DynamicResource RBSortingStyle}"

Content="По убыванию цены"

Grid.Row="2"

Checked="RBMaxCost\_Checked"/>

</Grid>

</Border>

</Grid>

</Window>

Рисунок Б.10 – Программный код разметки макета главного окна

public Grid ToutListUpdate(string tourName, string countryCode, decimal minPrice, decimal maxPrice, DateTime date, int countDays, string type, Grid gridTourList)

{

var tourList = entities.sp\_SelTourByFilters(tourName, countryCode, minPrice, maxPrice, date, countDays, type);

int rowCount = gridTourList.RowDefinitions.Count;

if (rowCount != 0)

{

gridTourList.RowDefinitions.RemoveRange(0, rowCount);

}

gridTourList.Children.Clear();

Label LabTourList = new Label();

LabTourList.Margin = new Thickness(0, 9, 0, 0);

LabTourList.Content = "Список туров";

LabTourList.Padding = new Thickness(0);

LabTourList.FontSize = 24;

LabTourList.HorizontalAlignment = Horizontal

Alignment.Center;

LabTourList.FontWeight = FontWeights.Bold;

LabTourList.FontFamily = new FontFamily("Calibri");

LabTourList.Width = 150;

LabTourList.HorizontalContentAlignment = HorizontalAlignment.Left;

RowDefinition labRow = new RowDefinition();

gridTourList.RowDefinitions.Add(labRow);

labRow.Height = new GridLength(60);

Grid.SetColumn(LabTourList, 0);

Grid.SetRow(LabTourList, 0);

gridTourList.Children.Add(LabTourList);

int rows = 1;

foreach (var tour in tourList)

{

UCTour ucTour = new UCTour();

int price = Convert.ToInt32(tour.Price);

ucTour.Id = tour.TourId;

ucTour.Title = tour.Name;

ucTour.TicketCount = tour.TicketCount;

ucTour.Price = price.ToString();

ucTour.Date = tour.Date;

ucTour.Status = tour.Status;

ucTour.CountDays = tour.CountDays;

ucTour.Type = tour.Type;

ucTour.CountryCode = tour.CountryCode;

ucTour.Description = tour.Description;

var PlacesOfVists = entities.sp\_SelPlaceOfVisitsByTourId(tour.TourId);

double response = 0;

int countResponse = 0;

List<byte[]> imgs = new List<byte[]>();

foreach (var item in PlacesOfVists)

{

imgs.Add(item.Image);

}

ucTour.Images.AddRange(imgs);

var selectResponse = entities.sp\_SelResponesByTourId(tour.TourId);

foreach (var item in selectResponse)

{

response += item.Estimation;

countResponse++;

}

ucTour.Estimation = Math.Round(response / countResponse, 1);

ucTour.CountEstimation = countResponse;

RowDefinition row = new RowDefinition();

row.Height = new GridLength(250);

gridTourList.RowDefinitions.Add(row);

Grid.SetColumn(ucTour, 0);

Grid.SetRow(ucTour, rows);

rows++;

gridTourList.Children.Add(ucTour);

}

return gridTourList;

}

Рисунок Б.11 – Программный код метода для загрузки туров с фильтрами

public Grid UpdateGridWithSecondColumn(Grid gridTourList)

{

List<UIElement> toursList = new List<UIElement>();

for (int i = 0; i < gridTourList.Children.Count; i++)

{

toursList.Add(gridTourList.Children[i]);

}

int rowCount = gridTourList.RowDefinitions.Count;

if (rowCount != 0)

{

gridTourList.RowDefinitions.RemoveRange(0, rowCount);

}

gridTourList.Children.Clear();

if (toursList.Count > 0)

{

Label label = (Label)toursList[0];

Grid.SetColumnSpan(label, 2);

RowDefinition labRow = new RowDefinition();

labRow.Height = new GridLength(60);

gridTourList.RowDefinitions.Add(labRow);

Grid.SetRow(label, 0);

gridTourList.Children.Add(label);

int rows = 1;

int column = 0;

for (int tour = 1; tour < toursList.Count; tour++)

{

RowDefinition row = new RowDefinition();

row.Height = new GridLength(250);

gridTourList.RowDefinitions.Add(row);

Grid.SetColumnSpan(toursList[tour], 1);

Grid.SetColumn(toursList[tour], column);

Grid.SetRow(toursList[tour], rows);

column++;

tour++;

if (tour < toursList.Count)

{

gridTourList.Children.Add(toursList[tour - 1]);

Grid.SetColumn(toursList[tour], column);

Grid.SetRow(toursList[tour], rows);

rows++;

gridTourList.Children.Add(toursList[tour]);

column--;

}

else

{

tour--;

Grid.SetColumnSpan(toursList[tour], 2);

gridTourList.Children.Add(toursList[tour]);

}

}

}

return gridTourList;

}

Рисунок Б.12 – Программный код метода для группировки туров по двум столбцам

<Window x:Class="ToursApp.UI.Wins.WinCreatePlaceOfVisit"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:uc="clr-namespace:ToursApp.UI.UC"

xmlns:local="clr-namespace:ToursApp.UI.Wins"

mc:Ignorable="d"

Title="Турагентство Турист: Создание места для посещения"

Height="600"

Width="500"

ResizeMode="NoResize"

KeyUp="Window\_KeyUp"

Loaded="Window\_Loaded"

Icon="/Resources/Icons/Icon.png"

Closing="Window\_Closing">

<Grid>

<uc:UCHeader x:Name="UCHeader"

Title="Создание места"

Margin="-30,10,0,0"

HorizontalAlignment="Left"

