МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий Кафедра Информационных систем и технологий Специальность 1-40 01 01 «Информационные системы и технологии»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОЙ РАБОТЫ:**

по дисциплине «Программирование в Internet»

Тема «Веб-приложение для управления почтой»

Исполнитель

студент (ка) 4 курса 1 группы А.В. Глушко

(Ф.И.О.)

Руководитель

Ст. преподаватель М. А. Левин

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Руководитель М. А. Левин

(подпись)

Минск 2017

Оглавление

[Введение 3](#_Toc531878679)

[1 Обзор аналогов систем почты 4](#_Toc531878680)

[2 Проектирование и разработка систем 6](#_Toc531878681)

[2.1 Архитектура БД 6](#_Toc531878682)

[2.1.1 Требования к СУБД 6](#_Toc531878683)

[2.1.2 Проектирование логической структуры базы данных 8](#_Toc531878684)

[2.2 Разработка приложения 10](#_Toc531878685)

[2.2.1 Алгоритм функционирования подпрограммной подсистемы 10](#_Toc531878686)

[2.2.2 Разграничение прав доступа пользователей в системе 11](#_Toc531878687)

[2.2.3 Структура решения 12](#_Toc531878688)

[3 Логическое описание работы приложения 16](#_Toc531878689)

[3.1 Реализация класса PostOperations 16](#_Toc531878690)

[3.2 Реализация класса HashPass 17](#_Toc531878691)

[3.3 Реализация Yandex Map используя JS API 18](#_Toc531878692)

[3.4 Организация работы с Таблицами и навигация окон 19](#_Toc531878693)

[3.5 Организация валидации вводимых данных 20](#_Toc531878694)

[3.6 Реализация меню приложения 21](#_Toc531878695)

[3.7 Реализация метода составления отчетов 22](#_Toc531878696)

[3.8 Реализация оповещений на E-mail 23](#_Toc531878697)

[4 Дизайн приложения 24](#_Toc531878698)

[4.1 Общий дизайн приложения 24](#_Toc531878699)

[5 Руководство пользователя 25](#_Toc531878700)

[5.1 Окно авторизации 25](#_Toc531878701)

[5.2 Окно регистрации 26](#_Toc531878702)

[5.3 Главное окно пользователя 27](#_Toc531878703)

[5.4 Главное окно администратора 30](#_Toc531878704)

[Тестирование 34](#_Toc531878705)

[Заключение 35](#_Toc531878706)

[Список литературы 36](#_Toc531878707)

# Введение

Информационные ресурсы в современном обществе играют не меньшую, а нередко и большую роль, чем ресурсы материальные. Знания, кому, когда и где продать товар, может цениться не меньше, чем собственно товар и в этом плане динамика развития общества свидетельствует о том, что на «весах» материальных и информационных ресурсов последние начинают превалировать. Именно поэтому надо знать какой функционал нужно предоставить для успешного внедрения информационного продукта на рынок.

Рассматривая данное направление в разделе «Почта», нужно понимать, как конкретно устроена данная система, чтобы создаваемое приложение оказалось полезным.

В настоящее время регулирование вопросов международной почтовой связи осуществляет межгосударственная организация - Всемирный почтовый союз (ВПС; международное сокращённое название - UPU, от фр. Union Postale Universelle и англ. Universal Postal Union). В ВПС входят практически все страны мира.

Страны, входящие в состав ВПС, применяют единообразные правила для обмена почтовыми отправлениями между своими почтовыми учреждениями.

Общие правила международных почтовых связей закреплены во Всемирной почтовой конвенции (англ. Universal Postal Convention) - международно-правовом соглашении, заключённым между государствами-членами ВПС.

Основной принцип взаимодействия между почтовыми операторами, входящими в ВПС простой – если в стране отправки международное почтовое отправлении (далее по тексту - МПО) передано национальному почтовому оператору, то в стране получения это МПО будет так же доставляться национальным почтовым оператором.

Упрощенно, в контексте международных покупок, международные почтовые отправления можно классифицировать следующим образом:

* письменная корреспонденция (вес до 2-х кг (в ряде случаев до 5 кг), в том числе письма и мелкие пакеты);
* посылки весом до 20 кг (до 31 кг в случае отправления EMS).

В общем случае надо реализовать выполнение таких функций как:

* Сохранение рабочей информации в централизованной базе данных;
* Составление рабочих планов;
* Анализ данных, создание отчётов о проделанной работе;
* Настройка параметров работы приложения;
* Авторизация для разных групп пользователей;
* Доступ /доступ и работа с информацией /изменение информации (для разных групп пользователей);
* Проверка местонахождения посылки.

Каждая из функций может разбиваться на множество меньших функций, т.е. любая функция является составной из более мелких. В частности работа с БД включает в себя множество других функций, от простого вывода информации, до заполнения и запросов для данной БД.

# Обзор аналогов систем почты

В общем случае приложения для почты должно выполнять перечень стандартизированных функций. И рассматривая один из аналогов приложений для почты (пример аналога приложения почты предоставлен на рисунке 1.1) можно заметить такие функции, как:

* отправка писем\посылок, денежный перевод;
* отслеживание посылки;
* открытки;
* дополнительная информация.

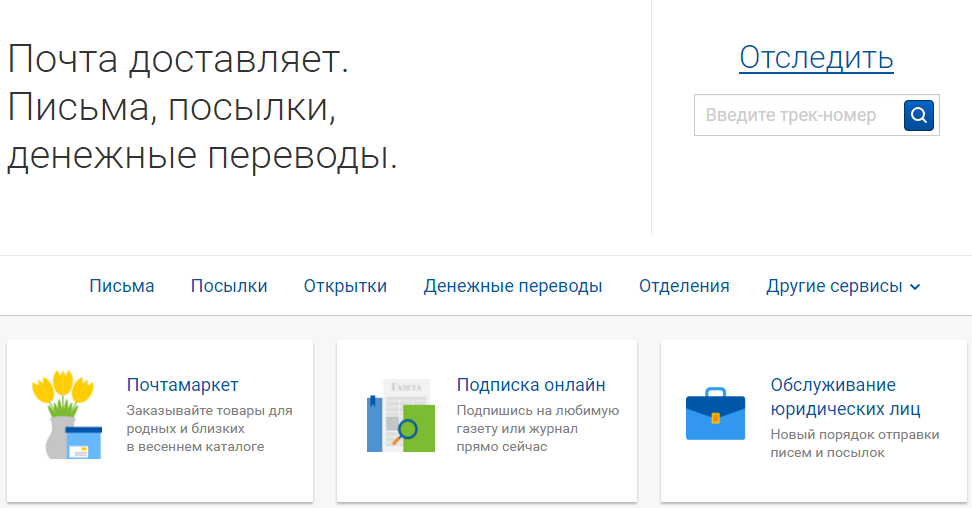


Рисунок 1.1 – Аналог приложения почты

Так как все программные средства содержат одинаковый перечень функций, то свой продукт можно сделать, опираясь на стандартный функционал для отделений почты.

Также в качестве аналога, было рассмотрено мобильное приложение почты России, представленное на рисунке 1.2.

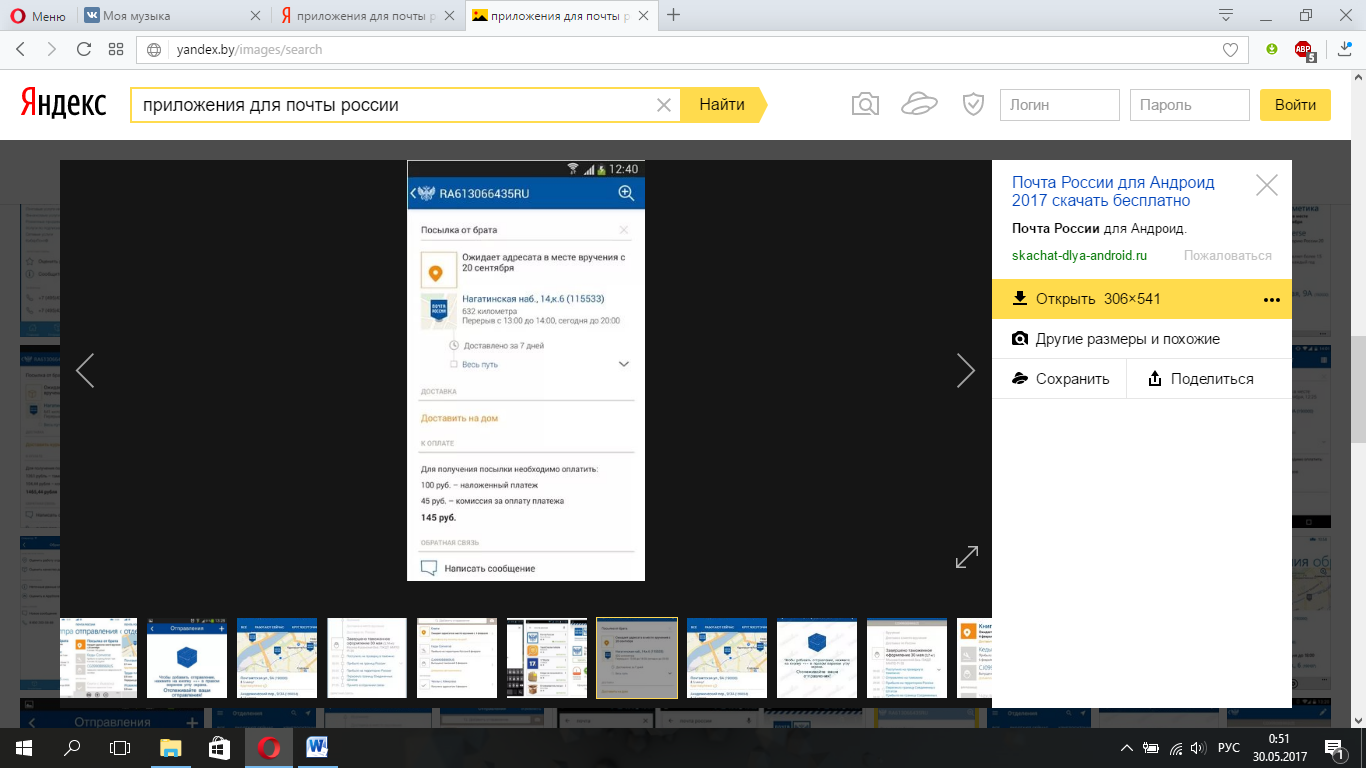


Рисунок 1.2 – Мобильное приложение почты России

Данное приложение сосредоточено только под пользователя, и предназначено только для просмотра местонахождений посылки, и просмотра отделений почтовой связи России.

Таким образом, можно сделать вывод о том, что итоговое программное средство может содержать выполнение данных функций. В набор функционала своего приложения, которое опирается не только на обычного повседневного пользователя, но и на служащего почты (администратора), был выбран следующий перечень функций:

Для пользователя:

* просмотр активных посылок;
* история посылок;
* просмотр активных и прошедших подписок на издания;
* оформление новой подписки на издание;
* поиска адреса по индексу (отображение на Google Maps);
* поиск индекса по адресу (отображение на Google Maps).

