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, инженер-программист
_____/ А.А. Лебедев /
«17» июня 2020 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой, д.т.н., с.н.с.
_____/ Б.М. Суховилов /
«_____»_____2020 г.

Разработка программы «Организация школьного питания» для Муниципального бюджетного общеобразовательного учреждения «Средняя (коррекционная) общеобразовательная школа № 128 г. Снежинска, Челябинская область»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.03.2020.317.ВКР

Руководитель, к.т.н., доцент
_____/ В.А. Конов /
«25» июня2020 г.

Автор
студент группы ЭУ – 462
_____/ Д.Е. Карякина /
«25» июня 2020 г.

Нормоконтролер, доцент
_____/ Е.А. Конова /
«17» июня 2020 г.

Челябинск 2020

АННОТАЦИЯ

Карякина Д.Е. Разработка программы «Организация школьного питания» для МБОУ «С(К)ОШ № 128 г. Снежинска, Челябинская область» – Челябинск: ЮУрГУ, ЭУ-462, 70 с., 45ил., 17 табл., библиогр. список –8 наим., 5 прил.

Выпускная квалификационная работа выполнена с целью разработки программы «Организация школьного питания» для МБОУ «С(К)ОШ № 128 г. Снежинск».

В ходе работы проанализирована работа школьного общепита, его организационная структура.

С помощью методологии DFD создана графическая модель деятельности школьного общепита, подробно описаны процессы, выполняемые работниками общепита.

В результате работы автоматизирована работа школьной столовой, реализована работа со справочниками продуктов питания, рецептов и поставщиков продуктов, заполнение меню для учеников школы и её сотрудников. Реализован экспорт данных в документы формата Wordи Excel.

ОГЛАВЛЕНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

1 МБОУ «С(К)ОШ № 128 г. Снежинск» – Муниципальное бюджетное общеобразовательное учреждение «Средняя (коррекционная) общеобразовательная школа № 128 города Снежинск».

2 «ОШП» – «Организация школьного питания».

3 DFD – DataFlowDiagram.

4 MDI – Medium Dependent Interface.

5 MS SQL Server – Microsoft SQL Server.

6 ПО – программное обеспечение.

7 MS Office – Microsoft Office.

ВВЕДЕНИЕ

Существует множество программ, автоматизирующих процессы общественного питания. Такие программы направлены на деятельность кафе, ресторанов, баров и столовых.

Узконаправленных решений для столовых, а именно для школьных столовых, очень мало, возможно потомучто сотрудники предпочитают работать «по-старинке», либо предпочитают использовать общее решение и адаптироваться под него. Однако использовать общие решения не всегда хорошо, ведь в них может не быть необходимой для школы отчетности, могут быть другие стандарты шаблонов для документов и многие другие аспекты, важные для школы.

Существующие узконаправленные решения не имеют возможности редактировать списки продуктов и рецептур, так как они все скачаны из Интернета и защищены от модификаций. Такой подход не всегда верен, ведь скачанные данные могут не соответствовать стандартам, принятым местным школьным комбинатом. Помимо этого, в таких решениях есть множество документации, попросту не использующейся в некоторых школах. Рядовой пользователь может запутаться в таком обилии отчетов.

Говоря о документации, у школьных технологов её действительно много – отчет о расходе продуктов, отчет о состоянии склада, меню-раскладка на неделю, меню-требование на выдачу продуктов и т.п. Все эти документы заполняются вручную в заранее распечатанных шаблонах. Не каждому технологу захочется на постоянной основе заполнять такой объем документации за один рабочий день. К тому же, заполняя документы вручную, не исключены ошибки в расчетах, которые могут привести к ошибочным заказам продуктов или к их некорректному списанию.

Целью дипломного проекта является автоматизация работы столовой для МБОУ «С(К)ОШ № 128 города Снежинск».

Основные задачи дипломного проекта следующие.

- создать и нормализовать базу данных для хранения номенклатур продуктов, рецептов блюд и информации, связанной с движением продуктов внутри столовой.
- разработать программу для работы с базой данных;
- реализовать возможность редактирования данных в программе;
- реализовать многопользовательский режим для программы – разграничение прав пользователей, ввод режима администратора;
- настроить вывод данных из программы в шаблоны документов wordи excel.

В первом разделе представлена постановка задачи дипломного проекта.

Во втором разделе проведен сравнительный анализ существующих систем автоматизации питания.

В третьем разделе описана структура разработанной программы в дипломном проекте.

В четвертом разделе представлена структура базы данных для программы с подробным описанием таблиц, представлений и других элементов базы данных.

В пятом разделе представлена поэтапная разработка программы.

В шестом разделе описаны затраты, необходимые для приобретения рабочего места сотрудников школьной столовой, а также программного обеспечения для работы с программой.

1 ПОСТАНОВКА ЗАДАЧИ

1.1 Требования к разработке программы

Необходимо разработать программу, автоматизирующую работу школьной столовой, основной задачей которой является учет продуктов питания и их движение от поступления на склад до выбытия на кухню. Программа должна рассчитывать количество пришедших и расходуемых продуктов, отображая их в остатках на складе, хранить необходимые для работы школьной столовой документы, справочники и отчеты.

В справочниках программы хранятся следующие документы.

1. Технологические карты, составленные по ГОСТ школьным комбинатом или школьным технологом. Технологическая карта – это рецепт блюда с указанием сборника рецептур, температурного режима, срока годности и состава блюда, т.е. из каких продуктов оно приготавливается[1].

2. Номенклатура продуктов питания с указанием номенклатурного номера и единицы измерения.

3. Информация о поставщиках, содержащая название организации, её адрес, реквизиты (ИНН организации, расчетный счет и т.п.) и контактное лицо для связи.

Поступление продуктов питания должно фиксироваться в приходных накладных от поставщиков с указанием общей информации о поставщиках, названий поступивших продуктов и их количества.

Программа должна составлять план-меню на каждый день для учащихся школы и её сотрудников. При составлении меню автоматически формируются меню-раскладка на конкретную дату с перечнем блюд и их весом в граммах, меню-требование на выдачу продуктов питания, необходимых для приготовления блюд, согласно плану-меню, с указанием количества расходуемого продукта.

Все изменения, связанные с конкретным продуктом, фиксируются в сортовой приходно-расходной карте. В ней содержится общая информация о продукте (номенклатурный номер, название, единица измерения), дата внесения

записи в карту, какое изменение с продуктом произошло (поступление или списание) и какое его количество ушло на это изменение. Списание проводится по обоим видам меню – для учеников школы и для её сотрудников. По окончании месяца подводятся итоги по всем изменениям, т.е. отдельно суммируются поступление и списание продукта.

По окончании недели формируется отчет (ведомость) о расходе продуктов питания, согласно которому составляется требование на дозаказ продуктов питания (отчет о заказе продуктов).

1.2 Пользователи программы

Основные пользователи программы – технолог и кладовщик. Они могут работать как с одного компьютера, так и с двух.

Технолог может редактировать все документы и справочники, настраивать макеты выводимых отчетов, редактировать информацию о самих пользователях. В обязанности технолога входит заполнение (загрузка) технологических карт, заполнение информации о поставщиках, составление плана-меню для учеников школы и её сотрудников, заполнение меню-требования на выдачу продуктов питания, заполнение плана-меню на день или на неделю.

Кладовщик имеет ограниченные права, то есть он может редактировать справочник номенклатур продуктов питания и накладные на поставку, все остальные документы доступны только для чтения. В обязанности кладовщика входит заполнение информации о продуктах, заполнение накладных на их поставку, формирование отчета о расходе продуктов и составление требования на дозаказ продуктов.

На данный момент обязанности кладовщика выполняет технолог. В связи с этим у него имеется двойная нагрузка, так как ему, помимо своих обязанностей, необходимо следить «вживую» за складом, самостоятельно считать продукты в остатках и в расходе, принимать поставки и лично подписывать накладные от поставщиков.

В обязанности технолога также входит заполнение документации. Однако сами документы заполняются не на компьютере, а вручную. В качестве примеров ручного заполнения – сортовая приходно-расходная карта (рисунок 1) и меню-требование на выдачу продуктов питания (рисунок 2).

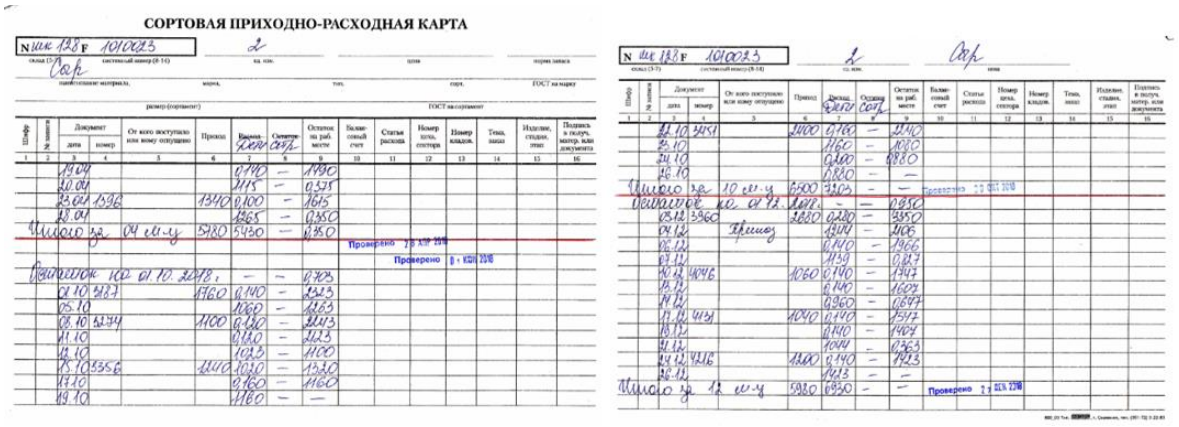


Рисунок 1 – Скан сортовой приходно-расходной карты

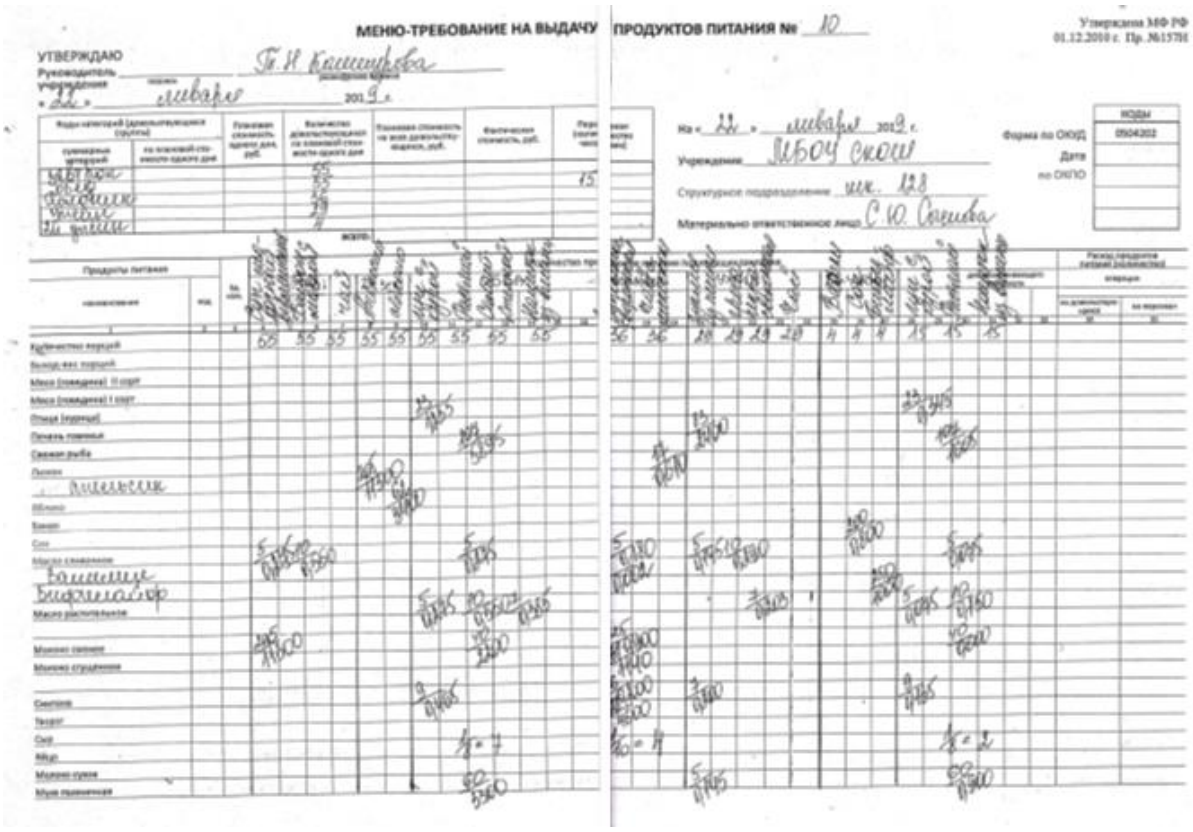


Рисунок 2 – Скан документа «Меню-требование на выдачу продуктов питания»

Вручную заполняется большой объем информации, в основном численной. При заполнении документов можно легко допустить ошибку, например, в списании, что может привести к некорректному заказу продукта.

Выводы по разделу один

Выведена постановка задачи для разработки программы, выделены основные цели разработки:

- упростить работу технолога, «разделив» программу между двумя пользователями – «технолог» и «кладовщик». у каждого пользователя должен быть свой набор функций, соответствующий их обязанностям;
- автоматизировать заполнение документов с большим объемом информации;
- реализовать складской учет продуктов питания. пользователь должен видеть остатки продуктов на складе, формировать необходимую отчетность исходя из информации об остатках.

2 АНАЛИЗ СУЩЕСТВУЮЩИХ СИСТЕМ

2.1 «Вижен-Софт:Питание в школе»

Программный продукт, который предназначен для автоматизации планирования и формирования рационального питания учеников школы[2].

Программа содержит готовый справочник продуктов с указанием пищевой ценности, витаминов и минералов, процента отходов при холодной обработке, а также готовую обширную картотеку, содержащую более 1800 блюд, с указанием рецептуры, технологии приготовления и ссылки на сборник рецептур.

«Вижен-Софт:Питание в школе» позволяет формировать меню на каждый день для всех категорий питающихся, с последующим выводом на печать следующих документов:

1. Меню для учащихся.
2. Требования-накладные на отпуск продуктов со склада.
3. Дневной заборный лист.
4. Калькуляционные карточки.
5. Акт реализации.
6. Меню-требование на выдачу продуктов питания формата А4 и А3.
7. Меню-раскладка.

Меню может формироваться под требуемое количество продуктов, в том числе «под остаток», т.е. собрать оставшиеся продукты на складе под ноль, с одновременной автоматической корректировкой продуктов в блюдах и пересчетом выхода блюд. После формирования меню можно произвести расчет средней стоимости питания (стоимости дня).

Программа позволяет вести складской учет продуктов питания, в том числе по договорам с поставщиками и формировать следующие аналитические отчеты:

1. Ведомость выполнения договора поставки продуктов питания.
2. Ведомость по поставщикам.
3. Отчет по остаткам на складе.

4. Оборотная ведомость за произвольный период.

5. Журнал учета продуктов питания, с указанием начального остатка, поступления и списания продуктов, и конечного остатка.

6. Движение продуктов питания можно отслеживать с помощью такой документации, как:

a. бракеражный журнал сырой и готовой продукции;

b. ведомость выполнения норм продуктового набора (накопительная ведомость);

c. ведомость выполнения норм потребления пищевых веществ, витаминов и минералов.

Программа позволяет выгружать информацию о движении продуктов в бухгалтерскую программу «1С: Бухгалтерия 7.7», «1С: Бухгалтерия 8.2», «1С: Бухгалтерия 8.3» и «Парус-Бюджет».

2.2 «1С:Предприятие 8. Общепит»

Решение для автоматизации деятельности предприятий питания любых форматов и концепций. Конфигурация позволяет автоматизировать рабочие места главного бухгалтера, бухгалтеров по различным участкам учета, бухгалтера-калькулятора, технолога, кладовщика[3].

«1С:Общепит» учитывает специфику предприятий общественного питания и включает в себя следующие возможности.

1. Обмен данными с ресторанно-кассовыми (фронт-офисными) системами.

2. Поддержка требований законодательства по учету алкогольной продукции для предприятий общественного питания.

3. Ведение списка рецептур, составление калькуляций (Технологических карт), с возможностью учета предварительной проработки «фирменных» блюд. Хранение рецептур в виде документов с выбранной хозяйственной операцией (приготовление, разделка).

4. Загрузка списка ингредиентов, блюд и рецептур с химико-энергетическими показателями (с заполнением соответствующих справочников и документов) из электронного онлайн-сборника рецептур.

5. Использование нескольких рецептур полуфабрикатов и блюд с возможностью выбора нужной в момент приготовления. Для сложных рецептур, реализован механизм учета "блюдо в блюде" с неограниченным количеством уровней вложенности.

6. Ведение списка взаимозаменяемых продуктов и его автоматическое использование при списании в производство и формировании калькуляционных карточек.

«1С:Общепит»позволяет вести учет стандартных складских операций: поступления, перемещения, инвентаризации и списания материально-производственных запасов (МПЗ). Ведение количественного или количественно-суммового учета МПЗ в разрезе складов также входит в возможности программы.

«1С:Общепит» учитывает приготовление полуфабрикатов и блюд, расход специй, калорийность и пищевую ценность продуктов, а также количество и сумму списанных в производство ингредиентов, согласно заведенным ранее рецептурам. Программа может оценивать нехватку ингредиентов, наличие необходимых полуфабрикатов и требуемое количество ингредиентов.

В особенности бухгалтерского учета входит учет на предприятиях с различными формами хозяйствования, розничных складов, товаров, материалов и готовой продукции, а также учет отложенной оплаты при оформлении розничных продаж.

«1С:Общепит» составляет отчетность по прибыли от реализации, по товарам, по анализу выпуска продукции, расходу продуктов и специй, а также по остаткам и оборотам товаров и блюд из дополнительно документации имеются заборный лист и калькуляция за период.

Конфигурация предоставляет на печать необходимый набор унифицированных форм: Калькуляционная карточка, План-Меню, Требование в

кладовую, Накладная на отпуск товара, Закупочный акт, Дневной заборный лист, Акт о реализации и отпуске изделий кухни, Акт о реализации изделий кухни за наличный расчет, Ведомость учета движения продуктов и тары на кухне, Ведомость учета остатков продуктов, Контрольный расчет расхода продуктов, Акт разделки мясо-сырья на полуфабрикаты, Инвентаризационная опись, Сличительная ведомость, Накладная на внутреннее перемещение, Акт о списании товаров, Товарный отчет.

Конфигурация «1С:Общепит»связана с сервисом «1С-Отчетность», который позволяет отправлять регламентированную отчетность в контролирующие органы.

2.3 «1С:Предприятие 8. Ресторан»

Это решение, с помощью которого могут быть автоматизированы одиночные и сетевые предприятия любых форматов и концепций – рестораны, кафе, бары, столовые, подразделения питания гостинично-ресторанных комплексов, развлекательных центров и другие предприятия питания[4]. Также решение подходит для автоматизации процесса обслуживания гостей в различных оздоровительных учреждениях.

Конфигурация может осуществлять ввод заказов посетителей с использованием сенсорного интерфейса в зависимости от формата обслуживания, особенностей оборудования и прав пользователя. Для работы с посетителями используется система бронирования столиков с указанием контактной информации и различных параметров резервирования.

В программе имеются графические схемы обслуживания посетителей, позволяющие официантам быстро ориентироваться в системе, выбирая нужный столик:

- table-service – классическое обслуживание с официантом, оформление предварительного заказа на столик;
- fast-food – быстрые продажи, обслуживание без официанта;

- смешанная – комбинация предыдущих пунктов, например, в каждом зале ресторана может быть установлена своя схема обслуживания посетителей.

Также встроен редактор плана заведения, в случае если в нём имеются несколько залов.

«1С:Ресторан» позволяет использовать несколько вариантов меню с ручным или автоматическим выбором при оформлении или корректировке заказа. Меню доступно по дате, времени, дням недели, а также могут устанавливатьсяразные цены на блюда для каждого вида меню.Подбор товаров и блюд в заказы может осуществляться на POS-терминалах при помощи "сенсорного" меню, "горячими" клавишами, по коду или штрих-коду.В случае переноса конкретных блюд или всего заказа на другие столы возможна корректировка заказов или разделение предварительного счета между гостями. В случае отмены заказа (частичной или полной) формируется отчет о причинах удаления при её предварительном указании. Все изменения, связанные с заказами, идут на автоматическую печать в виде чеков на сервис-принтеры, которые располагаются в местах приготовления блюда.

Оплата заказов может осуществляться наличным, безналичным и комбинированным (наличный вместе с безналичным) расчетами, талонами на питание и картами сотрудников. В конфигурации также предусмотрена системалояльности посетителей. В программе можно настроить различные дисконтные схемы: скидки по дате и времени, скидки на позицию или сумму счета, акции по типу «Три равно двум» и т.п.

На вывод имеется большое количество аналитических отчетов, причем, многие из них можно сформировать непосредственно с сенсорного экрана POS-терминала. Например, сменный отчет по блюдам и по кассирам, отчет о реализации товаров блюд и услуг, отчеты по скидкам, отменам и многие другие.

Конфигурация может быть интегрирована с ЕГАИС, «1С:Предприятие 8. Общепит»и «1С:Предприятие 8. Отель».

2.4 Сравнительный анализ систем

Сравнительный анализ систем представлен в таблице 2.1.

Таблица 2.1– Сравнительная таблица характеристик анализируемых программ

Критерии	«Вижен-Софт:Питание в школе»	«1С:Предприятие 8. Общепит»	«1С:Предприятие 8. Ресторан»	«Организация школьного питания»
Отсутствие излишней функциональности	+	–	+	+
Ведение складского учета	+	+	–	+
Составление сортовой приходно-расходной карты по продукту	–	–	–	+
Дешевизна программы	–	–	–	+
Загрузка собственных технологических карт и их редактирование	–	–	–	+
Простота интерфейса пользователя	–	+	–	+
Узконаправленное решение	+	–	–	+
Возможность доработки программы	–	+	–	+
Передача xml-отчетов для 1С:Бухгалтерии	+	–	–	+

Конкурентным преимуществом «ОШП» является возможность создания сортовой приходно-расходной карты – это документ, отражающий движение конкретного продукта по всем инстанциям (от поставки до выбытия на кухню). Пользователю программы нет необходимости заполнять её самостоятельно, т.к.

документ заполняется с помощью специального запроса из базы данных, где отражаются все изменения, происходящие с продуктом. Сортовая приходно-расходная карта содержит общую информацию о продукте (номенклатурный номер, единицу измерения, и т.д.), кто поставил продукт и в каком количестве, а также сколько продукта ушло на приготовление блюд. Все записи фиксируются по дате изменения, происходящих с продуктом, и номером документа, зарегистрировавшим данное изменение (выбытие на кухню, накладная на поставку и т.д.).

В «ОШП» имеется возможность загружать собственные технологические карты, а также их редактирование. Во всех предложенных для сравнения системах технологические карты выгружены из различных онлайн-сборников. Но, так как «ОШП» работает со школьной столовой, в ней необходимо хранить карты, составленные по ГОСТу школьным комбинатом, либо школьным технологом.

«ОШП» и её пользователи напрямую не работают с денежными средствами, поэтому программа может выгружать отчеты о расходе продуктов и о заказе продуктов у поставщиков в формате XML для отправки бухгалтерам. Данное решение обосновано ещё и тем, что бухгалтерия может находиться вне здания школы, поэтому есть необходимость отправки отчетности через Интернет.

В связи с тем, что программа не взаимодействует с деньгами, в ней отсутствует излишняя функциональность, под которой понимается бухгалтерский и налоговый учеты. Кладовщику и технологу не нужно работать с деньгами, потому что в первую очередь они ведут складской учет продуктов питания и контролируют их движение. Всё, что касается стоимости продуктов и общей суммы их расхода или заказа, оформляет бухгалтерия.

Дешевизна «ОШП» обоснована тем, что она является узконаправленным решением, имеет простой и понятный пользователю интерфейс, не интегрирована с другими приложениями или сервисами и сосредоточена только на складском учете и движении продуктов.

Вывод по разделу два

Проведен анализ существующих систем. По результатам анализа сделан вывод о целесообразности разработки нового программного продукта.

3 СТРУКТУРА ПРОГРАММЫ

Структура программы описана с помощью приложения ERwin Process Modeler. Модель разработана по методологии DFD.

На рисунке 3 изображена диаграмма, показывающая, как должны взаимодействовать программа и её пользователи.

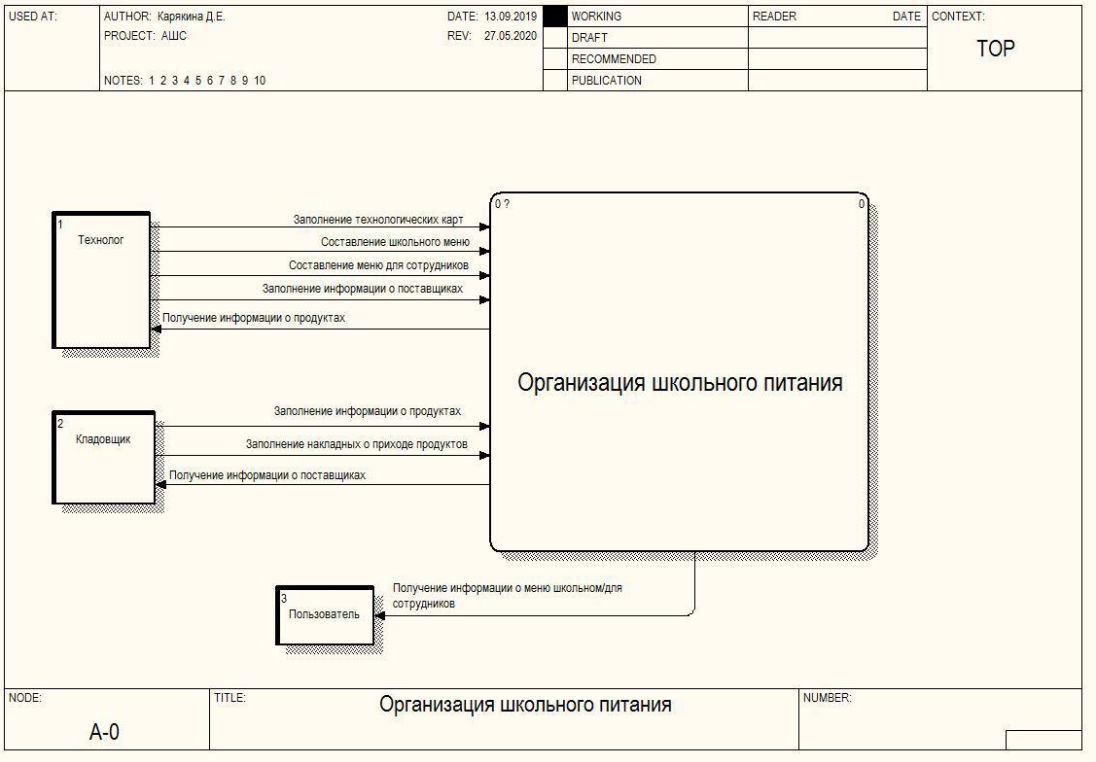


Рисунок 3 – DFDдиаграмма «Организация школьного питания»

Блок «Организация школьного питания» представляет саму программу и характеризует её общий функционал.

