МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
Высшая школа экономики и управления
Кафедра «Информационные технологии в экономике»

РАБОТА ПРОВЕРЕНА
Рецензент, должность
_____/ И.О. Фамилия /
«_____» _____20____ г.

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой, д.т.н., с.н.с.
_____/ Б.М. Суховилов /
«_____»_____20____ г.

Разработка информационной системы поиска и обработки информации о государственных закупках

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.04.03.2019.301/67.ВКР

Консультант, должность
_____/ И.О. Фамилия /
«_____» _____20____ г.

Руководитель, доцент
_____/ Е.М. Сартасов /
«_____»_____20____ г.

.

Автор
студент группы ЭУ – 235
/ А.О. Беляева /
«_____» _____20____ г.

Нормоконтролер, ст.преподаватель
/ Е.Н. Горных /
«_____»_____20____ г.

Челябинск 2019

АННОТАЦИЯ

Беляева А.О. Разработка информационной системы поиска и обработки информации о государственных закупках. – Челябинск: ЮУрГУ, ЭУ-235, 64 с., 16 ил., 1 схема, 1 график, 2 табл., библиогр. список – 10 наим., 1 прил.

Целями данного проекта является обеспечение информатизации организации тендерного сопровождения малого бизнеса.

Разработано приложение, необходимое для организации тендерного сопровождения малого бизнеса.

Задачами являются анализ существующих программ на рынке, разработка дополнительного модуля по поиску государственных закупок, формирование базы данных, создание клиент-серверного приложения по работе с созданной ранее базой данных, расчет затрат и срока окупаемости.

ОГЛАВЛЕНИЕ

Введение 4

1 Анализ существующих информационных систем 6

1.1 Система Тендерплан 6

1.2 CRM-система Мэлтор 8

1.3 Сервис поиска Закупки.Контур 12

1.4 Система поиска тендеров СБИС 14

Вывод по первому разделу 16

2 Описание базы данных 17

Вывод по второму разделу 18

3 Модуль поиска государственных закупок 19

Вывод по третьему разделу 22

4 Описание клиент-серверного приложения 23

Вывод по четвертому разделу 25

5 Экономические аспекты 26

4.1 Расчет затрат 26

4.2 Расчет срока окупаемости 27

Вывод по четвертому разделу 27

Заключение 29

Библиографический список 31

Приложение А 32

ВВЕДЕНИЕ

Для информатизации организации тендерного сопровождения создано приложение, которое объединило в себе клиент-серверную часть и модуль поиска государственных закупок, а также база данных.

База данных создана при помощи СУБД Microsoft Access, которая входит в пакет приложений Microsoft Office 365. Данная СУБД выбрана по причине наличия высокого спектра функций, высокой безопасности, наличия связи с внешними таблицами и базами данных, улучшенной производительности обработки транзакций в сравнении с другими версиями, наличие встроенного языка VBA, который позволяет увеличить функционал работы с базами данных.

Клиент-серверная часть написана на языке C# на платформе .NET Framework 4.5 в среде разработки Microsoft Visual Studio 2017 [1].

Платформа .NET [7] поддерживается средой VS. А язык C# является языком разработки приложений платформы .NET. C# [6] не поддерживает множественное наследование классов в отличие от C++, но поддерживает множественное наследование интерфейсов, что нам и необходимо.

Модуль поиска государственных закупок был также написан на платформе .NET Framework в среде разработки Microsoft Visual Studio 2017 [1] на языке C#.

Разработка приложения по информатизации организации сопровождения государственных торгов МСП является актуальной темой. У малых субъектов бизнеса тендерного отдела нет, в отличие от крупных предприятий. Чтобы разобраться и начать активно принимать участие в государственных торгах, МСП обращается к организациям тендерного сопровождения.

Тендерный специалист владеет большой клиентской базой и базой государственных закупок. Такие данные могут быть непостоянны и периодически изменяться. Большой поток информации и быстрая смена данных без их фиксации приведет к большому количеству ошибок со стороны специалиста, что

не допустимо в данной области и может привести к большим финансовым потерям со стороны клиента.

Существует много программ, где информатизация организаций сопровождения закупок представлена на высоком уровне, но не учтено одно – организаций, обращающихся за помощью сопровождения, может быть более одной. В нашем случае взаимодействие специалиста по торгам с другими организациями будет более полноценно благодаря приложению, в котором будет храниться вся информация о проведенных и ожидающих торгах.

Объектом данного проекта является организация тендерного сопровождения и извещение (государственная закупка). Предмет – информационная система.

Цели проекта – разработать приложение информатизации организации тендерного сопровождения.

Задачи проекта:

- Изучить работу официального государственного сайта Zakupki.gov.ru:
- разработать алгоритм поиска государственных закупок ;
- создание базы данных, для хранения всей информации о клиентах, а также необходимых данных для беспрерывной работы организации;
- разработка клиент-серверного приложения;
- создание синхронизации между клиент-серверным приложением и модулем поиска закупок.

Разработана программа для информатизации организации тендерного сопровождения СМП.

Новизна проекта заключается в дополнительном модуле поиска. Информатизация происходит быстрее и качественнее, чем в любой другой программе.

1 АНАЛИЗ СУЩЕСТВУЮЩИХ ИНФОРМАЦИОННЫХ СИСТЕМ

На сегодняшний день существует большое количество систем по поиску тендеров. Каждая из них имеет свои достоинства и недостатки. Ниже мы рассмотрим самые популярные, разберем их особенности и сравним с нашей системой информатизации.

1.1 Система Тендерплан

Система Тендерплан [9], пример интерфейса которой представлен на рисунке 1, применяется для поиска и управления тендерами.

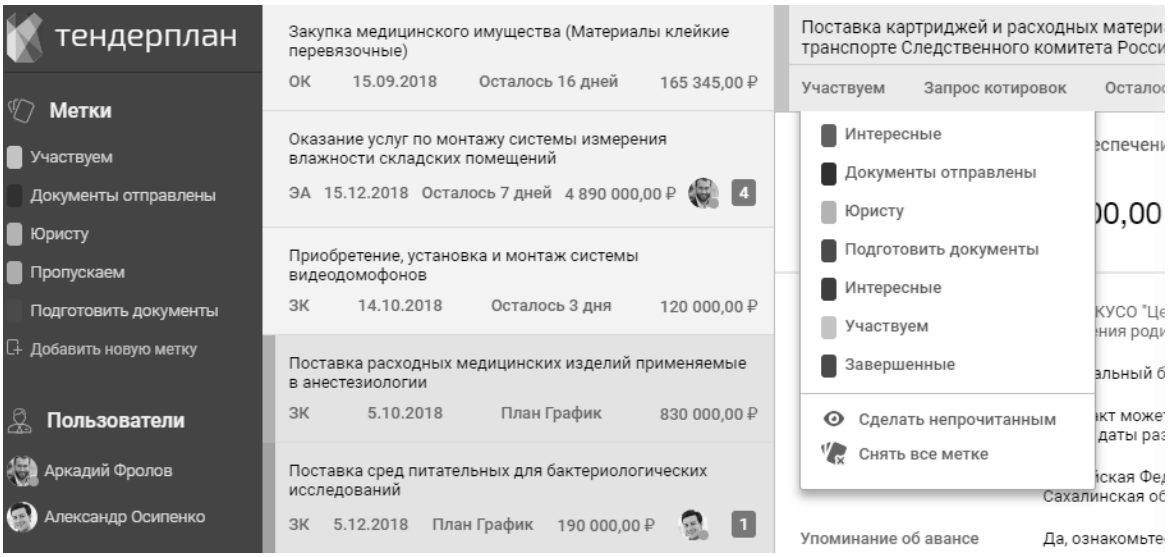

Рисунок 1 – Интерфейс системы Тендерплан

Система разработана для тендерного отдела организации. Функционал включает распределение задач между сотрудниками.

В настоящей системе он реализован через настройку ключевых слов и позволяет выполнять поиск тендеров по ключам, номенклатуре, названиям, кодам ОКПД (ОКПД2), региону, закону, площадкам, Заказчику, НМЦК, а также выполнять поиск внутри документации. Причем поиск происходит даже внутри архивов (zip, rar). Пример настройки поиска указан на рисунке 2.

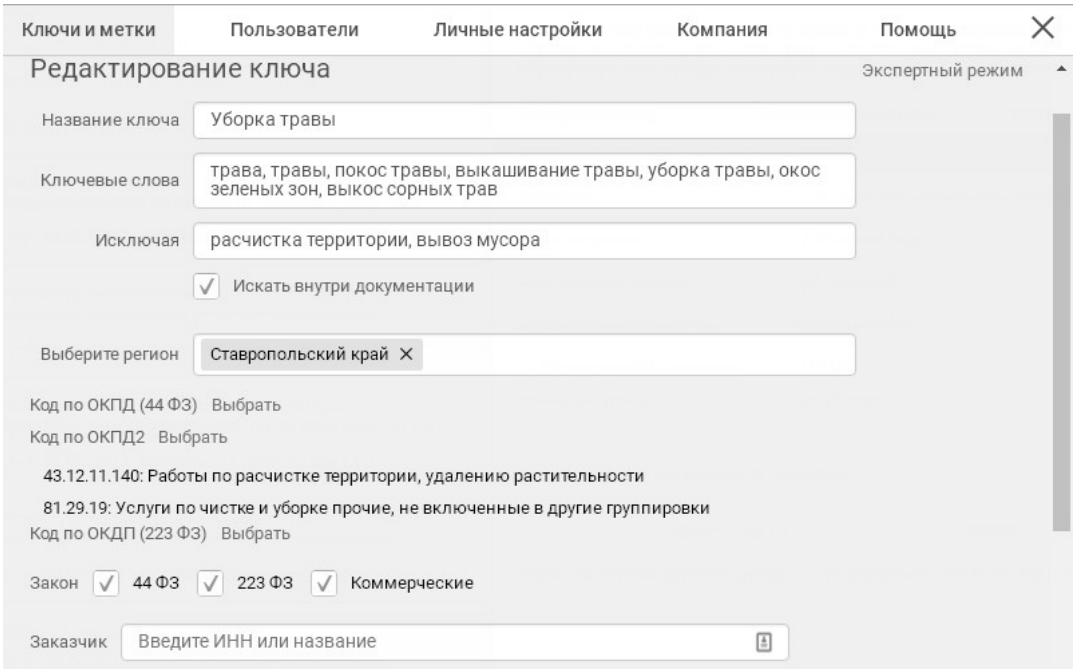

Рисунок 2 – Настройка поиска Тендерплан

Тендерплан использует нейросети, чтобы показывать дальнейшую конкуренцию в определенном тендере, возможные предложения по цене, предполагаемого победителя, рассчитывать риски участия, а также целесообразность участия.

К достоинствам данной системы относятся:

1) интуитивно понятный интерфейс;

2) поиск тендеров по ключевым словам в номенклатуре, названии, внутри документации или по ОКПД2;

3) функция «Метка» позволяет распределить тендера по нужным папкам для дальнейшей работы с ними;

4) удобный просмотр документации без скачивания;

5) использование нейросетей для анализа торгов, конкурентов и цен;

6) история обработки тендера;

7) аналитика участия своих конкурентов, оценивайте объем рынка, отслеживайте сезонность, проверяйте контрагентов;

8) выбор тарифа не влияет на функционал. В функции работают на любом тарифе.