VerticalAlignment="Top"/>

<Grid Margin="10,80,0,50">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="130px"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="50px"/>

<RowDefinition Height="50px"/>

<RowDefinition Height="240px"/>

<RowDefinition Height="80px"/>

</Grid.RowDefinitions>

<Label x:Name="LabPlaceOfVisitName"

Content="Название"

Grid.Column="0"

Grid.Row="0"/>

<TextBox x:Name="TBoxPlaceOfVisitName"

Text="{Binding Name}"

Margin="0,3,0,0"

Grid.Column="1"

Grid.Row="0"

Width="250"/>

<Label x:Name="LabPlaceOfVisitImage"

Content="Фотография"

Grid.Column="0"

Grid.Row="1"/>

<Button x:Name="BtnAddImage"

Content="Добавить"

Style="{DynamicResource BtnLogonStyle}"

FontSize="14"

Grid.Column="1"

Grid.Row="1"

Margin="10,10,0,0"

HorizontalAlignment="Left"

VerticalAlignment="Top"

Width="90"

Height="26"

Click="BtnAddImage\_Click"/>

Рисунок Б.13 – Программный код разметки макета окна создания места для посещения

<Button x:Name="BtnClearImage"

IsEnabled="False"

Content="Очистить"

Style="{DynamicResource ResourceKey=BtnLogonStyle}"

FontSize="14"

Grid.Column="1"

Grid.Row="1"

Margin="0,10,120,0"

HorizontalAlignment="Right"

VerticalAlignment="Top"

Width="90"

Height="26"

Click="BtnClearImage\_Click"/>

<Image x:Name="ImgPlaceOfVisitImage"

Source="{Binding Image}"

Grid.Column="1"

Grid.Row="2"

Width="220"

Height="220"

HorizontalAlignment="Left"

VerticalAlignment="Top"/>

<Label x:Name="LabPlaceOfVisitDescription"

Content="Описание"

Grid.Column="0"

Grid.Row="4"/>

<TextBox x:Name="TBoxPlaceOfVisitDescription"

Text="{Binding Description}"

Margin="0,3,0,0"

Grid.Column="1"

Grid.Row="3"

Width="250"

Height="60"/>

</Grid>

<Grid x:Name="GridFooter"

VerticalAlignment="Bottom"

Background="#FF4F93EF">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="190"/>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="51"/>

</Grid.RowDefinitions>

<Button x:Name="BtnBack"

Content="Назад"

Style="{DynamicResource BtnLogonStyle}"

Grid.Column="0"

Grid.Row="0"

Margin="20,10,0,0"

Width="100"

Click="BtnBack\_Click" />

<Button x:Name="BtnCreate"

IsEnabled="False"

Content="Создать"

Style="{DynamicResource BtnLogonStyle}"

Grid.Column="1"

Grid.Row="0"

Margin="0,10,0,0"

Width="120"

Click="BtnCreate\_Click" />

</Grid>

</Grid>

</Window>

Рисунок Б.14 – Программный код разметки макета окна создания места для посещения

private void BtnCreate\_Click(object sender, RoutedEventArgs e)

{

string countryCode = App.Current.Resources["Country\_Code"] + "";

placesOfVisit.Add(new PlOfVisit(TBoxPlaceOfVisitName.Text, ImgPlaceOfVisitImage.Source, img, countryCode, TBoxPlaceOfVisitDescription.Text, false, 0));

try

{

MessageBox.Show("Место для посещения успешно создано!", "Информация сохранена", MessageBoxButton.OK);

double left = this.Left;

double top = this.Top;

WinCreatePlaceOfVisit winCreatePlaceOfVisit = new WinCreatePlaceOfVisit(placesOfVisit, null, win);

App.Current.Resources["CloseTourWin"] = false;

this.Close();

winCreatePlaceOfVisit.ShowDialog();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

Рисунок Б.15 – Программный код метода создания места для посещения

public void AddImage()

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Изображения(\*.jpg,\*.jpeg,\*.png,\*.tiff,\*.bmp,\*.svg)|\*.jpg;\*.jpeg;\*.png;\*.tiff;\*.bmp;\*.svg";

if (openFileDialog.ShowDialog() == true)

{

img = File.ReadAllBytes(openFileDialog.FileName);

imageName = directory + @"\Data\Imgs\" + RandomString(10) + ".jpg";

while (File.Exists(imageName) == true)

{

imageName = directory + @"\Data\Imgs\" + RandomString(10) + ".jpg";

}

File.Copy(openFileDialog.FileName, imageName);

BitmapImage bitmapImage = new BitmapImage();

var uriSource = new Uri(imageName, UriKind.Absolute);

bitmapImage.BeginInit();

bitmapImage.CacheOption = BitmapCacheOption.OnLoad;

bitmapImage.UriSource = uriSource;

bitmapImage.EndInit();

ImgPlaceOfVisitImage.Source = bitmapImage;

bitmapImage.UriSource = null;

BtnAddImage.Content = "Изменить";

}

CheckImagePlaceOfVisit();

CheckFullnessFields();

}

Рисунок Б.16 – Программный код метода добавления фотографии

Приложение В

Скрипт базы данных

USE [TravelAgencyTourist]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Administrator](

[Id] [int] NOT NULL,

[UserId] [int] NOT NULL,

CONSTRAINT [PK\_Administrator] PRIMARY KEY CLUSTERED (

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Client](

[Id] [int] NOT NULL,

[UserId] [int] NOT NULL,

CONSTRAINT [PK\_Client] PRIMARY KEY CLUSTERED (

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Country](

[Code] [nchar](2) NOT NULL,

[Name] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Country] PRIMARY KEY CLUSTERED (

