|  |
| --- |
| **Министерство образования и науки Российской Федерации**  Федеральное государственное автономное образовательное  учреждение высшего образования  **«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**  **ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»** |

Подразделение: Инженерная школа энергетики

Направление подготовки: 09.04.03 – Прикладная информатика

Отделение: Электроэнергетики и электротехники

**Проектная документация**

**Отчёт по лабораторной работе №5**

по дисциплине: «Основы объектно-ориентированного программирования»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент гр. О-5КМ01 | |  | \_\_\_\_\_\_\_\_\_\_ | |  | Арбачаков М.Е. |
|  | |  |  | |  |  |
|  | |  |  |  | | |
|  | |  |  | |  |  |
| Отчёт принял | доцент, к.т.н. |  | \_\_\_\_\_\_\_\_\_\_ | |  | Калентьев А. А. |
|  |  |  |  | |  |  |
|  |  |  |  | \_\_\_ \_\_\_\_\_\_ | | |

Томск 2022

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc119108690)

[1 Основная часть 4](#_Toc119108691)

[1.1 UML диаграмма вариантов использования 4](#_Toc119108692)

[1.2 UML диаграмма классов 5](#_Toc119108693)

[1.3 Описание классов, образующих связь типа «общее-частное» 7](#_Toc119108694)

[1.4 Дерево ветвлений Git 8](#_Toc119108695)

[1.5 Тестирование программы 8](#_Toc119108696)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 17](#_Toc119108697)

**ВВЕДЕНИЕ**

Корректная и полная документация сопровождает разработку программного обеспечения (далее – ПО) от появления идеи до выпуска конечного продукта. Написание документации является обязательным критерием разработки и последующей поддержки проекта [1].

Целью данной лабораторной работы является разработка проектной документации на созданный программный продукт.

Для достижения поставленной цели должны быть выполнены следующие задачи:

* Составление технического задания (далее – ТЗ) на разработанную программу (Приложение А);
* Составление UML диаграммы вариантов использования для разработанной программы;
* Составление UML диаграммы классов;
* Описание классов, образующих связь типа «общее-частное»;
* Привести дерево ветвлений Git;
* Провести тестирование программы.

**1 Основная часть**

**1.1 UML диаграмма вариантов использования**

Вариант использования (use case) — это описание множества последовательных действий (включая вариации), которые выполняются некоторым субъектом с целью получения результата, значимого для некоторого действующего лица [1]. ВИ предполагает взаимодействие действующих лиц и системы или другого объекта. Действующее лицо представляет собой логически связанное множество ролей, которые играют пользователи системы во время взаимодействия с ней.

Диаграмма вариантов использования для разработанного ПО приведена на рисунке 1.



Рисунок 1 – Диаграмма вариантов использования

**1.2 UML диаграмма классов**

Диаграмма классов — это центральная методика моделирования, которая используется практически во всех объектно-ориентированных методах. Эта диаграмма описывает типы объектов в системе и различные виды статических отношений, которые существуют между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

Диаграмма классов приведена на рисунке 2.



Рисунок 2 – UML диаграмма классов

**1.3 Описание классов, образующих связь типа «общее-частное»**

В таблице 1 приведено описание абстрактного класса *DiscountBase* с его полями, свойствами и методами.

Таблица 1 – Описание класса *DiscountBase*

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *DiscountBase* – абстрактный базовый класс для скидок | | |
| Поля | | |
| #\_discount | int | Величина скидки |
| + MinDiscount | int | Минимальная величина скидки |
| Свойства | | |
| + Discount | int | Величина скидки |
| Методы | | |
| *+ GetResultPrice(discount value)* | double | Возвращает реальную цену товара с учетом скидки  value – стоимость покупки |
| # CheckDiscount(int value) | bool | Проверяет величину скидки  value – величина скидки |
| Конструкторы | | |
| # DiscountBase(int value) | DiscountBase | Конструктор класса  value – величина скидки |
| # DiscountBase() | DiscountBase | Конструктор по умолчанию |

В таблицах 2–3 приведены описания классов 2 *DiscountCertificate*, 3 *InterestCoupon*, которые наследуются от *DiscountBase*.