К недостаткам данной системы относятся:

1) высокая цена: 3 месяца – 7 500 руб., 6 месяцев – 12 500 руб., 12 месяцев – 24 000 руб.;

2) база тендеров обновляется не сразу. Есть возможность пропустить несколько аукционов;

3) нет рассылки на почту с новыми закупками;

4) Тендерплан подходит для поставщика с тендерным отделом. Для организации, предоставляющей услуги по подготовке аукционов, данная система не подойдет.

Тендерплан очень удобен в использовании, но подходит лишь для тендерного отдела, что нас не устраивает. Стоимость высокая. Некоторые из предоставленных функций являются лишними.

1.2 CRM-система Мэлтор

Система Мэлтор [7], интерфейс которой представлен на картинке 3, является CRM-системой.

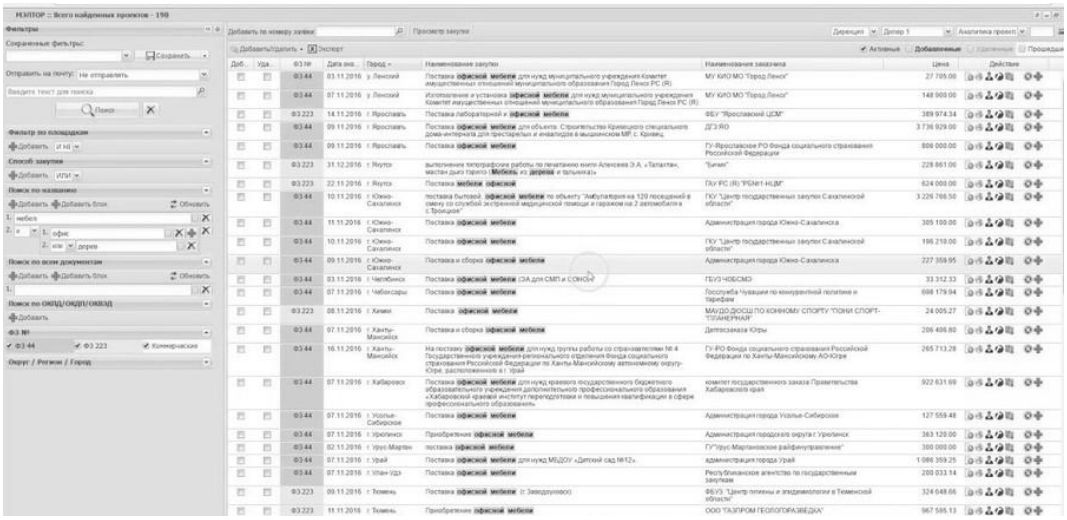

Рисунок 3 – Интерфейс системы Мэлтор

Продуктом компании МАСТЕР ЭЛЕКТРОННЫХ ТОРГОВ является программное обеспечение, предоставляющее доступ субъектов малого и среднего бизнеса к государственным и коммерческим закупкам.

Отличие от представленных на рынке конкурентных разработок заключается в том, что система Мэлтор обладает не только функционалом поиска государственных и коммерческих закупок, но и ERP модулем, а также функциями электронного документооборота.

Сервис ориентирован в первую очередь на поставщиков: генеральных директоров, представителей компаний и фирм, частных предпринимателей, сотрудников отделов, отвечающих за поиск тендеров и мониторинг потенциальных заказчиков. Благодаря индивидуальной настройке и интуитивно понятной структуре CRM подойдет как для людей, имеющих опыт работы с подобными системами, так и для новичков.

Автофильтры поиска позволяют подбирать тендеры по заданным критериям и уведомлять об их появлении на электронную почту. Можно находить предложения по ключевым запросам, их комбинациям и исключениям. Поиск осуществляется не только по названиям, но и по вложенным файлам документации и кодам классификаторов. Дополнительно можно установить фильтр по регионам или заказчикам. База компаний, избранные тендеры, задачи и сроки исполнения хранятся в Личном кабинете. Для удобства совместного

использования предусмотрен внутренний чат и распределение задач. Система также позволяет формировать отчеты в несколько кликов, выписывать счета, добавлять и удалять целые проекты, составлять планы закупок и календари задач. Прямо из CRM-системы доступны видеоинструкции по работе и архив всех проектов по годам и месяцам. МЭЛТОР позволяет одновременно управлять десятками и даже сотнями проектов, удобно систематизировав их и предоставив необходимые доступы сотрудникам.

Рабочий стол системы Мэлтор представлен на рисунке 4.

Рисунок 4 – Рабочий стол системы Мэлтор

К достоинствам данной системы относятся:

1) широкая настройка поиска;

2) функция отправки на электронную почту уведомления о появлении новых закупок;

3) функция выделения интересующего аукциона в новый проект, определение сроков и ответственных исполнителей, контроль подготовки документов;

4) документация аукциона и подготовленные документы для участия хранятся в системе;

5) календарь задач, который помогает не пропустить обозначенный срок, даже если проектный менеджер отсутствует;

6) настройка программы «под себя». Система может быть адаптирована под специфические требования бизнес-процессов. Есть возможность создания собственных проектов и задач, добавление сотрудников, настройки статусов и режимов доступа к информации;

7) гибкая настройка прав для каждого сотрудника.

К недостаткам данной системы относятся:

1) стоимость генеральной лицензии на программу составляет 1200 рублей в месяц, и пользовательская лицензия на 1-го сотрудника стоит 300 рублей;

2) сложный интерфейс;

3) сложная настройка поиска аукционов.

Стоимость системы Мэлтор с ростом сотрудников в тендерном отделе будет возрастать. Как и предыдущая система, Мэлтор рассчитан только для тендерного отдела организации.

1.3 Сервис поиска Закупки.Контур

Система Закупки.Контур [8], интерфейс которой представлен на рисунке 5, осуществляет поиск коммерческих и государственных закупок по 44-ФЗ и 223-ФЗ.

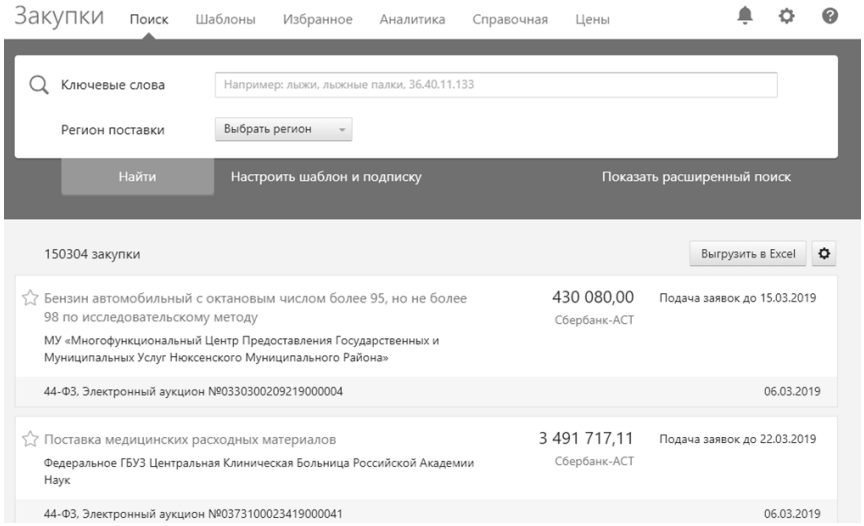

Рисунок 5 – Интерфейс Закупки.Контур

Закупки.Контур работает только с поставщиками и предоставляет поиск аукционов практически по всем торговым площадкам (229 площадок).

Базовый поиск в «Контур.Закупках» — это обычная поисковая строка. В ней можно ввести ключевые слова и выбрать интересующий регион. На рисунке 6 изображен пример базового поиска.

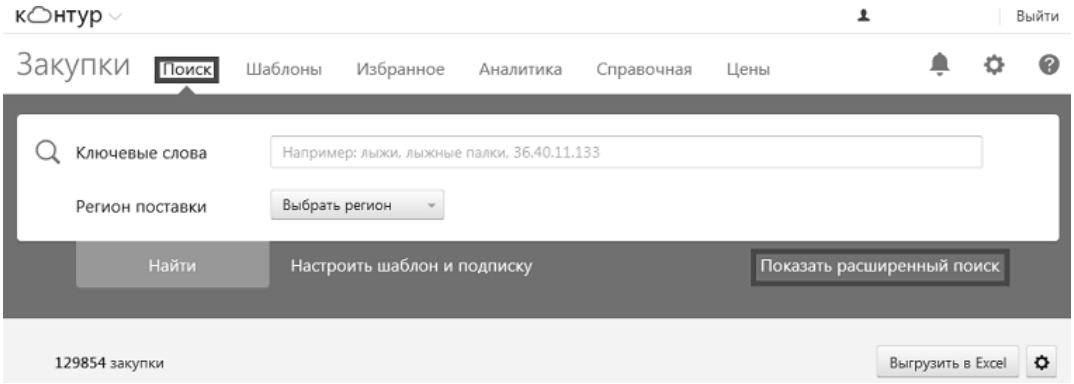

Рисунок 6 – Пример базового поиска

Наряду с этим, существует расширенный поиск, который позволяет использовать множество фильтров. Это дает гораздо более точные результаты. В окне расширенного поиска можно выбрать тип торгов, текущий этап процедуры, ЭТП, диапазон цены договора/контракта и другие параметры. Поиск также

осуществляется по ключевым словам, но помимо них, можно добавить слова, которые следуют из поиска исключить.

Пример расширенного поиска изображен на рисунке 7.

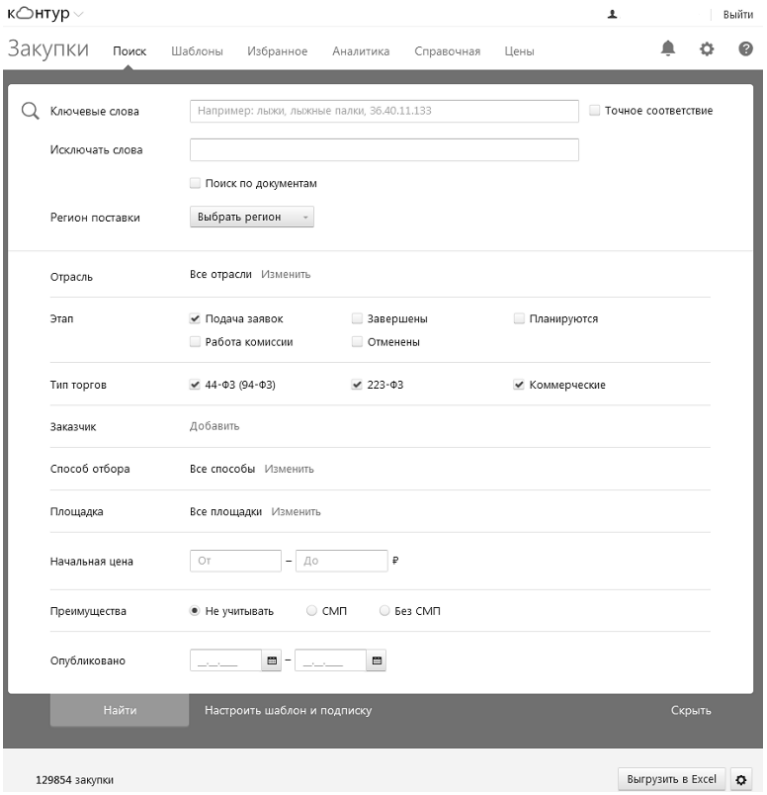

Рисунок 7 – Пример расширенного поиска

К достоинствам данной системы относятся:

1) интуитивно понятный и простой интерфейс;

2) регулярный поиск сохраняется в качестве шаблона;

3) функция «Избранное». Возможность разносить выбранные торги по разным папкам;

4) рассылка новых аукционов из выбранного шаблона на электронную почту;

5) выгрузка избранного в файл Excel;

6) анализ участников и результатов проведения торгов;

7) анализ заказчиков и конкурентов;

8) база документов жалоб в ФАС по 44-ФЗ;

9) доступны план-графики Заказчиков, что позволяет спланировать работу Поставщика.