С программой работают три пользователя: технолог, кладовщик и администратор.

В обязанности технолога входят: составление школьного меню и меню для сотрудников школы, заполнение или редактирование технологических карт, заполнение информации о поставщиках, а также редактирование шаблонов необходимых для отчетности документов. Данному пользователю доступна настройка программы и настройка прав пользователей.

Задача кладовщика – контролировать остатки продуктов и заносить накладные о поставке в базу данных. Однако, кладовщик не может редактировать

технологические карты, составлять меню. Всё перечисленное доступно пользователю только для просмотра. В настройках программы пользователь может изменять только те настройки, которые ему разрешены правами, также он может изменять имя пользователя и пароль.

Под «Пользователем» в диаграмме выступают ученики школы и её сотрудники, которые получают информацию о текущем меню на день или на неделю.

Администратор не показан в диаграмме, так как он может выполнять все задачи, указанные для других пользователей. Помимо этого, он может видеть и изменятьпароли пользователей, удалять данные, которые не могут удалить технолог и кладовщик. Администратор может просматривать недоступные для других пользователей таблицы и редактировать их.

Функции программы разделены на пять блоков, в которых содержится общая информация об интерфейсе формы, какие команды в ней используются, и какая информация в ней отображается.

На рисунке 4 представлены основные функции программы.

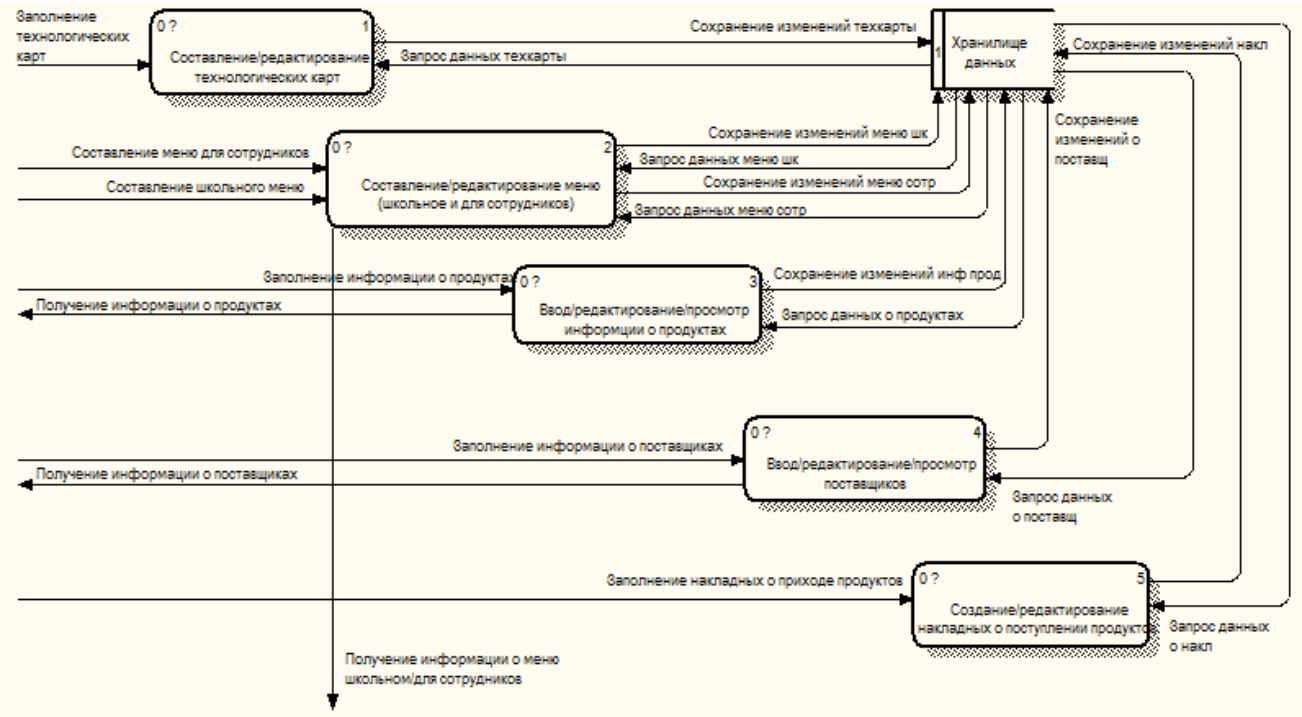


Рисунок 4 – DFDдиаграмма функционала программы

Блок «Хранилище данных» в диаграмме – это база данных, подключенная к программе. Она необходима для наглядного отображения сохранения данных и вывода данных.

3.1 Технологические карты

В блоке «Создание/редактирование технологических карт» описывается процесс создания и редактирования технологических карт.

На рисунке 5 изображена диаграмма блока «Составление/редактирование технологических карт».

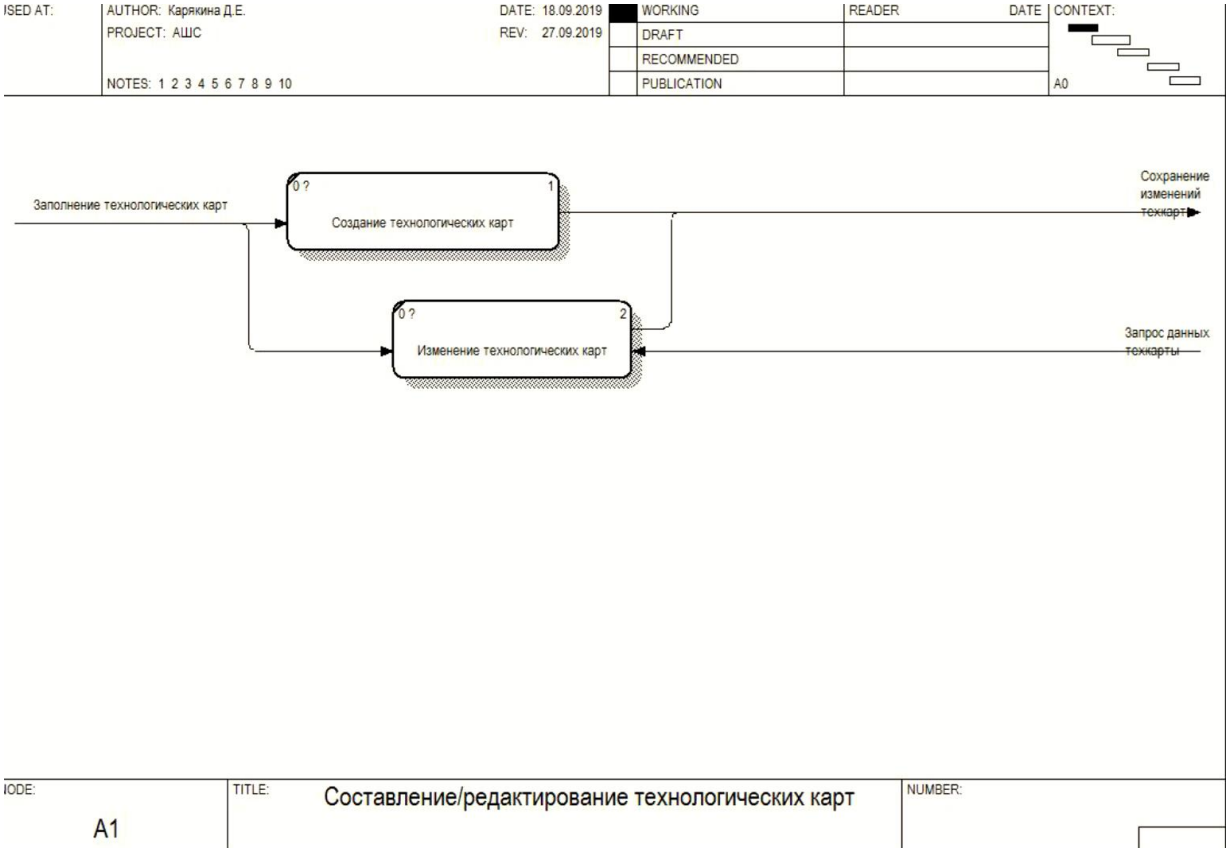


Рисунок 5– DFDдиаграмма «Составление/редактирование технологических карт»

В диаграмме показан процесс заполнения данных, сохранения этих данных и запроса данных из базы для изменения.

3.2 Составление меню для учеников школы и её сотрудников

На рисунке 6 изображена диаграмма «Составление/редактирование меню».

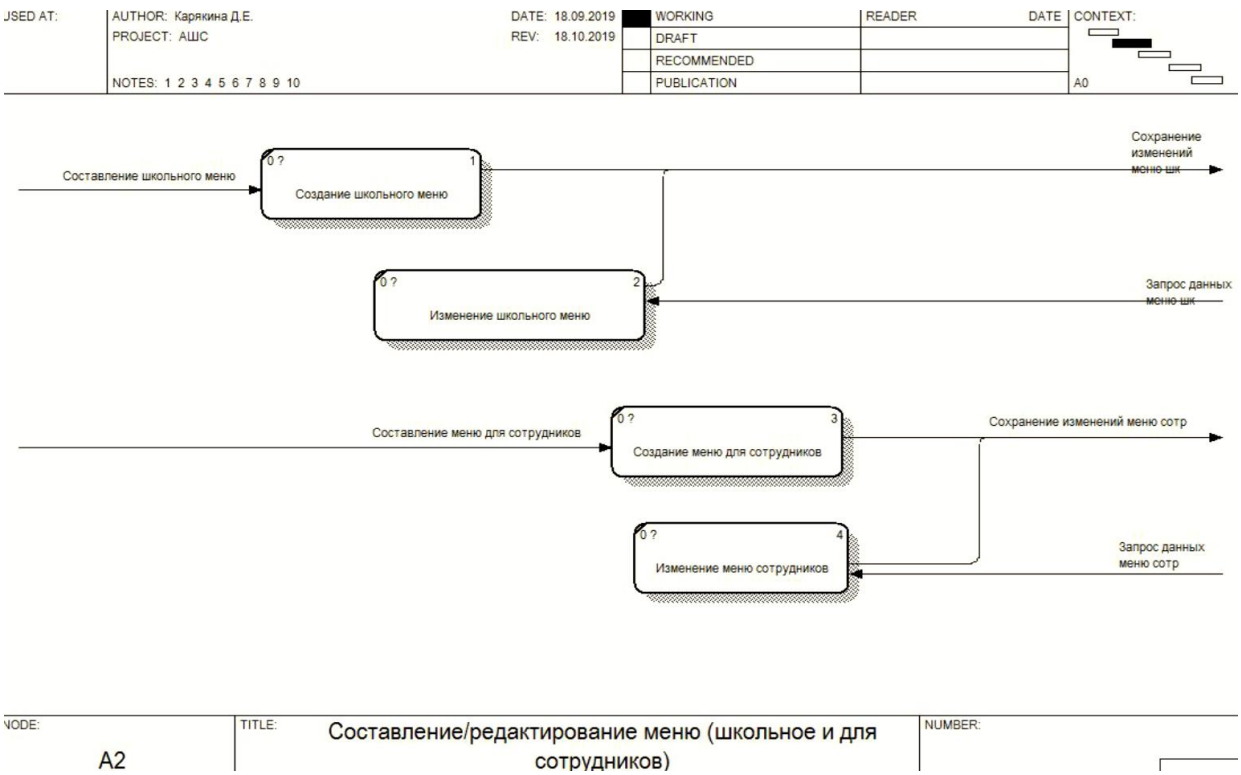


Рисунок 6 – DFDдиаграмма «Составление/редактирование меню»

В представленной диаграме показано, каким образом сохраняются и запрашиваются данные, а также процесс заполнения данных.

3.3Информация о продуктах питания

На рисунке 7 изображена диаграмма, показывающая процесс добавления и редактирования справочника продуктов питания.

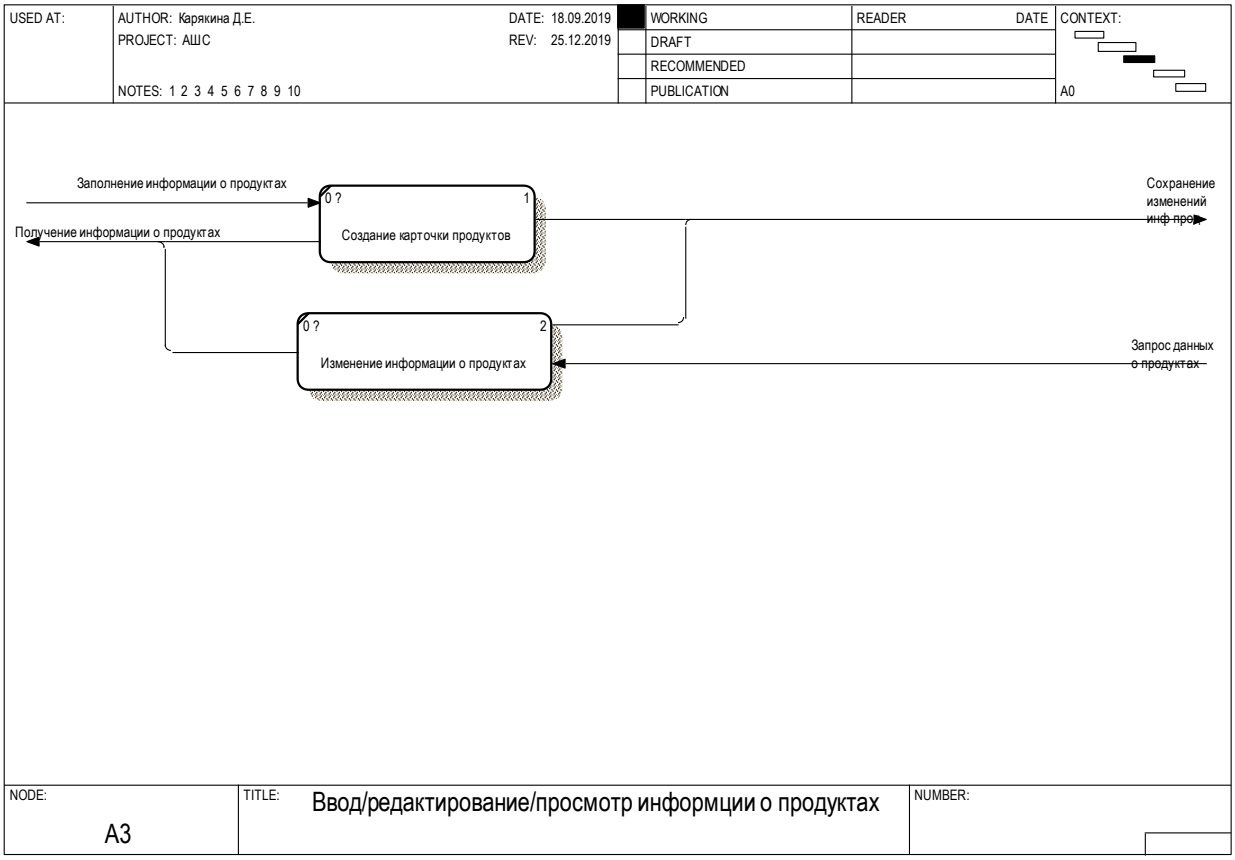


Рисунок 7 – DFDдиаграмма «Ввод/редактирование/просмотр продуктов»

3.4Информация о поставщиках продуктов питания

На рисунке 8 изображена диаграмма «Ввод/редактирование/просмотр поставщиков». Эта диаграмма показывает, как добавляются и запрашиваются данные по поставщикам из базы.

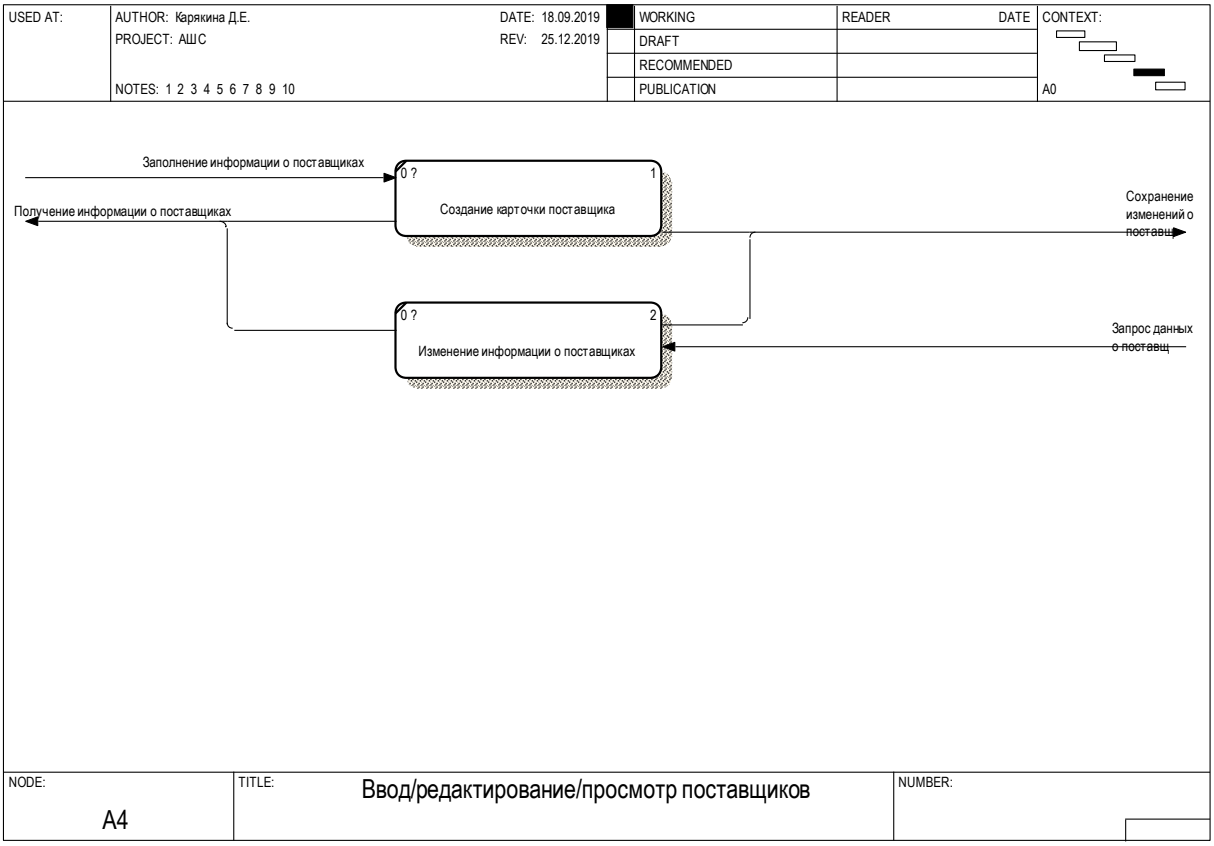


Рисунок 8 – DFDдиаграмма «Ввод/редактирование/просмотр поставщиков»

3.5 Накладные на поставку продуктов питания

На рисунке 9 изображена диаграмма «Создание/редактирование накладных о поступлении продуктов», показывающая как заполняются и добавляются данные в базу, а также как они запрашиваются из базы данных.

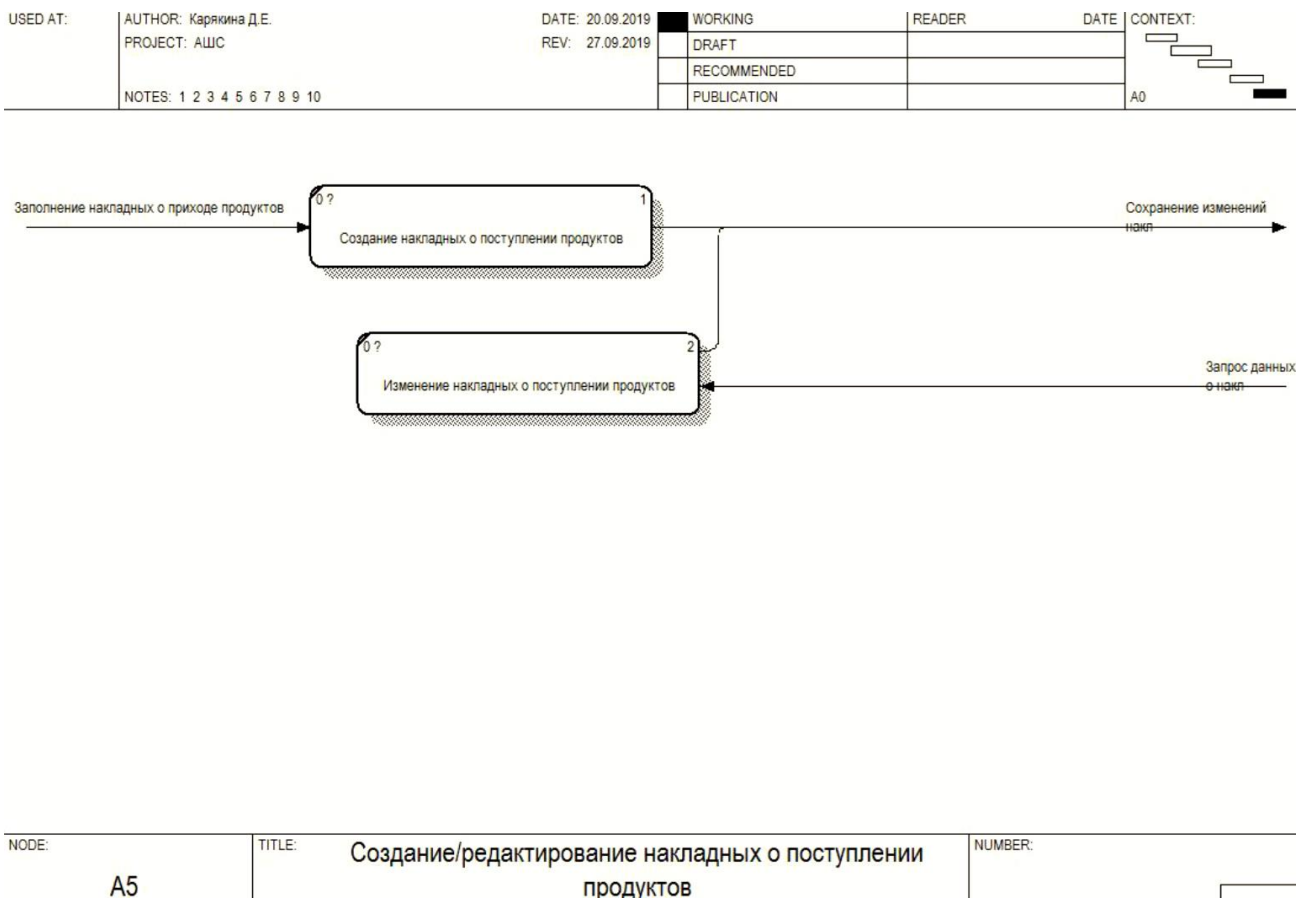


Рисунок 9 – DFDдиаграмма «Создание/редактирование накладных о поступлении продуктов»

Выводы по разделу три

Разработаны DFDдиаграммы, показывающие, как работает программа и каким образом данные добавляются и запрашиваются из базы данных. Описаны диаграммы отдельных компонентов программы, с которыми работают пользователи.

4 СТРУКТУРА БАЗЫ ДАННЫХ

База данных для программы хранится на локальном сервере Microsoft SQL Server, доступ к которой осуществляется с помощью программы SQL Server Management Studio.

База данных имеет табличную структуру со связанными таблицами, т.е. база данных является реляционной.

На рисунке 10 изображена схема базы данных.

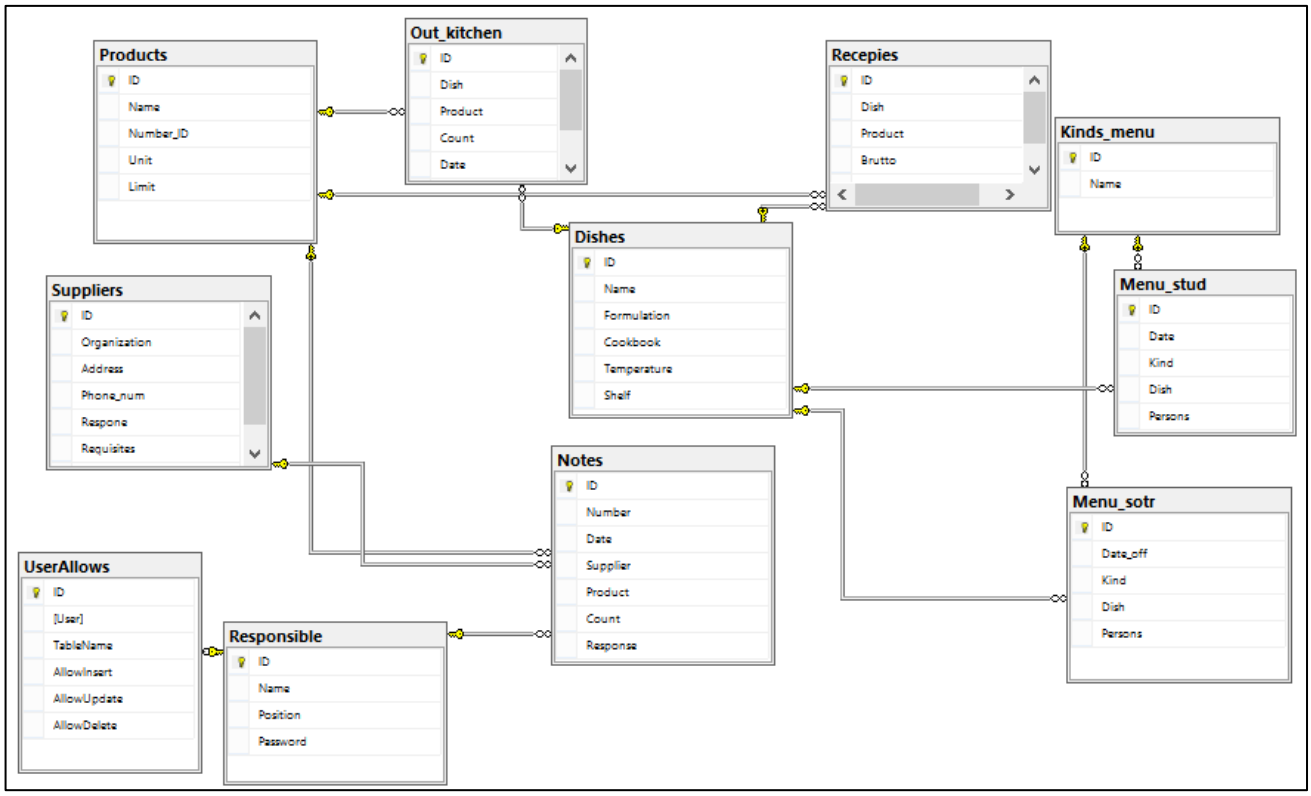


Рисунок 10 – Схема данных

4.1 Таблицы базы данных

Таблицы являются объектами, которые содержат все данные в базах данных. В таблицах данные логически организованы в виде строк и столбцов по аналогии с электронной таблицей[5].