Для администратора:

* оформление посылки между клиентами;
* поиск индекса или адреса (отображение на Google Maps);
* добавление и удаление издания;
* добавление удаление типа посылок;
* просмотр посылок: отправленные, полученные, и их местонахождение;
* отметка местонахождение посылки;
* оформление посылки от пользователя, пользователю приложения;
* просмотр информации о пользователе;
* оформление посылки от пользователя клиенту;
* оформление пользователю подписки на издание;
* просмотр отправленных посылок;
* просмотр активных изданий;
* оформление отчетов.

# Проектирование и разработка систем

## Архитектура БД

### Требования к СУБД

Требования к СУБД представлены:

* База данных должна обеспечивают возможность получения:

1. информации об оказанных услугах почтовой связи;
2. информации о пользователях (отправителях и адресатах) услугами почтовой связи;
3. информации о пользователях приложения;
4. информации о почтовых отправлениях.

* Создаваемая база данных о пользователях услугами почтовой связи и базы данных об оказанных услугах почтовой связи должны соответствовать оборудованию почтового отделения, в том числе и программному обеспечению, обеспечивающему выполнение установленных действий.
* Из базы данных о пользователях услугами почтовой связи обеспечивается возможность получения информации о пользователе услугами почтовой связи и о почтовом адресе пользователя услугами почтовой связи по персональным данным.
* Из базы данных об оказанных пользователям услугах почтовой связи обеспечивается возможность получения информации об оказанных услугах почтовой связи по персональным данным.
* Контроль почтовых отправлений осуществляется в процессе их обработки в сетях оператора почтовой связи. При передаче почтовых отправлений в упаковочной таре (мешках, ящиках и другой упаковочной таре), должна учитываться плата за вес и упаковку отправления.
* Исключение избыточности в данных.

Информационно-логическая модель отображает данные предметной области в виде совокупности информационных объектов и связей между ними. Эта модель представляет данные, подлежащие хранению в базе данных. При разработке модели данных могут использоваться два подхода. В первом подходе сначала определяются основные задачи, для решения которых строится база, и выявляются потребности задач в данных. При втором подходе сразу устанавливаются типовые объекты предметной области. Наиболее рационально сочетание обоих подходов. Это связано с тем, что на начальном этапе, как правило, нет исчерпывающих сведений обо всех задачах.

База данных «Почта» содержит следующие сущности:

Сущность «Посылка» - содержит информацию о возможных посылках: код операции и вид: вид посылки, описание, стоимость;

Сущность «Полученные посылки» - содержит информацию о полученной почте: код операции, вид получения, данные отправителя и получателя, вес и ценность;

Сущность «Отправленные посылки» - содержит информацию об отправленной почте: код операции, вид отправки, данные отправителя и получателя, вес и ценность;

Сущность «Подписка» - содержит информацию о произведенных подписках на газету или журнал: код операции, код подписчика, шифр издания, срок и стоимость подписки.

Сущность «Издание»- содержит информации о газетах и журналах доступных для подписки: шифр издания, название газеты или журнала, цена;

Сущность «Клиенты» - содержит информацию о клиентах, обратившихся в отделение почтовой связи, она содержит: код клиента, ФИО клиента, адрес клиента;

Сущность «Текущий статус» - содержит информацию о текущих отправленных посылках: код, ФИО отправителя, адрес получателя посылки;

Сущность «Индексы» - содержит информацию о индексах: индекс, адрес, регион распространения;

Сущность «Почтовые отделения» - содержит информацию о почтовых отделениях: Название офиса, код, адрес\индекс офиса;

Сущность «Пользователи» - содержит информацию о пользователях приложения, так как в приложении будет организовываться вход для нескольких групп пользователей (любой сможет скачать клиентскую пользовательскую часть приложения и пользоваться ею): код пользователя, пользователь, пароль, права доступа.

### Проектирование логической структуры базы данных

Для реализации поставленной задачи была создана база данных db.POST. Для её создания использовалось система управления реляционными базами данных Microsoft SQL Server.

База данных нашей программной подсистемы состоит из 10 таблиц, и ее схема изображена на рисунке 2.1.

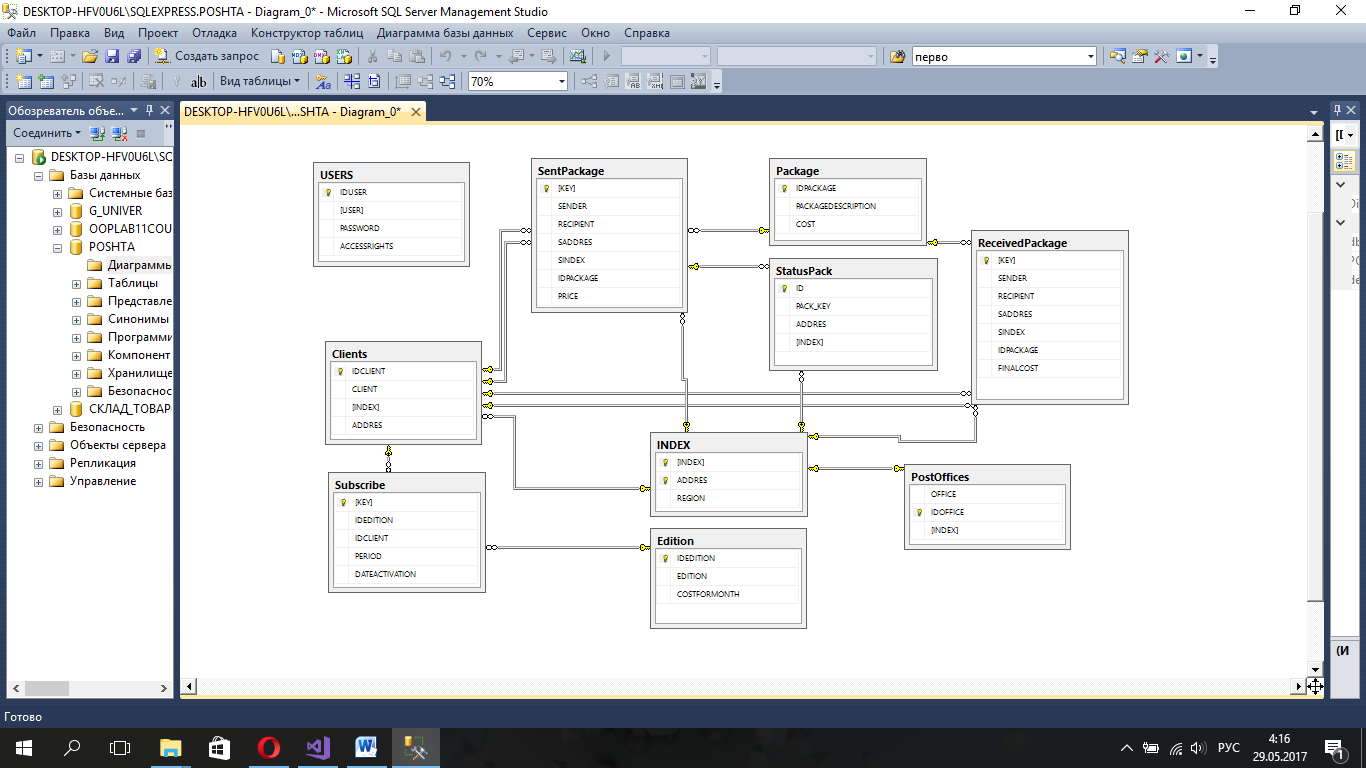


Рисунок 2.1 – схема базы данных db.POST

Для хранения зарегистрированных пользователей предназначена таблица USERS с полями:

* IDUSER – логин пользователя (первичный ключ);
* User – ФИО пользователя;
* Password – содержит хешированный пароль пользователя;
* AccessRights – права доступа.

Для хранения изданий, предоставляемых почтой, используется таблица Edition, которая содержит поля:

* IDEDTION – уникальный ключ издания (первичный ключ);
* Edition – описание издания;
* CostForMonth – содержит стоимость издания за месяц.

Для хранения типов посылок, предоставляемых почтой, используется таблица Package, содержащая поля:

* IDPACKAGE – уникальный ключ типа посылок (первичный ключ);
* PackageDescription – описание посылки;
* Cost – стоимость посылки.

Для хранения индексов предназначена таблица Index, содержащая поля:

* Index – индекс;
* Address – адрес;
* Region – область распространения (улица, город, страна и т.д.).

Для хранения подписок клиентов на издания используется таблица Subscribe, которая содержит следующие поля:

* Key – уникальный ключ подписки ( первичный ключ);
* IDEDITION – ключ издания (вторичный ключ на столбец IDEDTION таблицы Edition);
* IDCLIENT – ключ клиента (вторичный ключ на столбец IDCLIENT таблицы Clients);
* Period – срок оформления подписки;
* DateActivation – дата активации подписки.

Для хранения информации о клиентах используется таблица Clients с полями:

* IDCLIENT – уникальный ключ пользователя (первичный ключ);
* Client – ФИО пользователя;
* Index – индекс (вторичный ключ на столбец Index таблицы Index);
* Address - адрес (вторичный ключ на столбец Address таблицы Index).

Для хранения информации о посылках, находящихся в доставке, и доставленных посылках используются таблицы SentPackage и ReceivedPackage, которые содержат поля:

* Key – уникальный ключ состояния посылки ( первичный ключ);
* Sender – отправитель (вторичный ключ на столбец IDCLIENT таблицы Clients);
* Recipient – получатель (вторичный ключ на столбец IDCLIENT таблицы Clients);
* SAddress – адрес доставки (вторичный ключ на столбец Address таблицы Index);
* SIndex – индекс доставки (вторичный ключ на столбец Index таблицы Index);
* IDPACKAGE – ключ посылки (вторичный ключ на столбец IDPACKAGE таблицы Package);
* Price\FinalCost – стоимость\финальная стоимость посылки.

Для отслеживания текущего местонахождения посылок используется таблица StatusPack, включающая поля:

* ID – ключ состояния (первичный ключ);
* PACK\_KEY – ключ отправленной посылки (вторичный ключ на столбец Key таблицы SentPackage);
* Address – текущий адрес (вторичный ключ на столбец Address таблицы Index);
* Index – текущий индекс (вторичный ключ на столбец Index таблицы Index);

## Разработка приложения

### Алгоритм функционирования подпрограммной подсистемы

Если рассмотреть работу приложения с точки зрения временных операций, то мы получим диаграмму последовательности, представленную на рисунке 2.2.

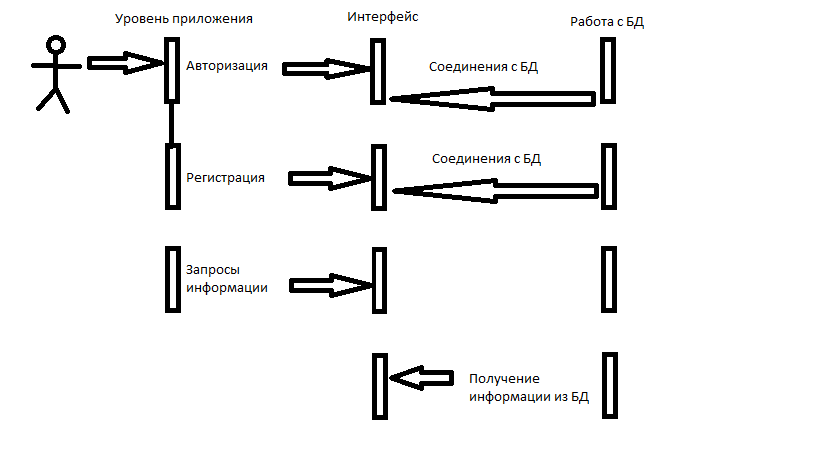


Рисунок 2.2 – диаграмма последовательности операций

На уровне подсистемы кроме соединения с базой данных выполняется другой функционал приложения.