[Code] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[PersonalData](

[Id] [int] NOT NULL,

[FirstName] [nvarchar](50) NOT NULL,

[LastName] [nvarchar](50) NOT NULL,

[Patronymic] [nvarchar](50) NOT NULL,

[Passport] [nvarchar](10) NOT NULL,

[Phone] [nvarchar](12) NOT NULL,

CONSTRAINT [PK\_PersonalData] PRIMARY KEY CLUSTERED (

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

Рисунок В.1 – Скрипт базы данных TravelAgencyTourist

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[PlaceOfVisit](

[Id] [int] NOT NULL,

[Name] [nvarchar](50) NOT NULL,

[Image] [image] NOT NULL,

[CountryCode] [nchar](2) NOT NULL,

[Description] [nvarchar](1000) NOT NULL,

CONSTRAINT [PK\_PlaceOfVisit] PRIMARY KEY CLUSTERED (

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Response](

[Id] [int] NOT NULL,

[SignupTourId] [int] NOT NULL,

[Estimation] [int] NOT NULL,

[Description] [nvarchar](200) NULL,

[Date] [date] NOT NULL,

CONSTRAINT [PK\_Response] PRIMARY KEY CLUSTERED (

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Role](

[Id] [int] NOT NULL,

[Name] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Role] PRIMARY KEY CLUSTERED (

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY] ) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[SignupTour] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[SignupTour](

[Id] [int] NOT NULL,

[ClientId] [int] NOT NULL,

[TourId] [int] NOT NULL,

CONSTRAINT [PK\_SignupTour] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

Рисунок В.2 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: Table [dbo].[Tour] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Tour](

[Id] [int] NOT NULL,

[Name] [nvarchar](50) NOT NULL,

[TicketCount] [int] NOT NULL,

[Price] [money] NOT NULL,

[Date] [datetime] NOT NULL,

[StatusId] [int] NOT NULL,

[CountDays] [int] NOT NULL,

[Type] [nvarchar](50) NOT NULL,

[CountryCode] [nchar](2) NOT NULL,

[Description] [nvarchar](500) NOT NULL,

CONSTRAINT [PK\_Tour] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[TourAdministration] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[TourAdministration](

[TourId] [int] NOT NULL,

[AdministratorId] [int] NOT NULL,

CONSTRAINT [PK\_TourAdministration] PRIMARY KEY CLUSTERED

(

[TourId] ASC,

[AdministratorId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[TourPlaceOfVisit] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[TourPlaceOfVisit](

[TourId] [int] NOT NULL,

[PlaceOfVisitId] [int] NOT NULL,

CONSTRAINT [PK\_TourPlaceOfVisit] PRIMARY KEY CLUSTERED

(

[TourId] ASC,

[PlaceOfVisitId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[TourStatus] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[TourStatus](

[Id] [int] NOT NULL,

[Name] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_TourStatus] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

Рисунок В.3 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: Table [dbo].[User] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[User](

[Id] [int] NOT NULL,

[Login] [nvarchar](50) NOT NULL,

[Password] [nvarchar](max) NOT NULL,

[RoleId] [int] NOT NULL,

[PersonalDataId] [int] NOT NULL,

CONSTRAINT [PK\_User] PRIMARY KEY CLUSTERED (

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

ALTER TABLE [dbo].[Administrator] WITH CHECK ADD CONSTRAINT [FK\_Administrator\_User1] FOREIGN KEY([UserId])

REFERENCES [dbo].[User] ([Id])

GO

ALTER TABLE [dbo].[Administrator] CHECK CONSTRAINT [FK\_Administrator\_User1]

GO

ALTER TABLE [dbo].[Client] WITH CHECK ADD CONSTRAINT [FK\_Client\_User1] FOREIGN KEY([UserId])

REFERENCES [dbo].[User] ([Id])

GO

ALTER TABLE [dbo].[Client] CHECK CONSTRAINT [FK\_Client\_User1]

GO

ALTER TABLE [dbo].[PlaceOfVisit] WITH CHECK ADD CONSTRAINT [FK\_PlaceOfVisit\_Country] FOREIGN KEY([CountryCode])

REFERENCES [dbo].[Country] ([Code])

GO

ALTER TABLE [dbo].[PlaceOfVisit] CHECK CONSTRAINT [FK\_PlaceOfVisit\_Country]

GO

ALTER TABLE [dbo].[Response] WITH CHECK ADD CONSTRAINT [FK\_Response\_SignupTour] FOREIGN KEY([SignupTourId])

REFERENCES [dbo].[SignupTour] ([Id])

GO

ALTER TABLE [dbo].[Response] CHECK CONSTRAINT [FK\_Response\_SignupTour]

GO

ALTER TABLE [dbo].[SignupTour] WITH CHECK ADD CONSTRAINT [FK\_SignupTour\_Client] FOREIGN KEY([ClientId])

REFERENCES [dbo].[Client] ([Id])

GO

ALTER TABLE [dbo].[SignupTour] CHECK CONSTRAINT [FK\_SignupTour\_Client]

GO

ALTER TABLE [dbo].[SignupTour] WITH CHECK ADD CONSTRAINT [FK\_SignupTour\_Tour] FOREIGN KEY([TourId])

REFERENCES [dbo].[Tour] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[SignupTour] CHECK CONSTRAINT [FK\_SignupTour\_Tour]

GO

ALTER TABLE [dbo].[Tour] WITH CHECK ADD CONSTRAINT [FK\_Tour\_Country] FOREIGN KEY([CountryCode])

REFERENCES [dbo].[Country] ([Code])

