МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий \_

Кафедра Программной инженерии \_

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-изданий)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии и стандарты проектирования»

Тема «Площадка объявлений»

Исполнитель

студент 2 курса группы 6 Храмых Владислав Олегович

Руководитель работы ассистент Радиванович Д.А.

Курсовой проект защищён с оценкой

Председатель Пацей Н.В.

Минск 2019

**Содержание**

[1. Аналитический обзор литературы 4](#_Toc9518112)

[2. Анализ требований к программному средству и разработка функциональных требований 8](#_Toc9518113)

[3. Проектирование программного средства 11](#_Toc9518114)

[4. Создание (реализация) программного средства 21](#_Toc9518115)

[5. Тестирование, проверка работоспособности и анализ полученных результатов 31](#_Toc9518116)

[6. Руководство по установке и использованию 37](#_Toc9518117)

ВВЕДЕНИЕ

Программное средство «Площадка объявлений» – это программное средство, где каждый желающий может вывесить своё объявление, а все посетители сайта — прочитать его.

Так же это приложение может вести сортировку предложений, и в зависимости от выбора критериев, предлагать подходящие варианты.

Приложение имеет 3 типа пользователей: администратор, пользователь, модератор. Они имеют разные полномочия. Администраторы проверяют предложения, которые поступают на проверку перед публикацией. Также они могут удалять предложения, которые уже неактуальны. Пользователи же могут следить за списком предложений, выполнять сортировку по названию, цене, продавцу, региону.

Так же будет реализована авторизация всех пользователей. Это значит, что перед тем, как предложить свой товар, нужно будет зарегистрироваться. Хранение всех пользователей будет осуществлено в базе данных, и при авторизации или регистрации будет непосредственное обращение к ней. Также вся информация о товаре будет хранится в базе данных.

В ходе использования это программы, пользователи смогут получать информацию о интересующих их товарах и продавцах (характеристика товара, номер телефона продавца).

Язык разработки проекта – C#. При выполнении курсового проекта будут использованы принципы и приемы ООП. Также будут использоваться технологии Windows Presentation Foundation (WPF) и MS SQL.

# Аналитический обзор литературы

Для выполнения этого курсового проекта не обошлось без использования дополнительной литературы. Таковой является информация на сайте <https://metanit.com/>. На этом сайте имеется много информации, нужной для разработки этого курсового проекта. Так же использовалась информация с сайта <http://professorweb.ru/>.

Самый большое количество информации было получено с официального сайта <https://docs.microsoft.com/en-us/dotnet/csharp/>. Статьи с <https://habr.com/ru/> были также очень полезны.

В качестве аналога я выбрал сайт [https://catalog.onliner.by](https://catalog.onliner.by/), т.к. он максимально приближен к моей задумке. Краткий обзор этого сервиса представлен ниже.

Для посетителей Onliner.by доступен  [каталог](http://catalog.onliner.by/) товаров. Вот так выглядит страница с товарами (рисунок 1.1).

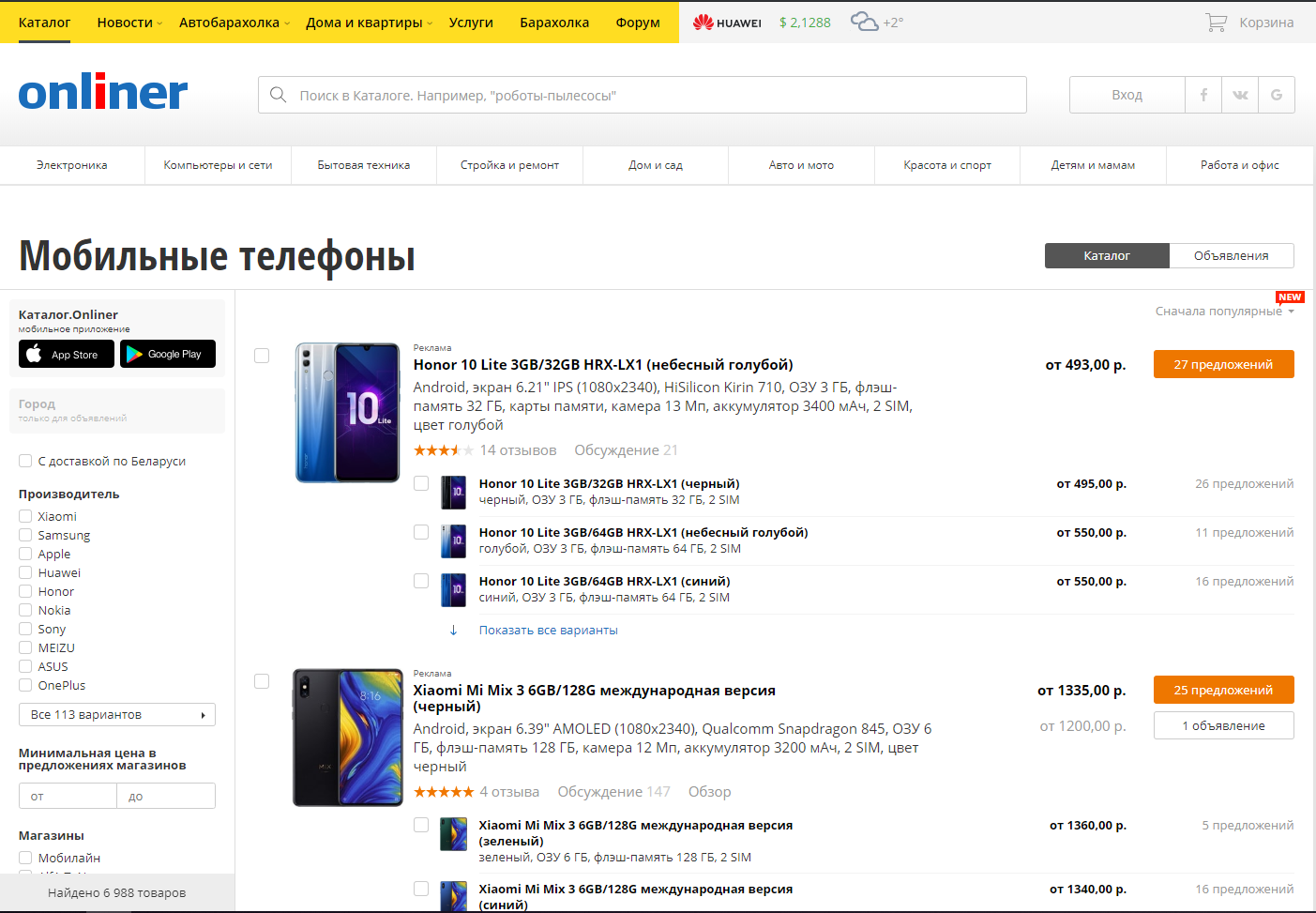


Рисунок 1.1 – раздел «Мобильные телефоны»

Все параметры для подбора товаров находятся в левой части экрана. Например, если речь идет о [телефонах](http://catalog.onliner.by/mobile), здесь с помощью checkbox можно выбрать среди наиболее популярных брендов, размеров и разрешений экрана, объемов оперативной и встроенной памяти и т. д. (рисунок 1.2).

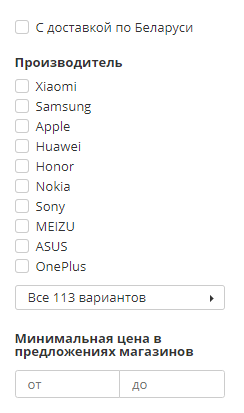


Рисунок 1.2 – параметры в каталоге

Если же вам нужно указать какой-то редкий параметр, его можно либо задать вручную, либо выбрать из списка. Дополнительные условия для поиска и всплывающие подсказки помогают в этом.

