Содержание

Введение 3

1 Назначение и цели разработки 5

2 Разработка технического проекта на основе анализа требований 6

2.1 Определение спецификаций программного обеспечения 6

2.2 Проектирование модели данных и диаграммы классов 16

2.3 Конструирование прототипа 21

3 Реализация 23

3.1 Обоснование выбора средств разработки 23

3.2 Разработка базы данных в среде СУБД 24

3.3 Описание программных модулей 26

4 Тестирование программных модулей 31

4.1 Модульное тестирование 31

4.2 Интеграционное тестирование 32

5 Эксплуатационная документация 33

5.1 Руководство пользователя 33

Заключение 35

Список использованных источников 36

Приложение А Техническое задание. Требования к программным модулям 38

Приложение Б Программный код 42

Приложение В Формы выходных документов 45

Приложение Г Скрипт базы данных 46

Приложение Д Тестирование 48

Введение

В наше время ни одна крупная организация не может обойтись без баз данных. Базы данных необходимы для хранения всей необходимой информации, которая может понадобиться для работы предприятия. Для большинства предприятий целесообразно хранить информацию о работниках, клиентах и услугах или товаре. Также в БД можно хранить бухгалтерский учёт. Особенно в предприятиях с большим количеством отделов базы данных помогают справиться с большими объёмами информации.

Автоматизация работы с базами данных актуальна на сегодняшний день как никогда раньше. Так как данных для обработки становится больше, необходимы большие вычислительные мощности для более быстрого доступа и более быстрой обработки нужных данных. Проектирование программного продукта включает в себя нахождения способа решения этих проблем. Разработка программного комплекса предполагает также создание удобного для пользователя интерфейса.

Для обработки информации в базах данных используются системы управления базами данных. Назначением СУБД является хранение всей необходимой информации, которая может понадобиться для работы предприятия, а также обработка и сортировка этой информации, заключающаяся в выполнении всех необходимых запросов. В настоящий момент самыми распространёнными являются реляционные СУБД, управляющие реляционными базами данных. Реляционная база данных должна быть целостной, то есть информация в БД должна соответствовать её внутренней логике, структуре и всем явно заданным правилам. Реляционная модель ориентирована на организацию данных в виде двумерных таблиц

Авиакомпания оказывает все необходимые действия для продажи билета клиенту. Одной из важнейших и крупных проблем, существующих в авиакомпаниях - является автоматизация деятельности бухгалтера. Кассир

производит оформление билета для клиента и от его работы зависит удовлетворенность клиента качеством всего процесса оформления билета.

Курсовой проект включает в себя проектирование, разработку базы данных и настольного приложения в рамках информационной системы, предназначенной для автоматизации рабочих мест бухгалтера, менеджера, а также администратора.

1 Назначение и цели разработки

Заданием предусмотрена разработка программных модулей для автоматизации процесса составления расписания авиарейсов информационной системы авиакомпании. Разработанный модуль предназначен для автоматизации рабочего места бухгалтера и менеджера авиакомпании, а также для сбора, обработки и хранения информации о сотрудниках организации, клиентах, а также о проданных билетах.

Данная автоматизация позволит получить следующие преимущества по сравнению с нынешней реализацией данных задач:

1- быстрое формирование отчетов по уже проданным билетам за выбранный период;

2- быстрое оформление билетов на игровые матчи;

3- удобное добавление маршрутов и расписаний маршрутов.

Данная система позволит максимально улучшить и упростить действия человека в ряде важных задач, таких как добавление офиса, формирование отчетности, учет сотрудников организации, добавление расписания и всех необходимых для него данных.

2 Разработка технического проекта на основе анализа требований

2.1 Определение спецификаций программного обеспечения

Рассмотрим определение вариантов использования (прецедентов).

Система авиакрмпаеии требуется прежде всего следующим заинтересованным лицам:

- менеджер авиакомпании;
- бухгалтер авиакомпании;
- администратор системы.

На начальном этапе проектирования системы можно ограничиться двумя важными ролями действующих лиц: менеджер и бухгалтер. Соответственно основные прецеденты (варианты использования) для разрабатываемой системы следующие.

Прецеденты для менеджера:

- П1 – сформировать отчет о проданных билетах;
- П2 – добавить новый аэропорт с возможностью добавления страны;
- П3 – добавить новый маршрут;
- П4 – добавить новый самолет;
- П5 – добавить новый офис с возможностью добавления страны;
- П6 – добавить новое расписание;

Прецеденты для бухгалтера:

- П7 – сформировать отчет о проданных билетах;
- П8 – оформить продажу билета клиенту;
- П9 – добавить новый маршрут;
- П10 – добавить новое расписание;

Диаграмма вариантов использования, созданная средствами Microsoft Visio, для проектируемой системы представлена на рисунке 1 [2].

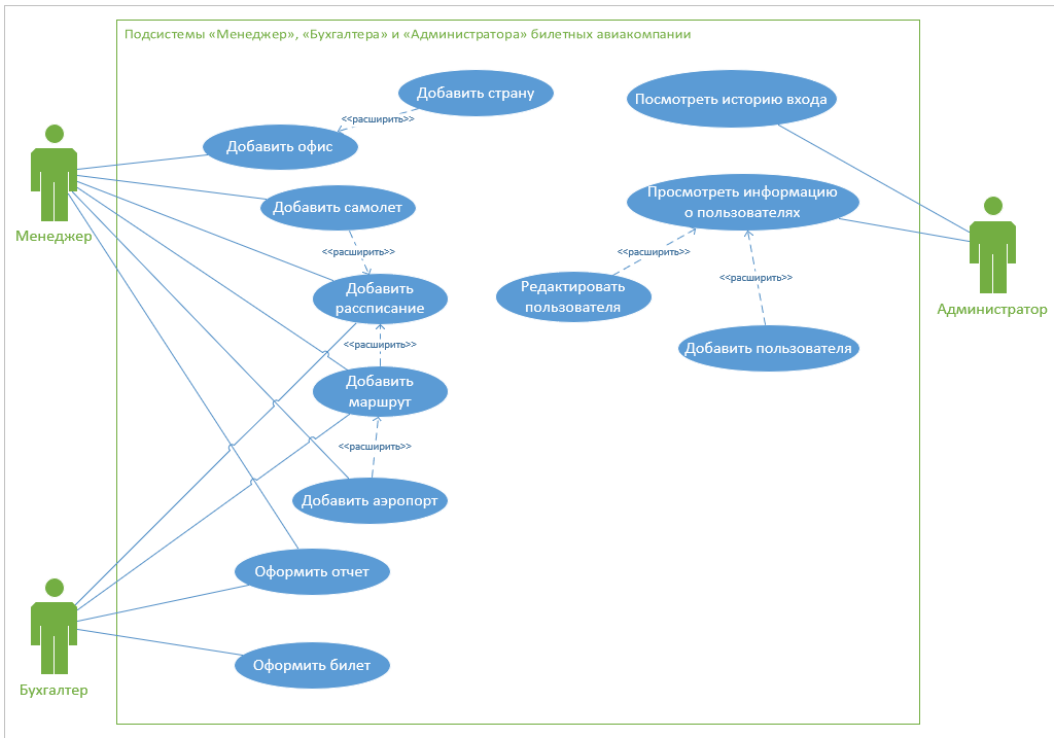

Рисунок 1 – Диаграмма вариантов использования программной системы авиакомпании

В таблице 1 представлено описание главного раздела сценария прецедента (варианта использования) «Оформить билет» [4].

Таблица 1 - Главный раздел сценария «Оформить билет»

Вариант использования	Оформить билет
Актеры	Бухгалтер
Краткое описание	Бухгалтер оформляет билет на определенное расписание полета
Цель	Оформление билета клиента
Тип	Базовый

В таблице 2 описана последовательность действий бухгалтера, приводящая к успешному выполнению прецедента (варианта использования) «Оформить билет»

Таблица 2 - Сценарий успешного выполнения варианта использования «Оформить билет»

Действия актеров	Отклик системы
1.Бухгалтер оформляет билет Исключение 1. В списке нет необходимого маршрута	2. Система возвращает список маршрутов
3. Бухгалтер сохраняет билет	4. Система сохраняет билет

В таблице 3 представлен сценарий обработки исключительных ситуаций для варианта использования «Оформить билет».

Таблица 3 - Обработка исключительных ситуаций для варианта использования «Оформить билет»

Действия актеров	Отклик системы
Исключение 1. В списке нет необходимого маршрута	
4. Бухгалтер добавляет новый маршрут	5. Система добавляет новый маршрут в базу
6. Бухгалтер добавляет расписание	7. Система сохраняет расписание
8. Бухгалтер оформляет билет	9. Система сохраняет билет

С помощью диаграммы деятельности представленной на рисунке 2, описан алгоритм реализации описанных сценариев для варианта использования «Оформить билет» [6].

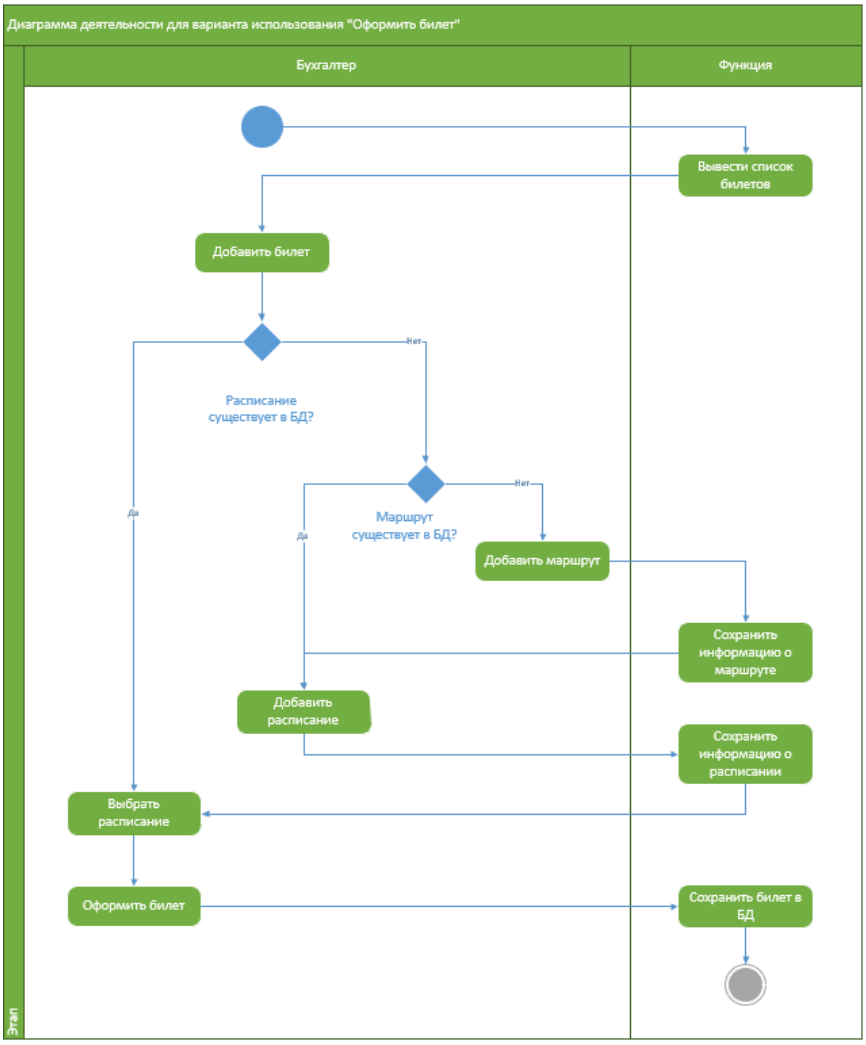

Рисунок 2 – Диаграмма деятельности для варианта использования «Оформить билет»

В таблице 4 представлено описание главного раздела сценария прецедента (варианта использования) «Сформировать отчёт».

