Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліні “ Об’єктно - орієнтоване програмування”

«Купи-продай»

Керівник , викл. каф ПІ Ляпота В.М.

Студент гр. ПІ-15-3 Шанiн А.О.

Комісія: проф.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дудар З.В.

проф.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бондарєв В.М.

Харків 2016

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ**

Кафедра: ***Програмної інженерії***

Дисципліна: ***Об’єктно-орієнтоване програмування***

Спеціальність: ***Програмна інженерія***

Курс 1 .Група *ПІ-15-3*. Семестр *2* .

***ЗАВДАННЯ***

***на курсовий проект студента***

*Шанiна Андрiя Олександровича*

1. Тема проекту: Купи-продай.
2. Термін здачі студентом закінченого проекту: *“31” - травня - 2016 р*.
3. Вихідні дані до проекту: *специфікація програми, методичні вказівки до виконання курсової роботи*.
4. Зміст розрахунково-пояснювальної записки: *вступ, програмна специфікація, проектна специфікація,* інструкція користувача (пояснює, як встановлювати програму та користуватися нею), висновки.
5. Перелік графічного матеріалу: екранні форми, ілюстрація взаємодії класів.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |
| --- | --- | --- |
| *№* | *Назва етапу* | *Термін виконання* |
| 1 | Видача теми, узгодження і затвердження теми | 1-03-2016 р. |
| 2 | Формулювання вимог до програми | 2-03-2016 – 12-03-2016 р. |
| 3 | Розробка підсистеми додавання, видалення оголошення | 13-03-2016 – 24-03-2016 р. |
| 4 | Розробка пiдсистеми реєстрацiї, авторизацiї | 25-03-2016 – 1-04-2016 р. |
| 5 | Розробка системи статистики, змiни оголошення, системи рейтинга. | 2-04-2016 – 9-04-2016 р. |
| 6 | Розробка адмiнiстративної системи та системи тегов поїска | 10-04-2016 – 16-04-2016 р. |
| 7 | Тестування і доопрацювання розробленої програмної системи. | 17-04-2016 – 29-04-2016 р. |
| 8 | Оформлення пояснювальної записки, додатків, графічного матеріалу | 30-04-2016 – 31-05-2016 р. |
| 9 | Захист | 31-05-2016 р. |

Студент Шанiн Андрiй Олександрович

Керівник Ляпота Віталій Миколайович

« 1 »\_березня\_ 2016 р.

РЕФЕРАТ

Пояснювальна записка до курсової роботи: 29 с., 24 рис., 1 додаток, 5 джерел.

Мета роботи: розробка програми під назвою «Пошук оголошень», яка буде використовуватися для пошуку оголошень, на мові об’єктно-орієнтованого програмування С#.

Метод розробки: Microsoft Visual Studio 2012, WINDOWS FORMS, .NET Framework 4.5.

В результаті розробки отримана програма під назвою “Пошук оголошень”, яка дозволяє здійснювати створення, змiну, видалення оголошень, а також, звiсно, пошук оголошень за зазначеними критерiями. Також існує система тегов пошуку, яку оновлює адмiнiстративна група.

КУРСОВА РОБОТА, ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, МОВА ПРОГРАМУВАННЯ С#, ПОШУК ОГОЛОШЕНЬ, ОГОЛОШЕННЯ.

СОДЕРЖАНИЕ

Введение 6

1 Спецификация программы 7

* 1. Постановка задания 7
  2. Функции программы 7
  3. Описание интерфейса, предназначенного для пользователя………8
  4. Описание интерфейса, предназначенного для администратора 13

2 Проектная спецификация 14

2.1 Объектная модель программы 14

2.1.1 Класс File\_work 15

2.1.2 Класс Advertisment 16

2.1.3 Класс Users……….………………………………………………17

2.1.4 Класс Table……………………………………………………….18

3 Руководство пользователя 19

3.1 Установка программы 19

3.2 Работа программы (для пользователя) 19

3.2.1 Вход 20

3.2.2 Главное меню 20

3.2.3 Добавление объявления 21

3.2.4 Удаление объявления 21

3.2.5 Просмотр/редактирование объявления.………...……………..22

3.2.6 Статистика 22

3.3 Работа программы (для администратора)………………………......23

ВЫВОДЫ 24

ПЕРЕЧЕНЬ ССЫЛОК 25

Приложение А – Фрагменты кода программы 26

ВВЕДЕНИЕ

Многие века торговцы не могли с точностью определить собственные доходы. Во-первых, для этого нужно было вести гроссбухи с учетом поступлений История развития коммерческих объявлений и рекламы началась в глубокой древности. В античности информация передавалась преимущественно в устной форме, для информирования людей служили глашатаи. Постоянной информационной составляющей рекламы были «выкрики торговцев». Ремесло публичного рекламирования также было востребовано в средневековом обществе.