К недостаткам данной системы относятся:

1) с системой может работать только один сотрудник;

2) функционал зависит от выбранного тарифа;

3) тариф действует 15 месяцев.

Стоимость подписки на систему Закупки.Контур ниже относительно рассмотренных программ. Нет лишнего функционала, только поиск и аналитика. Аналитика доступна только на самом дорогом тарифе «Эксперт», стоимостью 15 000 руб.

1.4 Система поиска тендеров СБИС

СБИС [10] — это онлайн-сервис, позволяющий организовать работу целого тендерного отдела, каждому сотруднику предоставляется доступ в личный кабинет с разграничением прав доступа.

Онлайн-система поиска и анализа закупок подключается по одному тарифу стоимостью 10 500 рублей, в который включен поиск по всем гос. площадкам, размещающим закупки по 44-ФЗ и 223-ФЗ, и более 100 коммерческим

площадкам. Поиск закупок идет по множеству параметров, а область поиска максимальна, включая конкурсную документацию. Выгрузка результатов поиска торгов возможна в PDF-формате - выгружается краткая информация по торгам, или в таблицу Excel - сохраняется подробное описание каждой процедуры. Система включает поставку мобильного приложения и сервис оповещения о новых торгах или отслеживание изменений выбранных закупок. По любому контрагенту возможно получение актуальной выписки из ЕГРЮЛ / ЕГРИП и подробной аналитики, собранной из всевозможных публичных источников (ФНС, Росстат, Казначейство, Арбитражный суд, электронные торговые площадки) и включающей сведения об участии в судебных процессах, данные о заключенных госконтрактах, о проведенных проверках со стороны гос. органов, об участии в электронных торгах: какие из них выиграны, какие проиграны, кто является заказчиками и конкурентами.

Пример интерфейса системы поиска тендеров СБИС представлен на рисунке 8.

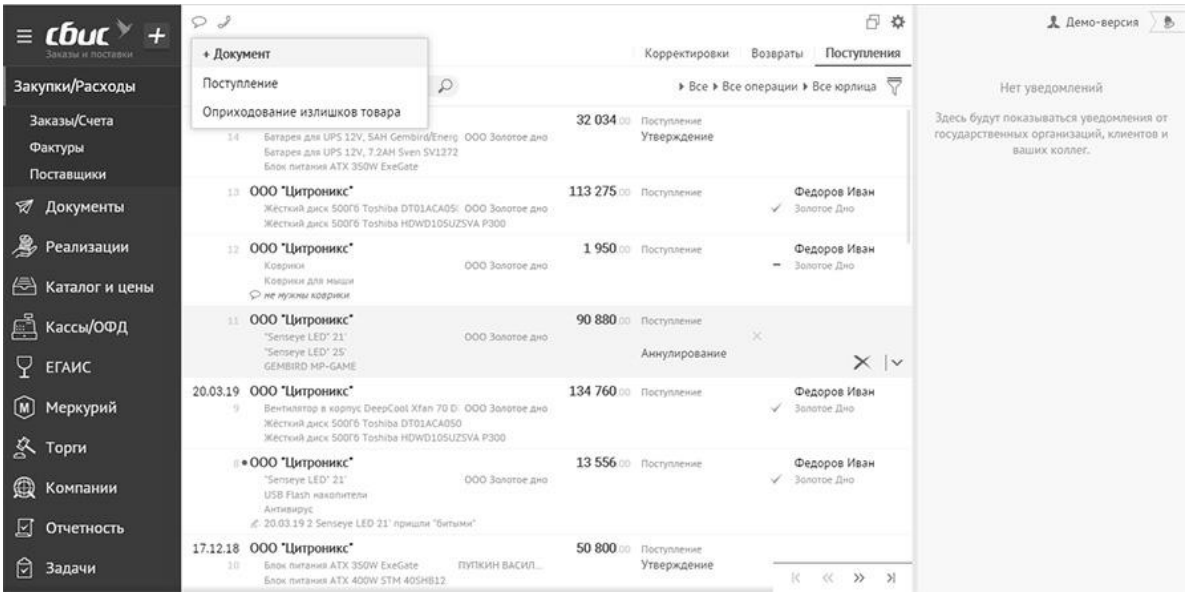

Рисунок 8 – Интерфейс системы СБИС

К достоинствам данной системы относятся:

1) Закупки по 44-ФЗ, 223-ФЗ и коммерческим площадкам
2) Сохраненные запросы и уведомления о новых торгах
3) Торги по компаниям, регионам, отраслям
4) Протоколы завершенных закупок
5) Отслеживание интересных Вам торгов
6) Базовые сведения по компаниям: реквизиты, выписки ЕГРЮЛ/ЕГРИП, данные о владельцах, руководителях и аффилированных лицах, финансовая отчетность, индекс надежности, статистика участия в судах

К недостаткам данной системы относятся:

1) с системой может работать только один сотрудник;
2) функционал зависит от выбранного тарифа;
3) поиск и анализ закупок доступен 12 месяцев;

Вывод по первому разделу

Из всех вышеперечисленных программ мы видим, что существует достаточно много систем с дружественным интерфейсом, доступных для малого бизнеса, но с ограниченным функционалом. Есть системы с увеличенной функциональностью, стоимость которых уже значительно повышается. Работа с этими системами достаточно сложна.

Главный недостаток всех описанных выше программных продуктов – отсутствие совмещения поиска тендеров и CRM-системы с интуитивно понятным интерфейсом.

Наша программа будет являться CRM-системой, иметь простой интерфейс, обладать возможностью поиска тендеров.

2 ОПИСАНИЕ БАЗЫ ДАННЫХ

Для хранения данных необходимо создать базу данных. База данных создана в СУБД Microsoft Access из пакета Microsoft Office 365. Схема данных изображена на рисунке 9.

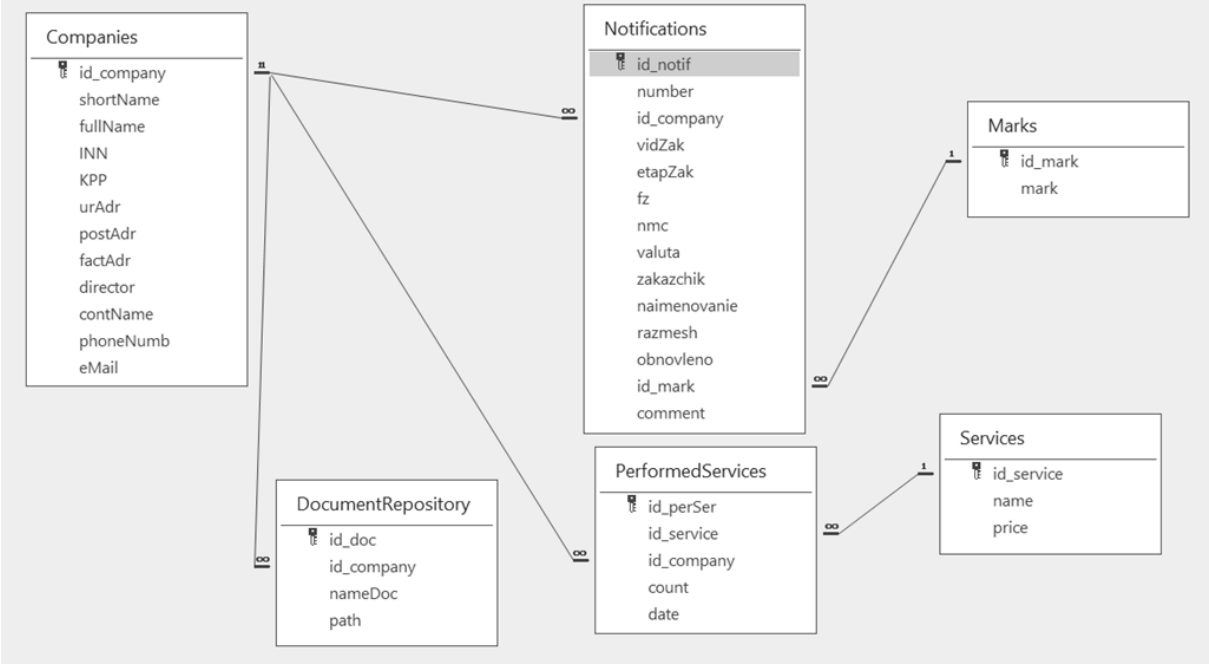

Рисунок 9 – Схема данных

Таблица Companies содержит информацию о всех организациях, что обратились за услугами тендерного сопровождения.

Таблица DocumentRepository содержит пути к документам клиентов, таких как карточка предприятия, выписка из СМП, выписка ЕРЮЛ/ЕРИП и так далее.

Таблицы Companies и DocumentRepository имеют связь «Один-ко-Многим» по полю id_company (Код компании).

Таблица Notifications содержит все извещения, что были отправлены на дальнейшую обработку после формы поиска извещений. Каждое извещение имеет привязку к одной организации. А также имеет статус, что позволяет отслеживать этап обработки извещения.

Таблицы Companies и Notifications имеют связь «Один-ко-Многим» по полю id_company (Код компании).

Таблица Marks содержит наименования всех статусов обработки извещений.

Таблицы Notifications и Marks имеют связь «Один-ко-Многим» по полю id_mark (Код статуса).

Таблица PerformedServices содержит все оказанные услуги клиенту, также стоимость этих услуг и дату их оказания.

Таблицы Companies и PerformedServices имеют связь «Один-ко-Многим» по полю id_company (Код компании).

Таблица Services содержит наименования всех услуг, которые оказывает наша организация.

Таблицы PerformedServices и Services имеют связь «Один-ко-Многим» по полю id_service (Код услуги).

Вывод по второму разделу

База данных создана в СУБД Microsoft Access. Построены 6 таблиц, связанных между собой. Созданы поля, определены первичные ключи.

На основе созданной базы данных будет реализовано клиент-серверное приложение.

3 МОДУЛЬ ПОИСКА ГОСУДАРСТВЕННЫХ ЗАКУПОК

Основной задачей данной работы является создание модуля поиска государственных закупок с настроенным поиском, используя пространство имен *System.Net.Http.*

Все государственные закупки по 44-ФЗ и 223-ФЗ расположены на официальном государственном сайте zakupki.gov.ru, в разделе Закупки. На рисунке 10 изображен раздел Закупки на сайте zakupki.gov.ru.