В данном приложении поддерживается перевод на различные языки, в частности: русский, белорусский, английский.

Поддержка данной функции обеспечивается подключением на уровне подсистемы словаря ресурсов, содержащего один из языков, тем самым, обеспечивая легкость перевода данного приложения на другие языки, путем простого добавления нового словаря ресурсов, и добавления пары строчек кода в программе.

Также, во время соединения базы данных с приложением, происходит и соединение с интерфейсом Yandex Map, который осуществляет подключение приложения к системе карт.

Такой принцип позволяет при загрузке приложения (авторизации) осуществить выгрузку в оперативную память данных, необходимых для работы приложения, что в дальнейшем избавляет пользователя от ожидания загрузки приложения.

### Разграничение прав доступа пользователей в системе

В программной системе существует несколько разновидностей ролей пользователей:

* администратор;
* пользователь.

Администратору предоставляется полный доступ к информационному ресурсу. В его перечень функций входит как просмотр любой информации из базы данных, так и её изменение.

Администратору запрещено изменять таблицы из базы данных такие как: Index, Clients, ReceivedPackage, Users. Это обусловлено тем, что ему не нужен доступ для изменения данных таблиц в силу возложенных на него функций.

Пользователю предоставляется доступ к информационным ресурсам базы данных, которые актуальны только для текущего авторизованного пользователя.

В частности пользователь получает доступ к такой информации как: адреса и индексы, информация о посылках и подписках, относящаяся к нему.

### Структура решения

Структура решения изображена на рисунке 2.3.

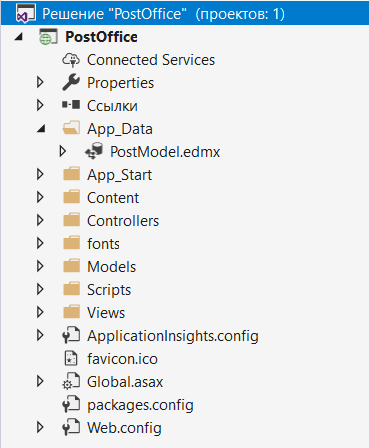


Рисунок 2.3 – структура проекта

Решение содержит один проект, который, в свою очередь, содержит остальные файлы.

Проект содержит папки для хранения разрабатываемых программных объектов, которые реализуют поставленную задачу, файл конфигураций, настройки приложения, подключенные ссылки, настройки подсистемы приложения.

Ссылками, подключенными в проект, являются стандартные ссылки языка C# и среды разработки ASP.NET MVC, за исключением ссылок, подключенных в результате добавления в проект специализированных пакетов, таких как: Entity Framework.

Файлом описывающим подключение данных пакетов является файл “packages.config”.

Папка Models содержит файлы логики работы приложения. Данная папка представлена на рисунке 2.4.

Класс “PostOperations” необходим для получения данных из базы данных.

В файле “HashPass.cs” содержится класс, с необходимыми методами для хеширования и проверки хешей паролей вводимых пользователями.

Также в папке “Models” располагаются все модели данных необходимые для функицонирования приложения. Данные модели используются для обмена данными между стороной пользователя и сервера.

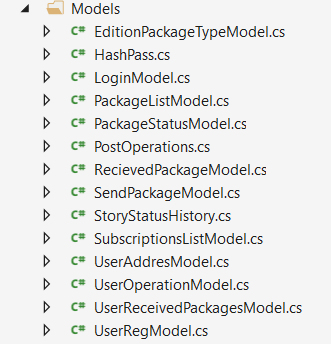


Рисунок 2.4 – Папка Models

В папке “Content”, представленной на рисунке 2.5, содержатся все необходимые стили для оформления дизайна в приложениями. Стили разделены на – общий стиль для каждой из страниц “CommonStyle.css”, стили для описания общих элементов страниц, и стили для описания дизайна каждой из страниц в отдельности.

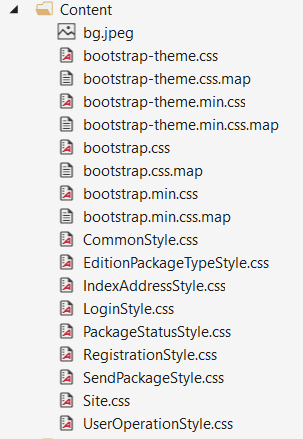


Рисунок 2.5 – Папка Content

Папка “Views”, представленная на рисунке 2.6, содержит все окна приложения.

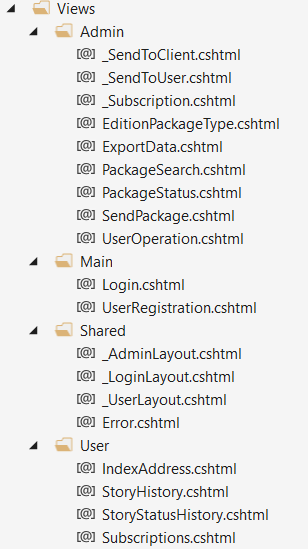


Рисунок 2.6 – Папка “Views”

Для управления логикой взаимодействия модели и возвращаемых на сторону клиента страниц в папке “Controllers” располагаются контроллеры управления данными взаимодействиями.

Для подключения базы данных использовался Entity Framework. В ходе подключения автоматически генерируется строка подключения, и конфигурации необходимые для подключения базы данных при помощи Entity Framework. Данная информации записывается в файл конфигураций “Web.config”, представленный на рисунке 2.7.



Рисунок 2.7 – описание конфигураций в файле Web.config

После подключения базы данных к приложению генерируются файлы контекста и описания базы данных, диаграмма, а также множество классов описывающие таблицы, подключенные из базы данных. Сгенерированная модель базы данных представлена на рисунке 2.8.

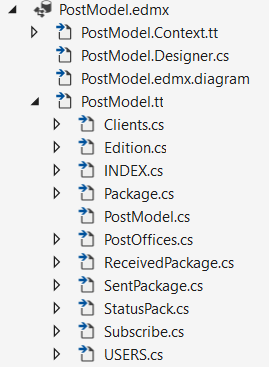


Рисунок 2.8 – Сгенерированная модель базы данных

Сгенерированные классы содержат автоматические свойства заполнения полей таблиц, хешированные связи с другими таблицами, и автоматические конструкторы, в которых и происходит связывание и согласовывание данных при помощи хешированных связей, что позволяет работать только с типами, определенными в базе данных, и наложенными на них ограничениями.

# Логическое описание работы приложения

## Реализация класса PostOperations

Для получения в приложении информации из базы данных необходимо создавать контекст подключения, а в дальнейшем получать необходимую информацию из базы данных. Для получения актуальных данных необходимо постоянное обновление, или пересоздание данного контекста.

Для данных операций в приложении используется статический класс PostOperations, представленный на рисунке 3.1.

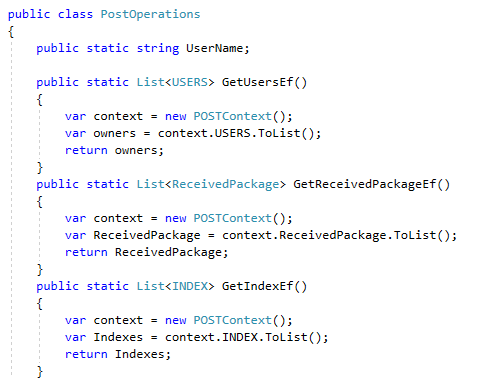


Рисунок 3.1 – Класс PostOperations

Данный класс содержит методы получения информации из базы данных, а также имя пользователя, который на данный момент авторизирован в приложении.

Как класс так и методы и поля публичные и статические.

Эта методика необходима для получения данных из любого окна приложения, так как этот класс является общим для всех окон, и доступ к его методам необходим из любой точки приложения.

Необходимость сделать его статическим заключается в ненадобности создания объекта данного класса.

Для последующего удобства использования информации, полученной из базы данных, она сохраняется в список, состоящий из объектов одной из таблиц в базе, что дает возможность воспользоваться удобствами методов списка, для выполнения операций в приложении.

Данный класс необходим как оболочка для методов, наполняющих его.

## Реализация класса HashPass

Так как в приложении реализуется как авторизация, так и регистрация пользователей, то требуется обеспечение конфиденциальности. Для этих целей используется класс, который позволяет генерировать хеш на основе входной строки(пароля). Данным классом является класс MD5. Для работы с классом MD5 необходим интерфейс, который и был разработан при создании HashPass, представленный на рисунке 3.2.

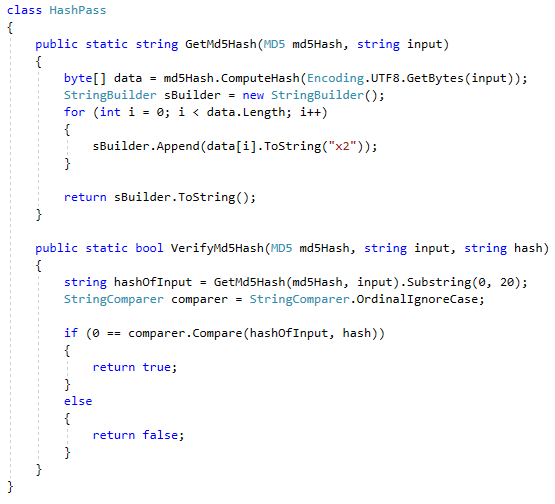


Рисунок 3.2 – Класс HashPass

Данный класс содержит два статических метода, которые осуществляют создание хеша на основе входной строки(пароля), и проверки хеша, который был ранее сгенерирован также но основе входной строки(пароля).

Использование этих методов (GetMd5Hash, VerifyMd5Hash) заключается в помещении результата хеширования(хеша) в базу данных. Либо проверки хеша из базы данных, на соответствие хешу, сгенерированному на основе пароля, вводимого пользователем.

## Реализация Yandex Map используя JS API

Пользователям необходима наглядность. Для такой цели в приложении используется Framework для работы с Yandex Map. Данная технология предоставляет в использование некоторые элементы управления, а так же методы для работы с Yandex Map.

В приложении используется элемент управления div с id равным “map”, представленный на рисунке 3.3, который при запуске приложения, загружает с сервера Yandex в одном из режимов, карту, при помощи JS API. Прогрузка карты происходит сегментно, т.е. если необходимо будет отобразить только один город, то при приближении зума к нему остальная карта не будет прогружаться.



Рисунок 3.3 – Элемент управления div для реализации Yandex map

JS API для Yandex map позволяет покрыть все потребности, необходимые почте, в частности в приложение были добавлены такие функции как: показ выбранного индекса при помощи маркера и показ маршрута посылки по индексу либо адресу.

При загрузке окна в котором необходима карта в приложении устанавливается исходное местоположение в Беларуси, а в частности, в Минске.

Регулировку области отрисовки карты можно производить несколькими способами. Одним из них является уменьшение или увеличение значения свойства Zoom, а также можно осуществлять поиск по ключевым словам, в ходе которого FrameWork сам подскажет где находится та, или иная точка на карте.