Новым этапом в истории рекламы стало изобретение в 1450 г. печатного станка. В объявлениях стало возможным совмещать текст с графическими изображениями. Размещение деловой и бытовой информации стало доступным широким слоям населения. После изобретения книгопечатания на городских стенах по всей Европе расклеивались не только рукописные, но и печатные листовки разнообразного содержания. Поток рекламных объявлений, поступающих в редакции периодических изданий, резко возрос в последней четверти XIX века. Коммерческая реклама превратилась в основной источник дохода газет. Существенно увеличились и тиражи периодических изданий. На рубеже – XX веков большие газеты распространялись тиражом в сотни тысяч экземпляров, и были рассчитаны на массовую аудиторию.

С возникновением Интернета появились невиданные возможности для рекламы и размещения объявлений. Стало возможным появление Интернет-магазинов, доступных к просмотру из любой точки мира, появились сайты бесплатных объявлений для различных регионов, мощные поисковые системы. Интернет предоставляет неограниченные возможности создания новых ресурсов для размещения объявлений, и получения информации пользователями.

В данном курсовом проекте разработана новая система с электронной доской объявлений, доступная широкому кругу пользователей. Система позволяет зарегистрированным пользователям размещать объявления о продаже и покупке товаров.

1 СПЕЦИФИКАЦИЯ ПРОГРАММЫ

1.1 Постановка задания

В курсовой работе необходимо разработать программу «Поиск Объявлений».

В программе должна храниться база объявлений (тему, заголовок, основной текст, характеристика “куплю-продам”, история изменения) и пользователей (логин, пароль, статус (модератор; администратор; пользователь) и история объявлений), программа должна уметь быстро находить то, или иное объявление а также оперировать с ними.

Будет предоставлен следующий функционал: авторизация, регистрация пользователя, создание, удаление, изменение объявлений, поиск объявлений по различным характеристикам, в том числе и система тегов поиска, администрирование.

Также важной частью программы является понятный интерфейс, он должен быть легким в освоении и приятным в пользовании для удобства работы пользователя.

1.2 Функции программы

Функции программы:

а) вход при помощи логина и пароля;

б) регистрация, с помощью логина, пароля и номера мобильного телефона;

в) добавление объявления;

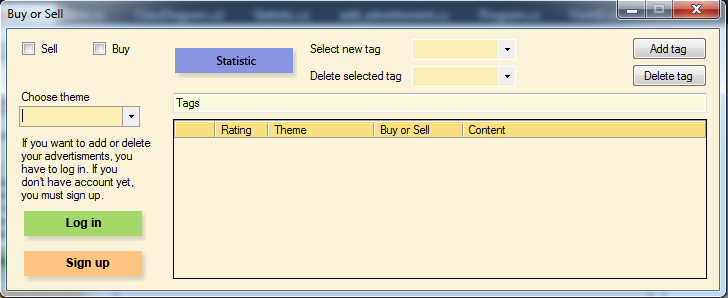
г) удаление объявления;

д) изменение объявления;

е) администрирование: добавление новых тегов поиска, изменение статуса пользователей.

1.3 Описание интерфейса, предназначенного для пользователя

Когда запустится программа на дисплее появится главная форма.



Главная форма приложения

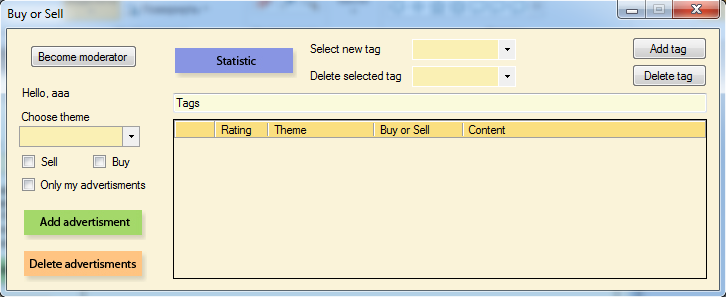
Когда откроется главная форма мы увидим таблицу объявлений. В левом верхнем углу кнопки “Buy” и “Sell” и выбигающий список “Choose theme”, из которого можно выбрать тему (тему можно также выбрать путем ввода с клавиатуры), в верхней правой части формы выделено место для выбора и удаления уже выбранных тегов поиска. При каждом изменении данных значений таблица объявлений будет изменятся, сортируя объявления в первую очередь по количеству тегов поиска, совпадающими с введенными, а во вторую – по рейтингу пользователя, добавившего это объявление.

Также в левой нижней части формы есть 2 кнопки “Log in” и “Sign up”, при нажатии на которые появятся форма регистрации или авторизации.

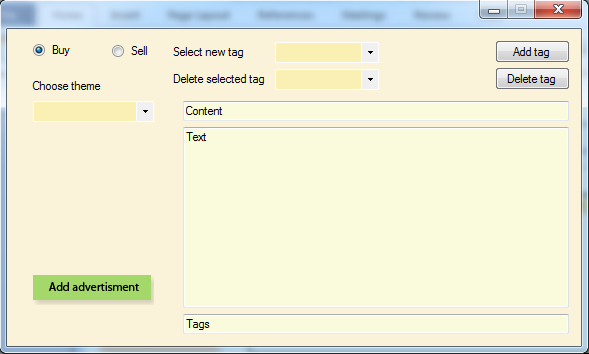
Формы регистрации и авторизации

При регистрации пользователь попадает в базу данных со статусом обычного пользователя. Как только пользователь авторизируется (возможно, после регистрации), основная форма изменится в один из трех вариантов, в зависимости от статуса пользователя.



