**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего образования**

**«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ТЕХНОЛОГИЙ И УПРАВЛЕНИЯ ИМ. К.Г.РАЗУМОВСКОГО (ПКУ)»**

**Университетский колледж информационных технологий**

КУРСОВОЙ ПРОЕКТ

**ПМ.02:МДК.02.01. «Разработка, внедрение и адаптация программного обеспечения отраслевой направленности»**

**Специальность 09.02.05 Прикладная информатика (по отраслям)**

Тема: «Разработка информационной системы театр»

|  |  |  |
| --- | --- | --- |
|  |  | **Группа И-405к** |
| **Студент** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (личная подпись)** | **Васильков А.А.** |
| **Руководитель работы** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (личная подпись)** | **Глускер А.И.** |

**Москва**

**2017**

**Содержание**

[**ВВЕДЕНИЕ 3**](#_Toc475379281)

[**ГЛАВА 1. ОСНОВНАЯ ЧАСТЬ 4**](#_Toc475379282)

[**1.1 Введение в предметную область 4**](#_Toc475379283)

[**1.2 Анализ бизнес-процессов 7**](#_Toc475379284)

[**1.3 Техническое задание 9**](#_Toc475379285)

[**1.4 Проектирование базы данных 10**](#_Toc475379286)

[**1.5 Разработка пользовательского интерфейса 26**](#_Toc475379287)

[**1.6 Разработка программного обеспечения 29**](#_Toc475379288)

[**1.7 Разработка плана тестирования программного обеспечения 30**](#_Toc475379289)

[**1.8 Изменение и контроль характеристик программного обеспечения 36**](#_Toc475379290)

[**ЗАКЛЮЧЕНИЕ 37**](#_Toc475379291)

[**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 38**](#_Toc475379292)

[**Приложение А. Словарь данных 39**](#_Toc475379293)

[**Приложение Б Листинг программы 42**](#_Toc475379294)

[**Приложение В Руководство по установке 50**](#_Toc475379295)

# ВВЕДЕНИЕ

Компьютерные технологии стали неотъемлемой частью жизнедеятельности современного человека. Компьютерная техника используется человеком в повседневной жизни для учебы, отдыха и для осуществления профессиональной деятельности.

Для более полного использования вычислительных мощностей современного компьютера создаются разнообразные программные продукты, каждый из которых предназначен для решения определенного круга задач.

Целью данной курсовой работы является обеспечение покупки билета на определённый сеанс через ИС театр для пользователей.

Для достижения поставленной цели необходимо решить следующие задачи:

* Провести анализ деятельности сервисного центра
* На основе анализа составить концептуальную модель и обозначить критерии оценки качества разрабатываемого программного средства
* На основе концептуальной модели составить функциональные и структурные модели
* По результатам моделирования спроектировать информационную систему
* Основываясь на проекте системы разработать готовую систему
* Провести тестирование системы на соответствие критериям качества

# ГЛАВА 1. ОСНОВНАЯ ЧАСТЬ

* 1. **Введение в предметную область**

Главной целью данного курсового проекта является создание информационной системы для театра. Первым этапом для достижения цели является задача по проведению анализа предметной области. Результат данного этапа – формирование общих представлений о деятельности театра, выделение основных производственных процессов для выяснения требований к функциональному наполнению проектируемой информационной системы.

Основными видами деятельности театра являются создание и показ спектаклей, организация гастролей, концертов, проведение творческих вечеров, фестивалей и конкурсов, реализация билетов на указанные мероприятия.

**Анализ существующих систем управления предприятий по ремонту техники**

**Билетная система управления "SaleTicket"**

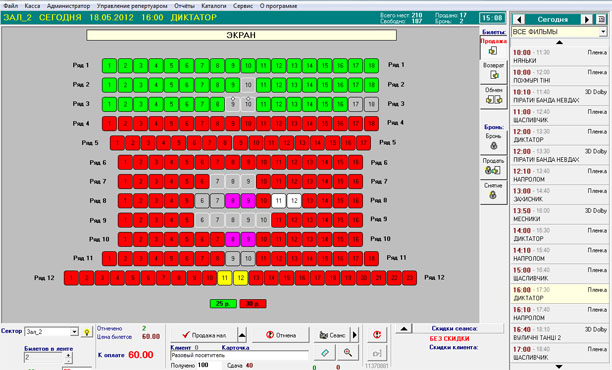


Рис.1 SaleTicket

Плюсы бронирования места этой системы:

* Подсветка места по его стоимости
* Вместимость

Минусы бронирования места:

* Очень простой стиль
* Подсветка места(фиолетовый, жёлтый и белый) не известно что с ним

**Бронирование билетов в Афиша**

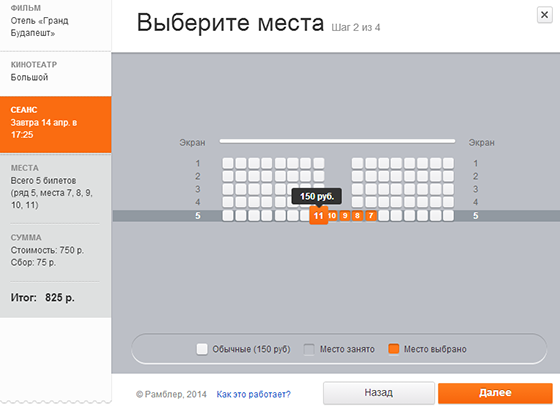
****

Рис.2 Афиша

Плюсы бронирования билета

* Имеется информация о местах
* Наведя на определённое место, указывается цена

Минусы бронирования места

* Мало мест
* Не подписаны места
* Нет информации о сеансе
  1. **Анализ бизнес-процессов**

Методологию IDEF0 придумали американские военные с целью успешного тиражирования бизнес-процессов предприятий аэрокосмической промышленности. В свое время американские военные столкнулись со следующей проблемой. При проектировании заводов было замечено, что каждый раз приходится заново проделывать один и тот же шаг - проектировать одинаковые подсистемы управления, на что уходило дополнительное время и ресурсы. После этого было предложено разработать язык или чертеж, с помощью которого можно было бы описать типовые подсистемы управления и при строительстве нового завода использовать наработанные схемы. Язык, который был придуман и использован для этих целей, лег в основу методологии описания бизнес-процессов IDEF0.

Данный стандарт описания бизнес-процессов предлагает показывать не просто входы и выходы, он предлагает ввести три типа входов. Первый тип входов назвали так же входом, а два других входа назвали управлением и механизмами.

В стандарте IDEF0 c помощью входа показывают объекты – информационные и материальные потоки, которые преобразуются в бизнес-процессе. С помощью управления показывают объекты – материальные и информационные потоки, которые не преобразуются в процессе, но нужны для его выполнения. С помощью механизмов стали показывать механизмы, при помощи которых бизнес-процесс реализуется: технические средства, люди, информационные системы и т.д. Выход бизнес-процесса, описанного в стандарте IDEF0 – это результат выполнения данного бизнес-процесса.

На первом этапе построения IDEF0 модели система представляется как единое целое (рисунок 3.1). Основная цель данного этапа – выявить входные и выходные данные, а также механизмы и правила для данной системы.

Входными данными для системы являются заявки клиентов, выходными: квитанция о принятии в ремонт, выполненный заказ, чек об оплате, акт выполненных работ.

Правила, которые регулируют деятельности сервисного центра – Устав предприятия и законы Российской Федерации.

Механизмы, или исполнители, это инженер сервисного центра и менеджер по работе с клиентами.

Рисунок 3.1 – Контекстная диаграмма

На следующем этапе построения модели IDEF0 система декомпозируется (разбивается, или детализируется) на составляющие её бизнес-процессы (рисунок 3.2). Система состоит из трех основных бизнес-процессов: оформление заказа клиента, проведение ремонта и завершение сделки.

Рисунок 3.2 – Диаграмма декомпозиции

* 1. **Техническое задание**

# Общие сведения

## **1.1 Полное наименование системы.**

Полное наименование системы – информационная система предприятия по ремонту   
сотовых телефонов

Краткое наименования системы – ИС «Ремонт» В дальнейшем просто – "Система".

## **1.2 Наименование заказчика.**

Заказчик – Университетский колледж информационных технологий ФГБОУ ВО «МГУТУ им.К.Г.Разумовского (ПКУ)»

## **1.3 Перечень документов на основании которых создается система.**

Основанием для разработки является техническое задание на курсовой проект, выданное 14 ноября 2016 г.

Разработка Технического задания проводилась с использованием следующих стандартов:

* ГОСТ 34.601-90 Автоматизированные системы. Стадии создания;
* ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы;
* ГОСТ 34.201-89 Виды, комплектность и обозначение документов при создании автоматизированных систем (частичное использование);
* ГОСТ 24.104-85 "Автоматизированные системы управления. Общие требования.

## **1.4 Сроки выполнения работ.**

Начало выполнения работ – ноябрь 2016 г.

Окончание выполнение работ – февраль 2017 г.

## **1.5 Назначение системы.**

Основным функциональным назначением системы является информатизация и автоматизация некоторых процессов средне специальных образовательных организаций.

Система должна обеспечить:

Сбор, обработку, хранение информации получаемой от сотрудников магазина;

## **1.6 Цели создания системы.**

Основной целью создания системы является разработка свободного отечественного программного обеспечения выполняющее свои функции и упрощающие выполнение привычных операций.

Внедрение Системы позволит:

Увеличить полезное время работы сотрудников;

Отказаться на некоторых этапах от бумажных носителей;

Создать единую базу магазина с возможностью удобного получения информации.

## **1.7 Краткие сведения объекта автоматизации.**

Объектом автоматизации является сервис ремонта мобильных телефонов.

# Требования к системе

## **2.1 Требование к составу системы.**

Разработка Системы должна быть основана на оптимальном выборе оборудования с использованием современных технических средств, имеющих необходимые сертификаты соответствия.

Разрабатываемая система должна соответствовать ГОСТ 24.104-85 ЕСС АСУ "Автоматизированные системы управления. Общие требования" с учетом требований, изложенных в данном разделе.

Система должна состоять из следующих подсистем:

* Подсистема хранения информации;
* Подсистема ввода информации;
* Подсистема вывода информации;

## **2.2 Требования к автоматизированной системе.**

Построенная Система должна обеспечивать следующие технологические процессы:

* Осуществлять помощь в поиске;
* Вести журнал;
* Вести учет работников;

## **2.3 Технические параметры системы.**

Система должна иметь возможность функционирования не менее 8 часов в день.

## **2.4 Требования к режимам функционирования.**

Система должна иметь возможность функционирования непрерывно и круглосуточно.

## **2.5 Требования к защите информации от несанкционированного доступа.**

Система должна позволять разграничивать права доступа к данным, различных категорий пользователей (обычный пользователь и администратор)

Система должна быть ориентирована на работу в реальном времени и обеспечивать выполнение всех функций с заданной периодичностью и точно в назначенный срок.

Должна быть обеспечена надежная защита Системы:

от несанкционированного доступа (через пару логин/пароль);

от разрушения или остановки работы программного обеспечения в результате некорректных действий пользователя (неправильный ввод данных или преднамеренное выключение компьютера);

от проникновения в систему вирусов.

## **2.6 Требования к патентной чистоте**

Патентная чистота системы должна быть обеспечена в отношении следующих стран: Российская Федерация.

Внедряемое программно-аппаратное обеспечение должно быть лицензировано. Состав лицензий на программное обеспечение должен определяться требованиями производителей программного обеспечения.

## **2.7 Требования к надежности.**

Надежность должна полностью соответствовать требованиям, предъявляемым к системам, функционирующим в режиме 24 часа в сутки 7 дней в неделю.

# Требования к видам обеспечения.

## **3.1 Программное обеспечение.**

Программное Обеспечение (ПО) Системы должно состоять из:

* Общесистемного ПО.
* Прикладного ПО, разрабатываемого исполнителем с целью реализации функциональных возможностей Системы.

Общесистемное ПО должно отвечать требованиям по стандартизации и унификации, а также, удовлетворять требованиям, перечисленным в пункте 2.6.

Общесистемное ПО должно включать в себя следующие компоненты:

Операционную систему;

Систему Управления Базами Данных;

Стандартные средства работы с Интернет;

ПО обеспечения информационной безопасности.

В случае построения Заказчиком Системы на основе платформы Microsoft Windows, возможна установка общесистемного ПО Заказчиком.

Прикладное ПО Системы должно обеспечивать реализацию перечисленных в данном ТЗ функций, а также выполнение операций конфигурирования, программирования, управления базами данных и документирования. Прикладное ПО Системы должно обеспечить реализацию требуемых алгоритмов контроля, защиты, отображения информации, сигнализации и архивирования данных.