Так же здесь есть такая фишка – динамическое обновление контента по мере выбора тех или иных параметров. Если раньше для того, чтобы вывести на экран страницу товаров с заданными параметрами, надо было нажать кнопку «Подобрать», то теперь изменения происходят на лету. Пока вы маркируете checkbox, работаете со списками или вводите показатели вручную, данные на странице моментально обновляются в соответствии с заданными условиями (рисунок 1.3).

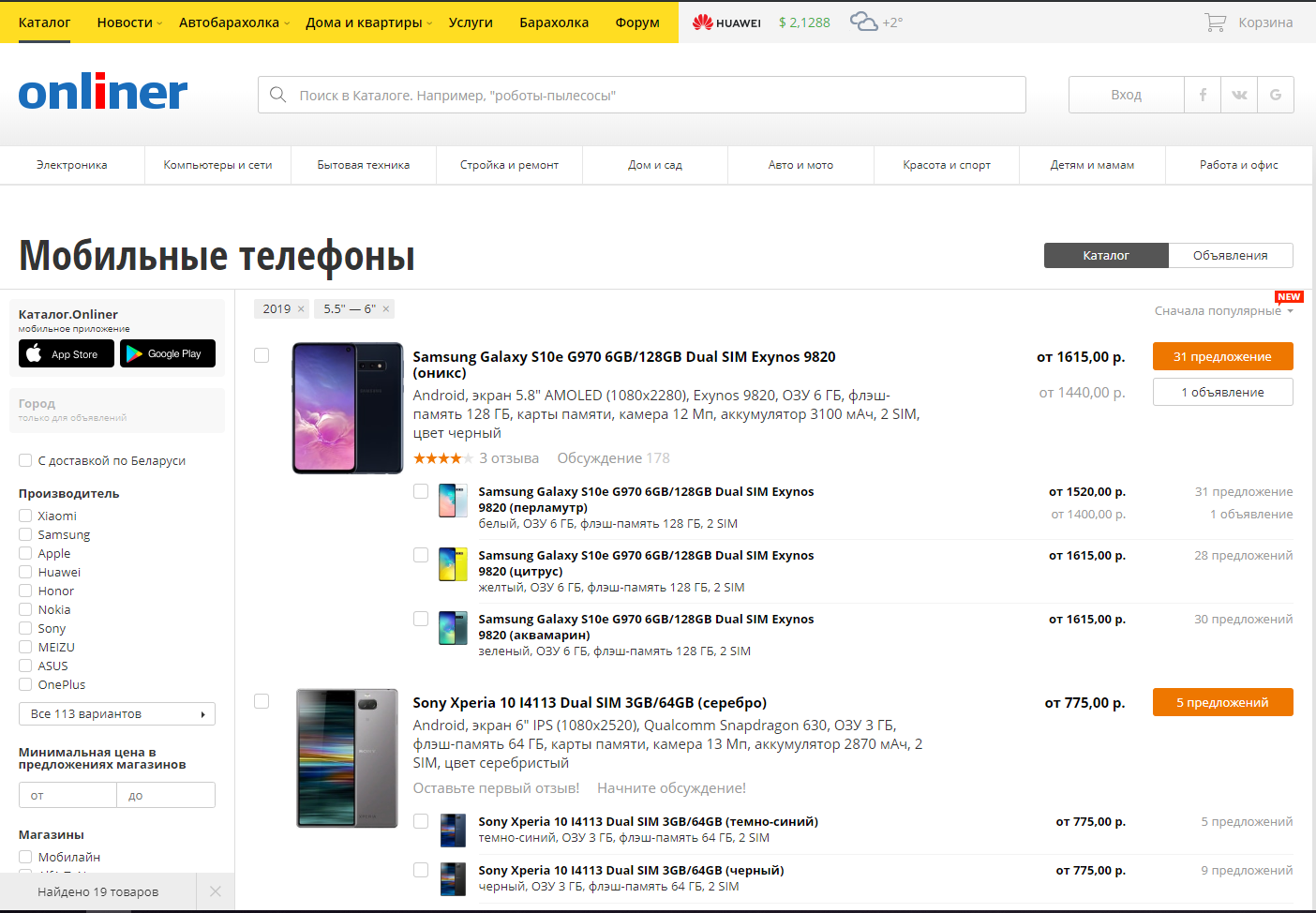
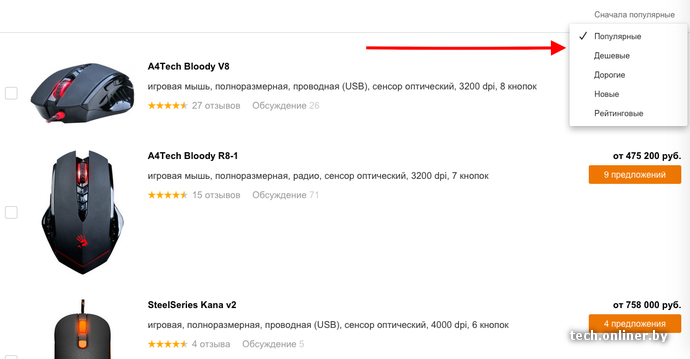


Рисунок 1.3 – Выбор параметров

Выбранные параметры выводятся вверху страницы с товарами. Чтобы убрать ненужные параметры, не обязательно искать их в левой панели и снимать «галочки». Достаточно «закрыть» их прямо наверху страницы.

Имеется сортировка по популярности, новизне, стоимости и рейтингу. Соответствующая опция расположена в правом верхнем углу страницы. По умолчанию задействован фильтр «Сначала популярные».

[[](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

Я решил сделать проект на эту тему, так как не видел ещё desktop-версии подобного приложения для Windows.

Этот сайт является образцом для моего проекта (<https://catalog.onliner.by/>).

# Анализ требований к программному средству и разработка функциональных требований

Задачей курсового проекта является разработка приложения, предназначенного осуществления торговли, а также ряда дополнительных операций: добавления товаров/услуг, редактирования, поиска, удаления информации. Информацию о товарах и услугах вводится непосредственно с самого приложения и проверяется администратором/модератором.

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:

* 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

Наглядно это видно на UML-схеме, изображённой на рисунке 2.1



Рисунок 2.1 – «UML-диаграмма возможностей»

# Проектирование программного средства

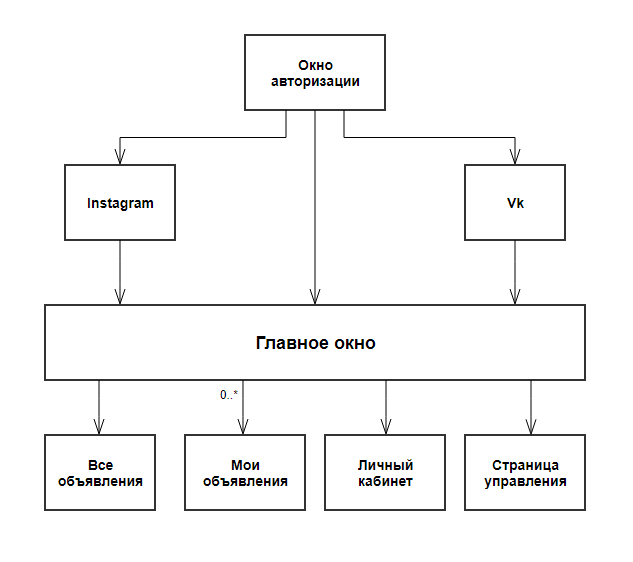


Рисунок 3.1 – «Авторизация и главное окно»

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. Так же 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.

