**Содержание**

[Техническое задание 4](#_Toc532955647)

[Введение 8](#_Toc532955648)

[1.Исследовательская часть 9](#_Toc532955649)

[1.1 Постановка задачи проектирования 9](#_Toc532955650)

[1.2 Описание предметной области 9](#_Toc532955651)

[1.3 Перечень задач, подлежащих решению в процессе разработки 10](#_Toc532955652)

[1.3 Обоснование выбора инструментов и платформы разработки 10](#_Toc532955653)

[2. Проектно-конструкторская часть 15](#_Toc532955654)

[2.1 Концептуальная схема данных 15](#_Toc532955655)

[2.2 Разработка архитектуры приложения 15](#_Toc532955656)

[3. Проектно-Технологическая часть 17](#_Toc532955657)

[3.1 Разработка руководства пользователя 17](#_Toc532955658)

[Заключение 19](#_Toc532955659)

[Список источников информации 20](#_Toc532955660)

# Техническое задание

**1. Введение**

**1.1 Наименование программы**

“Автоматизация деловых процедур логистики птицефабрики”.

**1.2 Краткая характеристика области применения**

Данное программное обеспечение может быть использовано логистическим отделом птицефабрики.

**2. Назначение разработки**

**2.1 Наименование темы разработки**

“Разработка приложения для автоматизации деловых процедур логистики птицефабрики”.

**2.2 Функциональное назначение программы**

Функциональным назначением программы повышение экономической эффективности системы путем оптимизации логистического процесса.

**3. Требования к программе**

**3.1 Требования к функциональным характеристикам**

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

* Функции планирования;
* Функции организации;
* Функции контроля;

**3.2 Требования к надежности**

Требования к обеспечению надежного (устойчивого) функционирования программы не предъявляются.

**3.3 Требования к составу и параметрам технических средств**

Минимальный набор требований к составу и параметрам технических средств будет сформирован на дальнейших этапах разработки.

**3.4 Требования к информационной и программной совместимости**

Входные данные предоставляются заказчиком.

Исходные коды программы должны быть реализованы на языке С#. В качестве интегрированной среды разработки программы должна быть использована среда Microsoft Visual Studio.

Требования к защите информации и программ не предоставляются.

**4. Технико-экономические показатели**

**4.1 Экономические преимущества разработки**

Принимая во внимание потребность предприятий поставки готовой продукции, за счёт приложение происходит уменьшение транспортной составляющей издержек благодаря эффективной обработки связи по заказам.

**5. Стадии и этапы разработки**

Содержимое разделов настоящего технического задания может быть изменено и дополнено по согласованию с руководителем.

**5.1 Стадии разработки**

Разработка должна быть проведена в пять стадий стадии:

1. Технического задания.
2. Эскизный проект.
3. Технический проект.
4. Рабочий проект.
5. Внедрение системы.

**5.2 Этапы разработки**

Стадия “ Технического задания” содержит следующие этапы:

* Обоснование необходимости разработки программы;
* Исследовательские работы;
* Разработка и утверждение технического задания.

Результатом на данной стадии будет являться тема, согласованная с руководителем, разработанное техническое задание, описание предметной области, концептуальная схема данных.

Сроки – 24 сентября.

Стадия “ Эскизного проекта” содержит следующие этапы:

* Разработка эскизного проекта;
* Утверждение эскизного проекта.

Результатом на данной стадии будет являться оформленная исследовательская часть, разработанные и реализованные алгоритмы функционирования приложения, формальное описание структуры, технологий обработки и интерфейсов. Обоснование выбора БД (при необходимости), логическая схема БД, физическая модель данных, прототип или эскиз интерфейса.

Сроки – 17 октября.

Стадия “ Технического проекта” содержит следующие этапы:

* Разработка технического проекта;
* Утверждение технического проекта.

Результатом на данной стадии будет являться оформленная проектно-конструкторская часть, работающий прототип приложения.

Сроки – 3 ноября.

Стадия “ Рабочего проекта” содержит следующие этапы:

* Разработка программы;
* Разработка программной документации;
* Испытание программы.

Результатом на данной стадии будет являться оформленная проектно-технологическая часть пояснительной записки. Окончательная версия документации.

Сроки – 1 декабря.

Стадия “ Внедрение” содержит следующие этапы:

* Подготовка и передача программы.

Результатом на данной стадии будет являться защищенная программная система.

# Введение

Целью данной курсовой работы является автоматизация деловых процедур птицефабрики и описание ключевых моментов проектирования.

Для достижения поставленной цели выли поставлены следующие задачи:

1. Обозначить и проанализировать задачи, которые необходимо решить в процессе разработки;
2. Определить инструменты и платформу разработки;
3. Разработать структуру и алгоритмы обработки информации системы;
4. Разработать приложение.

