ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ

Государственное бюджетное профессиональное образовательное учреждение города Москвы

**«МОСКОВСКИЙ АВТОМОБИЛЬНО-ДОРОЖНЫЙ КОЛЛЕДЖ**

**ИМ. А. А. НИКОЛАЕВА»**

|  |  |
| --- | --- |
|  |  |

**КУРСОВОЙ ПРОЕКТ**

**по профессиональному модулю**

**ПМ.01 Разработка модулей программного обеспечения**

**МДК 01.01. Разработка программных модулей**

**Тема** Разработка программного обеспечения для магазина автозапчастей

Студент Халилов A.A.

Группа 4ИП2

Руководитель курсового проекта Семенова Е.В.

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата « » 2024 г.

**Москва**

**2024**

ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ

Государственное бюджетное профессиональное образовательное учреждение города Москвы

**«МОСКОВСКИЙ АВТОМОБИЛЬНО-ДОРОЖНЫЙ КОЛЛЕДЖ**

**ИМ. А. А. НИКОЛАЕВА»**

УТВЕРЖДАЮ

Председатель ПЦК

«Информационные технологии»

\_\_\_\_\_\_\_\_\_\_\_\_\_Т.Г. Сянина

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г.

**З А Д А Н И Е**

к курсовому проекту

студента группы 1ИПсоз-1.23 специальности 09.02.07 «Информационные системы и программирование»

Климова Ивана Владимировича

Тема курсового проекта Разработка программного обеспечения для предоставления строительных услуг.

1. Постановка задачи: Разработать программное обеспечение с дружественным и интуитивно понятным интерфейсом для повышения оперативности работы и просмотра всей необходимой информации, связанной с магазином автозапчастей.

Функционал программного обеспечения должен обладать возможностью хранения данных о клиентах, услугах и транспортных средствах.

2. Исходные данные: Информация о клиентах, транспортных средствах, услугах.

3. Перечень вопросов, подлежащих разработке в курсовом проекте:

1. Постановка задачи и исследование предметной области;
2. Проектирование концептуальной модели данных;
3. Проектирование информационно-логической модели данных;
4. Обоснование выбора ОС, СУБД и языка программирования;
5. Проектирование физической модели данных;
6. Технические средства реализации проекта;
7. Проектирование интерфейса пользователя. ­

Задание выдано « » \_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. Срок выполнения « » \_\_\_\_\_\_\_ 2024 г.

Студент Климов И.В. /

Фамилия И.О. подпись

Руководитель курсового проекта Семенова Е.В. / Фамилия ИО подпись

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 3](#_Toc162513717)

[РАЗДЕЛ 1. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 4](#_Toc162513718)

[1.1 Постановка задачи 4](#_Toc162513719)

[1.2 Жизненный цикл программного обеспечения 4](#_Toc162513720)

[1.3 Разработка базы данных 7](#_Toc162513721)

[1.4 Методы проектирования архитектуры ПО 7](#_Toc162513722)

[1.4.1 Диаграмма классов 8](#_Toc162513723)

[1.4.2 Диаграмма вариантов использования 8](#_Toc162513724)

[1.4.3 Диаграмма последовательности 9](#_Toc162513725)

[1.4.4 Кооперативная диаграмма 9](#_Toc162513726)

[1.4.5 Диаграмма деятельности 9](#_Toc162513727)

[РАЗДЕЛ 2. РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 10](#_Toc162513728)

[2.1 Инструкция пользователю 10](#_Toc162513729)

[ЗАКЛЮЧЕНИЕ 14](#_Toc162513730)

[СПИСОК ЛИТЕРАТУРЫ 15](#_Toc162513731)

# ВВЕДЕНИЕ

Проектирование экономических информационных систем (ЭИС) - логически сложная, трудоёмкая и длительная работа, требующая высокой квалификации разработчиков. В процессе создания и функционирования ЭИС информационные потребности пользователей меняются, уточняются, что усложняет разработку и сопровождение таких систем.

Основная доля затрат приходится на прикладное программное обеспечение (ПО) и разработку базы данных (БД).

Необходимость контроля процесса разработки программного обеспечения привела к появлению совокупности методов и средств создания ПО, объединённых общим названием «программная инженерия». В основе её заложена идея: проектирование ПО есть формальный процесс, который можно изучать и совершенствовать.

Для успешной реализации проекта объект проектирования должен быть описан при помощи полных и непротиворечивых моделей архитектуры ПО. Здесь закладываются структурные элементы системы, связи между ними, иерархия подсистем.