Таблица 2 – Описание класса *DiscountCertificate*

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *DiscountCertificate* – Класс скидки по скидочному сертификату | | |
| Методы | | |
| + GetResultPrice (double value) | double | Возвращает реальную цену товара с учетом скидки  value – стоимость покупки |
| Конструкторы | | |
| + DiscountCertificate(int value) | DiscountCertificate | Конструктор класса  value – величина скидки |
| + DiscountCertificate() | DiscountCertificate | Конструктор по умолчанию |

Таблица 3 – Описание класса *InterestCoupon*

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Описание класса | | |
| Класс *InterestCoupon* – Класс скидки по процентному купону | | |
| Поля | | |
| + MaxDiscount | int | Максимальный процент скидки |
| Методы | | |
| + GetResultPrice(double value) | double | Возвращает реальную цену товара с учетом скидки  value – стоимость покупки |
| # CheckDiscount | bool | Проверяет величину скидки |
| Конструкторы | | |
| + InterestCoupon(int value) | InterestCoupon | Конструктор класса  value – величина скидки |
| + InterestCoupon () | InterestCoupon | Конструктор по умолчанию |

**1.4 Дерево ветвлений Git**

На рисунке 3 представлено дерево ветвлений Git, полученное по окончании работы с проектом.

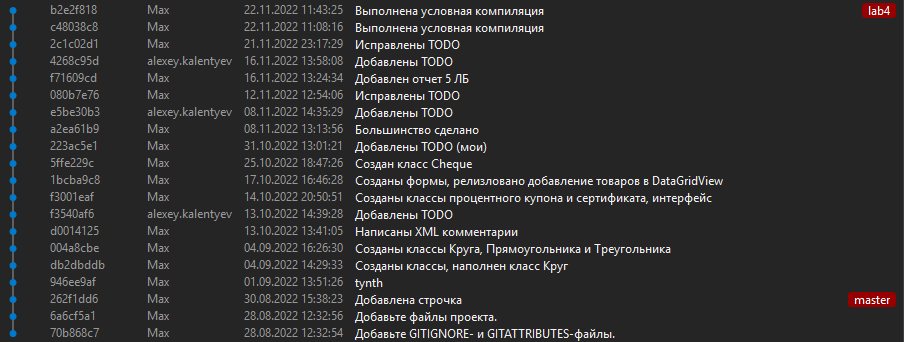


Рисунок 3 – Дерево ветвлений Git

**1.5 Тестирование программы**

Далее приводится процесс функционального тестирования программы.

Графический интерфейс пользователя представлен на рисунке 4.

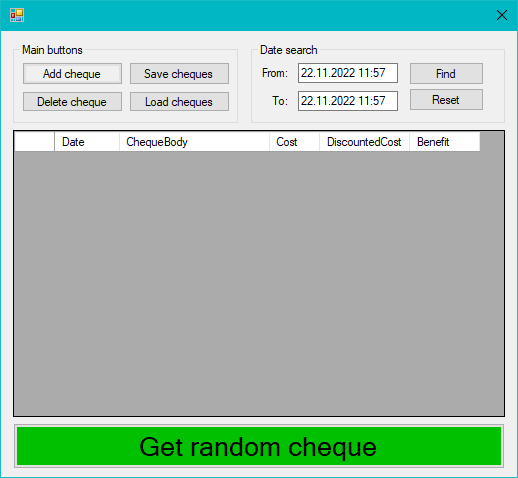


Рисунок 4 – Графический интерфейс пользователя

**1.5.1 Тестовый случай «Add cheque»**

Для добавления элемента необходимо вызвать соответствующую форму путём нажатия кнопки «Add cheque» (рисунок 5).

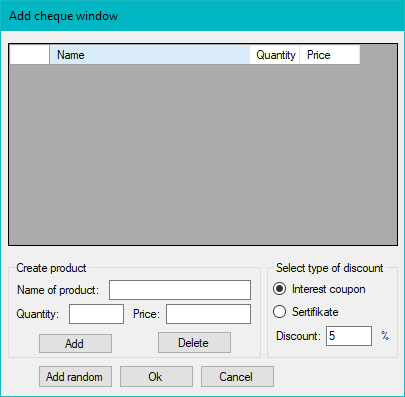


Рисунок 5 – Форма для добавления элемента

Параметры скидки (тип скидки и ее величину) можно указать с помощью переключателя и соответствующего поля. В поле возможно ввести только целые неотрицательные числа.