## **3.2 Техническое обеспечение.**

Структура и мощность оборудования Системы должны соответствовать следующим характеристикам:

Процессор Intel Core 2 DUO или новее;

ОЗУ 1 Гб или больше;

ОС Windows 7 или новее;

Не менее 1 Гб свободного места на ЖД;

## **3.3 Лингвистическое обеспечение.**

Вся представленная на экранах мониторов и текстовая информация для пользователей, как-то:

Информация в всплывающих диалоговых окнах;

Названия полей в меню и т.д.,

должна быть на русском языке.

Исключением, по взаимному согласию между Исполнителем и Заказчиком могут быть коды ошибок, служебные сообщения.

Вместе с Системой, Заказчику должна быть передана полная документация по описанию построенной системы, инструкциям пользования Системой для оператора, администратора, конфигуратора. Все инструкции и руководства для пользователей и обслуживающего персонала Системы должны быть представлены на русском языке.

Диалог пользователя с Системой должен быть максимально приближен к профессиональной терминологии. Диалог Системы с пользователем должен быть краток, понятен, исключать двойного толкования. Диалог Системы с пользователем должен быть направлен на обеспечение безошибочных действий и комфортных условий работы пользователя.

В процессе разработки технического проекта должен быть создан терминологический словарь Системы с целью более полного понимания использующейся информации.

Лингвистическое обеспечение системы должно быть отражено в документации (инструкциях, описаниях, руководствах) организационного обеспечения системы в виде правил общения пользователей с техническими средствами Системы во всех режимах функционирования Системы.

## **3.4 Информационное обеспечение.**

Информационное обеспечение должно отвечать следующим требованиям:

Для удобства работы с большими объемами разнообразной информации, информационное обеспечение Системы должно быть структурировано, и иметь иерархическую организацию;

Структура информационного обеспечения Системы разрабатывается в соответствии со стандартной структурой базы данных;

Система должна корректно работать со всеми информационными данными, предусмотренными структурой Системы;

Конфигурационные файлы Системы должны храниться на дублирующих носителях и обновляться при внесении изменений в Систему.

# Требования к составу и содержанию работ по созданию системы.

## **4.1 Стадии выполнения работ по созданию Системы.**

Работы по созданию системы выполняются в соответствии с   
ГОСТ 34.601-90, в 3 стадии:

* Технический проект;
* Рабочая документация;
* Ввод в эксплуатацию.

На стадии «Технический проект» должна быть осуществлена разработка проектных решений по Системе в целом и ее частям, включая информационную и функциональные модели деятельности, документации на Систему и ее частей, разработка и оформление документации на поставку изделий для комплектования системы.

На стадии «Рабочая документация» должна быть осуществлена разработка рабочей документации на Систему, выбраны производители оборудования и общесистемного ПО, выполнена разработка и адаптация программ, включая разработку базы данных (БД) и пользовательских интерфейсов.

На стадии «Ввод в эксплуатацию» должны быть проведены работы по подготовке объекта автоматизации к вводу Системы в действие, подготовке персонала, комплектации системы поставляемыми изделиями (в том числе программными и техническими средствами), пусконаладочные работы, предварительные испытаний, опытная эксплуатация и приемочные испытания.

По завершении стадии «Ввод в эксплуатацию» должна начаться стадия сопровождения, требования и условия которой должны быть определены отдельным договором.

Стадии «Технический проект» и «Рабочая документация» выполняются раздельно.

* 1. **Проектирование базы данных**

**Словесное описание модели системы**

Целью Театра является содействие процессу формирования общества с высокими моральными, нравственными, гражданскими и духовными качествами.

Задачами Театра являются:

• пропаганда высоких моральных, нравственных, гражданских и духовных идей и ценностей средствами театра;

• формирование и удовлетворение духовных потребностей зрителей в сценическом искусстве;

• развитие, пропаганда и популяризация всех видов театрального искусства и достижений театральной сцены;

• создание произведений театрального искусства;

• создание условий для роста профессионального мастерства деятелей театра;

• привлечение молодёжи к культуре и искусству;

• содействие развитию театрального искусства в регионах России;

• обеспечение информацией о мировой культурной жизни;

• объединение творческих работников;

• оказание научно-методической и практической помощи другим учреждениям и организациям по развитию культуры и искусства;

• формирование и развитие международных контактов между деятелями искусства и культуры разных стран;

• взаимодействие с государственными органами, учреждениями культуры, образовательными учреждениями, общественными организациями с целью создания благоприятных условий для развития детских и юношеских театров и других творческих коллективов.

• участие в реализации программ в области культуры, спорта, охраны здоровья, трудоустройства;

**Описание функций**

Основные функции, которые выполняет театр

* Работа с клиентами
* Продажа билетов
* Работа с персоналом

Работа с клиентами – к этой группе относится вся деятельность менеджера по работе с клиентами, основное направление которой – консультация клиентов по всем вопросам, непосредственно связанными с заказом билетов.

Продажа билетов – в этой группе производится проверка выбранного места в зале, уточняется сеанс, время и сумма по базе данных.

Работа с персоналом – в основном уклон идёт на работу с актёрами тетра и музыкантами. Так же и ведётся учёт работников вне зала, кассиры, уборщики, гардеробщики и охрана.

**Описание требований и ограничений к моделируемой системе**

Правила деятельности театра накладывают некоторые ограничения:

- при выборе определённого места клиентом, повторного выбора того же места быть не может, сделано это для того что бы не было ошибок в базе, так же это будет видно при выборе мест, клиент не сможет выбрать то место которое выбрали до него оно будет подсвечено менее слабо.

-так же клиент который будет взаимодействовать с программой не сможет закрыть её только определённые работники с определённой должностью смогут её закрыть и запустить при надобности.

**Определение критериев качества программного средства**

Разрабатываемое программное средство должно обладать следующими функциями:

* Хранение информации
* Ввод, редактирование и удаление информации
* Формирование документов на выходе
* Проверка корректности вводимых данных

Кроме того, интерфейс программы и все его составляющие должны быть рассчитаны на пользователей, не имеющих специального образования для работы с компьютером:

* Название всех полей для ввода, кнопок и других элементов должны иметь осмысленные названия на русском языке
* Все сообщения о некорректном вводе и возникающих ошибках должны быть на русском языке
* Элементы интерфейса должны повторять расположение и функции элементов интерфейса операционной систем Windows (управление окнами, расположение кнопок и т.д.)

Установка системы должна заключатся в простом копировании файлов со съемного носителя, на жесткий диск компьютера, в случае необходимости установки дополнительного программного обеспечения – должно выводится соответствующее сообщение при первом запуске системы. В случае возникновения нештатных ситуаций (отключение электричества, аппаратные неисправности компьютера и т.д.) восстановление данных должно заключаться в замене файла базы данных из резервной копии.

Обозначенные выше требования к функциональности, интерфейсу и надежности информационной системы являются критериями оценки качества созданного программного продукта.

**Разработка структуры системы и подсистем**

Проектируемая система состоит из двух подсистем: подсистема работы со справочниками и подсистема работы с заказами клиентов.

Справочники содержат в себе упорядоченную информацию, разбитую на отдельные категории, в соответствии с их назначением в рамках исследуемой предметной области. Справочники используются для быстрого заполнения повторяющих значений в выходных документах. Работа со справочниками заключается в поддержании информации, хранящейся в них, в актуальном состоянии, поэтому система должна реализовывать следующие функции работы с подсистемой справочников:

* Добавление новых записей
* Изменение записей, добавленных ранее
* Удаление устаревших и ненужных данных

**Разработка архитектуры системы**

В настоящее время, технология создания программных продуктов не дает однозначного определения понятия архитектура. Среди наиболее часто встречающихся определений можно выделить:

* Архитектурой программного обеспечения называют совокупность базовых концепций (принципов) его построения.
* Архитектура ПС — это его строение, как оно видно (или должно быть видно) извне его, т. е. представление ПС как системы, состоящей из некоторой совокупности взаимодействующих подсистем.
* Архитектура программы или компьютерной системы — это структура или структуры системы, которые включают элементы программы, видимые извне свойства этих элементов и связи между ними.
* Архитектура — это структура организации и связанное с ней поведение системы. Архитектуру можно рекурсивно разобрать на части, взаимодействующие посредством интерфейсов, связи, которые соединяют части, и условия сборки частей. Части, которые взаимодействуют через интерфейсы, включают классы, компоненты и подсистемы.
* Архитектура программного обеспечения системы или набора систем состоит из всех важных проектных решений по поводу структур программы и взаимодействий между этими структурами, которые составляют системы. Проектные решения обеспечивают желаемый набор свойств, которые должна поддерживать система, чтобы быть успешной. Проектные решения предоставляют концептуальную основу для разработки системы, ее поддержки и обслуживания.

Как видно из приведенных выше определений, выбор архитектуры разрабатываемой программы определяется требованиями к функциональности программы, условиям её эксплуатации, а также задачами, которые должно решать создаваемое программное средство.

С точки зрения количества пользователей, одновременно работающих с одной копией запушенного программного средства, различают:

* однопользовательскую архитектуру;
* многопользовательскую (сетевую) архитектуру.

В свою очередь, программные продукты однопользовательской архитектуры подразделяют на:

* программы;
* пакеты программ;
* программные комплексы;
* программные системы.

Самым простым видом архитектуры является программа. Это упорядоченная последовательность формализованных инструкций, выполнение которых решает какую-то определенную, обычно небольшую, задачу.

Пакеты программ – это несколько отдельных программ, которые решают задачи определенной прикладной области. Чаще всего, пакеты программ реализуются как набор отдельных программ, для каждой из которых характерны свои входные и выходные данные, т.е. программы пакета являются независимыми друг от друга и связаны только смысловой принадлежностью к некоторой прикладной области.

Программные комплексы представляют собой совокупность программ, совместно обеспечивающих решение небольшого класса сложных задач одной прикладной области. При этом для выполнения некоторой задачи программой-диспетчером последовательно вызываются несколько программ из программного комплекса. Поскольку несколько программ для решения одной задачи работают с одними и теми же исходными данными и промежуточными результатами, желательно хранить эти данные и результаты вызовов в оперативной памяти или в файлах в пределах одного пользовательского проекта.

Программы комплекса могут компилироваться как самостоятельные единицы или совместно. Программа-диспетчер может иметь примитивный интерфейс и простую справочную систему

Программные системы представляют собой организованную совокупность программ (подсистем), позволяющую решать широкий класс задач из некоторой прикладной области. Программы, входящие в программную систему, взаимодействуют через общие данные.

Разрабатываемое программное средство будет однопользовательским, то есть с одной запущенной копией программы будет одновременно работать только один пользователь. По сложности архитектурного строения информационная система относится к первой, самой простой, категории – к категории программ, т.к. она создается для решения определенного круга задач конкретной предметной области.

**Проектирование базы данных системы**

Для хранения данных, используемый в разрабатываемой информационной системе будет использоваться технология баз данных.

Базы данных представляют собой упорядоченной хранилище информации. В реляционных базах данных информация представляется в виде таблиц, где каждая таблица соответствует определенной сущности заданной предметной области. Например, для театра характерно 3 основных сущности:

* Время
* Работники
* Спектакли

Множество экземпляров одной сущности представляется в табличном виде, где каждая строка соответствует одному экземпляру сущности (запись в таблицы БД), а каждый столбец – характеристике этого экземпляра (атрибут сущности или поле таблицы БД). Соответственно на пересечении строки и столбца (ячейки таблицы) находятся значения атрибутов для каждого экземпляра сущности.

Для управления базой данных используются специальные программные средства, которые называются системами управления базами данных (СУБД). В данном курсовом проекте в качестве СУБД используется Microsoft Access.

Access – это, прежде всего, система управления базами данных. Как и другие продукты этой категории, она предназначена для хранения и поиска данных, представления информации в удобном виде и автоматизации часто повторяющихся операций. С помощью Access можно разрабатывать простые и удобные формы для ввода данных, а также осуществлять обработку данных и выдачу сложных отчетов.

Access – мощное приложение к Windows; впервые производительность СУБД органично сочетается с теми удобствами, которые имеются в распоряжении пользователей Microsoft Windows. Поскольку оба эти продукта – детища компании Microsoft, они прекрасно взаимодействуют между собой. Система Access работает под управлением Windows 95, Windows 98 или Windows NT, так что при работе с ней пользователю доступны все преимущества Windows. Можно вырезать, копировать и вставлять данные из любого приложения Windows в Access и наоборот; можно создавать проект формы в Access и сохранить его как отчет.