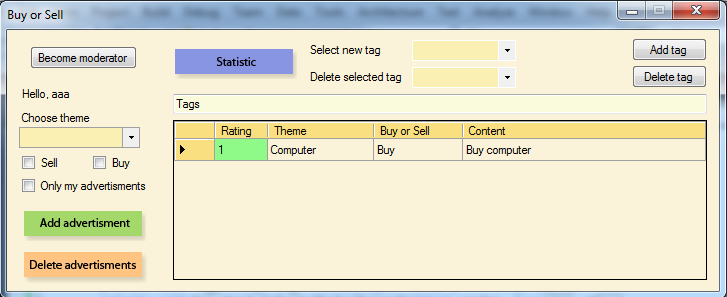
Интерфейс пользователя

В левом верхнем углу есть кнопка “Become moderator”, нажав на которую пользователь подаст заявку на пост модератора, которую обработает администратор. Также в левом нижнем углу есть 2 кнопки “Add advertisment” и “Delete advertisment”. При нажатии на кнопку “Add advertisment” появится форма добавления объявления.



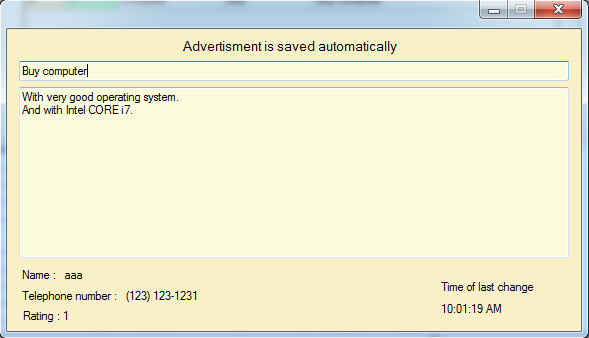
Форма добавления объявления

В левой части мы видим выпадающий список, в котором можно выбирать тему объявления. Если темы данного объявления в списке нету, то можно ввести собственную. Также мы видим в верхней части систему добавления и удаления тегов поиска, с помощью которых будет производится поиск данного объявления. В том случае, если конкретного тега поиска не окажется в базе существующих тегов, тег будет подтвержден (или нет) администратором или модератором. При нажатии на кнопку “Add advertisment” появится всплывающее окно, проверяющее действительно вы хотите добавить новое объявление. Если вы нажмете да, то объявление добавится в базу данных и объявление появится в таблице главной формы.



При нажатии на кнопку “Delete advertisment” появится всплывающее окно, проверяющее, действительно ли пользователь хочет удалить несколько объявлений. Если пользователь согласится, то из базы данных (а также таблицы) удалятся выделенные, как строка объявления.

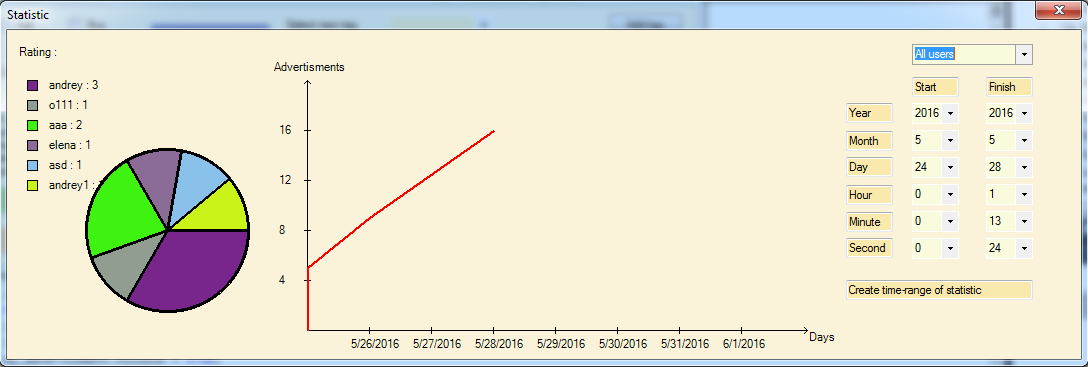
При двойном нажатии на объявление появится окно подробной информации о нем.



Форма просмотра объявлений

В том случае, если это объявление вывесил данный пользователь, то текстовые поля будут изменяемыми и пользователь сможет редактировать свое объявление. Иначе, если пользователь смотрит чужое объявление, то он сможет показать качество данного объявления, нажав на кнопку посредние нижней части формы, тем самым увеличив рейтинг пользователя, вывесевшего данное объявление.