Для создания товара необходимо ввести название товара, его количество и цену. В поле количества товаров возможно ввести только целые неотрицательные числа, в поле цены товара возможно ввести целые или дробные неотрицательные числа. Также существует кнопки для удаления продукта и создания случайного продукта.

После ввода данных необходимо нажать кнопку «Add», продукт появится в таблице формы (рисунок 6).

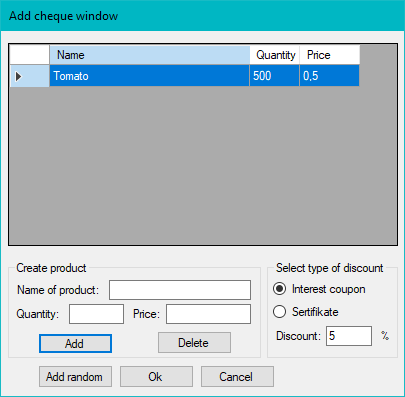
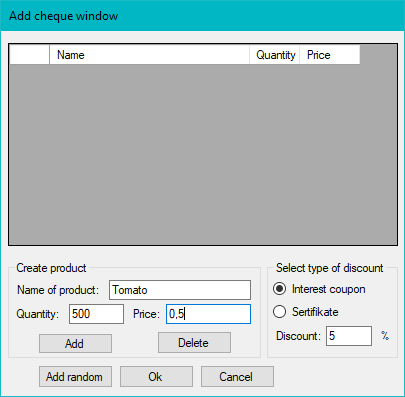


Рисунок 6 – Заполнение полей

После заполнения списка товаров необходимо нажать кнопку «Ok», тогда чек появится на главной форме, а форма создания исчезнет.

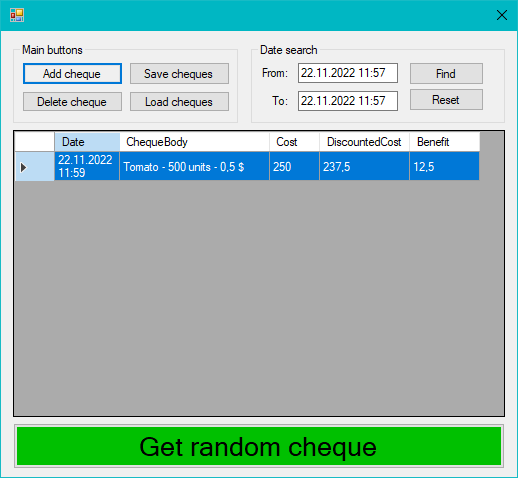


Рисунок 7 – Успешное добавление нового элемента

В программе предусмотрена система обработки некорректного ввода данных пользователем. Например, при попытке ввести число вне заранее определенного диапазона, появится соответствующее сообщение об ошибке (рисунки 8-9).

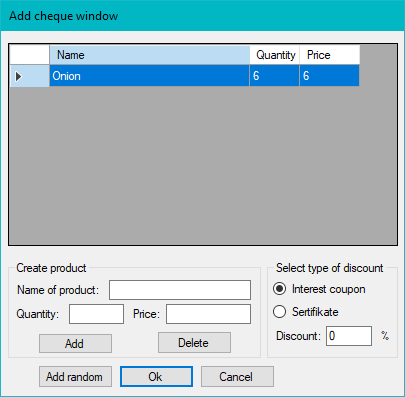
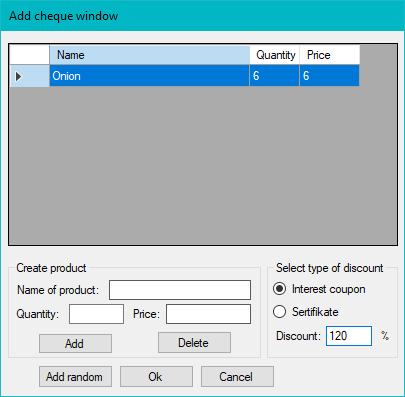
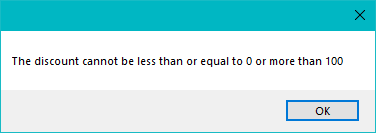


Рисунок 8 – Некорректный ввод (120% и 0)



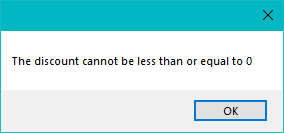


Рисунок 9 – Сообщение об ошибке

Если попытаться добавить товар без заполнения необходимых полей, , также появятся сообщения об ошибке.