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 идём в страничном режиме.

Рассмотрим каждую страницу в отдельности.

Страница «Все объявления» (рисунок 3.2).

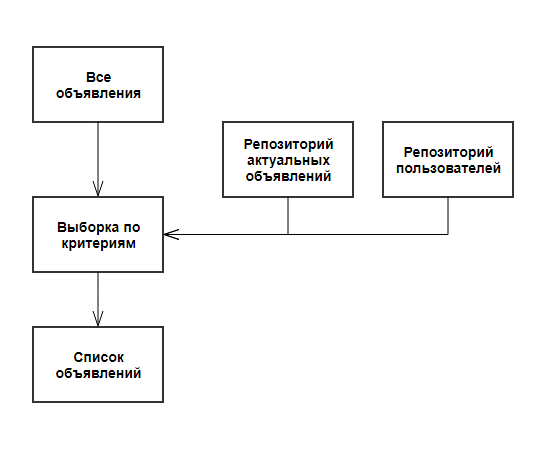


Рисунок 3.2 – «Все объявления»

Эта страница содержит список всех актуальных объявлений.

Так же 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.

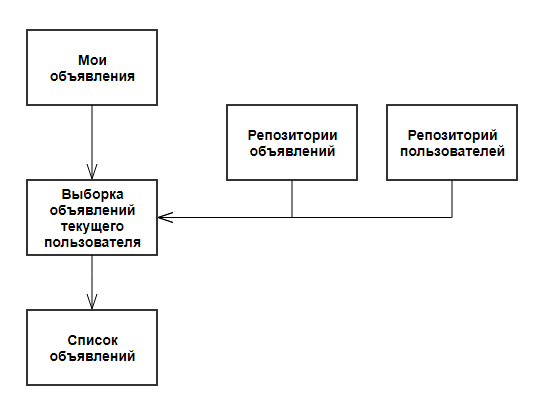


Рисунок 3.3 – «Мои объявления»

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. Так же 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.

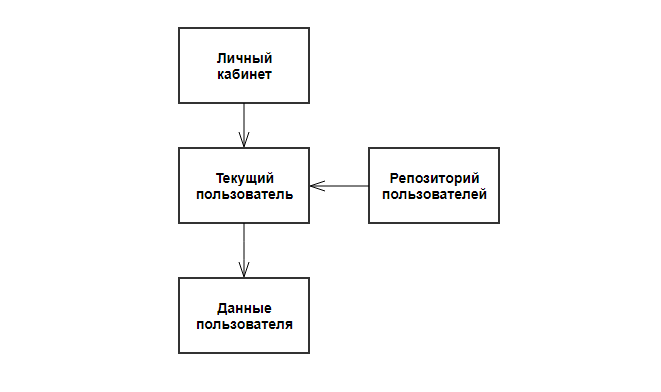


Рисунок 3.4 – «Личный кабинет»

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.

Так же 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.

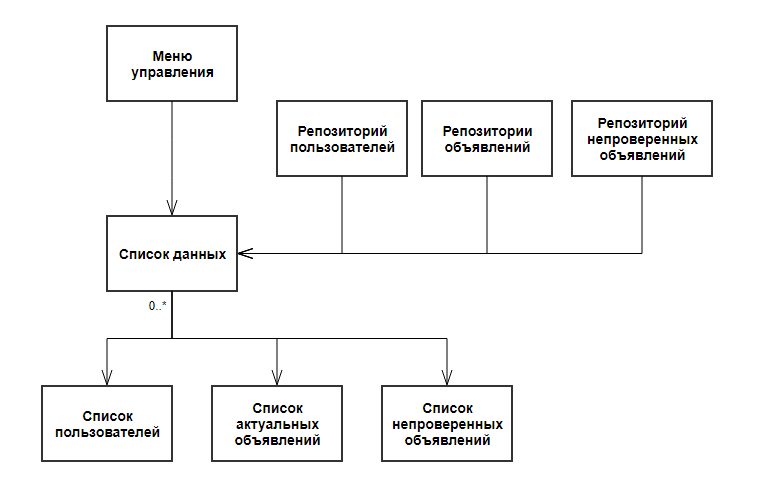


Рисунок 3.5 – «Меню управления»

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 такие 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 на рисунке 3.7

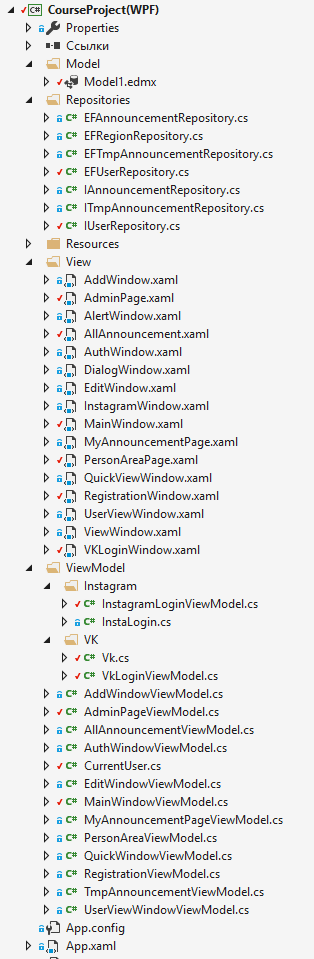


Рисунок 3.7 – «Структура проекта»

В таблице 3.8 описана структура проекта по пакетам классов.

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

Таблица 3.8 – 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 | Здесь находятся файлы, в которых описаны стили для элементов управления |
| View | Содержит все представления, которые позволяют пользователю работать с приложением. Они описывают графическую составляющую приложения, через которую связываются все манипуляции с данными |
| 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 |

С помощью этой таблицы, понять общую структуру проекта не составляет никакого труда.

Диаграмма в UML — это графическое представление набора элементов, изображаемое чаще всего в виде связанного графа с вершинами (сущностями) и ребрами (отношениями). Диаграммы рисуют для визуализации. Основная цель диаграмм — визуализация разрабатываемой системы с разных точек зрения.

Структуры данных, а также их зависимости друг от друга представлены на рисунке 3.9

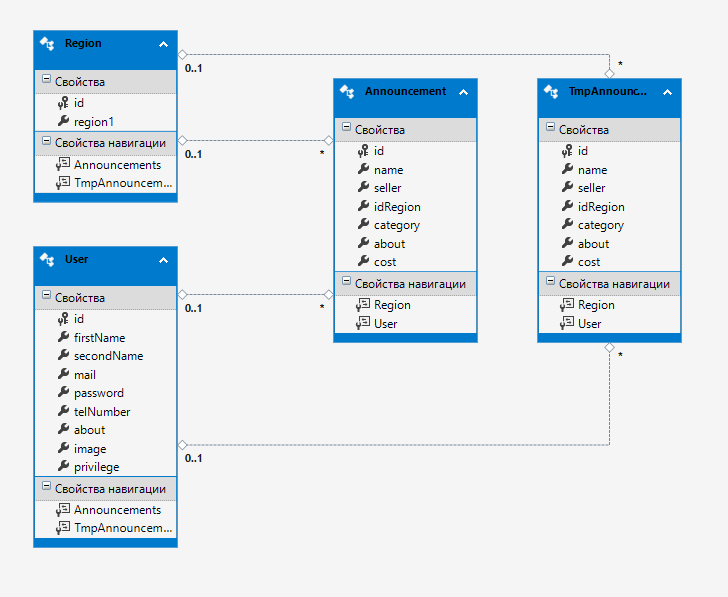


Рисунок 3.9 – «Типы данных»

Как показано на рисунке 3.9, в вершине иерархии стоят классы Announcement и TmpAnnouncement, который характеризует актуальное и непроверенное объявление соответственно. Класс Region характеризует все регионы, в которых 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.

