МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-98 01 03 Программное обеспечение информационной безопасности мобильных систем

Направление специальности 1-98 01 03 Программное обеспечение информационной безопасности мобильных систем

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Современные технологии программирования мобильных систем»

Тема Программное средство «Запись к терапевту»

Исполнитель

студент (ка) 2 курса группы 9 Кульнис Александр Юрьевич

(Ф.И.О.)

Руководитель работы ассистент Панченко О.Л.

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

Курсовой проект защищен с оценкой

Председатель Пацей Н.В. (подпись)

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПЕУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра программной инженерии

Утверждаю

Заведующий кафедрой

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_ Н.В. Пацей

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** "Объектно-ориентированное программирование"

|  |  |
| --- | --- |
| Специальность: 1-98 01 03 Программное обеспечение информационной безопасности мобильных систем  Студент: Кульнис А.Ю. | Группа:\_\_9\_\_ |
| **Тема: Программное средство «Запись к терапевту»** | |

**1. Срок сдачи студентом законченной работы**: "20 мая 2021 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС поддерживает:

* Выполнять функции администратора:
  + Выполнять авторизацию;
  + Просматривать список пользователей;
  + Добавление новых пользователей;
  + Удаление старых пользователей.
* Функции клиента:
  + Выполнять регистрацию и авторизацию;
  + Заполнение формы с личной информацией;
  + Возможность записаться на приём к терапевту;
  + Возможность просматривать личную карточку.

**2.2.** При выполнении курсового проекта необходимо использовать принципы и приемы ООП. Приложение разрабатывается под ОС Windows и представляет собой настольное приложение (desktop). Отображение, бизнес логика должны быть максимально независимы друг от друга для возможности расширения. Диаграммы вариантов использования, классов реализации задачи, взаимодействия разработать на основе UML. Язык разработки проекта – C#.

**3. Содержание расчетно-пояснительной записки**

(перечень вопросов подлежащих разработке)

* Введение
* Постановка задачи и обзор литературы (алгоритмы решения, обзор прототипов, актуальность задачи)
* Проектирование архитектуры проекта (структура модулей, классов).
* Заключение
* Список используемых источников
* Приложения

**4. Форма представления выполненной курсовой работы:**

* + Теоретическая часть курсового проекта должны быть представлены в формате docx. Оформление записки должно быть согласно выданным правилам.
  + Листинги программы представляются частично в приложении.
  + Пояснительную записку, листинги, проект (инсталляцию проекта) необходимо загрузить диск, указанный преподавателем.

#### Календарный план

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 19.02.2021 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 12.03.2021 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 26.03.2021 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 2.04.2021 |  |
| 5 | Кодирование программного средства | 23.04.2021 |  |
| 6 | Тестирования и отладка программного средства | 30.04.2021 |  |
| 7 | Оформление пояснительной записки | 7.05.2021 |  |
| 9 | Сдача проекта | 20.05.2021 |  |

**5. Дата выдачи задания \_\_\_\_**12.02.2021**\_\_\_\_**

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *О.Л. Панченко*

(подпись)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата и подпись студента)

Содержание

[ВВЕДЕНИЕ 3](#_Toc41541454)

[1 Аналитический обзор прототипов и литературных источников 4](#_Toc41541455)

ВВЕДЕНИЕ

Данный курсовой проект посвящён разработке программного средства «Запись к терапевту», основной целью которого является возможность пользователей записаться на приём к врачу.

Приложение «Запись к терапевту» предназначено для простого использования в случае необходимости посетить врача-терапевта. Данная программа проста в использовании и не требует никаких усилий, чтобы успешно забронировать время на приём к врачу.

В качестве интерфейса приклaдного программирования был выбран обширный API-интерфейс — Windows Presentation Foundation (WPF), предназначенный для создания настольных программ с графически насыщенным пользовательским интерфейсом.

Для работы с WPF использовался объектно-ориентированный язык программирования с С-подобным синтаксисом — С#, разработанный для создания приложений на платформе Microsoft .NET Framework.

Хранение данных осуществляется в Microsoft SQL Server.

Цель курсового проекта – создать бесплатный простой и быстрый инструмент для записи к врачу, который может быть использован в случае необходимости посетить врача-терапевта.

1. Аналитический обзор прототипов и литературных источников

Немаловажным этапом в разработке программного продукта является аналитический обзор прототипов и литературных источников.

Здесь я буду рассматривать бесплатные приложения, или приложения низкой стоимости. Я взял несколько программ для записи к врачу, но их нельзя назвать аналогами моего ПС, так как они разрабатывались целой командой под началом известных фирм, однако ориентировался я на их функционал. Все программы являются desktop-приложениями.

Desktop-приложения для записи к врачу:

* «Запись к врачу | записаться на прием онлайн»;
* «Aibolit онлайн-запись к врачам Беларуси».

«Запись к врачу | записаться на прием онлайн» - приложение, где есть все врачи и все больницы. Вы можете записаться онлайн на прием к врачу в любую государственную больницу.

Интерфейс «Запись к врачу | записаться на прием онлайн» представлен на рисунке 1.1.

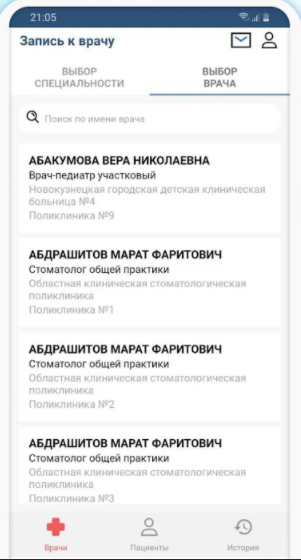


Рисунок 1.1 − Интерфейс «Запись к врачу | записаться на прием онлайн»

Проанализировав «Запись к врачу | записаться на прием онлайн», можно выделить основные минусы данного программного сред.

Основные минусы:

* Загруженный интерфейс;
* Нет записи для того что записаться на приём, врачей;

Основные плюсы:

* Поддерживается 1С;
* Простой интерфейс;
* Надежность;

Продукт «Aibolit онлайн-запись к врачам Беларуси» - приложение для онлайн записи к врачам по Беларуси для сайтов и соцсетей ресторанов и кафе. Онлайн приём броней, депозиты, статистика, база гостей, банкеты. Настраивается за 15 минут, работает в облаке. Стоит от 0 руб. в месяц.

Интерфейс «Aibolit» представлен на рисунке 1.2.

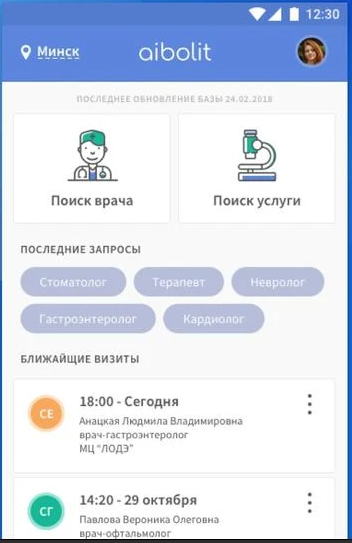


Рисунок 1.2 − Интерфейс «Aibolit»

Проанализировав ПО «Aibolit», можно выделить её основные минусы и плюсы.

Основные минусы:

* Примитивный интерфейс и плохие цвета;
* Отсутствие Фильтрация

Основные плюсы:

* Большой функционал;
* Простота в использовании;
* Наличие поисковой строки;

«К врачу» - Облачное решение для автоматизации ресторана, кафе, магазина или кальянной. Рабочее место кассира и официанта. Управление меню, складом, лояльностью, отчеты. Интеграция с принтерами чеков и фискальными регистраторами.

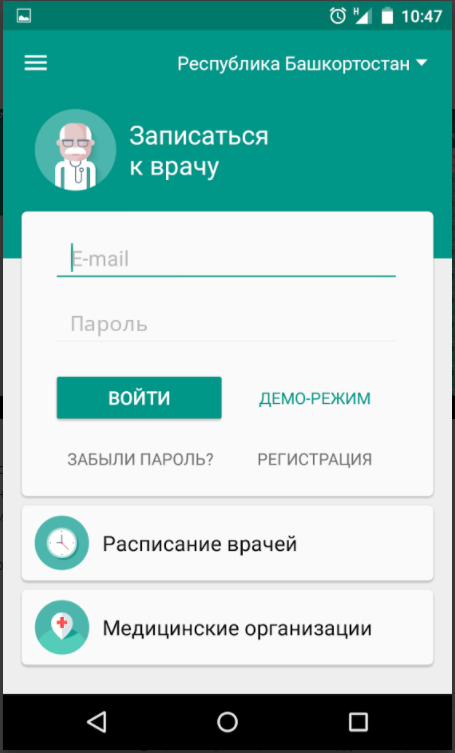
Интерфейс «К-врачу» представлен на рисунке 1.3.

Рисунок 1.3 – Интерфейс «К-врачу»

Данное ПС грамотно спроектировано и выполняет большое количество функций.

Основные минусы:

* Неудобное расписание;
* Много лишних категорий.

Основные плюсы:

* Может интегрироваться с кассовой системой;
* Возможность регистрации;

2. **Анализ требований к программному средству и разработка функциональных требований**

Задачей курсового проекта является разработка приложения, предназначенного для записи к терапевту, а также ряда дополнительных операций: добавления юзеров, редактирования, удаления; возможность заполнить личную карту пациента, забронировать талон, отменить бронирование. Информацию о талонах и услугах вводится непосредственно с самого приложения и проверяется администратором.