На диаграмме последовательностей, представленной на рисунке 3, описаны требования реализации сценария «Оформить билет».

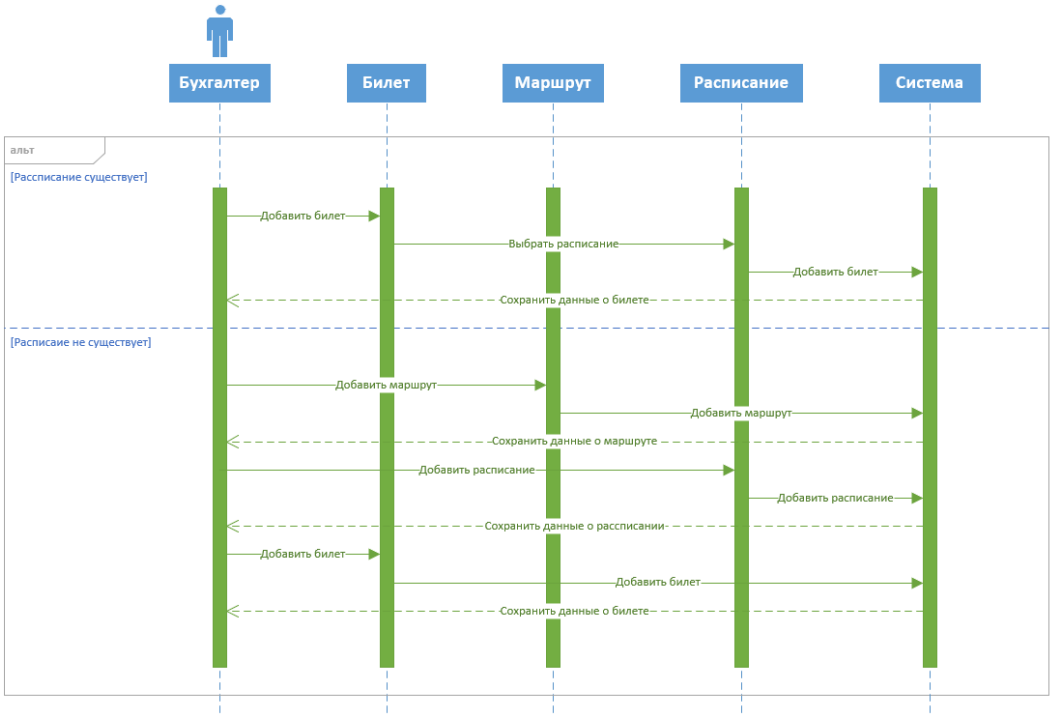

Рисунок 3 - Диаграмма последовательности для варианта использования «Оформить билет»

Таблица 4 - Главный раздел сценария «Сформировать отчет»

Вариант использования	Сформировать отчет
Актеры	Бухгалтер, менеджер
Краткое описание	Формирование отчета по проданным билетам
Цель	Сформировать отчет за выбранный период
Тип	Базовый
Ссылки на другие прецеденты	Отсутствуют

В таблице 5 описана последовательность действий бухгалтера, приводящая к успешному выполнению прецедента (варианта использования) «Сформировать отчет».

Таблица 5 - Сценарий успешного выполнения варианта использования «Сформировать отчет»

Действия актеров	Отклик системы
1.Бухгалтер или менеджер выбирает период Исключение 1. Неправильно выбран период Исключение 2. За выбранный период нет выполненных заказов	2. Система формирует отчет за выбранный период
3. Бухгалтер или менеджер сохраняет отчет	4. Система сохраняет отчет по выбранному пути в формате .pdf

В таблице 6 представлен сценарий обработки исключительных ситуаций для варианта использования «Сформировать отчет».

Таблица 6 - Обработка исключительных ситуаций для варианта использования «Сформировать отчет»

Действия актеров	Отклик системы
Исключение 1. Неправильно выбран период	
3. Бухгалтер или менеджер сохраняет отчет	4. Система выводит сообщение о том, что неправильно выбран период
Исключение 2. За выбранный период нет проданных билетов	
3. Бухгалтер или менеджер сохраняет отчет	4. Система выводит сообщение о том, что за выбранный период нет проданных билетов

С помощью диаграммы деятельности представленной на рисунке 4, описан алгоритм реализации описанных сценариев для варианта использования «Сформировать отчет».

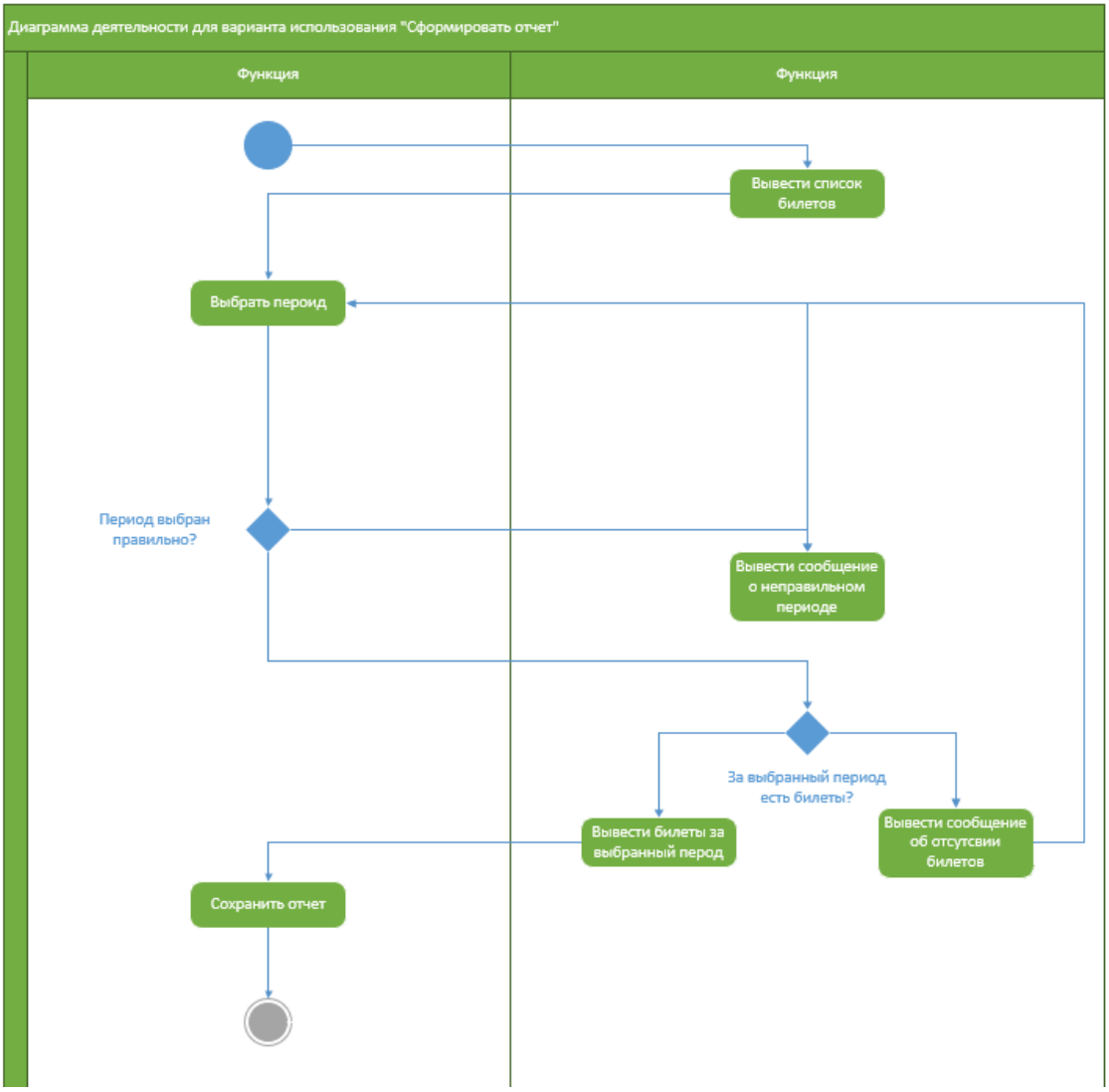

Рисунок 4 – Диаграмма деятельности для варианта использования «Сформировать отчет»

Таблица 7 - Главный раздел сценария «Добавить расписание»

Вариант использования	Сформировать отчет
Актеры	Бухгалтер, менеджер
Краткое описание	Бухгалтер, менеджер добавляет расписание полета
Цель	Добавить расписание полета
Тип	Базовый
Ссылки на другие прецеденты	Отсутствуют

В таблице 8 описана последовательность действий бухгалтера, приводящая к успешному выполнению прецедента (варианта использования) «Добавить расписание».

Таблица 8 - Сценарий успешного выполнения варианта использования «Добавить расписание»

Действия актеров	Отклик системы
1.Бухгалтер или менеджер выбирает маршрут Исключение 1. В списке нет необходимого маршрута	2. Система возвращает список маршрутов
3. Бухгалтер или менеджер выбирает самолет Исключение 2. В списке нет необходимого самолета	4. Система возвращает список самолетов
5. Бухгалтер или менеджер добавляет расписание	6. Система сохраняет расписание

Таблица 9 - Обработка исключительных ситуаций для варианта использования «Добавить расписание»

Действия актеров	Отклик системы
Исключение 1. В списке нет необходимого маршрута	
3. Бухгалтер или менеджер добавляет маршрут	4. Система сохраняет добавленный маршрут
5. Бухгалтер или менеджер выбирает самолет	6. Система сохраняет добавленное расписание
Исключение 2. В списке нет необходимого самолета	
5. Бухгалтер или менеджер добавляет самолет	6. Система сохраняет добавленный самолет
7. Бухгалтер или менеджер сохраняет добавленное расписание	8. Система сохраняет добавленное расписание

С помощью диаграммы деятельности представленной на рисунке 5, описан алгоритм реализации описанных сценариев для варианта использования «Сформировать отчет».