DB_changes – таблица, отображающая изменения, происходящие в выбранных таблицах. Данные в таблицу добавляются с помощью триггеров, прописанных для выбранных таблиц. Структура таблицы описана в таблице 4.1

Таблица 4.1 – Структура таблицы DB_Changes

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Date_of_change	Date(10)	Дата изменения таблицы
Changed_table	Nvarchar(MAX)	Измененная таблица
Action	Nvarchar(MAX)	Действие, произошедшее с таблицей – Insert, Update или Delete

Dishes – таблица технологических карт (блюд). Структура таблицы описана в таблице 4.2.

Таблица 4.2 – Структура таблицы Dishes

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Name	Nvarchar(MAX)	Название блюда
Formulation	Nvarchar(MAX)	Номер рецептуры блюда
Cookbook	Nvarchar(MAX)	Сборник рецептур
Temperature	Nvarchar(MAX)	Температура блюда
Shelf	Int32	Срок годности блюда (в часах)

Kinds_menu – категории меню (завтрак, обед и т.п.). Структура таблицы описана в таблице 4.3.

Таблица 4.3 – Структура таблицы Kinds_menu

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Name	Nvarchar(MAX)	Название категории меню

Menu_sotr – таблица для хранения меню для сотрудников школы. Описание структуры таблицы находится в таблице 4.4.

Таблица 4.4 – Структура таблицы Menu_sotr

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Date_off	Date(10)	Дата реализации меню
Kind	Int32	Категория меню
Dish	Int32	Код блюда
Persons	Int32	Количество питающихся

Menu_stud – таблица для хранения меню для школьников. Структура таблицы описана в таблице 4.5.

Таблица 4.5 – Структура таблицы Menu_stud

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Date_off	Date(10)	Дата реализации меню
Kind	Int32	Категория меню
Dish	Int32	Код блюда
Persons	Int32	Количество питающихся

Notes – таблица для хранения накладных на поставку продуктов питания. Описание структуры таблицы представлено в таблице 4.6.

Таблица 4.6 – Структура таблицы Notes

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Number	Nvarchar(MAX)	Номер накладной
Date	Date(10)	Дата поставки
Supplier	Int32	Код поставщика
Product	Int32	Код продукта
Count	Int32	Количество пришедшего продукта
Respone	Int32	Код сотрудника, принявшего накладную

Out_kitchen – таблица, отображающая списанные продукты по виду меню. Данные добавляются с помощью триггеров, прописанных длятаблицMenu_studиMenu_sotr. Описание структуры таблицы Out_kitchenпредставлено в таблице 4.7.

Таблица 4.7 – Структура таблицы Out_kitchen

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Dish	Int32	Код блюда
Product	Int32	Код продукта
Count	Int32	Количество списанного продукта
Date	Date(10)	Дата списания
Kind_menu	Nvarchar(MAX)	Вид меню, по которому прошло

		списание

Products – таблица продуктов питания. Структура таблицы описана в таблице 4.8.

Таблица 4.8 – Структура таблицы Products

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Name	Nvarchar(MAX)	Название продукта
Number_ID	Int32	Номенклатурный номер
Unit	Nvarchar(MAX)	Единица измерения продукта
Limit	Int32	Минимальный остаток продукта

Recepies – таблица для отображения рецепта блюда. Структура таблицы описана в таблице 4.9.

Таблица 4.9 – Структура таблицы Recepies

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Dish	Int32	Код блюда
Product	Int32	Код продукта
Brutto	Int32	Брутто – количество продукта до обработки
Netto	Int32	Нетто – количество продукта после обработки

Responsible – таблица сотрудников (ответственных лиц). Описание структуры таблицы представлено в таблице 4.9.

Таблица 4.9 – Структура таблицы Responsible

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Name	Nvarchar(MAX)	Имя сотрудника, оно же Имя пользователя
Position	Nvarchar(MAX)	Должность
Password	Varbinary(MAX)	Пароль

Suppliers– таблица поставщиков, работающих с школой. Описание структуры таблицы представлено в таблице 4.10.

Таблица 4.10 – Структура таблицы Suppliers

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
Organization	Nvarchar(MAX)	Название организации
Address	Nvarchar(MAX)	Адрес организации
Phone_num	Int32	Телефон организации
Respone	Int32	Контактное лицо организации
Requisites	Varbinary(160)	Реквизиты организации
Active	Bit	Статус поставщика

UserAllows– таблица прав пользователей. Структура таблицы описана в таблице 4.11.

Таблица 4.11 – Структура таблицы UserAllows

Название столбца	Тип данных столбца	Описание столбца
ID	PRIMARYKEY, int32	Ключевой столбец таблицы
User	Int32	Код пользователя
TableName	Bit	Название таблицы
AllowInsert	Bit	Разрешение добавления
AllowUpdate	Bit	Разрешение изменения
AllowDelete	Bit	Разрешение удаления

4.2 Триггеры таблиц

Триггер – это особая разновидность хранимой процедуры, которая автоматически выполняется при возникновении события на сервере базы данных[6]. Триггеры выполняются, когда пользователь пытается изменить данные с помощью событий языка обработки данных (Data Manage Language, сокращенно DML). Событиями DML являются процедуры INSERT, UPDATE или DELETE, применяемые к таблице или представлению. Эти триггеры срабатывают при запуске любого допустимого события независимо от наличия и числа затронутых строк таблицы.

Для таблиц Menu_stud и Menu_sotr прописаны триггеры на все три процедуры. Задача триггеров для этих таблиц – провести списание продуктов, записав данные списания в таблицу Out_kitchen, а также записать изменение, произошедшее с таблицей, в DB_Changes. При срабатывании триггера на процедуру UPDATE в таблице Out_kitchen удаляются старые записи, отфильтрованные по коду блюда, и добавляются новые записи.

Для Out_kitchen создан триггер, записывающий все изменения, происходящие в таблице.

Для таблицы Disheссоздан триггер, срабатывающий при удалении блюда из таблицы. Триггер с помощью курсора находит все записи из таблицы Recepiesпро коду блюда и удаляет их.

4.3Представления таблиц

Представление – это виртуальная таблица, содержимое которой определяется запросом[7]. Как и таблица, представление состоит из ряда именованных столбцов и строк данных. Строки и столбцы данных извлекаются из таблиц, указанных в определяющем представление запросе и динамически создаваемых при обращениях к представлению.

Подробное описание представлений в таблице 4.12.

Таблица 4.12 – Описание представлений базы данных

Название представления	Таблицы или представления, участвующие в создании	Описание представления
View_Dishes	Dishes	Представление таблицы Dishes
View_Kinds_menu	Kinds_menu	Представление таблицы Kinds_menu
View_Menu_sotr	Menu_sotr	Представление таблицы Menu_sotr
View_Menu_stud	Menu_stud	Представление таблицы Menu_stud
View_Notes	Notes	Представление таблицы Notes
View_Out_kitchen	Out_kitchen	Представление таблицы Out_kitchen

View_Products	Products	Представление таблицы Products
View_Recepies	Recepies	Представление таблицы Recepies
View_Responsible	Responsible	Представление таблицы Responsible
View_Suppliers	Suppliers	Представление таблицы Suppliers

Продолжение таблицы 4.12

View_UserAllows	UserAllows	Представление таблицы UserAllows
View_SumIn	Notes Products	Данное представлениепоказывает информацию из таблицы Products
		и выводит суммарный приход по продукту. Сумма рассчитывается из количества пришедшего продукта в накладной.
View_SumOut	Out_kitchen Products	Аналогично представлению View_SumIn, выводится основная информация из Products и выводится суммарное списание продукта. Сумма рассчитывается из количества списанного продукта по технологической карте (блюду)
View_Store	View_SumIn View_SumOut	Данное представление является отображением склада продуктов питания. Оно содержит основную информацию о продукте (взято из View_SumIn, хотя аналогичное можно взять и из View_SumOut), а также остаток продукта. Остаток рассчитывается из разности суммы прихода и суммы расхода. Если сумма расхода нулевая, то остаток равен сумме прихода.
View_Income	Notes Product	Представление аналогично View_SumIn, однако в этом случае сумма прихода не рассчитывается.
View_Outcome	Out_kitchen Product	Представление аналогично View_SumOut, однако в этом случае сумма расхода не рассчитывается. Дополнительно указывается вид меню как

		таблица Kind.
View_Sort_Card	View_Income View_Outcome	Представление является объединением View_Income иView_Outcome с помощью процедуры UNION. Дополнительно добавлен столбец «Вид», который

Окончание таблицы 4.12

		помогает разделить информацию в представлении. Если данные в строке взяты из View_Income, то в столбце «Вид» в этой строке ибудет значение «Приход», а если наоборот, то «Расход».
View_Weight	Dishes Recepies	Представление выводит общий вес блюда, высчитанный из суммы нетто по блюду.
Admin_ SelPwdEncryp	Responsible	Представление для администратора – показ дешифрованных паролей.

4.4Хранимые процедуры

Хранимая процедура в SQL Server – это группа из одной или нескольких инструкций Transact-SQL, представляет собой компонент Database Engine [8]. Процедуры аналогичны конструкциям в других языках программирования, поскольку обеспечивают следующее.

- обрабатывают входные параметры и возвращают вызывающей программе значения в виде выходных параметров;
- содержат программные инструкции, которые выполняют операции в базе данных, включая вызов других процедур;
- возвращают значение состояния вызывающей программе, таким образом передавая сведения об успешном или неуспешном завершении (и причины последнего).

Для всех таблиц созданы процедуры INSERT, UPDATE, DELETE. Они необходимы для работы с данными в приложении.

4.5 Безопасность базы данных

Для входа в SQL Server и работы с базой данных созданы два пользователя – Администратор (Admin) и Технолог.

Администратор имеет роль db_owner, т.е. он может выполнять все действия по настройке и обслуживанию базы данных, а также удалять базу данных в SQL Server. Вход администратора защищен паролем.

Технолог имеет несколько ролей.

- db_datareader – пользователь может просматривать данные в таблицах, но не может изменять эти данные;
- public – пользователь может просматривать таблицы, представления, хранимые процедуры, но не может увидеть сертификаты безопасности, ассиметричные и симметричные ключи.

В таблице Responsible указаны пользователи, которые будут работать с программой и базой данных. Так как в данной таблице хранятся их пароли, есть необходимость в их шифровании. Для этого был создан ассиметричный ключ с «строгим» паролем, который защищает закрытый ключ.

Асимметричный ключ состоит из закрытого ключа и соответствующего открытого ключа. Каждый ключ может расшифровать данные, зашифрованные другим. Асимметричное шифрование и дешифрование относительно ресурсоемки, но они обеспечивают более высокий уровень безопасности, чем симметричное шифрование.

Шифрование происходит следующим образом:

1. Создается пользователь с именем для входа и должностью или изменяются данные уже созданного пользователя.
2. Пользователь задает новый пароль.
3. В прописанной процедуре добавления и изменения данных пользователя в таблицу Responsible прописана функция шифрования ENCRYPTBYASYMKEY. В этой функции указывается ассиметричный ключ и параметр, который необходимо зашифровать.

Если пользователь, не имеющий прав администратора, захочет посмотреть пароль в таблице Responsible, то он увидит набор битовых символов, а не сами пароли.

Администратору известен пароль, который защищает закрытый ключ, поэтому только ему доступен просмотр зашифрованных паролей. Для этого создана хранимая защищенная процедура с оператором SELECT с использованием функции дешифрования DECRYPTBYASYMKEY, которая дешифрует пароли пользователей только при указании пароля от закрытого ключа. Таким образом, простому пользователю не удастся посмотреть пароли без знания ключевого пароля.

Выводы по разделу четыре

Разработана базы данных для работы с программой. Описаны структуры таблиц базы, её триггеров и их представлений. Рассмотрены хранимые процедуры, используемые в базе. Разработан способ защиты паролей пользователей программы в таблице Responsible и описан алгоритм работы защиты.

5 РАЗРАБОТКА ПРОГРАММЫ

Программа разработана с помощью среды разработки Microsoft Visual Studio 2017 Community. Язык программирования – C#.

Разработка программы разделена на несколько этапов.

1. Подключение базы данных к программе;
2. Создание главной формы, написание кода для событий формы;
3. Создание формы для авторизации пользователей, написание кода для событий формы, обработка авторизации пользователей;
4. Создание форм для работы с программой;
5. Разработка форм – заполнение элементами управления, обработка событий форм и элементов, написание кода;
6. Отладка программы.

5.1 Подключение базы данных к программе

В обозревателе серверов подключена база данных из SQL Server (рисунок 11).

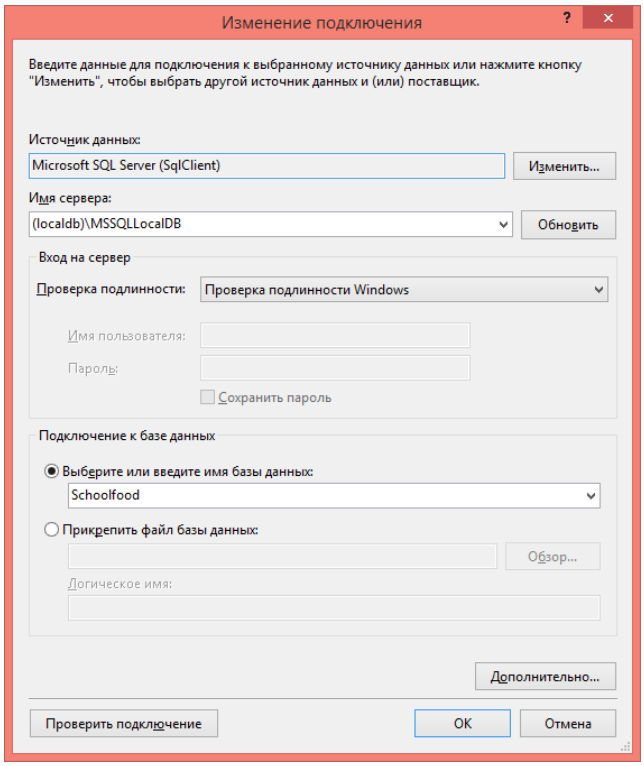


Рисунок 11 – Подключение базы данных в VisulStudio

После подключения базы данных, необходимо настроить источник данных (рисунок 12). В источник данных нельзя добавлять таблицы, так как для работы с ними созданы представления, защищенные от прямого редактирования.

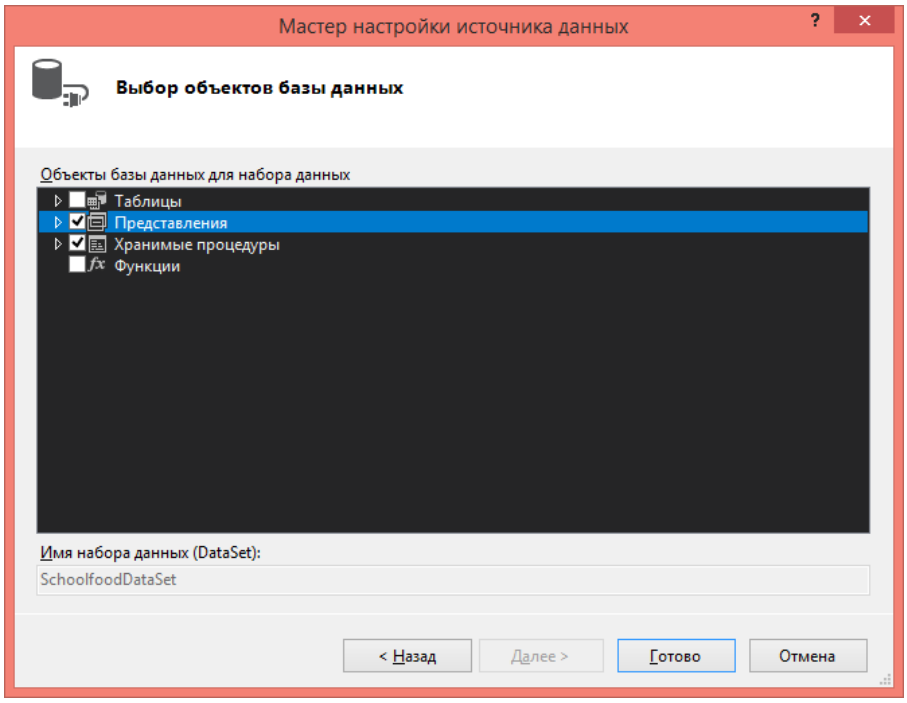


Рисунок 12 – Настройка источника данных приложения

5.2 Главная форма приложения

Главная форма является MDI контейнером. MDI или многодокументный интерфейс — способ организации графического интерфейса пользователя, предполагающий использование оконного интерфейса, в котором большинство окон расположены внутри одного общего окна. Таким образом, для форм, с которыми будут работать пользователи, в свойстве MdiParent (MDI родитель) необходимо указывать ссылку на главную форму.

Для открытия дочерних форм в главной форме добавлен элемент menuStrip (строка меню).

Для удобства пользователя в форму добавлен statusStrip (статусная строка). Она показывает текущего пользователя, текущую дату и время, показывает статус дополнительной цифровой клавиатуры NumLock и режима смены регистра CapsLock.

На рисунке 13 показан интерфейс главной формы.

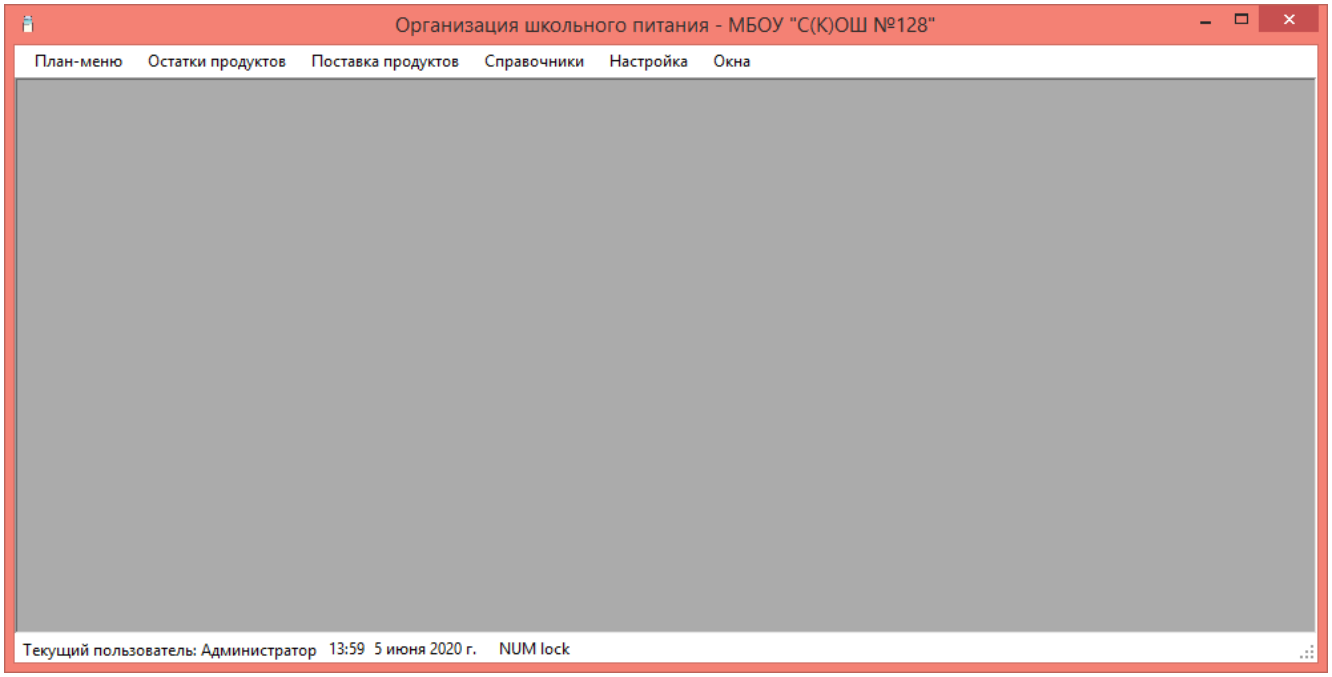


Рисунок 13 – Интерфейс главной формы приложения

Для показа даты используется свойство Today (сегодняшний день) структуры DateTime.Для показа текущего времени используются элемент управления timer (таймер). Оба действия запускаются в обработке открытия формы.

Показ времени осуществлен с помощью обработчика события Tick для таймера. В обработчике label'у (надписи) присваивается свойство Now (текущее время и дата) структуры DateTime. Таймер настроен на секундный интервал, то есть время будет идти точно также, как на компьютере.

Для показа статуса клавиш NumLock и CapsLock используется класс Console и свойства NumLock и CapsLock, возвращающие значение типа Boolean. Если клавиши включены, свойство вернет True, иначе False. Такая проверка используется в обработке открытия формы и события KeyDown (срабатывает если клавиша нажата один раз).

Показ статуса пользователя осуществляется с помощью формы авторизации, описанной в п.п. 5.3. Из формы авторизации передается имя пользователя, и оно записывается в статусную строку в dropDownButton (кнопка с

выпадающим списком). При нажатии на эту кнопку можно сменить пользователя (при условии, что он завершил необходимую работу с открытыми формами) или выйти из программы.

Меню главной формы содержит следующие элементы:

1. План-меню – через него можно открыть школьное меню и меню для сотрудников школы.
2. Остатки продуктов – открывает форму, показывающую состояние склада продуктов питания на текущую дату.
3. Поставка продуктов – открывает список накладных.
4. Справочники – через него можно открыть список продуктов питания, технологических карт и поставщиков.
5. Настройка – содержит основные настройки программы, настройки пользователей и переход в режим администратора. Также есть раздел «О программе».
6. Окна – показывает активные формы приложения.

5.3 Форма авторизации пользователей

Форма авторизации пользователей (рисунок 14) запускается при открытии главной формы. Данная форма является диалоговым окном, по завершению диалога возвращается значение типа Boolean, характеризующее результат работы формы.

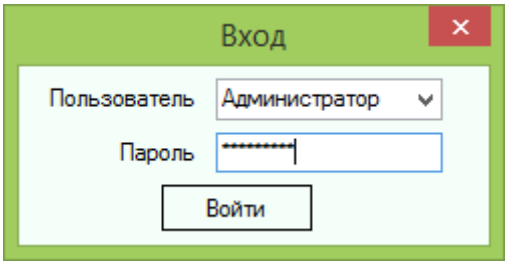


Рисунок 14– Интерфейс формы авторизации пользователей

Текстовое поле «Пароль» использует свойство PasswordChar, которые скрывает содержимое поля символом «*».

Пользователи выбираются с помощью comboBox (поле со списком). Источником данных для поля со списком является представление View_Responsible, элемент отображения – столбец Position, значение элемента – столбец ID.

Чтобы идентифицировать пользователя и подстроить функционал программы под его права, создан класс User, сохраняющий имя пользователя и его ID из формы авторизации. Кодакласса User приседен в листинге 1.

Листинг 1 − Код класса User:

```
Public class User//Пользователь программы
{
PublicstringUserName { get; set; } //имяпользователя
Public string UserData { get; set; } //ФИО пользователя - для документов
Publiccint UserID { get; set; } //Id взятыйизтаблицы Responsible
Public bool EnableInsert { get; set; } //разрешение добавления для пользователя
Public bool EnableUpdate { get; set; } //разрешение редактирования для пользователя
Public bool EnableDelete { get; set; } //разрешение удаления для пользователя
Public User (string un, string ud, int id) //конструктор класса с заданием параметров для всех
свойств класса
{
            UserName = un;
            UserData = ud;
            UserID = id;
}
Public User (int id) //конструктор класса с заданием ID пользователя (необходим для настроек
программы)
        {
            UserID = id;
        }
Public User () //конструктор по умолчанию
        {
            UserName = UserData = "";
            UserID = 0;
        }
Public void OpenConnection ()// Открытие подключения к базе данных с обработкой возможных
ошибок соединения
{
try
            {
                UserConnection().Open();
            }
catch (System.ComponentModel.Win32Exception)
            {
                UserConnection().Open();
            }
catch(System.Data.SqlClient.SqlException)
            {
                UserConnection().Open();
            }
        }
private Sql.SqlConnection UserConnection() //строкаподключениякбазеданных
        {
            Sql.SqlConnection conn = new
Sql.SqlConnection(Properties.Settings.Default.SchoolfoodConnectionString);
return conn;}
```

```
publicvoidAllowsForMenuStud() //методобработкиправпользователядляформыШкольногоменю
publicvoidAllowsForMenuSotr()//методобработкиправпользователядляформыМенюдлясотрудников
public void AllowsForNotes()//метод обработки прав пользователя для формы Поставка продуктов
public void AllowsForSuppliers()//метод обработки прав пользователя для формы Поставщики
public void AllowsForUserSettings()//метод обработки прав пользователя для настройки данных
пользователя
public void AllowsForStoreSettings()//метод обработки прав пользователя для настройки склада
public void AllowsForMenuSettings()//метод обработки прав пользователя для настройки меню
public void AllowsForUserList()//метод обработки прав пользователя для настройки прав
пользователей
        public void AllowsForNotesSettings()//метод обработки прав пользователя для настройки
поставок
```

Методы AllowsFor()содержат код, представленный в листинге 2, который дублируется в каждом методе.

Листинг 2 – Код методов AllowsFor()

```
Form1 f = new Form1(); //переменная класса Главной формы
f.view_UserAllowsTableAdapter.Fill(f.schoolfoodDataSet.View_UserAllows); //заполнение данными
представления User_Allows
//фильтр таблицы UserAllows по ID пользователя и блоку редактирования
f.view_UserAllowsBindingSource.Filter = "User = " + UserID + " AND Table-
Name=[Названиетаблицы]";
foreach(DataRowView drv in f.view_UserAllowsBindingSource)
{//цикл foreach проходит по отфильтрованной таблице
//и присваивает права на добавление, удаление и изменение для пользователя
EnableInsert = Convert.ToBoolean(drv.Row["AllowInsert"]);
                EnableUpdate = Convert.ToBoolean(drv.Row["AllowUpdate"]);
                        EnableDelete = Convert.ToBoolean(drv.Row["AllowDelete"]);
```

С помощью переменной класса Form1 функция обращается кисточнику данных представления View_UserAllows и фильтрует этот источник по IDпользователя и таблице (форме), с которой работает пользователь. После фильтрации свойствам разрешения добавления, изменения и удаления присваиваются значения типа Booleanиз отфильтрованного источника данных.