Программный продукт должен быть реализовать на объектно-ориентированном языке программирования C# с использованием технологии WPF. Windows Presentation Foundation (WPF) — система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

В основе WPF лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language), элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление.

Графической технологией, лежащей в основе WPF, является DirectX, в отличие от Window Forms, где используется GDI/GDI+. Производительность WPF выше, чем у GDI+ за счёт использования аппаратного ускорения графики через DirectX.

Также существует урезанная версия CLR, называющаяся WPF/E, она же известна как Silverlight.

База данных была разработана в Microsoft SQL Server.

SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире.

SQL Server характеризуется такими особенностями как:

* Производительность. SQL Server работает очень быстро.
* Надежность и безопасность. SQL Server предоставляет шифрование данных.
* Простота. С данной СУБД относительно легко работать и вести администрирование.

Для организации баз данных MS SQL Server использует реляционную модель, которая предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта.

Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос на языке SQL должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения.

Основной используемый язык запросов —Transact-SQL — реализован на структурированном языке запросов (SQL) с расширениями.

Для осуществления связи между базой данных и приложением на C# необходим посредник, которым будет являться технология Entity Framework. Она основана на платформе .NET Framework и предназначена для работы с данными.

Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Эта технология предоставляет нам набор классов, через которые мы можем отправлять запросы к базам данных, устанавливать подключения, получать ответ от базы данных и производить ряд других операций.

Entity Framework предполагает три возможных способа взаимодействия с базой данных:

* Database first: Entity Framework создает набор классов, которые отражают модель конкретной базы данных;
* Model first: сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере;
* Code first: разработчик создает класс модели данных, которые будут храниться в базе данных, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы.

В курсовом проектировании для разработки был выбран первый подход.

Приложение необходимо реализовать с использованием шаблона проектирования MVVM и с организацией взаимодействия с базой данных. В качестве клиентской части необходимо реализовать оконное приложение с использованием стандартных и иных библиотек. Приложение должно иметь возможность запускаться без использования интегрированных средств разработки.

Для разработанного продукта необходимо предусмотреть следующие задачи:

* Главное поле для отображения данных;
* Интуитивно понятный интерфейс;
* Механизм поиска товаром по названию, категории, цене;
* Возможность просмотра информации о товаре и о продавце.  
  По итогу работы над продуктом, его необходимо протестировать и отладить, проанализировать его возможности и, по необходимости, внести дополнения или улучшения проекта, с возможностью дальнейшего расширения проекта.

Программное средство предоставляет пользователю следующие функциональные возможности:

* Регистрация и идентификация пользователей
* Заполнение личной карточки
* Возможность забронировать талон к врачу
* Возможность выбрать врача и дату и получить сведения о свободных талонах на этот день
* Возможность отменить забронированный талон
* Просмотр забронированных талонов
* Просмотр истории посещения

Администратор может:

* Добавление пользователей
* Редактирование пользователей
* Удаление пользователей
* Осуществлять поиск по некоторым критериям
* Просмотр всех пользователей
* Подтверждение посещения пользователя врача
* Очистка всех талонов, которые просрочены

Наглядно это видно на UML-схеме, изображённой на рисунке 2.1 (Приложение А).

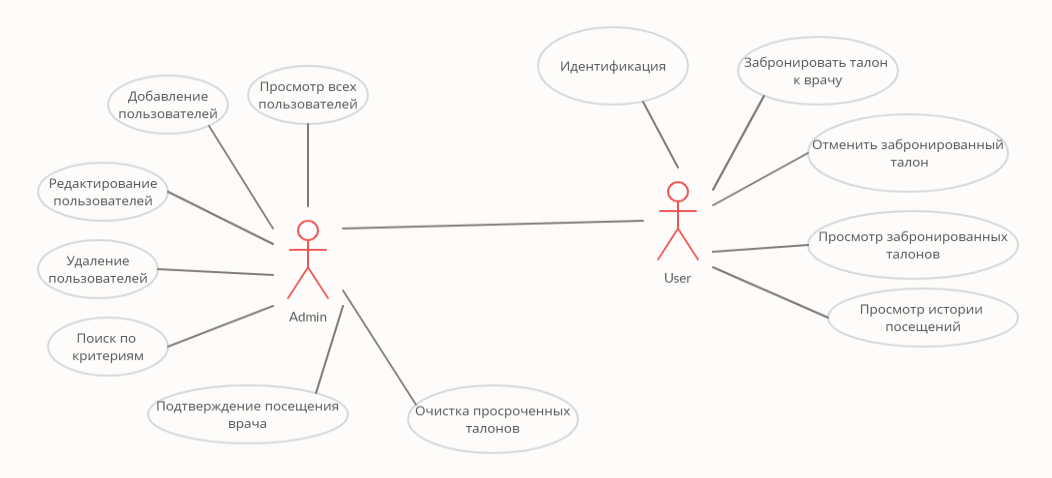


Рисунок 2.1 – «UML-диаграмма возможностей»

1. **Проектирование программного средства**

Рисунок 3.1 – «Авторизация и главное окно»

На рисунке 3.1 представлена общая схема работы приложения.

При запуске приложения у нас появляется окно авторизации, которое требует ввода логина и пароля для начала работы.

В случае, если вас нету своей учётной записи, её можно создать, нажав на кнопку «Регистрация» в окне авторизации. Вы будете направлены в окно регистрации, где вам нужно будет заполнить все поля. После успешной регистрации, вам нужно будет вернутся в окно авторизации и ввести ваши данные, указанные при регистрации.

Нажав на соответствующие кнопки в окне авторизации, у вас появится окно, где нужно будет ввести данные для входа в учётную запись, через которую производится вход.

После ввода логина и пароля в соответствующих полях, проверяется правильность ввода данных.

В случае неверного ввода данных, выведется соответствующее сообщение. В ином случае окно авторизации закроется и откроется главное окно приложения.

В главном окне работа идёт в страничном и оконных режимах.

Рассмотрим каждую страницу в отдельности.

Страница «Все объявления» (рисунок 3.2).

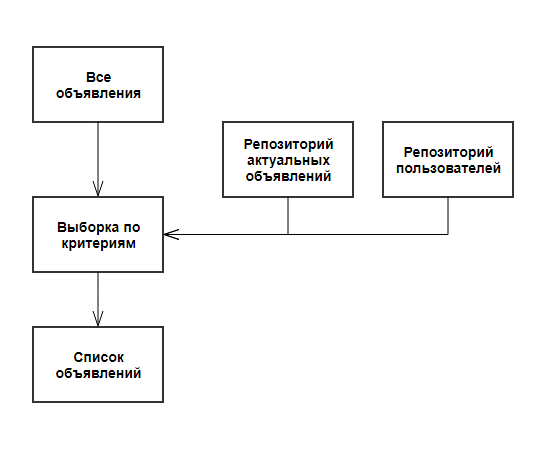


Рисунок 3.2 – «Все объявления»

Эта страница содержит список всех актуальных объявлений.

Также она содержит механизм поиска объявления по определённым критериям.

Осуществить поиск можно по таким критериям:

* Слово в названии или описании
* Продавец
* Цена
* Регион
* Категорий
* Комбинации вышеперечисленных

На основе критериев поиска, формируется результирующий набор. При нажатии мы видим более подробную информацию о объявлении и получаем возможность его закрепления, чтобы была возможность сравнить несколько объявлений между собой.

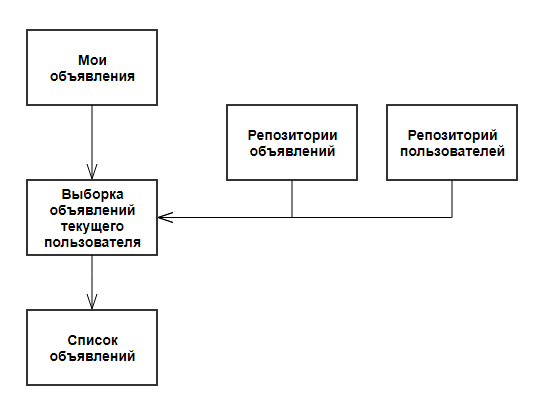


Рисунок 3.3 – «Мои объявления»

На рисунке 3.3 показана схема работы страницы «Мои объявления».

На этой странице ведётся редактирование, удаление, просмотр только тех объявлений, которые размещал вошедший пользователь.

Есть переключатель, который позволяет переключатся между объявлениями, которые ещё не прошли проверку и теми, которые уже находятся в разделе «Актуальные».

Редактирование происходит в отдельном окне, которое появляется при нажатии на соответствующую кнопку. В этом окне есть ограничения и коррекция недопустимых значений. На название отводится от 5 до 50 символов, а на описание от 10 до 1000 символов. Также нельзя установить отрицательную цену т.к. она, при некорректно введённых значениях, будет обращаться в нуль или в то значение, которое было изначально.

При просмотре своего объявления, открывается окно, в котором выводится вся информация о данном объявлении.

Если происходит изменение объявления из раздела «Актуальные», то после изменения, оно автоматически попадает в раздел «Непроверенные» и становится невидимым для других пользователей.