На следующем рисунке показана диаграмма классов (рисунок 3.10)

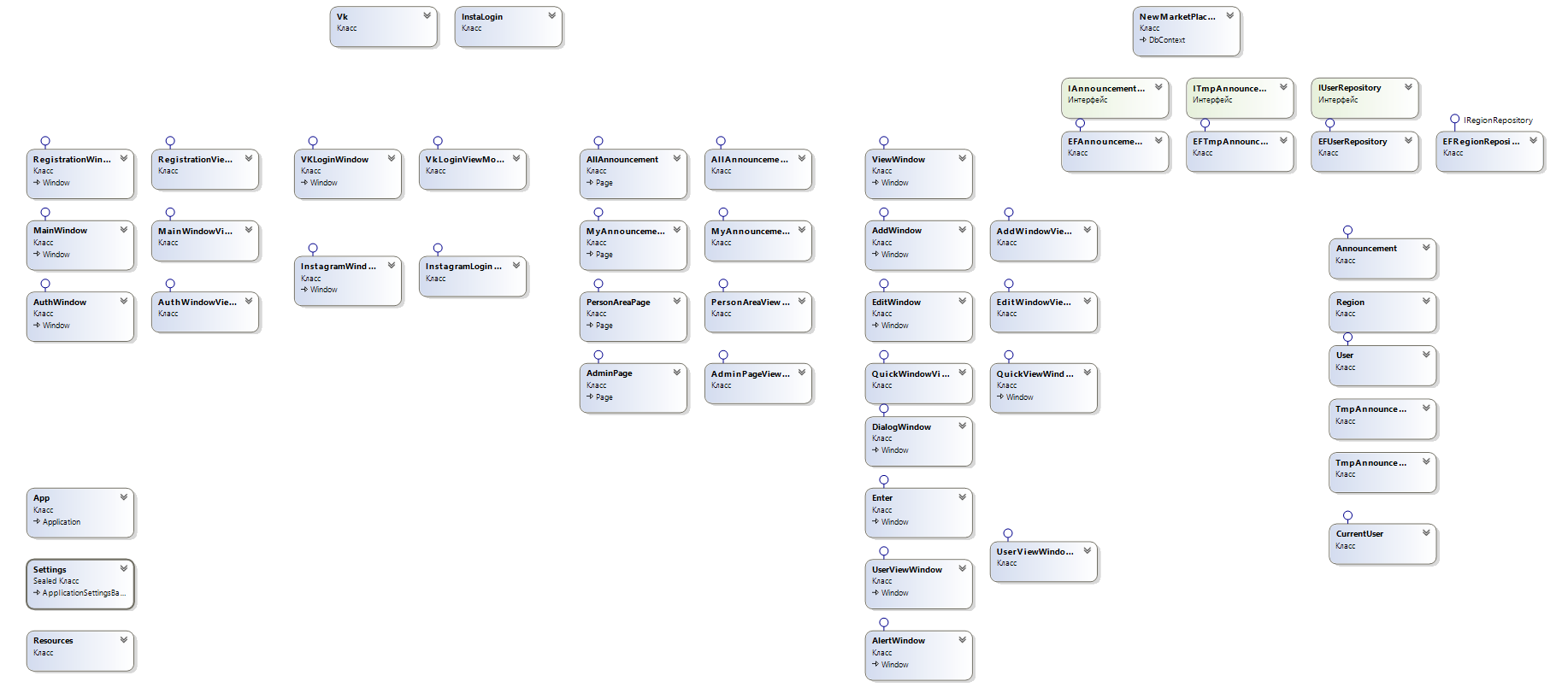


Рисунок 3.10 – Диаграмма классов

# Создание (реализация) программного средства

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

Почти весь проект (кроме файлов ресурсов и т.д.) разбит на 3 раздела как показано на рисунке 4.1

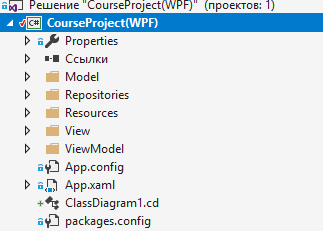


Рисунок 4.1 – «Структура пакетов проекта»

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.

Центральной концепцией Entity Framework является понятие сущности или entity. Сущность представляет набор данных, ассоциированных с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их наборами.

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() как показано ниже:

|  |
| --- |
| public event PropertyChangedEventHandler PropertyChanged;  void OnPropertyChanged(string propertyName)  {  if (PropertyChanged != null)  PropertyChanged(this, new PropertyChangedEventArgs(propertyName));  } |

Листинг 4.2 – «OnPropertyChanged»

Для того, чтобы начать отслеживание свойства, необходимо при его установке вызывать метод OnPropertyChanged(“Имя свойства”) как показано ниже:

|  |
| --- |
| public string FirstName  {  get { return firstName; }  set  {  firstName = value;  OnPropertyChanged("FirstName");  }  } |

Листинг 4.3 – «Применение OnPropertyChanged»

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 4.4 показана структура.

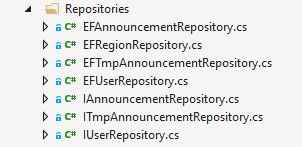


Рисунок 4.4 – «Паттерн Repository»

Интерфейсы IUserRepository, IAnnoouncementRepository, ITmpAnnouncementRepository, IRegionRepository имплицируются в соответствующих классах и указывают: какие методы (функционал) должен иметь этот репозиторий. Класс репозитория RegionRepository показан ниже:

|  |
| --- |
| public class EFRegionRepository : IRegionRepository  {  private NewMarketPlaceEntities1 context;  public EFRegionRepository()  {  context = new NewMarketPlaceEntities1();  }  public List<string> getRegions()  {  List<string> tmp = new List<string>();  foreach (Region region in context.Regions)  tmp.Add(region.region1);  return tmp.Distinct().ToList();  }  public string getRegion(int index)  {  return context.Regions.FirstOrDefault(x=> x.id == index).region1;  }  public int getIdRegion(string region)  {  return сontext.Regions.FirstOrDefault(x=>x.region1.Equals(region)).id;  }  } |

Листинг 4.5 – «Region репозиторий»

Интерфейс IRegionRepository показан ниже:

|  |
| --- |
| public interface IRegionRepository  {  List<string> getRegions();  string getRegion(int index);  int getIdRegion(string region);  } |

Листинг 4.6 – «Интерфейс IRegionRepository»

По такому же принципу и строятся остальные интерфейсы и репозитории.

На рисунке 4.7 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 4.7 – «Структура пакета Resources»

В проекте используется паттерн MVVM, а это значит, что должны присутствовать ViewModels, которые связывают View и Model. Во ViewModel также должна информация обрабатываться и выводится/записываться в нужном виде. Ниже представлена структура пакета ViewModel (рисунок 4.8)

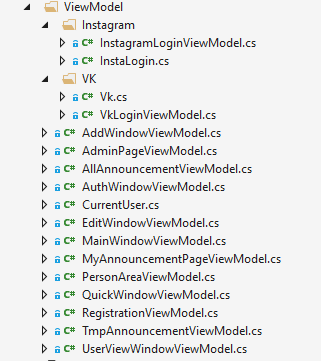


Рисунок 4.8 – «Структура пакета ViewModel»

На следующем рисунке 4.9 показана структура пакета View, в котором хранятся файлы представлений на языке разметки xaml, а также соответствующие им файлы на языке C#.