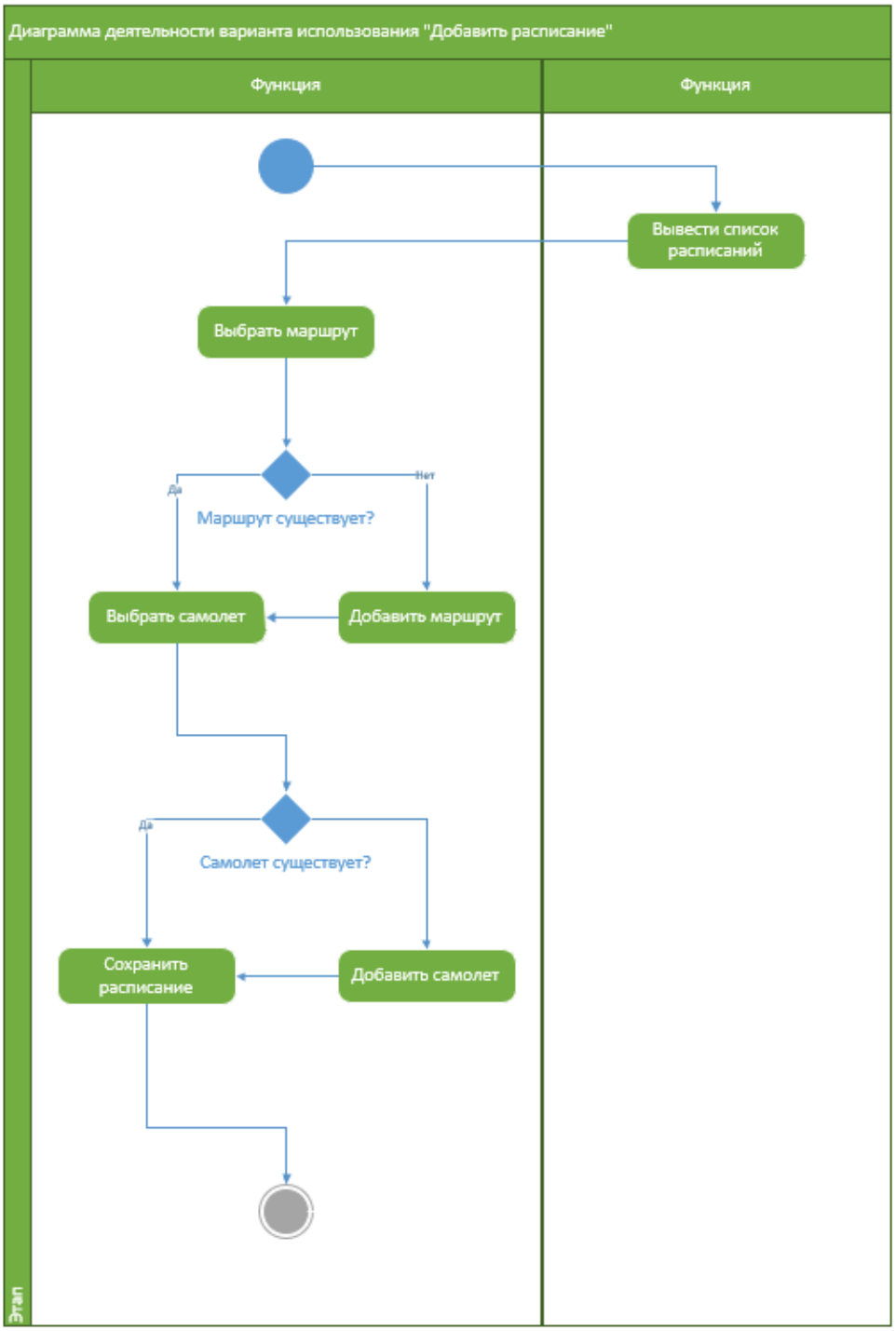

Рисунок 5 – Диаграмма деятельности для варианта использования «Добавить расписание»

Разработанные спецификации программного обеспечения на языке UML средствами MS Visio показывают основные действия пользователей автоматизируемой системы и алгоритмы их выполнения.

2.2 Проектирование модели данных и диаграммы классов

На основе анализа требований заказчика к функциям программной системы, требованиям к организации входных и выходных данных, с учетом спроектированных требований к реализации функций, описанных в диаграммах вариантов использования, деятельности и последовательности, была разработана модель данных системы и описана в виде ER-диаграммы, позволяющая четко описать требования к представлению логической структуры данных, на основе которой в последующем будет разработана физическая структура данных для хранения во внешней памяти и программной обработки.

На рисунке 6 представлена ER – диаграмма для программной системы «Авиакомпания» [6].

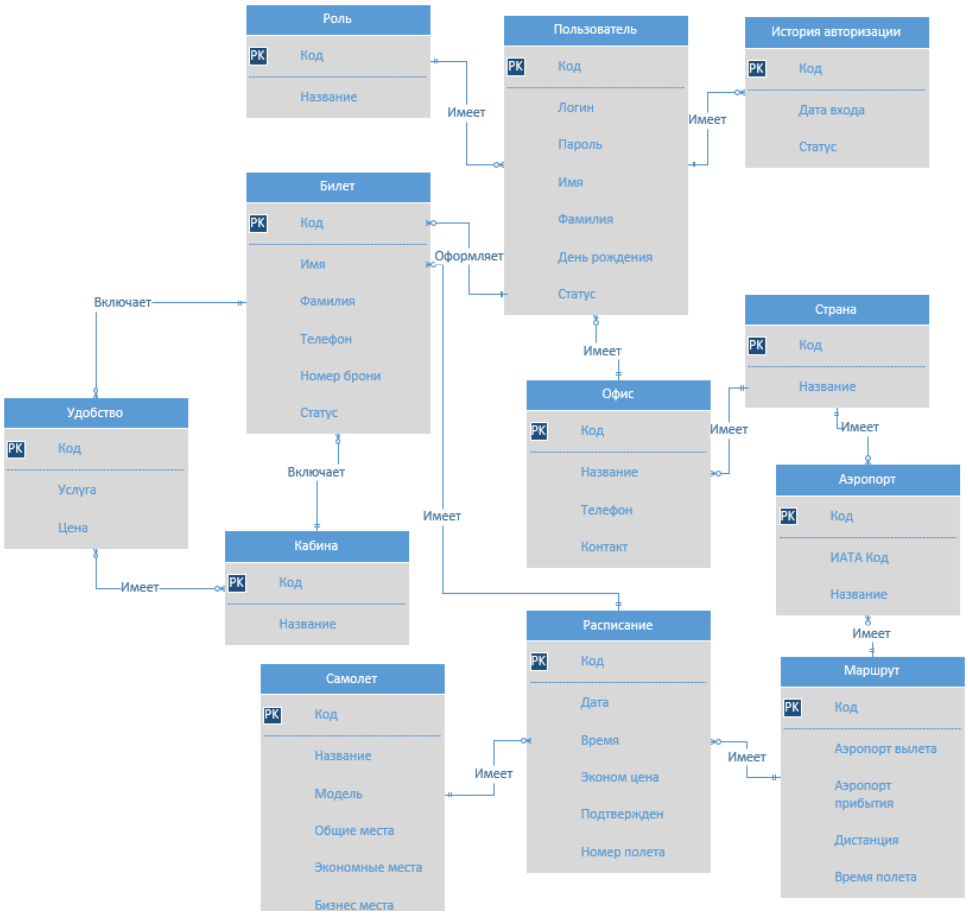

Рисунок 6 – ER-диаграмма для программной системы авиакомпании

В таблице 10 представлены сведения о сущностях и связях между сущностями ER-диаграммы из рисунка 6.

Таблица 10 – Характеристики сущностей и связей между сущностями ER-диаграммы для программной системы авиакомпании

Сущность	Назначение сущности	Ключ	Характеристики связей
Роль	Содержит данные о роли: - название.	Код роли (PK)	Роль –Пользователь (1..n)
Пользователь	Содержит информацию о пользователях: - логин; - пароль; - имя; - фамилия; - день рождения; - статус.	Код пользователя (PK)	Роль –Пользователь (1..n) Пользователь – История авторизации (1..n) Пользователь – Офис (n..1) Пользователь – Билеты (1..n)
История авторизации	Содержит данные об авторизации: - дата входа; - статус.	Код пользователя (PK)	Пользователь – История авторизации (1..n)
Офис	Содержит информацию об офисах: - название; - телефон; - контакт.	Код (PK)	Пользователь – Офис (n..1) Офис – Страна (n..1)
Страна	Содержит данные о странах: - название.	Код страны (PK)	Офис – Страна (n..1) Аэропорт – Страна (n..1)

Продолжение таблицы 10

Аэропорт	Содержит данные об аэропорту: - ИАТА код; - название.	Код аэропорта (PK)	Аэропорт – Страна (n..1) Аэропорт – Маршрут (1..n) Аэропорт – Маршрут (1..n)
Маршрут	Содержит данные о маршруте: - аэропорт вылета; - аэропорт прилета; - дистанция; - время полета.	Код маршрута (PK)	Аэропорт – Маршрут (1..n) Аэропорт – Маршрут (1..n) Маршрут – Расписание (1..n)
Расписание	Содержит данные о расписании: - дата; - время; - эконом цена; - подтвержден; - номер полета.	Код расписания (PK)	Маршрут – Расписание (1..n) Самолет – Расписание (1..n) Расписание – Билет (1..n)
Самолет	Содержит данные о самолете: - название; - модель; - общие места; - экономные места; - бизнес места.	Код самолета (PK)	Самолет – Расписание (1..n)
Билет	Содержит данные о билетах: - фамилия; - имя; - телефон; - номер паспорта; -номер брони; - статус.	Код билета (PK)	Пользователь – Билеты (1..n) Расписание – Билет (1..n) Кабина – Билет (1..n) Удобство – Билет (n..n)

Продолжение таблицы 10

Удобство	Содержит данные об удобстве: - услуга; - цена.	Код удобства (PK)	Удобство – Билет (n..n) Удобство – Кабина (n..n)
Кабина	Содержит данные о кабине: - название.	Код кабины (PK)	Удобство – Кабина (n..n) Кабина – Билет (1..n)

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 методов классов, правил их взаимодействия.

На рисунке 6 представлена диаграмма классов для программной системы авиакомпании [5].