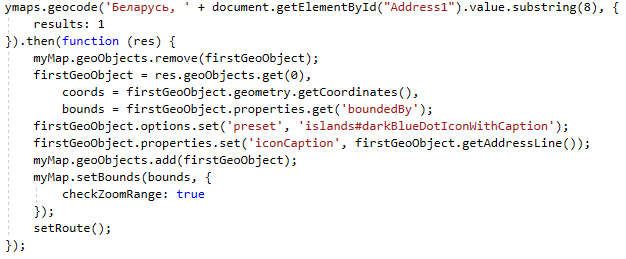


Рисунок 3.4 – Работа с Yandex Map при помощи JS API

На рисунке 3.4 представлен пример работы с картой. На рисунке выше с помощьй JS API для Yandex Map при выборе адреса, который необходимо отобразить, на карте ставится маркер для отображения выбранного адреса. Особенностью использования данного фреймворака в приложении является сопоставление адреса и индекса, т.е. если пользователь знает индекс местоположения посылки, то он может посмотреть где оно находится только при помощи него.

## Организация работы с Таблицами и навигация окон

Навигация между окнами в приложении описывается с помощью класса Html, и метода ActionLink, который генерирует в нужном месте ссылку на другую страницу либо событие. При необходимости открытия другого окна, текущее закрывается.

Для отображения информации из базы данных используется элемент управления WebGrid.

Отображения информации осуществляется за счет связывания определенных полей классов, используемых для отображения информации из базы данных с колонками WebGrid.

В html разметке, представленной на рисунке 3.5, для WebGrid, видно, что при генерации html страницы, создается переменная ReceivedPackageGrid, содержащая привязку к, переданному на страницу, набору данных. Для, уже в самой разметке страницы, создается html разметка таблицы, которая в свою очередь и содержит переданный на страницу набор данных.

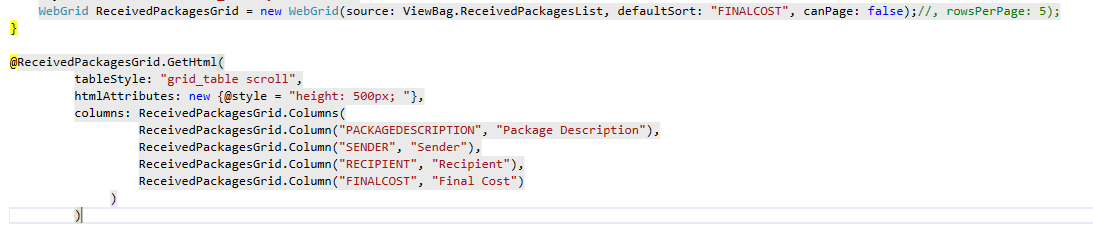


Рисунок 3.5 – Разметка элемента управления WebGrid

При работе приложения, и запросе данных от пользователя выполняется LINQ-запрос к базе данных, и с помощью привязок, данные уже отображаются в элементе управления WebGrid.

## Организация валидации вводимых данных

Валидация вводимых данных осуществляется на уровне элементов управления, так как в данном приложении пользователю почти не нужно вводить данных, а администратор кроме самостоятельного обеспечения правильности вводимой им информации, имеет наборы данных, из которых осуществляется выбор необходимого элемента.

При ошибке ввода предусмотрена валидация вводимых данных, которая осуществляется при помощи аннотаций в классе модели. Реализация валидации при вводе представлена на рисунке 3.6.

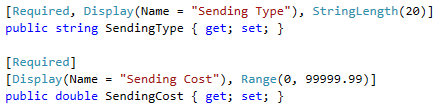


Рисунок 3.6 – Организация валидации

При вводе неправильного значения, обеспечивается подкраска поля красным в текущем элементе управления и отказ в проведении операции, и после необходимо будет или повторить ввод, или выбрать элемент из представленного списка существующие данные.

## Реализация меню приложения

Пользовательское меню реализовано при помощи общих для нескольких страниц Layout. Для страниц логина, пользователя, и администратора созданы разные Layout, для предоставления меню необходимого для конкретной роли, либо кокретной стадии выполнения приложения.

К примеру, на рисунке 3.7 представлена реализация меню на странице “User Operation”. На данной странице, меню, которое располагаяется сверху, используется для всех страниц роли администратора, а боковое меню используется только для страницы User operations, и оно, в свою очередь, предоставляет частные функции. То есть боковое меню – это под-меню приложения.

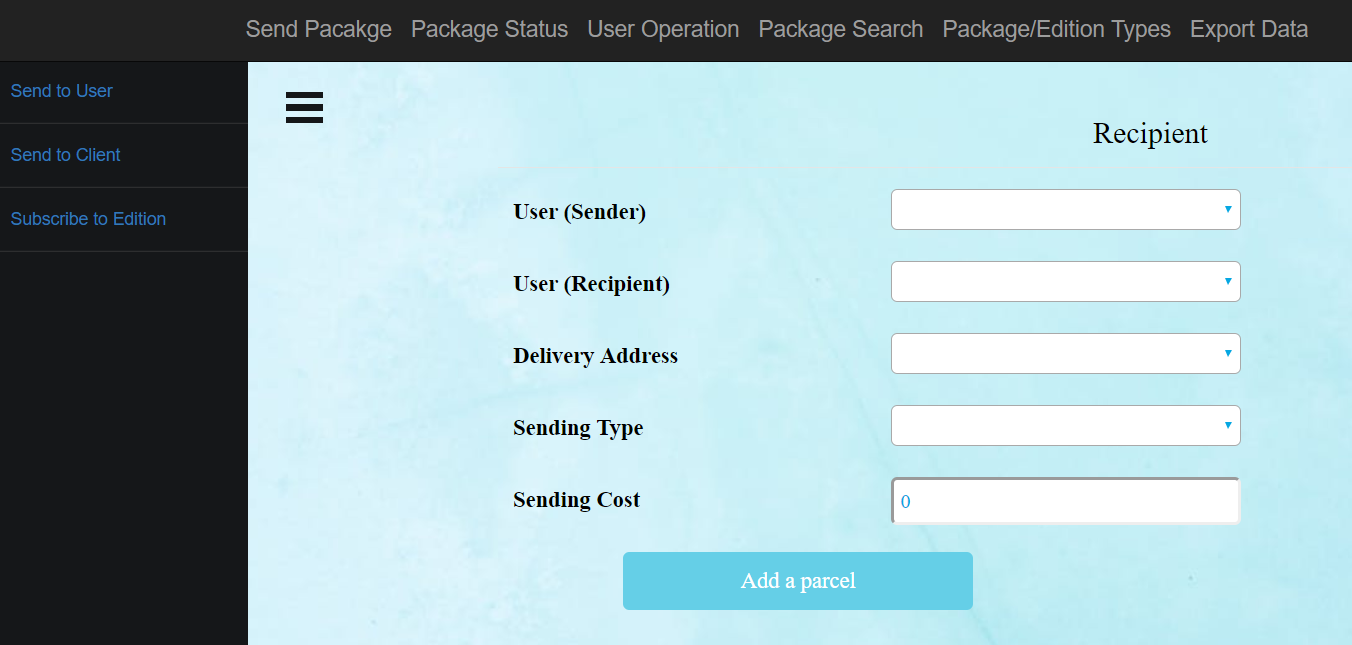


Рисунок 3.7 – Меню приложения

Также следует отметить, что страницы, которые отображаются при выборе пункта меню слева, реализованы с помощью PartialView, что в свою очередь придало гибкость разработке приложения, и гибкость в управлении конечному пользователю.

.

## Реализация метода составления отчетов

Немаловажным для приложения почты является составление отчетов. Реализуется это выводом необходимой форматированной информации в текстовый файл. Форматирование заключается в записи форматированных данных в таблицу в Excel. Реализация метода записи информации в файл представлена на рисунке 3.8.



Рисунок 3.8 – Методы сохранения информации в Excel

Суть метода заключается в заполнении необходимой информации в класс ExcelWorksheet, и далее, после заполнения информации, при помощи метода GetAsByteArray, Excel файл в байтовом формате записывается в ответ приложения, и отправляется на сторону клиента, где в свою очередь, загружается на ПК пользователя.

## Реализация оповещений на E-mail

Так как приложение включает роль обычного пользователя, то для взаимодействия с ним было реализовано оповещение через E-mail.

То есть при обновлении статуса посылки (когда посылка приехала на другой адрес), пользователю будет отправляться оповещение о текущем ее местоположении.

Пример отправленного сообщена представлен на рисунке 3.9.

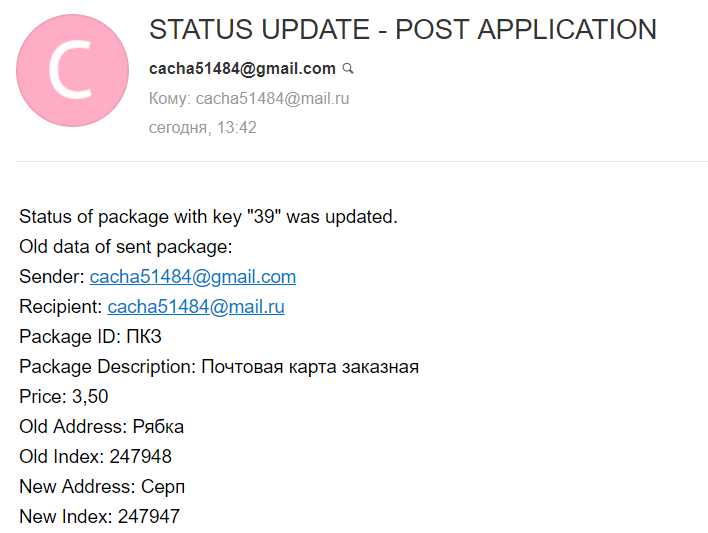


Рисунок 3.9 – Сообщение оповещения пользователя об изменении статуса посылки

Сообщение, представленное на рисунке 3.9 автоматически генерируется на стороне сервера с актуальной для обновленной посылки информацией. Также следует отметить, что сообщение отправляется через SMTP сервер с использованием SSL сертификата gmail.

# Дизайн приложения

## Общий дизайн приложения

Определение цветовой гаммы.

Исходя из цветов, которые используется во всемирных организациях почты, была определена цветовая схема. В качестве основных цветов выбран оттенок голубого цвета (AliaceBlue) – цвет элементов управления, (White) –основной текст, оттенок синего (CadetBlue) – граница элементов управления.

При создании приложения использовалась градиентная заливка от белого цвета к основному – AliaceBlue.

Голубой фон был выбран для переноса и уточнения внимания пользователя на нужной информации, которая располагается внутри белых блоков ввода и отображения информации.

Также была определена палитра используемых цветов и монохроматическая цветовая схема, представленная на рисунке 4.1.