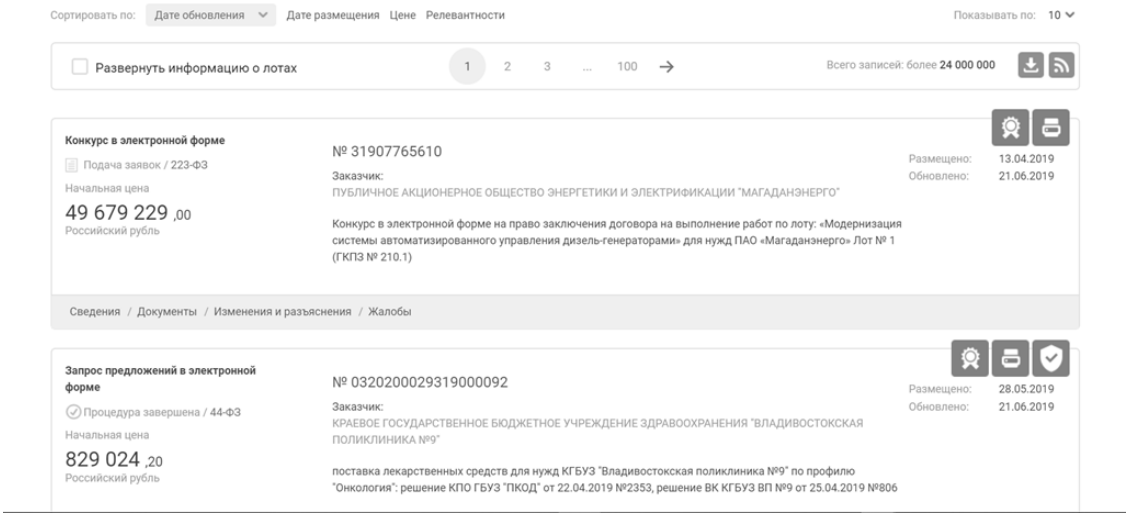

Рисунок 10 – Раздел «Закупки»

Для получения списка извещений необходимо создать HTTP-запрос и отправить его на сервер zakupki.gov.ru. Затем, получить ответ. Чтобы создать запрос, необходимо настроить выбранные параметры.

Схема взаимодействия приложения и сервера zakupki.gov.ru изображена на схеме 1.

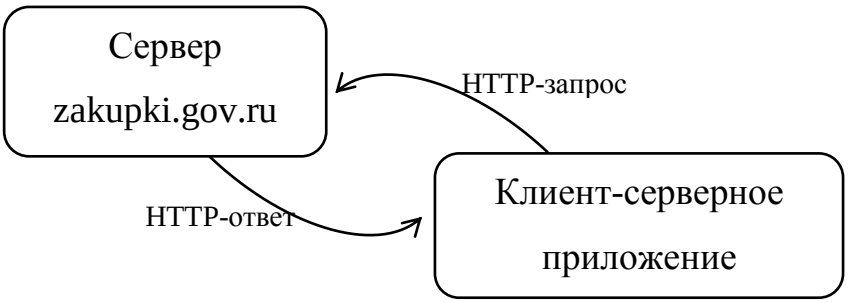

Схема 1 – Взаимодействие приложения и сервера zakupki.gov.ru

Интерфейс приложения имеет форму «Поиск закупок». На форме находятся элементы управления, которые отвечают за настройку приложения. Форма изображена на рисунке 11.

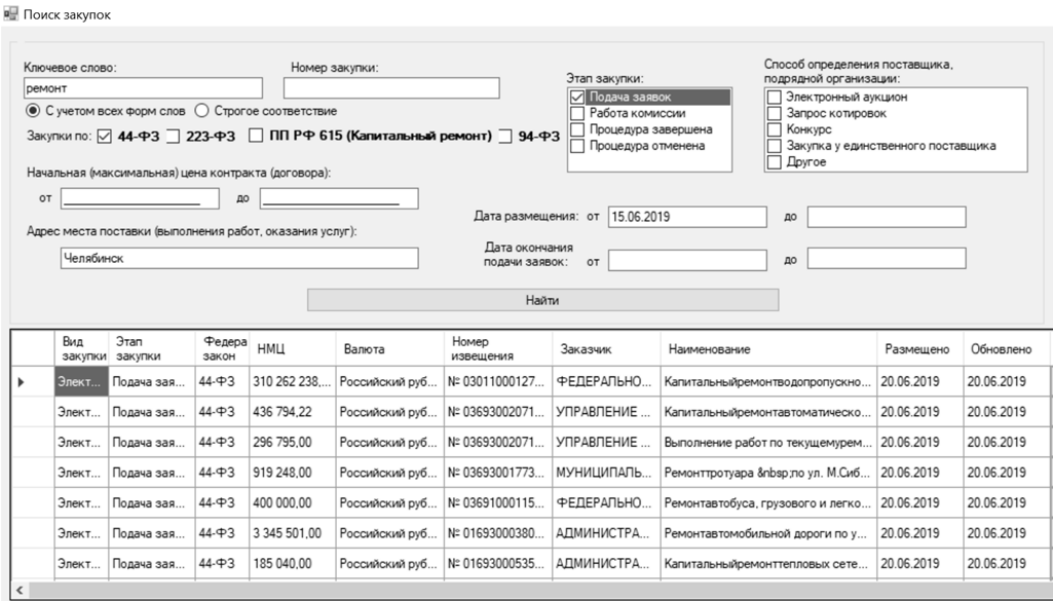

Рисунок 11 – Форма «Поиск Закупок»

Настройка поиска включает: ключевые слова, номер закупки, переключатель формы слов, выбор Федерального Закона, начальная максимальная цена, адрес места поставки, этап закупки, способ определения поставщика, дата размещения и дата окончания подачи заявок.

Можно настроить все поля, можно определить только часть, а можно оставить поля полностью пустыми, а chekBox’ы не отмеченными.

При полностью незаполненных полях на сервер отправляется пустой запрос. Затем получается ответ, что показано в листинге 1.

Листинг 1

```
            summaryStr = "http://zakupki.gov.ru/epz/order/extendedsearch/results.html?";
            HttpClient httpClient = new HttpClient();
            // Limit the max buffer size for the response so we don't get overwhelmed
            httpClient.MaxResponseContentBufferSize = 2560000;
            httpClient.DefaultRequestHeaders.Add("user-agent", "Mozilla/5.0 (compatible; MSIE
10.0; Windows NT 6.2; WOW64; Trident/6.0)");
            HttpResponseMessage response = await httpClient.GetAsync(summaryStr);
```

Если на форме какие-то поля заполнены, то на листинге 2, на примере выбора этапа и номера закупки, заполнение HTTP-запроса происходит следующим образом.

Листинг 2

```
  //номер закупки
if (this.textBox3.Text != "")
    orderNumber = "orderNumber=" + this.textBox3.Text + "&";
//этап закупки
if (this.checkedListBox1.GetItemChecked(0))
    stage = stage + "af=on&";
if (this.checkedListBox1.GetItemChecked(1))
    stage = stage + "ca=on&";
if (this.checkedListBox1.GetItemChecked(2))
    stage = stage + "pc=on&";
if (this.checkedListBox1.GetItemChecked(3))
            stage = stage + "pa=on&";
```

Следуя настройке, изображенной на рисунке 11, HTTP-запрос получается следующим:

http://zakupki.gov.ru/epz/order/extendedsearch/results.html?searchString=ремонт&morphology=on&recordsPerPage=_50&fz44=on&af=on&publishDateFrom=15.06.2019&deliveryAddress=Челябинск&.

Параметр *searchString* отвечает за ключевые слова, *morphology* – за формы слов, *recordsPerPage* – сколько страниц загружать на странице, *fz44* – 44-ФЗ включен, *publishDateFrom* – дата размещения извещения, *deliveryAddress* – место поставки.

Запрос, который отправляет браузер, выглядит следующим образом: *http://zakupki.gov.ru/epz/order/extendedsearch/results.html?morphology=on&openMode=USE_DEFAULT_PARAMS&pageNumber=1&sortDirection=false&recordsPerPage=_10&showLotsInfoHidden=false&fz44=on&af=on&currencyIdGeneral=-1&publishDateFrom=15.06.2019&regionDeleted=false&deliveryAddress=Челябинск&sortBy=UPDATE_DATE.*

Видно, что есть статичные параметры (параметры нашего запроса и запроса браузера совпадают), которые и определяют, какие именно извещения выбрать. Их настройку можно изменять, заполняя элементы управления на форме «Поиск закупок», получая различные варианты подборок.

В ответ на запрос получаем код страницы в HTML разметке. Отрывок полученного в ответ кода изображен на рисунке 12.

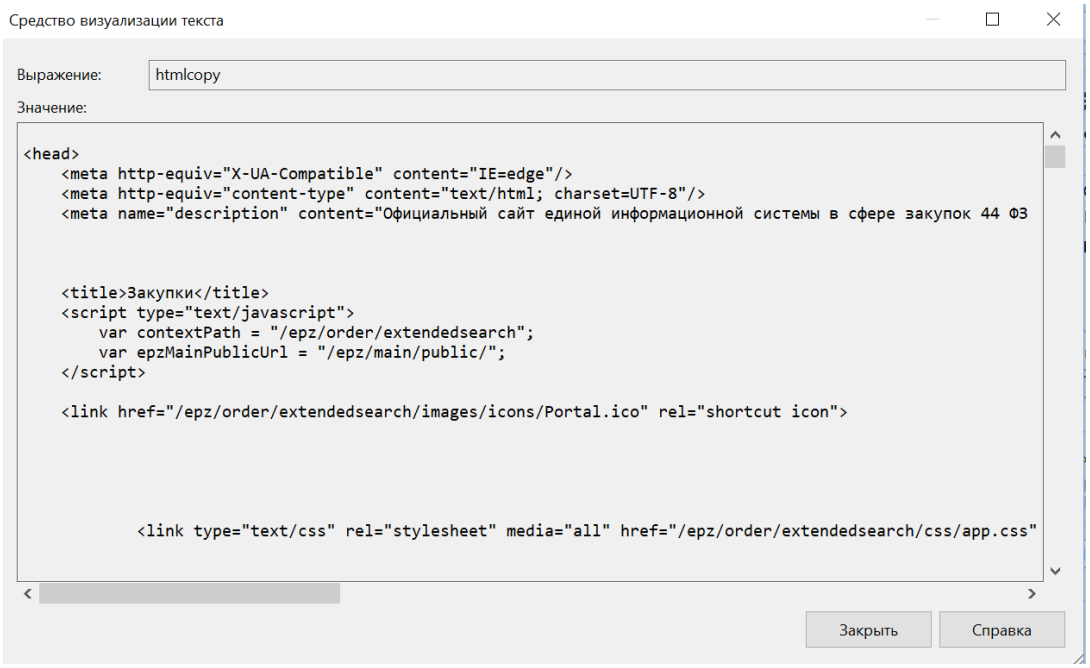

Рисунок 12 – HTTP-ответ

Полученный ответ содержит всю необходимую краткую информацию о закупках, чтобы сформировать список.

Следующим шагом является распарсивание данного HTML-кода.

Определяем имена классов необходимых нам полей. С помощью определенной логики, которая опирается на идею того, что параметры статичны, копируем нужные нам данные в массив, затем обрабатываем данные и выводим их в dataGridView. Результат работы видно на рисунке 11, что был показан ранее.

Вывод по третьему разделу

Удалось реализовать модуль поиска государственных закупок с использованием пространства имен System.Net.Http. Полученный ответ от сервера zakupki.gov.ru анализируется, обрабатывается и выводится в dataGridView.

4 ОПИСАНИЕ КЛИЕНТ-СЕРВЕРНОГО ПРИЛОЖЕНИЯ

Интерфейс клиент-серверного приложения реализован на .NET Framework [7] на языке C# [5] с использованием проекта Windows Forms.

Приложение запускается с помощью файла HTTP_.exe. После запуска открывается первая форма «Поиск закупок». На форме, изображенной на рисунке 13, возможно настроить условия поиска, выбрать необходимые параметры.