Если вход прошел успешно, т.е. пользователь ввел верный пароль, диалоговое окно авторизации возвращает значение True для главной формы, чтобы она обработала процедуру открытия до конца. Если пользователь закроет диалоговое окно, программа закроется полностью.

При успешном входе в главной форме создается переменная класса User, в которую сохраняются имя текущего пользователя и его ID, полученные при авторизации.

Если пользователь не имеет пароля, он всё равно может зайти в программу, так как права идентифицируются по ID пользователя. Если же пароль задан, то он уже не сможет просто так зайти в программу.

Для администратора обработка входа состоит из двух этапов – ввод пароля администратора и ввода пароля от закрытого ключа ассиметричного шифрования.

Обработка всех паролей происходит с помощью защищенной процедуры Admin_SelectPwdDecrypt, параметром которой является пароль от закрытого ключа. Если пользователь не является администратором, обработка происходит без ввода ключевого пароля. Он хранится в настройках программы и защищен от модификаций со стороны пользователей.

Если пользователь является администратором, запускается окно для ввода ключевого пароля.

На рисунке 15 показан интерфейс формы входа в режим администратора.

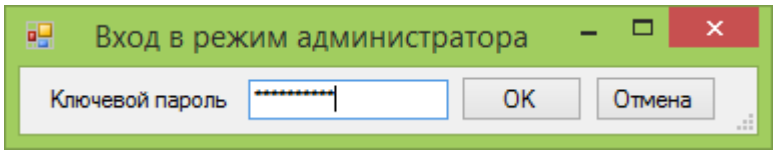


Рисунок 15 – Вход в режим администратора

Если введенный ключевой пароль верен, то произойдет обработка пароля, иначе вход под администратором будет отменен.

5.4 Разработка форм для работы с данными

5.4.1 Разработка формы «Продукты питания»

Форма «Продукты питания»представляет собой список всех продуктов питания.

Источник данных для dataGridView (таблицы) – представление View_Product. Столбец ID скрыт от пользователей.

В командной панели расположены поля для навигации по таблице (переходам по записям), кнопки добавления, редактирования и удаления продукта, поиск по названию продукта.

На рисунке 16 изображен интерфейс формы списка продуктов.

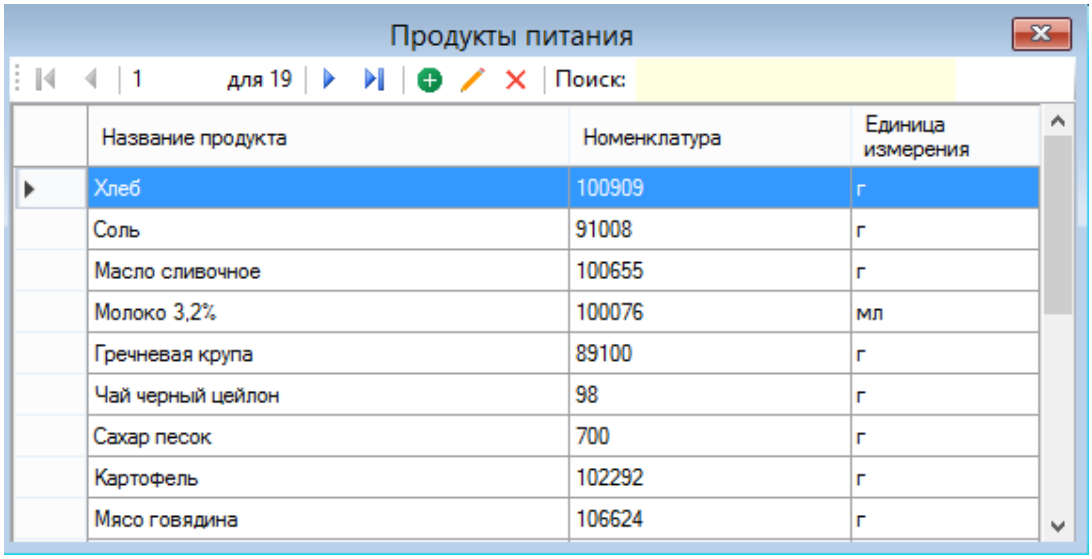


Рисунок 16 – Интерфейс формы «Продукты питания»

В командной панели расположены поля для навигации по таблице (переходам по записям), кнопки добавления, редактирования и удаления продукта, поиск по названию продукта.

Добавление и редактирование продукта происходит в отдельной форме.

Из формы со списком продуктов передаются ID продукта, его название, номенклатурный номер, единица измерения и минимально возможный остаток.

Форма (рисунок 17) открывается как диалоговое окно. Если пользователь нажмет кнопку «ОК», то форма закроется и передаст форме со списком продуктов значение True. При нажатии на кнопку «Отмена» передастся значение False.

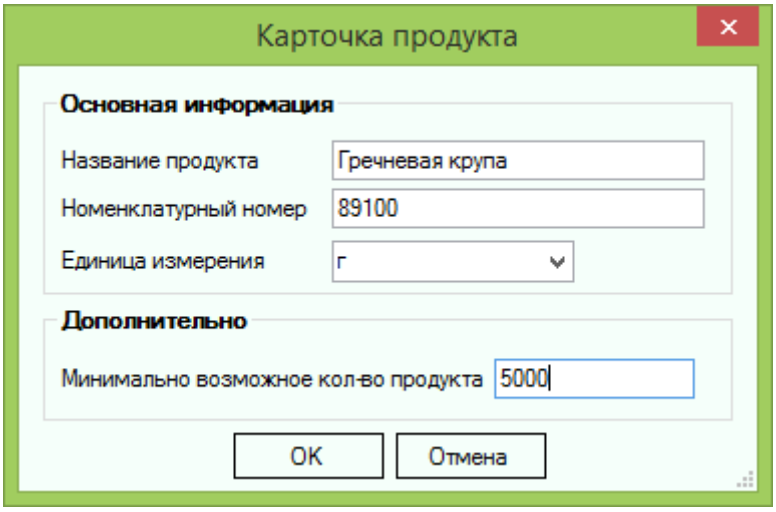


Рисунок 17 – Интерфейс формы добавления и редактирования продуктов

В форме прописаны свойства для хранения ID продукта, названия продукта, номенклатурного номера, единицы измерения, минимально возможного остатка. Когда обрабатывается одно из событий, «ОК» или «Отмена», все свойства сохраняют свои значения и могут передаваться в форму со списком продуктов для выполнения процедур добавления или изменения.

Процедуры для работы с данными хранятся в quiresTableAdapter – это набор хранимых процедур из базы данных, представленных в виде запросов с параметрами в источнике данных программы. Этот элемент расположен в главной форме и имеет уровень доступа Public, чтобы пользователь мог работать с ним из любой формы. Он вызывается с помощью переменной класса Form1 (класс главной формы). Для этого необходимо в конструкторе дочерней формы указать параметр с типом данных Form1. Пример указания переменной в форме «Продукты питания» представлен в листинге 3.

Листинг 3 – Пример указания переменной типа Form1

```
public readonly Form1 f1;
public Form_Products(Form1 f){ f1 = f;}
```

Ключевое слово readonly нужно для того, чтобы переменная была доступна только для чтения. Переменной f1 присваивается значение параметра конструктора f.Пример вызова quiriesTableAdapter представлен в листинге 4.

Листинг 4 – ПримервызоваquiriesTableAdapter

```
f1.queriesTableAdapter1.UPD_Products(id, nazv, nomen, unit, limit)
```

Подобным образом работают все формы, где имеет смысл использовать процедуры INSERT, UPDATE и DELETE.

После того, как процедура выполнена, необходимо заново заполнить таблицу данными. Пример заполнения данных указан в листинге 5.

Литсинг 5 – Пример заполнения таблицы данными после процедур INSERT, UPDATEи DELETE

```
view_ProductsTableAdapter.Fill(this.schoolfoodDataSet.View_Products)
```

После заполнения данных, таблица со списком продуктов обновится и покажет обновленные данные.

5.4.2 Разработка формы «Технологические карты»

Форма представляет собой просмотр информации о карте, выбранной из поля со списком. Источник данных для поля со списком – View_Dishes, элементы списка отображаются по столбцу Name, значение элемента – столбец ID.

По выбранной карте фильтруются сразу два источника данных формы: View_Dishes и View_Recepie.

Форма разделена на две части – основная информация по карте и состав блюда. Интерфейс формы показан на рисунке 18.

Основная информация представлена в четырех textBox'ах (текстовых полях) с привязанными к ним с помощью свойства DataBindingстолбцами из View_Dishes. К первому текстовому полю привязан столбец Formulation, ко второму – Cookbook, к третьему – Temperature, к четвертому – Shelf.

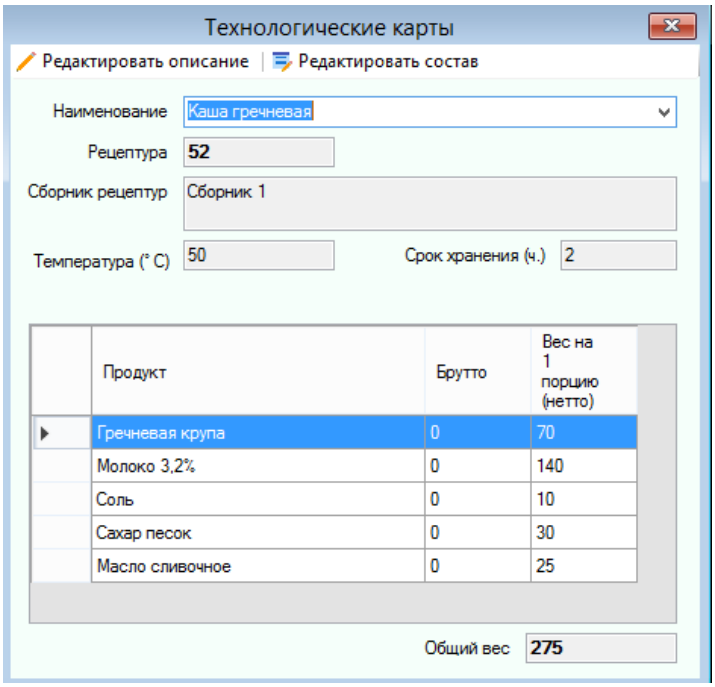


Рисунок 18 – Интерфейс формы «Технологические карты»

Для редактирования основной информации о карте в командной панели (toolStrip) создана кнопка «Редактировать описание» (рисунок 19).

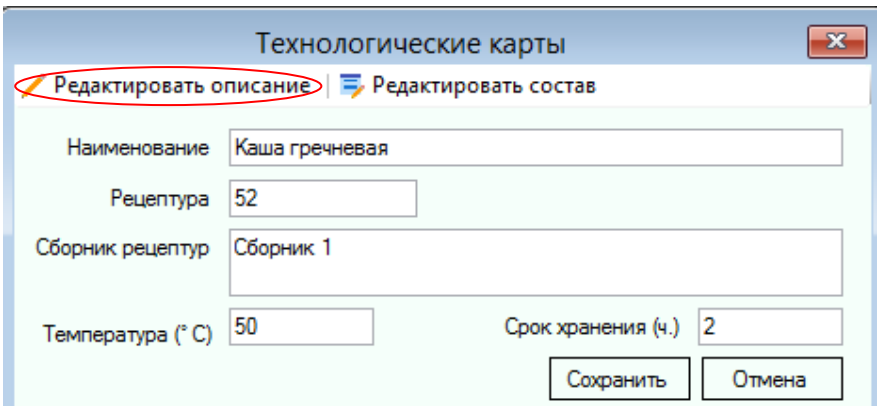


Рисунок 19 – Редактирование описания технологической карты

Чтобы редактированиеработало корректно, в форму добавлены текстовые поля, которые скрыты от пользователя до тех пор, пока он не решит отредактировать описание. Скрытые поля именуются идентично столбцам в представлении. Данные вносятся в эти поля с помощью чтения текущей записи из view_DishesBindingSource (источник данных представления View_Dishes). Для чтения записи используется переменная класса DataRowView. Кнопки сохранения и отмены редактирования также скрыты от пользователей до нажатия кнопки «Редактировать описание».

Для упрощения отображения скрытых полей и скрытия открытых данных, в коде формы прописана функция DefenitionChange с двумя параметрами типа Boolean. Код функции представлен в листинге 6.

Листинг 6 – КодфункцииDefenitionChange

```
PrivatevoidDefenitionChange(boolv1, boolv2)
{//метод изменения видимости текстовых полей для редактирования карты
comboBox1.Visible = v1;
            textBox1.Visible = v1;
            textBox2.Visible = v1;
            textBox3.Visible = v1;
            textBox4.Visible = v1;
            view_RecepiesDataGridView.Enabled = v1;
//открываются кнопки Сохранить и Отмена, а также поля для редактирования
button1.Visible = v2;
            button2.Visible = v2;
            NameTb.Visible = v2;
            FormulationTb.Visible = v2;
            CookbookTb.Visible = v2;
            TemperatureTb.Visible = v2;
ShelfTb.Visible = v2;
if (v1 == false) //если основные поля (недоступные для редактирования) скрыты
//то присваиваем полям для редактирования основные данные
{
                position = view_DishesBindingSource.Position;
NameTb.Text = drv.Row["Name"].ToString();
```

```
FormulationTb.Text = drv.Row["Formulation"].ToString();
CookbookTb.Text = drv.Row["Cookbook"].ToString(); ;
TemperatureTb.Text = drv.Row["Temperature"].ToString(); ;
ShelfTb.Text = drv.Row["Shelf"].ToString();}}
```

Первый параметр отвечает за открытые поля, второй параметр – за скрытые. Если первый параметр имеет значение False, то в скрытые поля переносятся данные из открытых полей и кнопки «Сохранить» и «Отмена» становятся доступными.Переменная position помогает перемещаться по источнику данных View_Dishes, не сбрасывая фильтрацию.

Вторая часть формы – состав блюда – это таблица dataGridView, источником данных которой является представление View_Recepies. Чтобы рецепт блюда отображался правильно, при изменении выбора блюда в поле со списком, источник данных таблицы фильтруется по ID блюда.

На рисунке 20 показан интерфейс редактирования состава рецепта.

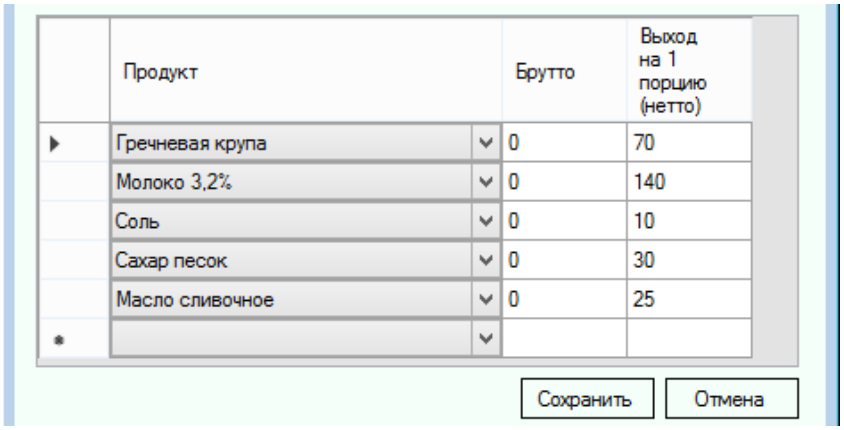


Рисунок 20 – Интерфейс редактирования состава рецепта

Аналогично редактированию основной информации, для изменения рецепта создана отдельная кнопка «Редактировать состав». В форму добавлена скрытая пустая таблица, которая будет заполняться данными из таблицы с рецептом, перенося при этом в скрытый столбец ID записи. Такое решение необходимо для того, чтобы программно распознать новую запись и старую запись и в зависимости от этого выполнять подходящую процедуру. Кнопки «Сохранить» и «Отмена» для редактирования рецепта также скрыты от пользователя.

Создана аналогичная DefenitionChange функция RecepieChange с двумя параметрами типа Boolean, где первый параметр отвечает за таблицу рецептов, а второй – за скрытую таблицу. Код функции представлен в листинге 7.

Листинг 7– Код функции RecepieChange

```
PrivatevoidRecepieChange(boolv1, boolv2)
{//метод открытия грида для редактирования рецепта блюда
view_RecepiesDataGridView.Visible = v1;
            textBox5.Visible = v1;
            label6.Visible = v1;
            comboBox1.Enabled = v1;
            RecepieGridView.Visible = v2;
            button3.Visible = v2;
button4.Visible = v2;
if (v1 == false) //если таблица для отображения рецепта скрыта
            {//то заполняем таблицу для редактирования, значения берутся из скрытой таблицы
for (int i = 0; i < viewRecepiesBindingSource.Count; i++)
                {
                    RecepieGridView.Rows.Add(view_RecepiesDataGridView.Rows[i].Cells[0].Value,
                        view_RecepiesDataGridView.Rows[i].Cells[3].Value,
                        view_RecepiesDataGridView.Rows[i].Cells[4].Value,
                        view_RecepiesDataGridView.Rows[i].Cells[5].Value);
                }
                position = view_DishesBindingSource.Position;
            }
else//иначеубираемфильтр
            {
                RecepieGridView.Rows.Clear();
                viewRecepiesBindingSource.Filter = "[Dish] = '" + ID + "'";
position = 0;}}
```

При первом параметре равном False происходит заполнение скрытой таблицы данными из таблицы рецептов. Если первый параметр равен True, то скрытая таблица очищается.

Процедуры добавления, изменения и удаления записей из таблиц Dishes и Recepies берутся из адаптера хранимых процедур quiresTableAdapter.

5.4.3 Разработка формы «Поставщики»

Форма представляет собой список поставщиков с возможностью просмотра подробной информации о организации и накладных за текущий месяц по выбранному поставщику.

На рисунке 21 изображен интерфейс формы.

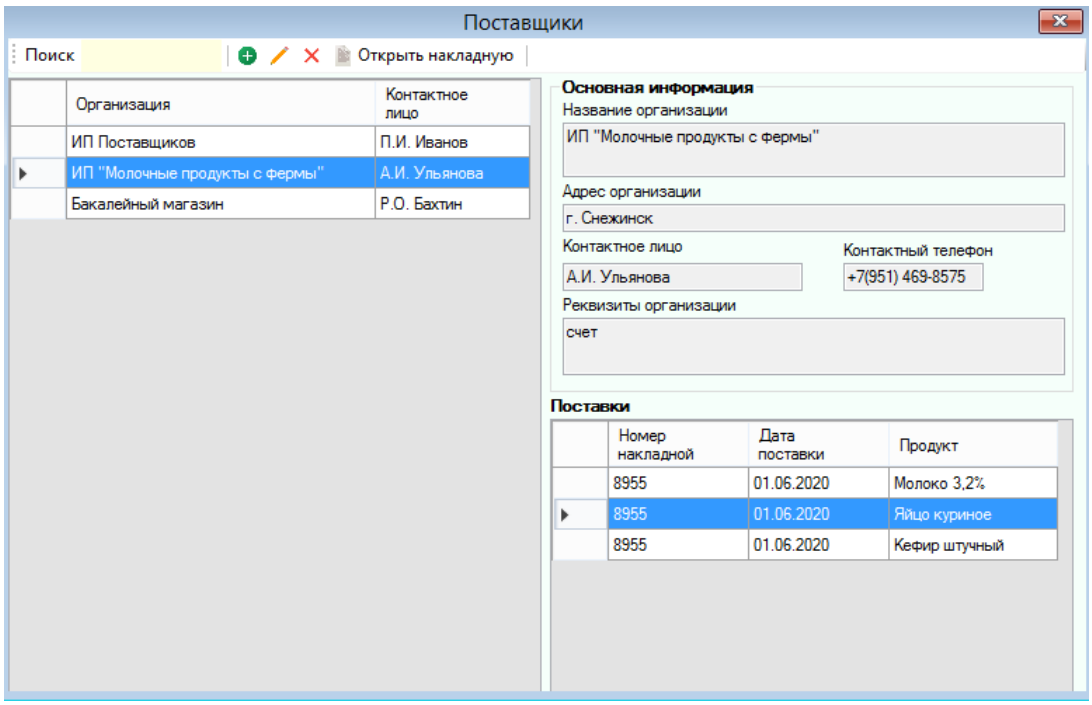


Рисунок 21 – Интерфейс формы «Поставщики»

Левая часть формы – таблица dataGridView со списком действующих поставщиков. Источник данных таблицы – представление View_Suppliers. Для работы с таблицей в командную панель toolStrip добавлен поиск по названию организации. Рядом с поиском расположены кнопки добавления, редактирования и удаления организации.

Права часть формы состоит из двух блоков – основная информация о поставщике и список накладных по выбранному поставщику.

Блок с основной информацией содержит текстовые поля соответственно полям таблицы – название организации, адрес, телефон, контактное лицо и реквизиты. При выборе поставщика в таблице, вся информация отображается в текстовых полях. Также блок имеет скрытый от обычных пользователей checkbox с статусом поставщика Active. Этот элемент может видеть только администратор.

Добавление и редактирование организации реализовано в одной форме. При добавлении все текстовые поля очищаются, таблица со списком поставщиков и накладными делается недоступными. В командной строке появляются кнопки «Сохранить» и «Отменить». Сохранение срабатывает при помощи флага – если

флаг равен 1, то идет процесс добавления организации, если флаг равен 2 – идет редактирование описания.

Таблица накладных по поставщику фильтруется по коду выбранного в поставщика и показывает, какие продукты поставлял поставщик в текущем месяце.

С помощью контекстного меню таблицы можно просмотреть выбранную накладную подробнее, либо снять фильтр по дате и просмотреть все накладные по поставщику.

На рисунке 22 изображен интерфейс формы при добавлении новой организации.

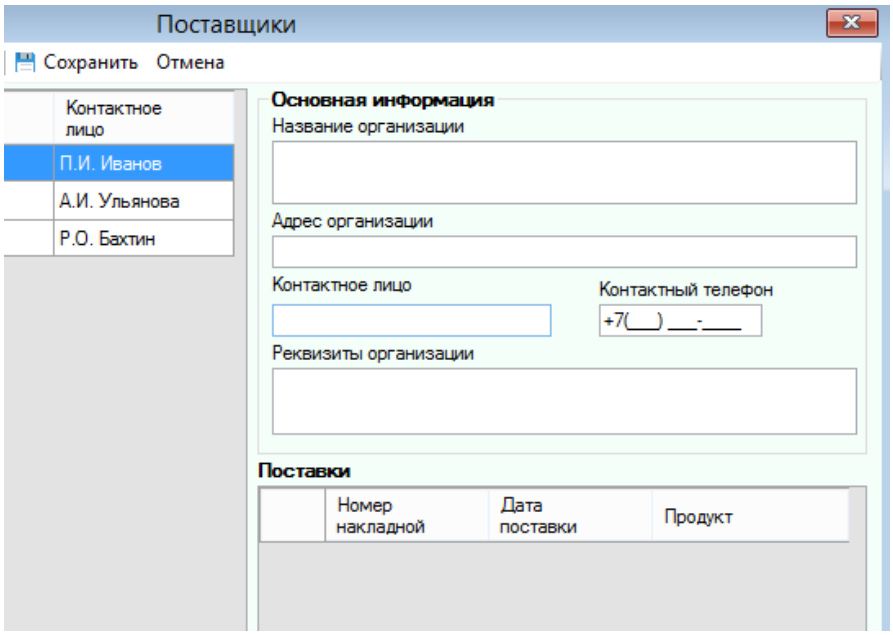


Рисунок 22 – Интерфейс добавления нового поставщика

Открыть накладную можно как из командной панели формы, так и из таблицы накладных по поставщику (рисунок 23).

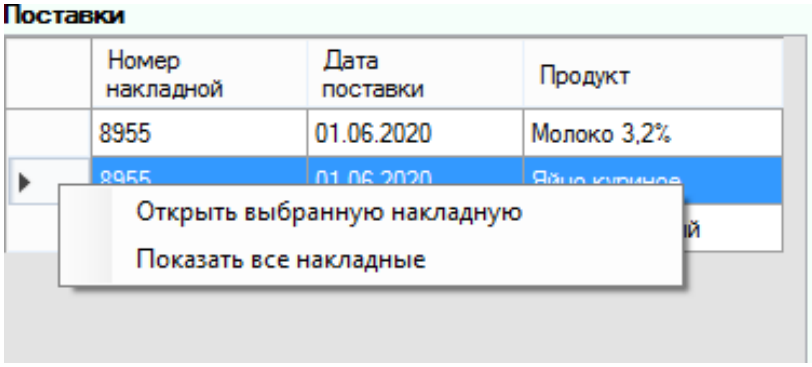


Рисунок 23 – Переход от поставщика к накладной

5.4.4 Разработка формы «Поставка продуктов»

Формапредставляет собой список накладных в разрезе продуктов, источник данных таблицы – представление View_Notes.

На рисунке 24 изображен интерфейс формы.

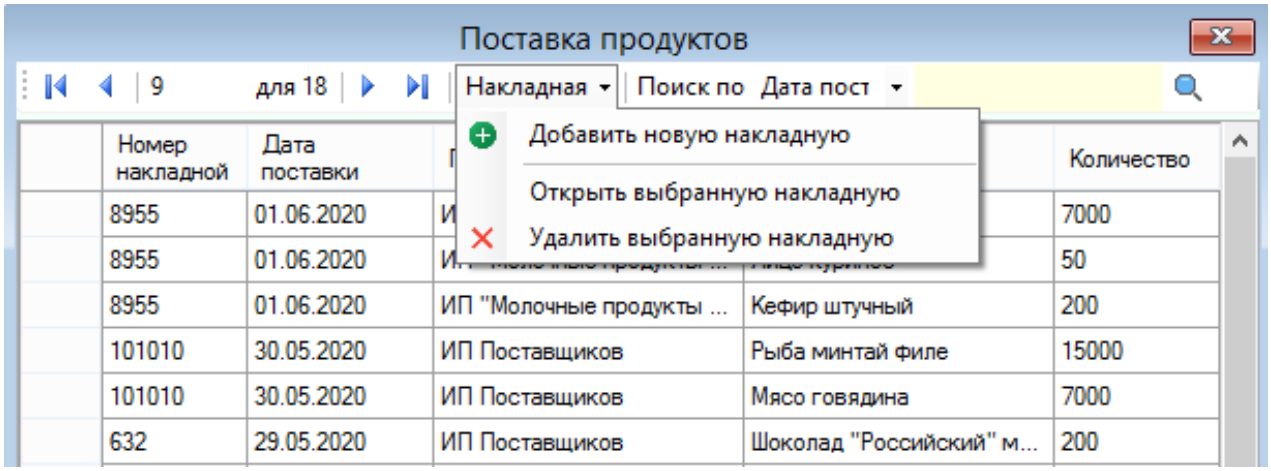


Рисунок 24 – Интерфейс формы «Поставка продуктов»

Список можно отфильтровать по поставщику, по продукту или дате поставки (раздел в командной панели «Поиск по»). В командной панели можно открыть выбранную накладную, удалить её или добавить новую.