В Access в полной мере реализовано управление реляционными базами данных. Система поддерживает первичные и внешние ключи и обеспечивает целостность данных на уровне ядра (что предотвращает несовместимые операции обновления или удаления данных). Кроме того, таблицы в Access снабжены средствами проверки допустимости данных, предотвращающими некорректный ввод, независимо от того, как он осуществляется, а каждое поле таблицы имеет свой формат и стандартные описания, что существенно облегчает ввод данных. Access поддерживает все необходимые типы полей, в том числе текстовый, числовой, счетчик, денежный, дата/время, МЕМО, логический, гиперссылка и поля объектов OLE. Если в процессе специальной обработки в полях не оказывается никаких значений, система обеспечивает полную поддержку пустых значений.

Рассмотрим структуру разработанной базы данных. БД состоит из 3 таблиц, в таблице ниже представлено описание структуры всех таблиц базы данных.

Таблица 1.5.1 Структура БД

Схема данных, на которой изображены все таблицы и связи между ними, изображена на рисунке 4.2.

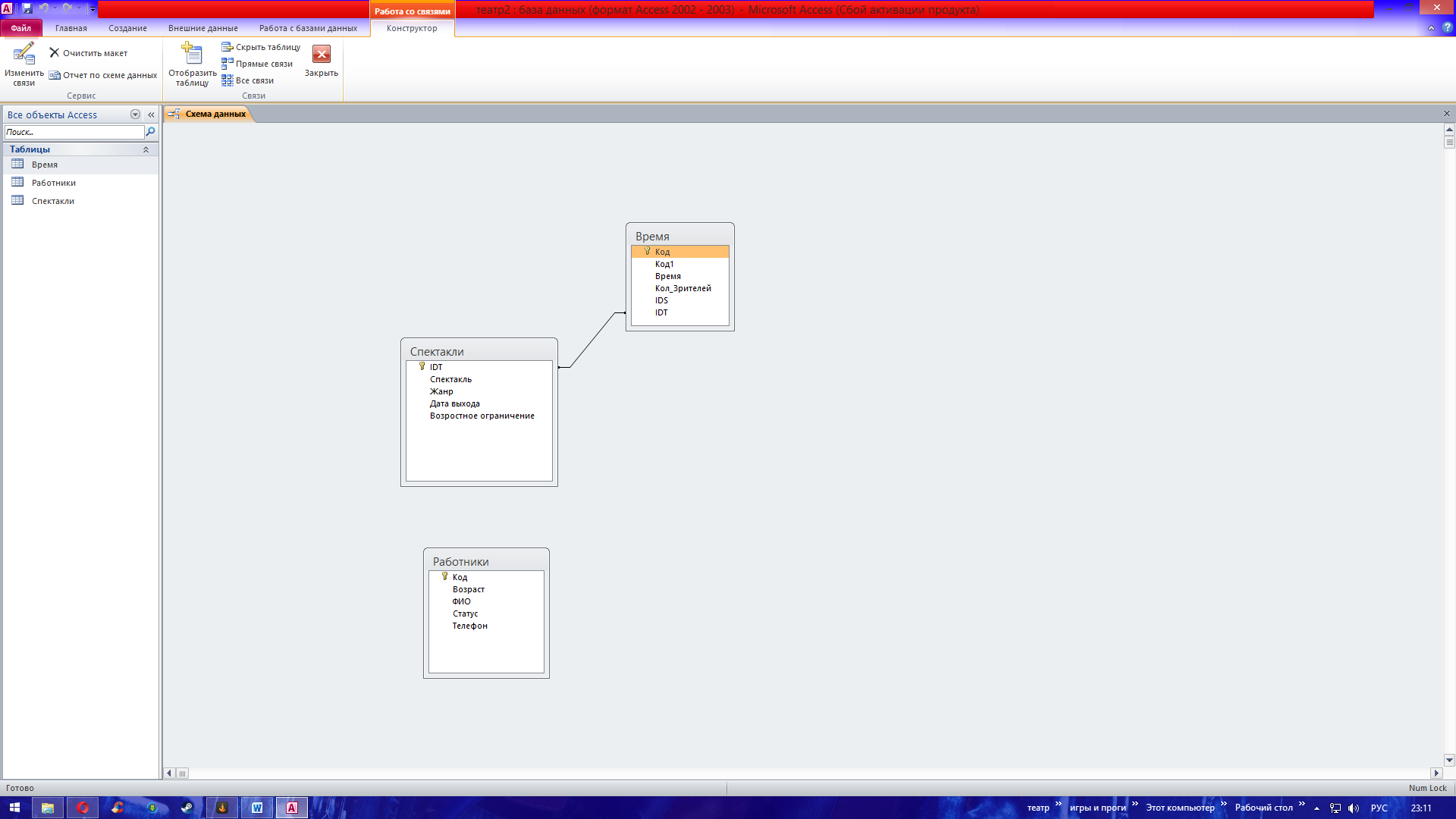


Рисунок 4.2 – Схема данных.

* 1. **Разработка пользовательского интерфейса**

После запуска программы пользователю отображается главное окно, внутри которого создаются все дочерние окна и формы для работы с данными. Такая организация интерфейса позволяет не запутаться среди всех окон и форм программы, а также более проста в освоении.

Доступ к основным функциям системы осуществляется через главное меню главного окна программы, главное меню разбито на разделы.

* 1. **Разработка программного обеспечения**

**Выбор и обоснование программных и технических средств разработки**

В настоящее время в мире информационных технологий имеются 3 основных семейства операционных систем для персональных компьютеров:

* Windows
* Mac OS
* Linux

Наиболее логичным будет выбрать самое популярное семейство операционных систем в роли внешней программной среды разрабатываемой ИС.

Согласно данным статистики, опубликованным компанией NetMarketShare, операционная система Windows 7 остается наиболее популярной из используемых операционных систем. Всего в статистической выборке приняло участие восемнадцать операционных систем. Тройка лидеров представлена операционными системами Microsoft - Windows 7 (44.71%), Windows XP (39.82%) и Windows Vista (5.70%).

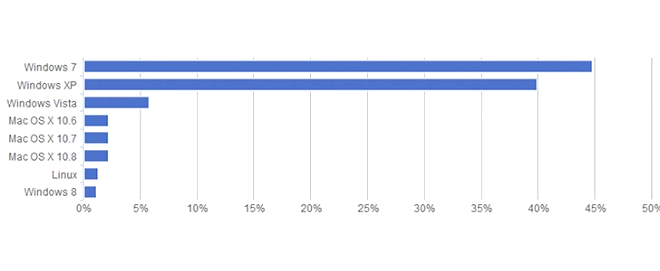


Рисунок 4.1 – Диаграмма популярности операционных системы

Таким образом, исходя из статистических данных, было решено разрабатывать информационную систему для работы в операционных системах семейства Microsoft.

Разрабатываемая ИС при необходимости должна работать на оборудовании с характеристиками:

* Процессор Intel Core 2 DUO или новее;
* ОЗУ 1 Гб или больше;
* ОС Windows 7 или новее;
* Не менее 1 Гб свободного места на ЖД;
* Принтер

Для разработки графического интерфейса предлагаемой к разработке информационной системы, было принято решение использовать среду программирования Delphi 7, поскольку Delphi позволяет, как создавать собственные базы, так и использовать уже созданные с помощью профессиональных СУБД и обладает всеми необходимыми средствами для подключения, использования и обработки данных, а также обеспечивает быстрое создание качественного интерфейса приложений.

Среда программирования Delphi обладает рядом преимуществ:

* ориентирована на начинающих разработчиков, имеющих опыт работы в операционных системах семейства Windows и обладающих минимальными навыками программирования (на любом из языков программирования);
* имеет визуальную разработку интерфейса пользователя;
* возможность работы с файлами на низком уровне;
* возможность работы с базами данных;
* совместимость со стандартными компонентами Windows;
* совместимость с большинством реляционных СУБД.
  1. **Разработка плана тестирования программного обеспечения**

**Проверка корректности ввода данных**

Система должна обеспечивать защиту от некорректно введенных данных, чтобы исключить неточность представленной в ИС информации.

Общий алгоритм проверки корректности заключается в сравнении введенных данных с некоторым шаблоном для этих данных, и в случае возникновения разногласий – вывести соответствующее сообщение пользователю и отменить ввод данных.

Также для снижения вероятности ввода некорректных данных в программе необходимо использовать блокировку ввода неподходящих символов. Например, в поле, где необходимо вводить только цифры (например, стоимость), запретить вводить какие-либо другие символы.

**Обработка данных**

Обработку данных в проектируемой ИС можно разделить на 3 этапа.

Первым этапом является обработка данных, вводимых с клавиатуры. На данном этапе программа проверяет корректность введенных данных и в случае их неправильного или некорректного ввода система блокирует ввод. Также для избегания ошибок в вводе повторяющихся параметров (модель телефон, производитель телефона и пр.) используются элементы управления в виде выпадающих списков.

Вторым этапом обработки данных является подсчет необходимых для каждого документа числовых значений и сохранение данных по документам в базе данных, как уже отмечалось выше, сохранение в базе данных информации происходит путем отправки SQL-запросов к системе управления базой данных (запрос INSERT).

Последним этапом обработки данных является сбор сохраненной информации по каждому документу и передача этой информации в шаблон документа, выполненный в программе Excel.

**Редактирование введенных данных**

Редактирование введенных ранее данных осуществляется с помощью форм пользовательского интерфейса, на котором эти данные отображаются.

Сначала запрашиваются необходимые данные из базы данных с помощью SQL-запросов (запрос SELECT), затем этими данными заполняются поля формы, после чего форма отображается пользователю.

Пользователь редактирует данные, и нажимает на кнопку «сохранить», все введенные данные проверяются на корректность, и в случае, если введенные значения корректны – к базе данных отправляется SQL-запрос на обновление данных (запрос UPDATE).

Для тестирования работоспособности системы необходимо смоделировать действия пользователя по введению информации в систему, проверить правильность обработки и сохранения всей информации в базе данных, а также получение информации из базы данных и вывод её в виде выходных документов (квитанция о приеме в ремонт, акт выполненных работ, чек об оплате).

Общий алгоритм тестирования выглядит следующим образом:

* подготовка тестовых данных для ввода;
* ввод данных в систему и тестирование системы проверки корректности вводимых данных;
* проверка корректности сохранения данных;
* тестирование возможности редактирования данных;
* тестирование выгрузки данных в выходные документы;

**Подготовка тестовых данных**

**Тестирование**

Введем тестовые данные в справочники системы, а затем создадим заказ клиента с использованием этих данных.

Главное меню программы с которого можно перейти на кабинет администратора, спектакли и справку, так же в правом верхнем углу спрятана кнопка для выхода из программы (для администрации). (Рисунок 5.1)

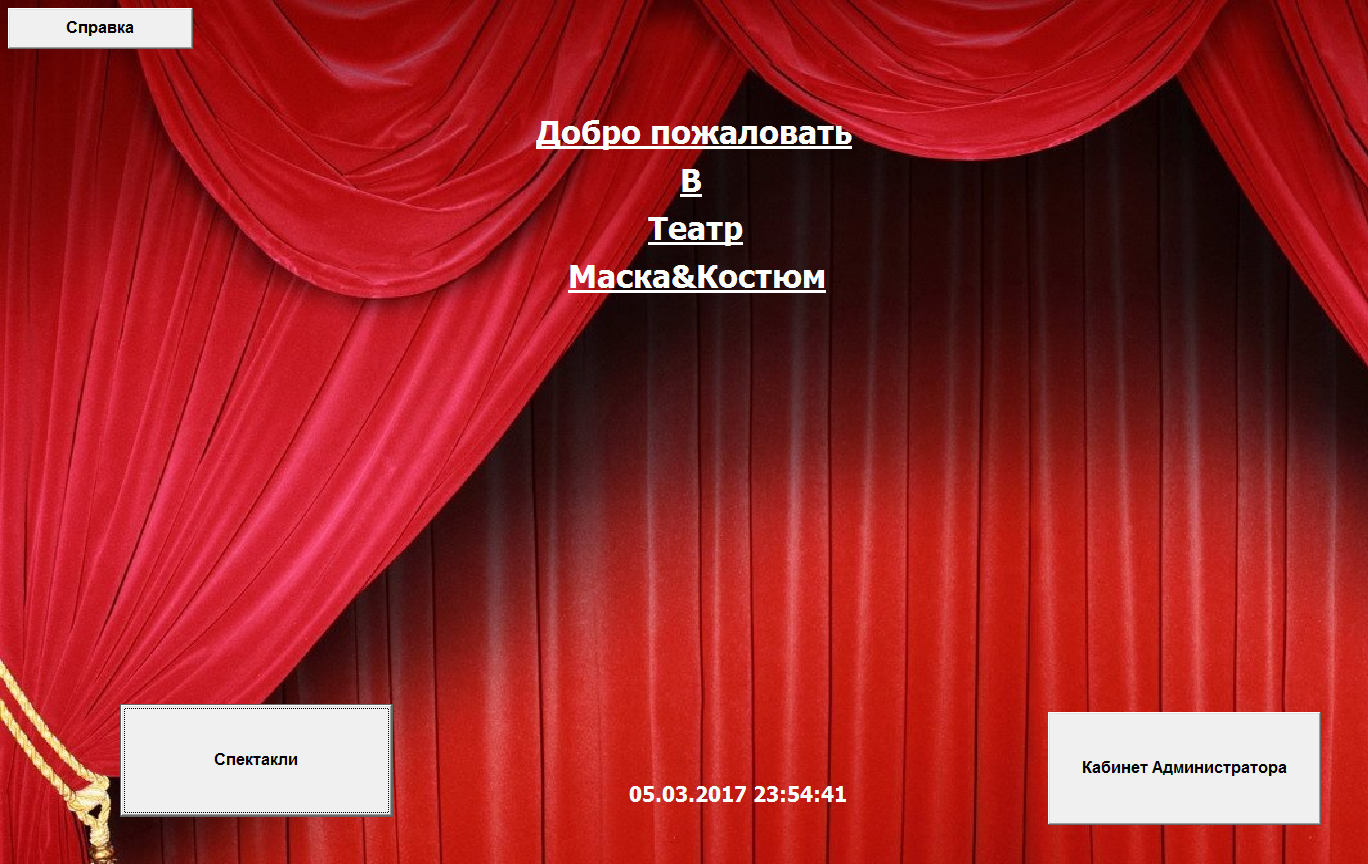


Рисунок 5.1 – Главное меню

На данной форме отображены спектакли на которые можно оформить билет, так же в правом нижнем углу расположена кнопка для просмотра 2-ой страницы в которой размещены спектакли (Рисунок 5.2)