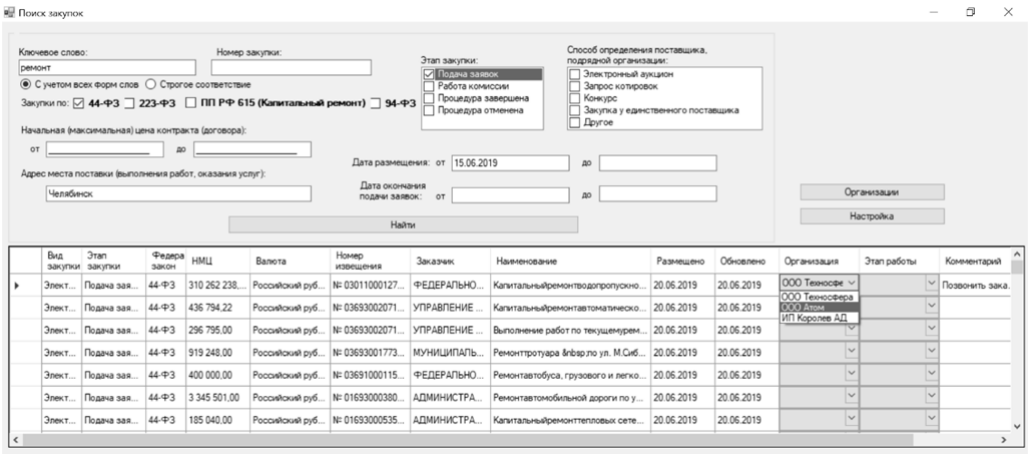

Рисунок 13 – Форма «Поиск закупок»

Нажимая на кнопку «Найти», в таблицу ниже выводится результат поиска извещений в зависимости от условий, выбранных выше.

Каждой строчке можно выбрать организацию с помощью поля «Организация», к которой данное извещение будет прикреплено для дальнейшей работы. А также определить статус извещения на поле «Этап работы».

Поле «Комментарий» можно заполнить любыми значениями, которые необходимы.

После редактирования трех ячеек, описанных выше, строчка с информацией об извещении автоматически записывается в таблицу «Notifications» базы данных.

На форме выше есть две кнопки.

Кнопка «Организации» открывает форму «Организации», показанную на рисунке 14.

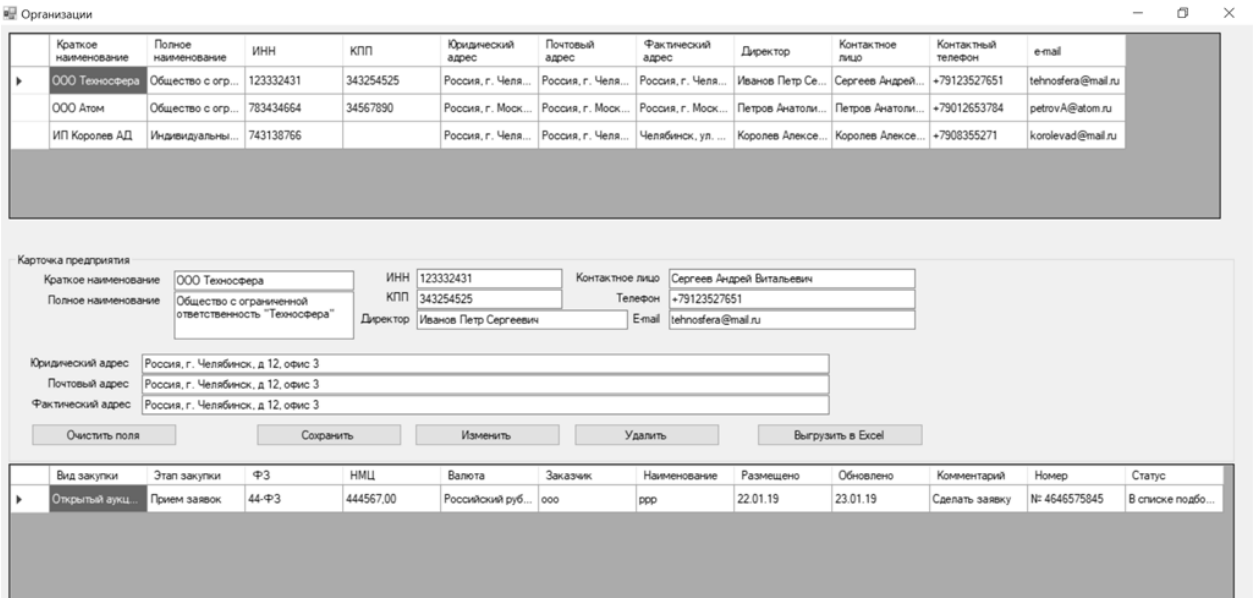

Рисунок 14 – Форма «Организации»

В первой таблице находится весь список организаций, которые являются нашими клиентами, и краткая информация.

Переходя по записям таблицы, информация в полях ниже будет изменяться в соответствии с выбранной организацией. Информацию можно изменить, удалить, а также добавить нового клиента.

Во второй таблице выводится информация о извещениях, которые привязаны к данному клиенту. И возможно изменить статус работы над аукционом, изменить комментарий.

Список аукционов организации возможно вывести в таблицу Excel для дальнейшей рассылки на электронную почту.

Кнопка «Настройка» открывает форму «Настройка», которая изображена на рисунке 15.

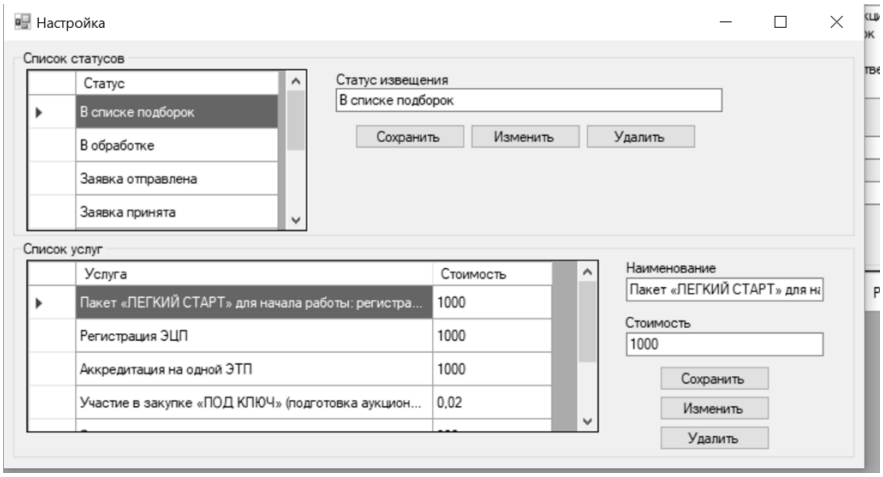

Рисунок 15 – Форма «Настройка»

В первой таблице выводится список всех статусов извещений.

В второй таблице выводятся наименования услуг и их стоиость.

Можно добавить, изменить или удалить статус извещения. А также добавить, изменить или удалить список услуг.

Вывод по четвертому разделу

Приложение реализовано на .NET Framework на языке C# с использованием проекта Windows Forms.

Настроен поиск государственных закупок и вывод подборки аукционов в таблицу.

Форма, содержащая информацию о клиентах, выводит информацию о прикрепленных аукционах и их статусе в работе.

Реализована форма настройки статусов аукциона и списком оказываемых услуг.

5 ЭКОНОМИЧЕСКИЕ АСПЕКТЫ

При принятии решения о создании и внедрении информационной системы тендерного сопровождения малого бизнеса необходимо рассчитать экономическую эффективность [4] и целесообразность внедрения разработки. Для определения экономической эффективности необходимо рассчитать эффект окупаемости [2].

Для определения эффекта окупаемости используется методика срока окупаемости [3], для этого рассчитываются затраты до внедрения и после внедрения проекта.

4.1 Расчет затрат

Затраты на покупку лицензионного программного обеспечения и ПК представлены в таблице 1.

Таблица 1 – Затраты на ПО и ПК

Наименование ПО	**Стоимость, руб.**
Microsoft Visual Studio 2017	0
Пакет Microsoft Office 365 (Access, Excel), 12 мес.	3 290
Microsoft Windows 10 Professional 64Bit Russian 1pk	12 299
Ноутбук: Core 5, 8Гб, HDD, 15”,	50 000
Итого:	65 589

Расчет заработной платы работников, разрабатывающих и внедряющих проект, приведен в таблице 2.

Таблица 2 – Расчет заработной платы

Должность	**Оклад, руб./день**	**Продолжительность работ, дней**	**Итого, руб.**
Программист C++/C#	3 319	10	33 160
Системный администратор	1 304	8	10 432
Итого:			43 592
Дополнительная заработная плата (20% от основной)			8 718,4

Основная и дополнительная заработная плата	52 310,4

Продолжение таблицы 1

Социальные платежи (26% от основной и дополнительной заработной платы)	10 462,08

В результате затраты на разработку и внедрение программного продукта составляют 128 361,48 руб.

4.2 Расчет срока окупаемости

Подсчитано, что в затраты на создание и внедрение программы в общей сумме составят 128 361,48 руб. После завершения разработки и внедрения программист не нужен на постоянной основе. Расходы на его заработную плату сокращаются. На рисунке 16 видно, что срок окупаемости программного продукта составит около двух месяцев.

Рисунок 16 – График срока окупаемости

Вывод по четвертому разделу

В организационно-экономическом разделе проанализированы и рассчитаны статьи затрат на разработку и внедрение, вычислен срок окупаемости проекта.

Затраты на разработку проекта оказались быстро окупаемыми. Организация тендерного сопровождения в месяц имеет значительную сумму прибыли, что и объясняет такой короткий срок окупаемости.

ЗАКЛЮЧЕНИЕ

Реализация информационной системы поиска и обработки информации о государственных закупках выполнена успешно. Проведен анализ, обоснована актуальность и подсчитан срок окупаемости.

В первой главе мы рассмотрели аналогичные программы для поиска государственных закупок. Рассмотрены четыре системы и их плюсы и минусы. Сделаны выводы и определены причины разработки собственной программы.

Во второй главе создана база данных. База данных создана в СУБД Microsoft Access. Реализованы таблицы, поля и связи между ними. Создана схема данных, где наглядно видно типы связей.

В третьей главе разработан модуль поиска государственных закупок. Реализована настройка параметров поиска. Вывод подборки, согласно введенным условиям, проходит успешно.

Подборка аукционов происходит только для государственных торгов. Существуют также коммерческие закупки, расположенные на коммерческих площадках. Эта задача оставлена для дальнейшей разработки.

На форме «Организации» не реализован функционал фильтрации аукционов по их статусу для правильной выгрузки в Excel. Эта задача также оставлена для дальнейшей разработки.

В четвертой главе описан функционал клиент-серверного приложения организации тендерного сопровождения. Описаны все формы, все элементы управления.

Желаемые функции приложения реализованы полностью не были. Эти задачи оставлены на дальнейшую разработку.

Целью проекта являлась разработка приложения информатизации организации тендерного сопровождения. Цель достигнута успешно.

Задачи проекта также реализованы успешно:

- Изучить работу официального государственного сайта Zakupki.gov.ru:

- разработать алгоритм поиска государственных закупок;
- создание базы данных, для хранения всей информации о клиентах, а также необходимых данных для беспрерывной работы организации;
- разработка клиент-серверного приложения;
- создание синхронизации между клиент-серверным приложением и модулем поиска закупок.