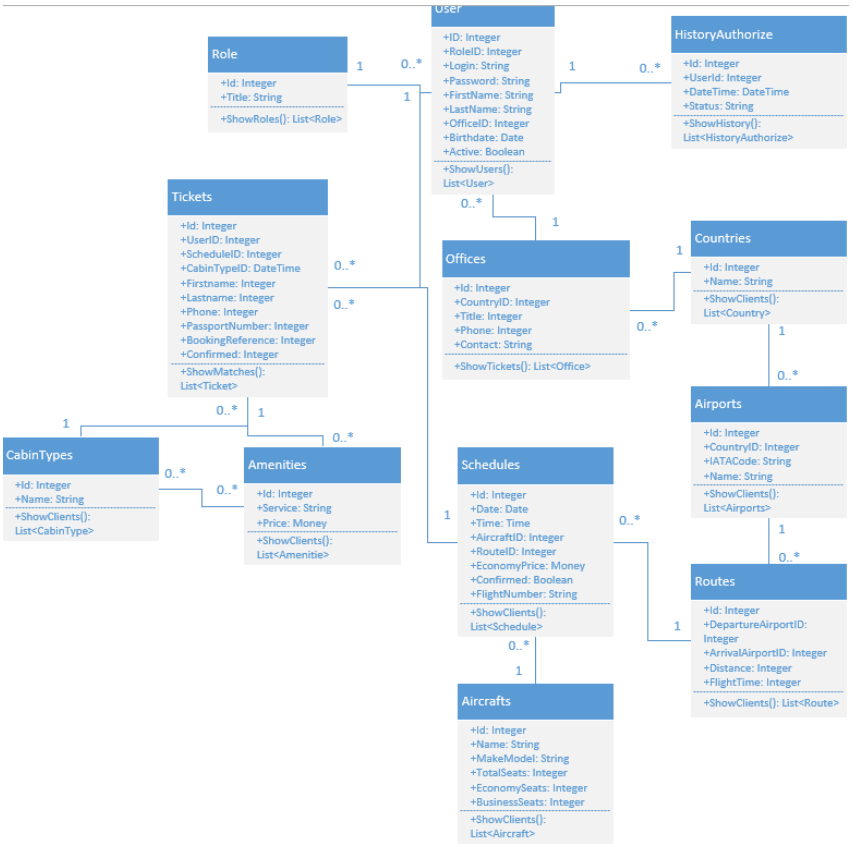

Рисунок 7 –Диаграмма классов для программной системы авиакомпании

Данная диаграмма классов описывает типы объектов системы авиакомпании и статические отношения, которые существуют между ними, а также отображаются операции классов.

Таблица 11 – Характеристики сущностей и связей между сущностями диаграммы классов для программной системы авиакомпании

Имя класса	Атрибуты класса	Операции класса
User	ID, RoleID, Login, Password, FirstName, LastName, OfficeID, Birthdate, Active	+ShowUsers(): List<User>
Role	ID, Title	+ShowUsers():List<Role>
HistoryAuthorize	ID, UserID, DateTime, Status	+ShowUsers(): List<HistoryAuthorize>
Offices	ID, CountryID, Title, Phone, Contact	+ShowUsers():List<Office>
Tickets	ID, UserID, ScheduleID, CabinTypeID, Firstname, Lastname, Phone, PassportNumber, BookingReference, Confirmed	+ShowUsers():List<Ticket>
Countries	ID, Name	+ShowUsers(): List<Countries>
CabinTypes	ID, Name	+ShowUsers(): List<CabinType>
Amentites	ID, Service, Price	+ShowUsers(): List<Amentites>
Schedules	ID, Date, Time, AircraftID, RouteID, EconomyPrice, Confirmed, FligthNumber	+ShowUsers(): List<Schedule>
Airports	ID, CountryID, IATACode, Name	+ShowUsers(): List<Airports>

Продолжение таблицы 11

Aircraft	ID, Name, MakeModel, TotalSeats, EconomySeats, BusinessSeats	+ShowUsers(): List<Aircraft>
Routes	ID, DepartureAirportID, ArrivalAirportID, Distance, FligthNumber	+ShowUsers(): List<Route>

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;

– расположение и выравнивание элементов;

– последовательный переход фокуса по элементам интерфейса;

– общая компоновка логична, понятна и проста в использовании;

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

– соответствующий заголовок на каждом окне приложения.

Основные требования руководства по стилю:

– все экранные формы пользовательского интерфейса должны иметь заголовок с логотипом, представленном на рисунке 7. Цвет, размер и пропорции логотипа не должны изменяться при использовании программы. Так же для приложения должна быть установлена иконка, соответствующая логотипу;

– тип шрифта – times new roman;

– цветовая схема предусматривает использование в качестве основного фона – белый цвет RGB (47, 49, 54), в качестве дополнительного – цвет RGB (55, 57, 61), для акцентирования внимания пользователя на целевое действие интерфейса – цвет RGB (126, 164, 150) или RGB(163, 14, 29).

Рисунок 8 – Логотип приложения

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

На рисунке 9 представлен прототип главного окна программы.

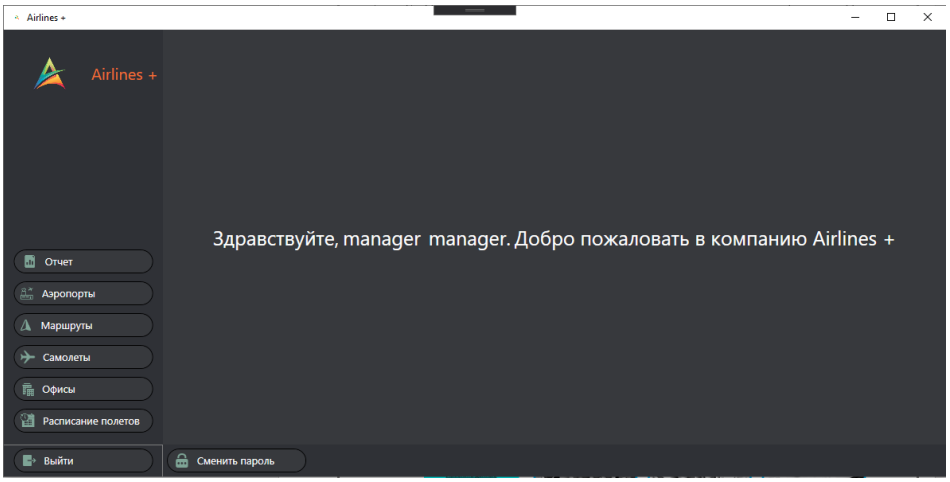

Рисунок 9 – Прототип главного окна программы

На рисунке 9 наблюдается, что окно соответствует вышеописанному руководству по стилю.

3 Реализация

3.1 Обоснование выбора средств разработки

В таблице 12 представлены некоторые возможные средства разработки для автоматизации авиакомпании.

Таблица 12 – Возможные средства разработки.

Язык программирования и технология	Плюсы	Минусы
1С:Предприятие	Гибкая настройка отчетов и группировок, настройка отображения списков «под себя» и сохранение этих настроек	Недостаточный уровень безопасности, необходимо платить за обновления, сложность внедрения
C#, .NET	Использует ООП везде, удобный конструктор форм, большое количество библиотек и шаблонов	Использует JIT компиляцию, приоритетная ориентированность на платформу Windows

C# и платформа .NET позволяет в короткие сроки создать настольное приложение с удобным графическим интерфейсом и работой с данными. Также имеет хороший инструменты для генерирования кода, что упрощает разработку.

Приложения на 1С создаются по определенным шаблонам, поэтому нельзя создать специфичный продукт.

Так как в техническом задании (см. Приложение А) указано, что система должна работать под операционной системой Windows, то выбор останавливается на языке программирования C# и платформе .NET Framework.

3.2 Разработка базы данных в среде СУБД

На основе ранее спроектированной ER-диаграммы в среде Microsoft SQL Server Management Studio 18 была разработана база данных AirlineCompany. На рисунке 10 представлена диаграмма разработанной базы данных [4].

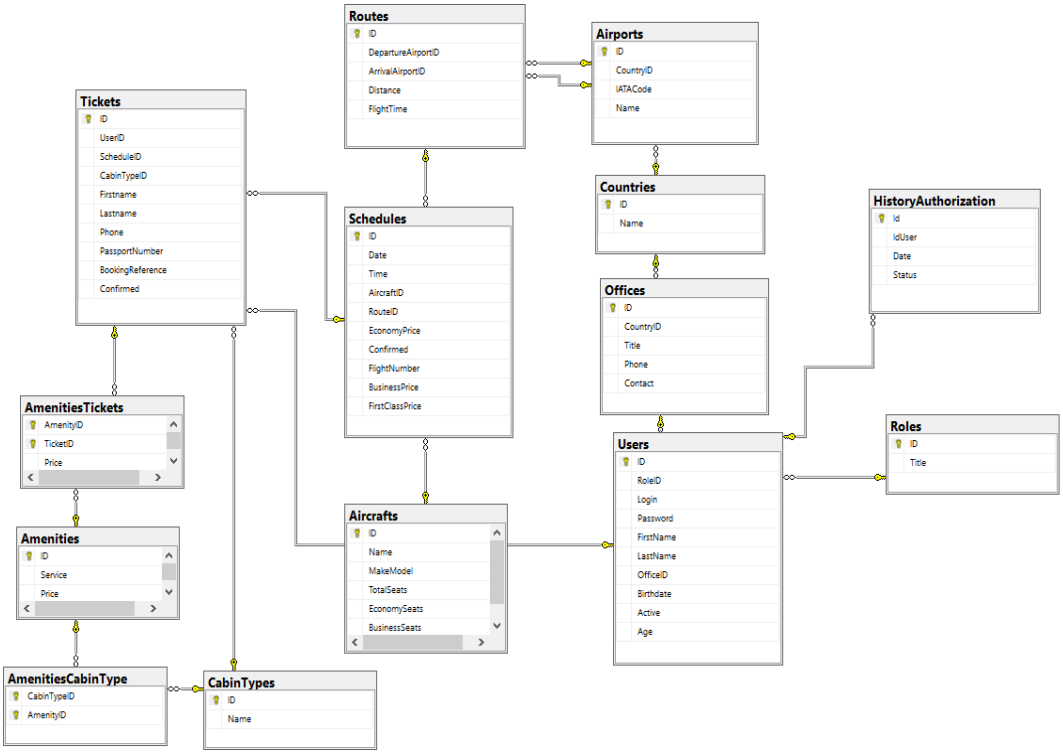

Рисунок 10 – Диаграмма базы данных AirlineCompany

В таблице 10 представлено описание таблиц базы данных AirlineCompany.

Таблица 13 – Описание таблиц базы данных Airline

Название таблицы	Краткое описание
Users	Предназначена для хранения информации о пользователях
Roles	Предназначена для хранения информации о ролях
HistoryAuthorization	Хранит историю авторизаций
Offices	Предназначена для хранения информации об офисах
Countries	Предназначена для хранения информации о странах
Airports	Предназначена для хранения информации об аэропортах
Routes	Предназначена для хранения информации о маршрутах
Schedules	Предназначена для хранения информации о странах
Aircrafts	Предназначена для хранения информации о самолетах
Tickets	Предназначена для хранения информации о билетах
AmenitiesTickets	Промежуточная таблица для хранения удобств билета
Amenities	Предназначена для хранения информации о удобствах
AmenitiesCabinType	Промежуточная таблица для хранения удобств класса полета

Продолжение таблицы 13

CabinTypes	Предназначена для хранения информации о классах полета

Скрипт разработанной базы со всеми данными находится в Приложении Г.