GO

ALTER TABLE [dbo].[Tour] CHECK CONSTRAINT [FK\_Tour\_Country]

GO

ALTER TABLE [dbo].[Tour] WITH CHECK ADD CONSTRAINT [FK\_Tour\_TourStatus] FOREIGN KEY([StatusId])

REFERENCES [dbo].[TourStatus] ([Id])

GO

ALTER TABLE [dbo].[Tour] CHECK CONSTRAINT [FK\_Tour\_TourStatus]

GO

ALTER TABLE [dbo].[TourAdministration] WITH CHECK ADD CONSTRAINT [FK\_TourAdministration\_Administrator] FOREIGN KEY([AdministratorId])

REFERENCES [dbo].[Administrator] ([Id])

GO

ALTER TABLE [dbo].[TourAdministration] CHECK CONSTRAINT [FK\_TourAdministration\_Administrator]

GO

ALTER TABLE [dbo].[TourAdministration] WITH CHECK ADD CONSTRAINT [FK\_TourAdministration\_Tour] FOREIGN KEY([TourId])

REFERENCES [dbo].[Tour] ([Id])

ON DELETE CASCADE

GO

Рисунок В.4 – Скрипт базы данных TravelAgencyTourist

ALTER TABLE [dbo].[TourAdministration] CHECK CONSTRAINT [FK\_TourAdministration\_Tour]

GO

ALTER TABLE [dbo].[TourPlaceOfVisit] WITH CHECK ADD CONSTRAINT [FK\_TourPlaceOfVisit\_PlaceOfVisit] FOREIGN KEY([PlaceOfVisitId])

REFERENCES [dbo].[PlaceOfVisit] ([Id])

GO

ALTER TABLE [dbo].[TourPlaceOfVisit] CHECK CONSTRAINT [FK\_TourPlaceOfVisit\_PlaceOfVisit]

GO

ALTER TABLE [dbo].[TourPlaceOfVisit] WITH CHECK ADD CONSTRAINT [FK\_TourPlaceOfVisit\_Tour] FOREIGN KEY([TourId])

REFERENCES [dbo].[Tour] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[TourPlaceOfVisit] CHECK CONSTRAINT [FK\_TourPlaceOfVisit\_Tour]

GO

ALTER TABLE [dbo].[User] WITH CHECK ADD CONSTRAINT [FK\_User\_PersonalData] FOREIGN KEY([PersonalDataId])

REFERENCES [dbo].[PersonalData] ([Id])

GO

ALTER TABLE [dbo].[User] CHECK CONSTRAINT [FK\_User\_PersonalData]

GO

ALTER TABLE [dbo].[User] WITH CHECK ADD CONSTRAINT [FK\_User\_Role] FOREIGN KEY([RoleId])

REFERENCES [dbo].[Role] ([Id])

GO

ALTER TABLE [dbo].[User] CHECK CONSTRAINT [FK\_User\_Role]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_DelEstimatin] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_DelEstimatin] (

@SignupId int

)

AS BEGIN

DELETE Response WHERE SignupTourId = @SignupId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_DelSignupClient] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_DelSignupClient](

@SignupId int

)

AS BEGIN

DELETE Response WHERE Response.SignupTourId = @SignupId

DELETE SignupTour WHERE SignupTour.Id = @SignupId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_DelTour] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

Create Procedure [dbo].[sp\_DelTour]

(

@Tour\_id int

)

AS BEGIN

Delete Tour Where Tour.id = @Tour\_id

END

GO

Рисунок В.5 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_EditResponse] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_EditResponse](

@Id int,

@SignupId int,

@Estimation int,

@Description nvarchar(200),

@Date date

)

AS BEGIN

UPDATE Response Set SignupTourId = @SignupId, Estimation = @Estimation, [Description] = @Description, [Date] = @Date

WHERE SignupTourId = @SignupId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_GetLooseClientId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_GetLooseClientId]

AS BEGIN

DECLARE @Id int = 1

while (1 = 1)

BEGIN

if ((SELECT Id FROM Client Where id = @Id) = @Id)

BEGIN

SET @Id = @Id + 1

END

ELSE

BEGIN

SELECT @Id

BREAK

END

END

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_GetLoosePlaceOfVisitId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_GetLoosePlaceOfVisitId]

AS BEGIN

DECLARE @Id int = 1

while (1 = 1)

BEGIN

if ((SELECT Id FROM PlaceOfVisit Where id = @Id) = @Id)

BEGIN

SET @Id = @Id + 1

END

ELSE

BEGIN

SELECT @Id

BREAK

END

END

END

GO

Рисунок В.6 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_GetLooseResponseId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_GetLooseResponseId]

AS BEGIN

DECLARE @Id int = 1

while (1 = 1)

BEGIN

if ((SELECT Id FROM Response Where id = @Id) = @Id)

BEGIN

SET @Id = @Id + 1

END

ELSE

BEGIN

SELECT @Id

BREAK

END

END

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_GetLooseSugnupId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_GetLooseSugnupId]

AS BEGIN

DECLARE @Id int = 1

while (1 = 1)

BEGIN

if ((SELECT Id FROM SignupTour Where id = @Id) = @Id)

BEGIN

SET @Id = @Id + 1

END

ELSE

BEGIN

SELECT @Id

BREAK

END

END

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_GetLooseTourId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_GetLooseTourId]

AS BEGIN

DECLARE @Id int = 1

while (1 = 1)

BEGIN

if ((SELECT Id FROM Tour Where id = @Id) = @Id)

BEGIN

SET @Id = @Id + 1

END

ELSE

BEGIN

SELECT @Id

BREAK

END

END

END

GO

Рисунок В.7 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_GetResponseBySignupId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_GetResponseBySignupId](