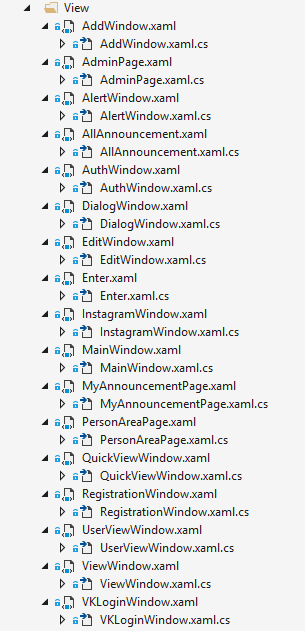


Рисунок 4.9 – «Структура пакета View»

В этом пакете находятся все «окна» и «страницы», которые мы видим, используя приложение. Назначение каждого файла описано в таблице ниже (таблица 4.10)

Таблица 4.10 – «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 должен быть не виден пользователю. Код, представленный ниже, показывает процесс авторизации:

|  |
| --- |
| VkApi vkApi = new VkApi();  const ulong AppId = \*\*\*\*\*\*\*;  public Vk(string login, string password, string code)  {  auth(login, password, code);  }  public void auth(string login, string password, string code)  {  vkApi.Authorize(new ApiAuthParams  {  ApplicationId = AppId,  Login = login,  Password = password,  Settings = Settings.All,  TwoFactorAuthorization = () =>  {  Enter enter = new Enter();  enter.ShowDialog();  return enter.getCode();  //return code;  }  });  } |

Листинг 4.11 – «VkNet API»

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.

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, представлен ниже:

|  |
| --- |
| public class InstaLogin  {  string username;  string password;  static UserSessionData user;  static IInstaApi api;  User tmp;  public InstaLogin(string username, string password)  {  this.username = username;  this.password = password;  auth();  }  void auth()  {  user = new UserSessionData();  user.UserName = username;  user.Password = password;  }  async Task Login()  {  api = InstaApiBuilder.CreateBuilder()  .SetUser(user)  .UseLogger(new DebugLogger(LogLevel.Exceptions))  .Build();  var loginRequest = await api.LoginAsync();  if (loginRequest.Succeeded)  {  IResult<InstaCurrentUser> result = await api.GetCurrentUserAsync();  var names = result.Value.FullName.Split(' ');  string firstname = names[0];  string secondName = secondName = names[1];  string mail = result.Value.Email;  string telNumber = result.Value.PhoneNumber;  byte[] image;  using (WebClient client = new WebClient())  {  client.DownloadFile(new Uri(result.Value.HdProfilePicture.Uri), User.filename);  }  using (System.IO.FileStream fs = new System.IO.FileStream(User.filename, System.IO.FileMode.OpenOrCreate))  {  image = new byte[fs.Length];  fs.Read(image, 0, image.Length);  }  string about = result.Value.FullName + "\n" + result.Value.Gender + "\n" + result.Value.UserName + "\n" + result.Value.CountryCode;  tmp = new User(firstname, secondName, mail, telNumber, about, image);  }  } |

Листинг 4.12 – «InstaSharp API»

На рисунке 4.13 изображена диаграмма таблиц со связями в базе данных.

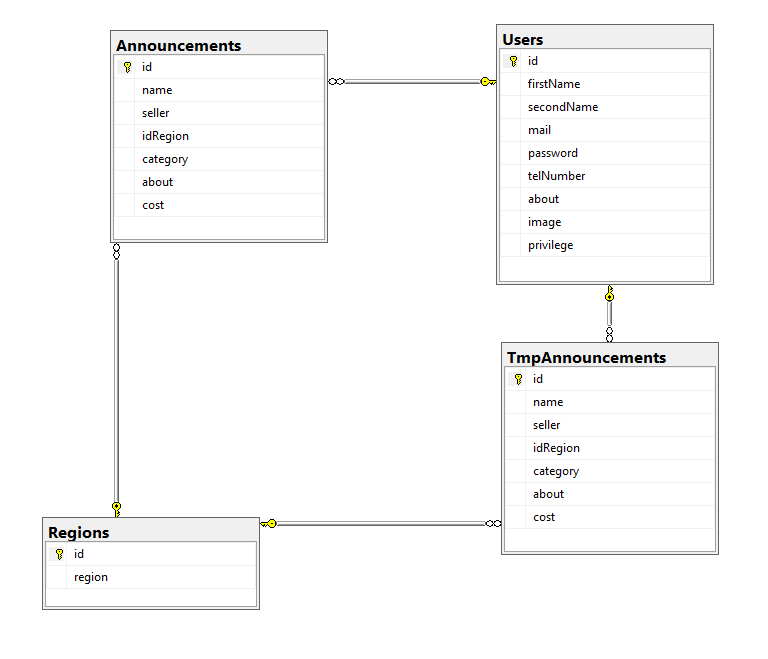


Рисунок 4.13 – «Диаграмма таблиц»

Описание таблицы Announcements (таблица 4.14)

Таблица 4.14 – «описание Announcements»

|  |  |
| --- | --- |
| id | Идентификатор |
| name | Название объявления/товара |
| seller | Id продавца |
| idRegion | Id региона |
| category | Категория |
| about | Описание объявления/товара |
| cost | Цена |

Описание таблицы TmpAnnouncements (таблица 4.15)

Таблица 4.15 – «описание TmpAnnouncements»

|  |  |
| --- | --- |
| id | Идентификатор |
| name | Название объявления/товара |
| seller | Id продавца |
| idRegion | Id региона |
| category | Категория |
| about | Описание объявления/товара |
| cost | Цена |

Описание таблицы (таблица 4.16)

Таблица 4.16 – «описание Users»

|  |  |
| --- | --- |
| Id | Идентификатор |
| firstName | Имя |
| secondName | Фамилия |
| mail | Email адрес |
| password | Хеш пароля |
| telNumber | Номер телефона |
| about | О себе |
| image | Поток байтов (изображение) |
| privilege | Привилегии |

Описание таблицы Regions (таблица 4.17)

Таблица 4.17 – «описание Regions»

|  |  |
| --- | --- |
| id | Идентификатор |
| region | Название объявления/товара |

# 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

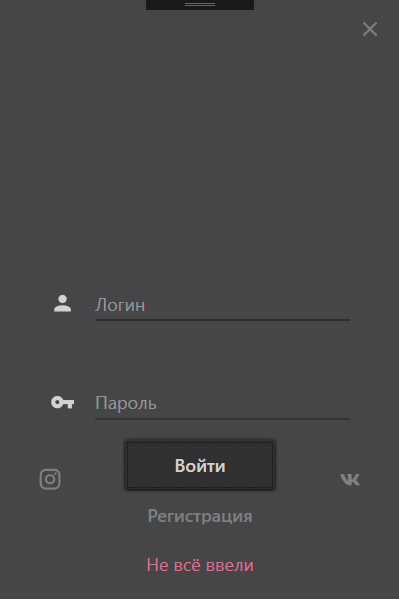


Рисунок 5.1 – «Ошибка корректности ввода»