Рисунок 5.2 – Выбор спектакля

Окно в котором можно выбрать время которое удобно пользователю. (Рисунок 5.3)

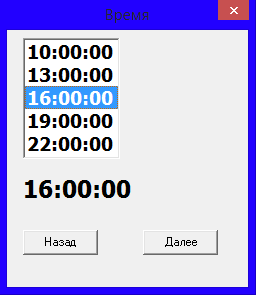


Рисунок 5.3 – Окно с выбором времени.

В данной форме можно выбрать любое удобное место, при выборе места в верхней панели указывается стоимость за место, выбранное время, количество выбранных мест и свободных мест, так же отображается изображение спектакля и его название. (Рисунок 5.4)



Рисунок 5.4 – Форма выбора места

В данном окне показывается контрольная информация о выборе спектакля, мест, время и суммы к оплате. (Рисунок 5.5)

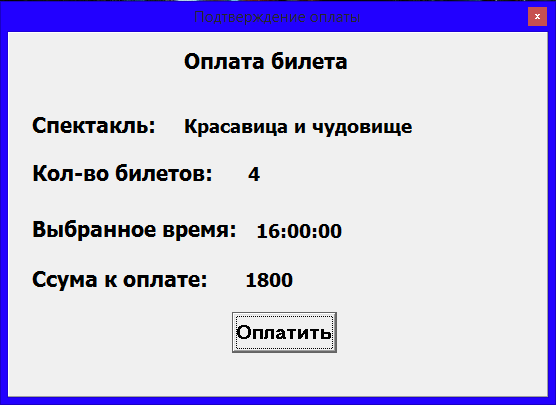


Рисунок 5.5 – Подтверждение оплаты

И последнее окно уведомление (Рисунок 5.6)

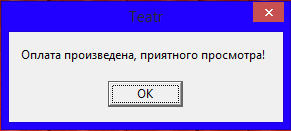


Рисунок 5.6 – Уведомление

**Интерпретация и описание результатов проведения тестирования**

Результаты тестирования разработанной информационной системы показывают хорошее функциональное наполнение ИС. Программное средство обладает следующими функциями:

* Позволяет вводить информацию, редактировать введенную ранее и удалять устаревшую информацию;
* Вся введенная информация сохраняется в подключенной базе данных
* На основании введенных данных система формирует требуемые выходные документы установленного образца
* Установлены ограничения на ввод некорректных данных, добавлены подтверждения действия пользователя, которые могут привести к потере информации, или её некорректному изменению

Кроме того, интерфейс программы и все его составляющие имеют спокойное оформление, выполненное в спокойных серых тонах. Все сообщения, отображаемые пользователю, выводятся на русском языке, без использования специализированных информационных терминов.

Установка приложения осуществляется простым копированием с внешнего носителя на жесткий диск компьютера. В случае аварийного завершения работы приложения все данные остаются сохраненными в файловой базе данных.

Таким образом можно сделать вывод, что разработанное программное обеспечение соответствует выделенным критериям качества по функциональному наполнению, внешнему оформлению интерфейса и защите информации от некорректного ввода и потери в случае аварийного завершения работы.

# ЗАКЛЮЧЕНИЕ

В ходе проделанной работы была создана информационная система для театра. Разработанное программное средство позволяет увеличить производительность работников сервисного центра за счет автоматизации создания заказов клиентов, формирования сопутствующих выходных документов.

В процессе разработки ИС были решены следующие задачи:

* Был проведен анализ деятельности сервисного центра, а также рассмотрены программные средства для сервисных центров, имеющиеся на рынке программных продуктов.
* На основе анализа было проведено концептуальное моделирование системы, обозначены критерии оценки качества разрабатываемого программного средства
* Используя результаты анализа предметной области и концептуального моделирования были созданы модели основных бизнес процессов в нотации IDEF0, разработана структура системы.
* По результатам функционального и структурного моделирования была спроектирована информационная система, выбраны программные средства реализации, разработаны модели данных и основные алгоритмы.
* Проект был реализован в виде готового программного средства для автоматизации работы сервисного центра
* Последним этапом было тестирование созданного программного средства на соответствие обозначенным критериям качества. Результатом тестирования стало полное соответствие созданной ИС заявленным требованиям.

Данная система является законченным решением для сервисного центра по ремонту сотовых телефонов, однако можно дополнять и расширять функционал системы. Например, можно добавить перечень услуг по ремонту в виде прайса для автоматического подсчета стоимости ремонта. Также можно добавить интерфейс инженера по ремонту сотовых телефонов, для того, чтобы он мог ставить пометки к заказам, а также вести учет затрат на комплектующие к ремонтируемым телефонам.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Черемных С. В. Моделирование и анализ систем. IDEF-технологии / Черемных С. В. [и др.]. М., 2003. – 30 стр.
2. Шафер Д.Ф., Фартрел Т., Шафер Л.И. Управление программными проектами – стр 21
3. Четвергов К. В. Информатика – учебное методическое пособие. Часть 3. Томск, 2011.
4. Архангельский А.Я. – 100 компонентов Delphi, 2007
5. Озеров В.И. - Советы по Delphi, 2009

# Приложение А. Словарь данных

# Приложение Б Руководство по установке

## **Руководство по установке**

Для установки приложения необходимо скопировать все файлы и папки программы с CD-диска на жесткий диск компьютера пользователя.

Запуск программы осуществляется с помощью двойного клика левой кнопкой мыши на запускном файле Service.exe.

После запуска пользователю отображается главное окно программы (рисунок А.1), внутри которого создаются все дочерние формы программы.

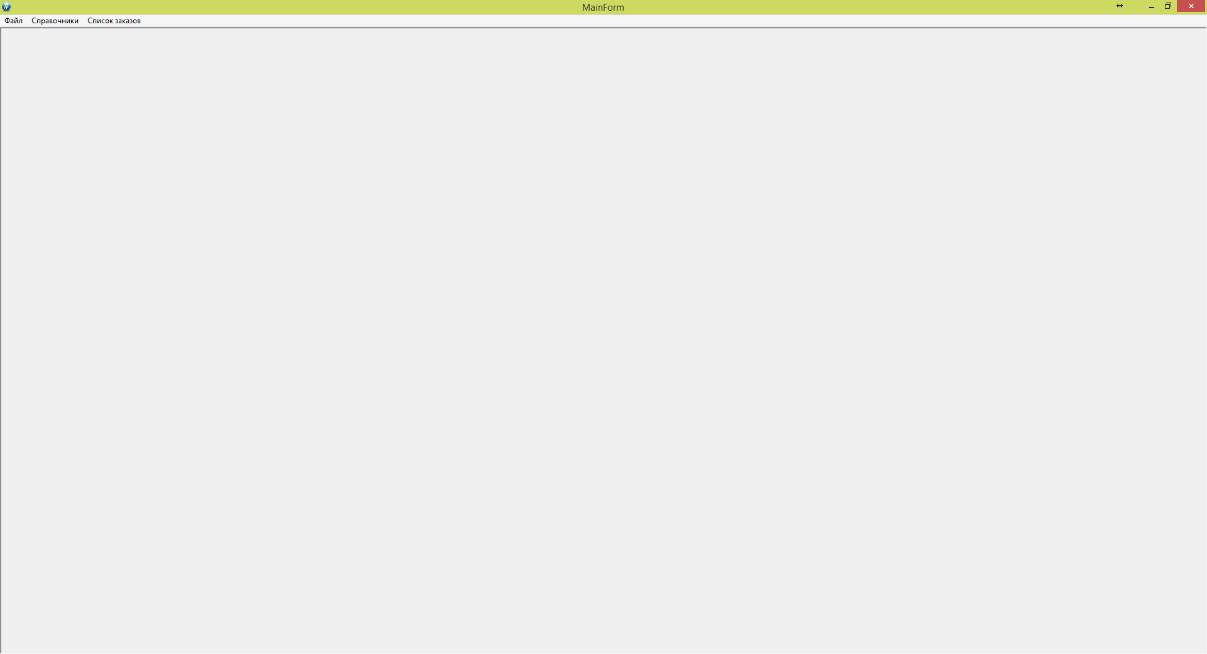


Рисунок А.1 – Главное окно информационной системы

Доступ к основным функциям ИС осуществляется с помощью главного меню главного окна приложения (Рисунок А.2)

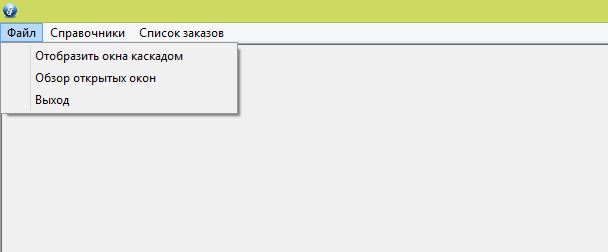


Рисунок А.2 – Главное меню программы

Главное меню состоит из трех разделов: Файл, Справочники, Заказы клиентов.

В разделе Файл имеется два пункта, предназначенные для управления режимами отображения дочерних окон и один пункт для выхода из программы. При выборе пункта Выход у пользователя спрашивается подтверждение (Рисунок А.3).

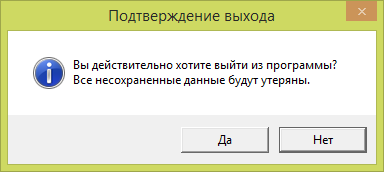


Рисунок А.3 – Подтверждение выхода из программы

Раздел справочник содержит пункты меню для вызова формы работы со справочниками. Всего в системе пять справочников, для работы с ними используется типовая форма (Рисунок А.4)

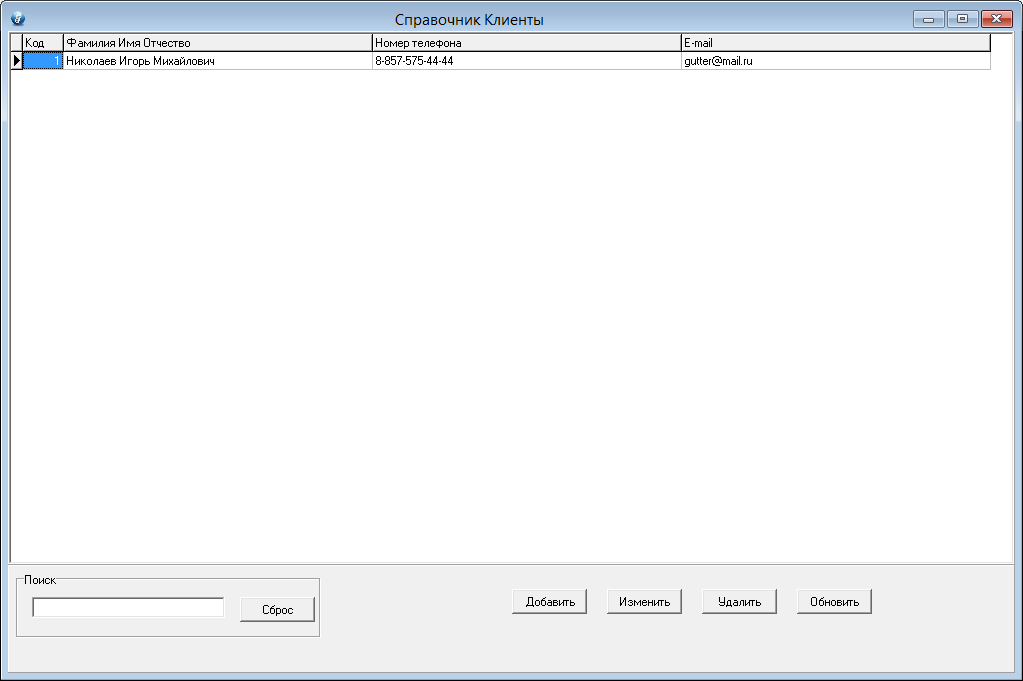


Рисунок А.4 – Форма Справочник

Центральную часть формы занимает таблица данных справочника, под таблицей расположены элементы управления справочников. Можно добавить данные, изменить уже имеющиеся, а также удалить устаревшие данные. Также на панели инструментов находится поле для поиска нужного значения в отображаемом справочнике. Следует заметить, что при удалении данных у пользователя запрашивается подтверждение (Рисунок А.5), это позволит избежать случайного удаления важных данных.