При добавлении новой накладной или открытии выбранной накладной откроется форма, представленная как диалоговое окно, которое возвращает значение флага Flag. Если Flagравен нулю, то список накладных обновится и покажет обновленные данные. Если Flagне равен нулю, то обновления списка не произойдет.

В форме отдельной накладнойпрописаны свойства, принимающие и передающие значения из и в форму со списком соответственно. Свойства идентичны типам и названиям столбцов таблицы накладных, т.е. это номер накладной, дата поставки, поставщик, ответственный сотрудник (кто принял накладную) и состав накладной – название продукта и его количество.

Аналогично технологическим картам (п.п. 5.4.2) отдельно можно редактировать основную информацию накладной и её состав.Код функций редактирования прописан в листинге 8.

Листинг 8 – Код функций DefinitionChange (изменение описания накладной) и NoteChange (изменение состава накладной)

```
Publicvoid DefenitionChange(bool v1, bool v2)//изменениеописаниянакладной
{//скрываются поля для отображения информации
            textBox1.Visible = v1;
textBox2.Visible = v1;
            textBox3.Visible = v1;
            textBox4.Visible = v1;
            view_NotesDataGridView.Enabled = v1;
//открываются поля для редактирования информации
NumberTb.Visible = v2;
DateTbMask.Visible = v2;
            SupplierCombo.Visible = v2;
RespCombo.Visible = v2;
//открываютсякнопкиСохранитьиОтмена
toolStripButton3.Visible = toolStripButton4.Visible = v2;
if (v1 == false&& ID_note != 0) //если поля для отображения информации скрыты
            {//то заполняем значениями поля для редактирования
NumberTb.Text = Number_note.ToString();
                DateTbMask.Text = Date_note.ToShortDateString();
                SupplierCombo.SelectedValue = Supplier_note;
                RespCombo.SelectedValue = Resp_note;
}
        }
PublicvoidNoteChange(boolv1, boolv2)//измененисоставанакладной
{//скрывается таблица с отображением информации
            view_NotesDataGridView.Visible = v1;
//и открывается таблица для редактирования информации
            dataGridView1.Visible = v2;
//открываются кнопки Сохранить и Отмена
toolStripButton3.Visible = toolStripButton4.Visible = v2;
if (v1 == false&& ID_note != 0) //если таблица для отображения информации скрыта
{
foreach (DataRowView drv in viewNotesBindingSource1)
{//заполняем значениями из таблицы для отображения информации
dataGridView1.Rows.Add(drv.Row["ID"], drv.Row["Product"], drv.Row["Count"]);
}
            }
if (v1 == true) //иначе чистим таблицу для редактирования
            {dataGridView1.Rows.Clear();}}
```

Если добавляется новая накладная, то флаг равен нулю, если редактируется описание – единице, а если редактируется состав – двойке. В зависимости от значения флага выполняются необходимые процедуры INSERTили UPDATE.

На рисунке 25 показан интерфейс формы редактирования накладной.

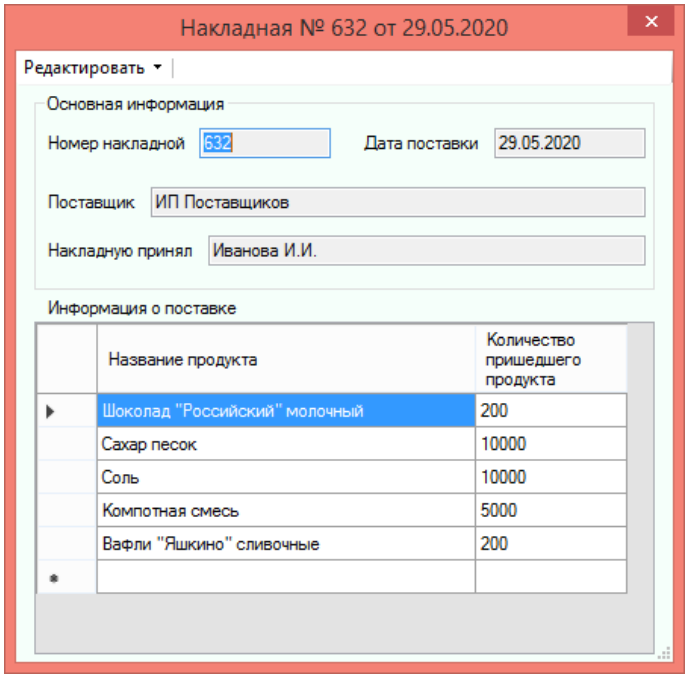


Рисунок 25 – Интерфейс формы редактирования накладной

На рисунке 26 показан пример редактирования основной информации по выбранной накладной.

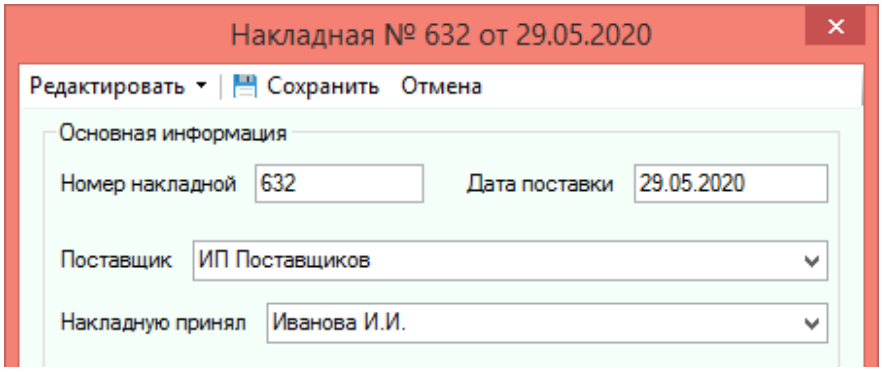


Рисунок 26 – Интерфейс редактирования основной информации о накладной

5.4.5 Разработка формы «Остатки продуктов»

Данная форма показывается остатки продуктов на складе на сегодняшний день.

Форма поделена на две части – список продуктов и приходно-расходный разрез. Интерфейс формы показан на рисунке 27.

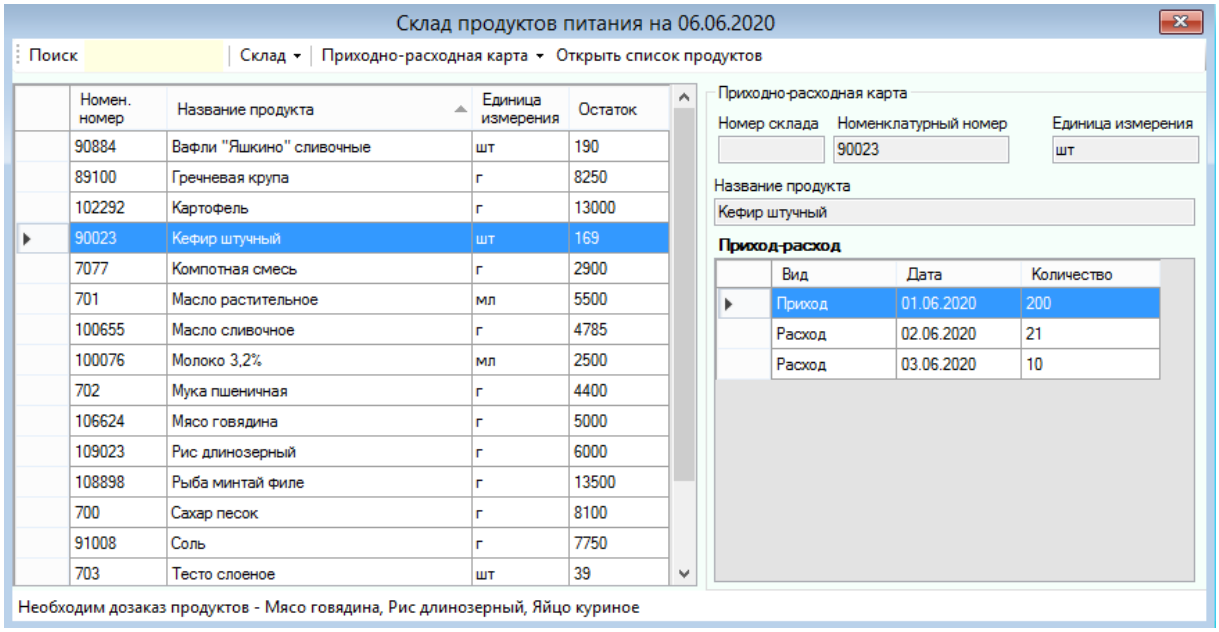


Рисунок 27 – Интерфейс формы «Отстаки продуктов»

Источник формы для списка продуктов – представление склада View_Store.

По выбранному из списка продукту в правой части формы показывается его приход-расход. Информация в текстовых полях показывается с помощью свойства текстовых полей DataBindings, которые связанны с столбцами представления View_Store.

В статусной строке формы выводится статус склада, а именно какие продукты необходимо заказать у поставщиков. Для наглядности в команде «Склад» эти позициив таблице можно выделить цветом. Рекомендация на заказ продуктов рассчитывается из разницы остатков продукта на складе и минимально возможного количества этого продукта. Если разница отрицательная или равна нулю, то в статусную строку выводится название продукта, необходимого для дозаказа.

Приход-расход – это представление View_Sort_Card, отфильтрованное по дате и показывающее приход-расход за текущий месяц.

В команде «Склад», помимо выделения цветом продуктов для дозаказа, можно провести корректировку списания, вывести отчет по состоянию склада и расходу продуктов по периоду, а также сформировать ведомость на заказ продуктов.

Корректировка списания – это процедура, в основном реализуемая в меню, которая корректирует остатки продуктов (рисунок 28). Например, в остатках имеются 15 яблок. По текущему меню списалось всего 12 яблок, а 3 штуки осталось на складе. Чтобы списать «под ноль» все яблоки для текущего меню, необходимо провести корректировку списания.

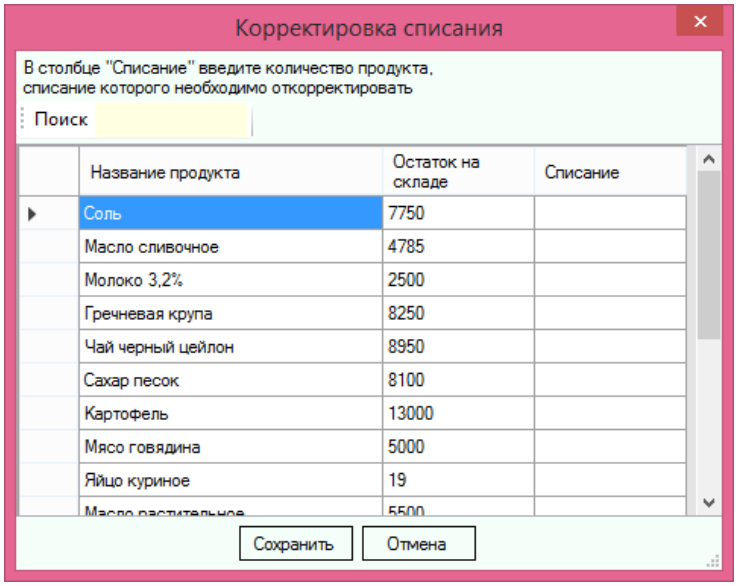


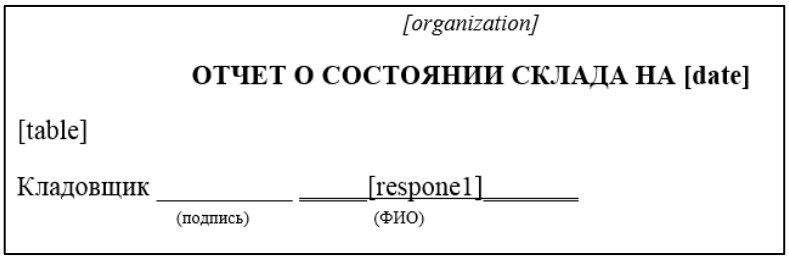
Рисунок 28 – Интерфейс формы «Корректировка списания»

В столбец «Списание» вводится количество продукта для корректировки списания. При сохранении корректировки, в таблицу Out_kitchenдобавится запись с пометкой «Correct» и кодом продукта. После сохранения форма остатков продуктов обновится и покажет обновленные записи в таблице остатков.

Отчет о состоянии склада, расходе продуктов и ведомость на заказ продуктов выводятся в документы Wordc помощью библиотеки Microsoft.Office.Interop. Для каждого документа создан шаблон Word (формат .dot)с пометками (Bookmarks) для добавления туда элемента из программы. Так, например, для отчета о состоянии склада в пометку «organization» добавляется

название организации, а в пометку «table» добавляется таблица, созданная и заполненная программно в самом приложении.

На рисунке 29 изображен шаблон Wordдля вывода отчета о состоянии склада.


[organization]

ОТЧЕТ О СОСТОЯНИИ СКЛАДА НА [date]

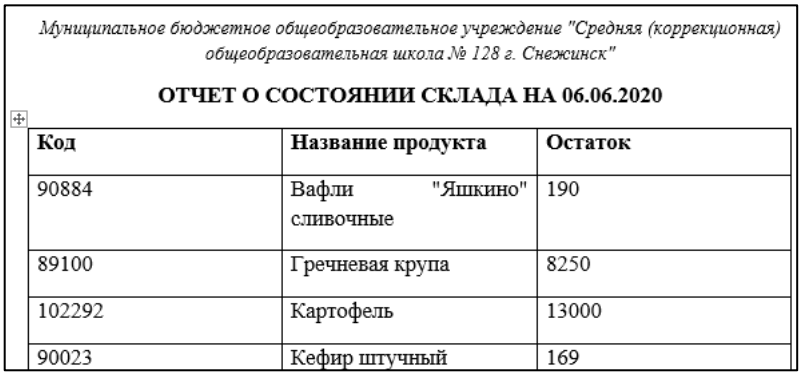
[table]

Кладовщик _____ _____[respone1]_____

(подпись) (ФИО)

Рисунок 29 – Шаблон Wordдля документа «Отчет о состоянии склада»

Пример отчета о состоянии склада изображен на рисунке 30.


Муниципальное бюджетное общеобразовательное учреждение "Средняя (коррекционная) общеобразовательная школа № 128 г. Снежинск"

ОТЧЕТ О СОСТОЯНИИ СКЛАДА НА 06.06.2020

Код	Название продукта	Остаток
90884	Вафли "Яшкино" сливочные	190
89100	Гречневая крупа	8250
102292	Картофель	13000
90023	Кефир штучный	169

Рисунок 30 – Документ «Отчет о состоянии склада»

Подобным образом работает вывод всех документов Wordв программе.

Заказ продуктов рассчитывается программой по следующему алгоритму:

1. Рассчитывается разница между остатком продукта на складе и минимально возможным количеством продукта (лимита).

2. Если разница больше нуля, то к рекомендации на заказ идет сумма разницы и половины лимита.

3. Если разница меньше или равна нулю, то к рекомендации на заказ идет сумма разницы, лимита и половины лимита, чтобы количество продукта превышало лимит в остатках.

Код алгоритма представлен в листинге 9.

Листинг 9 – Код алгоритма расчета заказа продукта

```
int order = rest - limit; //рассчитывается разница между остатком на складе и лимитом
```

```
if (order > 0) //если заказ больше нуля
{
double d = 0.5 * limit;
order = order + (int)d; //к заказу добавляется половина лимита продукта
}
if (order == 0) //если заказ равен нулю
{
double d = 0.5 * limit;
order = order + limit + (int)d; //к заказу добавляется лимит продукта и половина этого же
лимита
}
if (order < 0) //если заказ отрицательный
{
double d = 0.5 * limit;
order = (-1) * order + limit + (int)d;//делаем заказ положительным числом и добавляем половину
лимита продукта}
```

Пользователь сможет отредактировать количество продуктов на заказ в сохраненном документе.

В команде «Приходно-расходная» карта можно сбросить фильтр таблицы по текущему месяцу и просмотреть все изменения, происходящие с продуктом, вернуть фильтр по месяцу, экспортировать в Excelсортовую карту по выбранному продукту, либо сформировать карты по всем продуктам.

Шаблон сортовой приходно-расходной карты создан в книге Excel (формат .xltx) по утвержденному стандарту, данные в него добавляются с помощью библиотеки Microsoft.Office.Interop. Программа находит нужные ячейки и вставляет в них необходимые значения. Остаток на начало месяца рассчитывается в прописанной в коде формы функции RestToMonth, которая считает весь приход и весь расход до начала текущего месяца и возвращает разность этих сумм. Код функции представлен в листинге 10.

Листинг 10 – Код функции RestToMonth

```
PrivateintRestToMonth()
{
int count_in = 0, count_out = 0; //суммаприхода, суммарасхода
//источник данных для вывода в Excel фильтруется по ID выбранного продукта
//и по условию, что дата меньше даты начала текущего месяца
forExcelBindingSource.Filter = "[ID] = " + ID + " AND [Date] < '" + first + "'";
foreach (DataRowView drv in forExcelBindingSource)
{//в цикле считаем суммы прихода и расхода
if (drv.Row["Вид"].ToString() == "Приход")
{
  count_in = count_in + Convert.ToInt32(drv.Row["Count"]);
}
if (drv.Row["Вид"].ToString() == "Расход")
{
  count_out = count_out + Convert.ToInt32(drv.Row["Count"]);}
}
```

```
returncount_in - count_out; //навыводидетразностьсуммприходаирасхода}
PrivateintRestToMonth(intid_p) //перегрузка функции - параметр id_p - ID продукта
```

По итогу месяца в карте считаются сумма прихода, сумма расхода на школьное меню и на меню сотрудников. На рисунке 31 изображен пример сортовой приходно-расходной карты на текущий месяц (июнь).

СОРТОВАЯ ПРИХОДНО-РАСХОДНАЯ КАРТА

N F 7077 (склад (5-7) системный номер (8-14)) — г (ед.изм.) — (цена) — (норма запаса)

Компотная смесь (наименование материала,) (марка,) (тип,) (сорт,) (ГОСТ на марку)

(размер (сортамент)) (ГОСТ на сортамент)

Шифр	№ записи	Документ: дата	Документ: номер	От кого поступило или кому отпущено	Приход	Расход (дети)	Остаток (сотрудники)	Остаток на раб.месте	Балансовый счет	Статья расхода	Номер цеха, сектора	Номер кладов.	Тема, заказ	Изделие, стадия, этап	Подпись в получ. матер. или документа
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
		03.06.2020				600		4400							
		03.06.2020				1500		2900							
								0							
								0							
								0							
								0							
								0							
								0							
								0							
								0							
								0							
								0							
								0							
								0							
				Итого за 6 мес.	0	2100	0								

Рисунок 31 – Документ «Сортовая приходно-расходная карта»

Подробное описание и листинг кода вывода данных в документы Wordи Excelиз формы «Остатки продуктов» приведены в приложении А.

5.4.6 Разработка формы «Школьное меню»

Для распределения меню по категориям (завтрак, обед и т.д.) в форму добавлен элемент tabControl с соответствующими категориям вкладками (рисунок 32).

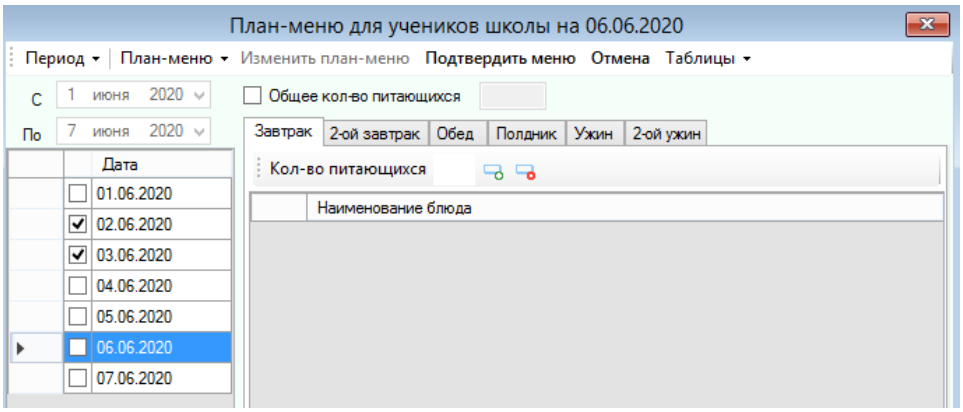


Рисунок 32 – Интерфейс формы «Школьное меню»

Рядом с tabControl расположен dataGridViewс датами по текущей неделе.

Таблица с датами заполняется с помощью прописанной функции Fill-DateGridView. Она определяет, какой сегодня день недели и дата, затем в цикле заполняется таблица по сегодняшней дате. После того, как даты заполнены, идёт проверка на наличие меню по дате. Если меню есть, то рядом с датой ставится галочка в checkBox. У функции FillDateGridView есть перегрузка в виде параметра типа DateTime. Данная перегрузка нужна для выбора недели в команде «Период».

ЛистингкодафункцииFillDateGridView представлен в приложении Б.

Командная панель формы содержит следующие команды:

1. Период –позволяет переключать недели. Есть три варианта переключения – на неделю вперед, на неделю назад и выбор недели в первом dateTimePicker’e.

2. План-меню – позволяет открыть форму технологических карт и остатков продуктов. Также экспортирует в шаблон Wordменю-раскладку на выбранный день, либо на всю неделю. Помимо меню-раскладки экспортируется меню-требование на выдачу продуктов по выбранному дню.

3. Изменить план-меню – команда может быть недоступной, если меню на выбранный день ещё не добавлено в базу данных (не отмечен checkBoxв data-GridViewс датами), иначе она разрешает редактирование меню.

4. Подтвердить меню – команда подтверждения меню.

5. Отмена – команда отмены редактирования.

Для редактирования меню прописана функция EditMenu с тремя параметрами типа Boolean: параметр для доступа к необходимым кнопкам для редактирования меню и доступа самих таблиц с меню, параметр для ограничения доступа к кнопке «Изменить план-меню» и таблице с датами, и параметр, определяющий право пользователя на редактирование. Для работы с таблицами меню в каждой вкладеtabControl есть командная панель с кнопками добавления и удаления строки. Пользователь может удалить уже подтвержденную в меню позицию после того как подтвердит удаление. Код функции представлен в листинге 11.

Листинг 11 – Код функции EditMenu

```
//метод открытия полей для редактирования меню
Privatevoid EditMenu(bool v1, bool v2, bool Enabled)
{
if (Enabled == true) //если пользователю разрешено редактирование меню
            {
//скрываем таблицы для отображения меню
                breakfastGridView.Visible = dinnerGridView.Visible =
afterdinnerGridView.Visible = eveningGridView.Visible
= seceveningGridView.Visible = secbreakfastGridView.Visible = v1;
//делаем доступными для нажатия командные панели для редактирования таблиц
toolStrip2.Enabled = toolStrip3.Enabled = toolStrip4.Enabled = toolStrip5.Enabled = tool-
Strip6.Enabled = toolStrip7.Enabled = v2;
//открываютсятаблицыдляредактированияменю
                dataGridView1.Visible = dataGridView2.Visible = dataGridView3.Visible = data-
GridView4.Visible
                    = dataGridView5.Visible = dataGridView6.Visible = v2;
//становятся доступными для нажатия кнопки подтверждения и отмены меню, а также checkbox для
ввода кол-ва питающихся
SubmitButton.Visible = SubCancelButton.Visible = v2;
checkBox1.Enabled = v2;
if (v1 == false) //если скрыты таблицы для отображения меню
                {//добавляем из bindingsource'ов информацию в таблицы для редактирования меню
foreach (DataRowView drv in breakfastbindingSource)
                    {
                        dataGridView1.Rows.Add(drv.Row["ID"], drv.Row["Dish"]);
                    }
foreach (DataRowView drv in secbreakfastbindingSource)
                    {
                        dataGridView6.Rows.Add(drv.Row["ID"], drv.Row["Dish"]);
                    }
foreach (DataRowView drv in dinnerbindingSource)
```

```
                    {
                        dataGridView2.Rows.Add(drv.Row["ID"], drv.Row["Dish"]);
                    }
foreach (DataRowView drv in afterdinnerbindingSource)
                    {
                        dataGridView3.Rows.Add(drv.Row["ID"], drv.Row["Dish"]);
                    }
foreach (DataRowView drv in eveningbindingSource)
                    {
                        dataGridView4.Rows.Add(drv.Row["ID"], drv.Row["Dish"]);
                    }
foreach (DataRowView drv in seceveningbindingSource)
                    {
                        dataGridView5.Rows.Add(drv.Row["ID"], drv.Row["Dish"]);
}
                }
if (v1 == true) //иначе чистим таблицы для редактирования меню
{
                    dataGridView1.Rows.Clear();        dataGridView6.Rows.Clear();        data-
GridView2.Rows.Clear();    dataGridView3.Rows.Clear();    dataGridView4.Rows.Clear();    data-
GridView5.Rows.Clear();}}
```

Чтобы вывести количество питающихся прописана функция GetPersons, которая получает из источника данных текущую запись меню и запоминает значение столбца Persons. Код функции представлен в листинге 12.

Листинг 12 – Код функции GetPersons

```
Private void GetPersons()
        {//получаем текущие запииси из каждой категории меню
DataRowView bdrv = breakfastbindingSource.Current as DataRowView;
            DataRowView abdrv = secbreakfastbindingSource.Current as DataRowView;
            DataRowView ddrv = dinnerbindingSource.Current as DataRowView;
            DataRowView addrv = afterdinnerbindingSource.Current as DataRowView;
            DataRowView edrv = eveningbindingSource.Current as DataRowView;
DataRowView sedrv = seceveningbindingSource.Current as DataRowView;
//если выбранная запись не нулевая, то добавляем кол-во питающихся в текстовые поля по
категориям
if (bdrv != null) { breakfastPers.Text = bdrv.Row["Persons"].ToString();}
else { breakfastPers.Text = ""; }
if (abdrv != null) { secbreakfastPers.Text = abdrv.Row["Persons"].ToString(); }
else { secbreakfastPers.Text = ""; }
if (ddrv != null) { dinnerPers.Text = ddrv.Row["Persons"].ToString(); }
else { dinnerPers.Text = ""; }
if (addrv != null) { afterdinnerPers.Text = addrv.Row["Persons"].ToString(); }
else { afterdinnerPers.Text = ""; }
if (edrv != null) { eveningPers.Text = edrv.Row["Persons"].ToString(); }
else { eveningPers.Text = ""; }
if (sedrv != null) { seceveningPers.Text = sedrv.Row["Persons"].ToString(); }
else { seceveningPers.Text = ""; }}
```

Длядобавленияиобновленияменюпрописаныдвефункции – Submit-WithCheckиSubmitWithNoCheck. Первая функция подтверждает меню с общим заданным количеством питающихся, а вторая – с прописанным количеством питающихся под каждой категорией меню.При подтверждении меню функции

проверяют, заполнял ли пользователь каждую категорию или нет. Если какие-то категории не заполнены, меню всё равно будет подтверждено, но пользователю будет выдано предупреждение. Ещё одна проверка связана с триггером таблицы Menu_stud, который проверяет наличие продуктов в остатках. Если в каком-то из блюд продукта будет недостаточно для списания, пользователю будет выдана ошибка подтверждения меню и указание конкретной причины появления ошибки.