3.3 Описание программных модулей

На рисунке 11 представлена информация о структуре проекта приложения

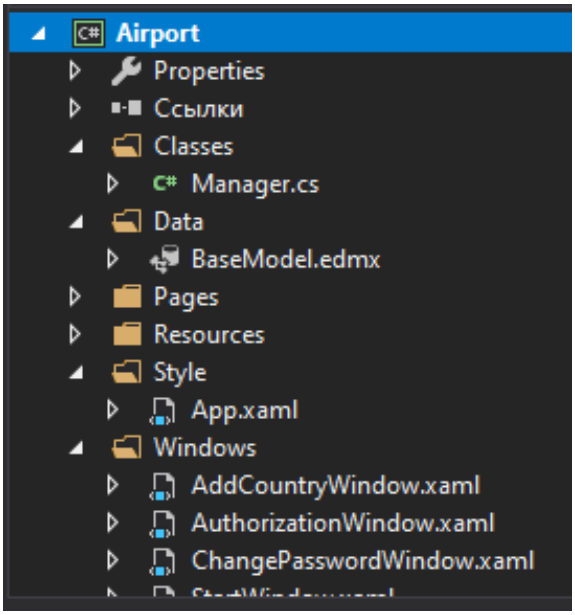

Рисунок 11 – Структура проекта

В таблице 14 описаны назначения директорий из проекта приложения.

Таблица 14 – Назначение директорий

Название директории	Назначение
Data	Хранит модель базы данных

Продолжение таблицы 14

Resources	Хранит графические ресурсы
Classes	Хранит разработанные инструменты
Pages	Хранит все страницы приложения
Style	Хранит файл со стилями
Windows	Хранит все окна приложения

На рисунке 12 представлена экранная форма модуля «Редактировать клиента»

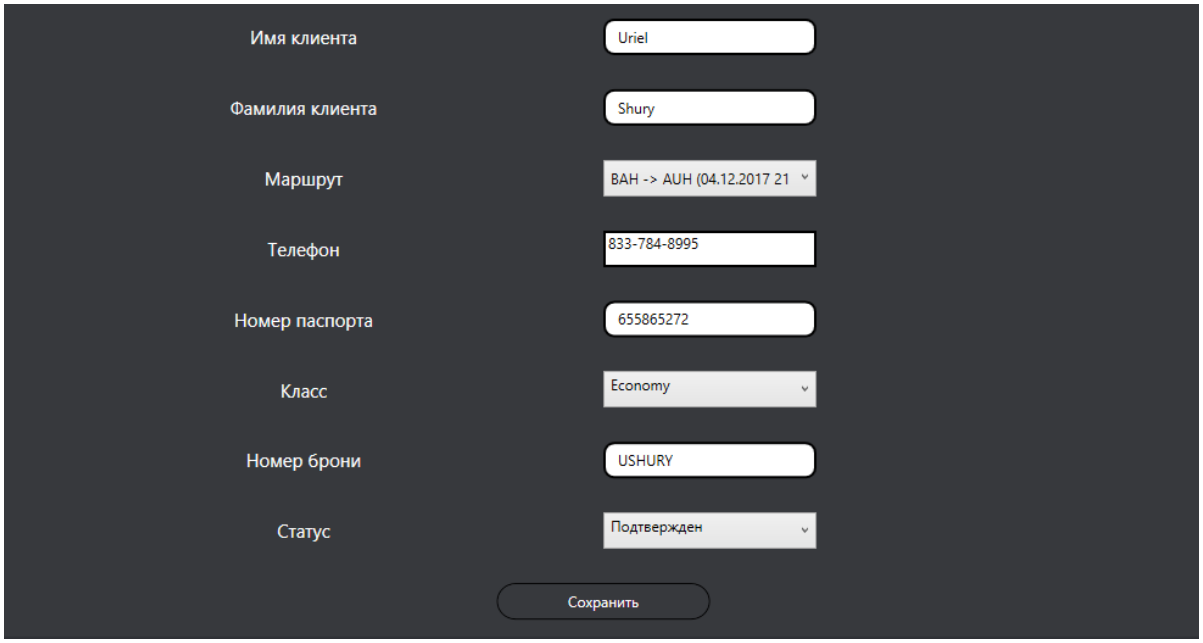

Рисунок 12 – Форма модуля «Редактировать билет»

Код разметки окна, представленного на рисунке 12 представлен на рисунке Б.1.

Код обработчика фильтрации btnApplyParameters_Click (Schedule.cs) для комбо боксов представлен на рисунке Б.3

Результат работы фильтрации представлен на рисунке 13

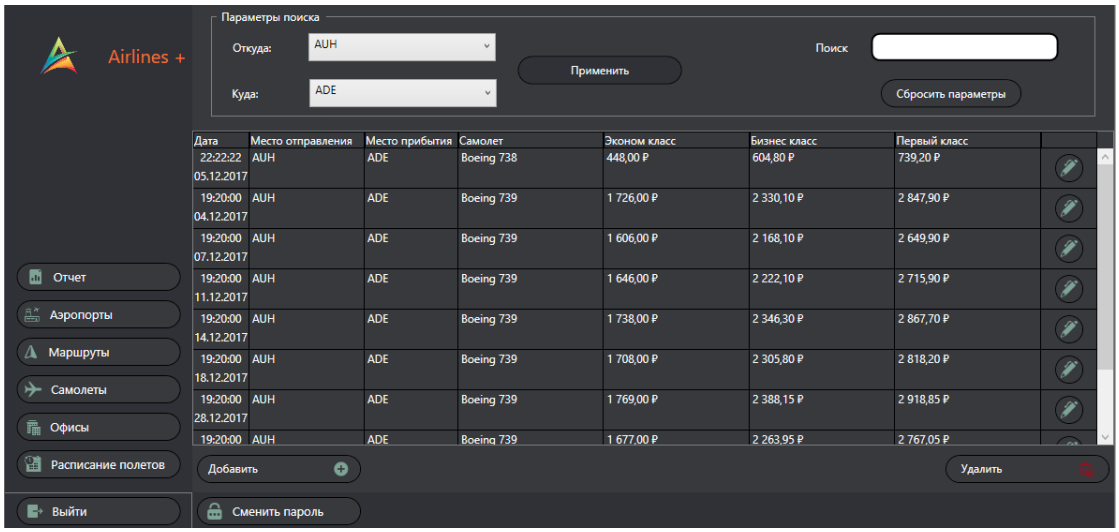

Рисунок 13 – Результат работы фильтрации

Код обработчика txtBoxSearch_TextChanged (Ticket.cs) для текстового поля поиска представлен на рисунке Б.2.

Результат работы поиска представлен на рисунке 14

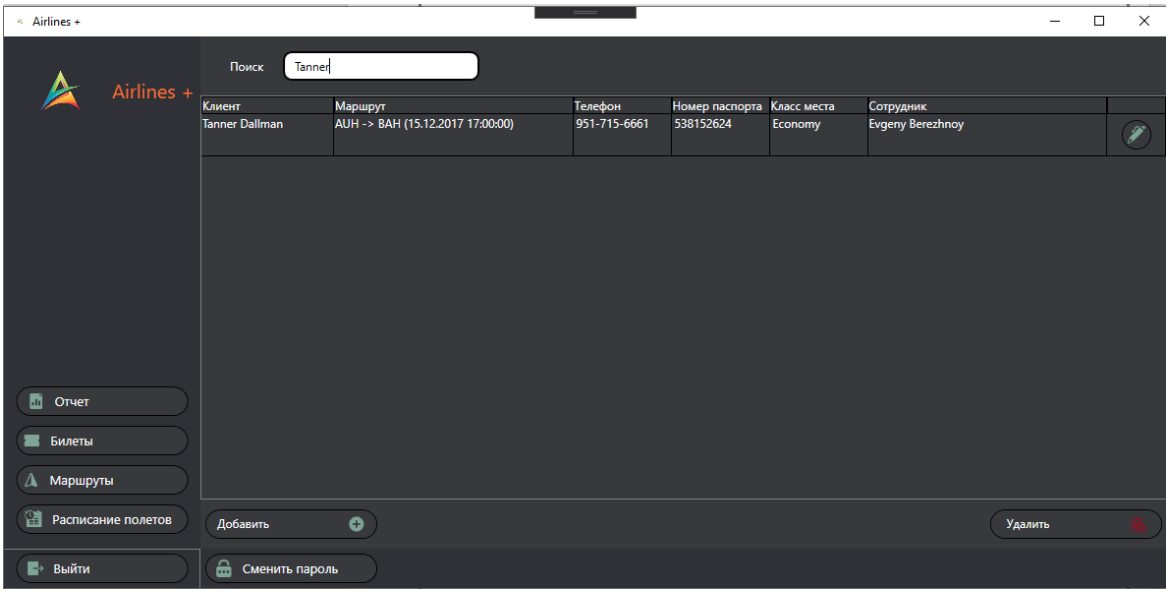

Рисунок 14 – Результат работы поиска

Окно для формирования отчета представлено на рисунке 15.

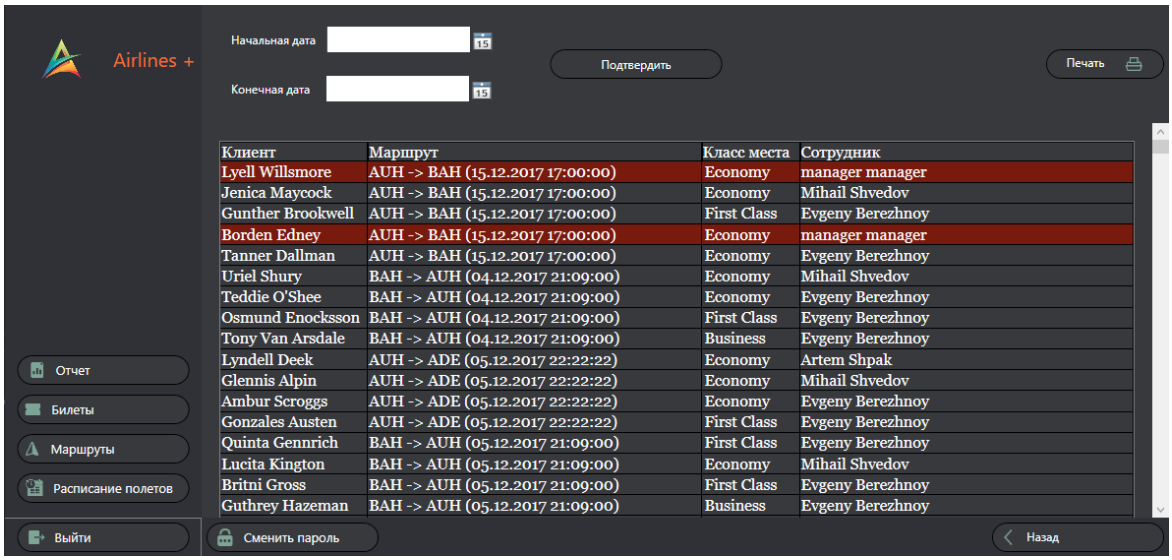

Рисунок 15 – Окно формирования отчетов

Код разметки окна, представленного на рисунке 14 представлен на рисунке Б.4, Б.5.