Актуальность обусловлена потенциальной возможностью повышения эффективности функционирования материалопроводящих систем с помощью планирования. Основной приоритет птицефабрики - поиск возможностей сокращения производственных затрат и издержек обращения в целях увеличения прибыли фирмы и роста качества, оказания комплекса услуг потребителю.

# 1.Исследовательская часть

## 1.1 Постановка задачи проектирования

Основное средство проектирования в рамках курсовой работы – разработка программной системы, которая поможет автоматизировать деловые процедуры логистики птицефабрики.

## 1.2 Описание предметной области

Задачей логистики птицефабрики является складирование и хранение сырья, его транспортировка заказчикам и постоянный контроль за объёмами запасов сырья.

Для работы в приложении сотруднику логистического отдела необходимо пройти идентификацию. Сотрудник добавляет новых заказчиков, заказчик представляет собой организацию, которая заинтересована в продукции птицефабрики. Сотрудник и заказчик составляют договор, в котором прописаны состав заказа и оплата по которой будет происходить заказ.Договор — это добровольное соглашение двух лиц, заключаемое на предмет выполнения каждым из них принимаемых на себя обязательств по отношению к другому участнику. В случае нарушения договора будут произведены санкции к одной из сторон. Оплата – это выдача денег по договору. Заказ – предложение заказчика поставить ему продукцию с определённом сроком, объёмом, количеством, ассортиментом. Вся готовая продукция храниться на складе и поставляется партиями, также указывается сколько партий прибыло и дата прибытия на склад. У каждого товара своя единица измерения, называемая партией. После составления состава заказ выбирается дата отгрузки товара, товар отправляется по принципу первый пришёл первый ушёл.

В случае не предвиденных ситуаций все проблемы решет сотрудник логистического отдела, отвечающий за заказа или же выше стоящие руководство.

## 1.3 Перечень задач, подлежащих решению в процессе разработки

Во время исследования предметной области и технического задания был определен ряд задач:

1. Выбор способа реализации приложения;

От способа реализации системы зависит подходы и методы решения всех поставленных задач. Также, техническое задание накладывает ряд ограничений. В связи с этим, необходимо определить наиболее подходящий вариант.

1. Разработка архитектуры приложения;

Данная задача является наиболее важной в связи с тем, что приложение может постоянно развиваться, и надо с самого начала заложить правильную архитектуру, которая позволит легко масштабировать приложение.

1. Разработка структуры базы данных:

* Анализ требований или определение цели базы данных;
* Организация данных в таблицах;
* Указание первичных ключей и анализ связей;
* Нормализация таблиц.

## 1.3 Обоснование выбора инструментов и платформы разработки

Исходя из личных предпочтений, для реализации приложения автоматизации деловых процедур птицефабрики была выбрана среда разработки Microsoft Visual Studio а в качестве языка программирования для написания приложения .NET Framework язык C#.

Microsoft Visual Studio является одной из лучших сред разработки приложений любого типа для платформы .NET Framework и имеет следующие особенности:

* является интуитивно понятной, расширяемой, унифицированной средой для языков программирования высокого уровня;
* предоставляет разработчикам набор модернизированных взаимодействующих языков .NET Framework, соответствующих имеющимся у разработчиков навыкам;
* реализует высокопроизводительные инструментальные средства для всех этапов жизненного цикла разработки программного обеспечения (ПО) – от определения требований и планирования до его последующего сопровождения.

Microsoft Visual Studio разрабатывался таким образом, чтобы обеспечить высокую надежность и совместимость создаваемых с его использованием приложений.

Выбор СУБД является одним из важных этапов разработки архитектуры системы. Основные критерии, которым должна удовлетворять СУБД следующие:

* Функциональные возможности – СУБД должна обладать всеми необходимыми инструментами для обеспечения работы системы;
* Производительность – СУБД должна обеспечивать хорошую производительность при работе с данными;
* Надежность – СУБД должна обеспечивать надежность хранения данных;
* Качество и полнота документации для работы СУБД;
* Масштабируемость – СУБД должна обеспечивать масштабируемость системы.

Наиболее подходящей СУБД, удовлетворяющей всем вышеперечисленным критериям, является PostgreSQL.

PostgreSQL является одной из самых развитых на сегодняшний день СУБД. Она свободно распространяемая и максимально соответствует стандартам SQL.