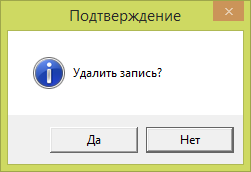


Рисунок А.5 – Подтверждение удаления данных из справочника

Следующий раздел главного меню программы – Список заказов, при нажатии на данный раздел открывается форма, в которой отображаются все заказа, в хронологическом порядке (Рисунок А.6).

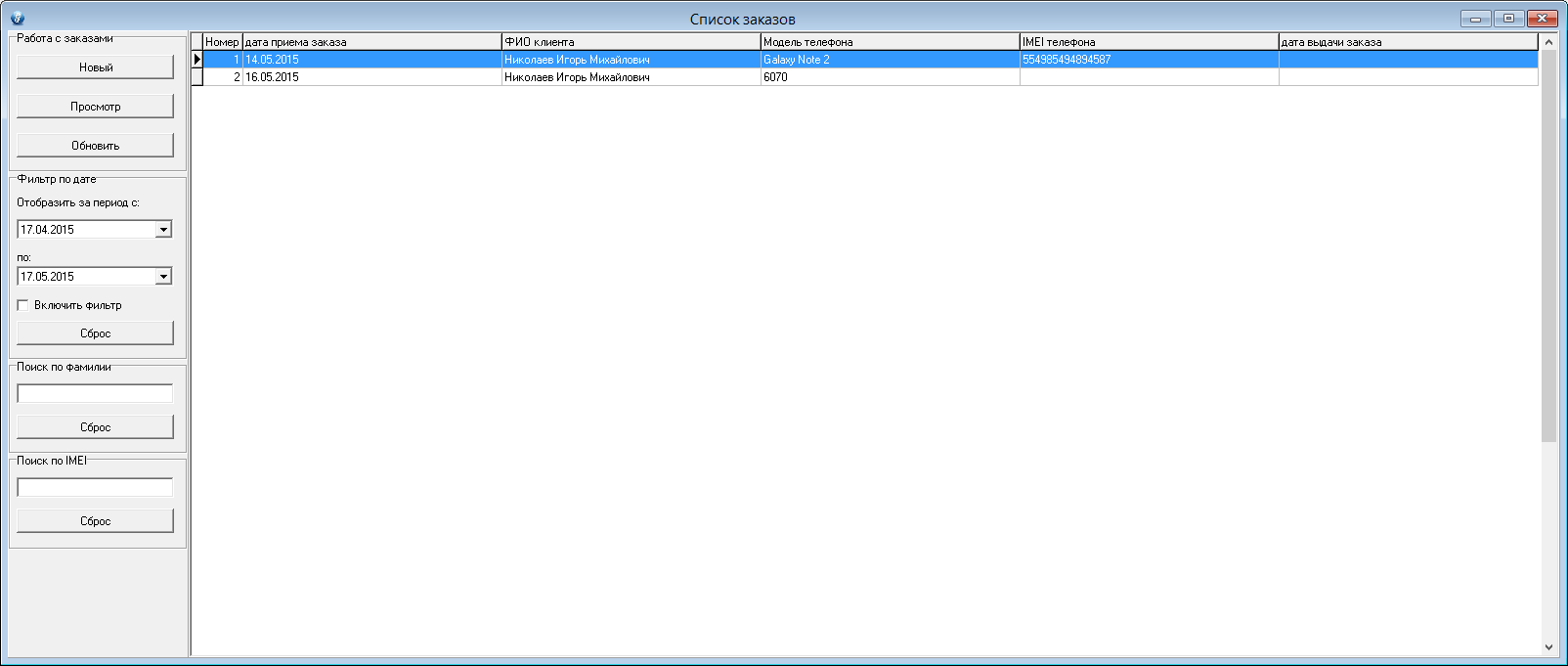


Рисунок А.6 – Форма Список заказов

В левой части формы расположены элементы для работы с заказами, а также инструменты для фильтрации списка заказов и поиска нужного, имеется фильтр по интервалу дат, поиск по ФИО клиента, а также по IMEI телефона клиента.

В поле для IMEI можно вводить только числа.

При нажатии на кнопку Новый открывается форма Заказ в режиме создания нового заказа (Рисунок А.7).

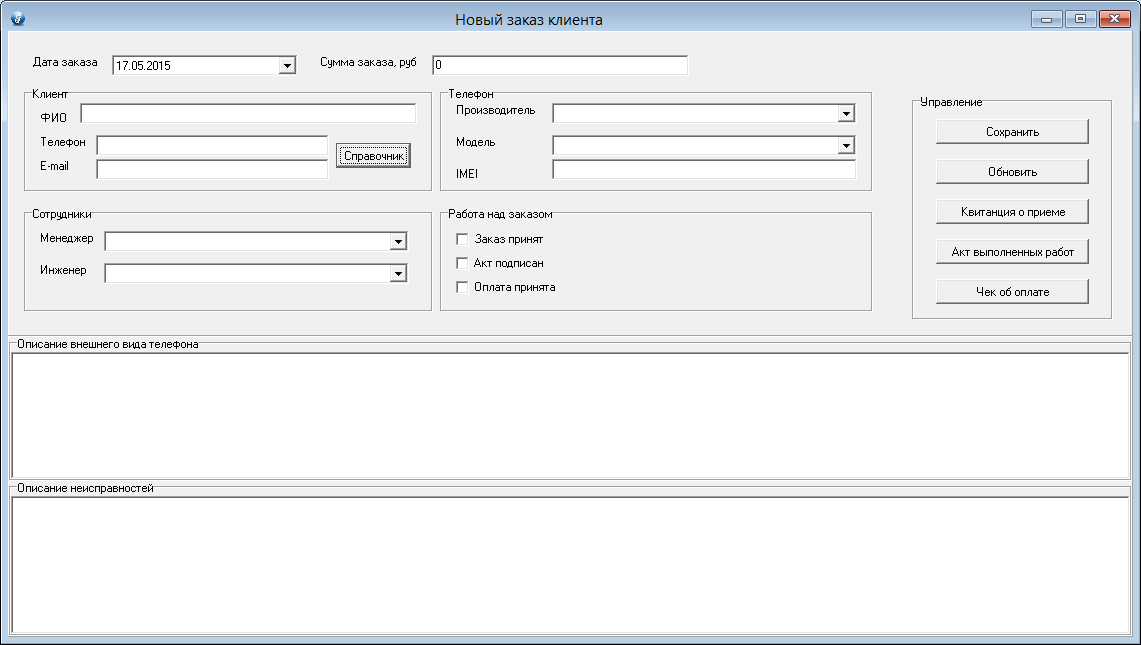


Рисунок А.7 – Форма для создания нового заказа.

При нажатии на кнопку Просмотр открывается форма Заказ в режиме редактирования заказа, то есть в поля формы подставляются данные выбранного из списка заказа (Рисунок А.8).

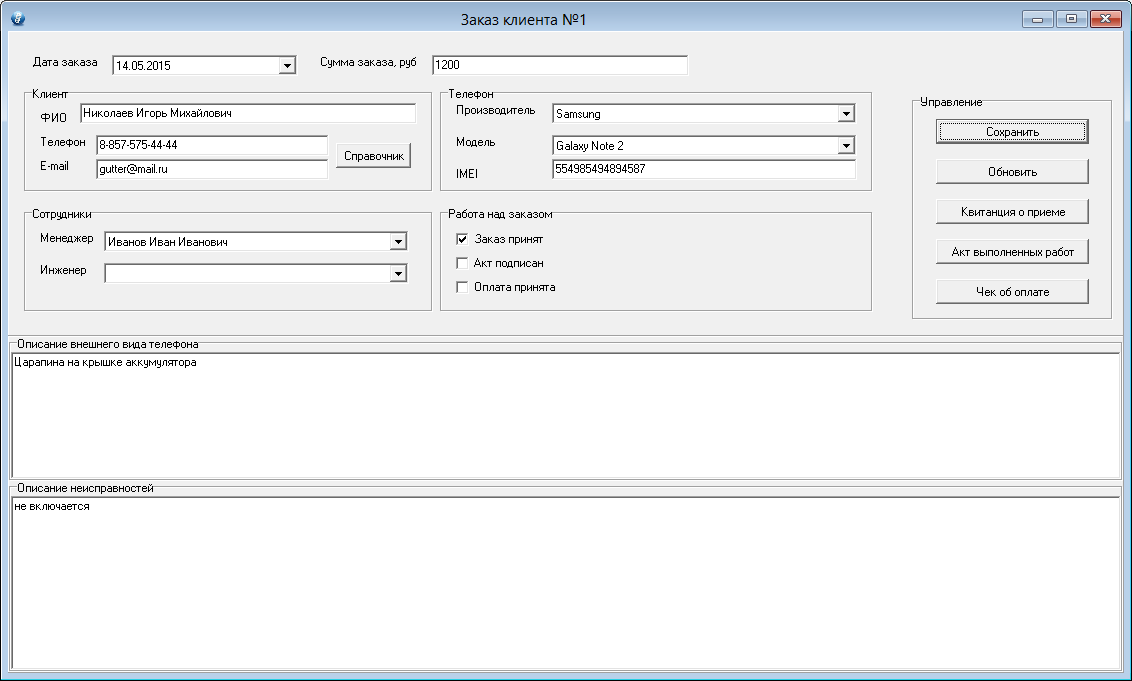


Рисунок А.8 – Форма для редактирования заказа.

Если в заказ были внесены какие-то изменения, и не изменены, то перед закрытием формы программа спросит у пользователя что делать с несохраненными данными (Рисунок А.9)

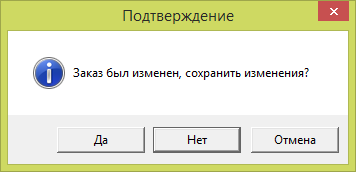


Рисунок А.10 – Диалоговое окно при закрытии несохраненных изменений в заказе.

Все поля заказа заполняются либо непосредственно текстовыми/числовыми значениями, либо с помощью выпадающих списков. Исключение составляет лишь выбор клиента, чтобы выбрать клиента необходимо нажат на кнопку справочник (откроется форма Справочник), затем выбрать пункт выбрать из контекстного меню нужного клиента в таблице справочника (Рисунок А.11).

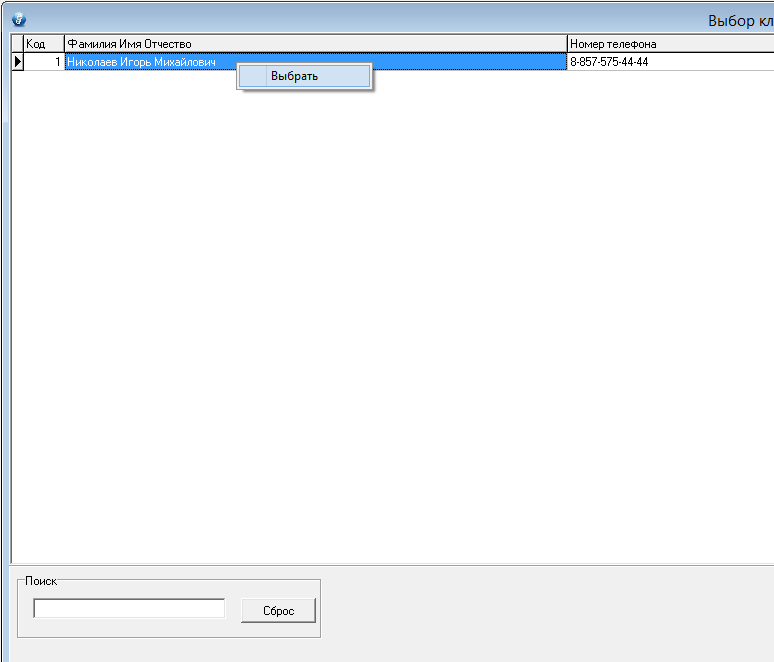


Рисунок А.11 – Выбор клиента из справочника для формирования заказа

Также из формы заказ печатаются выходные документы: Квитанция о приеме в ремонт (Рисунок А.12), акт выполненных работ (Рисунок А.13), чек об оплате услуг сервисного центра (Рисунок А.14).