Модель - это полное описание системы ПО с некоторой точки зрения. Моделирование является центральным звеном всей работы по созданию качественного ПО. Модели строятся для того, чтобы понять структуру и поведение создаваемой системы, облегчить управление процессом её создания, уменьшить возможный риск и документировать принимаемые проектные решения.

Язык моделирования должен включать элементы модели (фундаментальные концепции моделирования и их семантику), нотацию (визуальное представление элементов моделирования), руководство по использованию.

Конечная цель разработки ПО - получение работающих приложений (кода).

# РАЗДЕЛ 1. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## 1.1 Постановка задачи

Необходимо разработать программное обеспечение для предоставления строительных услуг автотранспортного предприятия.

Требования к приложению.

1. Выбор пользовательских функций - Закладки.
2. Вариант диаграммы - Столбиковая.
3. Каталоги с файлами БД не определяются в программе.

## 1.2 Жизненный цикл программного обеспечения

Понятие жизненного цикла ПО (ЖЦ ПО) является одним из базовых понятий программной инженерии.

ЖЦ ПО — это период времени, который начинается с момента решения о необходимости создания ПО и заканчивается полным изъятием его из эксплуатации.

Основным нормативным документом, регламентирующим состав процессов ЖЦ ПО, является международный стандарт ISO/IEC 12207: 1995 «Information Technology - Software Life Cycle Processes». Он определяет структуру ЖЦ, содержащую процессы, действия и задачи, которые должны быть выполнены во время создания ПО (его российский аналог ГОСТ РИСО/МЭК 12207-99 введен в действие в июле 2000 г.). В данном стандарте процесс определяется как совокупность взаимосвязанных действий, преобразующих некоторые входные данные в выходные. Каждый процесс характеризуется определенными задачами и методами их решения, исходными данными, полученными от других процессов, и результатами.

Каждый процесс разделен на набор действий, каждое действие — на набор задач. Каждый процесс, действие или задача инициируется и выполняется другим процессом по мере необходимости, причем не существует заранее определенных последовательностей выполнения (естественно, при сохранении связей по входным данным).

Модель жизненного цикла программного обеспечения — структура, содержащая процессы действия и задачи, которые осуществляются в ходе разработки, использования и сопровождения программного продукта. При выборе схемы модели жизненного цикла для конкретной предметной области, решаются вопросы включения важных для создаваемого продукта видов работ или не включения несущественных работ. К широко используемым типам моделей ЖЦ относятся следующие: каскадная, спиральная, инкрементная, эволюционная, стандартизованная и др.

В данной работе выбрана каскадная модель. Согласно данной модели ЖЦ работы и задачи процесса разработки обычно выполняются последовательно. Однако вспомогательные и организационные процессы обычно выполняются параллельно с процессом разработки. В данной модели возвращение к начальному процессу предусматривается после сопровождения и исправления ошибок.

Особенность такой модели состоит в фиксации последовательных процессов разработки программного продукта. В ее основу положена модель фабрики, где продукт проходит стадии от замысла до производства, затем передается заказчику как готовое изделие, изменение которого не предусмотрено, хотя возможна замена на другое подобное изделие в случае рекламации или некоторых ее деталей, вышедших из строя.

**ФОТО**

Рисунок 1.1 – Каскадная модель ЖЦ

Недостатки этой модели:

* процесс создания ПС не всегда укладывается в такую жесткую форму и последовательность действий;
* не учитываются изменившиеся потребности пользователей, изменения во внешней среде, которые вызовут изменения требований к системе в ходе ее разработки;
* большой разрыв между временем внесения ошибки (например, на этапе проектирования) и временем ее обнаружения (при сопровождении), что приводит к большой переделке ПС.

При применении каскадной модели имеют место следующие факторы риска:

* требования к ПС недостаточно четко сформулированы, либо не учитывают перспективы развития ОС, сред и т.п.;
* большая система, не допускающая компонентной декомпозиции, может вызвать проблемы с размещением ее в памяти или на платформах, не предусмотренных в требованиях;
* внесение быстрых изменений в технологию и в требования может ухудшить процесс разработки отдельных частей системы или системы в целом;
* ограничения на ресурсы (человеческие, программные, технические и др.) в ходе разработки могут сузить отдельные возможности реализации системы.

Полученный продукт может оказаться плохим для применения по причине недопонимания разработчиками требований или функций системы, или недостаточно проведенного тестирования.

Преимущества реализации системы с помощью каскадной модели следующие:

* все задачи подсистем и системы реализуются одновременно (ни одна задача не забыта), что способствует установлению стабильных связей и отношений между ними;
* полностью разработанную систему с документацией на нее легче сопровождать, тестировать, фиксировать ошибки и вносить изменения не беспорядочно, а целенаправленно, начиная с требований (например, добавить или заменять некоторые функции) и повторить процесс.