Проект выполнен успешно, поставленная цель достигнута, описанные задачи реализованы успешно. Некоторый функционал оставлен для дальнейшей разработки.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Рендольф, Н. Visual Studio для профессионалов / Ник Рендольф и др. ; пер. с англ. Д. А. Клюшина, В. А. Коваленко, И. В. Красикова. – М: Диалектика, 2011. – 1177 с.
2. Матвейкин, В.Г. Информационный менеджмент: курс лекций / Б.С. Дмитриевский, К.А. Садов. – Тамбов: Тамб. гос. техн. ун-та, 2009. – 44 с.
3. Ример, М.И. Экономическая оценка инвестиций / А.Д. Касатов, Н.Н. Матиенко – СПб: Питер, 2008. — 480 с.
4. Непомнящий, Е.Г. Экономическая оценка инвестиций / уч. Пособие – Таганрог: ТРТУ, 2005. – 292с.
5. Виссер, Дж. Разработка обслуживаемых программ на языке C# – М: ДМК Пресс, 2017. – 192с.
6. Шарп, Дж. Microsoft Visual C#. Подробное руководство – СПб: Питер, 2017. – 848с.
7. Мэлтор: https://meltor.ru
8. Закупки.Контур: https://zakupki.kontur.ru
9. Тендерплан: https://tenderplan.ru
10. СБИС: https://sbis.ru/tenders

ПРИЛОЖЕНИЕ А

Код формы «Поиск закупок»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Net.Http;
using Excel = Microsoft.Office.Interop.Excel;

namespace HTTP_
{
    public partial class FormSearch : Form
    {
        private Excel.Application excelapp;                 // Программа Excel
        private Excel.Window excelWindow;                   // Окно программы Excel
        private Excel.Workbooks excelappworkbooks;  // Рабочие книги
        private Excel.Workbook excelappworkbook;    // Рабочая книга
        private Excel.Sheets excelsheets;                   // Рабочие листы
        private Excel.Worksheet excelworksheet;         // Рабочий лист
        private Excel.Range excelcells;                         // Диапазон ячеек или
ячейка
```

```
public FormSearch()
        {
            InitializeComponent();



        }

        private async void button1_Click(object sender, EventArgs e)
        {
            this.companiesTableAdapter.Fill(this.zakupkiDataSet.Companies);
            this.marksTableAdapter.Fill(this.zakupkiDataSet.Marks);

            this.dataGridView1.Rows.Clear();
            this.button1.Enabled = false;
            //====================Формирование переменных для
запроса============================================================

            string searchString = ""; //textbox1
            string strictEqual = ""; //radiobutton1 и radiobutton2
            string orderNumber = ""; //textbox3
            string stage = ""; //checklistbox1
            string priceFromGeneral = ""; //maskedtexbox1
            string priceToGeneral = ""; //maskedtextbox2
            string fz = ""; //checkbox1-4
            string publishDateFrom = ""; //datetimepicker1
            string publishDateTo = ""; //datetimepicker2
            string deliveryAddress = ""; //textbox4
            string applSubmissionCloseDateFrom = "";//datetimepicker3
            string applSubmissionCloseDateTo = "";//datetimepicker4
```

```
string placingWaysList = "";//checkedlistbox2
string summaryStr = "";



//ключевые слова
if (this.textBox1.Text != "")
    searchString = "searchString=" + textBox1.Text + "&";

// форм слов или строгое соответствие
if (this.radioButton2.Checked)// форм слов или строгое соответствие
    strictEqual = "strictEqual=on&";
else strictEqual = "morphology=on&";

//выбор ФЗ
if (this.checkBox1.Checked)
    fz = fz + "fz44=on&";
if (this.checkBox2.Checked)
    fz = fz + "fz223=on&";
if (this.checkBox3.Checked)
    fz = fz + "ppRf615=on&";
if (this.checkBox4.Checked)
    fz = fz + "fz94=on&";

//номер закупки
if (this.textBox3.Text != "")
    orderNumber = "orderNumber=" + this.textBox3.Text + "&";

//этап закупки
```

```
            if (this.checkedListBox1.GetItemChecked(0))
                stage = stage + "af=on&";
            if (this.checkedListBox1.GetItemChecked(1))
                stage = stage + "ca=on&";
            if (this.checkedListBox1.GetItemChecked(2))
                stage = stage + "pc=on&";
            if (this.checkedListBox1.GetItemChecked(3))
                stage = stage + "pa=on&";

            //начальная максимальная цена контракта
            if (this.maskedTextBox1.Text != "")
                priceFromGeneral = "priceFromGeneral=" +
this.maskedTextBox1.Text + "&";
            if (this.maskedTextBox2.Text != "")
                priceToGeneral = "priceToGeneral=" + this.maskedTextBox2.Text +
"&";

            //дата размещения
            if (this.textBox5.Text != "")
                publishDateFrom = "publishDateFrom=" + this.textBox5.Text + "&";
            if (this.textBox6.Text != "")
                publishDateTo = "publishDateTo=" + this.textBox6.Text + "&";

            //Адрес места поставки
            if (this.textBox4.Text != "")
                deliveryAddress = "deliveryAddress=" + this.textBox4.Text.Replace("
", "+") + "&";

            //Дата окончания подачи заявок
```

```
            if (this.textBox7.Text != "")
                applSubmissionCloseDateFrom = "applSubmissionCloseDateFrom="
+ this.textBox7.Text + "&";
            if (this.textBox8.Text != "")
                applSubmissionCloseDateTo = "applSubmissionCloseDateTo=" +
this.textBox8.Text + "&";

            //Способ определения поставщика
            if (this.checkedListBox2.GetItemChecked(0) ||
this.checkedListBox2.GetItemChecked(1) || this.checkedListBox2.GetItemChecked(2) ||
this.checkedListBox2.GetItemChecked(3) || this.checkedListBox2.GetItemChecked(4))
            {
                placingWaysList = "placingWaysList=";

                if (this.checkedListBox2.GetItemChecked(0))
                    placingWaysList = placingWaysList +
"2CAESMBO%2CEA615%2CZA44%2CEF";
                if (this.checkedListBox2.GetItemChecked(1))
                    placingWaysList = placingWaysList +
"%2CZK504%2CZP504%2CZKESMBO%2CZPESMBO%2CZP44%2CZKB44%2CZ
K";
                if (this.checkedListBox2.GetItemChecked(2))
                    placingWaysList = placingWaysList +
"%OK504%2COKU504%2COKD504%2CKESMBO%2CZKK44%2COKU44%2COK
D44%2COK%2CZKKU44%2CZKKD44";
                if (this.checkedListBox2.GetItemChecked(3))
                    placingWaysList = placingWaysList + "%2CES%2CEP";
                if (this.checkedListBox2.GetItemChecked(4))
```

```
                placingWaysList = placingWaysList + "%2CPO%2CPO615%2CTB%2CZH%2CSZ%2CINM111%2CPR";
            }

            //==================================Формирование строки запроса=============================================================

            summaryStr = "http://zakupki.gov.ru/epz/order/extendedsearch/results.html?";
            summaryStr += searchString;//добавляем ключевые слова
            summaryStr += strictEqual;//добавляем соответствие (первые радиобаттоны)
            summaryStr += "recordsPerPage=_50&";//количество записей на странице
            summaryStr += fz;//выбор ФЗ
            summaryStr += orderNumber;//добавление номера закупки
            summaryStr += stage;// этап закупки
            summaryStr += priceFromGeneral + priceToGeneral;//НМЦ контракта
            summaryStr += publishDateFrom + publishDateTo;//дата размещения
            summaryStr += deliveryAddress;//адрес места поставки
            summaryStr += applSubmissionCloseDateFrom + applSubmissionCloseDateTo;//дата окончания подачи заявок

            //
============================================================================================================
            HttpClient httpClient = new HttpClient();
```

```
// Limit the max buffer size for the response so we don't get overwhelmed
httpClient.MaxResponseContentBufferSize = 2560000;
httpClient.DefaultRequestHeaders.Add("user-agent", "Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.2; WOW64; Trident/6.0)");
//try
//{
string responseBodyAsText;
string Buf2 = "";
string Buf3 = " Нал. USD";
//  textBox2.Text = "Waiting for response ...";
//подстроки для парсинга
string substr_boxIcons = "<div class=\"boxIcons\">";
string substr_reportBox = "<div class=\"reportBox\">";
string[] lines = new string[100000000];
string[,] parsinglines = new string[100000, 1000];
int i = 0;
int j = 0;
string substr_div = "</div>";
string summaryline = "";


HttpResponseMessage response = await httpClient.GetAsync(summaryStr);
response.EnsureSuccessStatusCode();

// textBox2.Text = response.StatusCode + " " + response.ReasonPhrase + Environment.NewLine;
responseBodyAsText = await response.Content.ReadAsStringAsync();
```

```
            //responseBodyAsText = responseBodyAsText.Replace("<br>",
Environment.NewLine); // Insert new lines
            // textBox2.Text = textBox2.Text + responseBodyAsText;
            string htmlcopy = responseBodyAsText;
            string htmlcopy2 = responseBodyAsText;
            int countOfPages;

            //находим количество страниц в результате поиска
            int rr = htmlcopy2.LastIndexOf("<span class=\"link-text\">");
            if (rr != -1)//если страниц больше одной, то rr больше -1
            {
               htmlcopy2 = htmlcopy2.Substring(rr);
               rr = htmlcopy2.IndexOf('>');
               htmlcopy2 = htmlcopy2.Substring(rr + 1);
               rr = htmlcopy2.IndexOf('<');
               countOfPages = Convert.ToInt32(htmlcopy2.Substring(0, rr));
            }
            else countOfPages = 1;//если страница одна, то rr=-1, так как такого
тэга в коде не будет
                                  //тогда сообщаем, что у нас одна страница


            for (int p = 1; p <= countOfPages; p++)
            {
               //открываем следующую страницу с результатами поиска
               response = await httpClient.GetAsync(summaryStr + "pageNumber="
+ Convert.ToString(p));
               response.EnsureSuccessStatusCode();
               responseBodyAsText = await response.Content.ReadAsStringAsync();
```

```
htmlcopy = responseBodyAsText;

while (responseBodyAsText.IndexOf(substr_reportBox) != -1)
{
   summaryline = "";
   int indexofsubstr_boxIcons = responseBodyAsText.IndexOf(substr_boxIcons);
   int indexofsubstr_reportBox = responseBodyAsText.IndexOf(substr_reportBox);
   summaryline += responseBodyAsText.Substring(indexofsubstr_boxIcons, indexofsubstr_reportBox - indexofsubstr_boxIcons);
   responseBodyAsText = responseBodyAsText.Substring(indexofsubstr_reportBox);
   indexofsubstr_reportBox = responseBodyAsText.IndexOf(substr_div);
   summaryline += responseBodyAsText.Substring(0, indexofsubstr_reportBox);
   responseBodyAsText = responseBodyAsText.Substring(indexofsubstr_reportBox);
   lines[i] = summaryline;

   j = 0;
   while (true)
   {
      int k1 = lines[i].IndexOf("<");
      int k2 = lines[i].IndexOf(">");
      if (k1 < 0 || k2 < 0) break;
      string s = lines[i].Substring(0, k1);
```

```
                    string ms = MyTrim(s);
                    if (ms.Length > 0)
                    {
                        parsinglines[i, j] = ms;
                        j++;
                    }
                    lines[i] = lines[i].Substring(k2 + 1);
                }
                i++;
            }
        }


//========================================================== Разбор полученных данных в массиве ==========================================================
=======
        for (int r = 0; r < 100000; r++)
        {

            if (parsinglines[r, 0] != null)
            {
                if (String.Compare(parsinglines[r, 0], "Не пройден контроль") ==
0)
                    for (int t = 2; t < 1000; t++)
                        parsinglines[r, t - 2] = parsinglines[r, t];
            }
        }
```