# 

# Приложение В Листинг программы

**Модуль Main.pas**

unit Main;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Menus;

type

TMainForm = class(TForm)

MainMenu1: TMainMenu;

N1: TMenuItem;

N3: TMenuItem;

N4: TMenuItem;

N5: TMenuItem;

N6: TMenuItem;

N7: TMenuItem;

N8: TMenuItem;

N9: TMenuItem;

N10: TMenuItem;

N11: TMenuItem;

N13: TMenuItem;

N14: TMenuItem;

procedure CreateSpravForm(const childName, id: string; table: string;

p: integer; search: string);

procedure N7Click(Sender: TObject);

procedure N8Click(Sender: TObject);

procedure N9Click(Sender: TObject);

procedure N10Click(Sender: TObject);

procedure N11Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure N3Click(Sender: TObject);

procedure N5Click(Sender: TObject);

procedure N13Click(Sender: TObject);

procedure N14Click(Sender: TObject);

private

procedure CreateZakazListForm(const childName:string);

{ Private declarations }

public

{ Public declarations }

end;

var

MainForm: TMainForm;

implementation

Uses DataM, Sprav, zakazlist;

{Процедура создания окна-справочника. последний параметр - номер элемента контекстного менб для отображения:

1 - Подробнее о модели автомата

2 - Подробнее о продукте питания

3 - Выбрать для слота

4 - Подробнее о реальном автомате}

procedure TMainForm.CreateSpravForm(const childName, id : string; table: string; p: integer; search: string);

var

Child: TSpravForm;

i,k :integer;

f : boolean;

begin

f:=False; //влаг-ложь

k:=0;

for i:=0 to MainForm.MDIChildCount-1 do //проверяем создан ли экземнляр такого окна

if (MainForm.MDIChildren[i] is TSpravForm) and (MainForm.MDIChildren[i].caption=childName) then

begin

f:=True;

k:=i;

end;

if (f) then begin //если создан, то

MainForm.MDIChildren[k].bringToFront; //переносим его вперед

if MainForm.MDIChildren[k].WindowState=wsMinimized then

MainForm.MDIChildren[k].WindowState:=wsNormal;

end

else

begin //иначе - создаем форму

Child := TSpravForm.Create(MainForm);

Child.Caption := childName;

if table='sotr' then

begin

Child.DbGrid1.DataSource.DataSet:= Child.Sotr;

Child.DbGrid1.DataSource.DataSet.open;

end;

if table='dolg' then

begin

Child.DbGrid1.DataSource.DataSet:= Child.dolg;

Child.DbGrid1.DataSource.DataSet.Active:=true;

end;

if table='klient' then

begin

Child.DbGrid1.DataSource.DataSet:= Child.klient;

Child.DbGrid1.DataSource.DataSet.Active:=true;

end;

if table='phone' then

begin

Child.DbGrid1.DataSource.DataSet:= Child.phone;

Child.DbGrid1.DataSource.DataSet.Active:=true;

end;

if table='prod' then

begin

Child.DbGrid1.DataSource.DataSet:= Child.prod;

Child.DbGrid1.DataSource.DataSet.Active:=true;

end;

Child.Setgridcolumnwidths;

Child.Label1.Caption:=id;

Child.label2.caption:=Search;

Child.Edit1.Clear;

Child.Visible:=true;

{Case p of

1: Child.n1.visible:=true; //отображаем первый пункт сонтекстного меню

2: Child.N2.Visible:=true;

3: Child.N3.Visible:=true;

4: Child.N4.Visible:=true;

end; }

end;

end;

procedure TMainForm.CreateZakazListForm(const childName:string);

var

Child: TzakazlistForm;

i,k :integer;

f : boolean;

begin

f:=False;

k:=0;

for i:=0 to MainForm.MDIChildCount-1 do

if (MainForm.MDIChildren[i] is TzakazlistForm) and (MainForm.MDIChildren[i].caption=childName) then

begin

f:=True;

k:=i;

end;

if (f) then begin

MainForm.MDIChildren[k].bringToFront;

if MainForm.MDIChildren[k].WindowState=wsMinimized then

MainForm.MDIChildren[k].WindowState:=wsNormal;

end

else

begin //иначе - создаем форму

Child := TzakazlistForm.Create(MainForm);

Child.Caption := childName;

Child.Edit1.Clear;

Child.Edit2.Clear;

Child.DateTimePicker1.Date:=Now-30;

Child.DateTimePicker2.Date:=Now;

Child.Visible:=true;

end;

end;

{$R \*.dfm}

procedure TMainForm.N7Click(Sender: TObject);

begin

CreateSpravForm('Справочник Сотрудники','0','sotr',0,'fio');

end;

procedure TMainForm.N8Click(Sender: TObject);

begin

CreateSpravForm('Справочник Должности','0','dolg',0,'nam');

end;

procedure TMainForm.N9Click(Sender: TObject);

begin

CreateSpravForm('Справочник Клиенты','0','klient',0,'fio');

end;

procedure TMainForm.N10Click(Sender: TObject);

begin

CreateSpravForm('Справочник Телефоны','0','phone',0,'modelname');

end;

procedure TMainForm.N11Click(Sender: TObject);

begin

CreateSpravForm('Справочник Производители телефонов','0','prod',0,'nam');

end;

procedure TMainForm.FormClose(Sender: TObject; var Action: TCloseAction);

begin

If MessageBox(Handle,PChar('Вы действительно хотите выйти из программы?'+#10+#13+'Все несохраненные данные будут утеряны.'),PChar('Подтверждение выхода'),MB\_ICONINFORMATION+MB\_YESNO+MB\_DEFBUTTON2)=IdYes

then Application.Terminate

else

action:=canone;

end;

procedure TMainForm.N3Click(Sender: TObject);

begin

close;

end;

procedure TMainForm.N5Click(Sender: TObject);

begin

CreateZakazListForm('Список заказов');

end;

procedure TMainForm.N13Click(Sender: TObject);

begin

Cascade;

end;

procedure TMainForm.N14Click(Sender: TObject);

begin

Tile;

end;

end.

**Модуль Sprav.pas**

unit Sprav;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, DB, StdCtrls, Grids, DBGrids, ExtCtrls, ADODB, Buttons, Menus;

type

TSpravForm = class(TForm)

Panel1: TPanel;

DBGrid1: TDBGrid;

DataSource1: TDataSource;

Sotr: TADOTable;

Panel2: TPanel;

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

BitBtn4: TBitBtn;

Label1: TLabel;

Label2: TLabel;

Dolg: TADOTable;

Klient: TADOTable;

phone: TADOTable;

prod: TADOTable;

GroupBox1: TGroupBox;

BitBtn5: TBitBtn;

Edit1: TEdit;

Sotrid: TAutoIncField;

Sotrfio: TWideStringField;

Sotrphonenumber: TWideStringField;

Sotremail: TWideStringField;

Sotrpassword: TWideStringField;

Sotrdolg: TIntegerField;

SotrdolgL: TStringField;

Dolgid: TAutoIncField;

Dolgnam: TWideStringField;

Klientid: TAutoIncField;

Klientfio: TWideStringField;

Klientphonenumber: TWideStringField;

Klientemail: TWideStringField;

phoneid: TAutoIncField;

phoneprodid: TIntegerField;

phonemodelname: TWideStringField;

phoneopisanie: TMemoField;

phoneProdL: TStringField;

prodid: TAutoIncField;

prodnam: TWideStringField;

PopupMenu1: TPopupMenu;

N1: TMenuItem;

N2: TMenuItem;

N3: TMenuItem;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormResize(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure Edit1Change(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure DBGrid1TitleClick(Column: TColumn);

procedure N1Click(Sender: TObject);

private

{ Private declarations }

public

procedure Setgridcolumnwidths;

{ Public declarations }

end;

var

SpravForm: TSpravForm;

implementation

Uses DataM;

{$R \*.dfm}

Procedure TSpravForm.Setgridcolumnwidths;

var

i,x,dx:integer;

begin

if DBGrid1.Columns.count>1 then

begin

x:=Round((DbGrid1.width-80)/(DbGrid1.Columns.Count-1));

DbGrid1.Columns[0].Width:=40;

For i:=1 to DbGrid1.Columns.Count-1 do

begin

DbGrid1.Columns[i].Width:=x;

end;

X:=Round(Panel2.Width/2);

dx:=20;

BitBtn1.Left:=x;

BitBtn2.Left:=x+75+dx;

BitBtn3.Left:=x+2\*75+2\*dx;

BitBtn4.Left:=x+3\*75+3\*dx;

end;

end;

procedure TSpravForm.FormClose(Sender: TObject; var Action: TCloseAction);

begin

action:=cafree;

end;

procedure TSpravForm.FormResize(Sender: TObject);

begin

Setgridcolumnwidths

end;

procedure TSpravForm.BitBtn4Click(Sender: TObject);

begin

DbGrid1.DataSource.DataSet.Refresh;

end;

procedure TSpravForm.BitBtn1Click(Sender: TObject);

begin

DbGrid1.DataSource.DataSet.Insert;

end;

procedure TSpravForm.BitBtn2Click(Sender: TObject);

begin

DbGrid1.DataSource.DataSet.Edit;

end;

procedure TSpravForm.BitBtn3Click(Sender: TObject);

begin

If MessageBox(Handle,PChar('Удалить запись?'),PChar('Подтверждение'),MB\_ICONINFORMATION+MB\_YESNO+MB\_DEFBUTTON2)=IdYes

then Dbgrid1.DataSource.DataSet.Delete;

end;

procedure TSpravForm.Edit1Change(Sender: TObject);

begin

if Edit1.text='' then

DBGRid1.DataSource.DataSet.Filtered:=false

else

begin

DBGRID1.DataSource.DataSet.Filter:=label2.caption+' LIKE '+QuotedSTR('\*'+Edit1.Text+'\*');

DbGrid1.DataSource.DataSet.Filtered:=true;

end;

end;

procedure TSpravForm.BitBtn5Click(Sender: TObject);

begin

Edit1.Clear;

end;

procedure TSpravForm.DBGrid1TitleClick(Column: TColumn);

begin

if (Column.FieldName<>'ProdL') and (Column.FieldName<>'dolgL') then

if TADODataSet(Column.Field.DataSet).Sort = Column.FieldName+' ASC' then TADODataSet(Column.Field.DataSet).Sort := Column.FieldName+' DESC' else TADODataSet(Column.Field.DataSet).Sort := Column.FieldName+' ASC';

if (Column.FieldName='ProdL') then

if TADODataSet(Column.Field.DataSet).Sort = 'prodid ASC' then TADODataSet(Column.Field.DataSet).Sort := 'prodid DESC' else TADODataSet(Column.Field.DataSet).Sort := 'prodid ASC';

if (Column.FieldName='dolgL') then

if TADODataSet(Column.Field.DataSet).Sort = 'dolg ASC' then TADODataSet(Column.Field.DataSet).Sort := 'dolg DESC' else TADODataSet(Column.Field.DataSet).Sort := 'dolg ASC';

end;

procedure TSpravForm.N1Click(Sender: TObject);

begin

DataModule1.Sotr.Filter:='id = '+DbGrid1.DataSource.DataSet.FieldByName('id').AsString;

close;

end;

end.