Листинг кода обработчика события кнопки подтверждения меню и функций SubmitWithCheck, SubmitWithNoCheck представлен в приложении В.

Команда «План-меню» позволяет открыть формы для справочной информации для составления меню – технологические карты и остатки продуктов. Из этой команды можно открыть форму «Корректировка списания».

Также данная команда экспортирует в Wordплан-меню на неделю, либо план-меню на выбранный день. В первом случае на каждый день создается отдельный документ Word, составленный по заранее созданному шаблону Word (.dot). Все документы сохраняются в заранее заданную в настройках директорию.

Ещё один экспортируемый документ – меню-требование на выдачу продуктов. Оно составляется по выбранному дню недели. Пример составленного документа представлен в рисунке 33.

МЕНЮ-ТРЕБОВАНИЕ НА ВЫДАЧУ ПРОДУКТОВ ПИТАНИЯ №_____

Утверждена МФ РФ 01.12.2010 г. Пр. №157Н

УТВЕРЖДАЮ

Руководитель учреждения _____ (подпись) Кашпурова Т.Н. (ФИО руководителя)

«3» июня 2020 г.

Коды категорий		Плановая стоимость одного дня, руб.	Количество довольствующихся по плановой стоимости дня	Плановая стоимость на всех довольствующихся, руб.	Фактическая стоимость, руб.	Персонал (количество человек)
суммарных категорий	по плановой стоимости одного дня					
Завтрак						
2й завтрак						
Обед						
Полдник						
Ужин						
2й ужин						
			ВСЕГО:			

На «3» июня 2020 г.

Учреждение МБОУ СКОШ

Структурное подразделение шк № 128

Материально-ответственное лицо Сысоева

	КОДЫ
Форма по ОКУД	0504202
Дата	
по ОКПО	

Продукты питания		Ед. изм.	Количество продуктов питания подлежащих закладке																													Расход продуктов питания (количество)	
наименование	код																															операция	
			Чай черный	Каша пшенная	Вафли	Картофель отв. с маслом	Компот из смешанных	Грубочка сложная	Чай черный	Филе минтая	Компот из смешанных	Кефир	Шоколад																			на довольствующихся	на персонал
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
Кол-во порций			**25**	**25**	**10**	**25**	**25**	**10**	**10**	**10**	**10**	**10**	**10**																					
Выход-вес порций			1250	6875	10	8125	2000	170	500	3600	800	10	10																					
Чай черный цейлон			30						30/300																								
Сахар песок			20	20/500			20/500		20/200		20/200																						
Гречневая крупа				70																													
Молоко 3,2%				90		90/2250																											
Соль				45		45/1125																											

Рисунок 33 – Документ «Меню-требование на выдачу продуктов питания»

Шаблон документа «Меню-требование на выдачу продуктов» представляет собой документ формата А3, оформленный по утвержденному школой стандарту. Заполняются только те данные, которые находятся в компетенции школьной столовой, а именно – количество питающихся, какие продукты понадобятся для списания и на какие блюда идут списания продуктов. В таблице списаний показывается количество людей, которое будет питаться конкретным блюдом и общий вес блюда (количество порций, умноженное на количество питающихся). По блюду составляется список продуктов на списание. По каждому продукту считается, какое количество списалось по выбранному блюду на одну порцию и количество списанного продукта на все порции. Например, на одну порцию черного чая списывается 30 грамм чая, а на 10 порций – 300 грамм. Помимо этого, в документе указывается сотрудник, который является ответственным за формирование меню-требования, дата формирования, каким подразделением было сформировано (какой школой) и поля для подтверждения меню-требования врачом, директором и самим сотрудником, оформившим требование.

Подробное описание и листинг кода вывода данных в Wordиз формы «Школьное меню» и «Меню для сотрудников школы» приведены в приложении Г.

5.4.7 Разработка формы «Меню для сотрудников школы»

Форма «Меню для сотрудников» выглядит идентично форме «Школьное меню» и работает точно также. Единственное их отличие – категории меню. Если в школьном меню можно сформировать план-меню на завтрак, второй завтрак, обед, полдник, ужин и второй ужин, то в меню для сотрудников план-меню формируется по двум категориям – завтрак и обед.

На рисунке 34 изображен интерфейс формы «Меню для сотрудников школы».

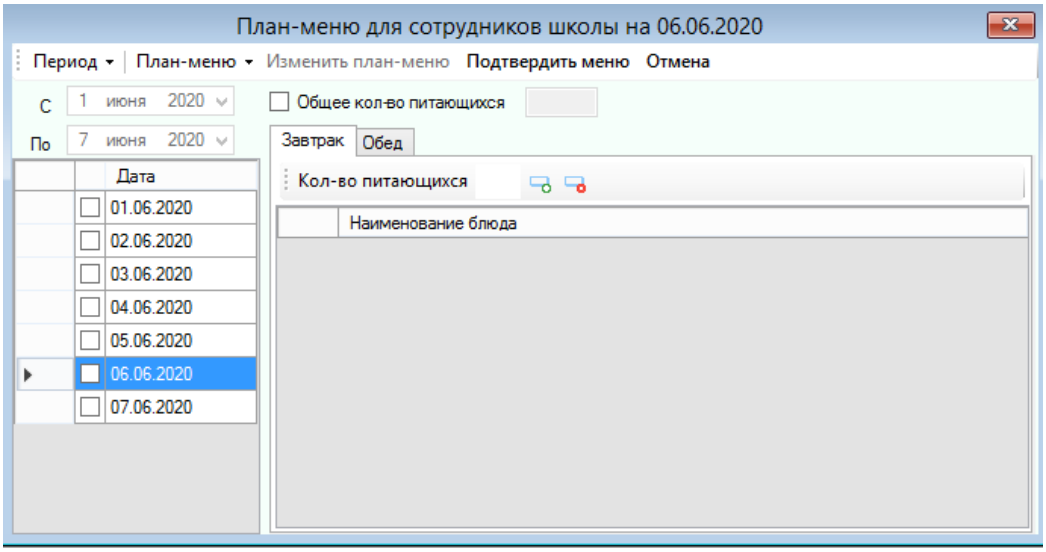


Рисунок 34 – Интерфейс формы «Меню для сотрудников школы»

5.4.8 Разработка формы «Настройки программы»

Настройки программы разделены на пять блоков:

1. Настройкиплана-меню.
2. Настройки склада.
3. Настройки поставок.
4. Настройки пользователей – данные пользователя и права пользователей.
5. Режим администратора.

Для каждого блока создан свой пользовательский элемент управления (UserControl), чтобы не перегружать форму настроек большим количеством элементов. Переключаться между блоками помогает элемент управления treeView.

Каждая настройка хранится в параметрах проекта Properties.Settings (рисунок 35).Параметры могут быть двух уровней доступа – приложение (доступно только для чтения) и пользователь (доступно и чтение, и запись). К параметру можно обратиться с помощью директивы Properties.Settings.Default.

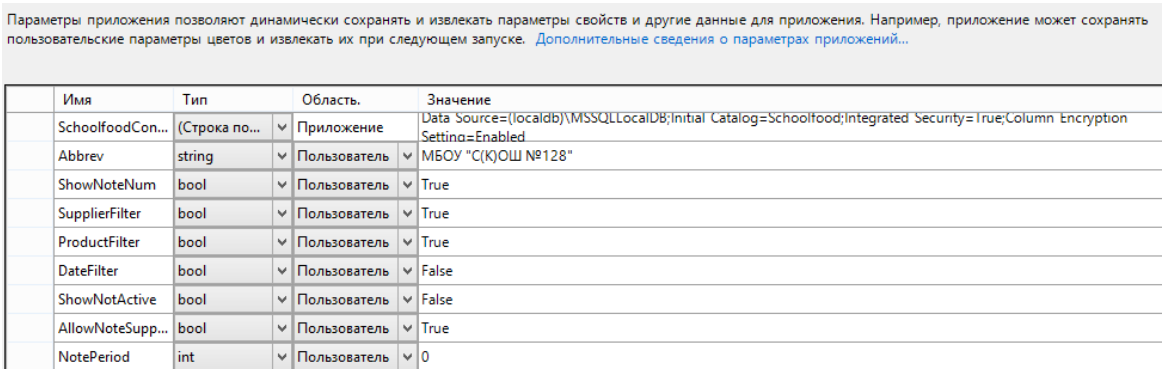


Рисунок 35 – Параметры приложения в файле Properties

В каждом блоке есть скрытые кнопки «Сохранить» и «Отмена». Кнопки открываются только при изменении настройки. При нажатии первой кнопки, текущие настройки сохранят свои значения и обновятся в параметрах проекта. При нажатии «Отмена» все обновленные настройки вернут свои старые значения.

Блок «Настройки плана-меню» содержит настройку шаблонов для вывода планов-меню и меню-требования, настройка показа количества питающихся (выводить общее количество питающихся человек или нет), возможность выбирать период для отображения меню, настройка для сохранения файлов в выбранную директорию.

На рисунке 36 показан интерфейс блока «Настройки плана-меню».

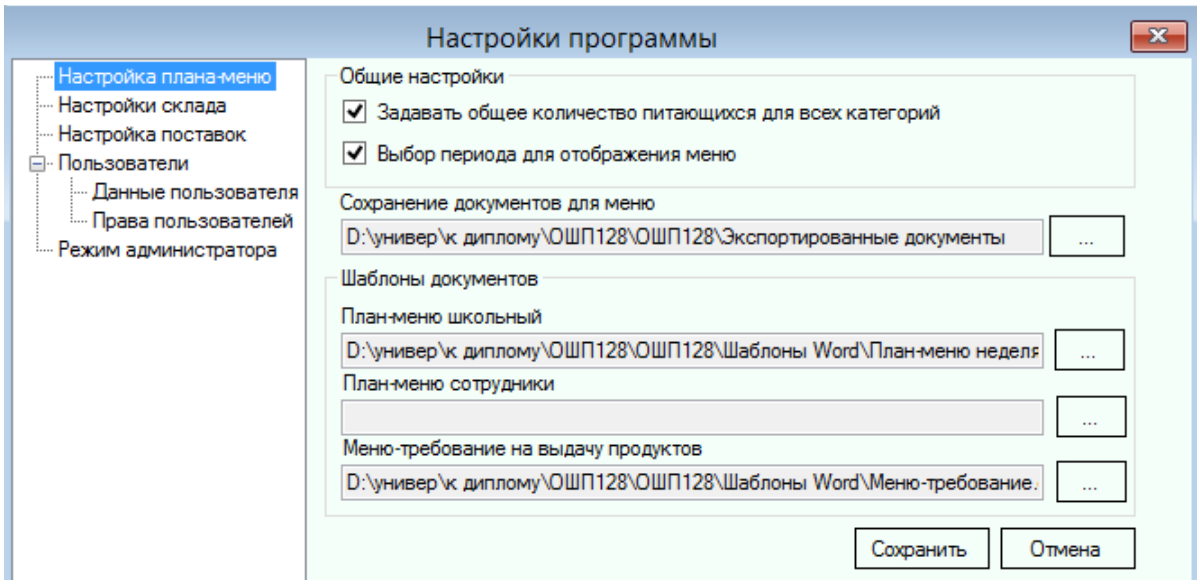


Рисунок 36 – Интерфейс блока «Настройки плана-меню»

Блок «Настройки склада» (рисунок 37) содержит настройку вывода в статус склада продуктов с минимальным остатком (для дозаказа), настройку

шаблонов документов для склада, настройку директивы для сохранения документов по складу, а также настройка вывода отчета о расходе продуктов и ведомости на заказ продуктов.

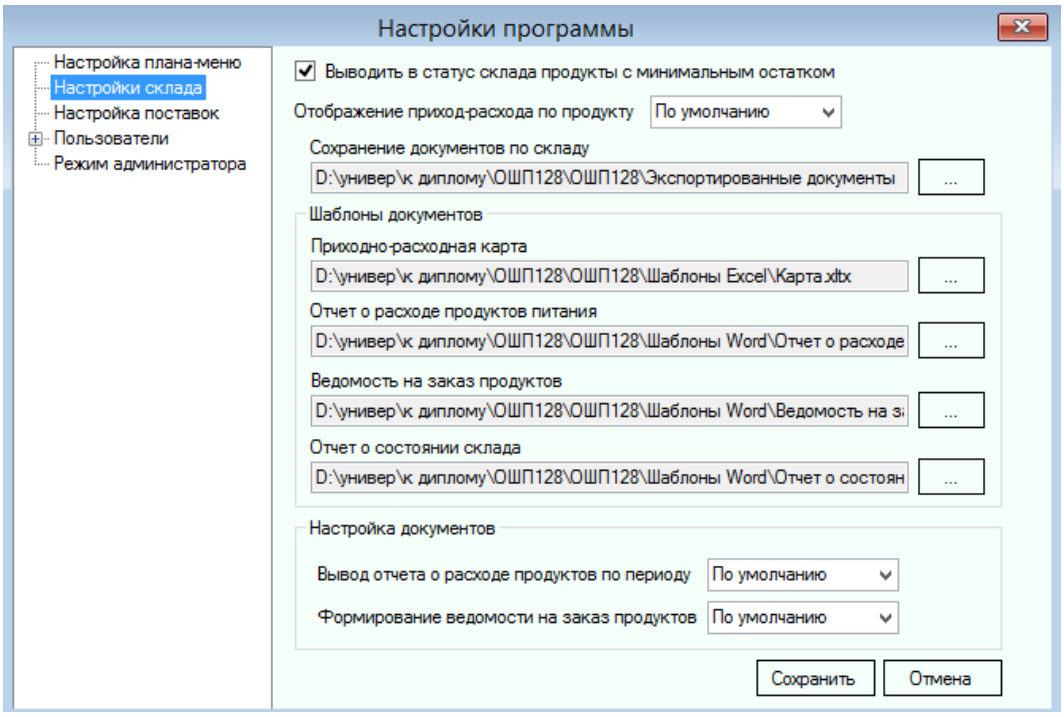


Рисунок 37– Интерфейс блока «Настройки склада»

Отчет о расходе продуктов можно формировать по месяцу (стоит по умолчанию), по неделе, по кварталу и по полугодию. Ведомость на заказ продуктов по умолчанию формируется по состоянию склада, но можно настроить по меню на неделю.

Блок «Настройки поставок» настраивает отображение накладных (фильтрация по месяцу (стоит по умолчанию), по кварталу и по полугодию), возможность отображения номера накладной в таблице, по каким полям искать данные в таблице. Дополнительно указаны параметры для формы «Поставщики» - отображение неактивных поставщиков и возможность перехода к накладной от поставщика.

На рисунке 38 изображен интерфейс блока «Настройки поставок».

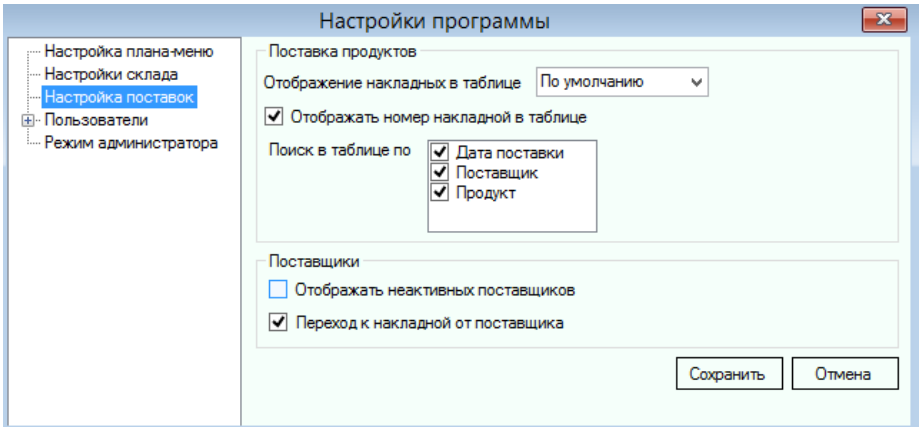


Рисунок 38 – Интерфейс блока «Настройки поставок»

Блок «Настройки пользователей» делится на два раздела – «Данные пользователя» и «Права пользователей».

«Данные пользователя» (рисунок 39) включают в себя настройку имени пользователя и пароля. Должность может изменить только администратор и только в режиме администратора.

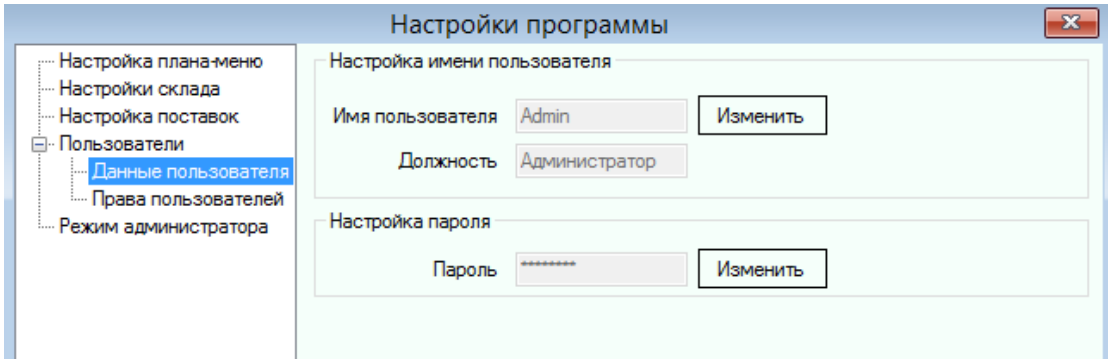


Рисунок 39 – Интерфейс блока «Данные пользователя»

«Права пользователя» (рисунок 40) отображают таблицу пользователей программы и таблицу прав по выбранному пользователю. Пользователю, которому разрешено редактирование прав, можно изменять права других пользователей, кроме администратора.

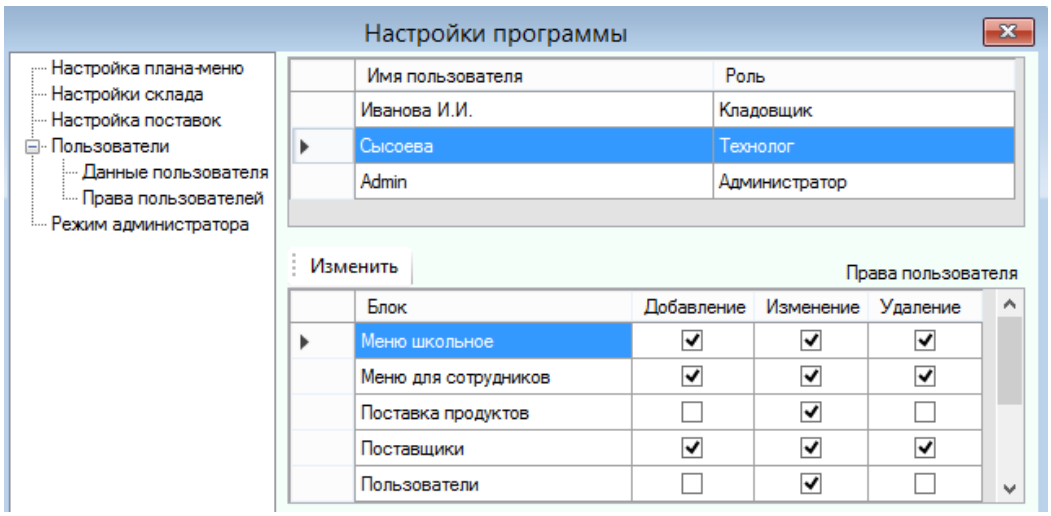


Рисунок 40 – Интерфейс блока «Права пользователя»

Блок «Режим администратора» (рисунок 41) доступен только для администратора. Если любой другой пользователь попытается открыть этот блок, программа выдаст ему ошибку доступа.

В данном блоке администратор может просматривать таблицу DB_Changes, т.е. смотреть какие процедуры и в какое время происходили с таблицами. Помимо этого, администратор из блока может просмотреть формы с таблицами Responsible, Out_kitchenи Recepies. Все три формы недоступны обычному пользователю и их можно открыть только из режима администратора. Эти таблицы добавлены в режим для того чтобы можно было добавить или удалить пользователя, или откорректировать какую-либо запись во всех таблицах без вхождения в базу данных.

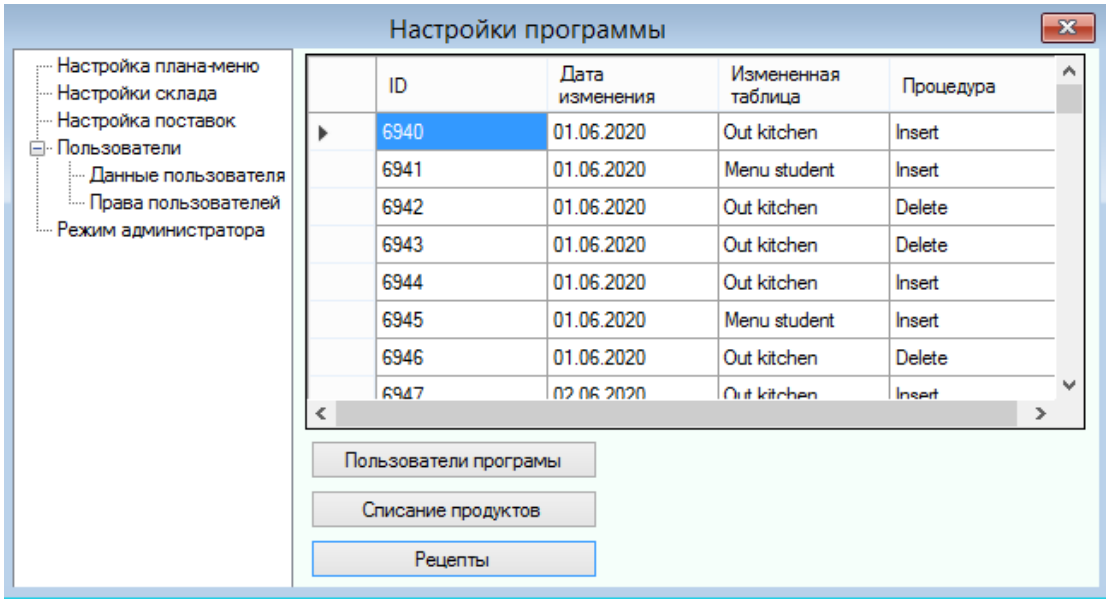


Рисунок 41 – Интерфейс блока «Режим администратора»

В каждой форме прописана функция LoadSettings, инициализирующая настройки из Properties.Settings.Default. Код одной из функйи LoadSettingsпредставлен в листинге 13.

Листинг 13 – Код функцииLoadSettings для формы «Остатки продуктов»

```
using Setting = ОШП128.Properties.Settings;
privatevoid LoadSettings()
        {
if (Setting.Default.ShowStatusProduct == false) //еслипоказнастатуспродуктовнеразрешен
            {
                statusStrip1.Visible = false; //скрываемстатуснуюстроку
}
if (Setting.Default.PeriodSortCard == 0) //еслипериодотображениясортовойкартыстоитпоумолчанию
{
                view_Sort_Card1BindingSource.Filter = ""; //фильтрповсемупериоду
            }
if (Setting.Default.PeriodSortCard == 1) //еслипериодотображения - месяц
            {
                view_Sort_Card1BindingSource.Filter = "Date >= '" + first + "' AND Date <= '"
+ last + "'";
                filter = view_Sort_Card1BindingSource.Filter; //присваиваемпеременной filter
значениефильтра
            }
if (Setting.Default.PeriodSortCard == 2) //еслипериодотображения - квартал
            {
                view_Sort_Card1BindingSource.Filter = "Date >= '" + first + "' AND Date <= '"
+ last.AddMonths(2) + "'";
                filter = view_Sort_Card1BindingSource.Filter;
            }
if (Setting.Default.PeriodSortCard == 3) //еслипериодотображения - полгодда
            {
                view_Sort_Card1BindingSource.Filter = "Date >= '" + first + "' AND Date <= '"
+ last.AddMonths(5) + "'";
                filter = view_Sort_Card1BindingSource.Filter;
}
//если период для вывода отчета о расходе продуктов стоит по умолчанию (неделя)
if (Setting.Default.PeriodforReportOut == 0 || Setting.Default.PeriodforReportOut == 1)
```

```
            {
                last1 = endweek();
                view_Out1BindingSource.Filter = "Date >= '" + startweek() + "' AND Date <= '"
+ last1 + "'";
            }
if (Setting.Default.PeriodforReportOut == 2) //еслипериодо - 10 дней
            {
                last1 = endweek().AddDays(3);
                view_Out1BindingSource.Filter = "Date >= '" + startweek() + "' AND Date <= '"
+ last1 + "'";
            }
if (Setting.Default.PeriodforReportOut == 3) //еслипериод - месяц
            {
                last1 = last;
                view_Out1BindingSource.Filter = "Date >= '" + first + "' AND Date <= '" +
last1 + "'";
}
//если период для формирования заказа стоит по умолчанию (по состоянию склада)
if (Setting.Default.PeriodforOrder == 0 || Setting.Default.PeriodforOrder == 2)
            {
                forOrderbindingSource.Filter = "[Вид] = 'Расход'";
            }
//если период для формирования - по меню на неделю
if (Setting.Default.PeriodforOrder == 1)
            {
                forOrderbindingSource.Filter = "[Вид] = 'Расход' AND [Date] >= '" + start-
week() + "' AND [Date] <= '" + endweek() + "'";
}
        }
private DateTime startweek() // возвращает дату начала недели
        {
if (now.DayOfWeek == DayOfWeek.Monday) //если текущий день - понедельник
            {
return now;
            }
if (now.DayOfWeek == DayOfWeek.Tuesday) //если текущий день - вторник
            {
returnnow.AddDays(-1);
            }
if (now.DayOfWeek == DayOfWeek.Wednesday) //если текущий день - среда
            {
returnnow.AddDays(-2);
            }
if (now.DayOfWeek == DayOfWeek.Thursday) // если текущий день - четверг
            {
returnnow.AddDays(-3);
            }
if (now.DayOfWeek == DayOfWeek.Friday) //если текущий день - пятница
            {
returnnow.AddDays(-4);
            }
if (now.DayOfWeek == DayOfWeek.Saturday) //если текущий день - суббота
            {
returnnow.AddDays(-5);
            }
if (now.DayOfWeek == DayOfWeek.Sunday) //если текущий день - воскресенье
{
return now.AddDays(-6);
            }
return now;
}
private DateTime endweek() //возвращает дату конца недели
{
```

```
if (now.DayOfWeek == DayOfWeek.Monday)
            {
return now.AddDays(6);
            }
if (now.DayOfWeek == DayOfWeek.Tuesday)
            {
return now.AddDays(5);
            }
if (now.DayOfWeek == DayOfWeek.Wednesday)
            {
return now.AddDays(4);
            }
if (now.DayOfWeek == DayOfWeek.Thursday)
            {
return now.AddDays(3);
            }
if (now.DayOfWeek == DayOfWeek.Friday)
            {
return now.AddDays(2);
            }
if (now.DayOfWeek == DayOfWeek.Saturday)
            {
return now.AddDays(1);
            }
if (now.DayOfWeek == DayOfWeek.Sunday)
{
return now;
            }
return now;
        }
```

ПодробныйлистингкодафункцийLoadSettingsдлявсехформпредставленвприложенииД.