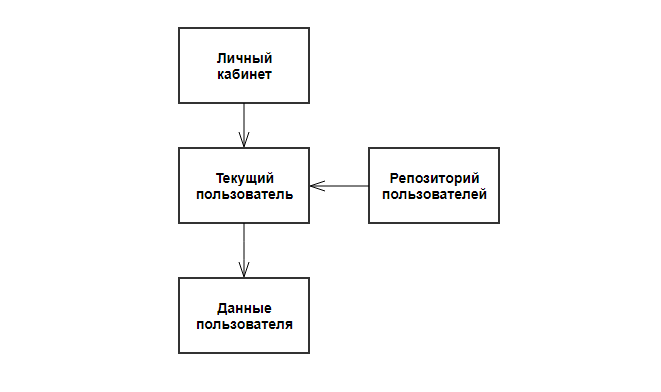


Рисунок 3.4 – «Личный кабинет»

На рисунке 3.4 показана схема страницы «Личный кабинет».

На этой странице можно редактировать свои данные, такие как:

* Имя
* Фамилия
* Mail
* Телефон
* Описание (о себе)
* Картинка профиля

Под картинкой профиля находится кнопка «Загрузить», нажатие на которую приводит к открытию окна с выбором файла картинки профиля, которую вы хотите установить в качестве основой.

На этой странице есть 2 кнопки: «Сохранить изменения», «Удалить свой профиль».

При нажатии на 1-ю кнопку происходит обновление информации о текущем пользователе в базе данных.

При на нажатии на 2-ю кнопку – удаление всех объявлений текущего пользователя, последующее удаление самого профиля и перенаправление на окно авторизации.

Также на эту страницу можно попасть, нажав на элемент, находящийся в верху самого окна чуть правее середины, и содержащий имя и фамилию текущего пользователя, а также инициалы.

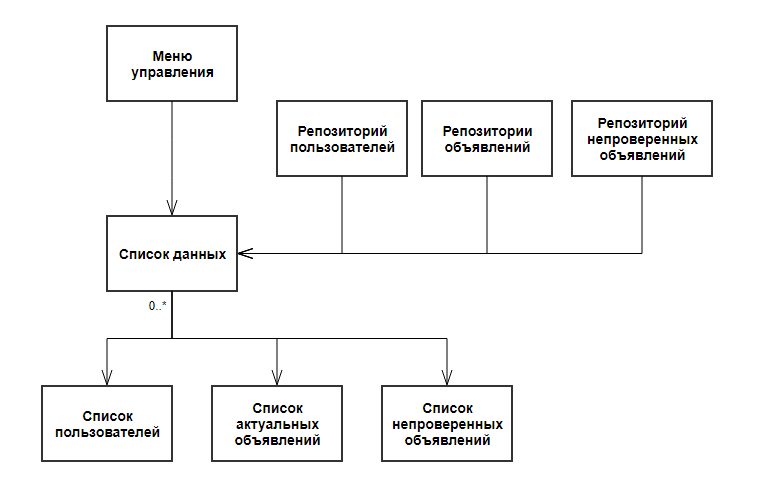


Рисунок 3.5 – «Меню управления»

В этом окне у нас производится администрирование нашего приложения.

Всего есть 3 типа привилегий:

* Пользователь
* Модератор
* Администратор

Пользователь не имеет доступ к этой странице. При попытке зайти в это меню, у него будет выведено сообщение о недостатке прав для доступа к этой странице.

Модератор имеет доступ к этой странице и имеет возможности:

* Проверка всех объявлений
* Удаление объявлений как из разряда «Актуальные», так и из разряда «Непроверенные»
* Просмотр информации о всех пользователях
* Одобрение и перенос объявления в «Актуальные»

Администратор имеет все те же права что и модератор, а также возможность назначать и изменять привилегии пользователей и удалять пользователей из приложения.

Структура проекта представлена в приложении В.

В таблице 3.7 описана структура проекта по пакетам классов.

В этой таблице приведены основные логические составляющие пакеты, которые используются в работе приложения.

Таблица 3.7 – Описание структурных пакетов проекта

|  |  |
| --- | --- |
| Имя пакета | Описание |
| Model | Здесь описаны модели, с которыми происходит вся работа в приложении:   * Пользователь * Актуальное объявление * Непроверенное объявление * Регион |
| Repositories | Содержит классы и интерфейсы для работы с данными БД. Они являются связующим элементом между данными в БД и теми данными, с которыми работает приложение |
| Resources | Файлы стилей |
| View | Содержит все представления, которые позволяют пользователю работать с приложением. Они описывают графическую составляющую приложения. |
| ViewModel | Содержит логику, которая позволяет получить данные при помощи View, обработать их, использую при этом Model, после чего передать в базу данных. |
| App.config | Файл конфигурации приложения |
| App.xaml | Основная задача данного файла состоит в определении ресурсов, общих для приложения |

Структуры данных, а также их зависимости друг от друга представлены в приложении Б.

Как показано в приложении Б, в вершине иерархии стоят классы Announcement и TmpAnnouncement, который характеризует актуальное и непроверенное объявление соответственно. Класс Region характеризует регион, в котором может быть размещено объявление. User, характеризует пользователя с любым уровнем доступа. Announcement и TmpAnnouncement связаны с User по полю seller и id соответственно. Announcement и TmpAnnouncement связаны с Region по полю idRegion и id соответственно. TmpAnnouncement и Announcement между собой никак не связаны. Они по разным таблицам для того, чтобы не сливать воедино объявления всех типов.

Диаграмма классов показана в приложении Г.

# **Создание (реализация) программного средства**

При создании приложения использовался паттерн проектирование MVVM. Он заключается разделении представления от бизнес логики. Это достигается за счёт ввода новой логической конструкции ViewModel. ViewModel связывает представление и бизнес логику приложения.

Пакетная структура проекта показана на рисунке 4.1

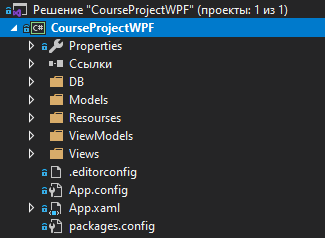


Рисунок 4.1 – «Структура пакетов проекта»

В Model находится структура модели базы данных приложения. При проектировании базы данных данного курсового проекта для взаимодействия с базой данных используется Entity Framework.

Entity Framework представляет специальную объектно-ориентированную технологию на базе фреймворка .NET для работы с данными. Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами.

Отличительной чертой Entity Framework является использование запросов LINQ для выборки данных из БД. С помощью LINQ мы можем не только извлекать определенные строки, хранящие объекты, из бд, но и получать объекты, связанные различными ассоциативными связями.

Нередко все сущности (модель) наследует интерфейс INotifyPropertyChanged или INotifyCollectionChanged, которые позволяют отлавливать изменения и правильно на их реагировать и сохранять.

Для того, чтобы реализовать интерфейс INotifyPropertyChanged, нужно объявить свойство PropertyChanged и метод OnPropertyChanged().

Для того, чтобы начать отслеживание свойства, необходимо вызывать метод OnPropertyChanged(“Имя свойства”) как показано в приложении Д.

Для удобной работы с данными, используется паттерн Repository. Репозиторий позволяет абстрагироваться от конкретных подключений к источникам данных, с которыми работает программа, и является промежуточным звеном между классами, непосредственно взаимодействующими с данными, и остальной программой. На рисунке 4.2 показана структура.

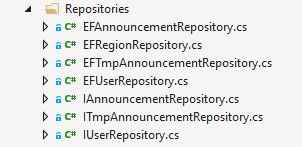


Рисунок 4.2 – «Паттерн Repository»

Интерфейсы IUserRepository, IAnnoouncementRepository, ITmpAnnouncementRepository, IRegionRepository имплицируются в соответствующих классах и указывают: какие методы (функционал) должен иметь этот репозиторий. Репозитории показаны в приложении Е.

Интерфейсы репозиториев также показаны в приложении Е.

На рисунке 4.3 показана структура пакета Resources, в котором находятся файлы стилей кнопок (ButtonDictionary.xaml), а также стили для других элементов управления (Styles.xaml). Эти файлы написаны на языке разметки xaml.



Рисунок 4.3 – «Структура пакета Resources»

В проекте используется паттерн MVVM, а это значит, что должны присутствовать ViewModels, которые связывают View и Model. Во ViewModel информация обрабатываться и выводится/записываться в нужном виде. Структура пакета ViewModel представлена в приложении К.