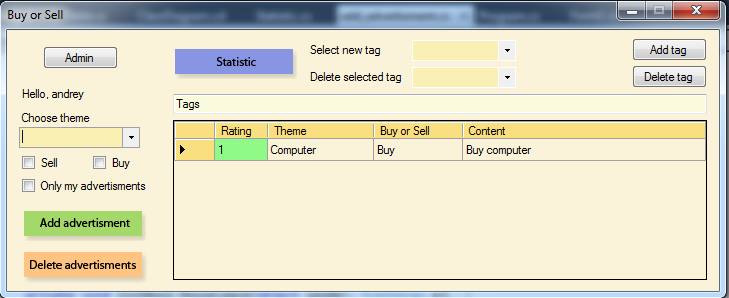
Также в главной форме есть кнопка “Statistics”, при нажатии на которую появится окно статистики.



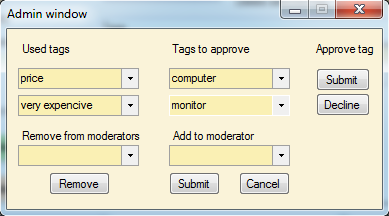
Форма статистики

В данном окне мы можем увидеть диаграмму, показывающую соотношение рейтинга текущего пользователя относительно суммы всех рейтингов. А также график зависимости колличества объявлений конкретного пользователя (или всех вместе) от времени. Также мы можем менять пользователя в выпадающем списке в верхнем правом углу и просматриваемые на графике даты.

1.4 Интерфейс администратора и модератора



Единственное отличие интерфейса администратора от интерфейса пользователя в том, что администрутор может открыть окно администрирования.



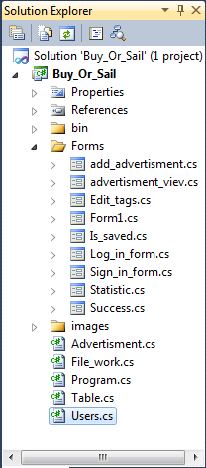
Меню администрирования

В окне администрирования можно добавлять теги поиска, которые были предложены пользователями. Что бы не было совпадений, есть список уже существующих тегов. Также, если вы администратор, а не модератор, то можете добавлять (или нет) пользователей, подавший заявку на модератора, и лишать модераторов прав. Иначе, если вы не администратор, то у вас нету возможности работать с пользователями.

2 ПРОЕКТНАЯ СПЕЦИФИКАЦИЯ

2.1 Объектная модель программы

Была разработана объектная модель программы, в которой продемонстрированы классы, что взаимодействуют друг с другом.

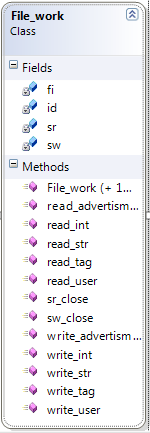


Демонстрация объектной модели

Папка “Forms” содержит все формы, которые используются в проекте. В ней содержится форма добавления объявления (“add\_advertisment.cs”), подробного просмотра объявления (“advertisement\_view.cs”), форма администрирования(“Edit\_tags.cs”), главная форма программы (“Form1.cs”), форма, указывающая на то, что объявления пользователя сохраннено (“Is\_saved.cs”), формы авторизации(“Log\_in\_form.cs”), и регистрации(“Sign\_in\_form”), форма статистики (“Statistics.cs”) и форма, вызывающаяся при необходимости подтвердить действие пользователя (“Success.cs”).

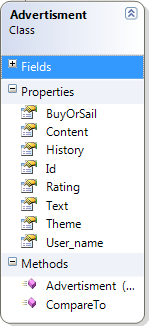
В данной программе реализованы классы: Advertisment.cs, File\_work.cs, Table.cs, Users.cs.

2.1.1 Класс File\_work



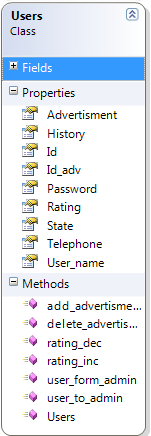
Класс File\_work используется для общего взаимодействия с базой данных. Поля класса : fi, id, sr, sw; Mетоды : read\_int, read\_str, read\_advertisment, read\_tag, read\_user, write\_advertisment, write\_int, wdite\_str, write\_tag, write\_user. Поле fi является объектом класса System.IO.FileReader. Поле id является числовым значением, которое хранит последний используемый уникальный идентефикатор. Поле sr является объектом класса System.IO.StreamReader и служит для считывания данных с текстового файла. Поле sw является объектом класса System.IO.StreamWriter и служит для записи данных в текстовый файл. Метод read\_int считывает числовое значение. Метод read\_str считывает строку. Метод read\_advertisment возвращает считанное объявление. Метод read\_tag возвращает объект, содержащий все теги данного характера. Метод read\_user возвращает считанные данные о пользователе. Методы write действуют аналогично, только записывая данные в файл.