Вывод по разделу пять

Описаны этапы разработки программы, показан интерфейс основных форм. Разработаны функции для упрощения работы некоторых форм. Настроен экспорт данных в Wordи Excel.

6 ОРГАНИЗАЦИОННО-ЭКОНОМИЧЕСКАЯ ЧАСТЬ

6.1 Технико-экономическое обоснование

В разделе дается технико-экономическое обоснование разработки программы «ОШП».

Для работы с программой необходимо отдельное рабочее место для сотрудников школьной столовой. Поэтому данный дипломный проект предусматривает покупку необходимого оборудования и материалов для бесперебойной работы.

Персональный компьютер сотрудника должен иметь лицензированный пакет программ Microsoftи MicrosoftOffice, лицензированную операционную систему Windows, настроенное подключение к принтеру.

Контроль над работой программ и работой компьютера в целом лежит на инженере-программисте школы. Помимо этого, он должен контролировать работу локальной сети компьютера для бесперебойной работы базы данных.

6.2 Организационная часть

Для установки оборудования и программного обеспечения необходим инженер-программист, работающий в школе.

Оклад инженера-программиста представлен в таблице 6.1.

Таблица 6.1 – Таблица окладов сотрудников, вовлеченных в организационный этап

Должность	Количество	Оклад руб./мес.
Инженер-программист	1	25 000

Этапы подготовки оборудования и ПО для работы с программой представлены в таблице 6.2.

Все этапы выполнены школьным инженером-программистом.

Таблица 6.2 – Этапы подготовки оборудования и ПО

Этап	Содержание работы	Вид отчетности о работе	Продолжи-тельность
Подготовительный	Изучение подходящей версии Windows для работы с программой	Согласование с директором школы и местным муниципалитетом	1
	Изучение подходящей для работы с программой версии MSOffice	Согласование с директором школы и местным муниципалитетом	1
	Выбор сборки ПК под требования ОС	Согласование с директором школы и местным муниципалитетом	2
	Выбор принтера для печати документации	Согласование с директором школы и местным муниципалитетом	1
Закупка оборудования	Покупка ПК для сотрудников столовой	Товарный чек, гарантийный талон	1
	Покупка принтера для ПК	Товарный чек, гарантийный талон	1
Установка ОС		Результат работы	2
Установка ПО	УстановкапрограммMicrosoft: SQL Server Management, Visual Studio 2017 Community	Результат работы	2
	УстановкапакетаMS Office: Word, Excel, PowerPoint, Outlook	Результат работы	1
Настройка принтера	Настройка принтера в параметрах ПК для вывода документов	Результат работы	1
Внедрение программы		Результат работы	1
		ВСЕГО:	14

Заработная плата инженера-программиста представлена в таблице 6.3. Зарплата рассчитывается как оклад деленый на 22 (количество рабочих дней в месяце).

Таблица 6.3 – Заработная плата сотрудников, вовлеченых в организационный этап

Должность	Оклад (руб.)	Дневная оплата (руб.)	Кол-во часов	Зарплата (руб.)

Инженер-программист	25 000	1136,36	14	15 910
Дополнительная заработная плата (премиальные – 30% от оклада)				7 500
Заработная плата + премиальные				23 410

6.3 Экономическая часть

В экономической части рассчитываются затраты на приобретение оборудования и ПО для работы с программой. Все затраты представлены в таблице 6.4.

Таблица 6.4 – Затраты на приобретение ПО и оборудования

№ п/п	Наименование	Единица измерения	Количество	Цена (руб.)
1	Монитор 16.5" HP 19ka	шт	1	2 900
2	Проводная компьютерная клавиатураОКЛИК 530S	шт	1	450
3	Мышь компьютерная ОКЛИК 485 MW	шт	1	450
4	Материнская плата ASUSPRIMEH310M-RR2.0, LGA 1151v2	шт	1	3 450
5	Процессор INTEL Pentium Gold G5400, LGA 1151v2	шт	1	4 290
6	Блокпитания LINKWORLD LW2-350W (LPE) case	шт	1	870
7	Жесткийдиск Seagate BarraCuda 500 Гб	шт	1	2 899
8	Корпус mATX LINKWORLD VC-13M35, черный	шт	1	1 190
9	Оперативнаяпамять AMD Radeon R7 Performance 4 ГБ	шт	1	1 399
10	Операционнаясистема Microsoft Windows 7 Professional RU x32/x64 ESD	шт	1	2 490
11	Пакет программ Microsoft Office 2007	шт	1	2 140
12	Принтерлазерный HP LaserJet Pro M104a	шт	1	4 299
			ИТОГО	26 827

Общие затраты на зарплату инженеру-программисту и покупку оборудования и ПО для сотрудников столовой равны 50 237 рублей.

Вывод по разделу шесть

Рассчитаны затраты на покупку и оборудование рабочего места для сотрудников столовой.

ЗАКЛЮЧЕНИЕ

Разработана программа под названием «Организация школьного питания», полностью удовлетворяющая поставленным целям проекта.

Для программы создана и нормализована база данных в MSSQLServer. В таблицы добавлены данные, созданы представления этих таблиц, разработаны процедуры для работы с данными. База имеет защиту паролей для пользователей программы.

С программой работают три пользователя – технолог, кладовщик и администратор. Администратор имеет защищенный вход. Технолог и Кладовщик могут работать без пароля. У каждого пользователя есть свой набор прав для работы с программой. Права можно редактировать в настройках программы, доступ к редактированию определяет Администратор.

В программе можно составлять меню для школьников и для сотрудников школы. Составление меню не ограничено текущей неделей, его можно составлять на любой момент времени, учитывая остатки продуктов на складе. Редактировать меню также можно в любое время.

Разработан удобный интерфейс программы, позволяющий пользователю легко ориентироваться в формах.

Составлены справочники программы – продукты питания, технологические карты и поставщики. Во всех справочниках можно добавлять, редактировать и удалять информацию без потери данных (например, по поставке продуктов).

Специально для программы составлены шаблоны документов по утвержденным стандартам школы. Также настроен вывод данных из программы в эти шаблоны и их сохранение в заранее заданную в настройках директорию.

Рассчитаны затраты на приобретение рабочего места для сотрудников столовой и программного обеспечения для работы с программой. Подробно расписаны этапы подготовки места.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Доценко, В.А. Практическое руководство по надзору за организацией питания и здоровья населения: учебное пособие для ВУЗов / В.А. Доценко – Москва: Изд-во Фолиант Россия, 2006 – 312 с.

2 «Вижен:Софт Питание в школе» – https://pitaniesoft.ru/nutrition_programs/school_food/.

3 1С:Предприятие 8 Общепит – https://solutions.1c.ru/catalog/public-catering.

4 1С:Предприятие 8 Ресторан – https://solutions.1c.ru/catalog/restaurant/features.

5 Таблицы – SQL Server – https://docs.microsoft.com/ru-ru/sql/relational-databases/tables/tables?view=sql-server-ver15.

6 ТриггерыDML – SQL Server – https://docs.microsoft.com/ru-ru/sql/relational-databases/triggers/dml-triggers?view=sql-server-ver15/.

7 Представления – SQLServer – https://docs.microsoft.com/ru-ru/sql/relational-databases/views/views?view=sql-server-ver15.

8 Хранимые процедуры – SQLServer – https://docs.microsoft.com/ru-ru/sql/relational-databases/stored-procedures/stored-procedures-database-engine?view=sql-server-ver15

ПРИЛОЖЕНИЕ А

Листинг кода экспорта данных из формы «остатки продуктов» в документы Word и Excel

```
private Excel.Application excelapp;       // Программа Excel
private Word.Application WordApp;       // Программа Word
private Word.Document WordDocument; // Документ
private Word.Range WordRange;            // Выделенныйдиапазон
privatevoidпечатьКартыПоВыбранномуПродуктуToolStripMenuItem_Click(object sender, EventArgs e)
        {
            excelapp = new Excel.Application();
            excelapp.Workbooks.Open(Properties.Settings.Default.SortCardPath);
            excelapp.Range["B3"].Select(); excelapp.ActiveCell.Value = StoreNumTb.Text;
            excelapp.Range["D3"].Select(); excelapp.ActiveCell.Value = NomenTb.Text;
            excelapp.Range["F3"].Select(); excelapp.ActiveCell.Value = UnitTb.Text;
            excelapp.Range["A6"].Select(); excelapp.ActiveCell.Value = NameTb.Text;
excelapp.Range["E30"].Select(); excelapp.ActiveCell.Value = "Итогоза " + now.Month + " мес.";
int i = 15;
            forExcelBindingSource.Filter = filter + " AND [ID] = " + ID;
foreach (DataRowView drv in forExcelBindingSource)
            {
                excelapp.Range["C" + Convert.ToString(i)].Select(); excelapp.ActiveCell.Value
= drv.Row["Date"];
if (drv.Row["Вид"].ToString() == "Приход")
                {
if(i==15)
                    {
int count = RestToMonth() + Convert.ToInt32(drv.Row["Count"]);
                        excelapp.Range["J15"].Select();
                        excelapp.ActiveCell.Value = count;
                    }
                    excelapp.Range["D" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv.Row["Kind"];
                    excelapp.Range["G" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv.Row["Count"];
                }
if (drv.Row["Вид"].ToString() == "Расход")
                {
if (drv.Row["Kind"].ToString() == "Menu_stud")
                    {
if (i == 15)
                        {
int count = RestToMonth() - Convert.ToInt32(drv.Row["Count"]);
                            excelapp.Range["J15"].Select();
                            excelapp.ActiveCell.Value = count;
                        }
                        excelapp.Range["H" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv.Row["Count"];
                    }
if (drv.Row["Kind"].ToString() == "Menu_sotr")
                    {
if (i == 15)
                        {
int count = RestToMonth() - Convert.ToInt32(drv.Row["Count"]);
                            excelapp.Range["J15"].Select();
                            excelapp.ActiveCell.Value = count;
                        }
                        excelapp.Range["I"      +      Convert.ToString(i)].Select();      ex-
celapp.ActiveCell.Value = drv.Row["Count"];
```

```
        }
        if (drv.Row["Kind"].ToString() == "Correct")
        {
        if (i == 15)
                                {
        int count = RestToMonth() - Convert.ToInt32(drv.Row["Count"]);
                                    excelapp.Range["J15"].Select();
                                    excelapp.ActiveCell.Value = count;
                                }
                                excelapp.Range["D" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = "КС";
                                excelapp.Range["I" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv.Row["Count"];
                            }
                        }
                        i++;
                    }
        string month = now.ToLongDateString();
                    month = month.Remove(0, 2);
                    month = month.Remove(month.Length - 7, 7);
        string filename = Properties.Settings.Default.SavePathStore + "\\" + month + " " +
now.Year + " " + drv.Row["Name"].ToString() + " - " + drv.Row["Number_ID"].ToString() +
".xlsx";
        excelapp.ActiveWorkbook.SaveAs(filename);
                }
        privatevoidпечатьВсехКартToolStripMenuItem_Click(object sender, EventArgs e)
                {
        foreach (DataRowView drv in viewStore1BindingSource)
                    {
                        excelapp = new Excel.Application();
                        excelapp.Workbooks.Open(Properties.Settings.Default.SortCardPath);
                        excelapp.Range["B3"].Select(); excelapp.ActiveCell.Value = Store-
NumTb.Text;
                        excelapp.Range["D3"].Select(); excelapp.ActiveCell.Value =
drv.Row["Number_ID"].ToString();
                        excelapp.Range["F3"].Select(); excelapp.ActiveCell.Value =
drv.Row["Unit"].ToString();
                        excelapp.Range["A6"].Select(); excelapp.ActiveCell.Value =
drv.Row["Name"].ToString();
        excelapp.Range["E30"].Select(); excelapp.ActiveCell.Value = "Итогоза " + now.Month + "
мес.";
        int i = 15;
                        forExcelBindingSource.Filter = "[ID] = " + drv.Row["ID"] + " AND [Date]
<= '" + now + "'";
        foreach (DataRowView drv1 in forExcelBindingSource)
                        {
                            excelapp.Range["C" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv1.Row["Date"];
        if (drv1.Row["Вид"].ToString() == "Приход")
                            {
        if (i == 15)
                                {
        int count = RestToMonth() + Convert.ToInt32(drv.Row["Count"]);
                                    excelapp.Range["J15"].Select();
                                    excelapp.ActiveCell.Value = count;
                                }
                                excelapp.Range["D" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv1.Row["Kind"];
```

```
       excelapp.Range["G" + Convert.ToString(i)].Select(); excelapp.ActiveCell.Value =
drv1.Row["Count"];
                            }
       if (drv1.Row["Вид"].ToString() == "Расход")
if (drv1.Row["Kind"].ToString() == "Menu_stud")
{
if (i == 15)
                            {
int count = RestToMonth(Convert.ToInt32(drv.Row["ID"])) - Convert.ToInt32(drv1.Row["Count"]);
                                excelapp.Range["J15"].Select();
                                excelapp.ActiveCell.Value = count;
                            }
                            excelapp.Range["H" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv1.Row["Count"];
                        }
if (drv1.Row["Kind"].ToString() == "Menu_sotr")
                        {
if (i == 15)
                            {
int count = RestToMonth(Convert.ToInt32(drv.Row["ID"])) - Convert.ToInt32(drv.Row["Count"]);
                                excelapp.Range["J15"].Select();
                                excelapp.ActiveCell.Value = count;
                            }
                            excelapp.Range["I" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv1.Row["Count"];
                        }
if (drv1.Row["Kind"].ToString() == "Correct")
                        {
if (i == 15)
                            {
int count = RestToMonth(Convert.ToInt32(drv.Row["ID"])) - Convert.ToInt32(drv.Row["Count"]);
                                excelapp.Range["J15"].Select();
                                excelapp.ActiveCell.Value = count;
                            }
                            excelapp.Range["D" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = "КС";
                            excelapp.Range["I" + Convert.ToString(i)].Select(); ex-
celapp.ActiveCell.Value = drv1.Row["Count"];
                        }
                    }
                    i++;
                }
string month = now.ToLongDateString();
                month = month.Remove(0, 2);
                month = month.Remove(month.Length - 7, 7);
string filename = Properties.Settings.Default.SavePathStore + "\\" + month + " " + now.Year +
" " + drv.Row["Name"].ToString() + " - " + drv.Row["Number_ID"].ToString() + ".xlsx";
                excelapp.ActiveWorkbook.SaveAs(filename);
                excelapp = null;
excelapp.Quit();
            }
        }
privatevoidотчетОРасходеПродуктовПитанияToolStripMenuItem_Click(object sender, EventArgs e)
        {
            WordApp = new Word.Application();
try
            {
```

```
WordDocument = WordApp.Documents.Add(Properties.Settings.Default.ReportOutPath);
            }
catch (Exception ex)
            {
          WordDocument.Close();
WordApp.Quit();
                WordDocument = null;
                WordApp = null;
throw ex;
            }
            WordDocument.Bookmarks["Organisation"].Range.Text = Setting.Default.Organization;
            WordDocument.Bookmarks["Date"].Range.Text = DateTime.Today.ToShortDateString();
            WordDocument.Bookmarks["Firstdate"].Range.Text = first.ToShortDateString();
            WordDocument.Bookmarks["Lastdate"].Range.Text = last1.ToShortDateString();
WordDocument.Bookmarks["Resp"].Range.Text = "ИвановаИ.И.";
            WordRange = WordDocument.Bookmarks["TableRep"].Range;
            WordRange.Tables.Add(WordRange, view_Out1BindingSource.Count+1, 4);
            Word.Table t1 = WordDocument.Tables[1];
            t1.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;
            t1.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;
            t1.Rows.SetHeight((float)0.5, Word.WdRowHeightRule.wdRowHeightAtLeast);
            WordRange = t1.Cell(1, 1).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Код";
            WordRange = t1.Cell(1, 2).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Названиепродукта";
            WordRange = t1.Cell(1, 3).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Датасписания";
            WordRange = t1.Cell(1, 4).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Кол-вопродукта";
int Row = 2;
foreach (DataRowView drv in view_Out1BindingSource)
            {
                WordRange = t1.Cell(Row, 1).Range;
                WordRange.Text = drv.Row["Number_ID"].ToString();
                WordRange = t1.Cell(Row, 2).Range;
                WordRange.Text = drv.Row["Name"].ToString();
                WordRange = t1.Cell(Row, 3).Range;
                WordRange.Text = Convert.ToDateTime(drv.Row["Date"]).ToShortDateString();
                WordRange = t1.Cell(Row, 4).Range;
                WordRange.Text = drv.Row["Count"].ToString();
                Row++;
            }
string filename = Setting.Default.SavePathStore + "\\Отчеторасходепродуктовна " +
DateTime.Today.ToLongDateString() + ".docx";
            WordDocument.SaveAs2(filename);
            WordDocument.Close();
            WordApp.Quit();
            WordDocument = null;
            WordApp = null;
            MessageBox.Show("Файл 'Отчеторасходепродуктовна " +
DateTime.Today.ToShortDateString() + "' сохранен", "Сохранениефайла");
}
```

```
PrivatevoidотчетОСостоянииСкладаToolStripMenuItem_Click(object sender, EventArgs e){
            WordApp = new Word.Application();
try
            {
                WordDocument =
WordApp.Documents.Add(Properties.Settings.Default.ReportStorePath);
            }
catch (Exception ex)
{
                WordDocument.Close();
                WordApp.Quit();
                WordDocument = null;
                WordApp = null;
throw ex;
            }
            WordDocument.Bookmarks["Organization"].Range.Text = Setting.Default.Organization;
            WordDocument.Bookmarks["Date"].Range.Text = DateTime.Today.ToShortDateString();
WordDocument.Bookmarks["Resp"].Range.Text = "ИвановаИ.И.";
            WordRange = WordDocument.Bookmarks["TableRep"].Range;
            WordRange.Tables.Add(WordRange, viewStore1BindingSource.Count+1, 3);
            Word.Table t1 = WordDocument.Tables[1];
            t1.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;
            t1.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;
            t1.Rows.SetHeight((float)0.5, Word.WdRowHeightRule.wdRowHeightAtLeast);
            WordRange = t1.Cell(1, 1).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Код";
            WordRange = t1.Cell(1, 2).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Названиепродукта";
            WordRange = t1.Cell(1, 3).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Остаток";
int Row = 2;
foreach (DataRowView drv in viewStore1BindingSource)
            {
                WordRange = t1.Cell(Row, 1).Range;
                WordRange.Text = drv.Row["Number_ID"].ToString();
                WordRange = t1.Cell(Row, 2).Range;
                WordRange.Text = drv.Row["Name"].ToString();
                WordRange = t1.Cell(Row, 3).Range;
                WordRange.Text = drv.Row["Rest"].ToString();
                Row++;
            }
string filename = Setting.Default.SavePathStore + "\\Отчетосостояниискладана " +
DateTime.Today.ToLongDateString() + ".docx";
            WordDocument.SaveAs2(filename);
            WordDocument.Close();
            WordApp.Quit();
            WordDocument = null;
            WordApp = null;
            MessageBox.Show("Файл 'Отчетосостояниискладана " +
DateTime.Today.ToShortDateString() + "' сохранен","Сохранениефайла");
}
privatevoidсформироватьВедомостьНаЗаказПродуктовToolStripMenuItem_Click(object sender, Even-
tArgs e)
{
                DialogResult ans = MessageBox.Show("Программа предложит Вам СВОЙ вариант
заказа продуктов, исходя из остатков и минимально " +
```

```
"возможного кол-ва продукта на складе. Вы можете самостоятельно отредактировать ведомость в
документе.", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);
if (ans == DialogResult.OK)
                {
                    WordApp = new Word.Application();
try
                    {
                        WordDocument =
WordApp.Documents.Add(Properties.Settings.Default.ReportOrderPath);
                    }
catch (Exception ex)
                    {
                        WordDocument.Close();
                        WordApp.Quit();
                        WordDocument = null;
                        WordApp = null;
throw ex;
                    }

                }
                WordDocument.Bookmarks["Organisation"].Range.Text = Set-
ting.Default.Organization;
                WordDocument.Bookmarks["Date"].Range.Text =
DateTime.Today.ToShortDateString();
WordDocument.Bookmarks["Resp"].Range.Text = "ИвановаИ.И.";
                WordRange = WordDocument.Bookmarks["TableRep"].Range;
                WordRange.Tables.Add(WordRange, viewStore1BindingSource.Count + 1, 3);
                Word.Table t1 = WordDocument.Tables[1];
                t1.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;
                t1.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;
                t1.Rows.SetHeight((float)0.5, Word.WdRowHeightRule.wdRowHeightAtLeast);
                WordRange = t1.Cell(1, 1).Range;
                WordRange.Bold = 1;
                WordRange.Text = "Названиепродукта";
                WordRange = t1.Cell(1, 2).Range;
                WordRange.Bold = 1;
                WordRange.Text = "Единицаизмерения";
                WordRange = t1.Cell(1, 3).Range;
                WordRange.Bold = 1;
WordRange.Text = "Кол-во продукта";
int Row = 2;
foreach (DataRowView drv in forOrderbindingSource)
                {
int rest = Convert.ToInt32(drv.Row["Rest"]);
int limit = Convert.ToInt32(drv.Row["Limit"]);
                    WordRange = t1.Cell(Row, 1).Range;
                    WordRange.Text = drv.Row["Name"].ToString();
                    WordRange = t1.Cell(Row, 2).Range;
                    WordRange.Text = drv.Row["Unit"].ToString();
int order = rest - limit; //рассчитывается разница между остатком на складе и лимитом
if (order > 0) //если заказ больше нуля
{
                        WordRange = t1.Cell(Row, 3).Range;
double d = 0.5 * limit;
                        order = order + (int)d; //к заказу добавляется половина лимита
продукта
                        WordRange.Text = order.ToString();
                    }
if (order == 0) //если заказ равен нулю
```

```
{
WordRange = t1.Cell(Row, 3).Range;
double d = 0.5 * limit;
                        order = order + limit + (int)d; //к заказу добавляется лимит продукта
и половина этого же лимита
                        WordRange.Text = order.ToString();
                    }
if (order < 0) //если заказ отрицательный
{
WordRange = t1.Cell(Row, 3).Range;
double d = 0.5 * limit;
                        order = (-1) * order + (int)d;//делаем заказ положительным числом и
добавляем половину лимита продукта
WordRange.Text = order.ToString();
}
                    Row++;
                }
string filename = Setting.Default.SavePathStore + "\\Ведомостьназаказна " +
DateTime.Today.ToLongDateString() + ".docx";
                WordDocument.SaveAs2(filename);
                WordDocument.Close();
                WordApp.Quit();
                WordDocument = null;
                WordApp = null;
                MessageBox.Show("Файл 'Ведомостьназаказна " +
DateTime.Today.ToShortDateString() + "' сохранен", "Сохранениефайла");
}
```

ПРИЛОЖЕНИЕ Б

Листинг кода функции FillDateGridView из форм «Школьное меню» и «Меню для сотрудников школы»

```
if (dow == DayOfWeek.Wednesday) //если текущий день - среда
{
                dateTimePicker1.Value = DateTime.Today.AddDays(-2);
                dateTimePicker2.Value = DateTime.Today.AddDays(4);
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-2).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-1).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(1).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(2).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(3).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(4).ToShortDateString());
                foreach (DataRowView drv in view_Menu_studBindingSource)
                {
                    for (int j = 0; j < dateGridView.RowCount; j++)
                    {
                        if (Convert.ToDateTime(drv.Row["Date"]) == Con-
vert.ToDateTime(dateGridView.Rows[j].Cells[1].Value))
                        {
                            dateGridView.Rows[j].Cells[0].Value = true;
                            dateGridView.Rows[j].Selected = false;
                        }
                    }
                }
                dateGridView.Rows[2].Selected = true;
            }
            if (dow == DayOfWeek.Thursday) //если текущий день - четверг
            {
                dateTimePicker1.Value = DateTime.Today.AddDays(-3);
                dateTimePicker2.Value = DateTime.Today.AddDays(3);
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-3).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-2).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-1).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(1).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(2).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(3).ToShortDateString());
                foreach (DataRowView drv in view_Menu_studBindingSource)
                {
                    for (int j = 0; j < dateGridView.RowCount; j++)
                    {
                        if (Convert.ToDateTime(drv.Row["Date"]) == Con-
vert.ToDateTime(dateGridView.Rows[j].Cells[1].Value))
                        {
                            dateGridView.Rows[j].Cells[0].Value = true;
                            dateGridView.Rows[j].Selected = false;
                        }
                    }
                }
                dateGridView.Rows[3].Selected = true;
            }
if (dow == DayOfWeek.Friday) //еслитекущийдень - пятница
{
dateTimePicker1.Value = DateTime.Today.AddDays(-4);
dateTimePicker2.Value = DateTime.Today.AddDays(2);
```

```
dateGridView.Rows.Add(false, DateTime.Today.AddDays(-4).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-3).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-2).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-1).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(1).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(2).ToShortDateString());
                foreach (DataRowView drv in view_Menu_studBindingSource)
                {
                    for (int j = 0; j < dateGridView.RowCount; j++)
                    {
                        if (Convert.ToDateTime(drv.Row["Date"]) == Con-
vert.ToDateTime(dateGridView.Rows[j].Cells[1].Value))
                        {
                            dateGridView.Rows[j].Cells[0].Value = true;
                            dateGridView.Rows[j].Selected = false;
                        }
                    }
                }
                dateGridView.Rows[4].Selected = true;
            }
            if (dow == DayOfWeek.Saturday) //если текущий день - суббота
            {
                dateTimePicker1.Value = DateTime.Today.AddDays(-5);
                dateTimePicker2.Value = DateTime.Today.AddDays(1);
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-5).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-4).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-3).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-2).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-1).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(1).ToShortDateString());
                foreach (DataRowView drv in view_Menu_studBindingSource)
                {
                    for (int j = 0; j < dateGridView.RowCount; j++)
                    {
                        if (Convert.ToDateTime(drv.Row["Date"]) == Con-
vert.ToDateTime(dateGridView.Rows[j].Cells[1].Value))
                        {
                            dateGridView.Rows[j].Cells[0].Value = true;
                            dateGridView.Rows[j].Selected = false;
                        }
                    }
                }
                dateGridView.Rows[5].Selected = true;
            }
            if (dow == DayOfWeek.Sunday) //еслитекущийдень - воскресенье
{
                dateTimePicker1.Value = DateTime.Today.AddDays(-6);
                dateTimePicker2.Value = DateTime.Today;
dateGridView.Rows.Add(false, DateTime.Today.AddDays(-6).ToShortDateString());
dateGridView.Rows.Add(false, DateTime.Today.AddDays(-5).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-4).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-3).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-2).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.AddDays(-1).ToShortDateString());
                dateGridView.Rows.Add(false, DateTime.Today.ToShortDateString());
                foreach (DataRowView drv in view_Menu_studBindingSource)
```

```
dateGridView.Rows.Add(false, DateTime.Today.ToShortDateString());
foreach (DataRowView drv in view_Menu_studBindingSource)
                {
                    for (int j = 0; j < dateGridView.RowCount; j++)
                    {
                        if (Convert.ToDateTime(drv.Row["Date"]) == Con-
vert.ToDateTime(dateGridView.Rows[j].Cells[1].Value))
                        {
                            dateGridView.Rows[j].Cells[0].Value = true;
                            dateGridView.Rows[j].Selected = false;
}
                    }
                }
dateGridView.Rows[6].Selected = true;
            }
        }
        //метод заполнения таблиц с датами с заданным параметром (только понедельник)
private void FillDateGridView(DateTime first)
        {
            dateGridView.Rows.Clear();
            dateTimePicker1.Value = first;
            dateTimePicker2.Value = first.AddDays(6);
            dateGridView.Rows.Add(false, first.ToShortDateString());
            dateGridView.Rows.Add(false, first.AddDays(1).ToShortDateString());
            dateGridView.Rows.Add(false, first.AddDays(2).ToShortDateString());
            dateGridView.Rows.Add(false, first.AddDays(3).ToShortDateString());
            dateGridView.Rows.Add(false, first.AddDays(4).ToShortDateString());
            dateGridView.Rows.Add(false, first.AddDays(5).ToShortDateString());
            dateGridView.Rows.Add(false, first.AddDays(6).ToShortDateString());
            foreach (DataRowView drv in view_Menu_studBindingSource)
            {
                for (int j = 0; j < dateGridView.RowCount; j++)
                {
                    if (Convert.ToDateTime(drv.Row["Date"]) == Con-
vert.ToDateTime(dateGridView.Rows[j].Cells[1].Value))
                    {
                        dateGridView.Rows[j].Cells[0].Value = true;
                        dateGridView.Rows[j].Selected = false;
}
                }
            }
dateGridView.Rows[0].Selected = true;
        }
```