В приложении И показана структура пакета View, в котором хранятся файлы представлений на языке разметки xaml, а также соответствующие им файлы на языке C#. В этом пакете находятся все «окна» и «страницы», которые мы видим, используя приложение. Назначение каждого файла описано в таблице ниже (таблица 4.4)

Таблица 4.4 – «Описание файлов View»

|  |  |
| --- | --- |
| **Название** | **Краткое описание** |
| AddWindow.xaml (окно) | Окно добавления объявления, в котором указывается информация о будущем объявлении |
| AdminPage.xaml (страница) | Окно для управления работы приложения и ведения проверки данных |
| AlertWindow.xaml (окно) | Окно уведомления |
| AllAnnouncement.xaml (страница) | Страница, отображающая все актуальные объявления |
| AuthWindow.xaml (окно) | Окно авторизации. Требует ввода логина и пароля или выбор способа авторизации |
| DialogWindow.xaml (страница) | Окно, в котором нужно выбрать вариант ответа «Да» или «Нет» на заданный там вопрос |
| EditWindow.xaml (окно) | Окно изменения свойств объявления |
| Enter.xaml (окно) | Окно ввода кода подтверждения |
| InstagramWindow.xaml (окно) | Окно авторизации через Instagram, в котором требуется ввести логин и пароль от аккаунта Instagram |
| MainWindow.xaml (окно) | Главное окно приложения |
| MyAnnouncement.xaml (страница) | Страница, на которой показаны все ваши объявления и выбор действий над ними |
| PersonAreaPage.xaml (страница) | Страница профиля, в которой можно посмотреть или изменить свои личные данные |
| QuickViewWindow.xaml (окно) | Окно для закрепления объявления или его быстрого просмотра |
| RegistrationWindow.xaml (окно) | Окно регистрации нового пользователя |
| UserViewModel.xaml (окно) | Окно просмотра информации о пользователе (доступно для модератора/админа) |
| ViewWindow.xaml (окно) | Основное окно для просмотра информации о объявлениях |
| VKLoginWindow.xaml (окно) | Окно авторизации через Вконтакте, в котором требуется ввести логин и пароль от аккаунта Вконтакте |

В разработке авторизации через «Вконтакте» требовалось зарегистрировать своё приложение на сайте ВК как разработчику (<https://vk.com/apps?act=manage>), после чего вам будет выдан ID приложения, защищённый ключ, сервисный ключ доступа. Используя ID приложения нужно производить авторизацию, но этот ключ не должен быть виден пользователю. Код, из приложения Ж, показывает процесс авторизации, где:

\*\*\*\*\*\*\* – это ID приложения, который получили после регистрации приложения в ВК,

login – логин для входа, password – пароль для входа, Settings – разрешения, которые мы имеем пользуясь приложением для этого аккаунта.

В случае, если у аккаунта стоит двойная аутентификация, code – код подтверждения.

С помощью методов, описанных в классе VkViewModel и VkLoginViewModel мы «подтягиваем» данные из Вконтакте и загружаем их в базу данных (приложение Ж).

Точно такой же принцип работает и для Instagram.

Здесь используется InstaSharp API. Для загрузки данных из Instagram используются классы InstaLogin и InstagramLoginViewModel.

Класс, в котором проходит авторизация через Instagram, представлен в приложении З.

На рисунке 4.5 изображена диаграмма таблиц со связями в базе данных.

Рисунок 4.5 – «Диаграмма таблиц»

Описание таблицы Announcements (таблица 4.6)

Таблица 4.6 – «описание Doctors»

|  |  |
| --- | --- |
| Id | Идентификатор |
| FIO | ФИО врача |
| Post | Специальность врача |
| Cabinet | Кабинет |
| EvenDay\_start | Начало работы врача в чётный день |
| EvenDay\_end | Конец работы врача в чётный день |
| OddDay\_start | Начало работы врача в нечётный день |
| OddDay\_end | Конец работы врача в нечётный день |

Описание таблицы UserDoctors (таблица 4.7)

Таблица 4.7 – «описание UserDoctors»

|  |  |
| --- | --- |
| User\_id | Идентификатор юзера |
| Doctor\_id | Идентификатор врача |

Описание таблицы Medcards (таблица 4.8)

Таблица 4.8 – «описание Medcards»

|  |  |
| --- | --- |
| Id | Идентификатор |
| Name | Имя |
| Surname | Фамилия |
| Patronomic | Отчество |
| Gender | Пол |
| BDay | День рождения |
| City | Город |
| Street | Улица |
| House | Номер дома |
| Housing | Корпус дома |
| Flat | Номер квартиры |
| Image | Изображение |

Описание таблицы HistoryVisitings (таблица 4.9)

Таблица 4.9 – «описание HistoryVisitings»

|  |  |
| --- | --- |
| id | Идентификатор |
| FIO | ФИО доктора |
| Doctor | Специальность |
| Cabinet | Номер кабинета |
| VisitDay | Дата посещения |
| VisitTime | Время посещения |
| Info | Информация в карточке |
| UserId | Идентификатор |

Описание таблицы Recordings (таблица 4.9)

Таблица 4.9 – «описание Recordings»

|  |  |
| --- | --- |
| id | Идентификатор |
| FIO | ФИО доктора |
| Doctor | Специальность |
| Cabinet | Номер кабинета |
| VisitDay | Дата посещения |
| VisitTime | Время посещения |
| UserId | Идентификатор |
| IsVisited | Посещение |

# **Тестирование, проверка работоспособности и анализ полученных результатов**

При тестировании данного приложения, были применены сценарии, которые могли бы привести к ошибке. В этой главе мы рассмотрим некоторые такие сценарии и посмотрим на их обработку.

В момент регистрации, возможна такая ситуация, в которой пользователь ничего не ввёл или ввёл неверные данные. Обработка данного сценария приведена на рисунке 5.1

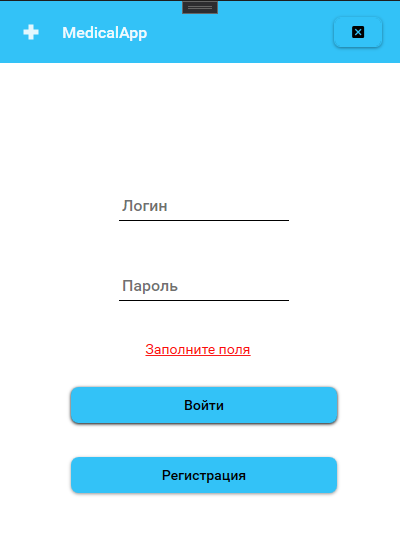


Рисунок 5.1 – «Ошибка корректности ввода»

При неверных данных – рисунок 5.2

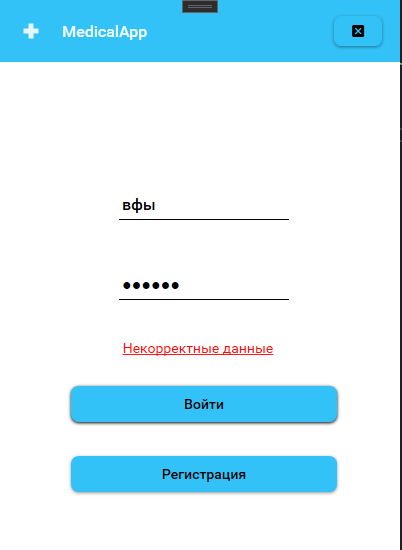


Рисунок 5.2 – «Сценарий, с неверными данными»

Также возможен такие случаи:

* Пользователь хочет зарегистрироваться, при этом не указав никаких данных (рисунок 5.3)
* Пользователь, когда осуществляет ввод паролей, указывает несовпадающие пароли (рисунок 5.4)

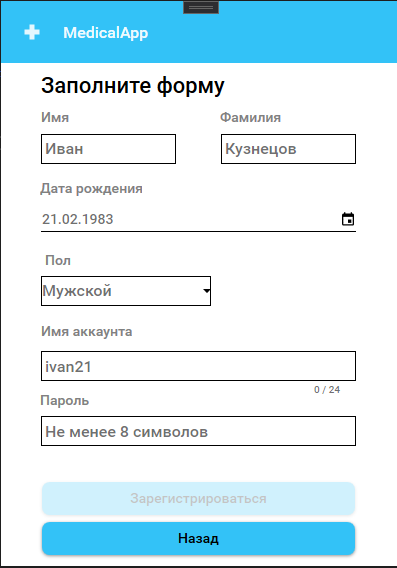


Рисунок 5.3 – «Отсутствие данных»

Как видно на рисунке 5.3, кнопка регистрации просто не будет активна, если не введены данные, необходимые для регистрации.

При несовпадении паролей будет точно такая ситуация – неактивная кнопка «Регистрация»

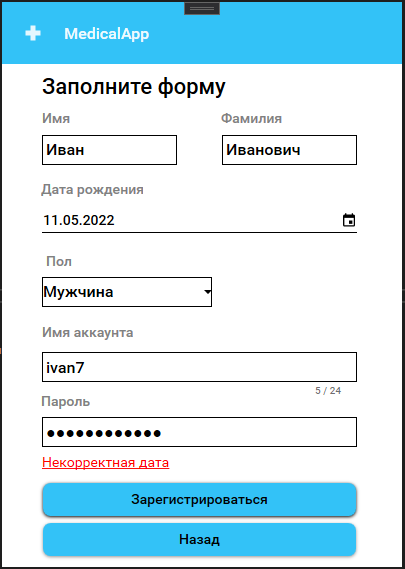


Рисунок 5.4 – «сценарий, при несовпадении паролей»

Также возможен такой исход, при котором пользователь хочет зарегистрироваться при помощи такого mail/логина, который уже есть. Обработка такого рода сценариев показана на рисунке 5.5.