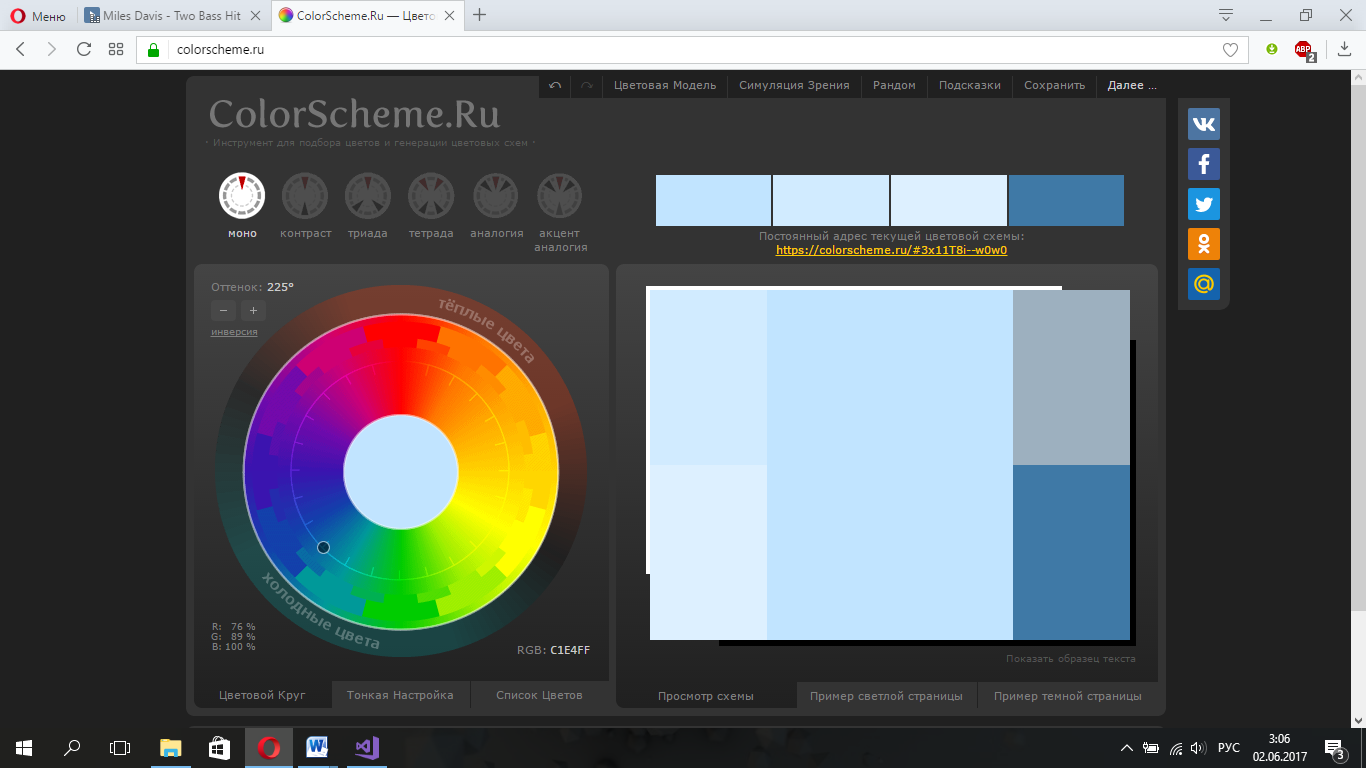


Рисунок 4.1 - Монохроматическая цветовая схема приложения

Выбор шрифта.

Для основного текста используется шрифт Times New Roman. Он прост, легко читабелен, с отсутствием засечек, и заметен.

Определение общего положения сетки и расстояний

Ширина холста равна в основном 1024 пикселей. Но также на предусмотрено и использование с более низкой или высокой шириной. Для этого элементы страницы занимают общее положение. Так как веб-страница не фиксирована по длине, то для выбора сетки для приложения можно ориентироваться приблизительно на ширину в ~980 пикселей.

Высота строки для всего приложения была выбрана, и установлена в:

Кегль – 12 пикселей, высота строки – 14 пикселей. На этой сетке лежит весь основной текст: абзацы, списки, заголовки и т. п.

Отступы от края окна были выбраны в 10 пикселей, а отступы между элементами управления выбраны в 5 пикселей. На данных отступах лежит основная разметка приложения.

Величина, равная высоте строки, есть микромодуль. В ходе использования микромодуля можно легко вычислить расстояния красной строки, между колонками и отделение абзацев друг от друга.

Размер модуля, при наличии ширины в ~980 пикселей, и зная их количество располагающихся по ширине, определяется их размер. Но так как сетка на три колонки не удобна и некомпактна, то увеличив число модулей от пяти до десяти. Рассчитав ширину одной колонки можно объединить модули в регионы и определить процентное соотношение каждого региона.

# Руководство пользователя

## Окно авторизации

После того как пользователь попадает на главную страницу приложения, до того как он пройдёт процедуру входа в систему, для него доступна только функция авторизации, либо регистрации. По умолчанию пользователя в приложении встречает авторизация. Если пользователь не зарегистрирован в системе, он может пройти процедуру регистрации, перейдя по ссылке Регистрация, которая находится в левом верхнем углу страницы. Окно авторизации представлено на рисунке 5.1

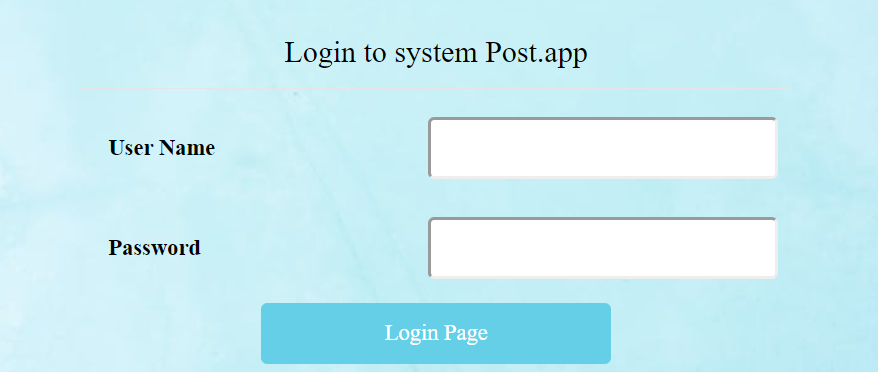


Рисунок 5.1 – Окно авторизации пользователя

Данное окно позволяет пользователю выполнить вход в систему путем ввода логина и пароля, полученного при регистрации.

## Окно регистрации

Если пользователь уже является зарегистрированным, ему необходимо пройти процедуру аутентификации. В противном случае ему потребуется заполнить форму регистрации, представленную на рисунке 5.2, для использования данного информационного ресурса.



Рисунок 5.2 – Окно регистрации пользователя

На рисунке видно, что в приложении поддерживается динамический интерфейс, в данном случае, используя одну форму для регистрации пользователя, можно зарегистрировать как обычного пользователя, так и администратора, введя специальный код доступа для получения специальных прав.

Специальный код доступа доступен для ввода при выборе пункта «Да», ответа на вопрос «Вы администратор?».

## Главное окно пользователя

После регистрации учетной записи пользователя, и пройдя процедуру авторизации, открывается главное окно пользователя, представленное на рисунке 5.3. Здесь пользователю предоставлен перечень функций, доступных для использования.

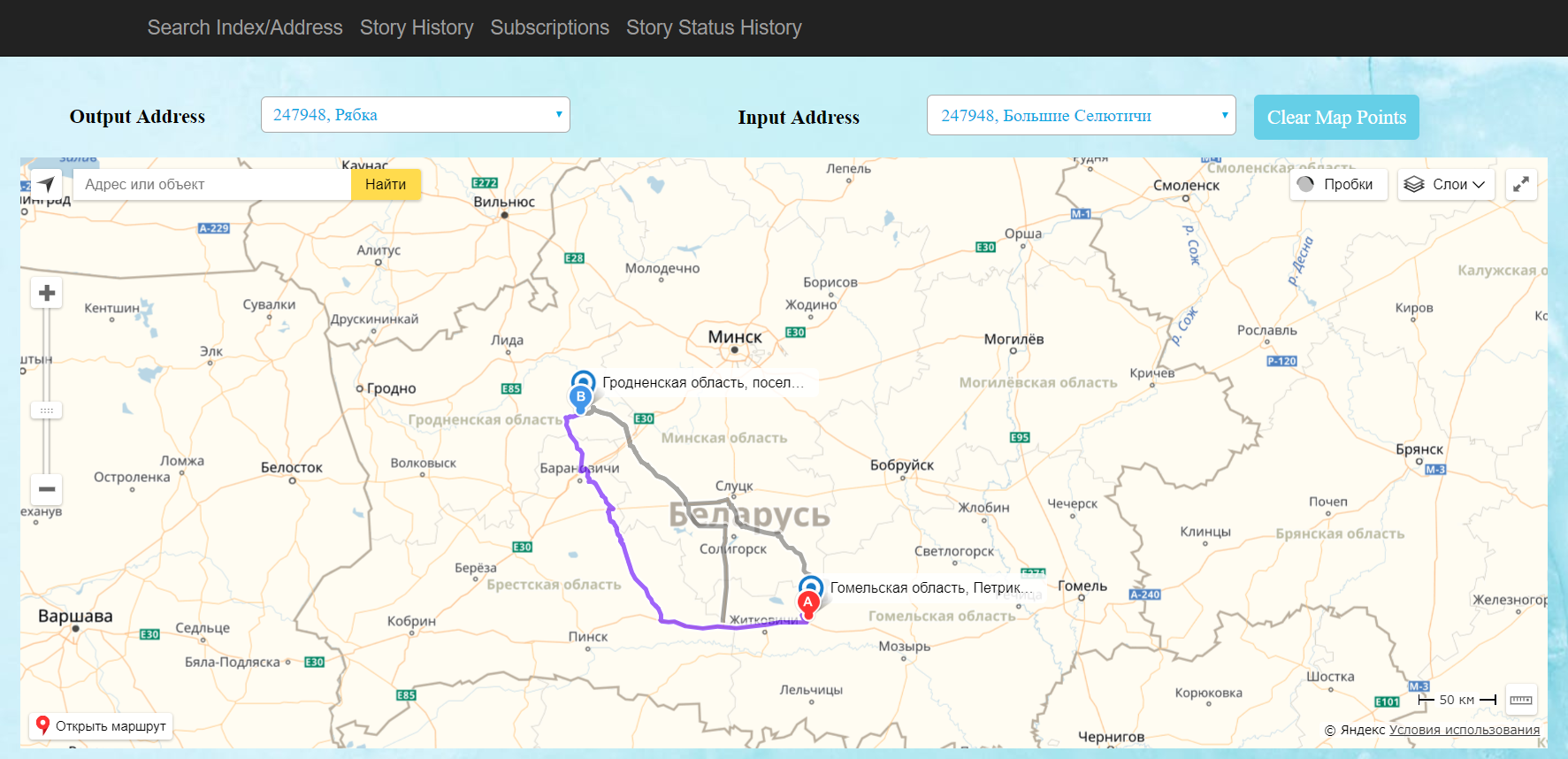


Рисунок 5.3 – Главное окно пользователя

В данном окне пользователь сразу может найти необходимую ему информацию об индексах. Пользователю необходимо ввести хотя бы частичный адрес, и ему будут представлены индексы адресов подходящих под его введенный адрес.

Одновременно с поиском при выборе адреса, он отображается на карте Google, что дает еще и визуальное представление о местонахождении данного адреса. При необходимости приближения или отдаления карты пользователь может сделать это, пользуясь колесом мыши.

Также, при выборе адреса, сразу осуществляется установка маркера в нужную точку на карте, при этом, также происходит перемещение взгляда на карту, в нужное местоположение, и с нужным зумом. Все данные функции реализованы с помощью JS API, что означает, что после загрузки страницы, подключение к серверу не требуется.

При выборе пункта меню «История посылок», окно меняет свой вид, на вид, представленный на рисунке 5.4.