@SignupId int

)

AS BEGIN

SELECT \* FROM Response

WHERE SignupTourId = @SignupId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_GetTourById] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_GetTourById](

@TourId int

)

AS BEGIN

SELECT \* From Tour WHERE Tour.Id = @TourId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SelDistinctCountDays] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SelDistinctCountDays]

AS BEGIN

SELECT DISTINCT CountDays From Tour

inner join TourStatus on Tour.StatusId = TourStatus.Id

WHERE TourStatus.[Name] = 'Запланирован'

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SelPersonalDataByUserId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SelPersonalDataByUserId](

@UserId int

)

AS BEGIN

SELECT [Login], FirstName, LastName, Patronymic, [Role].[Name], Passport, Phone From [User]

inner join PersonalData on [User].PersonalDataId = PersonalData.Id

inner join [Role] on [User].RoleId = [Role].Id

WHERE [User].Id = @UserId

end

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SelPlaceOfVisitIdList] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SelPlaceOfVisitIdList] (

@TourId int

)

AS BEGIN

Select DISTINCT PlaceOfVisitId From TourPlaceOfVisit

Where TourId != @TourId

END

GO

Рисунок В.8 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SelPlaceOfVisitsByTourId] \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SelPlaceOfVisitsByTourId]

(

@TourId int ) AS BEGIN

SELECT Id, [Name], [Image], PlaceOfVisit.CountryCode, [Description] From PlaceOfVisit inner join TourPlaceOfVisit on PlaceOfVisit.Id = TourPlaceOfVisit.PlaceOfVisitId WHERE TourPlaceOfVisit.TourId = @TourId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SelResponesByTourId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SelResponesByTourId]

( @TourId int

) AS BEGIN

SELECT Estimation, Response.[Description], Response.[Date], [Login] FROM Response

inner join SignupTour on Response.SignupTourId = SignupTour.Id

inner join Tour on SignupTour.TourId = Tour.Id

inner join Client on SignupTour.ClientId = Client.Id

inner join [User] on Client.[UserId] = [User].Id

WHERE Tour.Id = @TourId

ORDER BY [Date] DESC

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SelSignupsByClientId] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SelSignupsByClientId](

@ClientId int ) AS BEGIN

SELECT SignupTour.Id, ClientId, TourId,Tour.[Name], TicketCount, Price, [Date], TourStatus.[Name] as 'Status', CountDays, [Type], Tour.CountryCode, Tour.[Description] FROM SignupTour

inner join Tour on TourId = Tour.Id

inner join TourStatus on TourStatus.Id = Tour.StatusId

WHERE ClientId = @ClientId

ORDER BY SignupTour.Id DESC

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SelTourByFilters] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SelTourByFilters]

( @Name nvarchar(50),

@CountryCode nchar(2),

@MinPrice money,

@MaxPrice money,

@Date datetime,

@CountDays int,

@Type nvarchar(50)

) AS BEGIN

SELECT Tour.Id as 'TourId', Tour.[Name], TicketCount, Price, [Date], TourStatus.[Name] as 'Status', CountDays, [Type], Tour.CountryCode, Tour.[Description] FROM Tour

inner join TourStatus on Tour.StatusId = TourStatus.Id

WHERE (Tour.[Name] LIKE CONCAT('%', @Name, '%') OR @Name IS NULL) AND

(Tour.CountryCode = @CountryCode OR @CountryCode IS NULL) AND

(Tour.Price BETWEEN @MinPrice AND @MaxPrice OR @MinPrice IS NULL) AND

(Tour.[Date] >= @Date or @Date IS NULL) AND

(Tour.CountDays = @CountDays OR @CountDays = -1) AND

(Tour.[Type] = @Type OR @Type IS NULL) AND

(Tour.StatusId = 1)

END

GO

Рисунок В.9 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SelTours] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SelTours]

AS BEGIN

SELECT Tour.[Id], Tour.[Name] as TourName, TicketCount, [Price], [Date], CountDays, [Type], TourStatus.[Name] as 'Status', CountryCode, [Description], Country.[Name] as CountryName FROM Tour

inner join Country on Tour.CountryCode = Country.Code

inner join TourStatus on Tour.StatusId = TourStatus.Id

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetCancelTour] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetCancelTour] ( @id int,

@StatusId int ) AS BEGIN

UPDATE Tour Set StatusId = @StatusId

WHERE Id = @id

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetDecrementTicketCount] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetDecrementTicketCount](

@TourId int ) AS BEGIN

UPDATE Tour Set TicketCount = TicketCount - 1

WHERE Id = @TourId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetEditPlaceOfVisit] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetEditPlaceOfVisit]( @PlaceOfVisitId int,

@Name nvarchar(50),

@Image image,

@Description nvarchar(1000),

@Country\_Code nvarchar(2) ) AS BEGIN

Update PlaceOfVisit SET Id = @PlaceOfVisitId, [Name] = @Name, [Image] = @Image, CountryCode = @Country\_Code, [Description] = @Description

where Id = @PlaceOfVisitId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetEditTour] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetEditTour]

( @Id int,

@Name varchar(50),

@TicketCount int,

@Price money,

@Date datetime,

@CountOfDays int,

@Type nvarchar(50),

@Country\_Code nchar(2),

@Description nvarchar(500) ) AS BEGIN

UPDATE Tour SET Id = @Id, [Name] = @Name, TicketCount = @TicketCount, Price = @Price, [Date] = @Date, CountDays = @CountOfDays, [Type] = @Type, CountryCode = @Country\_Code, [Description] = @Description