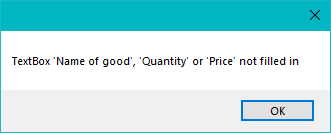


Рисунок 10 – Сообщения об ошибках

**1.5.2 Тестовый случай «Удалить элемент»**

Для удаления элемента необходимо выбрать его в таблице и нажать на кнопку «Delete cheque» (рисунки 11 и 12).

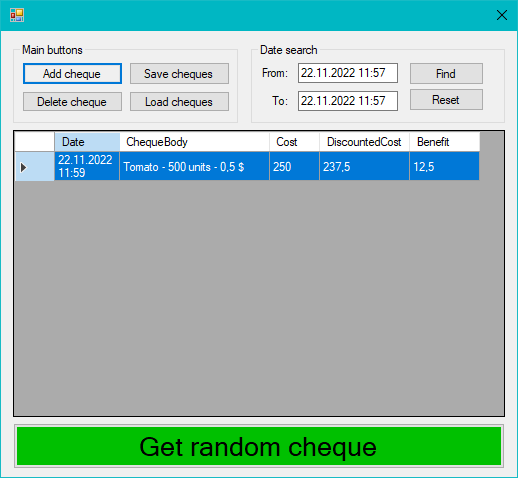


Рисунок 11 – Выбор элемента в таблице

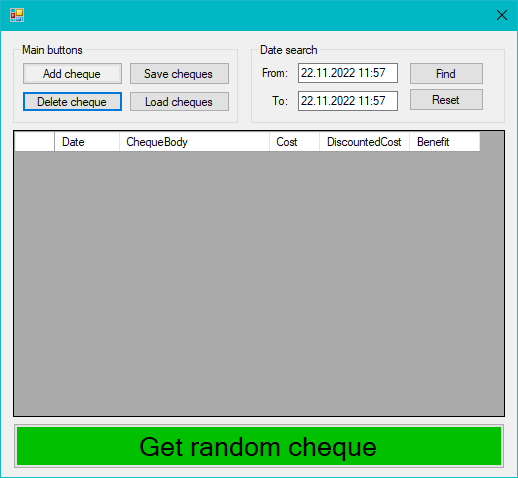


Рисунок 12 – Результат нажатия кнопки «Remove Discount»

**1.5.3 Тестовый случай «Search»**

Для поиска элементов предусмотрена панель параметров поиска (рисунок 13)

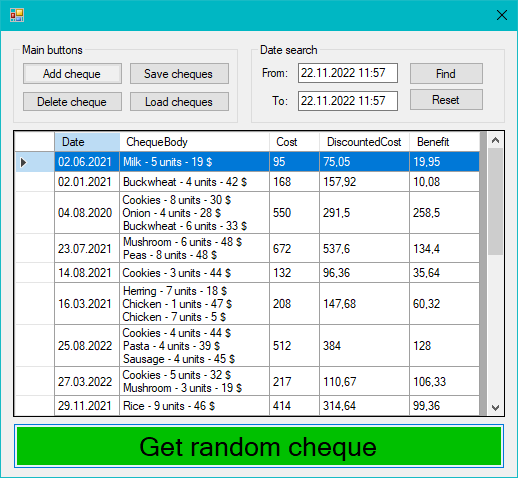


Рисунок 13 – Настройка параметров для поиска элементов

Пользователь выбирает диапазон дат, по которым требуется найти элемент, и нажимает кнопку «Find» (рисунок 14).

Для того, чтобы сбросить фильтр, предусмотрена соответствующая кнопка «Reset» (рисунок 15).