Образец выходного документа представлен на рисунке 16.

Клиент	Маршрут	Класс места	Сотрудник
Lyell Willsmore	AUH -> BAH (15.12.2017 17:00:00)	Economy	manager manager
Jenica Maycock	AUH -> BAH (15.12.2017 17:00:00)	Economy	Mihail Shvedov
Gunther Brookwell	AUH -> BAH (15.12.2017 17:00:00)	First Class	Evgeny Berezhnoy
Borden Edney	AUH -> BAH (15.12.2017 17:00:00)	Economy	manager manager
Tanner Dallman	AUH -> BAH (15.12.2017 17:00:00)	Economy	Evgeny Berezhnoy
Uriel Shury	BAH -> AUH (04.12.2017 21:09:00	Economy	Mihail Shvedov
Teddie O'Shee	BAH -> AUH (04.12.2017 21:09:00	Economy	Evgeny Berezhnoy
Osmund Enocksson	BAH -> AUH (04.12.2017 21:09:00	First Class	Evgeny Berezhnoy
Tony Van Arsdale	BAH -> AUH (04.12.2017 21:09:00	Business	Evgeny Berezhnoy
Lyndell Deek	AUH -> ADE (05.12.2017 22:22:22	Economy	Artem Shpak
Glennis Alpin	AUH -> ADE (05.12.2017 22:22:22	Economy	Mihail Shvedov
Ambur Scroggs	AUH -> ADE (05.12.2017 22:22:22	Economy	Evgeny Berezhnoy
Gonzales Austen	AUH -> ADE (05.12.2017 22:22:22	First Class	Evgeny Berezhnoy
Quinta Gennrich	BAH -> AUH (05.12.2017 21:09:00	First Class	Evgeny Berezhnoy
Lucita Kington	BAH -> AUH (05.12.2017 21:09:00	Economy	Mihail Shvedov
Britni Gross	BAH -> AUH (05.12.2017 21:09:00	First Class	Evgeny Berezhnoy
Guthrey Hazeman	BAH -> AUH (05.12.2017 21:09:00	Business	Evgeny Berezhnoy
Cad Bemand	BAH -> AUH (05.12.2017 21:09:00	Economy	Artem Shpak
Domeniga Slobom	BAH -> AUH (05.12.2017 21:09:00	Economy	Evgeny Berezhnoy
Sheridan Ipwell	BAH -> AUH (05.12.2017 21:09:00	Economy	Mihail Shvedov

Рисунок 16 – Образец отчета

Другие образцы выходных документов находятся на рисунках В.1 и В.2.

4 Тестирование программных модулей

4.1 Модульное тестирование

Задачей тестирования является проверка метода IsValidLogin(), который проверяет логин на правильное содержание [3]. Тест-кейсы представлены в Приложении Д.

На рисунке 17 представлен результат позитивного теста.

```
[TestMethod]
Ссылок: 0
public void PositiveTest()
{
    string positivePassword = "arkov632";
    Assert.AreEqual(true, Manager.IsValidLogin(positivePassword));
}
```

Рисунок 17 – Результат позитивного теста

На рисунке 18 представлен результат негативного теста.

```
[TestMethod]
Ссылок: 0
public void NegativeTest()
{
    string negativePassword = "arkov";
    Assert.AreEqual(true, Manager.IsValidLogin(negativePassword));
}
```

Рисунок 18 – Результат негативного теста

Негативный тест заключался в том, чтобы проверить, как отработает метод IsValidLogin(), если логин не соответствует нужному содержанию.

4.2 Интеграционное тестирование

5 Эксплуатационная документация

5.1 Руководство пользователя

Руководство пользователя для менеджера: при открытии приложения появляется окно авторизации, в которое необходимо ввести свои учетные данные; после успешной авторизации появляется меню с возможностью выбора просмотра страниц расписание полетов, офисы, самолеты, маршруты, аэропорты, отчет; при выборе расписание полетов появится список с существующими расписаниями полетов с возможностью добавления, редактирования и удаления; при выборе офисов появится список с существующими офисами с возможностью добавления, редактирования и удаления; при выборе самолетов появится список с существующими самолетами с возможностью добавления, редактирования и удаления; при выборе маршрутов появится список с существующими маршрутами с возможностью добавления, редактирования и удаления; при выборе аэропортов появится список с существующими аэропортами с возможностью добавления, редактирования и удаления; при выборе отчет появится список с проданными билетами, возможность найти билеты по определенной дате и распечатать текущий список проданных билетов; также существует возможность изменить свои персональные и учетные данные.

Руководство пользователя для бухгалтера: при открытии приложения появляется окно авторизации, в которое необходимо ввести свои учетные данные; после успешной авторизации появляется меню с возможностью выбора просмотра страниц расписание полетов, маршруты, билет, отчет; при выборе расписание полетов появится список с существующими расписаниями полетов с возможностью добавления, редактирования и удаления; при выборе маршрутов появится список с существующими маршрутами с возможностью добавления, редактирования и удаления; при выборе билеты появится список с проданными

билетами с возможностью добавления, редактирования и удаления; при выборе отчет появится список с проданными билетами, возможность найти билеты по определенной дате и распечатать текущий список проданных билетов; также существует возможность изменить свои персональные и учетные данные.

Руководство пользователя для администратора: при открытии приложения появляется окно авторизации, в которое необходимо ввести свои учетные данные; после успешной авторизации появляется меню с возможностью выбора пользователей или истории авторизаций; при выборе истории авторизаций появляется список с авторизациями пользователей отсортированный по времени; при выборе пользователей появляется список с пользователями системы с возможностью добавления, редактирования и удаления информации о них; также существует возможность изменить свои персональные и учетные данные.

Заключение

В результате выполнения курсовой проектной работы было разработано настольное приложение “Airlines+” с помощью средств разработки языка программирования C# и платформы .NET Framework, а также с использованием системой управления базами данных SQL Server Management Studio.

Настольное приложение соответствует требованиям и пожеланиям заказчика и обладает следующими функциональными возможностями:

1) Ведение учета расписания полетов авиакомпании в электронном виде;
2) Ведение учета офисов авиакомпании в электронном виде;
3) Введение учета самолетов авиакомпании в электронном виде;
4) Введение учета маршрутов авиакомпании в электронном виде;
5) Введение учета аэропортов авиакомпании в электронном виде;
6) Введение учета по проданным билетам авиакомпании в электронном виде;
7) Ведение учета сотрудников и их персональных данных в электронном виде;
8) Формирование отчетов за выбранный период времени по проданным билетам.

Разработанное настольное приложения можно рассматривать как удобное средство для автоматизации рабочих мест менеджера и бухгалтера в авиакомпании небольшого размера.

Список использованных источников

1) 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.

2) METANIT.COM. Сайт о программировании. Полное руководство по языку программирования C# 9.0 и платформе .NET 5: [сайт] – Текст: электронный. – URL: https://metanit.com/sharp/tutorial (дата обращения 20.10.2020).

3) Федорова, Г.Н. Разработка, администрирование и защита баз данных: учебник для студ. учреждений среднего профессионального образования / Г.Н. Федорова. – 2-е изд., стер. – М.: Издательский центр «Академия», 2018. – 288 с. – (Профессиональное образование). – ISBN 978-5-4468-6994-7. – Текст: непосредственный

4) Онлайн-школа IT профессий и сообщество программистов. [сайт] – Текст: электронный. – URL: https://itproger.com/ (дата обращения 15.05.2021).

5) Мюллер Джон Пол, Семпф Билл, Сфер Чак. C# для чайников.: Пер. с англ. - СПб.: ООО "Диалектика", 2019. - 608 с. – (Профессиональное образование). - ISBN 978-5-907144-43-9. - Текст: непосредственный

6) 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

7) Портал для помощи программистам. Текст: электронный. – URL: https://codernet.ru/ (дата обращения 27.03.2021).

8) 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 промышленности и приборостроения СССР. – Москва.: ИПК Издательство стандартов, 2002 г. – 27 с. –Текст непосредственный.

9) IT Курсы программирования онлайн – обучение программированию, видео уроки. [сайт] - Текст: электронный. Видеоматериалы. – URL: https://itvdn.com/ru (дата обращения 21.04.2021).

10) UML-диаграммы классов. Программирование. [сайт] – Текст: электронный. – URL: https://prog-cpp.ru/uml-classes (дата обращения 17.04.2021).

Приложение А

(обязательное)

Техническое задание. Требования к программным модулям

1 Введение

Настоящее техническое задание распространяется на разработку программных модулей подсистем «Менеджер» и «Бухгалтер» информационной системы авиакомпании, предназначенных для автоматизации рабочих мест менеджера и бухгалтера. Использовать данную систему будут сотрудники авиакомпании.

Программа «Airlines+» предоставляет графический интерфейс для сбора, хранения, обработки и управления информацией о маршрутах, расписания полетов, списка самолетов, офисах, билетах.

Подобная автоматизация авиакомпании повышает эффективность управленческой деятельности и позволяет создать единую систему, работающую как слаженный механизм, своевременно анализировать процессы работы организации и повысить качество обслуживания клиентов и работы авиакомпании в целом.

2 Основания для разработки

Основанием для разработки является Договор 21 от 10.04.2021. Договор утвержден Директором ООО «Авиакомпания» Ивановым Иваном Ивановичем и Арьковым Александром Сергеевичем.

Наименование темы разработки – «Разработка программных модулей для автоматизации процесса составления расписания авиарейсов информационной системы авиакомпании».

3 Назначение разработки

Система призвана автоматизировать работу сотрудников авиакомпании:

- вести учет продажи билетов;

- автоматизировать составление маршрутов и расписания полетов;

Пользователями системы выступают менеджеры, бухгалтеры и администраторы.

4 Требования к программе или программным модулям

4.1 Требования к функциональным характеристикам

После запуска программы пользователю отображается окно авторизации с формой для ввода логина и пароля.

В системе существует всего 3 типа пользователей – менеджер, бухгалтер и администратор. В соответствии с типом пользователя программа открывает соответствующий интерфейс.

Для менеджера программа предоставляет следующие возможности:

- просмотр и изменение списка расписания полетов организации;

- просмотр и изменение списка офисов организации;

- просмотр и изменение списка самолетов организации;

- просмотр и изменение списка маршрутов организации;

- просмотр и изменение списка аэропортов организации;

- формирование отчета по проданным билетам за выбранный период.