WHERE Id = @Id

DELETE FROM TourPlaceOfVisit WHERE @Id = TourId

END

GO

Рисунок В.10 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetIncrementTicketCount] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetIncrementTicketCount](

@TourId int

)

AS BEGIN

UPDATE Tour Set TicketCount = TicketCount + 1

WHERE Id = @TourId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetNewClient] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetNewClient](

@ClientId int,

@UserId int )

AS BEGIN

INSERT INTO Client VALUES (@ClientId, @UserId)

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetNewPersonalData] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetNewPersonalData](

@PersonalId int,

@FirstName nvarchar(50),

@LastName nvarchar(50),

@Patronymic nvarchar(50),

@Phone nvarchar(12),

@Passport nvarchar(10)

) AS BEGIN

INSERT INTO PersonalData VALUES (@PersonalId, @FirstName, @LastName, @Patronymic, @Passport, @Phone)

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetNewUser] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetNewUser](

@UserId int,

@Login varchar(50),

@Passport varchar(2000),

@PersonalDataId int ) AS BEGIN

INSERT INTO [User] VALUES (@UserId, @Login, @Passport, 2, @PersonalDataId)

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetResponse] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetResponse](

@Id int,

@SignupId int,

@Estimation int,

@Description nvarchar(200),

@Date DateTime

) AS BEGIN

INSERT INTO Response Values (@Id, @SignupId, @Estimation, @Description, @Date)

END

GO

Рисунок В.11 – Скрипт базы данных TravelAgencyTourist

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetSignupTour] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetSignupTour](

@SignupId int,

@ClientId int,

@TourId int

)

AS BEGIN

INSERT INTO SignupTour VALUES (@SignupId, @ClientId, @TourId)

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetTourPlaceOfVisit] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE Procedure [dbo].[sp\_SetTourPlaceOfVisit]

(

@Tour\_id int,

@PlaceOfVisit\_id int

)

AS BEGIN

INSERT INTO TourPlaceOfVisit (TourId, PlaceOfVisitId) Values (@Tour\_id, @PlaceOfVisit\_id)

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[sp\_SetTourStatus] Script Date: 17.06.2024 14:13:21 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp\_SetTourStatus] (

@SystemDate DateTime

)

AS BEGIN

UPDATE Tour Set StatusId = 1

WHERE [Date] > @SystemDate AND StatusId != 4

UPDATE Tour SET StatusId = 2

WHERE @SystemDate BETWEEN [Date] AND DATEADD(day, Tour.CountDays, [Date]) AND StatusId != 4

UPDATE Tour SET StatusId = 3

WHERE @SystemDate > DATEADD(day, Tour.CountDays, [Date]) AND StatusId != 4

END

GO

Рисунок В.12 – Скрипт базы данных TravelAgencyTourist

Приложение Г

(обязательное)

Тест-кейсы

Таблица Г.1 – Тест-кейс №1

|  |  |
| --- | --- |
| Test Case # | TC\_1 |
| Приоритет теста | Высокий |
| Название тестирования | Проверка создания нового тура |
| Резюме испытания | В результате тестирования нужно убедиться в успешном создании тура |
| Шаги тестирования | 1. Ввести данные тура;  2. Выбрать доступное место для посещения из списка;  3. Нажать кнопку «Создать». |
| Данные тестирования | Администратор: Евгений Габриелян  Название тура: Ледниковый период 2024  Количество билетов: 125  Цена: 3000  Дата: 20.07.2024  Количество дней: 7  Тип: Машинный  Страна: Россия  Описание: Отсутствует  Место для посещения: Эльбрус  Дата и время: 17.06.2024 14:00 |
| Ожидаемый результат | Сообщение об успешном создании тура |
| Фактический результат | Сообщение об успешном создании тура |
| Предпосылки | Необходимо войти в систему за администратора |
| Постусловия | Отсутствует |
| Статус | Pass |
| Комментарии | Нет |

Таблица Г.2 – Тест-кейс №2

|  |  |
| --- | --- |
| Test Case # | TC\_2 |
| Приоритет теста | Высокий |
| Название тестирования | Проверка записи клиента на тур |
| Резюме испытания | В результате тестирования нужно убедиться в корректной записи клиента на выбранный тур |
| Шаги тестирования | 1. На одном из туров нажать на кнопку «Посмотреть».  2. Нажать кнопку «Оформить тур».  3. Нажать кнопку «Оформить».  4. Перейти на главное окно и нажать на кнопку «Оформленные туры». |
| Данные тестирования | Клиент: Шанаурин Андрей  Тур: Ледниковый период 2024  Дата и время: 17.06.2024 15:00 |
| Ожидаемый результат | Тур добавился в список оформленных клиентом туров |
| Фактический результат | Тур добавился в список оформленных клиентом туров |
| Предпосылки | Необходимо войти в систему за клиента |
| Постусловия | Отсутствует |
| Статус | Pass |
| Комментарии | Нет |

Таблица Г.3 – Тест-кейс №3

|  |  |
| --- | --- |
| Test Case # | TC\_3 |
| Приоритет теста | Высокий |
| Название тестирования | Проверка добавления отзыва о туре |
| Резюме испытания | В результате тестирования нужно убедиться в успешном добавлении отзыва в выбранном туре |
| Шаги тестирования | 1. Перейти на главное окно и нажать на кнопку «Оформленные туры».  2. Нажать кнопку «Добавить отзыв».  3. Нажать на кнопку «Добавить».  4. Нажать на кнопку «Посмотреть». |

Продолжение таблицы Г.3

|  |  |
| --- | --- |
| Данные тестирования | Клиент: Шанаурин Андрей  Тур: Ледниковый период 2024  Дата и время: 17.06.2024 15:30 |
| Ожидаемый результат | Отзыв добавился в список всех отзывов о туре |
| Фактический результат | Отзыв добавился в список всех отзывов о туре |
| Предпосылки | Необходимо войти в систему за клиента |
| Постусловия | Отсутствует |
| Статус | Pass |
| Комментарии | Нет |

Приложение Д

(обязательное)

Руководство пользователя

1. Введение
   1. Область применения

Данная программа предназначена для автоматизации работы турагентства, обеспечивая удобный доступ к управлению турами, местами для посещения и записями клиентов.