```
for (int r = 0; r < 100000; r++)
{
   if (parsinglines[r, 0] != null)
   {
      for (int t = 0; t < 1000; t++)
      {
         parsinglines[r, 1] = parsinglines[r, 1].Replace(" /", "");

         if (String.Compare(parsinglines[r, t], "Начальная цена") == 0)//объединяем два столбца с ценой в один (были раздельно рубли и копейки)
            parsinglines[r, t + 1] = String.Concat(parsinglines[r, t + 1], parsinglines[r, t + 2]);

         if (String.Compare(parsinglines[r, t], "Заказчик:") == 0 || String.Compare(parsinglines[r, t], "Организация, осуществляющая размещение:") == 0)//формируем наименование аукциона в одну ячейку
         {
            string str1 = "";
            int u = t;
            while ((String.Compare(parsinglines[r, u], "Идентификационный код закупки(ИКЗ):") != 0 || String.Compare(parsinglines[r, u], "Размещено:") != 0) && u < 98)
            {
               str1 += parsinglines[r, u + 2];
               u++;
            }
            parsinglines[r, t + 2] = str1;
         }
```

```
                if (String.Compare(parsinglines[r, t], "Размещено:") == 0)
                {
                    parsinglines[r, t - 1] = "";
                    parsinglines[r, t - 2] = "";
                }
            }
        }
    }
    for (int r = 0; r < 100000; r++)
    {
        for (int t = 0; t < 1000; t++)
        {
            if (String.Compare(parsinglines[r, t], "Сведения") == 0)
            {
                parsinglines[r, t] = "";
                parsinglines[r, t + 1] = "";
                parsinglines[r, t + 2] = "";
                parsinglines[r, t + 3] = "";
                parsinglines[r, t + 4] = "";
                parsinglines[r, t + 5] = "";
            }
        }
    }
```

```
    string[,] parsinglines2 = new string[100000, 10];//формируем итоговый массив с данными

    for (int r = 0; r < 100000; r++)
    {
```

```
                if (parsinglines[r, 0] != null)
                {
                    parsinglines2[r, 0] = parsinglines[r, 0];
                    parsinglines2[r, 1] = parsinglines[r, 1];
                    parsinglines2[r, 2] = parsinglines[r, 2];

                    for (int t = 0; t < 1000; t++)
                    {
                        if (parsinglines[r, t] != null)
                        {
                            if (String.Compare(parsinglines[r, t], "Начальная цена") == 0)//переносим цену аукциона
                            {
                                parsinglines2[r, 3] = parsinglines[r, t + 1];//цена
                                parsinglines2[r, 4] = parsinglines[r, t + 3];//валюта
                            }

                            if (parsinglines[r, t].IndexOf("№ ") != -1)
                                parsinglines2[r, 5] = parsinglines[r, t];//номер аукциона

                            if (String.Compare(parsinglines[r, t], "Заказчик:") == 0 || String.Compare(parsinglines[r, t], "Организация, осуществляющая размещение:") == 0)
                            {
                                parsinglines2[r, 6] = parsinglines[r, t + 1];//заказчик
                                parsinglines2[r, 7] = parsinglines[r, t + 2];//наименование
                            }

                            if (String.Compare(parsinglines[r, t], "Размещено:") == 0)
```

```
                    {
                        parsinglines2[r, 8] = parsinglines[r, t + 1];//размещено
                        parsinglines2[r, 9] = parsinglines[r, t + 3];//обновлено
                    }
                }
            }
        }

        // dataGridView1.RowCount = 100000;
        //dataGridView1.ColumnCount = 10;
        for (i = 0; i < 100000; i++)
        {
            if (parsinglines2[i, 0] != null)
            {
                dataGridView1.Rows.Add();
                for (j = 0; j < 10; j++)
                {
                    if (parsinglines2[i, j] != null)
                    {
                        dataGridView1.Rows[i].Cells[j].Value = parsinglines2[i,
j].ToString();
                    }
                }
                this.dataGridView1.Rows[i].Cells[13].Value = "hhhh";
            }
        }
```

```
//=================================== Вывод грида в Excel
==========================================================
==========================================================
//
//// Запустим Excel
//excelapp = new Excel.Application();
//// Сделаем Excel видимым
//excelapp.Visible = true;
//// В книге, которую создадим позже, будет 3 листа
//excelapp.SheetsInNewWorkbook = 3;
//// Создадим книгу
//excelapp.Workbooks.Add(Type.Missing);
//// Получаем набор ссылок на объекты Workbook (на созданные книги)
//excelappworkbooks = excelapp.Workbooks;
////Получаем ссылку на книгу 1 - нумерация от 1
//excelappworkbook = excelappworkbooks[1];
//// Получаем ссылку на рабочие листы книги
//excelsheets = excelappworkbook.Worksheets;
////Получаем ссылку на лист 1
//excelworksheet = (Excel.Worksheet)excelsheets[1];
//// Сделаем первый лист активным
//excelworksheet.Activate();

//string[] c = { "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z",
```

```
            //"AA", "AB", "AC", "AD", "AE", "AF", "AG", "AH", "AI", "AJ",
"AK", "AL", "AM", "AN", "AO", "AP", "AQ", "AR", "AS", "AT", "AU", "AV", "AW",
"AX", "AY", "AZ",
            //"BA", "BB", "BC", "BD", "BE", "BF", "BG", "BH", "BI", "BJ", "BK",
"BL", "BM", "BN", "BO", "BP", "BQ", "BR", "BS", "BT", "BU", "BV", "BW", "BX",
"BY", "BZ",
            //"CA", "CB", "CC", "CD", "CE", "CF", "CG", "CH", "CI", "CJ", "CK",
"CL", "CM", "CN", "CO", "CP", "CQ", "CR", "CS", "CT", "CU", "CV", "CW", "CX",
"CY", "CZ"};
            //// Цикл по строкам таблицы
            //for (int g = 0; g < dataGridView1.RowCount - 1; g++)
            //{
            //    // Цикл по столбцам
            //    for (int q = 0; q < dataGridView1.ColumnCount - 1; q++)
            //    {
            //                    // Вывод в ячейку i+2, j Excel-я содержимого
соответствующей ячейки
            //        // dataGridView1
            //         excelcells = excelworksheet.get_Range(c[q] + Convert.ToString(g
+ 2), c[q] +
            //             Convert.ToString(g + 2));
            //      if(dataGridView1.Rows[g].Cells[q].Value!=null)
            //                                                  excelcells.Value2 =
dataGridView1.Rows[g].Cells[q].Value.ToString();
            //    } // for
            //} // for


            this.button1.Enabled = true;
```

```
}


string MyTrim(string s)
{
   int l1 = -1, l2 = -1;
   for (int i = 0; i < s.Length; i++)
   {
      string c = s.Substring(i, 1);
      if (c != " " && c != "\n" && c != "\t")
      {
         l1 = i;
         break;
      }
   }
   if (l1 == -1) return "";
   for (int i = s.Length - 1; i >= 0; i--)
   {
      string c = s.Substring(i, 1);
      if (c != " " && c != "\n" && c != "\t")
      {
         l2 = i;
         break;
      }
   }
   if (l2 == -1) return "";
   return s.Substring(l1, l2 - l1 + 1);

} // MyTrim
```

```
        private void checkedListBox2_SelectedIndexChanged(object sender, EventArgs e)
        {

        }

        private void FormSearch_Load(object sender, EventArgs e)
        {

            // TODO: данная строка кода позволяет загрузить данные в таблицу "zakupkiDataSet.Marks". При необходимости она может быть перемещена или удалена.
            this.marksTableAdapter.Fill(this.zakupkiDataSet.Marks);
            // TODO: данная строка кода позволяет загрузить данные в таблицу "zakupkiDataSet.Companies". При необходимости она может быть перемещена или удалена.
            this.companiesTableAdapter.Fill(this.zakupkiDataSet.Companies);
            // TODO: данная строка кода позволяет загрузить данные в таблицу "zakupkiDataSet.Notifications". При необходимости она может быть перемещена или удалена.
            this.notificationsTableAdapter.Fill(this.zakupkiDataSet.Notifications);

        }

        private void dataGridView1_CellEndEdit(object sender, DataGridViewCellEventArgs e)
        {
            if (dataGridView1.CurrentRow != null)
```

```
        {
            string vidZak = "";
            string etapZak = "";
            string fz = "";
            string nmc = "";
            string valuta = "";
            string nomIzv = "";
            string zakazchik = "";
            string naimen = "";
            string razmesh = "";
            string obnov = "";
            int id_comp;
            int id_mark;
            string comment = "";

            vidZak                                                    =
Convert.ToString(dataGridView1.CurrentRow.Cells[0].Value);
            etapZak                                                   =
Convert.ToString(dataGridView1.CurrentRow.Cells[1].Value);
            fz = Convert.ToString(dataGridView1.CurrentRow.Cells[2].Value);
            nmc = Convert.ToString(dataGridView1.CurrentRow.Cells[3].Value);
            valuta                                                    =
Convert.ToString(dataGridView1.CurrentRow.Cells[4].Value);
            nomIzv                                                    =
Convert.ToString(dataGridView1.CurrentRow.Cells[5].Value);
            zakazchik                                                 =
Convert.ToString(dataGridView1.CurrentRow.Cells[6].Value);
            naimen                                                    =
Convert.ToString(dataGridView1.CurrentRow.Cells[7].Value);
```

```
                razmesh                                                    =
Convert.ToString(dataGridView1.CurrentRow.Cells[8].Value);
                obnov                                                      =
Convert.ToString(dataGridView1.CurrentRow.Cells[9].Value);

                if (dataGridView1.CurrentRow.Cells[10].Value != null &&
                   dataGridView1.CurrentRow.Cells[11].Value != null &&
                   dataGridView1.CurrentRow.Cells[12].Value != null)
                {
                   id_comp                                                 =
Convert.ToInt32(dataGridView1.CurrentRow.Cells[10].Value);
                   id_mark                                                 =
Convert.ToInt32(dataGridView1.CurrentRow.Cells[11].Value);
                   comment                                                 =
Convert.ToString(dataGridView1.CurrentRow.Cells[12].Value);

                   //// Зададим параметры запроса
                   notificationsTableAdapter.Insert(nomIzv, id_comp, vidZak,
etapZak, fz, nmc, valuta, zakazchik, naimen, razmesh, obnov, id_mark, comment);
                   this.notificationsTableAdapter.Fill(zakupkiDataSet.Notifications);
                   notificationsTableAdapter.Connection.Close();
                }
             }
          }

          private void button2_Click(object sender, EventArgs e)
          {
             FormCompanies f = new FormCompanies();
             f.Show();
```