2.1.2 Класс Advertisment



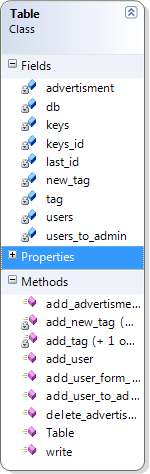
Класс Advertisment служит для описания объявления. Содержит свойства : Id, User\_name, Buy\_or\_Sail, Theme, Content, Text, History. Поле Id хранит в себе уникальный идентефикатор объявления. Поле User\_name хранит в себе логин пользователя, вывесевшего это объявление. Поле Buy\_or\_Sail хранит в себе информацию о том, объявление ли это продажи, или покупки. Поле Theme хранит в себе тему данного объявления. Поле Content хранит заголовок данного объявления. Поле Text хранит в себе основной текст данного объявления. Поле History хранит в себе всю историю данного объявления, пока его не удалят.

2.1.3 Класс Users



Класс Users служит для хранения данных о пользователе. Свойства : Advertisment, History, Id, Id\_adv, Password, Rating, State, Telephone, User\_name. Свойство Advertisment хранит в себе список объявлений, вывешенных этим пользователем. Свойство History хранит в себе историю добавления, удаления объявлений. Свойство Id хранит в себе уникальный идентефикатор пользователя. Свойство Ud\_adv хранит в себе уникальные идентефикаторы объявлеий, оцененых пользователем. Свойство Password хранит в себе пароль данного пользователя. Свойство Rating хранит в себе текущий рейтинг пользователя. Свойство State хранит в себе статус вользователя (admin/moderator/user). Свойство Telephone хранит в себе телефон пользователя. Свойство User\_name хранит в себе логин данного пользователя.

2.1.4 класс Table



Класс Table хранит в себе базу данных. Имеет поля : advertisement, db, last\_id, new\_tag, tag, users, user\_to\_admin. Поле advertisement хранит в себе словарь объявлений по id. Поле db хранит в себе словарь объявлений по темам. Поле last\_id хранит в себе последний используемы уникальный идентефикатор, увеличивающийся на 1 с помощью свойства Id. Поле new\_tag хранит в себе неподтвержденные теги. Поле tag хранит в себе подтвержденные теги. Поле users хранит в себе счисок пользователей. Поле user\_to\_admin хранит в себе пользователей, подавших заявку на статус модератора.

3 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

3.1 Установка программы

Для установки программы необходимо выполнить следующие шаги:

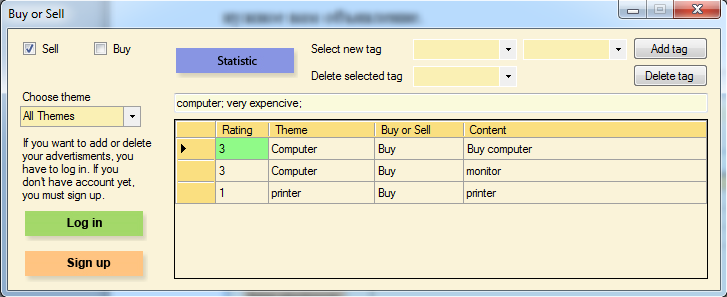
Шаг 1 – Открываем носитель с программой.

Шаг 2 – Извлекаем из архива программу в желаемую папку.

Шаг 3 – Запускаем, двойным нажатием на «.exe» файл, который можно найти по пути bin/Debug.

3.2 Работа с программой (для пользователя)

Запускаем программу. Появляется окно объявлений. Если вы не хотите добавлять, удалять, редактировать свои объявления, то вам не нужно регистрироваться или авторизироваться. Все что вам нужно это найти нужное вам объявление.



Окно поиска объявлений

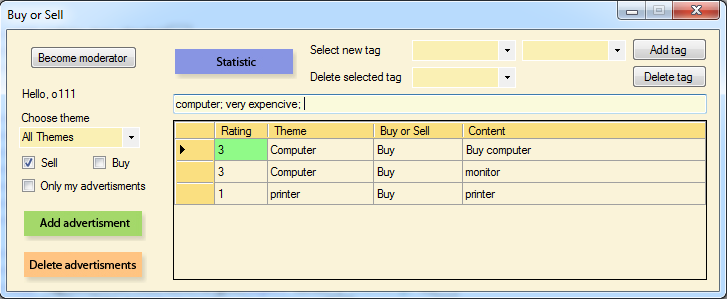
3.2.1 Вход

Чтобы войти в систему необходимо в начале зарегистрироваться. Нужно ввести логин, пароль, повторить пароль и телефон. В случае воода некорректных данных, или же если пользователь с таким логином уже существует, то программа покажет всплывающую строку, показывающую некорректность введенных данных. В случае, если вы уже зарегистрированы, вам нужно ввести свои логин и пароль в окно авторизации и нажать кнопку “Log in”. В случае ввода некорректнух данных будет показана всплывающая строка, показывающая что конкретно не то в введенных данных.