* 1. Краткое описание возможностей

Программа включает модули для администраторов, клиентов и посетителей. Основные функции включают составление туров и мест для посещения, просмотр информации о турах и оформление записей на составленные туры. Также в системе предусмотрена авторизация и регистрация пользователей.

* 1. Уровень подготовки пользователя

Для эффективного использования программы рекомендуется иметь базовые навыки работы с операционной системой, такой как Microsoft Windows, а также быть ознакомленным с основными принципами информационной безопасности.

* 1. Перечень эксплуатационной документации

Пользователь может быть ознакомлен с дополнительной документацией, включая руководства программиста и системного программиста, для более полного использования функционала программы.

2 Назначение и условия применения

2.1 Виды деятельности и функции

Программа предназначена для автоматизации следующих видов деятельности: создание, редактирования и отмена тура, создание, редактирование и добавление места для посещения в тур, а также запись клиентов на запланированный тур и оставление отзыва о проведённом туре.

2.2 Условия применения

Для корректной работы программы требуется установка на рабочие станции под управлением ОС Microsoft Windows 10 Pro (Education). Серверная часть должна функционировать на ОС Windows Server 2019 с установленной СУБД Microsoft SQL Server Express 2019 и SQL Server Management Studio 18.8. Пользователи должны иметь учетные записи с логинами и паролями.

3 Подготовка к работе

3.1 Состав и содержание дистрибутивного носителя данных

Для начала работы с системой необходимо получить программное обеспечение от системного администратора или ответственного лица.

Программное обеспечение может быть предоставлено на компакт-диске или USB-накопителе в соответствии с правами доступа пользователя.

3.2 Порядок загрузки данных и программ

Для установки программы необходимо загрузить установочный файл из браузера, запустить его и следовать инструкциям мастера установки. После завершения установки следует выполнить настройку подключения к базе данных, введя соответствующие параметры.

3.3 Порядок проверки работоспособности

После установки и настройки необходимо запустить программу и выполнить тестовую авторизацию. Проверить доступность основных функций: запись на тур, добавления отзыва о туре и управление турами.

4 Описание операций

4.1 Авторизация

Для авторизации в системе необходимо ввести логин и пароль в соответствующие поля на странице авторизации и нажать кнопку "Войти". После успешной авторизации пользователю будет доступен интерфейс в зависимости от его роли (администратор или клиент).

4.2 Регистрация

Для регистрации в системе необходимо посетителю заполнить поля логин, пароль, ФИО, номер телефона, паспорт и нажать кнопку "Зарегистрироваться". После успешной регистрации будет авторизирован в системе.

4.3 Фильтрация, сортировка и группировка туров

Для фильтрации туров посетитель, клиент или администратор должны ввести данные текстовые поля (название, цена), или выбрать данные из списочных полей (страна, дата, количество билетов и тип) и нажать на кнопку применить. Для сортировки необходимо выбрать один из трёх доступных видов сортировки туров (по рейтингу, возрастанию и убыванию цены). Для группировки необходимо выбрать один из двух доступных видов группировки: по одному столбцу, когда список туров будет отображаться одним столбцом, вне зависимости от текущего размера окна, и группировка по двум столбцам, когда в зависимости от размера окна список туров будет группироваться по одному или двум столбцам.

4.4 Оформление тура

Для оформления тура необходимо выбрать в главном окне соответствующий тур, после чего перейти оформлению тура, в открывшемся окне можно проверить корректность данных о клиенте и туре и оформить тур.

4.5 Добавление отзыва

Для добавления отзыва необходим хотя бы один завершённый, оформленный клиентом тур в списке оформленных туров, если такой есть, то клиент кликает на кнопку «Добавить отзыв», после чего переходит в окно добавления отзыва, в котором можно выбрать оценку тура от 1 до 5 и добавить описание отзыва и нажать на кнопку «Добавить», после чего отзыв будет добавлен в тур.

4.6 Удаление отзыва

Для удаления отзыва в списке оформленных туров необходим хотя бы один оформленный клиентом тур с раннее добавленным отзывом об этом туре, если такой есть, то клиент кликает на кнопку «Удалить отзыв», после чего отзыв о туре будет удалён.

4.7 Скачивание билета

Для скачивания билета в списке оформленных туров необходим хотя бы один запланированный и оформленный клиентом тур, если такой есть, то клиент нажимает на кнопку «Скачать билет», далее выбирает место загрузки билета и нажимает на кнопку «Сохранить», после чего билет начнёт загружаться в выбранное клиентом место на диске.

4.7 Создание тура

Для создания тура администратору в главном окне администратора необходимо нажать на кнопку «Создать», далее заполнить открывшиеся окно данными о туре и нажать на кнопку «Далее», после чего администратор может выбрать уже существующие места для посещения в выбранной стране, нажав на кнопку «Выбрать» и добавив одно или несколько мест для посещения в тур, или создать новое место для посещений, нажав на кнопку «Создать» и заполнив данными окно создания места для посещения, и после заполнения тура одним или несколькими местами для посещения нажать на кнопку «Создать», в результате чего тур будет создан, и выведется в список.