При неверных данных – рисунок 5.2

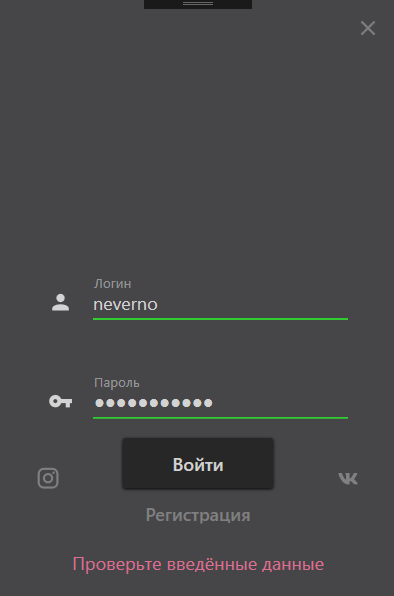


Рисунок 5.2 – «Сценарий, с неверными данными»

Так же 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)

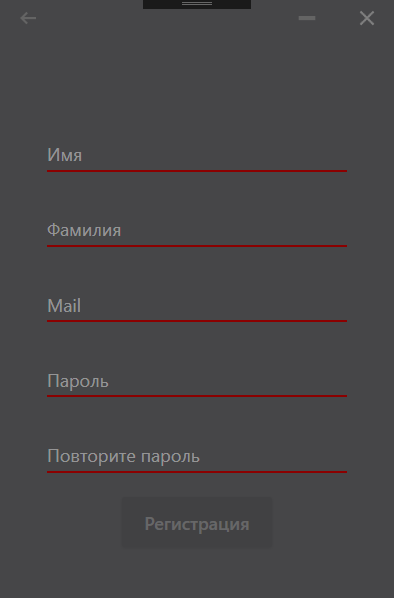


Рисунок 5.3 – «Отсутствие данных»

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»

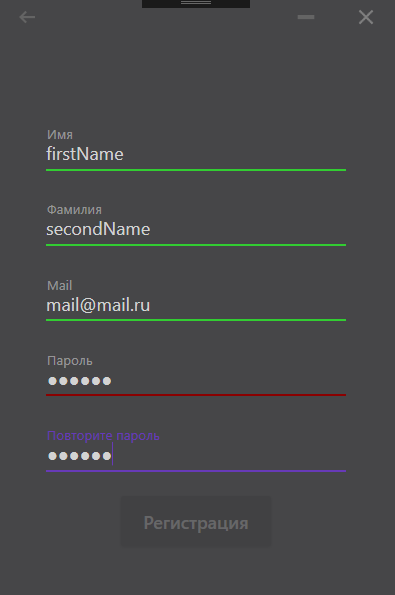


Рисунок 5.4 – «сценарий, при несовпадении паролей»

Так же 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.

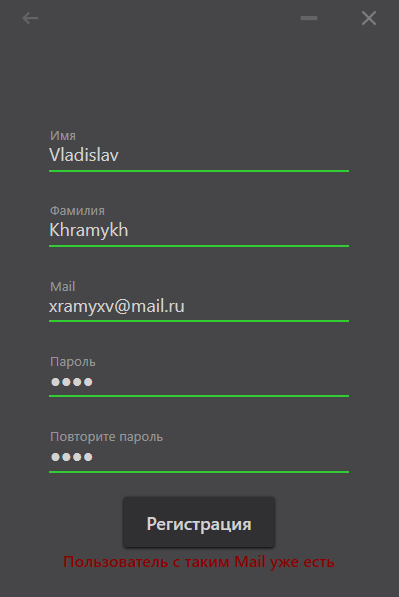


Рисунок 5.5 – «Регистрация пользователя с занятым логином»

Так же может возникать такая ситуация, когда пользователь, который не имеет достаточные привилегии, хочет зайти в меню управления. Ему не даст туда зайти ограничение по привилегиям. Пример показан на рисунке 5.6

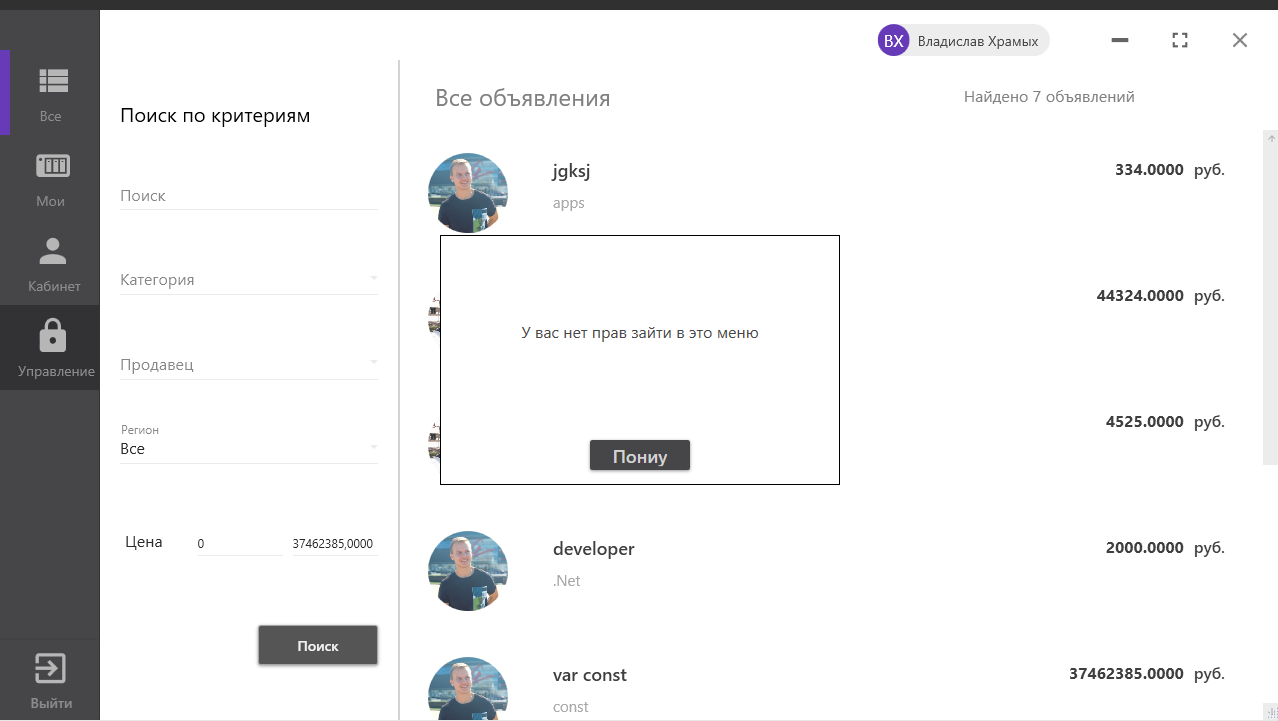


Рисунок 5.6 – «ограничение привилегий»

# 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. Так же 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.

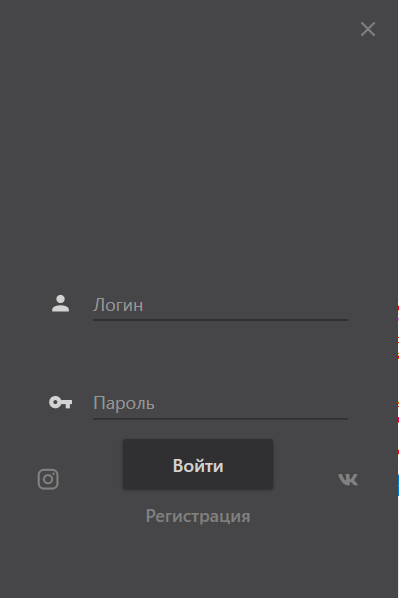


Рисунок 6.1 – «Окно авторизации»

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».