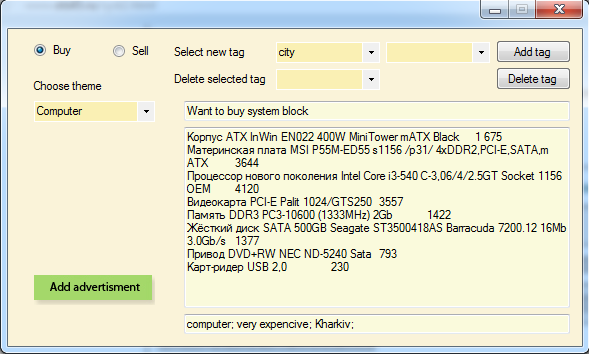
3.2.2 Главное меню

Когда пользователь авторизуется в программе, он увидит гзменения в основном окне. Там появятся кнопки добавления и удаления объявлений. Также, если пользователь не является модератором, он может увидеть кнопку “Become moderator”, при нажатии на которую в административную группу попадет заявка на становление пользователя модератором.



3.2.3 Добавление объявления.

Для добавления объвления нужно заполнить поля темы, заголовка и основного текста, а также выбрать одну из кнопок “Buy” и “Sell”, характеризующие то, преднозначено ли ваше объявление для поиска покупателей, или же продавцов. Также можно заполнить поле тегов поиска. В начале нужно выбрать (или ввести) тему тега, а потом и сам тег. Так же можно добавленные теги удалять.



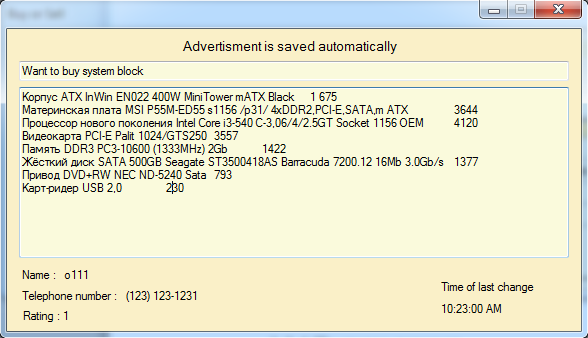
Добавление объявления

3.2.4 Удаление объявления

Что бы удалить объявление, или объявления, нужно выделить одну или несколько строк в таблице объявлений и нажать кнопку “Delete advertisment”. После подтверждения из таблицы исчезнут написанные вами объявления.

3.2.5 Просмотр/редактирование объявления.

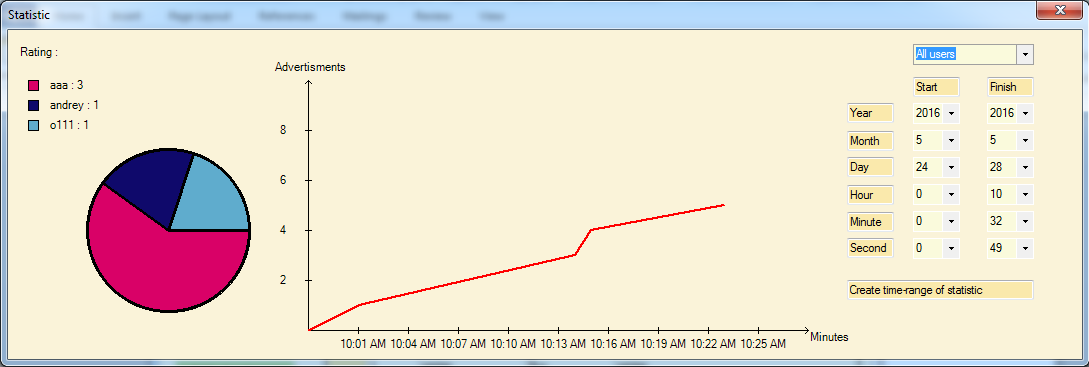
Что бы посмотреть объявление нужно дважды нажать на какой-нибудь элемент данного объявления в таблице. После этого появится окно объявления.



Если объявление написанно вами, то вы сможете его редактировать. Будте осторожны!!! Измененное объявление сохранится автоматически при закрытии окна. Если же это объявление было написанно не вами, то в нижней части окна посередине будет картинка руки с выгнутым большим пальцем, нажав на которую вы повысите рейтинг автора объявления. Таким способом вы можете показать что объявления этого автора достойны рассмотрения.

3.2.6 Статистика

Если нажать на кнопку “Statistic” в основном окне, то откроется окно статистки, в котором вы можете увидеть диаграмму, соотношения вешго рейтинга к рейтингу остальных пользователей. А также график зависимости колличества ваших объявлений от времени, напиная с даты регистрации пользователя. Также вы можете менять просматриваемого пользователя в выпадающем списке в верхнем правом углу и просматриваемые на графике даты.



Статистика

3.3 Работа с программой (для администратора или модератора)

Окно администрирования.

После входа в программу вы также можете увидеть список объявлений. Также можете добавлять, удалять после авторизации. Единственное отличие интерфейса административной группы это дополнительная кнопка, открывающее окно администрирования.



Если пользователь является модератором, то ему доступно лишь возможность редактировать систему тегов. То есть, добавлять (или отвергать добавление) того или иного тега, добавленного другими пользователями.

Если же пользователь является администратором, то ему доступна возможность добавлять (или отвергать заявку) пользователя на получение статуса модератора. Или же он может просто лишить некоторых модераторов дополнительных прав.