```
        }

        private void button3_Click(object sender, EventArgs e)
        {
            FormParams f = new FormParams();
            f.Show();
        }
    }
}
```

Код формы «Организации»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Excel = Microsoft.Office.Interop.Excel;

namespace HTTP_
{
    public partial class FormCompanies : Form
    {
        public FormCompanies()
        {
            InitializeComponent();
```

```
        }

        private void FormCompanies_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу "zakupkiDataSet.Notifications". При необходимости она может быть перемещена или удалена.
            this.notificationsTableAdapter.Fill(this.zakupkiDataSet.Notifications);
            // TODO: данная строка кода позволяет загрузить данные в таблицу "zakupkiDat this.dataTable1TableAdapter.Fill(this.zakupkiDataSet1.DataTable1);
            // TODO: данная строка кода позволяет загрузить данные в таблицу "zakupkiDataSet.Companies". При необходимости она может быть перемещена или удалена.
            this.companiesTableAdapter.Fill(this.zakupkiDataSet.Companies);

        }
        public void DelTextBoxes()
        {
            this.textBox1.Text = "";
            this.textBox2.Text = "";
            this.textBox3.Text = "";
            this.textBox4.Text = "";
            this.textBox5.Text = "";
            this.textBox9.Text = "";
            this.textBox10.Text = "";
            this.textBox11.Text = "";
            this.textBox12.Text = "";
            this.textBox6.Text = "";
            this.textBox7.Text = "";
```

```
            this.textBox8.Text = "";
        }

        private void companiesBindingSource_CurrentChanged(object sender,
EventArgs e)
        {

            DataRowView drv = companiesBindingSource.Current as
DataRowView;

            int id_company = 0;
            string shortName = "";
            string fullName = "";//год начала учебного плана
            string INN = "";//гд окончания учебного плана
            string KPP = "";//количество семестров в ученом плане
            string urAdr = "";
            string postAdr = "";
            string factAdr = "";
            string director = "";
            string contName = "";
            string phoneNumb = "";
            string eMail = "";

            if (drv != null)
            {
                id_company = Convert.ToInt32(drv.Row["id_company"]);
                shortName = Convert.ToString(drv.Row["shortName"]);
                fullName = Convert.ToString(drv.Row["fullName"]);
                INN = Convert.ToString(drv.Row["INN"]);
```

```
                KPP = Convert.ToString(drv.Row["KPP"]);
                urAdr = Convert.ToString(drv.Row["urAdr"]);
                postAdr = Convert.ToString(drv.Row["postAdr"]);
                factAdr = Convert.ToString(drv.Row["factAdr"]);
                director = Convert.ToString(drv.Row["director"]);
                contName = Convert.ToString(drv.Row["contName"]);
                phoneNumb = Convert.ToString(drv.Row["phoneNumb"]);
                eMail = Convert.ToString(drv.Row["eMail"]);

                //dataTable2TableAdapter.Fill(_ЭиУ_235DataSet.DataTable2,
idCurric);

                this.textBox1.Text = Convert.ToString(id_company);
                this.textBox2.Text = Convert.ToString(shortName);
                this.textBox3.Text = Convert.ToString(fullName);
                this.textBox4.Text = Convert.ToString(INN);
                this.textBox5.Text = Convert.ToString(KPP);
                this.textBox9.Text = Convert.ToString(director);
                this.textBox10.Text = Convert.ToString(contName);
                this.textBox11.Text = Convert.ToString(phoneNumb);
                this.textBox12.Text = Convert.ToString(eMail);
                this.textBox6.Text = Convert.ToString(urAdr);
                this.textBox7.Text = Convert.ToString(postAdr);
                this.textBox8.Text = Convert.ToString(factAdr);

                this.dataTable1TableAdapter.Fill(zakupkiDataSet.DataTable1,
id_company);
            }
        }
```

```
        private void button1_Click(object sender, EventArgs e)
        {
            if (this.textBox2.Text != "" && this.textBox3.Text != "" && this.textBox4.Text != ""
                && this.textBox5.Text != "" && this.textBox6.Text != "" && this.textBox7.Text != "" && this.textBox8.Text != ""
                && this.textBox9.Text != "" && this.textBox10.Text != "" && this.textBox11.Text != "" && this.textBox12.Text != "")
            {
                // Откроем подключение адаптера к базе данных
                companiesTableAdapter.Connection.Open();
                // Зададим параметры запроса
                companiesTableAdapter.Insert(this.textBox2.Text, this.textBox3.Text, this.textBox4.Text,
                this.textBox5.Text, this.textBox6.Text, this.textBox7.Text, this.textBox8.Text,
                this.textBox9.Text, this.textBox10.Text, this.textBox11.Text, this.textBox12.Text);
                this.companiesTableAdapter.Fill(zakupkiDataSet.Companies);

                DelTextBoxes();

            }
            else
            {
                MessageBox.Show("Заполните все поля формы");
            }
        }
```

```
        private void button4_Click(object sender, EventArgs e)//кнопка очищения всех полей
        {
            DelTextBoxes();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            DataRowView drv = companiesBindingSource.Current as DataRowView;
            if (drv != null)
            {
                if (this.textBox2.Text != "" && this.textBox3.Text != "" && this.textBox4.Text != ""
                && this.textBox5.Text != "" && this.textBox6.Text != "" && this.textBox7.Text != "" && this.textBox8.Text != ""
                && this.textBox9.Text != "" && this.textBox10.Text != "" && this.textBox11.Text != "" && this.textBox12.Text != "")
                {
                    int id_company = 0;
                    string shortName = "";
                    string fullName = "";
                    string INN = "";
                    string KPP = "";
                    string urAdr = "";
                    string postAdr = "";
                    string factAdr = "";
                    string director = "";
```

```
                string contName = "";
                string phoneNumb = "";
                string eMail = "";


                id_company = Convert.ToInt32(drv.Row["id_company"]);
                shortName = Convert.ToString(drv.Row["shortName"]);
                fullName = Convert.ToString(drv.Row["fullName"]);
                INN = Convert.ToString(drv.Row["INN"]);
                KPP = Convert.ToString(drv.Row["KPP"]);
                urAdr = Convert.ToString(drv.Row["urAdr"]);
                postAdr = Convert.ToString(drv.Row["postAdr"]);
                factAdr = Convert.ToString(drv.Row["factAdr"]);
                director = Convert.ToString(drv.Row["director"]);
                contName = Convert.ToString(drv.Row["contName"]);
                phoneNumb = Convert.ToString(drv.Row["phoneNumb"]);
                eMail = Convert.ToString(drv.Row["eMail"]);

                // Откроем подключение адаптера к базе данных
                companiesTableAdapter.Connection.Open();
                // Зададим параметры запроса
                companiesTableAdapter.Update(this.textBox2.Text,
this.textBox3.Text, this.textBox4.Text,
                this.textBox5.Text, this.textBox6.Text, this.textBox7.Text,
this.textBox8.Text,
                this.textBox9.Text, this.textBox10.Text, this.textBox11.Text,
this.textBox12.Text, id_company,
                shortName, fullName, INN, KPP, urAdr, postAdr, factAdr, director,
contName, phoneNumb, eMail);
```

```
            this.companiesTableAdapter.Fill(zakupkiDataSet.Companies);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@shortName"].
            //    Value = Convert.ToString(this.textBox2.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@fullName"].
            //    Value = Convert.ToString(this.textBox3.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@INN"].
            //    Value = Convert.ToString(this.textBox4.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@KPP"].
            //    Value = Convert.ToString(this.textBox5.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@urAdr"].
            //    Value = Convert.ToString(this.textBox6.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@postAdr"].
            //    Value = Convert.ToString(this.textBox7.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@factAdr"].
            //    Value = Convert.ToString(this.textBox8.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@director"].
            //    Value = Convert.ToString(this.textBox9.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@contName"].
            //    Value = Convert.ToString(this.textBox10.Text);
```

```
//companiesTableAdapter.Adapter.InsertCommand.Parameters["@phoneNumb"].
                    //    Value = Convert.ToString(this.textBox11.Text);

//companiesTableAdapter.Adapter.InsertCommand.Parameters["@eMail"].
                    //    Value = Convert.ToString(this.textBox12.Text);



                    DelTextBoxes();
                    companiesTableAdapter.Connection.Close();
                }
            }
            else
            {
                MessageBox.Show("Заполните все поля формы");
            }
        }

        private void button3_Click(object sender, EventArgs e)
        {
            DataRowView drv = companiesBindingSource.Current as
DataRowView;
            if (drv != null)
            {
                if (this.textBox2.Text != "" && this.textBox3.Text != "" &&
this.textBox4.Text != ""
                && this.textBox5.Text != "" && this.textBox6.Text != "" &&
this.textBox7.Text != "" && this.textBox8.Text != ""
```

```
                && this.textBox9.Text != "" && this.textBox10.Text != "" && this.textBox11.Text != "" && this.textBox12.Text != "")
                {
                    int id_company = 0;
                    string shortName = "";
                    string fullName = "";//год начала учебного плана
                    string INN = "";//гд окончания учебного плана
                    string KPP = "";//количество семестров в ученом плане
                    string urAdr = "";
                    string postAdr = "";
                    string factAdr = "";
                    string director = "";
                    string contName = "";
                    string phoneNumb = "";
                    string eMail = "";

                    id_company = Convert.ToInt32(drv.Row["id_company"]);
                    shortName = Convert.ToString(drv.Row["shortName"]);
                    fullName = Convert.ToString(drv.Row["fullName"]);
                    INN = Convert.ToString(drv.Row["INN"]);
                    KPP = Convert.ToString(drv.Row["KPP"]);
                    urAdr = Convert.ToString(drv.Row["urAdr"]);
                    postAdr = Convert.ToString(drv.Row["postAdr"]);
                    factAdr = Convert.ToString(drv.Row["factAdr"]);
                    director = Convert.ToString(drv.Row["director"]);
                    contName = Convert.ToString(drv.Row["contName"]);
                    phoneNumb = Convert.ToString(drv.Row["phoneNumb"]);
                    eMail = Convert.ToString(drv.Row["eMail"]);
                    companiesTableAdapter.Connection.Open();
```

```
                    // Зададим параметры запроса
                    companiesTableAdapter.Delete(id_company,
                    shortName, fullName, INN, KPP, urAdr, postAdr, factAdr, director,
contName,
                    phoneNumb, eMail);
                    this.companiesTableAdapter.Fill(zakupkiDataSet.Companies);
                    companiesTableAdapter.Connection.Close();
                    DelTextBoxes();

                }
            }
        }

        private void button6_Click(object sender, EventArgs e)
        {

        }
    }
}
```

Код формы «Настройка»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
using System.Windows.Forms;

namespace HTTP_
{
    public partial class FormParams : Form
    {
        public FormParams()
        {
            InitializeComponent();
        }

        private void FormParams_Load(object sender, EventArgs e)
        {
            // TODO: данная строка кода позволяет загрузить данные в таблицу "zakupkiDataSet.Services". При необходимости она может быть перемещена или удалена.
            this.servicesTableAdapter.Fill(this.zakupkiDataSet.Services);
            // TODO: данная строка кода позволяет загрузить данные в таблицу "zakupkiDataSet.Marks". При необходимости она может быть перемещена или удалена.
            this.marksTableAdapter.Fill(this.zakupkiDataSet.Marks);

        }

        private void marksBindingSource_CurrentChanged(object sender, EventArgs e)
        {
            DataRowView drv = marksBindingSource.Current as DataRowView;
```

```
            int id_mark = 0;
            string mark = "";


            if (drv != null)
            {
                id_mark = Convert.ToInt32(drv.Row["id_mark"]);
                mark = Convert.ToString(drv.Row["mark"]);

                this.textBox1.Text = mark;
            }
        }

        private void servicesBindingSource_CurrentChanged(object sender,
EventArgs e)
        {
            DataRowView drv = servicesBindingSource.Current as DataRowView;

            int id_service = 0;
            string name = "";
            string price = "";

            if (drv != null)
            {
                id_service = Convert.ToInt32(drv.Row["id_service"]);
                name = Convert.ToString(drv.Row["name"]);
                price = Convert.ToString(drv.Row["price"]);
                this.textBox2.Text = name;
                this.textBox3.Text = price;
```

```
            }
        }
    }
}
```