Основные преимущества данной СУБД перед аналогами следующие:

* Открытость - PostgreSQL является бесплатным ПО с открытым исходным кодом;
* Большое сообщество - существует довольно большое сообщество, благодаря чему можно быстро находить ответы на возникающие в ходе работы с СУБД вопросы;
* Большое количество различных типов данных, с возможностью создания собственных типов;
* Расширяемость СУБД за счет создания хранимых процедур – PostrgreSQL дает возможность писать хранимые процедуры на многих языках.
* Резервное копирование и восстановление – PostgreSQL предлагает несколько режимов резервного копирования и восстановления БД;
* Возможность хранения больших объемов данных – PostgreSQL позволяет хранить большие объемы данных.

Обеспечение взаимодействия системы с базой данных можно обеспечить различными способами, например, при написании приложения можно писать SQL запросы непосредственно в коде или же использовать собственноручно созданные хранимые процедуры. Однако этот подход не является оптимальным, т.к. создание отдельных SQL запросов и хранимых процедур для каждого действия может занять большое количество времени, а также не позволяет обеспечивать необходимую расширяемость системы.

Оптимальным способом обеспечения взаимодействия системы с базой данных является использование ORM. ORM (англ. Object-Relational Mapping, рус. объектно-реляционное отображение) — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». Упрощенная схема взаимодействия объектно-ориентированной системы с базой данных посредствам ORM приведена на рисунке 1.

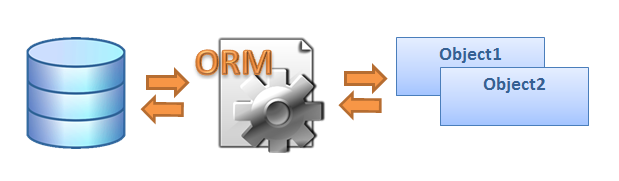


Рисунок 1 – Схема взаимодействия системы с базой данных посредствам ORM

Ключевой особенностью ORM является отображение, которое используется для привязки объекта к его данным в БД. ORM как бы создает «виртуальную» схему базы данных в памяти и позволяет манипулировать данными уже на уровне объектов. Отображение показывает, как объект и его свойства связанны с одной или несколькими таблицами и их полями в базе данных. ORM использует информацию этого отображения для управления процессом преобразования данных между базой и формами объектов, а также для создания SQL-запросов для вставки, обновления и удаления данных в ответ на изменения, которые приложение вносит в эти объекты.

Использование ORM в проекте избавляет разработчика от необходимости работы с SQL и написания большого количества кода, часто однообразного и подверженного ошибкам. Весь генерируемый ORM код предположительно хорошо проверен и оптимизирован, поэтому не нужно в целом задумывается о его тестировании.

Entity Framework представляет специальную объектно-ориентированную технологию на базе фреймворка .NET для работы с данными. Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами.

Способы взаимодействия с БД

Entity Framework предполагает три возможных способа взаимодействия с базой данных:

* Database first: Entity Framework создает набор классов, которые отражают модель конкретной базы данных
* Model first: сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере.
* Code first: разработчик создает класс модели данных, которые будут храниться в бд, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы.

# 2. Проектно-конструкторская часть

## 2.1 Концептуальная схема данных

На этапе проектирования информационной системы происходит накопление и обработка информации.

Концептуальное представление— это представление данных в том виде, какими они являются на самом деле, а не в том, какими их вынужден рассматривать пользователь в рамках, например, определенного языка или используемого аппаратного обеспечения.

Основными конструктивными элементами концептуальной моделей являются сущности, атрибуты и связи между ними.

В данной курсовой работе представлены следующие сущности: Заказчик, Заказ, Договор, Оплата, Сотрудник, Состав заказа, Номенклатура, Склад, Отгрузочная единица. Каждая сущность в свою очередь имеет список атрибутов, по которым будут осуществляться связи. Тип связи будет определять отношения между атрибутами сущности.

Концептуальное представление определяется с помощью концептуальной схемы, включающей определения для каждого существующего типа концептуальных записей (Приложение А).

## 2.2 Разработка архитектуры приложения

Для построения приложения был выбран шаблон проектирования MVVM. Он используется для разделения модели и её представления, что необходимо для изменения их отдельно друг от друга. Паттерн имеет три основных компонента:

* Модель (англ. Model), представляет собой бизнес логику и фундаментальные данные, необходимые для работы приложения;
* Представление (англ. View) — это графический интерфейс, то есть окно, кнопки и т. п. Представление является подписчиком на событие изменения значений свойств или команд, предоставляемых Моделью- Представления. В случае, если в Модели-Представления изменилось какое-либо свойство, то она оповещает всех подписчиков об этом, и Представление, в свою очередь, запрашивает обновленное значение свойства из Модели-Представления. В случае, если пользователь воздействует на какой-либо элемент интерфейса, Представление вызывает соответствующую команду, предоставленную Моделью- Представления;
  + - Модель-Представления (англ. ViewModel) является, с одной стороны, абстракцией Представления, а с другой, предоставляет обёртку данных из Модели, которые подлежат связыванию. То есть, она содержит Модель, которая преобразована к Представлению, а также содержит в себе команды, которыми может пользоваться Представление, чтобы влиять на Модель [6].