Для бухгалтера программа предоставляет следующие возможности:

- просмотр и изменение списка расписания полетов организации;

- просмотр и изменение списка маршрутов организации;

- просмотр и изменение списка билетов организации;

- формирование отчета по проданным билетам за выбранный период.

Для администратора программа предоставляет следующие возможности:

- просмотр и изменение списка пользователей системы;

- просмотр истории авторизации.

Для всех типов пользователей программа предоставляет возможность изменения своих персональных данных.

4.2 Требования к надежности и безопасности

Программа должна соответствовать современному уровню требований к надежности программного обеспечения:

- предусматривать контроль вводимой информации и блокировку некорректных действий пользователя при работе с системой;

- обеспечивать целостность информации, хранящейся в базе данных;

- обрабатывать все возможные исключение и сообщать о них пользователю.

4.3 Требования к составу и параметрам технических средств

Требования к компьютеру сотрудника представлены в таблице 1.

Таблица А.1 – Требования к персональному компьютеру

Компонент	Требование
Жесткий диск	Минимум 4 ГБ свободного места
Монитор	Требуется монитор с разрешением 1280x720 или более высоким
Оперативная память	Не менее 4 ГБ
Быстродействие процессора	Минимум: процессор x64 с тактовой частотой 1,4 ГГц
Операционная система	Windows 10

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).

Приложение Б

(обязательное)

Программный код

Программный код представлен в контексте текстового редактора MS Visual Studio 2019 Enterprise 2019 Версия 16.9.0.

```
        <RowDefinition Height="*"/>
        <RowDefinition Height="*"/>
        <RowDefinition Height="*"/>
        <RowDefinition Height="*"/>
        <RowDefinition Height="*"/>

    </Grid.RowDefinitions>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="*"/>
        <ColumnDefinition Width="*"/>
    </Grid.ColumnDefinitions>
    <TextBlock Text="Имя клиента" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="15" ></TextBlock>
    <TextBlock Text="Фамилия клиента" Grid.Row="1" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="15" ></TextBlock>
    <TextBlock Text="Маршрут" Grid.Row="2" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="15" ></TextBlock>
    <TextBlock Text="Телефон" Grid.Row="3" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="15" ></TextBlock>
    <TextBlock Text="Номер паспорта" Grid.Row="4" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="15" ></TextBlock>
    <TextBlock Text="Класс" Grid.Row="5" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="15" ></TextBlock>
    <TextBlock Text="Номер брони" Grid.Row="6" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="15" ></TextBlock>
    <TextBlock Text="Статус" Name="txtStatus" Grid.Row="7" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="15" ></TextBlock>

    <TextBox Text="{Binding Firstname}" MaxLength="50" Name="txtFirstNameClient" PreviewTextInput="txtFirstNameClient_PreviewTextInput"  Grid.Column="1" Grid.Row="0" Width="175" HorizontalAlignment="Left"></TextBox>
    <TextBox Text="{Binding Lastname}" MaxLength="50" Name="txtLastNameClient" PreviewTextInput="txtLastNameClient_PreviewTextInput" Grid.Column="1" Grid.Row="1" Width="175" HorizontalAlignment="Left"></TextBox>
    <ComboBox VerticalContentAlignment="Center" SelectedItem="{Binding Schedules}" Grid.Row="2" Grid.Column="1" HorizontalAlignment="Left" Width="175" Height="30" Name="comboSchedule">
            <ComboBox.ItemTemplate>
                <DataTemplate>
                    <StackPanel Orientation="Horizontal">
                        <TextBlock Name="txtDepartureAirport" Text="{Binding Routes.Airports.IATACode}"></TextBlock>
                        <TextBlock Text=" -> "></TextBlock>
                        <TextBlock Name="txtArrivalAirport" Text="{Binding Routes.Airports1.IATACode}"></TextBlock>
                        <TextBlock Text=" ("></TextBlock>
                        <TextBlock Name="txtDateSchedule" Text="{Binding Date, StringFormat=dd.MM.yyyy}"></TextBlock>
                        <TextBlock Text=" "></TextBlock>
                        <TextBlock Name="txtTimeSchedule" Text="{Binding Time}"></TextBlock>
                        <TextBlock Text=")"></TextBlock>
                    </StackPanel>
                </DataTemplate>
            </ComboBox.ItemTemplate>

        </ComboBox>
    <xctk:MaskedTextBox Mask="000-000-0000" Width="175" BorderBrush="Black" BorderThickness="2" Height="30" Text="{Binding Phone}" MaxLength="50" Name="txtPhone" PreviewTextInput="txtPhone_PreviewTextInput" Grid.Col
    <TextBox Text="{Binding PassportNumber}" Name="txtPassportNumber" MaxLength="9" PreviewTextInput="txtPassportNumber_PreviewTextInput" Grid.Column="1" Grid.Row="4" Width="175" HorizontalAlignment="Left"></TextBox
    <ComboBox HorizontalAlignment="Left" Grid.Row="5" Grid.Column="1" SelectedItem="{Binding CabinTypes}" DisplayMemberPath="Name" Width="175" Height="30" Name="comboCabinTypes"></ComboBox>
    <TextBox Text="{Binding BookingReference}" MaxLength="6" Name="txtBookingReference" PreviewTextInput="txtBookingReference_PreviewTextInput" Grid.Column="1" Grid.Row="6" Width="175" HorizontalAlignment="Left"></T
    <ComboBox HorizontalAlignment="Left" Grid.Row="7" Grid.Column="1" SelectedItem="{Binding Confirmed}" Width="175" Height="30" Name="comboConfirmed"></ComboBox>
    <Button Content="Сохранить" Grid.ColumnSpan="2" Grid.Row="8" Name="btnSave" Click="btnSave_Click" ></Button>
</Grid>
</Page>
```

Рисунок Б.1 – Программный код разметки окна «Редактирование билета»

```
private void txtSearchTicket_TextChanged(object sender, TextChangedEventArgs e)
{
    List<Tickets> currentTicket = AirlineCompanyEntities.getContext().Tickets.ToList();
    currentTicket = currentTicket.Where(p => p.Firstname.ToLower().Contains(txtSearchTicket.Text.ToLower())
    || p.Lastname.ToLower().Contains(txtSearchTicket.Text.ToLower())
    || p.Users.LastName.ToLower().Contains(txtSearchTicket.Text.ToLower())
    || p.Users.FirstName.ToLower().Contains(txtSearchTicket.Text.ToLower())
    || p.Schedules.Routes.Airports.IATACode.ToLower().Contains(txtSearchTicket.Text.ToLower())
    || p.Schedules.Routes.Airports1.IATACode.ToLower().Contains(txtSearchTicket.Text.ToLower())
    || p.Schedules.Date.ToShortDateString().Contains(txtSearchTicket.Text)).ToList();
    dGridTickets.ItemsSource = currentTicket.ToList();
    if (dGridTickets.Items.Count == 0)
    {
        labelNotFound.Visibility = Visibility.Visible;
        dGridTickets.Visibility = Visibility.Collapsed;
    }
    else
    {
        dGridTickets.Visibility = Visibility.Visible;
        labelNotFound.Visibility = Visibility.Collapsed;
    }
}
```

Рисунок Б.2 – Программный код обработчика текстового поля «Поиск» на странице «Билеты»

```
private void btnApplyParameters_Click(object sender, RoutedEventArgs e)
{
    if (comboFromAirport.SelectedItem != null && comboToAirport.SelectedItem != null)
    {
        if (comboFromAirport.SelectedItem != comboToAirport.SelectedItem)
        {
            airportFrom = (Airports)comboFromAirport.SelectedItem;
            airportTo = (Airports)comboToAirport.SelectedItem;
            dGridSheduleFlight.ItemsSource = AirlineCompanyEntities.getContext().Schedules.Where(p => p.Routes.ArrivalAirportID == airportTo.ID
            && p.Routes.DepartureAirportID == airportFrom.ID).ToList();
        }
    }
    if (comboFromAirport.SelectedItem != null && comboToAirport.SelectedItem == null)
    {
        if (comboFromAirport.SelectedItem != comboToAirport.SelectedItem)
        {
            airportFrom = (Airports)comboFromAirport.SelectedItem;
            airportTo = (Airports)comboToAirport.SelectedItem;
            dGridSheduleFlight.ItemsSource = AirlineCompanyEntities.getContext().Schedules.Where
                (p => p.Routes.DepartureAirportID == airportFrom.ID).ToList();
        }
    }
    if (comboFromAirport.SelectedItem == null && comboToAirport.SelectedItem != null)
    {
        if (comboFromAirport.SelectedItem != comboToAirport.SelectedItem)
        {
            airportFrom = (Airports)comboFromAirport.SelectedItem;
            airportTo = (Airports)comboToAirport.SelectedItem;
            dGridSheduleFlight.ItemsSource = AirlineCompanyEntities.getContext().Schedules.Where
                (p => p.Routes.ArrivalAirportID == airportTo.ID).ToList();
        }
    }

}
```