ВЫВОДЫ

Получена программа под названием «Поиск объявлений», которая позволяет оперировать с объявлениями : искать по определенным критериям, добавлять, удалять, изменять, а также изменять статус пользователей.

В результате написания курсового проекта был разработан на платформе .NET, в среде разработки Microsoft Visual Studio 2012 C#, полноценный программный продукт. Программа была полностью разработана согласно составленному плану. Также были изучены отдельные фазы разработки программного обеспечения и комбинирования их в единое целое. Главное то, что я не только закрепил знания с дисциплины, но и получил опыт в разработке программного продукта, изучил стандарты оформления документации.

В дальнейшем возможно расширение функционала программы.

ПЕРЕЛІК ПОСИЛАНЬ

|  |
| --- |
| 1. Архипов В.Е. [Текст]: «Принцип эффективности менеджмента и маркетинга» / В.Е. Архипов - М.,1998. – 270 c. |
| 1. Серегина Т.К. [Текст]: Реклама в бизнесе / Т.К. Серегина -Информационно-внедренческий центр "Маркетинг". М -1995. |

1. Microsoft Developer Network. Библиотека MSDN. Разработка на .NET [Електронный ресурс] / MSDN – сеть разработчиков Microsoft. – Режим доступа : www/URL:https://msdn.microsoft.com/ru-ru/ – 25.05.2016 г. – Загл. с экрана.
2. Бондарев В. М. «Объектно-ориентированное программирование на С#» [Текст] : учеб. Пособ / В. М. Бондарев. – Х.: Смит, 2009. – 224 с.;
3. Документація. Звіти у сфері науки і техніки : Структура і правила оформлення : ДСТУ 3008-95. – [Чинний від 1996–01–01]. – К. : Держстандарт України, 1995. – 37 с.

Приложение А

Фрагменты кода программы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Buy\_Or\_Sail.Classes

{

public class Table

{

int last\_id;

List<string> keys;

Dictionary<string, Users> users;

List<Users> users\_to\_admin;

Dictionary<string, List<int>> keys\_id;

Dictionary<int, Advertisment> advertisment;

Dictionary<string, Dictionary<int, Advertisment>> db;

Dictionary<string, Dictionary<string, List<Advertisment>>> tag;

Dictionary<string, Dictionary<string, List<Advertisment>>> new\_tag;

public List<Users> Users\_to\_admin { get { return users\_to\_admin; } }

public Dictionary<string, Dictionary<string, List<Advertisment>>> Tags { get { return tag; } }

public Dictionary<string, Dictionary<string, List<Advertisment>>> New\_tags { get { return new\_tag; } }

public Dictionary<int, Advertisment> Advertisment { get { return advertisment; } }

public Dictionary<string, List<int>> Keys\_id { get { return keys\_id; } }

public int Last\_id { get { last\_id++; return last\_id; } }

public Dictionary<string, Users> Users { get { return users; } }

public Dictionary<string, Dictionary<int, Advertisment>> DB

{

get { return db; }

}

public List<string> Keys

{

get { return keys; }

}

public Table()

{

File\_work file = new File\_work(new System.IO.StreamReader("DB.txt"));

last\_id = 0;

db = new Dictionary<string, Dictionary<int, Advertisment>>();

keys\_id = new Dictionary<string, List<int>>();

users = new Dictionary<string, Users>();

keys = new List<string>();

users\_to\_admin = new List<Buy\_Or\_Sail.Users>();

advertisment = new Dictionary<int, Advertisment>();

tag = new Dictionary<string, Dictionary<string, List<Buy\_Or\_Sail.Advertisment>>>();

new\_tag = new Dictionary<string, Dictionary<string, List<Buy\_Or\_Sail.Advertisment>>>();

//------------------------------------Read\_users--------------------------------

int n = file.read\_int();

for (int i = 0; i < n; i++)

{

Users user = file.read\_user();

users.Add(user.User\_name, user);

last\_id = Math.Max(last\_id, user.Id);

}

//------------------------------------Read\_users\_to\_admin----------------------------

n = file.read\_int();

for (int i = 0; i < n; i++) users\_to\_admin.Add(users[file.read\_str()]);

//------------------------------------Read\_advertisments--------------------------------

n = file.read\_int();

for (int i = 0; i < n; i++)

{

Advertisment adv = file.read\_advertisment();

if (!(db.ContainsKey(adv.Theme))) { db.Add(adv.Theme, new Dictionary<int, Advertisment>()); keys.Add(adv.Theme); keys\_id.Add(adv.Theme, new List<int>()); }

keys\_id[adv.Theme].Add(adv.Id);

db[adv.Theme].Add(adv.Id, adv);

advertisment.Add(adv.Id, adv);

users[adv.User\_name].Advertisment.Add(adv.Id, adv);

last\_id = Math.Max(last\_id, adv.Id);

}

//------------------------------------Read\_tags--------------------------------

n = file.read\_int();

for (int i = 0; i < n; i++)

{

string name = file.read\_str();

tag.Add(name, new Dictionary<string, List<Advertisment>>());

int m = file.read\_int();

for (int q = 0; q < m; q++)