Разработанное ПО основано на каскадной модели. Это обосновано тем, что каждая работа выполняется полностью, и после ее завершения и перехода к следующему этапу возвращение к предыдущему не требуется. Промежуточный результат проверяется известными методами верификации и фиксируется в качестве готового эталона для следующего процесса.

## 1.3 Разработка базы данных

Методика определения и документирования требований к базе данных заключается в составлении словаря данных. Словарь данных перечисляет и определяет отдельные элементы данных, которые должны храниться в базе (табл. 1).

Таблица 1

Далее модель развивается путем определения атрибутов для каждого объекта. Для этого из составленного ранее словаря данных выделяем необходимые элементы (табл.2).

Таблица 2

Для того чтобы данные схемы стала реляционной моделью необходимо использование ключей (первичных и внешних) и отношений (рис.2).

**ФОТО**

Рисунок 1.2 – Логическая модель

## 1.4 Методы проектирования архитектуры ПО

Архитектура системы - это структурная схема компонентов системы, взаимодействующих между собой через интерфейсы. Основным условием построения архитектуры системы является декомпозиция системы на

компоненты или модули. Кроме того, существует необходимость определения целей и проверка их выполнимости, определение входных и выходных данных и иерархическое представление абстракции системы.

Фактически создаваемая архитектура состоит из четырех уровней:

* 1-й уровень - системные компоненты;
* 2-й уровень - общесистемные компоненты;
* 3-й уровень - специфические компоненты определенной проблемной области;
* 4 -й уровень -прикладные программные системы, реализуют конкретные задачи отдельных групп потребителей.

При проектировании архитектуры программная система рассматривается как композиция компонент третьего уровня.

Результатом архитектурного проектирования представляются нотациями в виде диаграмм (сущность-связь, переходы состояний, потоки данных и действий и т.п.).

Структурные (structural) модели:

* диаграммы классов (class diagrams);
* диаграммы компонентов (component diagrams);
* диаграммы размещения (deployment diagrams).

Модели поведения (behavioral):

* диаграммы вариантов использования (use case diagrams);
* диаграммы взаимодействия (interaction diagrams);
* диаграммы последовательности (sequence diagrams);
* диаграммы состояний (statechart diagrams);
* диаграммы деятельности (activity diagrams).

Рассмотрим их подробнее.

### 1.4.1 Диаграмма классов

Данная диаграмма используется для моделирования статической структуры классов системы и связей между ними. На рис.7 изображен пример диаграммы классов для разрабатываемого ПО.

**ФОТО**

Рисунок 1.3 – Диаграмма классов

### 1.4.2 Диаграмма вариантов использования

Эта диаграмма используется для моделирования процессов и функциональных требований к создаваемой системе. На рис. 8 представлена диаграмма вариантов использования для разрабатываемого ПО.

**ФОТО**

Рисунок 1.4 – Диаграмма вариантов использования

### 1.4.3 Диаграмма последовательности

Диаграмма последовательности применяется для задания взаимодействия объектов, с помощью сценариев, отображающих события, связанные с их созданием и уничтожением. Взаимодействие объектов контролируется событиями, которые происходят в сценарии и поддерживаются сообщениями к другим объектам (рис 5).

**ФОТО**

Рисунок 1.5 – Диаграмма последовательности

### 1.4.4 Кооперативная диаграмма

На кооперативных диаграммах объекты (или классы) показываются в виде прямоугольников, а стрелками обозначаются сообщения, которыми они обмениваются в рамках одного варианта использования. Временная последовательность сообщений отражается их нумерацией (рис.6).

**ФОТО**

Рисунок 1.6 – Кооперативная диаграмма

### 1.4.5 Диаграмма деятельности

Диаграмма деятельности применяется для моделирования поведения системы в рамках различных вариантов использования, или потоков управления (рис.7).

**ФОТО**

Рисунок 1.7 – Диаграмма деятельности

# РАЗДЕЛ 2. РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## 2.1 Инструкция пользователю

Реализация программы и разработка приложения осуществлялась в программе «Microsoft Visual Studio C# 2022». Основой алгоритма приняты разработанные выше схемы. Итоговый вид формы в режиме конструктора представлен на рис. 8-10.

**ФОТО**

Рисунок 2.1 – Закладка «Клиенты ТС» в режиме конструктора

**ФОТО**

Рисунок 2.2 – Закладка «Заказ услуги» в режиме конструктора

**ФОТО**

Рисунок 2.3 – Закладка «Отчеты» в режиме конструктора

После запуска приложения, пользователь имеет право выбрать информацию, которую он хочет увидеть на экране с помощью соответствующих закладок.