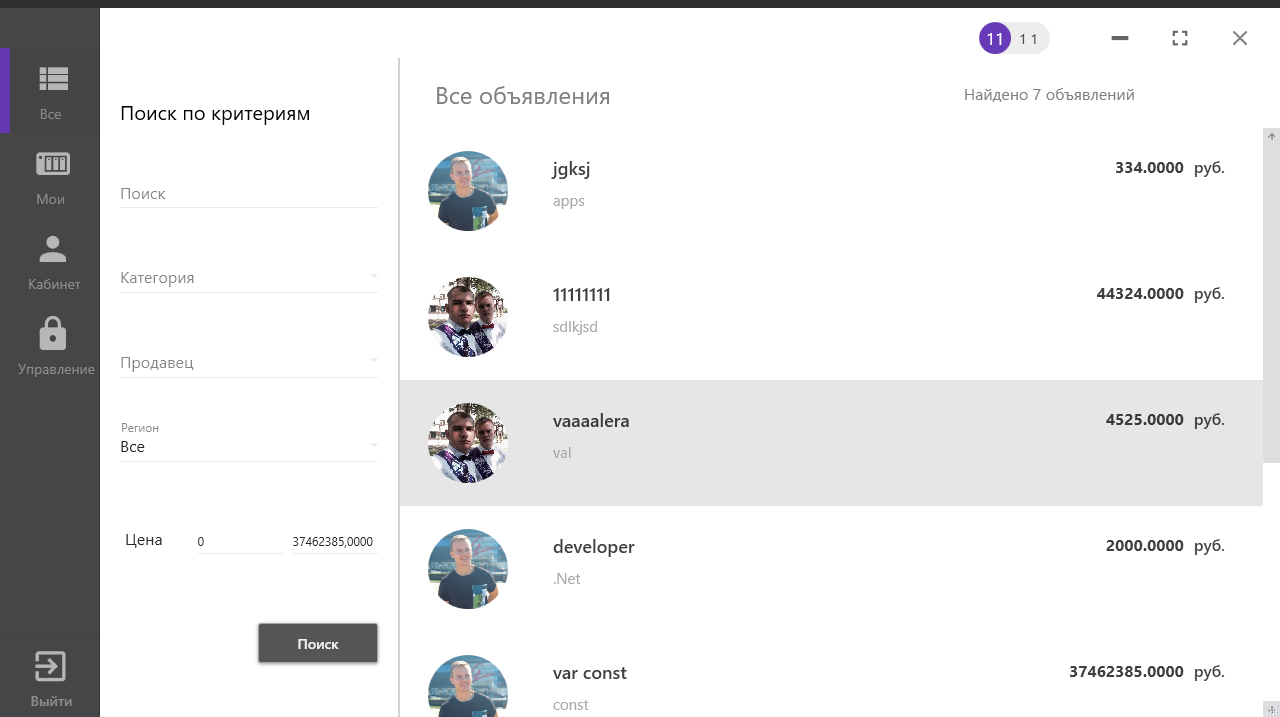


Рисунок 6.2 – «Главное окно»

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».

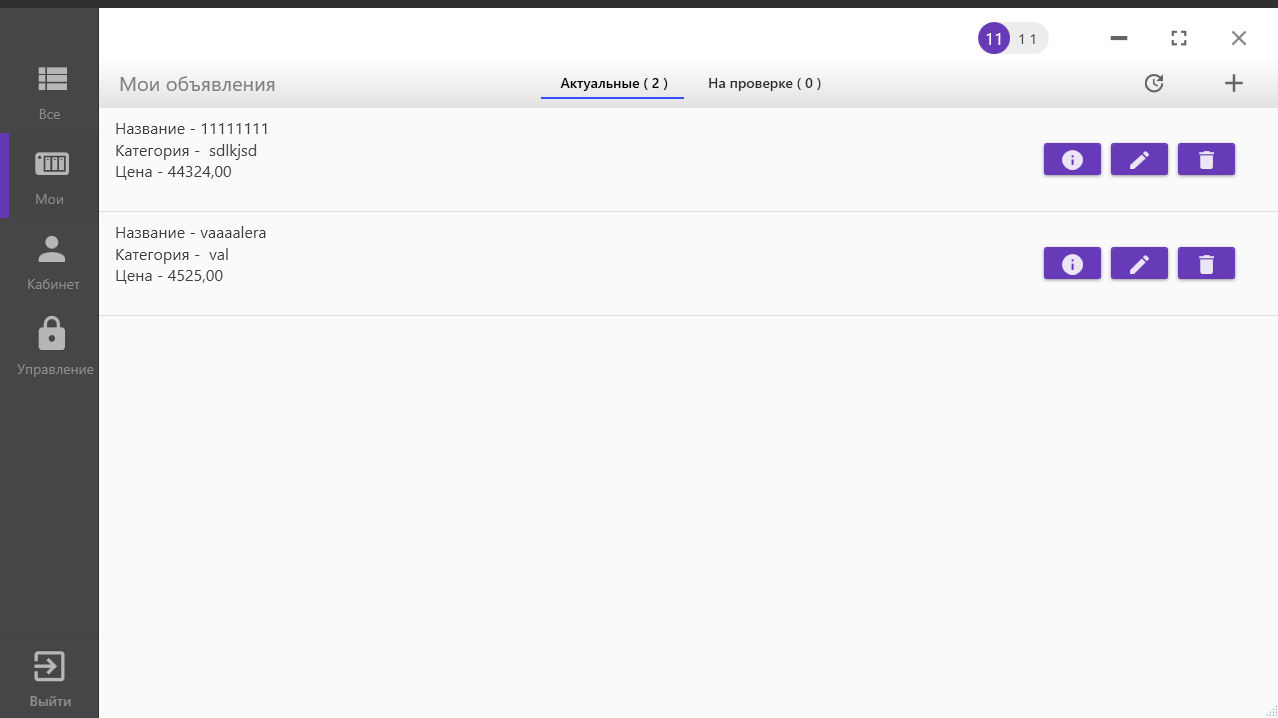


Рисунок 6.3 – «Страница Мои»

Ниже показана страница добавления объявления (рисунок 6.4)