{

KeyValuePair<string, List<int>> a = file.read\_tag();

tag[name].Add(a.Key, new List<Advertisment>());

for (int w = 0; w < a.Value.Count; w++)

if (advertisment.ContainsKey(a.Value[w]))

tag[name][a.Key].Add(advertisment[a.Value[w]]);

}

}

//------------------------------------Read\_new\_tags--------------------------------

n = file.read\_int();

for (int i = 0; i < n; i++)

{

string name = file.read\_str();

new\_tag.Add(name, new Dictionary<string, List<Advertisment>>());

int m = file.read\_int();

for (int q = 0; q < m; q++)

{

KeyValuePair<string, List<int>> a = file.read\_tag();

new\_tag[name].Add(a.Key, new List<Advertisment>());

for (int w = 0; w < a.Value.Count; w++)

if (advertisment.ContainsKey(a.Value[w]))

new\_tag[name][a.Key].Add(advertisment[a.Value[w]]);

}

}

file.sr\_close();

}

public void add\_user(Users User)

{

users.Add(User.User\_name, User);

}

public void add\_user\_to\_admin(Users User)

{

users[User.User\_name].user\_to\_admin();

}

public void add\_user\_form\_admin(Users User)

{

users[User.User\_name].user\_form\_admin();

}

public void add\_advertisment(Advertisment a, Dictionary<string, List<string>> tags)

{

string theme = a.Theme;

if (!(db.ContainsKey(theme)))

{

db.Add(theme, new Dictionary<int, Advertisment>());

keys.Add(theme);

keys\_id.Add(theme, new List<int>());

}

keys\_id[theme].Add(a.Id);

db[theme].Add(a.Id, a);

advertisment.Add(a.Id, a);

users[a.User\_name].add\_advertisment(a);

foreach (KeyValuePair<string, List<string>> x in tags)

{

add\_tag(a, x);

}

}

public void delete\_advertisment(Advertisment a)

{

keys\_id[a.Theme].Remove(a.Id);

advertisment.Remove(a.Id);

db[a.Theme].Remove(a.Id);

users[a.User\_name].delete\_advertisment(a);

}

public void write()

{

File\_work file = new File\_work(new System.IO.StreamWriter("DB.txt"), new System.IO.FileInfo("DB.txt"));

List<Users> User = new List<Users>(users.Values);

Dictionary<string, List<Advertisment>> dict = new Dictionary<string, List<Advertisment>>();

List<Advertisment> adv = new List<Advertisment>(advertisment.Values);

file.write\_int(User.Count);

for (int i = 0; i < User.Count; i++) file.write\_user(User[i]);

file.write\_int(users\_to\_admin.Count);

for (int i = 0; i < users\_to\_admin.Count; i++) file.write\_str(users\_to\_admin[i].User\_name);

file.write\_int(adv.Count);

for (int i = 0; i < adv.Count; i++) file.write\_advertisment(adv[i]);

file.write\_int(tag.Count);

foreach (KeyValuePair<string, Dictionary<string, List<Advertisment>>> a in tag)

{

file.write\_str(a.Key);

file.write\_int(a.Value.Count);

foreach (KeyValuePair<string, List<Advertisment>> b in a.Value)

{

file.write\_tag(b);

}

}

file.write\_int(new\_tag.Count);

foreach (KeyValuePair<string, Dictionary<string, List<Advertisment>>> a in new\_tag)

{

file.write\_str(a.Key);

file.write\_int(a.Value.Count);

foreach (KeyValuePair<string, List<Advertisment>> b in a.Value)

{

file.write\_tag(b);

}

}

file.sw\_close();

}

private void add\_new\_tag(Advertisment a, KeyValuePair<string, List<string>> x)

{

if (!new\_tag.ContainsKey(x.Key)) new\_tag.Add(x.Key, new Dictionary<string, List<Advertisment>>());

for (int i = 0; i < x.Value.Count; i++)

{

add\_new\_tag(a, new KeyValuePair<string, string>(x.Key, x.Value[i]));

}

}

private void add\_new\_tag(Advertisment a, KeyValuePair<string, string> x)

{

if (!new\_tag.ContainsKey(x.Key)) new\_tag.Add(x.Key, new Dictionary<string, List<Advertisment>>());

if (!new\_tag[x.Key].ContainsKey(x.Value)) new\_tag[x.Key].Add(x.Value, new List<Advertisment>());

new\_tag[x.Key][x.Value].Add(a);

}

private void add\_tag(Advertisment a, KeyValuePair<string, List<string>> x)

{

if (!tag.ContainsKey(x.Key)) add\_new\_tag(a, x);

else

{

for (int i = 0; i < x.Value.Count; i++)

{

add\_tag(a, new KeyValuePair<string, string>(x.Key, x.Value[i]));

}

}

}

private void add\_tag(Advertisment a, KeyValuePair<string, string> x)

{

if (!tag[x.Key].ContainsKey(x.Value)) add\_new\_tag(a, x);

else

tag[x.Key][x.Value].Add(a);

}

}

}