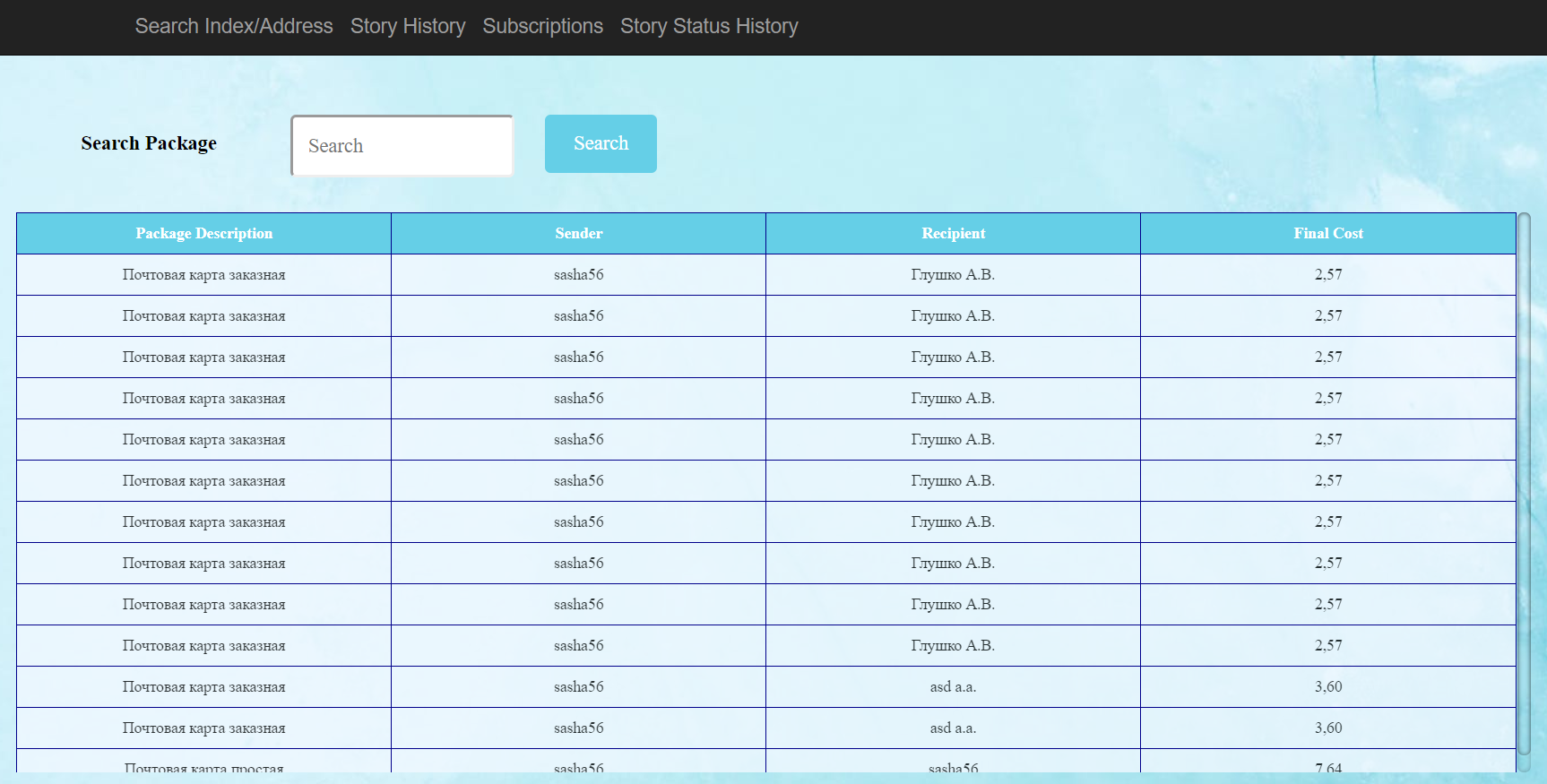


Рисунок 5.4 – История посылок

В данном окне пользователь может просмотреть перечень успешно завершенных посылок, где он является отправителем или получателем, и произвести поиск необходимой посылки. Также на данном виде окна можно наблюдать дополнительную информацию в виде типа посылки, и ее итоговой стоимость.

При выборе пункта меню «Подписки», окно меняет свой вид, на вид, представленный на рисунке 5.5.

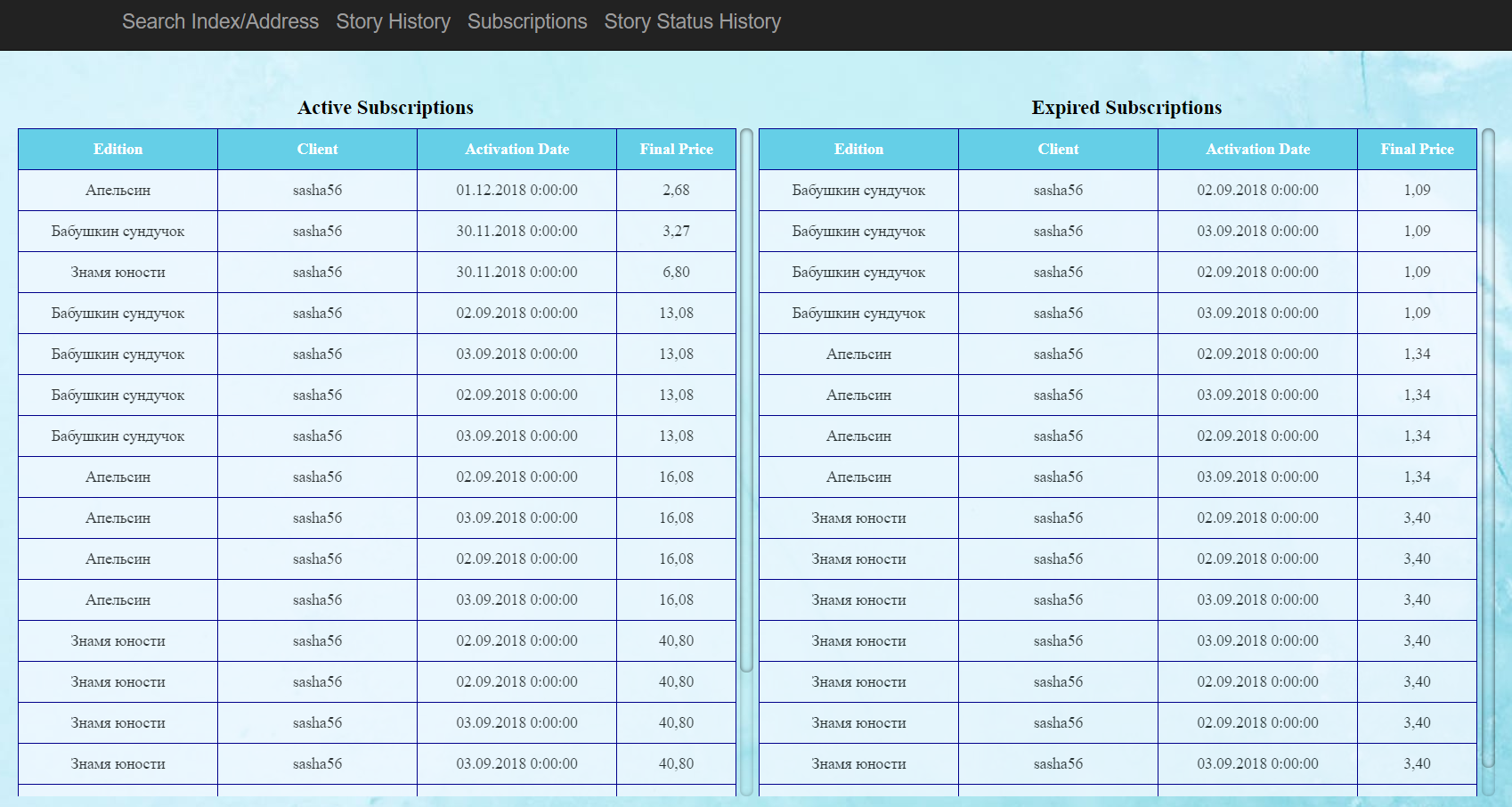


Рисунок 5.5 – Подписки

При данном виде окна пользователю доступна информация об активных и истёкших подписках, относящихся к нему. По истечению срока активации запись из активных подписок будет автоматически перенесена в таблицу истекших подписок.

При выборе пункта меню «Местонахождение посылок», окно меняет свой вид, на вид, представленный на рисунке 5.6.

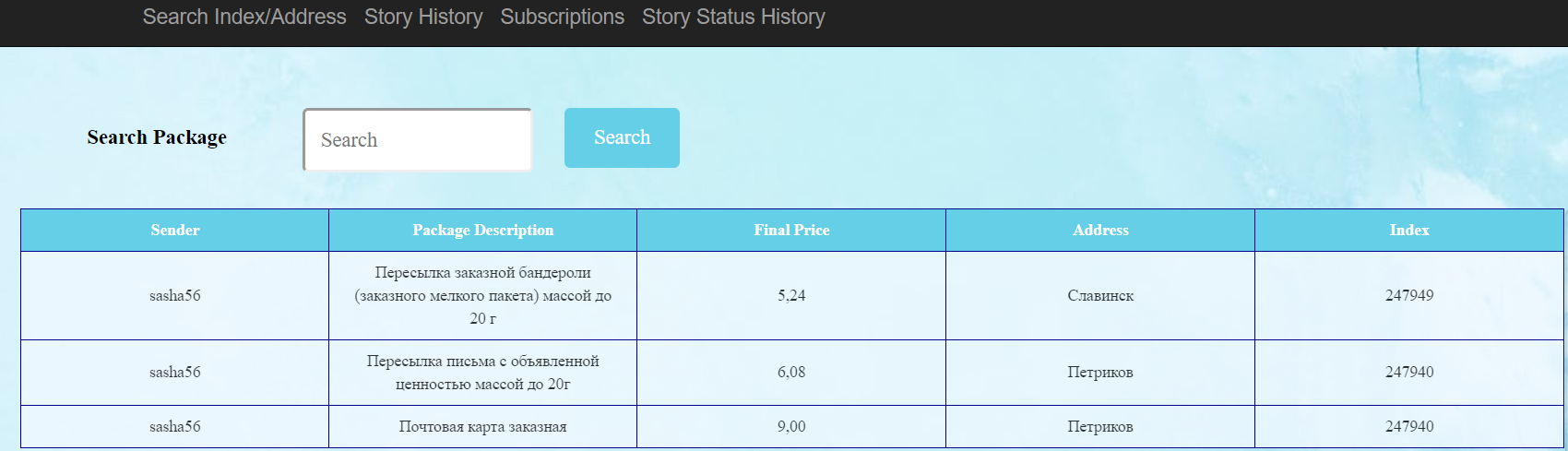


Рисунок 5.6 – Местонахождение посылок

При данном виде окна пользователю доступна информация о местонахождении посылок, которые являются отправленными, а также доступна дополнительная информация такая, как: описание посылки, итоговая стоимость, местонахождение, индекс.

## Главное окно администратора

Пройдя процедуру авторизации, если пользователь является администратором, открывается главное окно администратора, представленное на рисунке 5.7.