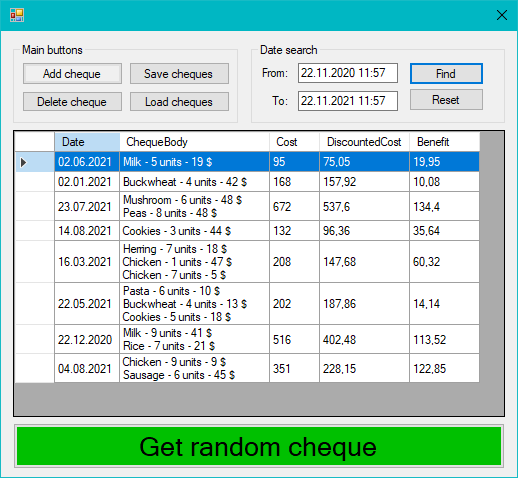


Рисунок 14 – Поиск по диапазону дат

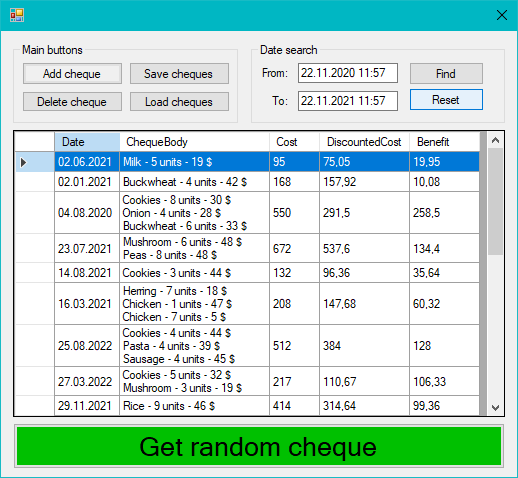


Рисунок 15 – Сброс условий поиска

**1.5.4 Тестовый случай «Save cheques»**

Для сохранения данных в таблице необходимо нажать на кнопку «Save». Откроется системный диалог сохранения файла, где пользователь выбирает директорию и указывает имя файла (рисунок 16).

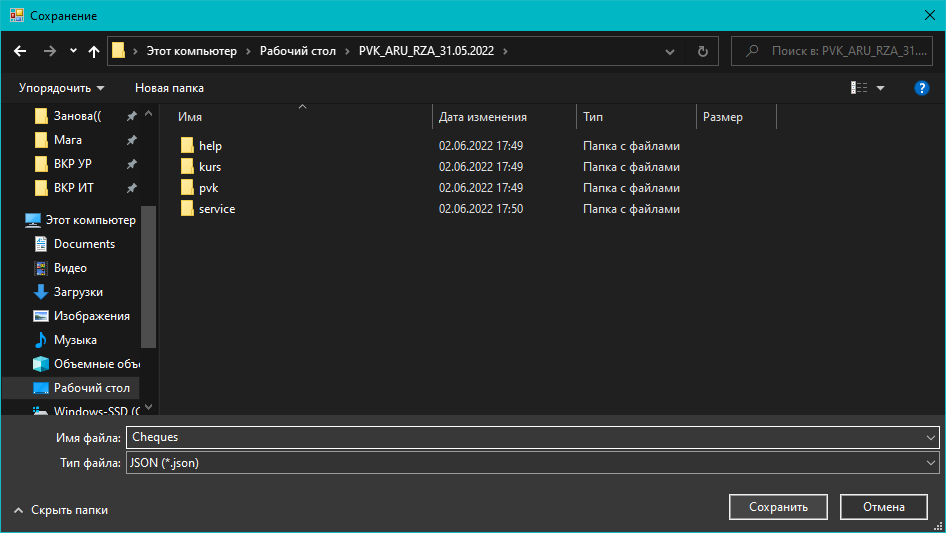


Рисунок 16 – Сохранение файла

**1.5.5 Тестовый случай «Load cheques»**

Для загрузки данных в таблицу необходимо нажать на соответствующую кнопку «Load cheques».

Далее откроется системный диалог загрузки файла (рисунок 17).