На рисунке 3 представлена диаграмма шаблона MVVM.



Рисунок 3 – Диаграмма, описывающая работу MVVM

В ходе разработки был произведен ряд отклонений от правил общей реализации данного шаблона. В результате чего, больше половины бизнес- логики оказалось во ViewModel представлении. Это в свою очередь привело к сильным информационным связям между компонентами, что сказывается на независимости отдельных модулей приложения. В связи с этим была поставлена задача на переопределение архитектуры и ее реализации, в процессе дальнейшей разработки системы.

# 3. Проектно-Технологическая часть

## 3.1 Разработка руководства пользователя

После запуска системы вы увидите окно «Идентификация» (рис. 2).

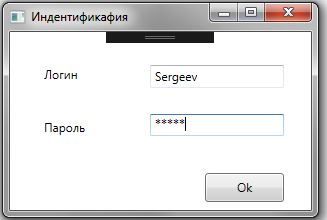


Рисунок 2 – Индетификация.

В этом окне вы должны ввести свой логин и пароль.

При нажатии на кнопку «Ок» вы попадаете в главное меню.

Главное меню сотрудника логистического отдела (рис. 3). В таблице представлены заказчики за которых отвечает сотрудник, также сотрудник может добавить заказчика, создать договор, посмотреть номенклатуру, склад и выполнить отгрузку.

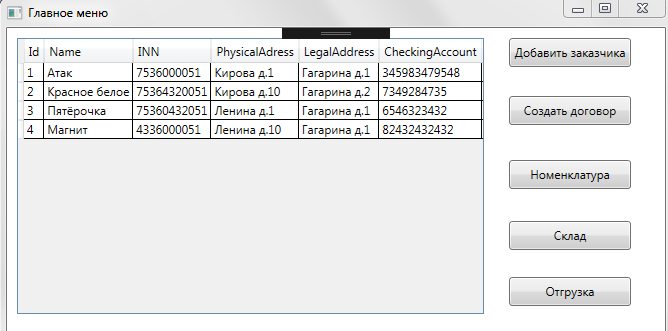


Рисунок 3 – Главное меню.

При нажатии на кнопку «Добавить заказчика» нам выводиться окно в котором необходимо заполнить информацию о заказчике (рис. 4).

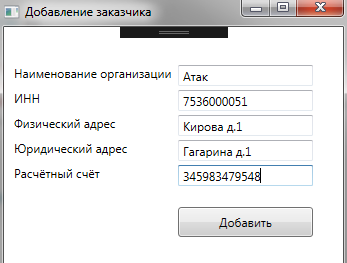


Рисунок 4 – Добавление заказчика.

При нажатии на кнопку «Добавить» добавляется заказчик в базу.

При нажатии на кнопку «Склад» нам выводиться весь ассортимент, который может быть отгружен (рис. 5).

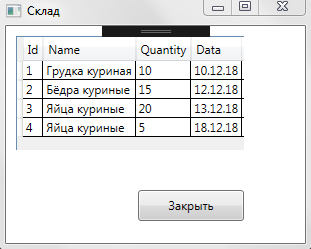


Рисунок 5 – Склад.

При нажатии на кнопку «Закрыть» вы попадаете в главное меню.

# Заключение

В результате выполнения курсовой работы было разработана приложение автоматизации деловых процедур логистики птицефабрики

Задачи, поставленные перед выполнением курсовой работы, решены.

Для достижения цели был решен ряд задач:

* Изучена и проанализирована выбранная предметная область;
* Изучены взаимосвязи между объектами исследуемой предметной области.
* Разработано приложение автоматизации деловых процедур логистики птицефабрики.

Приложение, созданное в результате выполнения курсовой работы, отвечает поставленным в начале работы требованиям и готово к эксплуатации.

# Список источников информации

1. Работа с базами данных на языке C#. Технология ADO.NET: Учебное пособие / сост. О. Н. Евсеева, А. Б. Шамшев. - Ульяновск: УлГТУ, 2009. - 170 с. (дата обращения: 9.10.2018).
2. Microsoft Visual Studio [Электронный ресурс] // Википедия — свободная энциклопедия. URL:   
   https://ru.wikipedia.org/wiki/Microsoft\_Visual\_Studio (дата обращения: 22.11.2018).
3. Habr URL: <https://habr.com/post/160845/> (дата обращения: 4.11.2018).
4. MAFREF URL: <https://magref.ru/logistika-predpriyatiya/> (дата обращения: 17.09.2018).