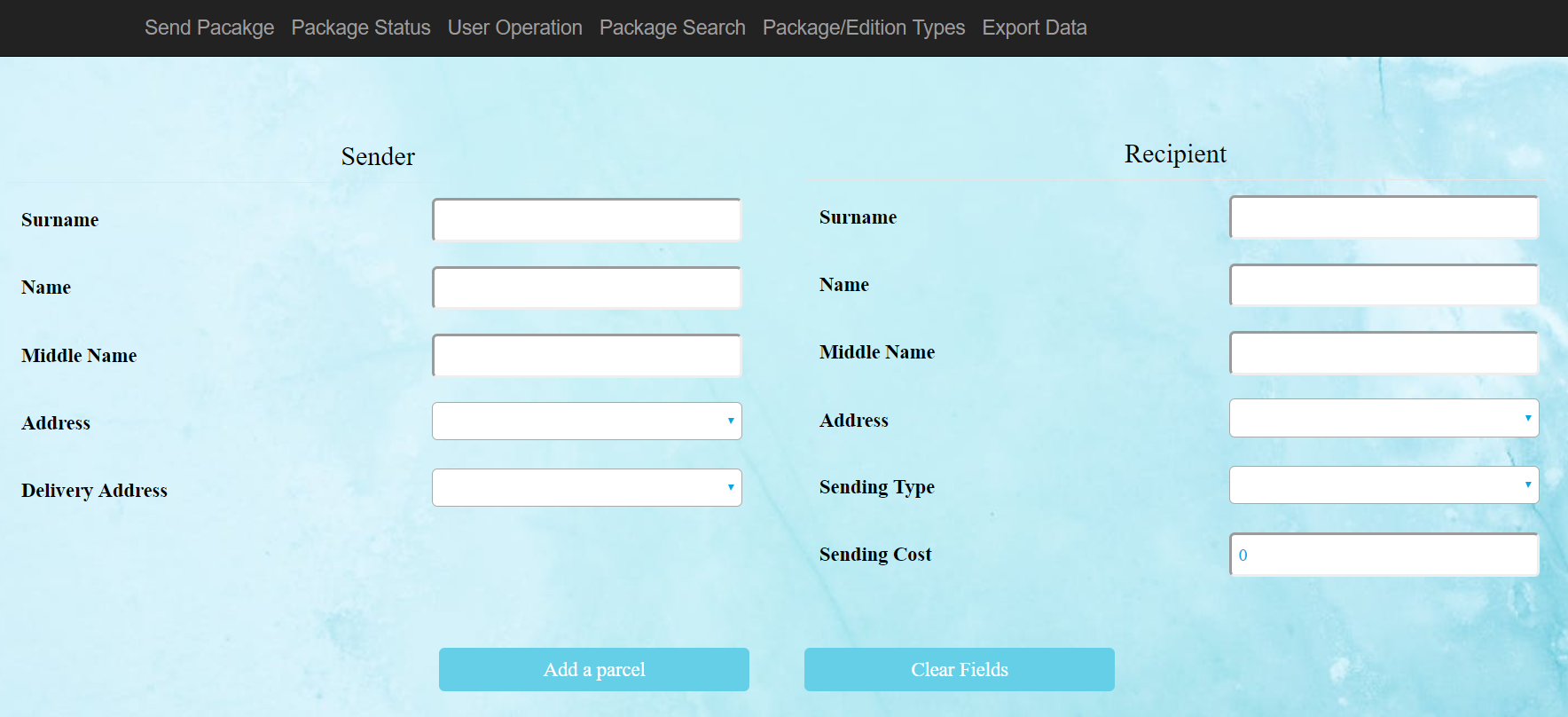


Рисунок 5.7 – Главное окно администратора

Данное окно позволяет администратору добавить отправленную посылку от клиента и клиенту. Ввод поля осуществляется либо вручную, либо выбором элементов из соответствующих списков.

Добавление посылки осуществляется кнопкой «Добавить посылку».

Кнопка «Очистить поля» позволяет, при необходимости очистить все поля данной формы.

При выборе пункта меню «Издание\Посылка» администратор может управлять посылками и изданиями предоставляемыми почтой. Форма представлена на рисунке 5.8.

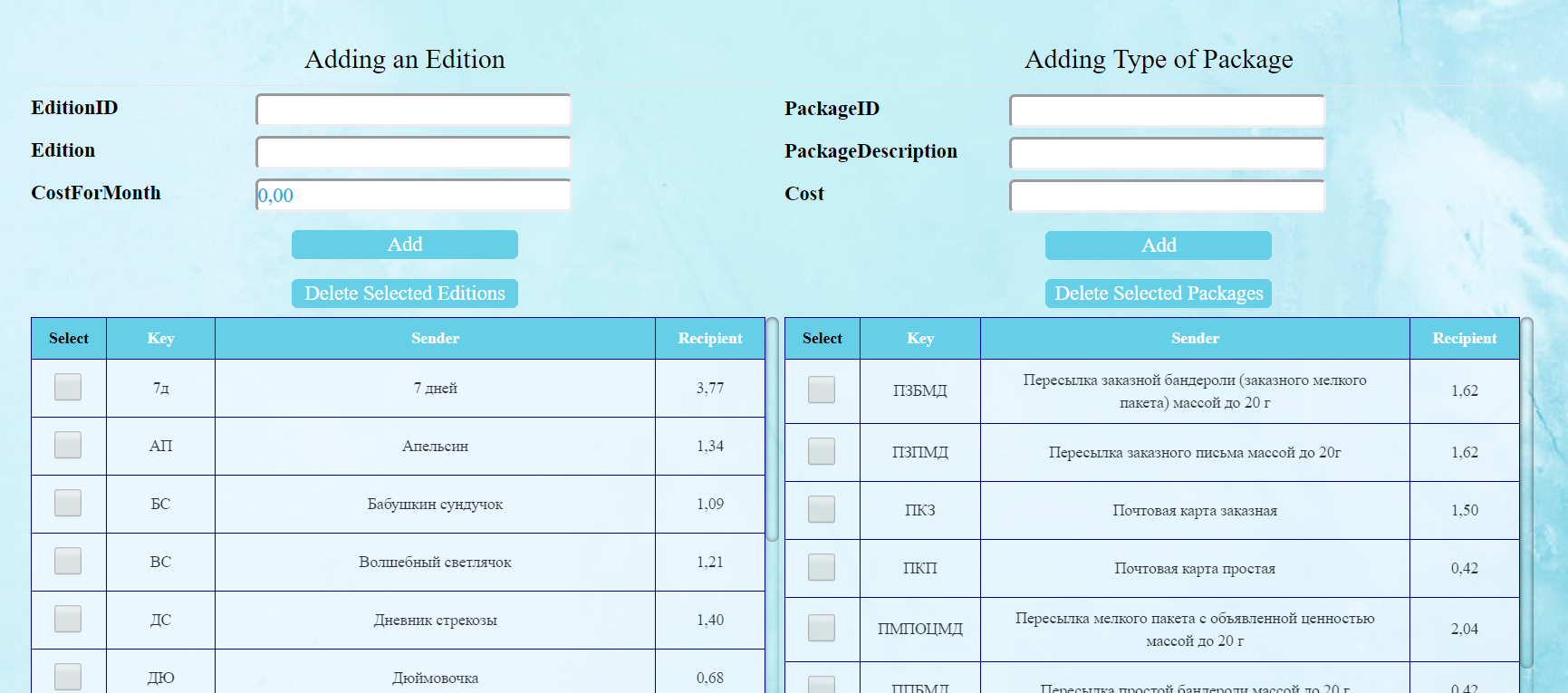


Рисунок 5.8 – Пункт меню «Издание\Посылка»

Добавление нового элемента осуществляется на кнопку «Добавить».

Кнопка «Удаление выбранных Изданий\Посылок» позволяет удалить выбранные посылки либо издания и обновить таблицы, отображающие информацию.

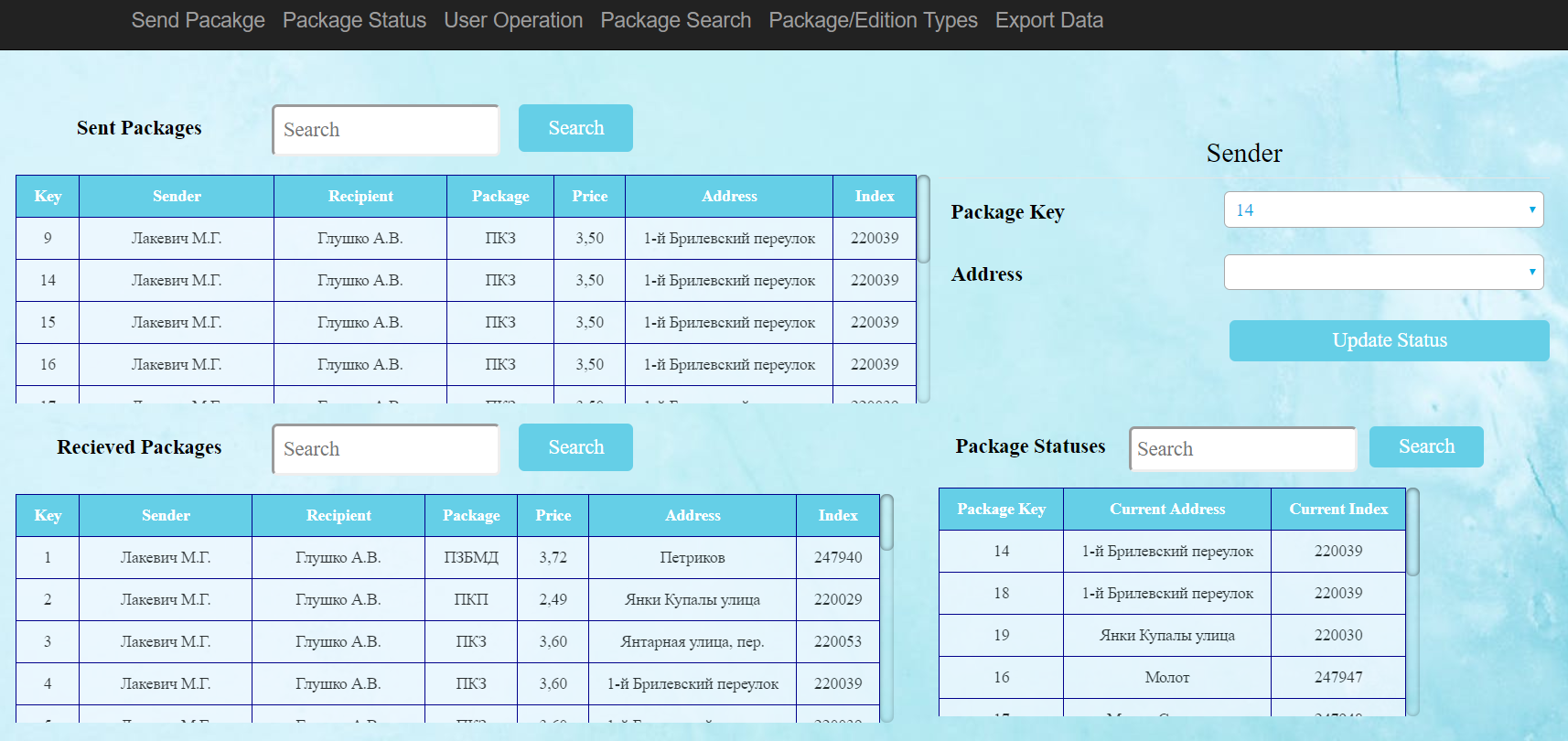


Рисунок 5.9 – Пункт меню «Статус посылки»

При выборе пункта меню «Статус посылки» открывается форма, представленная на рисунке 5.9.