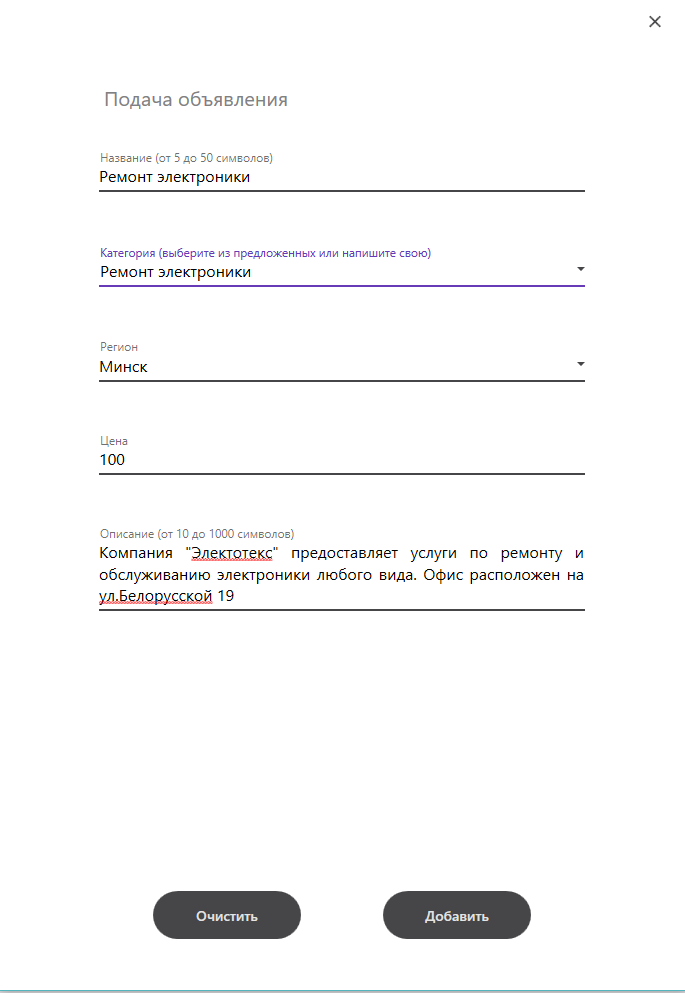


Рисунок 6.4 – «Окно добавления объявления»

Так же 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».

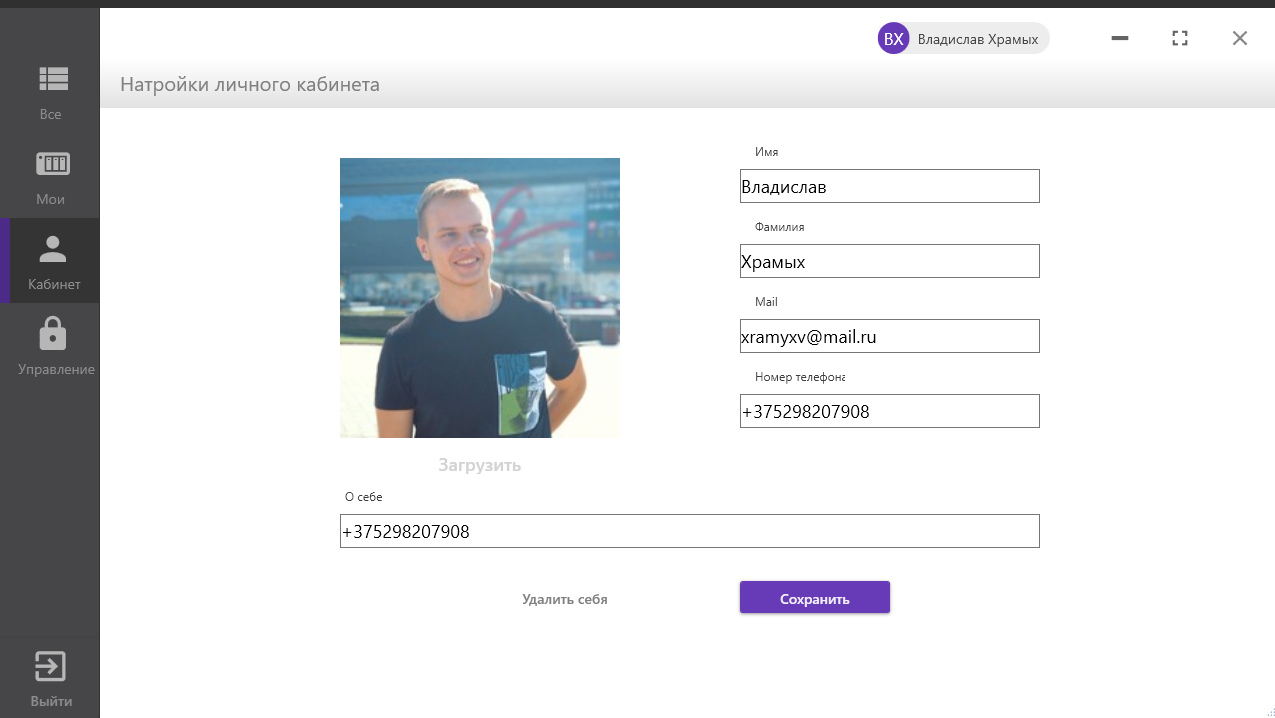


Рисунок 6.5 – «Страница Кабинет»

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.

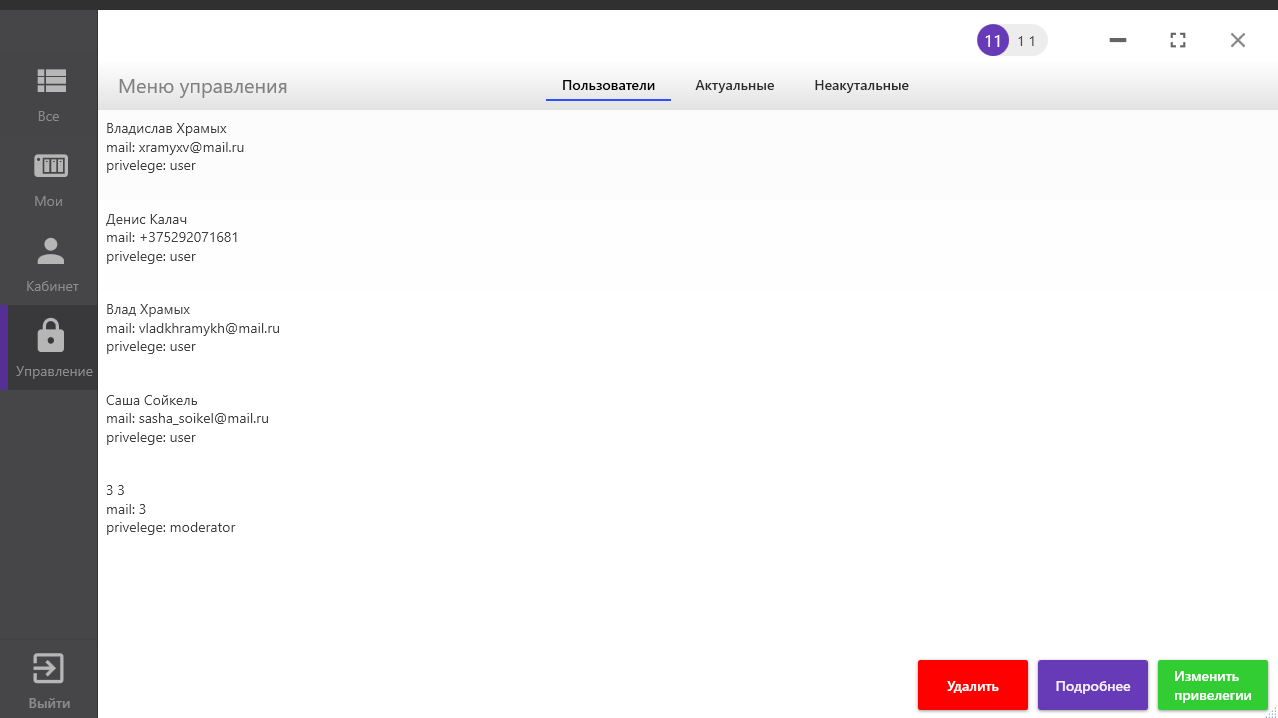


Рисунок 6.6 – «Меню администратора»

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.

Проектирование осуществлялось по паттерну проектирования MVVM.

Для удобной работы с данными, был реализован паттерн Repository.

Так же 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.

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 использованных 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>