Рисунок Б.3 – Программный код обработчика кнопки «Применить» для комбо боксов на странице «Расписание»

```
    <Grid.RowDefinitions>
        <RowDefinition Height="120"></RowDefinition>
        <RowDefinition></RowDefinition>
    </Grid.RowDefinitions>
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition></ColumnDefinition>
            <ColumnDefinition Width="150"></ColumnDefinition>
        </Grid.ColumnDefinitions>
        <StackPanel Orientation="Vertical" VerticalAlignment="Top" HorizontalAlignment="Left" Margin="0 20 0 0">
            <StackPanel Orientation="Horizontal">
                <TextBlock Text="Начальная дата" VerticalAlignment="Center" Margin="30 0 10 0"></TextBlock>
                <DatePicker Name="datePickerStartDate"></DatePicker>
            </StackPanel>
            <StackPanel Orientation="Horizontal" Margin="0 20 0 0">
                <TextBlock Text="Конечная дата" VerticalAlignment="Center" Margin="30 0 15 0"></TextBlock>
                <DatePicker Name="datePickerEndDate"></DatePicker>
            </StackPanel>
        </StackPanel>
        <Button Content="Подтвердить" Name="btnConfirm" Click="btnConfirm_Click" Margin="50 0 0 0"></Button>
        <Button Width="120" Name="btnPrint" Click="btnPrint_Click" Grid.Column="2" HorizontalAlignment="Right">
            <StackPanel Orientation="Horizontal">
                <TextBlock Text="Печать" Margin="0 0 20 0"></TextBlock>
                <Image Source="/Airport;component/Resources/printer.png" Height="20" Width="20"></Image>
            </StackPanel>
        </Button>
    </Grid>
    <Label Name="labelNotFound" Content="Не найдено совпадений" Grid.Row="1" HorizontalAlignment="Center" VerticalAlignment="Center"></Label>
    <FlowDocumentScrollViewer Name="flowDoc" Grid.Row="1" ScrollViewer.HorizontalScrollBarVisibility="Disabled">
        <FlowDocument Name="flowDocPrint">
            <BlockUIContainer>
                <DataGrid AutoGenerateColumns="False" Name="dGridTickets" Grid.Row="1" IsReadOnly="True" >
                    <DataGrid.ItemContainerStyle...>
                    <DataGrid.Columns>
                        <DataGridTemplateColumn Header="Клиент" Width="150">
                            <DataGridTemplateColumn.CellTemplate>
                                <DataTemplate>
                                    <StackPanel Orientation="Horizontal">
                                        <TextBlock Text="{Binding Firstname}"></TextBlock>
                                        <TextBlock Text=" "></TextBlock>
                                        <TextBlock Text="{Binding Lastname}"></TextBlock>
                                    </StackPanel>
                                </DataTemplate>
                            </DataGridTemplateColumn.CellTemplate>
                        </DataGridTemplateColumn>
                        <DataGridTemplateColumn Header="Маршрут" Width="*">
                            <DataGridTemplateColumn.CellTemplate>
                                <DataTemplate>
                                    <StackPanel Orientation="Horizontal">
                                        <TextBlock Text="{Binding Schedules.Routes.Airports.IATACode}"></TextBlock>
                                        <TextBlock Text=" -> "></TextBlock>
                                        <TextBlock Text="{Binding Schedules.Routes.Airports1.IATACode}"></TextBlock>
                                        <TextBlock Text=" ("></TextBlock>
                                        <TextBlock Name="txtDateSchedule" Text="{Binding Schedules.Date, StringFormat=dd.MM.yyyy}"></TextBlock>
                                        <TextBlock Text=" "></TextBlock>
                                        <TextBlock Name="txtTimeSchedule" Text="{Binding Schedules.Time}"></TextBlock>
                                        <TextBlock Text=")"></TextBlock>
```

Рисунок Б.4 – Программный код разметки страницы «Отчеты»

```
                                    </StackPanel>
                                </DataTemplate>
                            </DataGridTemplateColumn.CellTemplate>
                        </DataGridTemplateColumn>
                        <DataGridTextColumn Header="Класс места" Binding="{Binding CabinTypes.Name}" Width="100"></DataGridTextColumn>
                        <DataGridTemplateColumn Header="Сотрудник" Width="*">
                            <DataGridTemplateColumn.CellTemplate>
                                <DataTemplate>
                                    <StackPanel Orientation="Horizontal">
                                        <TextBlock Text="{Binding Users.FirstName}"></TextBlock>
                                        <TextBlock Text=" "></TextBlock>
                                        <TextBlock Text="{Binding Users.LastName}"></TextBlock>
                                    </StackPanel>
                                </DataTemplate>
                            </DataGridTemplateColumn.CellTemplate>
                        </DataGridTemplateColumn>
                    </DataGrid.Columns>
                </DataGrid>
            </BlockUIContainer>
        </FlowDocument>
    </FlowDocumentScrollViewer>
</Grid>
</Page>
```

Рисунок Б.5 – Программный код разметки страницы «Отчеты»

Приложение В

(обязательное)

Формы выходных документов

Клиент	Маршрут	Класс места	Сотрудник
Lyell Willsmore	AUH -> BAH (15.12.2017 17:00:00)	Economy	manager manager
Jenica Maycock	AUH -> BAH (15.12.2017 17:00:00)	Economy	Mihail Shvedov
Gunther Brookwell	AUH -> BAH (15.12.2017 17:00:00)	First Class	Evgeny Berezhnoy
Borden Edney	AUH -> BAH (15.12.2017 17:00:00)	Economy	manager manager
Tanner Dallman	AUH -> BAH (15.12.2017 17:00:00)	Economy	Evgeny Berezhnoy
Uriel Shury	BAH -> AUH (04.12.2017 21:09:00	Economy	Mihail Shvedov
Teddie O'Shee	BAH -> AUH (04.12.2017 21:09:00	Economy	Evgeny Berezhnoy
Osmund Enocksson	BAH -> AUH (04.12.2017 21:09:00	First Class	Evgeny Berezhnoy
Tony Van Arsdale	BAH -> AUH (04.12.2017 21:09:00	Business	Evgeny Berezhnoy
Lyndell Deek	AUH -> ADE (05.12.2017 22:22:22	Economy	Artem Shpak
Glennis Alpin	AUH -> ADE (05.12.2017 22:22:22	Economy	Mihail Shvedov
Ambur Scroggs	AUH -> ADE (05.12.2017 22:22:22	Economy	Evgeny Berezhnoy
Gonzales Austen	AUH -> ADE (05.12.2017 22:22:22	First Class	Evgeny Berezhnoy
Quinta Gennrich	BAH -> AUH (05.12.2017 21:09:00	First Class	Evgeny Berezhnoy
Lucita Kington	BAH -> AUH (05.12.2017 21:09:00	Economy	Mihail Shvedov
Britni Gross	BAH -> AUH (05.12.2017 21:09:00	First Class	Evgeny Berezhnoy
Guthrey Hazeman	BAH -> AUH (05.12.2017 21:09:00	Business	Evgeny Berezhnoy
Cad Bemand	BAH -> AUH (05.12.2017 21:09:00	Economy	Artem Shpak
Domeniga Slobom	BAH -> AUH (05.12.2017 21:09:00	Economy	Evgeny Berezhnoy
Sheridan Ipwell	BAH -> AUH (05.12.2017 21:09:00	Economy	Mihail Shvedov
Devonna Marl	AUH -> BAH (06.12.2017 17:00:00	Economy	Evgeny Berezhnoy
Patty Marmon	AUH -> BAH (06.12.2017 17:00:00	Economy	Evgeny Berezhnoy
Julee Dear	AUH -> BAH (06.12.2017 17:00:00	First Class	Evgeny Berezhnoy
Othella Kirkhouse	BAH -> AUH (06.12.2017 21:09:00	Economy	Mihail Shvedov
Neel Bawme	BAH -> AUH (06.12.2017 21:09:00	First Class	Evgeny Berezhnoy
Clair Mabbott	BAH -> AUH (06.12.2017 21:09:00	Business	Evgeny Berezhnoy
Jimmy Cornier	BAH -> AUH (06.12.2017 21:09:00	Economy	Artem Shpak
Con Dowty	AUH -> BAH (07.12.2017 17:00:00	Economy	Mihail Shvedov
Rhoda Felten	AUH -> BAH (07.12.2017 17:00:00	Economy	Evgeny Berezhnoy
Ardine Stear	AUH -> BAH (07.12.2017 17:00:00	Business	Evgeny Berezhnoy
Leslie Borrel	AUH -> BAH (07.12.2017 17:00:00	Business	Evgeny Berezhnoy
Caye Roja	BAH -> AUH (07.12.2017 21:09:00	Economy	Mihail Shvedov
Carlee Davis	BAH -> AUH (07.12.2017 21:09:00	Economy	Evgeny Berezhnoy

Рисунок В.1 – Отчет по выполненным заказам в формате .pdf

Приложение Г

(обязательное)

Скрипт базы данных

```
script.sql - PC\MSS...pany (PC\user (53))
USE [master]
GO
/****** Object:  Database [AirlineCompany]    Script Date: 06.06.2021
CREATE DATABASE [AirlineCompany]
 CONTAINMENT = NONE
 ON  PRIMARY
( NAME = N'AirlineCompany1', FILENAME = N'C:\Program Files\Microsoft S
 LOG ON
( NAME = N'AirlineCompany1_log', FILENAME = N'C:\Program Files\Microso
 WITH CATALOG_COLLATION = DATABASE_DEFAULT
GO
ALTER DATABASE [AirlineCompany] SET COMPATIBILITY_LEVEL = 150
GO
IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))
begin
EXEC [AirlineCompany].[dbo].[sp_fulltext_database] @action = 'enable'
end
GO
ALTER DATABASE [AirlineCompany] SET ANSI_NULL_DEFAULT OFF
GO
ALTER DATABASE [AirlineCompany] SET ANSI_NULLS OFF
GO
ALTER DATABASE [AirlineCompany] SET ANSI_PADDING OFF
GO
ALTER DATABASE [AirlineCompany] SET ANSI_WARNINGS OFF
GO
ALTER DATABASE [AirlineCompany] SET ARITHABORT OFF
GO
ALTER DATABASE [AirlineCompany] SET AUTO_CLOSE OFF
GO
ALTER DATABASE [AirlineCompany] SET AUTO_SHRINK OFF
GO
ALTER DATABASE [AirlineCompany] SET AUTO_UPDATE_STATISTICS ON
GO
ALTER DATABASE [AirlineCompany] SET CURSOR_CLOSE_ON_COMMIT OFF
GO
ALTER DATABASE [AirlineCompany] SET CURSOR_DEFAULT  GLOBAL
GO
ALTER DATABASE [AirlineCompany] SET CONCAT_NULL_YIELDS_NULL OFF
GO
ALTER DATABASE [AirlineCompany] SET NUMERIC_ROUNDABORT OFF
GO
ALTER DATABASE [AirlineCompany] SET QUOTED_IDENTIFIER OFF
GO
ALTER DATABASE [AirlineCompany] SET RECURSIVE_TRIGGERS OFF
```

Рисунок Г.1 – Скрипт БД

Приложение Д

(обязательное)

Тестирование

Общая информация о тестировании

Название проекта	Airline
Номер версии	1
Имя тестировщика	Арьков Александр
Дата тестирования	06.06.2021

Test case #1:

Test Case #	TC_1.
Приоритет теста	Средний.
Название тестирования/Имя	Проверка содержания логина.
Резюме испытания	Логин должен содержать буквы латинские буквы, цифры, и длина должна быть больше чем 6.
Шаги тестирования	1) Ввести логин
Данные тестирования	Логин: arkov632.
Ожидаемый результат	Модульный тест PositiveTest пройдет успешно.
Фактический результат	Соответствует ожидаемому.
Предпосылки	Запустить приложение.
Постусловия	Приложение должно вывести успешный результат тестирования.
Статус (Pass/Fail)	Pass.
Комментарии	-

Test case #2:

Test Case #	TC_2.
Приоритет теста	Средний.
Название тестирования/Имя	Проверка содержания логина.
Резюме испытания	Логин должен содержать буквы латинские буквы, цифры, и длина должна быть больше чем 6.
Шаги тестирования	1) Ввести логин
Данные тестирования	Логин: arkov.
Ожидаемый результат	Модульный тест PositiveTest пройдет успешно.
Фактический результат	Соответствует ожидаемому.
Предпосылки	Запустить приложение.
Постусловия	Приложение должно вывести успешный результат тестирования.
Статус (Pass/Fail)	Pass.
Комментарии	-