4.8 Редактирование тура

Для редактирования тура администратору в главном окне администратора необходимо в выбранном туре нажать на кнопку «Редактировать», после чего откроется окно редактирования тура, в котором можно изменить данные о туре, а также добавить новые или удалить уже существующие места для посещения в туре, после чего нажать на кнопку «Изменить», в результате чего изменения о данные тура и местах для посещения, прикреплённых к нему будут сохранены.

4.9 Отмена тура

Для отмены тура администратору в главном окне администратора необходимо выбрать один или несколько туров, после чего нажать на кнопку «Отменить», в результате чего выбранные туры будут отменены и не будут отображаться в списке туров в главном окне.

5 Аварийные ситуации

При возникновении ошибок при работе системы, не описанных ниже в данном разделе, необходимо обращаться к дежурному системному администратору.

Ошибка: требуется ввести действительный логин пользователя

- Описание ошибки: при авторизации введен несуществующий логин пользователя.

- Требуемые действия пользователя при возникновении ошибки: ввести существующий логин пользователя.

Ошибка: требуется ввести корректный пароль

- Описание ошибки: при авторизации введен неправильный пароль пользователя.

- Требуемые действия пользователя при возникновении ошибки: ввести правильный пароль пользователя.

Ошибка: сбой в электропитании рабочей станции

- Описание ошибки: нет электропитания рабочей станции или произошел

сбой в электропитании.

- Требуемые действия пользователя при возникновении ошибки:

перезагрузить рабочую станцию и повторить попытку входа в систему.

Ошибка: сбой локальной сети

- Описание ошибки: нет сетевого взаимодействия между рабочей станцией

и сервером системы.

- Требуемые действия пользователя при возникновении ошибки:

перезагрузить рабочую станцию и повторить попытку входа в систему, в случае неудачи обратиться к системному администратору.

6 Рекомендации по освоению

Для освоения программы рекомендуется начать с изучения данного руководства и выполнения контрольных примеров, включенных в документацию. Полезно провести обучение пользователей на примере выполнения основных операций, таких как просмотр туров, оформление записи на тур, составление туров и мест для посещения.

Приложение Е

(обязательное)

Руководство системного программиста

1 Общие сведения о программе

1.1 Назначение программы

Программная система «Турагентство Турист» предназначена для автоматизации туристического агентства ООО «Nature Tours».

1.2 Функции программы:

Основные функции включают:

- авторизация пользователей;

- регистрация посетителя;

- просмотр списка туров (для клиента, посетителя и администратора);

- просмотр информации о туре (для клиента, посетителя и администратора);

- оформление записи на тур (только для клиента);

- добавление, редактирование и удаление отзыва о туре (только для клиента);

- скачивание билета (только для клиента)

- составление туров (только для администратора);

- отмена туров (только для администратора);

- создание, редактирование и удаление мест для посещения (только для администратора).

1.3 Технические и программные средства

Состав технических средств предусматривает наличие компьютеров – рабочих станций и компьютера – сервера.

Компьютер-рабочая станция (для пользоваетелей) включает:

* процессор x64 с тактовой частотой, не менее 2,3 ГГц;
* оперативную память объемом, не менее 4 Гб;
* видеокарту, монитор, клавиатура и мышь.

Компьютер – сервер (для функционирования СУБД), включают в себя:

* процессор x64 с тактовой частотой, не менее 2,3 ГГц;
* оперативную память объемом, не менее 8 Гб;
* видеокарту, монитор, клавиатура и мышь.

2 Структура программы

2.1 Составные части программы

Система состоит из двух модулей:

– модуль работа с турами;

– модуль запись клиентов.

2.2 Связи между модулями

Модули программы взаимодействуют между собой через общую базу данных. Модуль работа с турами обеспечивает создание, редактирование и отмену туров, создание, редактирование и удаление мест для посещения, а также просмотр списка туров и информацию о каждом туре. Модуль запись клиентов обеспечивает оформление записей клиентов, отмену записей, скачивание билетов, а также добавление, редактирование и удаление отзывов о туре.

3. Настройка программы

3.1 Установка программы

Запустить установочный файл и следовать инструкциям мастера установки.

3.2 Установка и конфигурация сервера баз данных

Установить сервер баз данных и создать базу данных для хранения данных программы. Выполнить необходимую конфигурацию сервера баз данных.

3.3 Создание пользовательских учетных записей и назначение прав доступа

Создать учетные записи пользователей программы и назначить соответствующие права доступа в базе данных

4. Проверка программы

4.1 Тестирование функциональности

Проверка основных функций программы, включая создание документов прихода или расхода товаров, создание товаров, получение корректных ролей пользователей, выборка товаров, которые действительно содержатся в документе.

4.2 Тестирование безопасности

Проверка правильности разграничения доступа пользователей к данным и функциям программы.

5. Сообщения системному программисту

Сообщение: "Не удалось подключиться к базе данных. Проверьте параметры подключения."

- Описание: ошибка подключения к базе данных.

- Действия: проверьте правильность указанных параметров в конфигурационном файле и доступность сервера базы данных.

Сообщение: "Успешное подключение к базе данных."

- Описание: подключение к базе данных установлено успешно.

- Действия: продолжить настройку программы.

Сообщение: "Ошибка выполнения запроса к базе данных."

- Описание: произошла ошибка при выполнении запроса к базе данных.

- Действия: проверьте корректность запроса и состояние базы данных.

Сообщение: "Операция выполнена успешно."

- Описание: операция выполнена без ошибок.

- Действия: продолжить работу с программой