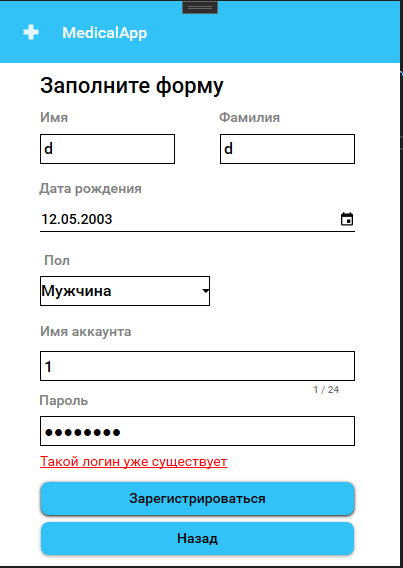


Рисунок 5.5 – «Регистрация пользователя с занятым логином»

Также может возникать такая ситуация, когда пользователь хочет сохранить данные по личной карточке, но кнопка не будет активна пока не будут заполнены необходимые поля. Пример показан на рисунке 5.6

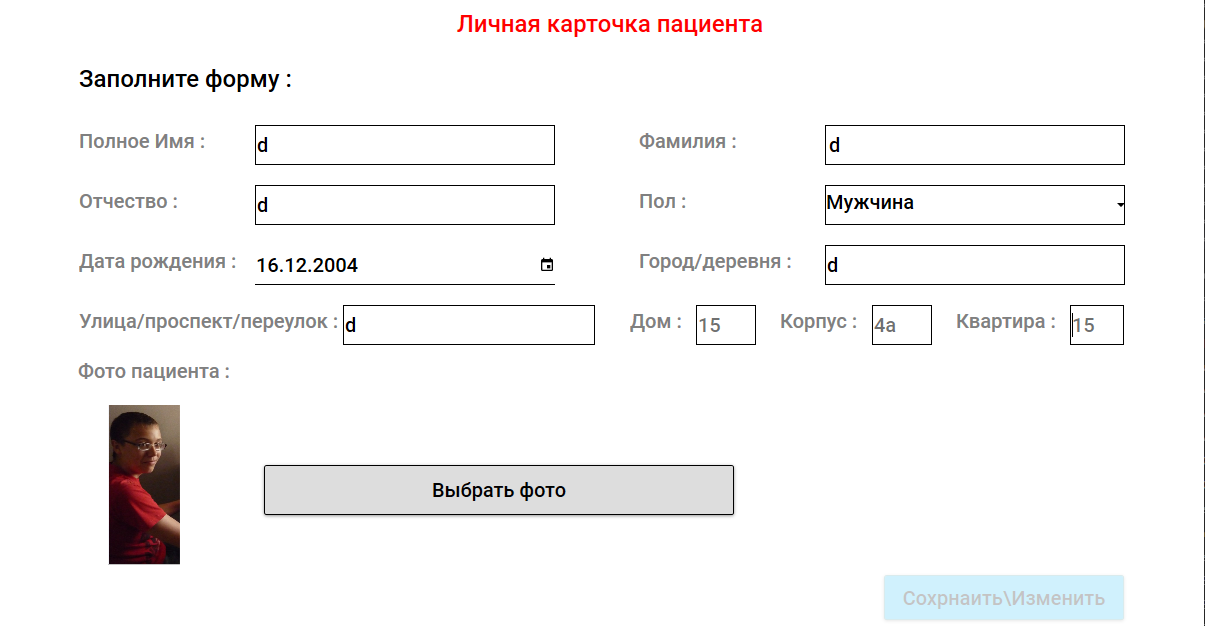


Рисунок 5.6 – «ограничение привилегий»

# **Руководство по использованию**

При запуске приложения у нас появляется окно авторизации, которое требует ввода логина и пароля для начала работы.

В случае, если вас нету своей учётной записи, её можно создать, нажав на кнопку «Регистрация» в окне авторизации. Вы будете направлены в окно регистрации, где вам нужно будет заполнить все поля. После успешной регистрации, вам нужно будет вернутся в окно авторизации и ввести ваши данные, указанные при регистрации. Также поддерживается вход через социальные сети «ВКонтакте» и «Instagram».

Нажав на соответствующие кнопки в окне авторизации, у вас появится окно, где нужно будет ввести данные для входа в учётную запись, через которую производится вход.

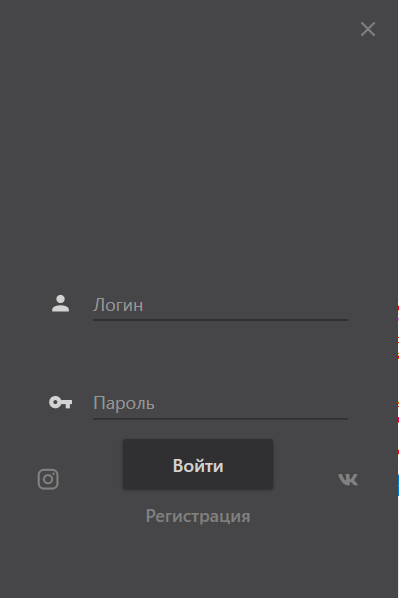


Рисунок 6.1 – «Окно авторизации»

После успешного входа, мы попадаем в главное окно, которое изображено на рисунке 6.2.

В этом окне мы может по нескольким критериям выбрать набор тех объявлений, которые нам подходя. Эти критерии находятся слева страницы. Поиск производится по нажатию кнопки «Поиск».

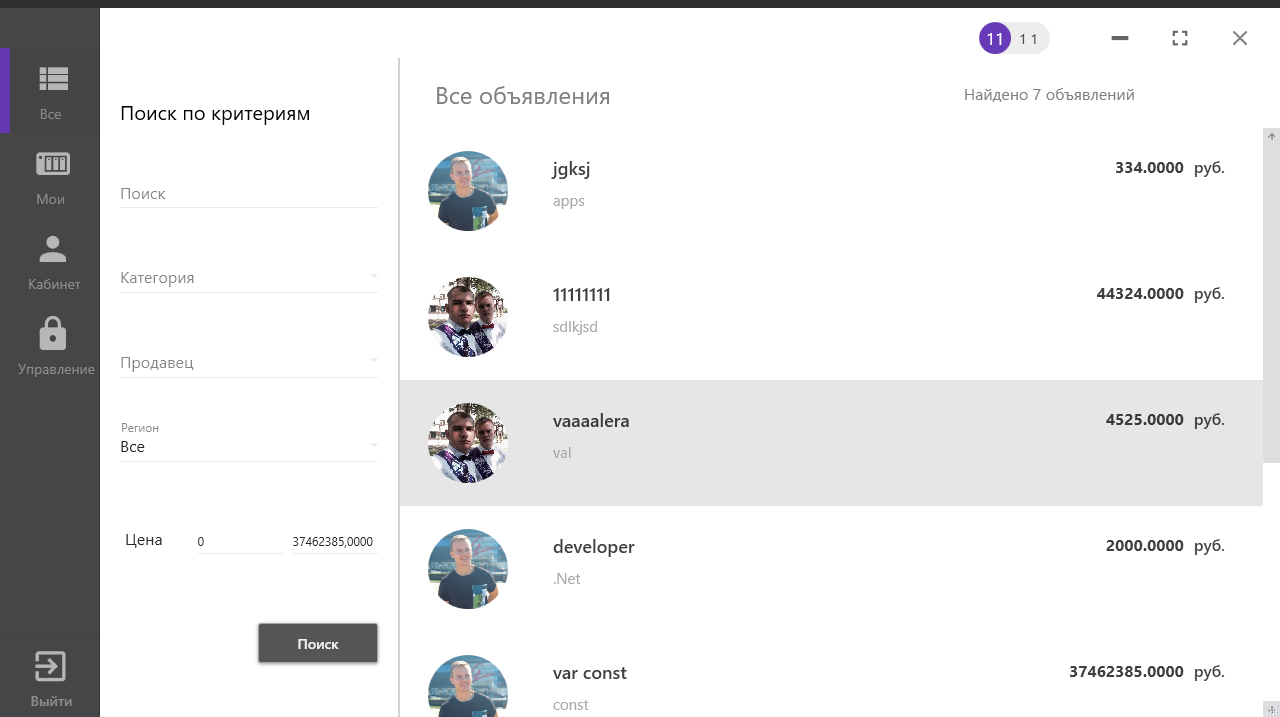


Рисунок 6.2 – «Главное окно»

Для добавления своего объявления нам потребуется перейти на страницу «Мои» и нажать на кнопку « + » в верхнем правом углу экрана. После чего у нас появится окно, в котором нам необходимо будет ввести данные нового объявления, а именно:

* Название
* Категория
* Цена
* Регион
* Описание

После этого, объявление будет показано в разделе «Неактуальные» на странице «Мои».

После проверки этого объявления модератором или админом, оно будет показано как актуальное и видно в общем списке объявлений.

Ниже показан рисунок 6.3, на котором показана страница «Мои».