* закладка «Клиенты и ТС»:

Вид приложения после запуска представлено на рис.11.

**ФОТО**

Рисунок 2.4 – Закладка «Клиенты и ТС»

С выведенными таблицами можно выполнить несколько действий.

1. Добавление данных в таблицу «Клиенты». Для этого необходимо ввести данные клиента и нажать кнопку «Добавить».

**ФОТО**

Рисунок 2.5 – Вводим данные

Пример результата добавления представлен на рис.13.

**ФОТО**

Рисунок 2.6 – Результаты добавления

Добавление данных в таблицу «ТС». Для этого необходимо ввести данные ТС и нажать кнопку «Добавить».

**ФОТО**

Рисунок 2.7 – Добавление нового ТС

**ФОТО**

Рисунок 2.8 – Результаты добавления

1. Поиск по таблице «Клиенты». Для этого необходимо ввести данные в любое поле таблицы. После ввода данных необходимо нажать на кнопку «Поиск». Пример результата поиска представлен на рис.16.

**ФОТО**

Рисунок 2.9 – Результаты поиска

После можно нажать кнопку «Отменить» и ввести новый параметр поиска (рис.17).

**ФОТО**

Рисунок 2.10 – Результаты поиска

Поиск по таблице «ТС». Для этого необходимо ввести данные в любое поле таблицы. После ввода данных необходимо нажать на кнопку «Поиск» (рис.18).

**ФОТО**

Рисунок 2.11 – Результаты поиска

После можно нажать кнопку «Отменить» и ввести новый параметр

поиска.

1. Удаление в таблице «Клиенты». Для этого необходимо ввести данные в поле «Код клиента» (рис.19).

**ФОТО**

Рисунок 2.12 – Вводим данные

После ввода данных необходимо нажать на кнопку «Удалить». Пример

результата удаления представлен на рис. 20.

**ФОТО**

Рисунок 2.13 – Результат удаления

Удаление в таблице «ТС». Для этого необходимо ввести данные в

поле «Код тс» (рис.21).

**ФОТО**

Рисунок 2.14 – Вводим данные

После ввода данных необходимо нажать на кнопку «Удалить». Пример

результата удаления представлен на рис. 22.

**ФОТО**

Рисунок 2.15 – Результат удаления

* закладка «Заказ услуги»

Вид приложения после запуска представлено на рис.23.

**ФОТО**

Рисунок 2.16 – Закладка «Заказ услуги»

С выведенной таблицей можно выполнить несколько действий.

1. Добавление данных в таблицу «Услуги». Для этого необходимо ввести данные услуги и нажать кнопку «Добавить».

**ФОТО**

Рисунок 2.17 – Вводим данные

Пример результата добавления представлен на рис.25.

**ФОТО**

Рисунок 2.18 – Результат добавления

1. Поиск по таблице «Услуги». Для этого необходимо ввести данные в любое поле таблицы. После ввода данных необходимо нажать на кнопку «Поиск». Пример результата поиска представлен на рис. 26.

**ФОТО**

Рисунок 2.19 – Результат поиска

После можно нажать кнопку «Отменить» и ввести новый параметр поиска

(рис.27).

**ФОТО**

Рисунок 2.20 – Результат поиска

* закладка «Отчеты»

Вид приложения после запуска представлено на рис.28.

**ФОТО**

Рисунок 2.21 – Закладка «Отчеты»

1. Отчёт 1. При нажатии на кнопку «Отчёт 1» перед пользователем предстанет следующее окно (рис. 29). В нём будет отражена таблица статистических данных типов услуг. Для наглядности вся информация представлена на диаграмме.

**ФОТО**

Рисунок 2.22 – Вывод «Отчет 1»

1. Отчёт 2. При нажатии на кнопку «Отчёт 2» перед пользователем предстанет другое окно (рис.30). В нём будет отражена таблица статистических данных заказов услуг. На диаграмме отображается на какую сумму сделаны услуги.

**ФОТО**

Рисунок 2.23 – Вывод «Отчет 2»

Чтобы посмотреть изменения графиков, добавим новый заказ на услугу (рис.31).

**ФОТО**

Рисунок 2.24 – Добавление новой услуги

**ФОТО**

Рисунок 2.25 – Результат изменения «Отчета 1»

**ФОТО**

Рисунок 2.26 – Результат изменения «Отчета 2»

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы было реализовано приложение с помощью программы «Microsoft Visual Studio C# 2022». Целью которой была автоматизация системы учета услуг автотранспортного предприятия. Для зрительной реализации полученных результатов в программе использованы диаграммы.