**Модуль Zakaz.pas**

unit Zakaz;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, DBCtrls, Mask, Buttons, ComCtrls, ExtCtrls, DB, ADODB, ComObj, shellapi;

type

TZakazForm = class(TForm)

Panel1: TPanel;

Panel2: TPanel;

Label1: TLabel;

GroupBox1: TGroupBox;

GroupBox2: TGroupBox;

GroupBox3: TGroupBox;

GroupBox4: TGroupBox;

GroupBox5: TGroupBox;

DateTimePicker1: TDateTimePicker;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

Label8: TLabel;

Label9: TLabel;

CheckBox1: TCheckBox;

CheckBox2: TCheckBox;

CheckBox3: TCheckBox;

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

BitBtn4: TBitBtn;

BitBtn5: TBitBtn;

DBLookupComboBox1: TDBLookupComboBox;

DBLookupComboBox2: TDBLookupComboBox;

Edit3: TEdit;

DBLookupComboBox3: TDBLookupComboBox;

DBLookupComboBox4: TDBLookupComboBox;

GroupBox6: TGroupBox;

GroupBox7: TGroupBox;

Memo1: TMemo;

Memo2: TMemo;

BitBtn6: TBitBtn;

ADOQuery1: TADOQuery;

DBEdit1: TDBEdit;

DBEdit2: TDBEdit;

DBEdit3: TDBEdit;

DataSource1: TDataSource;

ADOTable1: TADOTable;

DBText1: TDBText;

DataSource2: TDataSource;

DataSource3: TDataSource;

ADOTable2: TADOTable;

ADOTable3: TADOTable;

ADOTable4: TADOTable;

ADOTable5: TADOTable;

DataSource4: TDataSource;

DataSource5: TDataSource;

ADOQuery2: TADOQuery;

Label10: TLabel;

Edit1: TEdit;

ADOQuery3: TADOQuery;

Label11: TLabel;

procedure BitBtn6Click(Sender: TObject);

procedure DBEdit1Change(Sender: TObject);

procedure DBLookupComboBox1Click(Sender: TObject);

procedure DBLookupComboBox2Click(Sender: TObject);

procedure Edit3Change(Sender: TObject);

procedure CheckBox1Click(Sender: TObject);

procedure CheckBox2Click(Sender: TObject);

procedure CheckBox3Click(Sender: TObject);

procedure DBLookupComboBox3Click(Sender: TObject);

procedure DBLookupComboBox4Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure Memo1Change(Sender: TObject);

procedure Memo2Change(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure Edit1Change(Sender: TObject);

procedure DateTimePicker1Change(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

private

procedure CreateSpravForm(const childName, id: string; table: string;

p: integer; search: string);

{ Private declarations }

public

{ Public declarations }

end;

var

ZakazForm: TZakazForm;

implementation

Uses DataM, Sprav, Main;

{$R \*.dfm}

procedure TZakazForm.CreateSpravForm(const childName, id : string; table: string; p: integer; search: string);

var

Child: TSpravForm;

i,k :integer;

f : boolean;

begin

f:=False; //влаг-ложь

k:=0;

for i:=0 to MainForm.MDIChildCount-1 do //проверяем создан ли экземнляр такого окна

if (MainForm.MDIChildren[i] is TSpravForm) and (MainForm.MDIChildren[i].caption=childName) then

begin

f:=True;

k:=i;

end;

if (f) then begin //если создан, то

MainForm.MDIChildren[k].bringToFront; //переносим его вперед

if MainForm.MDIChildren[k].WindowState=wsMinimized then

MainForm.MDIChildren[k].WindowState:=wsNormal;

end

else

begin //иначе - создаем форму

Child := TSpravForm.Create(MainForm);

if table='sotr' then

begin

Child.DbGrid1.DataSource.DataSet:= DataModule1.Sotr;

Child.DbGrid1.DataSource.DataSet.open;

end;

if table='dolg' then

begin

Child.DbGrid1.DataSource.DataSet:= DataModule1.dolg;

Child.DbGrid1.DataSource.DataSet.Active:=true;

end;

if table='klient' then

begin

Child.DbGrid1.DataSource.DataSet:= DataModule1.klient;

Child.DbGrid1.DataSource.DataSet.Active:=true;

end;

if table='phone' then

begin

Child.DbGrid1.DataSource.DataSet:= DataModule1.phone;

Child.DbGrid1.DataSource.DataSet.Active:=true;

end;

if table='prod' then

begin

Child.DbGrid1.DataSource.DataSet:= DataModule1.prod;

Child.DbGrid1.DataSource.DataSet.Active:=true;

end;

Child.Setgridcolumnwidths;

Child.Label1.Caption:=id;

Child.label2.caption:=Search;

Child.Edit1.Clear;

Child.Visible:=true;

Case p of

1: Child.n1.visible:=true; //отображаем первый пункт сонтекстного меню

2: Child.N2.Visible:=true;

3: Child.N3.Visible:=true;

end;

Child.Caption := childName;

end;

end;

procedure TZakazForm.BitBtn6Click(Sender: TObject);

begin

CreateSpravForm('Выбор клиента','','klient', 1,'fio');

end;

procedure TZakazForm.DBEdit1Change(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.DBLookupComboBox1Click(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.DBLookupComboBox2Click(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.Edit3Change(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.CheckBox1Click(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.CheckBox2Click(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.CheckBox3Click(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.DBLookupComboBox3Click(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.DBLookupComboBox4Click(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.BitBtn2Click(Sender: TObject);

begin

AdoQuery1.Close;

AdoQuery1.Open;

AdoQuery2.Close;

AdoQuery2.Open;

DBEdit1.DataSource.DataSet.Filter:='id = '+AdoQuery1.FieldByName('klientid').AsString;

DBEdit1.DataSource.DataSet.Open;

DBLookupComboBox2.ListSource.DataSet.Open;

DBLookupComboBox2.KeyValue:= AdoQuery1.FieldByName('phoneid').AsVariant;

DBLookupComboBox1.ListSource.DataSet.Open;

DBLookupComboBox1.KeyValue:= AdoQuery2.FieldByName('prodid').AsVariant ;

DBLookupComboBox3.ListSource.DataSet.Open;

DBLookupComboBox3.KeyValue:= AdoQuery1.FieldByName('menedgerid').AsVariant;

DBLookupComboBox4.ListSource.DataSet.Open;

DBLookupComboBox4.KeyValue:= AdoQuery1.FieldByName('ingenerid').AsVariant ;

Edit3.Text:=AdoQuery1.FieldByName('phoneIMEI').AsString;

CheckBox1.Checked:=AdoQuery1.FieldByName('prinyat').AsBoolean;

CheckBox2.Checked:=AdoQuery1.FieldByName('akt').AsBoolean;

CheckBox3.Checked:=AdoQuery1.FieldByName('oplata').AsBoolean;

Memo1.Text:=AdoQuery1.FieldByName('prim').AsString;

Memo2.Text:=AdoQuery1.FieldByName('opisanie').AsString;

DateTimePicker1.DateTime:=AdoQuery1.FieldByName('Datepriem').AsDateTime;

Edit1.Text:=AdoQuery1.FieldByName('summ').AsString;

end;

procedure TZakazForm.Memo1Change(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.Memo2Change(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.BitBtn1Click(Sender: TObject);

begin

if label11.caption='0' then

begin

With AdoQuery3 do

begin

close;

SQL.Clear;

SQL.add('INSERT INTO ZAKAZ ( klientid, phoneid, phoneIMEI, datepriem, summ, opisanie, prim, menedgerid, ingenerid, prinyat, akt, oplata) ');

SQL.add( 'VALUES ( :klientid, :phoneid, :phoneIMEI, :datepriem, :summ, :opisanie, :prim, :menedgerid, :ingenerid, :prinyat, :akt, :oplata) ');

Parameters.ParamByName('klientid').Value:= DBEdit1.DataSource.DataSet.FieldByName('id').AsString;

Parameters.ParamByName('phoneid').Value:= DbLookupCombobox2.KeyValue;

Parameters.ParamByName('datepriem').Value:= DateTImePicker1.dateTime;

Parameters.ParamByName('phoneIMEI').Value:= Edit3.text;

Parameters.ParamByName('summ').Value:= Edit1.text;

Parameters.ParamByName('opisanie').Value:= Memo2.text;

Parameters.ParamByName('prim').Value:= memo1.text;

Parameters.ParamByName('menedgerid').Value:= DBLookupCombobox3.KeyValue;

Parameters.ParamByName('ingenerid').Value:= DBLookupCombobox4.KeyValue;

Parameters.ParamByName('prinyat').Value:= CheckBox1.Checked;

Parameters.ParamByName('akt').Value:= CheckBox2.Checked;

Parameters.ParamByName('oplata').Value:= CheckBox3.Checked;

EXECSQL;

end;

With AdoQuery3 do

begin

close;

SQL.clear;

SQl.Add('SELECT max(id) As Maxid FROM zakaz');

open;

end;

label11.caption:=AdoQuery3.FieldByName('maxid').AsString;

caption:='Заказ клиента № '+AdoQuery3.FieldByName('maxid').AsString;

end

else

begin

With AdoQuery3 do

begin

close;

SQL.Clear;

SQL.add('UPDATE zakaz SET klientid = :klientid, phoneid = :phoneid, phoneIMEI = :phoneIMEI, datepriem = :datepriem, summ = :summ, ');

SQL.add(' opisanie = :opisanie, prim = :prim, menedgerid = :menedgerid, ingenerid = :ingenerid, prinyat = :prinyat, akt = :akt, oplata = :oplata WHERE id = '+label11.caption);

Parameters.ParamByName('klientid').Value:= DBEdit1.DataSource.DataSet.FieldByName('id').AsString;

Parameters.ParamByName('phoneid').Value:= DbLookupCombobox2.KeyValue;

Parameters.ParamByName('phoneIMEI').Value:= Edit3.text;

Parameters.ParamByName('datepriem').Value:= DateTImePicker1.dateTime;

Parameters.ParamByName('summ').Value:= Edit1.text;

Parameters.ParamByName('opisanie').Value:= Memo2.text;

Parameters.ParamByName('prim').Value:= memo1.text;

Parameters.ParamByName('menedgerid').Value:= DBLookupCombobox3.KeyValue;

Parameters.ParamByName('ingenerid').Value:= DBLookupCombobox4.KeyValue;

Parameters.ParamByName('prinyat').Value:= CheckBox1.Checked;

Parameters.ParamByName('akt').Value:= CheckBox2.Checked;

Parameters.ParamByName('oplata').Value:= CheckBox3.Checked;

EXECSQL;

end;

caption:= copy(caption, 0, length(caption)-1);

end;

end;

procedure TZakazForm.FormClose(Sender: TObject; var Action: TCloseAction);

var

answer:integer;

begin

if copy(caption, length(caption), 1) = '\*' then

begin

answer:=MessageBox(Handle,PChar('Заказ был изменен, сохранить изменения?'),PChar('Подтверждение'),MB\_ICONINFORMATION+MB\_YESNOCANCEL+MB\_DEFBUTTON2);

if answer=6 then begin BitBtn1.Click; action:=cafree; end;

if answer=7 then begin action:=cafree; end;

if answer=2 then begin action:=canone; end;

end

else

begin

action:=cafree;

end;

end;

procedure TZakazForm.Edit1Change(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.DateTimePicker1Change(Sender: TObject);

begin

if copy(caption, length(caption), 1) <> '\*' then caption:=caption+'\*';

end;

procedure TZakazForm.BitBtn3Click(Sender: TObject);

var

Excel:variant;

begin

if FileExists(Pchar(ExtractFilePath(ParamStr(0))+'docs\kvitok1.xls')) then DeleteFile(Pchar(ExtractFilePath(ParamStr(0))+'docs\kvitok1.xls'));

CopyFile(Pchar(ExtractFilePath(ParamStr(0))+'docs\kvitok.xls'), Pchar(ExtractFilePath(ParamStr(0))+'docs\kvitok1.xls'), true);

Excel := CreateOleObject('Excel.Application');

Excel.Workbooks.open[ExtractFilePath(ParamStr(0))+'docs\kvitok1.xls'];

Excel.Range['B1']:='Квитанция о приеме в ремонт № '+label11.caption;

Excel.Range['C3']:=dateToStr(DateTimePicker1.Date);

Excel.Range['C4']:=DBLookupCombobox2.ListSource.Dataset.FieldByName('modelname').AsString+' ('+DBLookupCombobox1.ListSource.Dataset.FieldByName('nam').AsString+')';

Excel.Range['C5']:=Edit3.text;

Excel.Range['B8']:=memo1.text;

Excel.Range['B10']:=memo2.text;

Excel.Range['C11']:=Edit1.text+' руб.';

Excel.Range['C13']:=DBEdit1.DataSource.DataSet.FieldByName('fio').AsString+' ('+DbEdit1.dataSource.Dataset. FieldByName('phonenumber').AsString+')';

Excel.Range['C15']:=DBLookupCombobox3.ListSource.Dataset.FieldByName('fio').AsString+' ('+DBLookupCombobox3.ListSource.Dataset.FieldByName('phonenumber').AsString+')';

Excel.visible:=true;

end;

procedure TZakazForm.BitBtn4Click(Sender: TObject);

var

Excel:variant;

begin

if FileExists(Pchar(ExtractFilePath(ParamStr(0))+'docs\akt1.xls')) then DeleteFile(Pchar(ExtractFilePath(ParamStr(0))+'docs\akt1.xls'));

CopyFile(Pchar(ExtractFilePath(ParamStr(0))+'docs\akt.xls'), Pchar(ExtractFilePath(ParamStr(0))+'docs\akt1.xls'), true);

Excel := CreateOleObject('Excel.Application');

Excel.Workbooks.open[ExtractFilePath(ParamStr(0))+'docs\akt1.xls'];