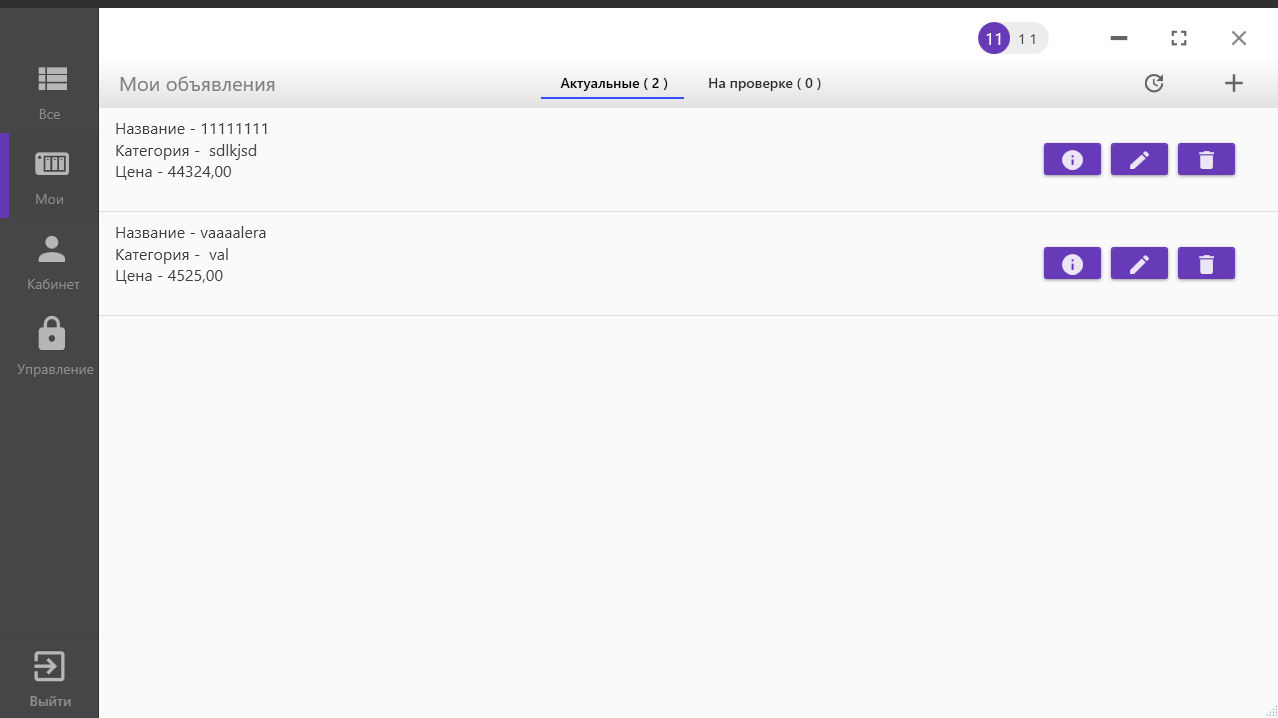


Рисунок 6.3 – «Страница Мои»

Ниже показана страница добавления объявления (рисунок 6.4)

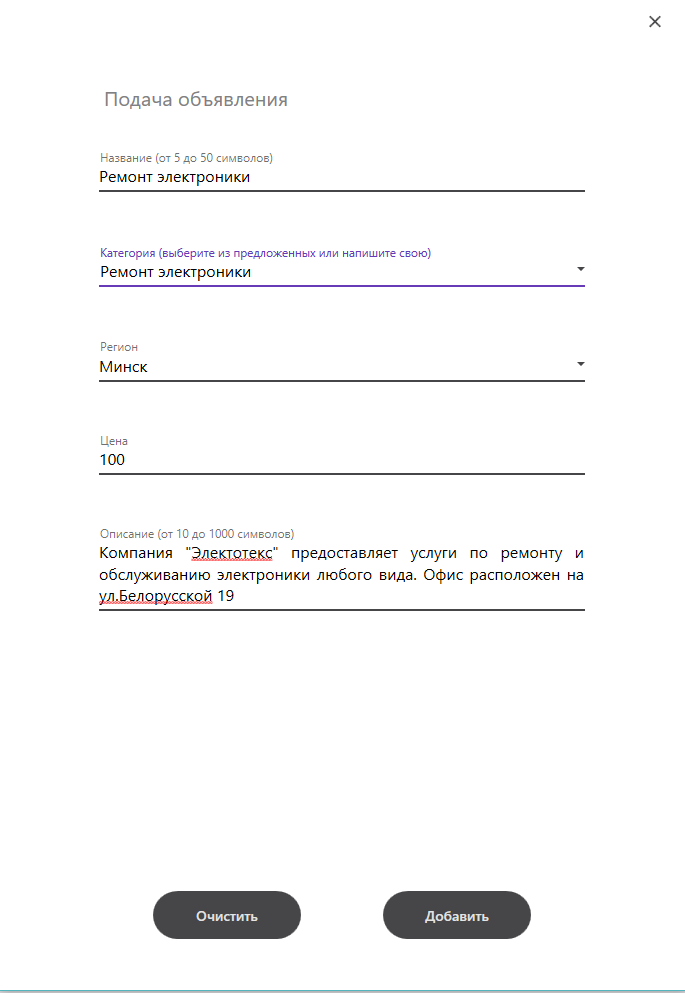


Рисунок 6.4 – «Окно добавления объявления»

Также на странице «Мои» есть кнопки, нажав на которые будут открыты соответствующие иконкам окна:

* Быстрый просмотр
* Редактирование
* Удаление

При переходе на страницу «Кабинет» мы попадаем на страницу, в которой отображены наши личные данные, которые можем изменять. Если был выполнен вход из «Вконтакте» или «Instagram», то все эти поля заполнятся теми данными, которыми были заполнены соответствующие поля в социальной сети.

Ниже, на рисунке 6.5, представлена страница «Кабинет».

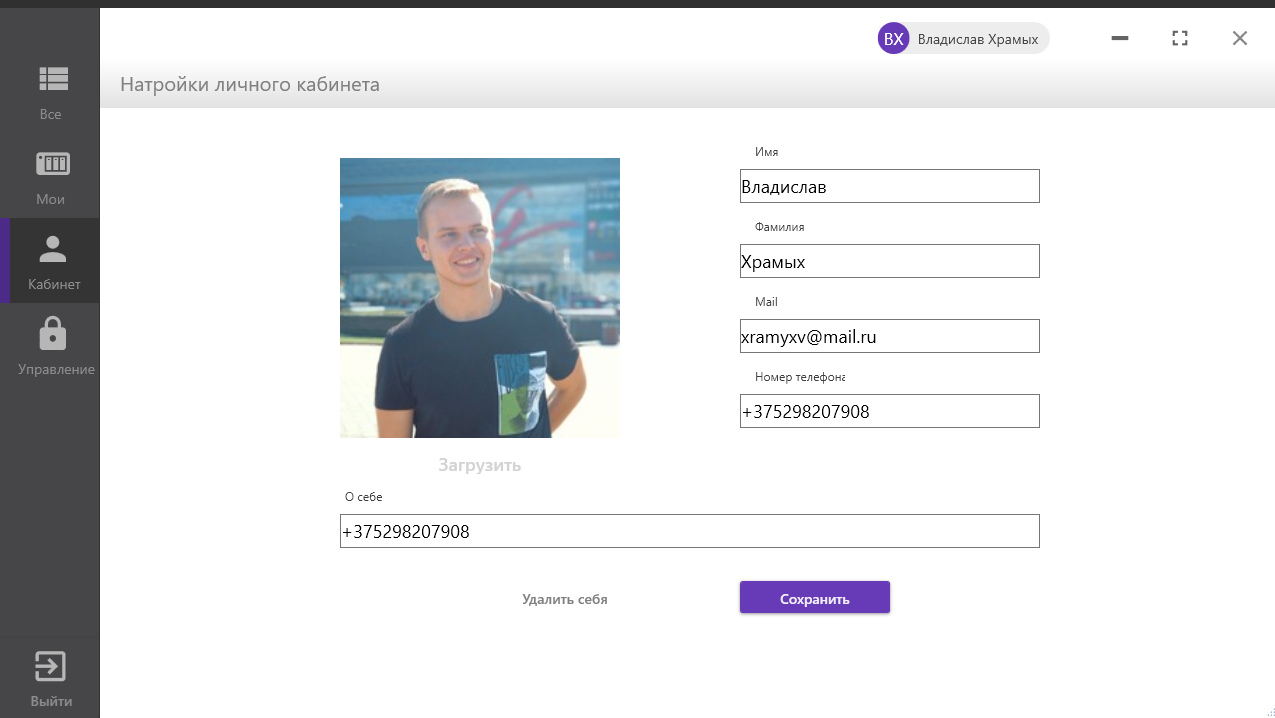


Рисунок 6.5 – «Страница Кабинет»

На рисунке 6.6 показана страница администратора/модератора.

Есть 3 вкладки: Пользователи, Актуальные, Неактуальные.

На этой странице можно совершать действия над пользователями, временными объявлениями, актуальными объявлениями.

На рисунке 6.6 представлен скриншот страницы администратора/модератора.