Данная форма позволяет на кнопку «Добавить» добавить текущее местоположение посылки по ключу посылки. При добавлении местонахождения посылки, которое является конечным, запись автоматически помещается в таблицу полученных посылок, и удаляется из двух оставшихся таблиц.

При выборе пункта меню «Операции пользователя» открывается форма, представленная на рисунке 5.10.

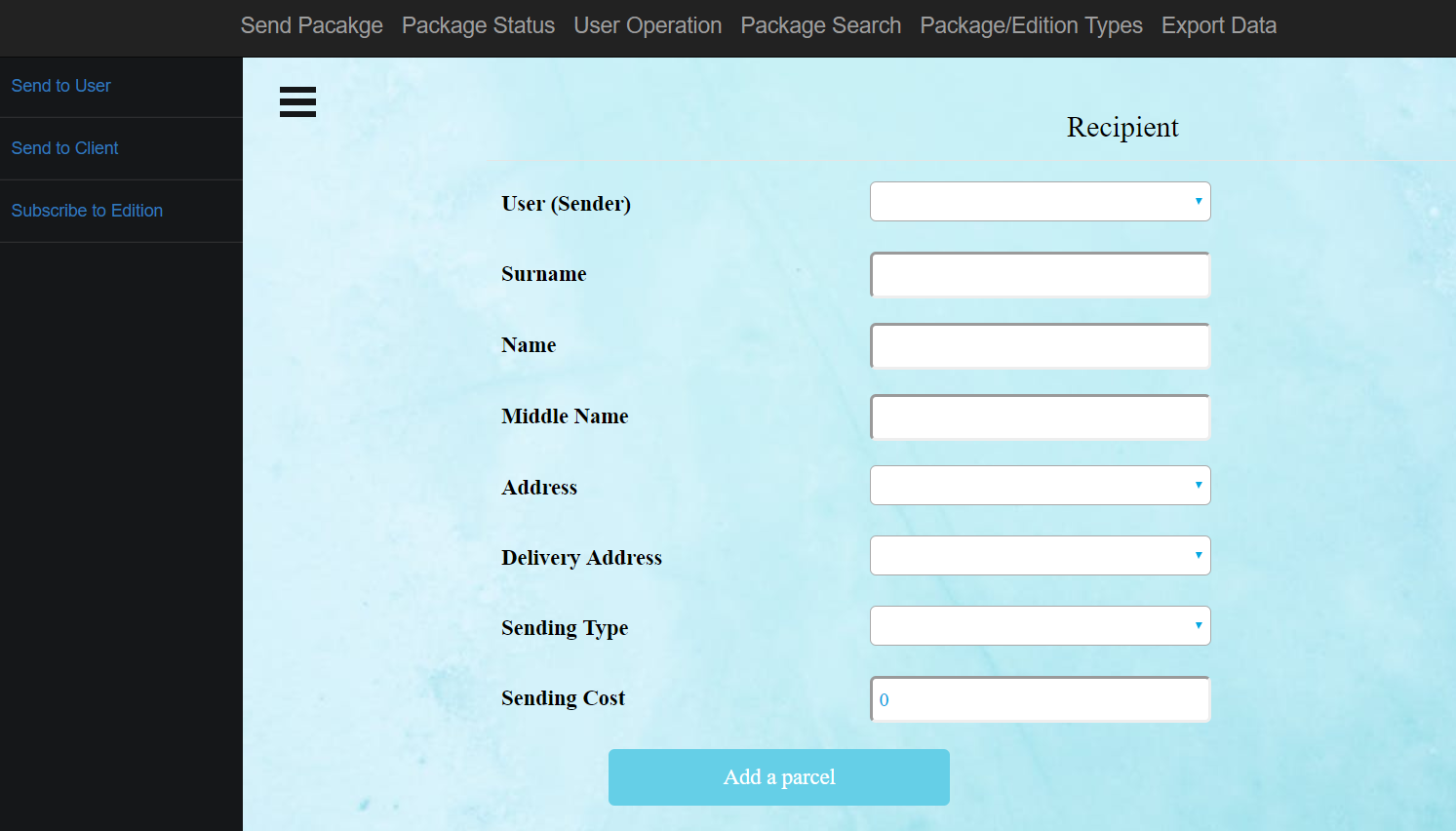


Рисунок 5.10 – Пункт меню «Операции пользователя»

Данная форма, в зависимости от выбранного пункта меню слева, позволяет выполнить следующий перечень функций:

* добавление отправленной посылки от пользователя пользователю;
* добавление отправленной посылки от пользователя клиенту;
* добавление подписки на издание пользователя.

Перечень данных функций выполняется на соответствующую кнопку «Добавить».

Также как и на других формах для заполнения, на этой форме поддерживается авто заполнение индекса или адреса, в зависимости от входного параметра ввода.

При выборе пункта меню «Экспорт данных» открывается форма, представленная на рисунке 5.11.

Данная форма предназначена для сохранения информации об отправленных посылках, а также об активных изданиях на данный момент.



Рисунок 5.11 Пункт меню «Экспорт данных»

На данной форме автоматически отображаются отправленные посылки, а также текущие подписки.

Сохранение информации осуществляется при выборе соответствующей кнопки, находящейся над таблицей.

При нажатии на кнопки «Сохранить текущие статусы посылок» и «Сохранить текущие статусы подписок» сгенерированный Excel документ сохраняется в папку загрузок соответствующего ПК.

Вид сохраняемого файла представлен на рисунке 5.12.

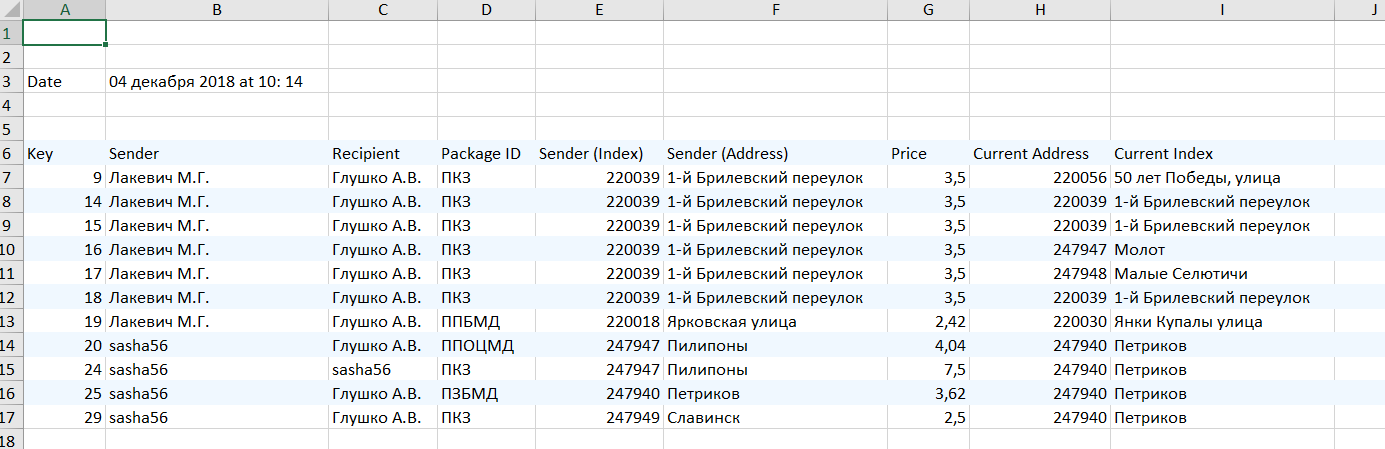


Рисунок 5.12 – Вид сохраняемого файла.

Сохраненный файл содержит необходимую информацию для последующей ее обработки, а также текущую дату составления отчета.

# Тестирование

Для тестирования приложения был выбран интерфейс Visual Studio 2017.

Главным требованием к приложению является быстродействие и корректность работы. Так как в приложении осуществляется загрузка ресурсов необходимых для дальнейшей работы при генерации страницы, и все ресурсы загружаются при инициализации соответствующих окон, то после загрузки окна приложение тратит максимум ~150 Мб памяти, и выше этой планки не поднимается. Результаты тестов представлены на рисунке 6.1.

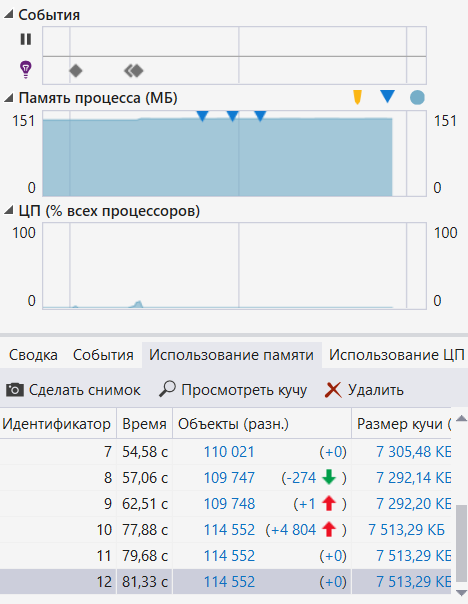


Рис 6.1 – Результат тестирования

Данный результат является хорошим, так как объемы памяти в настоящее время довольно велики, и даже на слабых девайсах приложение покажет хорошие результаты. Так как 150 Мб может тратиться только в какой-то один такт времени, и при наличии более слабого ПК разница будет почти не заметна.

# Заключение

В ходе написания курсового проекта мною были подробно рассмотрены принципы работы отделений почты и возлагаемые на них функции. К наиважнейшим из них относят пересылка писем и посылок, осуществление подписки на различные издания.

В процессе работы над приложением почты мною было предложено программное решение важных функций отделений почтовой связи и рассмотрение почты не только со стороны администратора, но и со стороны пользователя.

Разработанное приложение включает реализацию следующих основных функций:

* письменная корреспонденция (вес до 2-х кг, в том числе письма и мелкие пакеты);
* посылки весом до 20 кг;
* сохранение рабочей информации в централизованной базе данных;
* составление рабочих планов;
* анализ данных, создание отчётов о проделанной работе;
* настройка параметров работы приложения;
* авторизация для разных групп пользователей;
* доступ /доступ и работа с информацией /изменение информации (для разных групп пользователей);
* проверка местонахождения посылки.

Хоть основными функциями являются пересылки почтовых отправлений, но приложение позволяет решать и другие задачи. В частности, предусмотрена работа с Yandex Map, для работы с которыми использовался JS API.

# Список литературы

1. Developer Community – <https://stackoverflow.com/>
2. Программная библиотека для работы с картами в браузерах, а также в приложениях через WebView. – <https://tech.yandex.ru/maps/jsapi/>
3. Регулярные выражения в .NET Framework – [https://msdn.microsoft.com/ru-ru/library/hs600312](https://vk.com/away.php?to=https%3A%2F%2Fmsdn.microsoft.com%2Fru-ru%2Flibrary%2Fhs600312&cc_key=)
4. How to: Create and Execute an SQL Statement that Returns Rows – <https://msdn.microsoft.com/en-us/library/fksx3b4f.aspx>
5. Коллекции (C# и Visual Basic) - MSDN – Microsoft – <https://msdn.microsoft.com/ru-ru/library/ybcx56wz(v=vs.120).aspx>
6. ASP.NET MVC 5 с примерами на C# 5.0 для профессионалов – Адам Фримен