Excel.Range['B1']:='Акт выполненных работ к заказу № '+label11.caption;

Excel.Range['C3']:=dateToStr(DateTimePicker1.Date);

Excel.Range['C4']:=DBLookupCombobox2.ListSource.Dataset.FieldByName('modelname').AsString+' ('+DBLookupCombobox1.ListSource.Dataset.FieldByName('nam').AsString+')';

Excel.Range['C5']:=Edit3.text;

Excel.Range['B8']:=memo2.text;

Excel.Range['C10']:=Edit1.text+' руб.';

Excel.Range['C13']:=DBEdit1.DataSource.DataSet.FieldByName('fio').AsString+' ('+DbEdit1.dataSource.Dataset. FieldByName('phonenumber').AsString+')';

Excel.Range['C15']:=DBLookupCombobox3.ListSource.Dataset.FieldByName('fio').AsString+' ('+DBLookupCombobox3.ListSource.Dataset.FieldByName('phonenumber').AsString+')';

Excel.visible:=true;

end;

procedure TZakazForm.BitBtn5Click(Sender: TObject);

var

Excel:variant;

begin

if FileExists(Pchar(ExtractFilePath(ParamStr(0))+'docs\chek1.xls')) then DeleteFile(Pchar(ExtractFilePath(ParamStr(0))+'docs\chek1.xls'));

CopyFile(Pchar(ExtractFilePath(ParamStr(0))+'docs\chek.xls'), Pchar(ExtractFilePath(ParamStr(0))+'docs\chek1.xls'), true);

Excel := CreateOleObject('Excel.Application');

Excel.Workbooks.open[ExtractFilePath(ParamStr(0))+'docs\chek1.xls'];

Excel.Range['C3']:=label11.caption;

Excel.Range['C4']:=dateToStr(DateTimePicker1.Date);

Excel.Range['C5']:=DBLookupCombobox2.ListSource.Dataset.FieldByName('modelname').AsString+' ('+DBLookupCombobox1.ListSource.Dataset.FieldByName('nam').AsString+')';

Excel.Range['C6']:=Edit3.text;

Excel.Range['C8']:=Edit1.text+' руб.';

Excel.Range['C11']:=DBEdit1.DataSource.DataSet.FieldByName('fio').AsString+' ('+DbEdit1.dataSource.Dataset. FieldByName('phonenumber').AsString+')';

Excel.Range['C13']:=DBLookupCombobox3.ListSource.Dataset.FieldByName('fio').AsString+' ('+DBLookupCombobox3.ListSource.Dataset.FieldByName('phonenumber').AsString+')';

Excel.visible:=true;

end;

end.

**Модуль zakazlist**

unit zakazlist;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, ExtCtrls, DB, ADODB, StdCtrls, ComCtrls, Buttons, Grids, DBGrids;

type

TzakazlistForm = class(TForm)

Panel1: TPanel;

Panel2: TPanel;

DBGrid1: TDBGrid;

GroupBox1: TGroupBox;

GroupBox2: TGroupBox;

GroupBox3: TGroupBox;

GroupBox4: TGroupBox;

BitBtn1: TBitBtn;

BitBtn2: TBitBtn;

BitBtn3: TBitBtn;

Label1: TLabel;

Label2: TLabel;

DateTimePicker1: TDateTimePicker;

DateTimePicker2: TDateTimePicker;

BitBtn4: TBitBtn;

Edit1: TEdit;

BitBtn5: TBitBtn;

Edit2: TEdit;

BitBtn6: TBitBtn;

ADOQuery1: TADOQuery;

DataSource1: TDataSource;

ADOQuery1id: TAutoIncField;

ADOQuery1datepriem: TDateTimeField;

ADOQuery1fio: TWideStringField;

ADOQuery1modelname: TWideStringField;

ADOQuery1datevidach: TDateTimeField;

ADOQuery1phoneIMEI: TWideStringField;

CheckBox1: TCheckBox;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure FormResize(Sender: TObject);

procedure Edit1Change(Sender: TObject);

procedure BitBtn5Click(Sender: TObject);

procedure Edit2Change(Sender: TObject);

procedure BitBtn6Click(Sender: TObject);

procedure ADOQuery1AfterOpen(DataSet: TDataSet);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

procedure CheckBox1Click(Sender: TObject);

procedure DateTimePicker2Change(Sender: TObject);

procedure DateTimePicker1Change(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn1Click(Sender: TObject);

private

procedure CreatezakazForm(const childName, id: string);

{ Private declarations }

public

{ Public declarations }

procedure Setgridcolumnwidths;

end;

var

zakazlistForm: TzakazlistForm;

implementation

Uses Main, DataM, Zakaz;

{$R \*.dfm}

Procedure TzakazlistForm.Setgridcolumnwidths;

var

i,x,dx:integer;

begin

if DBGrid1.Columns.count>1 then

begin

x:=Round((DbGrid1.width-80)/(DbGrid1.Columns.Count-1));

DbGrid1.Columns[0].Width:=40;

For i:=1 to DbGrid1.Columns.Count-1 do

begin

DbGrid1.Columns[i].Width:=x;

end;

end;

end;

procedure TzakazlistForm.FormClose(Sender: TObject;

var Action: TCloseAction);

begin

action:= cafree;

end;

procedure TzakazlistForm.FormResize(Sender: TObject);

begin

Setgridcolumnwidths;

end;

procedure TzakazlistForm.Edit1Change(Sender: TObject);

begin

With AdoQuery1 do

begin

close;

SQL[3]:=' AND klient.fio LIKE "%'+Edit1.Text+'%" ';

open;

end;

end;

procedure TzakazlistForm.BitBtn5Click(Sender: TObject);

begin

Edit1.Clear;

end;

procedure TzakazlistForm.Edit2Change(Sender: TObject);

begin

With AdoQuery1 do

begin

close;

SQL[4]:=' AND zakaz.phoneIMEI LIKE "%'+Edit2.Text+'%" ';

open;

end;

end;

procedure TzakazlistForm.BitBtn6Click(Sender: TObject);

begin

Edit2.Clear;

end;

procedure TzakazlistForm.ADOQuery1AfterOpen(DataSet: TDataSet);

begin

Setgridcolumnwidths;

end;

procedure TzakazlistForm.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if not (Key in [#0..#31, '0'..'9']) then Key := #0;

end;

procedure TzakazlistForm.CheckBox1Click(Sender: TObject);

begin

if checkbox1.Checked then

begin

With AdoQuery1 do

begin

close;

SQL[5]:= ' AND (zakaz.datepriem >= :begindate) AND (zakaz.datepriem < :enddate) ';

Parameters.ParamByName('begindate').Value:=DateTimePicker1.DateTime;

Parameters.ParamByName('enddate').Value:=DateTimePicker2.DateTime;

open;

end;

end

else

begin

With AdoQuery1 do

begin

close;

SQL[5]:= ' ';

open;

end;

end;

end;

procedure TzakazlistForm.DateTimePicker2Change(Sender: TObject);

begin

if checkbox1.Checked then

begin

With AdoQuery1 do

begin

close;

SQL[5]:= ' AND (zakaz.datepriem >= :begindate) AND (zakaz.datepriem < :enddate) ';

Parameters.ParamByName('begindate').Value:=DateTimePicker1.DateTime;

Parameters.ParamByName('enddate').Value:=DateTimePicker2.DateTime;

open;

end;

end

else

begin

With AdoQuery1 do

begin

close;

SQL[5]:= ' ';

open;

end;

end;

end;

procedure TzakazlistForm.DateTimePicker1Change(Sender: TObject);

begin

if checkbox1.Checked then

begin

With AdoQuery1 do

begin

close;

SQL[5]:= ' AND (zakaz.datepriem >= :begindate) AND (zakaz.datepriem < :enddate) ';

Parameters.ParamByName('begindate').Value:=DateTimePicker1.DateTime;

Parameters.ParamByName('enddate').Value:=DateTimePicker2.DateTime;

open;

end;

end

else

begin

With AdoQuery1 do

begin

close;

SQL[5]:= ' ';

open;

end;

end;

end;

procedure TzakazlistForm.BitBtn3Click(Sender: TObject);

begin

DBGRID1.DataSource.DataSet.close;

Dbgrid1.DataSource.DataSet.Open;

end;

procedure TZakazlistForm.CreatezakazForm(const childName, id : string );

var

Child: TZakazForm;

i,k :integer;

f : boolean;

begin

f:=False;

k:=0;

for i:=0 to MainForm.MDIChildCount-1 do

if (MainForm.MDIChildren[i] is TZakazForm) then

begin

f:=True;

k:=i;

end;

if (f) then begin

MainForm.MDIChildren[k].bringToFront;

if MainForm.MDIChildren[k].WindowState=wsMinimized then

MainForm.MDIChildren[k].WindowState:=wsNormal;

end

else

begin //иначе - создаем форму

Child := TZakazForm.Create(ZakazListForm);

if id<>'0' then

begin

With Child.AdoQuery1 do

begin

close;

SQL.clear;

SQL.add('SELECT \* FROM Zakaz WHERE id = '+id);

open;

end;

With Child.AdoQuery2 do

begin

close;

SQL.clear;

SQL.add('SELECT \* FROM phone WHERE id= '+Child.AdoQuery1.FieldByName('phoneid').AsString);

open;

end;

Child.label11.caption:=id;

Child.DBEdit1.DataSource.DataSet.Filter:='id= '+Child.AdoQuery1.FieldByName('klientid').AsString;

Child.DBEdit1.DataSource.DataSet.Filtered:=true;

Child.DBEdit1.DataSource.DataSet.Open;

Child.DBLookupComboBox2.ListSource.DataSet.Open;

Child.DBLookupComboBox2.KeyValue:= Child.AdoQuery1.FieldByName('phoneid').AsVariant;

Child.DBLookupComboBox1.ListSource.DataSet.Open;

Child.DBLookupComboBox1.KeyValue:= Child.AdoQuery2.FieldByName('prodid').AsVariant ;

Child.DBLookupComboBox3.ListSource.DataSet.Open;

Child.DBLookupComboBox3.KeyValue:= Child.AdoQuery1.FieldByName('menedgerid').AsVariant;

Child.DBLookupComboBox4.ListSource.DataSet.Open;

Child.DBLookupComboBox4.KeyValue:= Child.AdoQuery1.FieldByName('ingenerid').AsVariant ;

Child.Edit3.Text:=Child.AdoQuery1.FieldByName('phoneIMEI').AsString;

Child.CheckBox1.Checked:=Child.AdoQuery1.FieldByName('prinyat').AsBoolean;

Child.CheckBox2.Checked:=Child.AdoQuery1.FieldByName('akt').AsBoolean;

Child.CheckBox3.Checked:=Child.AdoQuery1.FieldByName('oplata').AsBoolean;

Child.Memo1.Text:=Child.AdoQuery1.FieldByName('prim').AsString;

Child.Memo2.Text:=Child.AdoQuery1.FieldByName('opisanie').AsString;

Child.DateTimePicker1.DateTime:=Child.AdoQuery1.FieldByName('Datepriem').AsDateTime;

Child.Edit1.Text:=Child.AdoQuery1.FieldByName('summ').AsString;

Child.Caption := childName;

Child.Visible:=true;

end

else

begin

Child.label11.caption:=id;

Child.DBEdit1.DataSource.DataSet.Filter:='id = 0';

Child.DBEdit1.DataSource.DataSet.Filtered:=true;

Child.DBEdit1.DataSource.DataSet.Open;

Child.DBLookupComboBox2.ListSource.DataSet.Open;

Child.DBLookupComboBox2.KeyValue:= -1;

Child.DBLookupComboBox1.ListSource.DataSet.Open;

Child.DBLookupComboBox1.KeyValue:= -1 ;

Child.DBLookupComboBox3.ListSource.DataSet.Open;

Child.DBLookupComboBox3.KeyValue:= -1;

Child.DBLookupComboBox4.ListSource.DataSet.Open;

Child.DBLookupComboBox4.KeyValue:= -1 ;

Child.Edit3.Text:='';

Child.CheckBox1.Checked:=false;

Child.CheckBox2.Checked:=false;

Child.CheckBox3.Checked:=false;

Child.Memo1.Text:='';

Child.Memo2.Text:='';

Child.DateTimePicker1.DateTime:=now;

Child.Edit1.Text:='0';

Child.Caption := childName;

Child.Visible:=true;

end;

end;

end;

procedure TzakazlistForm.BitBtn2Click(Sender: TObject);

begin

CreatezakazForm('Заказклиента №'+DBGrid1.DataSource.DataSet.FieldByName('id').AsString,DBGrid1.DataSource.DataSet.FieldByName('id').AsString);

end;

procedure TzakazlistForm.BitBtn1Click(Sender: TObject);

begin

CreatezakazForm('Новый заказ клиента','0');

end;

end.