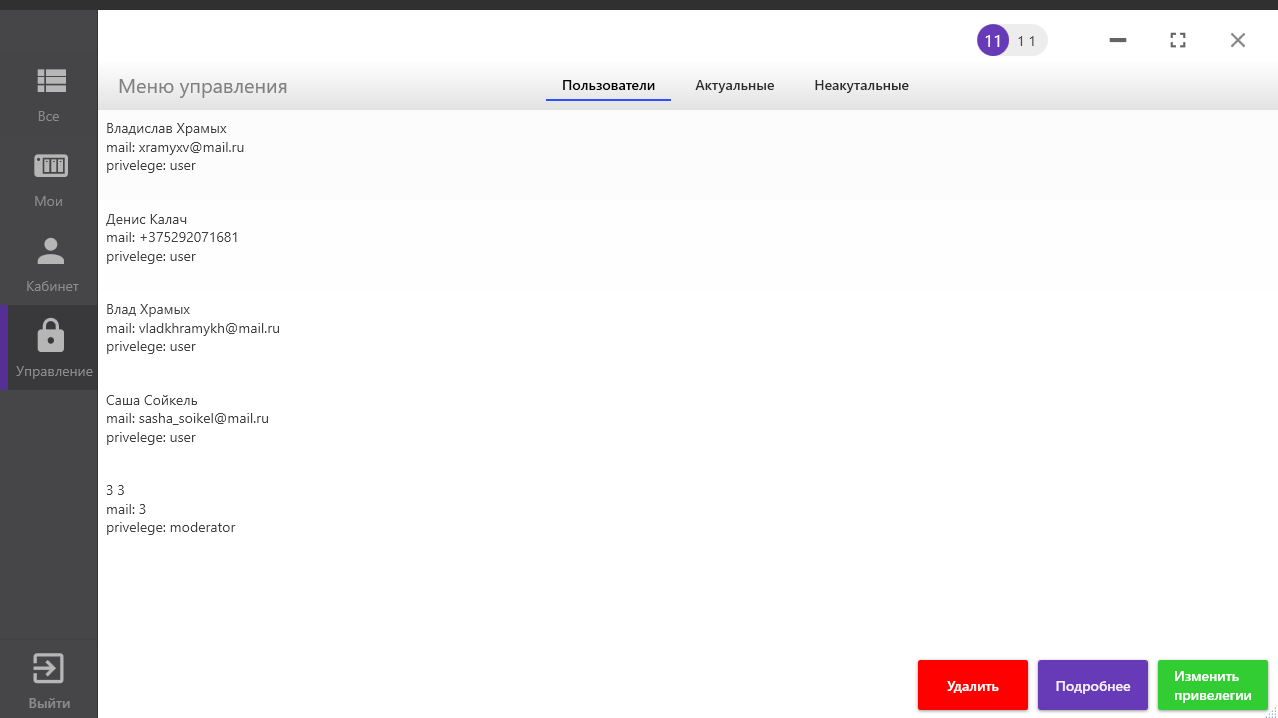


Рисунок 6.6 – «Меню администратора»

# **ЗАКЛЮЧЕНИЕ**

В данном курсовом проекте было разработано программное средство «Площадка объявлений» при помощи языка программирования C#, API-интерфейса Windows Presentation Foundation, технологии Entity Framework и базы данных Microsoft SQL Server. При выполнении курсового проекта использовались принципы и приемы ООП.

Разработанное программное средство предоставляет пользователю следующие функциональные возможности:

* Идентификация пользователей;
* Размещение объявлений разного типа;
* Выборка объявлений по различным критериям;
* Вход в приложение, используя социальные сети;
* Хранение и отображение объявлений;
* Управление своими объявлениями (редактирование, удаление);
* Сравнение объявлений (визуальное).

Модератор может всё то же самое, и плюс ко всему:

* Проверка поданных объявлений;
* Удаление объявлений;
* Просмотр информации о пользователях.

Администратор может всё то же, что и модератор, и плюс ко всему:

* Удаление пользователей любого типа
* Изменение привилегий у любого пользователя.

Проектирование осуществлялось по паттерну проектирования MVVM.

Для удобной работы с данными, был реализован паттерн Repository.

Приложение хорошо работает с базой данных. В ходе тестирования, не было выявлено нарушений в работе приложения с базой данных.

Также осуществлено взаимодействие с социальными сетями VK и Instagram, а именно вход в приложение, использую данные одной из социальных сетей.

Для разработки дизайна использовалась библиотека MaterialDesign, которая имеет огромное количество красивых и удобных элементов управления.

Приложение было успешно протестировано, что означает пригодность этого приложения для своей цели.

Для хранения исходного кода и удобства контроля версий проекта использовался крупнейший веб-сервис для совместной разработки – GitHub. Ссылка на проект: (<https://github.com/VladKhramykh/MarketPlace_WPF>).

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

[1]. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2018. – 175 с.

[2]. Пацей, Н.В. Технология разработки программного обеспечения / Н.В. Пацей. – Минск: БГТУ, 2016. – 129 с. [3]. MSDN сеть разработчиков в Microsoft [Электронный ресурс] / Режим доступа: http://msdn.microsoft.com/library/

[4]. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: https://metanit.com

[5]. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: https://professorweb.ru

[6]. StackOverflow [Электронный ресурс] / Режим доступа: <https://stackoverflow.com>

[7]. Хабр [Электронный ресурс] / Режим доступа: <https://habr.com>

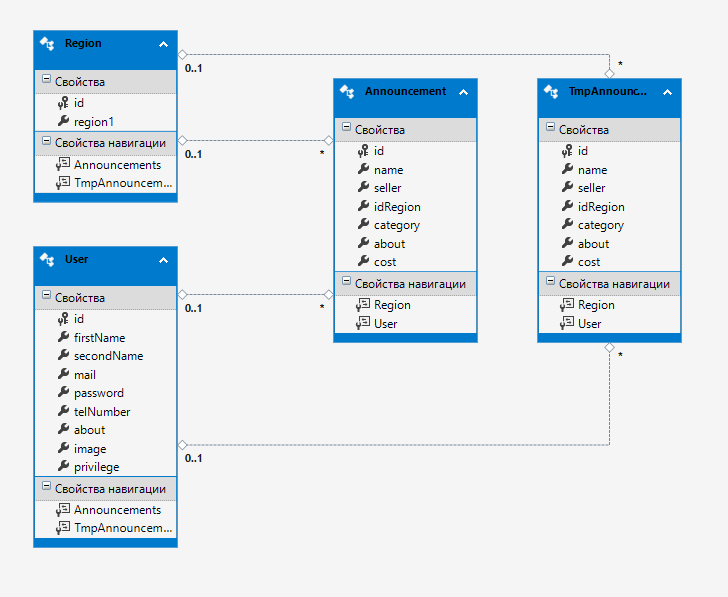
[8]. VkNet – Вконтакте API для C# [Электронные данные] / Режим доступа: <https://vknet.github.io/vk/>

[9]. InstaSharp – API для взаимодействия с Instagram [электронный ресурс] / Режим доступа: <https://github.com/instasharp>