ПРИЛОЖЕНИЕ В

Листинг кода обработки подтверждения меню в формах «Школьное меню» и «Меню для сотрудников школы»

```
privatevoidSubmitWithCheck()
        {
try
            {
if (PersonsTb.Text == "") //если не указано количество человек
                {//то по умолчанию ставим 0
PersonsTb.Text = "0";
                }
if (dataGridView1.RowCount != 0) //если таблица заполнена
{
                        for (int i = 0; i < dataGridView1.RowCount - 1; i++)
                        {
                            if (dataGridView1.Rows[i].Cells[0].Value == null) //если ID записи
не указан
{//то идет добавление записи в таблицу
f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 1,
                                    Convert.ToInt32(dataGridView1.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
}
else //если указан
                            {//то идет обновление записи
f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView1.Rows[i].Cells[0].Value),
                                    Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 1,
                                    Convert.ToInt32(dataGridView1.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                        }
                    }
                    if (dataGridView2.RowCount != 0)
                    {
                        for (int i = 0; i < dataGridView2.RowCount - 1; i++)
                        {
                            if (dataGridView2.Rows[i].Cells[0].Value == null)
                            {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 3,
                                    Convert.ToInt32(dataGridView2.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                            else
                            {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView2.Rows[i].Cells[0].Value),
                                    Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 3,
                                    Convert.ToInt32(dataGridView2.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
}
```

Продолжение приложения В

```
}
                    }
if (dataGridView3.RowCount != 0)
                    {
                        for (int i = 0; i < dataGridView3.RowCount - 1; i++)
                        {
                            if (dataGridView3.Rows[i].Cells[0].Value == null)
                            {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 4,
                                    Convert.ToInt32(dataGridView3.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                            else
                            {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView3.Rows[i].Cells[0].Value),
                                    Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 4,
                                    Convert.ToInt32(dataGridView3.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                        }
                    }
                    if (dataGridView4.RowCount != 0)
                    {
                        for (int i = 0; i < dataGridView4.RowCount - 1; i++)
                        {
                            if (dataGridView4.Rows[i].Cells[0].Value == null)
                            {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 5,
                                    Convert.ToInt32(dataGridView4.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                            else
                            {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView4.Rows[i].Cells[0].Value),
                                    Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 5,
                                    Convert.ToInt32(dataGridView4.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                        }
                    }
                    if (dataGridView5.RowCount != 0)
                    {
                        for (int i = 0; i < dataGridView5.RowCount - 1; i++)
                        {
                            if (dataGridView5.Rows[i].Cells[0].Value == null)
                            {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 6,
```

ПродолжениеприложенияВ

```
Convert.ToInt32(dataGridView5.Rows[i].Cells[1].Value), Convert.ToInt32(PersonsTb.Text));
                            }
                            else
                            {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView5.Rows[i].Cells[0].Value),
                                    Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 6,
                                    Convert.ToInt32(dataGridView5.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                        }
                    }
                    if (dataGridView6.RowCount != 0)
                    {
                        for (int i = 0; i < dataGridView6.RowCount - 1; i++)
                        {
                            if (dataGridView6.Rows[i].Cells[0].Value == null)
                            {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 2,
                                    Convert.ToInt32(dataGridView6.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                            else
                            {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView6.Rows[i].Cells[0].Value),
                                    Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 2,
                                    Convert.ToInt32(dataGridView6.Rows[i].Cells[1].Value),
Convert.ToInt32(PersonsTb.Text));
                            }
                        }
                    }
                this.view_Menu_studTableAdapter.Fill(this.schoolfoodDataSet.View_Menu_stud);
//обновление источника данных
                EditMenu(true, false, f1.User.EnableUpdate); //отображение добавленного меню
                dateGridView.Enabled = true;
            }
            catch (System.Data.SqlClient.SqlException ex) //если сработал триггер таблицы
Menu_stud
            {
                if (ex.Message == "The transaction ended in the trigger. Thebatchhasbeenabort-
ed.")
                {
MessageBox.Show("Недостаточно продуктов для списания. Проверьте остатки на складе.",
                        "Ошибка подтверждения", MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
        }
privatevoidSubmitWithNoCheck()
        {
try
            {
```

Продолжение приложения В

```
//если в какой-то из категорий не заполнено кол-во питающихся, по умолчанию ставится 0
if (breakfastPers.Text == "") {breakfastPers.Text = "0";}
                if (secbreakfastPers.Text == "") { secbreakfastPers.Text = "0"; }
                if (dinnerPers.Text == "") { dinnerPers.Text = "0"; }
                if (afterdinnerPers.Text == "") { afterdinnerPers.Text = "0"; }
                if (eveningPers.Text == "") { eveningPers.Text = "0"; }
                if (seceveningPers.Text == "") { seceveningPers.Text = "0"; }
                if (dataGridView1.RowCount != 0) //если таблица заполнена
                {
                    for (int i = 0; i < dataGridView1.RowCount; i++)
                    {
                        if (dataGridView1.Rows[i].Cells[0].Value == null) //если ID записи не
указан
{//то идет добавление записи в таблицу
f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 1,
                                Convert.ToInt32(dataGridView1.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(breakfastPers.Text));
                        }
                        else //иначе обновление записи
                        {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView1.Rows[i].Cells[0].Value),
                                Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 1,
                                Convert.ToInt32(dataGridView1.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(breakfastPers.Text));
                        }
                    }
                }
                if (dataGridView6.RowCount != 0)
                {
                    for (int i = 0; i < dataGridView6.RowCount; i++)
                    {
                        if (dataGridView6.Rows[i].Cells[0].Value == null)
                        {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 2,
                                Convert.ToInt32(dataGridView6.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(secbreakfastPers.Text));
                        }
                        else
                        {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView6.Rows[i].Cells[0].Value),
                                Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 2,
                                Convert.ToInt32(dataGridView6.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(secbreakfastPers.Text));
                        }
                    }
                }
                if (dataGridView2.RowCount != 0)
                {
                    for (int i = 0; i < dataGridView2.RowCount; i++)
{
```

ПродолжениеприложенияВ

```
if (dataGridView2.Rows[i].Cells[0].Value == null)
                        {
```

```
f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 3,
                                Convert.ToInt32(dataGridView2.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(dinnerPers.Text));
                        }
                        else
                        {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView2.Rows[i].Cells[0].Value),
                                Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 3,
                                Convert.ToInt32(dataGridView2.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(dinnerPers.Text));
                        }
                    }
                }
                if (dataGridView3.RowCount != 0)
                {
                    for (int i = 0; i < dataGridView3.RowCount; i++)
                    {
                        if (dataGridView3.Rows[i].Cells[0].Value == null)
                        {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 4,
                                Convert.ToInt32(dataGridView3.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(afterdinnerPers.Text));
                        }
                        else
                        {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView3.Rows[i].Cells[0].Value),
                                Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 4,
                                Convert.ToInt32(dataGridView3.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(afterdinnerPers.Text));
                        }
                    }
                }
                if (dataGridView4.RowCount != 0)
                {
                    for (int i = 0; i < dataGridView4.RowCount; i++)
                    {
                        if (dataGridView4.Rows[i].Cells[0].Value == null)
                        {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 5,
                                Convert.ToInt32(dataGridView4.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(eveningPers.Text));
                        }
                        else
                        {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView4.Rows[i].Cells[0].Value),
```



```
Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 5,
```

```
                                Convert.ToInt32(dataGridView4.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(eveningPers.Text));
                        }
                    }
                }
                if (dataGridView5.RowCount != 0)
                {
                    for (int i = 0; i < dataGridView5.RowCount; i++)
                    {
                        if (dataGridView5.Rows[i].Cells[0].Value == null)
                        {

f1.queriesTableAdapter1.INS_Menu_stud(Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1]
.Value), 6,
                                Convert.ToInt32(dataGridView5.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(seceveningPers.Text));
                        }
                        else
                        {

f1.queriesTableAdapter1.UPD_Menu_stud(Convert.ToInt32(dataGridView5.Rows[i].Cells[0].Value),
                                Con-
vert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value), 6,
                                Convert.ToInt32(dataGridView5.Rows[i].Cells[1].Value),     Con-
vert.ToInt32(seceveningPers.Text));
                        }
                    }
                }
                this.view_Menu_studTableAdapter.Fill(this.schoolfoodDataSet.View_Menu_stud);
                EditMenu(true, false, f1.User.EnableUpdate);
                dateGridView.Enabled = true;
            }
            catch (System.Data.SqlClient.SqlException ex) //если сработал триггер таблицы
Menu_stud
            {
                if (ex.Message == "The transaction ended in the trigger. Thebatchhasbeenabort-
ed.")
                {
MessageBox.Show("Недостаточно продуктов для списания. Проверьте остатки на складе.",
                        "Ошибка подтверждения", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
            }
        }
        //сохранение меню на день
        private void SubmitButton_Click(object sender, EventArgs e)
{
            //если какая-то из категорий меню не заполнена
if (dataGridView1.RowCount == 0 || dataGridView2.RowCount == 0||
                dataGridView3.RowCount == 0 || dataGridView4.RowCount == 0 ||
                dataGridView5.RowCount == 0 || dataGridView6.RowCount == 0)
{
DialogResultans = MessageBox.Show("Не все категории меню заполнены. Всё равно подтвердить
меню?", "Подтверждение меню",
MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
if (ans == DialogResult.Yes) //если пользователь подтвердил меню
{
if(checkBox1.Checked == true)
```

ОкончаниеприложенияВ

```
                    {
SubmitWithCheck();
```

```
                    }
else
                    {
SubmitWithNoCheck();
                    }
                }
            }
else //есливсекатегорииименюзаполнены
{
                if (checkBox1.Checked == true)
                {
                    SubmitWithCheck();
}
else
                {
SubmitWithNoCheck();
                }
            }
        }
```

ПРИЛОЖЕНИЕ Г

Листинг кода экспорта данных из форм «Школьное меню» и «Меню для сотрудников школы» в документ Word

```
privatevoid InsertTableInWord(BindingSource bd, string BookmarkName, int TableCount)
        {
            WordRange = WordDocument.Bookmarks[BookmarkName].Range;
            WordRange.Tables.Add(WordRange, bd.Count+1, 3);
            Word.Table t1 = WordDocument.Tables[TableCount];
            t1.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;
            t1.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;
            t1.Rows.SetHeight((float)0.5, Word.WdRowHeightRule.wdRowHeightAtLeast);
            WordRange = t1.Cell(1, 1).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Названиеблюда";
            WordRange = t1.Cell(1, 2).Range;
            WordRange.Bold = 1;
            WordRange.Text = "Вес 1 порции, г (шт)";
WordRange = t1.Cell(1, 3).Range;
            WordRange.Bold = 1;
WordRange.Text = "Выход, г (шт)";
FillTableInWord(bd, t1);
        }
privatevoid FillTableInWord(BindingSource bd, Word.Table t)
        {
int Row = 2;
foreach (DataRowView drv in bd)
            {
                WordRange = t.Cell(Row, 1).Range;
                WordRange.Text = drv.Row["Name"].ToString();
                WordRange = t.Cell(Row, 2).Range;
                WordRange.Text = drv.Row["Weight"].ToString();
                WordRange = t.Cell(Row, 3).Range;
                WordRange.Text = drv.Row["Exit"].ToString();
                Row++;
            }
        }
privatevoidпечатьПланаменюНаНеделюToolStripMenuItem_Click(object sender, EventArgs e)
        {
for (int i = 0; i < dateGridView.RowCount; i++)
            {
if (Convert.ToBoolean(dateGridView.Rows[i].Cells[0].Value) == true)
                {
                    dateGridView.Rows[i].Selected = true;
                    WordApp = new Word.Application();
try
                    {
                        WordDocument =
WordApp.Documents.Add(Properties.Settings.Default.PlanMenuStudPath);
                    }
catch (Exception ex)
                    {
                        WordDocument.Close();
                        WordApp.Quit();
                        WordDocument = null;
                        WordApp = null;
throw ex;
                    }
```

```
WordDocument.Bookmarks["Org"].Range.Text = Setting.Default.Organization;
WordDocument.Bookmarks["Date"].Range.Text = dateGridView.Rows[i].Cells[1].Value.ToString();
                    InsertTableInWord(breakfastbindingSource, "Breakfast", 1);
                    InsertTableInWord(secbreakfastbindingSource, "Secbreacfast", 2);
                    InsertTableInWord(dinnerbindingSource, "Dinner", 3);
                    InsertTableInWord(afterdinnerbindingSource, "Afterdinner", 4);
                    InsertTableInWord(eveningbindingSource, "Evening", 5);
                    InsertTableInWord(secevеningbindingSource, "Secevening", 6);
                    WordDocument.Bookmarks["Resp"].Range.Text = f1.User.UserData;
string filename = Setting.Default.SavePathMenu + "\\Менюшкольноена " +
dateGridView.Rows[i].Cells[1].Value.ToString() + ".docx";
                    WordDocument.SaveAs2(filename);
                    WordDocument.Close();
WordApp.Quit();
                    WordDocument = null;
                    WordApp = null;
                    MessageBox.Show("Файл 'Менюшкольноена " +
dateGridView.Rows[i].Cells[1].Value.ToString() + "' сохраненв "
+ Setting.Default.SavePathMenu, "Документ", MessageBoxButtons.OK, MessageBoxIcon.Information);
}
            }
        }
privatevoidпечатьПланаменюНаВыбранныйДеньToolStripMenuItem_Click(object sender, EventArgs e)
        {
            WordApp = new Word.Application();
try
            {
                WordDocument =
WordApp.Documents.Add(Properties.Settings.Default.PlanMenuStudPath);
            }
catch (Exception ex)
            {
                WordDocument.Close();
                WordApp.Quit();
                WordDocument = null;
                WordApp = null;
throw ex;
            }
            WordDocument.Bookmarks["Org"].Range.Text = Setting.Default.Organization;
            WordDocument.Bookmarks["Date"].Range.Text =
dateGridView.SelectedRows[0].Cells[1].Value.ToString();
            InsertTableInWord(breakfastbindingSource, "Breakfast", 1);
            InsertTableInWord(secbreakfastbindingSource, "Secbreacfast", 2);
            InsertTableInWord(dinnerbindingSource, "Dinner", 3);
            InsertTableInWord(afterdinnerbindingSource, "Afterdinner", 4);
            InsertTableInWord(eveningbindingSource, "Evening", 5);
            InsertTableInWord(secevеningbindingSource, "Secevening", 6);
            WordDocument.Bookmarks["Resp"].Range.Text = f1.User.UserData;
string filename = Setting.Default.SavePathMenu + "\\Менюшкольноена " +
dateGridView.SelectedRows[0].Cells[1].Value.ToString() + ".docx";
            WordDocument.SaveAs2(filename);
            WordDocument.Close();
            WordApp.Quit();
            WordDocument = null;
            WordApp = null;
```

ПродолжениеприложенияГ

```
MessageBox.Show("Файл 'Менюшкольноена " +
dateGridView.SelectedRows[0].Cells[1].Value.ToString() + "' сохраненв "
+ Setting.Default.SavePathMenu, "Документ", MessageBoxButtons.OK, MessageBoxIcon.Information);
}

privatevoidсформироватьМенютребованиеToolStripMenuItem_Click(object sender, EventArgs e)
        {
            DateTime d = Convert.ToDateTime(dateGridView.SelectedRows[0].Cells[1].Value);
string month = d.ToLongDateString();
            month = month.Remove(0, 2);
            month = month.Remove(month.Length-7, 7);
            WordApp = new Word.Application();
try
            {
                WordDocument =
WordApp.Documents.Add(Properties.Settings.Default.MenuTrebPath);
            }
catch (Exception ex)
            {
                WordDocument.Close();
                WordApp.Quit();
                WordDocument = null;
                WordApp = null;
throw ex;
            }
            WordDocument.Bookmarks["CEO"].Range.Text = "КашпуроваТ.Н.";
            WordDocument.Bookmarks["Day"].Range.Text = WordDocu-
ment.Bookmarks["Day1"].Range.Text = d.Day.ToString();
            WordDocument.Bookmarks["Month"].Range.Text = WordDocu-
ment.Bookmarks["Month1"].Range.Text = month;
            WordDocument.Bookmarks["Year"].Range.Text = WordDocu-
ment.Bookmarks["Year1"].Range.Text = d.Year.ToString();
            WordDocument.Bookmarks["Abbrev"].Range.Text = "МБОУСКОШ";
            WordDocument.Bookmarks["Schoolnum"].Range.Text = "шк № 128";
            WordDocument.Bookmarks["Resp"].Range.Text = WordDocu-
ment.Bookmarks["Resp1"].Range.Text = f1.User.UserData;
            Word.Table t = WordDocument.Tables[3];
            view_Menu_studBindingSource.Filter = "Date = '" + d + "'";
int Row = 8, Column = 4;
foreach (DataRowView drv in view_Menu_studBindingSource)
            {
                t.Cell(3, Column).Range.Text = drv.Row["Name"].ToString();
                t.Cell(6, Column).Range.Text = drv.Row["Persons"].ToString();
                t.Cell(7, Column).Range.Text = drv.Row["Exit"].ToString();
                view_Out_kitchenBindingSource.Filter
                    = "Date = '" + d + "' AND Kind_menu = 'Menu_stud' AND Kind = " +
drv.Row["Kind"] + " AND Dish = " + drv.Row["Dish"].ToString();
foreach (DataRowView drv1 in view_Out_kitchenBindingSource)
                {
int flag = 0, row_i = 0;
for (int i = 8; i <= t.Rows.Count; i++)
                    {
string srav = t.Cell(i, 1).Range.Text.Remove(t.Cell(i, 1).Range.Text.Length-2);
if (srav == drv1.Row["Name"].ToString()) //если продукт уже добавлен в таблицу
```

Окончание приложения Г

```
 {//флаг меняет значение на 1
                            flag = 1;
//запоминается строка в которую добавлен продукт
row_i = i;
```

```
                        }
                    }
if (flag == 0)
                    {
                        t.Cell(Row, 1).Range.Text = drv1.Row["Name"].ToString();
int weight_p = Convert.ToInt32(drv1.Row["Count"]) / Convert.ToInt32(drv.Row["Persons"]);
                        t.Cell(Row, Column).Range.Text = weight_p.ToString();
}
if(flag == 1)
                    {
int weight_p = Convert.ToInt32(drv1.Row["Count"]) / Convert.ToInt32(drv.Row["Persons"]);
                        t.Cell(row_i, Column).Range.Text = weight_p.ToString() + "/" +
drv1.Row["Count"].ToString();
                    }
                    Row++;
                }
                Column++;
            }
string filename = Setting.Default.SavePathMenu + "\\Меню-требованиена" +
DateTime.Today.ToLongDateString() + ".docx";
            WordDocument.SaveAs2(filename);
            WordDocument.Close();
            WordApp.Quit();
            WordDocument = null;
            WordApp = null;
            MessageBox.Show("Файл 'Меню-требованиена" + DateTime.Today.ToLongDateString() + "'
сохраненв "
                        + Setting.Default.SavePathMenu, "Документ", MessageBoxButtons.OK, Mes-
sageBoxIcon.Information);
}
```

ПРИЛОЖЕНИЕ Д

Листинг кода функции LoadSettings для форм приложения

1. Формы «Школьное меню» и «Меню для сотрудников школы».

```
Private void LoadSettings()
        {
//если нет разрешения на переключение периода (недели)
if(Setting.Default.AllowChoosePeriod == false)
{
                выбратьНеделюToolStripMenuItem.Visible = false;
            }
//если нет разрешения ставить общее количество питающихся
if(Setting.Default.ShowAllPersons == false)
            {
                checkBox1.Visible = false;
                PersonsTb.Visible = false;
}
        }
```

2. Форма «Поставка продуктов»

```
Private voidLoadSettings()
        {
//если разрешен фильтр по дате накладной
if (Setting.Default.DateFilter == true)
            {
                toolStripComboBox1.Items.Add("Датапоставки");
            }
//еслиразрешенфильтрпопродукту
if (Setting.Default.ProductFilter == true)
            {
                toolStripComboBox1.Items.Add("Продукт");
            }
//еслиразрешенфильтрпопоставщику
if (Setting.Default.SupplierFilter == true)
{
                toolStripComboBox1.Items.Add("Поставщик");
            }
//если нет разрешения на показ номера накладной
if(Setting.Default.ShowNoteNum == false)
            {
                view_NotesDataGridView.Columns[1].Visible = false;
}
//если фильтрация по дате накладной стоит по умолчанию - показывает все накладные
if(Setting.Default.NotePeriod == 0)
            {
                viewNotesBindingSource.Filter = "";
}
//если фильтрация по дате накладной = месяц - показывает накладные за весь месяц
if (Setting.Default.NotePeriod == 1)
            {
                DateTime now = DateTime.Today;
                DateTime first = new DateTime(now.Year, now.Month, 1);
                DateTime last = new DateTime(now.Year, now.Month + 1, 1).AddDays(-1);
                viewNotesBindingSource.Filter = "Date >= '" + first + "' AND Date <= '" + last
+ "'";
}
//если фильтрация по дате накладной = квартал - показывает накладные за весь квартал
if (Setting.Default.NotePeriod == 2)
```

```
{
                DateTime now = DateTime.Today;
                DateTime first = new DateTime(now.Year, now.Month, 1);
                DateTime last = new DateTime(now.Year, now.Month + 2, 1).AddDays(-1);
                viewNotesBindingSource.Filter = "Date >= '" + first + "' AND Date <= '" + last
+ "'";
}
//если фильтрация по дате накладной = полгода - показывает накладные за полгода
if (Setting.Default.NotePeriod == 3)
            {
                DateTime now = DateTime.Today;
                DateTime first = new DateTime(now.Year, now.Month, 1);
                DateTime last = new DateTime(now.Year, now.Month + 5, 1).AddDays(-1);
                viewNotesBindingSource.Filter = "Date >= '" + first + "' AND Date <= '" + last
+ "'";
}
        }
```

3. Форма «Остатки продуктов»

```
Private voidLoadSettings()
        {
if (Setting.Default.ShowStatusProduct == false) //если показан на статус продуктов не разрешен
            {
                statusStrip1.Visible = false; //скрываем статусную строку
            }
if (Setting.Default.PeriodSortCard == 0) //если период отображения сортовой карты стоит по
умолчанию
{
                view_Sort_Card1BindingSource.Filter = ""; //фильтрповсемупериоду
            }
if (Setting.Default.PeriodSortCard == 1) //еслипериодотображения - месяц
            {
                view_Sort_Card1BindingSource.Filter = "Date >= '" + first + "' AND Date <= '"
+ last + "'";
                filter = view_Sort_Card1BindingSource.Filter; //присваиваемпеременной filter
значениефильтра
            }
if (Setting.Default.PeriodSortCard == 2) //еслипериодотображения - квартал
            {
                view_Sort_Card1BindingSource.Filter = "Date >= '" + first + "' AND Date <= '"
+ last.AddMonths(2) + "'";
                filter = view_Sort_Card1BindingSource.Filter;
            }
if (Setting.Default.PeriodSortCard == 3) //еслипериодотображения - полгодда
            {
                view_Sort_Card1BindingSource.Filter = "Date >= '" + first + "' AND Date <= '"
+ last.AddMonths(5) + "'";
                filter = view_Sort_Card1BindingSource.Filter;
}
//если период для вывода отчета о расходе продуктов стоит по умолчанию (неделя)
if (Setting.Default.PeriodforReportOut == 0 || Setting.Default.PeriodforReportOut == 1)
            {
                last1 = endweek();
                view_Out1BindingSource.Filter = "Date >= '" + startweek() + "' AND Date <= '"
+ last1 + "'";
            }
if (Setting.Default.PeriodforReportOut == 2) //еслипериодо - 10 дней
```

```
{
last1 = endweek().AddDays(3);
view_Out1BindingSource.Filter = "Date >= '" + startweek() + "' AND Date <= '" + last1 + "'";
            }
if (Setting.Default.PeriodforReportOut == 3) //еслипериод - месяц
            {
                last1 = last;
                view_Out1BindingSource.Filter = "Date >= '" + first + "' AND Date <= '" +
last1 + "'";
}
//если период для формирования заказа стоит по умолчанию (по состоянию склада)
if (Setting.Default.PeriodforOrder == 0 || Setting.Default.PeriodforOrder == 2)
            {
                forOrderbindingSource.Filter = "[Вид] = 'Расход'";
            }
//если период для формирования - по меню на неделю
if (Setting.Default.PeriodforOrder == 1)
            {
                forOrderbindingSource.Filter = "[Вид] = 'Расход' AND [Date] >= '" + start-
week() + "' AND [Date] <= '" + endweek() + "'";
}
```