# СПИСОК ЛИТЕРАТУРЫ

1. Бессмертный, И. А. Системы искусственного интеллекта : учеб. пособие для СПО / И. А. Бессмертный. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2018. — 130 с. - ISBN 978-5-534-07467-3.
2. Гниденко, И. Г. Технология разработки программного обеспечения : учеб. пособие для СПО / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — М. : Издательство Юрайт, 2017. — 235 с. - ISBN 978-5-534-18131-9.
3. Гордеев, С. И. Организация баз данных в 2 ч. Часть 2 : учебник для вузов / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2019. — 501 с. - ISBN 978-5-534-11625-0.
4. Жмудь, В. А. Моделирование замкнутых систем автоматического управления : учеб. пособие для академического бакалавриата / В. А. Жмудь. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2019. — 128 с. - ISBN 978-5-534-09487-9.
5. Зыков, С. В. Программирование. Объектно-ориентированный подход : учебник и практикум для академического бакалавриата / С. В. Зыков. — М. : Издательство Юрайт, 2019. — 155 с. - ISBN 978-5-534-16941-6.
6. Иванов, В. М. Интеллектуальные системы : учеб. пособие для СПО / В. М. Иванов ; под науч. ред. А. Н. Сесекина. — М. : Издательство Юрайт, 2019. — 93 с. - ISBN 978-5-534-07819-0.
7. Иванов, В. М. Интеллектуальные системы : учеб. пособие для вузов / В. М. Иванов ; под науч. ред. А. Н. Сесекина. — М. : Издательство Юрайт, 2017. — 91 с. - ISBN 978-5-534-00551-6.
8. Кубенский, А. А. Функциональное программирование : учебник и практикум для академического бакалавриата / А. А. Кубенский. — М. : Издательство Юрайт, 2019. — 348 с. - ISBN 978-5-9916-9242-7.
9. Кудрина, Е. В. Основы алгоритмизации и программирования на языке c# : учеб. пособие для СПО / Е. В. Кудрина, М. В. Огнева. — М. : Издательство Юрайт, 2019. — 322 с. - ISBN 978-5-534-10772-2.
10. Кудрина, Е. В. Основы алгоритмизации и программирования на языке c# : учеб. пособие для бакалавриата и специалитета / Е. В. Кудрина, М. В. Огнева. — М. : Издательство Юрайт, 2019. — 322 с. - ISBN 978-5-534-10772-2.
11. Кудрявцев, К. Я. Методы оптимизации : учеб. пособие для вузов / К. Я. Кудрявцев, А. М. Прудников. — 2-е изд. — М. : Издательство Юрайт, 2019. — 140 с. - ISBN 978-5-534-08523-5.
12. Лаврищева, Е. М. Программная инженерия и технологии программирования сложных систем : учебник для вузов / Е. М. Лаврищева. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2019. — 432 с. - ISBN 978-5-534-07604-2.
13. Лебедев, В. М. Программирование на vba в ms excel : учеб. пособие для академического бакалавриата / В. М. Лебедев. — М. : Издательство Юрайт, 2019. — 272 с. - ISBN 978-5-534-15949-3.
14. Малявко, А. А. Формальные языки и компиляторы : учеб. пособие для вузов / А. А. Малявко. — М. : Издательство Юрайт, 2018. — 429 с. - ISBN 978-5-534-04288-7.
15. Мамонова, Т. Е. Информационные технологии. Лабораторный практикум : учеб. пособие для СПО / Т. Е. Мамонова. — М. : Издательство Юрайт, 2019. — 178 с. - ISBN 978-5-534-07791-9.
16. Маркин, А. В. Программирование на sql в 2 ч. Часть 2 : учебник и практикум для бакалавриата и магистратуры / А. В. Маркин. — М. : Издательство Юрайт, 2019. — 292 с. ISBN 978-5-534-18371-9.
17. Нагаева, И. А. Программирование: delphi : учеб. пособие для академического бакалавриата / И. А. Нагаева, И. А. Кузнецов ; под ред. И. А. Нагаевой. — М. : Издательство Юрайт, 2017. — 302 с. - ISBN 978-5-534-09124-3.
18. Плескунов, М. А. Операционное исчисление : учеб. пособие для вузов / М. А. Плескунов ; под науч. ред. А. И. Короткого. — М. : Издательство Юрайт, 2019. — 141 с. - ISBN 978-5-534-09142-7.

**ПРИЛОЖЕНИЕ**