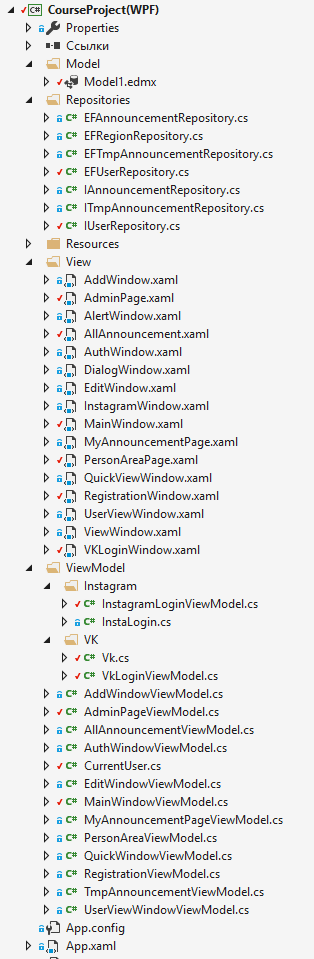
Приложение А



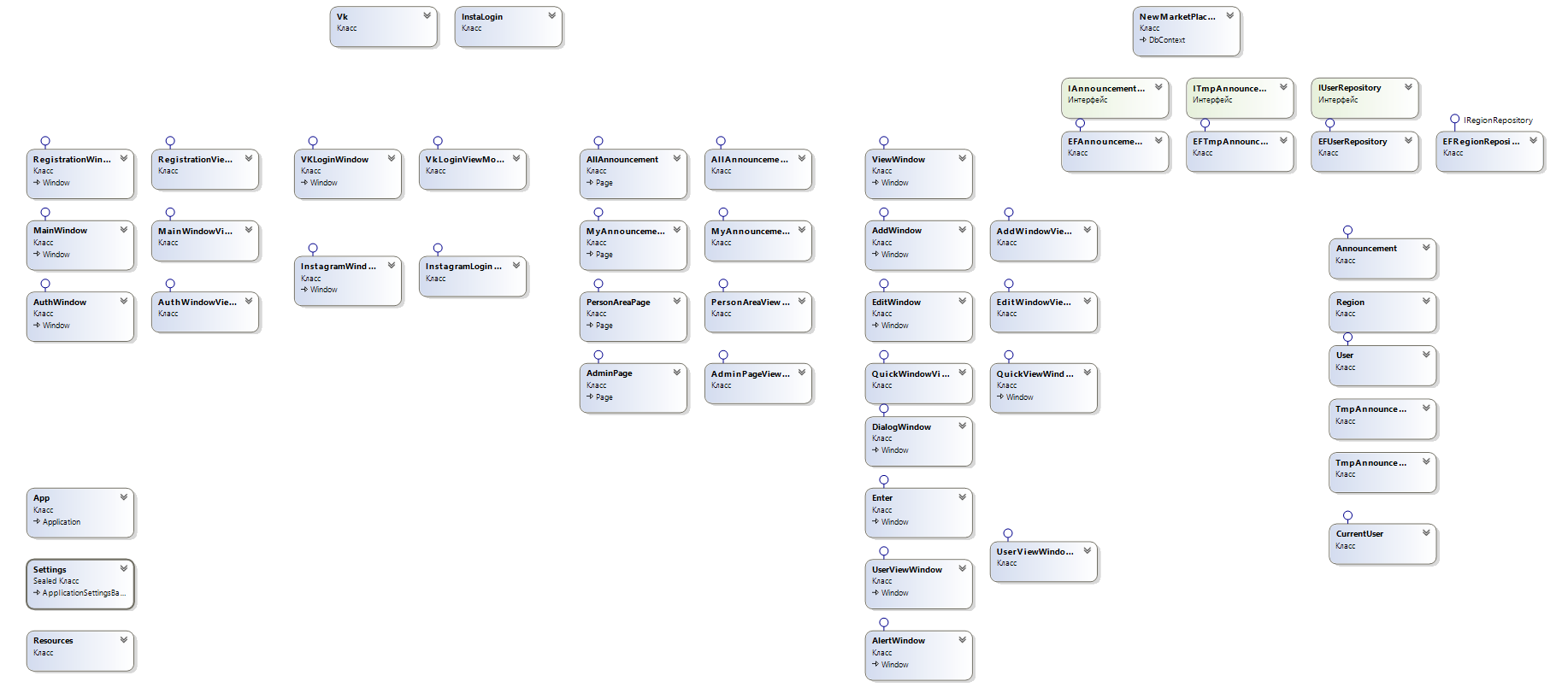
Приложение Б



Приложение В

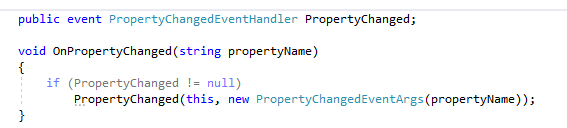


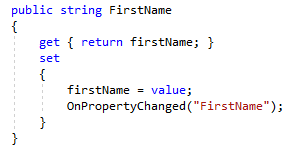
Приложение Г



Приложение Д

Листинг 1 – «INotifyPropertyChanged»

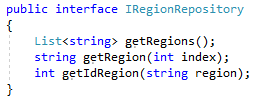


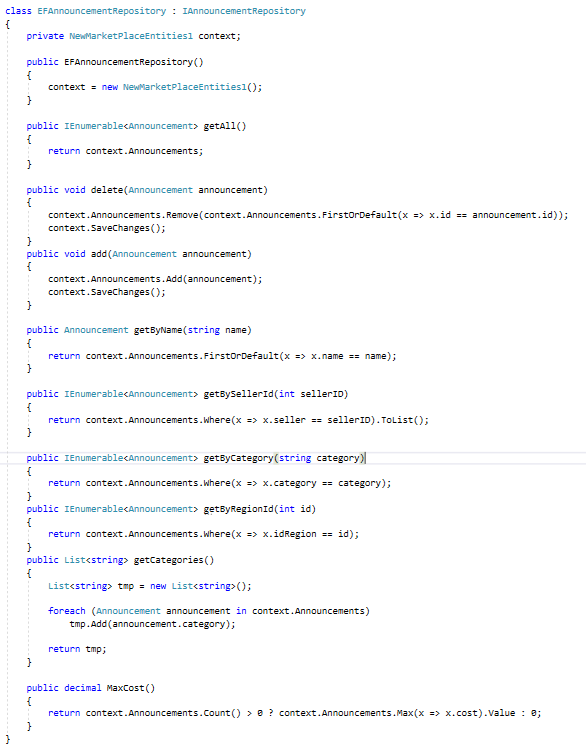


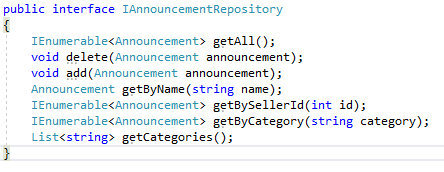
Приложение Е

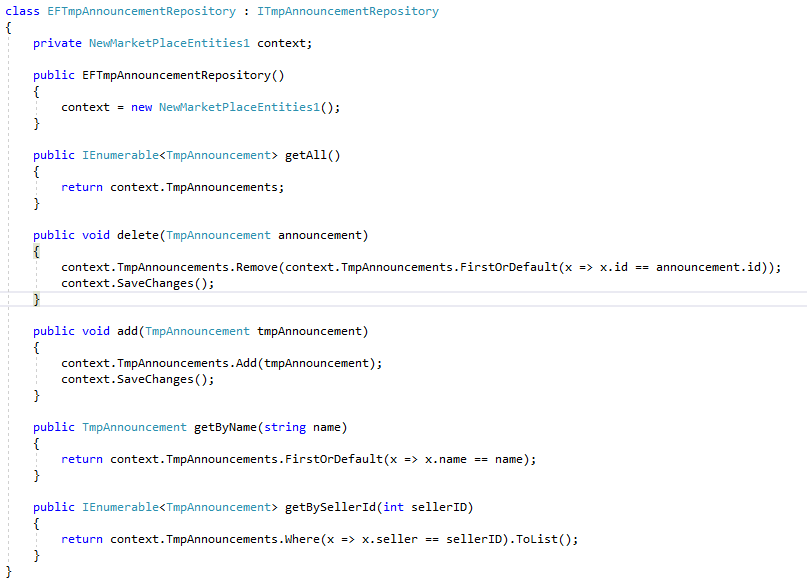
Листинг 2 – «Repositories»

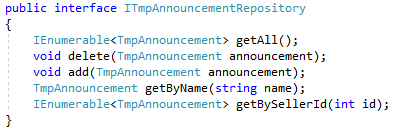


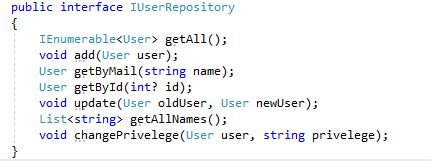


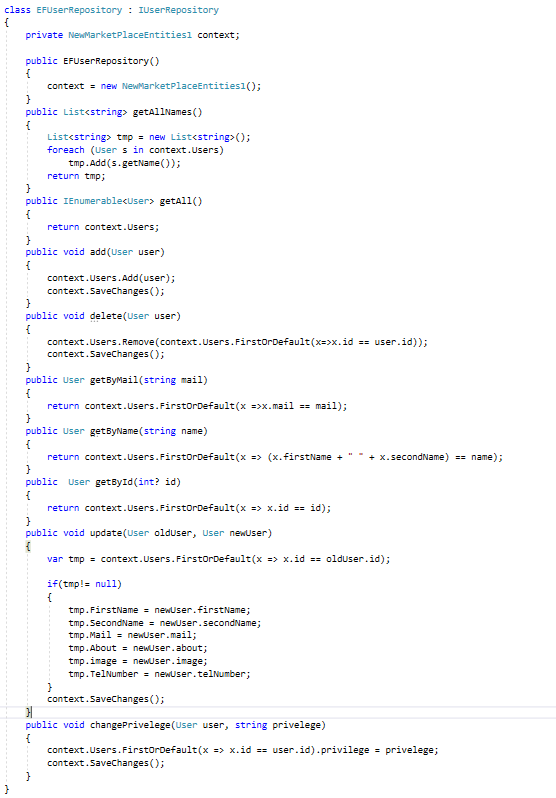






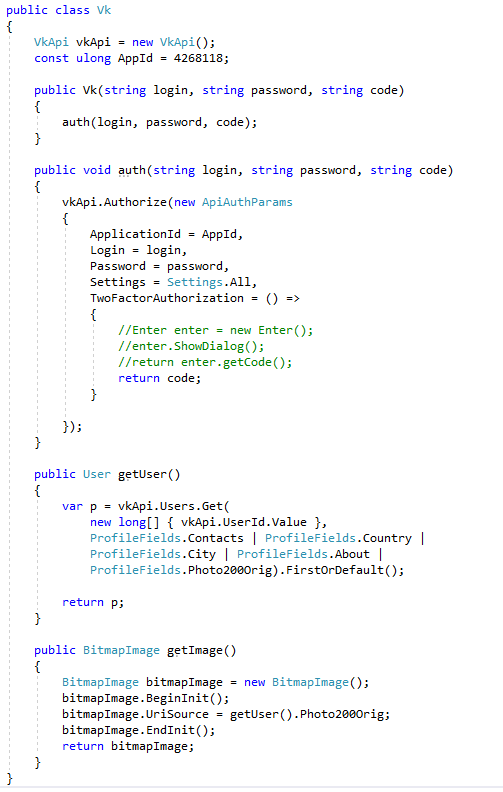






Приложение Ж

Листинг 3 – «Vk API»



Приложение З

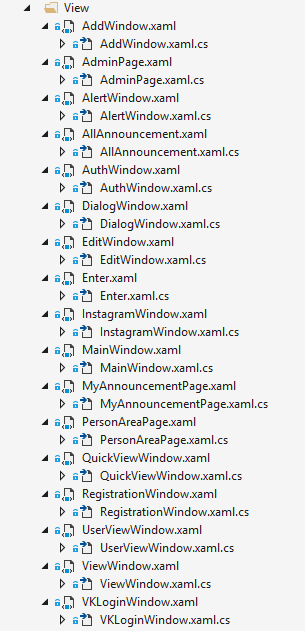
Листинг 4 – «Instagram API»





Приложение И

Листинг 5 – «Структура пакета View»



Приложение К

Листинг 5 – «Структура пакета ViewModel»