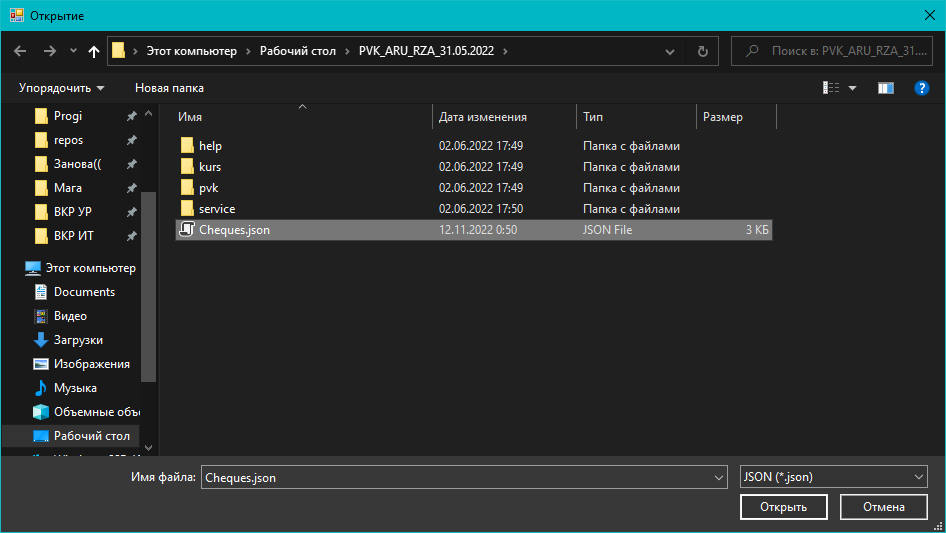


Рисунок 17 – Выбор файла для загрузки

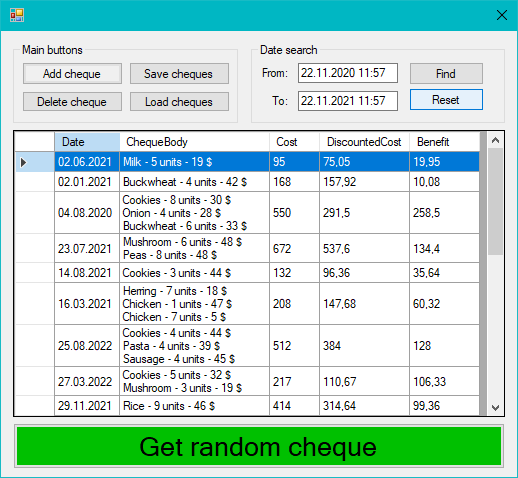


Рисунок 18 – Результат загрузки данных

В случае, если схема файла не соответствует установленному формату, появится соответствующее сообщение (рисунок 19).

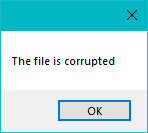


Рисунок 19 – Загрузка повреждённого файла

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Калентьев, А. А. Новые технологии в программировании : учебное пособие / А. А. Калентьев, Д. В. Гарайс, А. Е. Гориянов. – Томск : Эль Контент, 2014. – 176 с. – ISBN 978-5-4332-0185-9.

**ПРИЛОЖЕНИЕ А**

**Техническое задание на создание автоматизированной системы**

Программа для просмотра списка работников

Разработчик: студент гр. О-5КМ01 Арбчаков М.Е.

Заказчик: Канд. техн. наук, доцент каф. КСУП ТУСУР Калентьев А. А.

Томск 2022

**1 Общие сведения**

**1.1 Полное наименование системы и её условное обозначение**

Полное наименование: «Программа для просмотра чеков».

Условное обозначение: Система.

**1.2 Наименование предприятий разработчика и заказчика системы**

Заказчик: Канд. техн. наук, доцент каф. КСУП ТУСУР Калентьев А. А.

Разработчик: Студент гр. О-5КМ01 НИ ТПУ Арбачаков М.Е.

**1.3 Плановые сроки начала и окончания работы по созданию системы**

Начало работ: 1 апреля 2022 г.

Окончание работ: 1 июня 2022 г.

**2 Назначение и цели создания системы**

**2.1 Назначение системы**

Система предназначена для просмотра чеков с учетом скидок и добавления новых.

**2.2 Цели создания системы**

Система создается в целях автоматизации расчета скидок покупок и хранения чеков.

**3 Характеристика объектов автоматизации**

Раньше кассирам необходимо было вручную проводить расчет скидки на товар и запоминать купленный посетителем товар. Этот монотонный труд мог приводить к ошибкам, которые приводят к экономическим потерям. С целью ликвидации данных ситуаций разрабатывается данная система.

**4 Требования к системе**

Таблица 4.1 – Префиксы мнемонических идентификаторов требований и их расшифровка

|  |  |
| --- | --- |
| Префикс | Тип требования |
| A | Архитектурное требование |
| С | Требование к программной или аппаратной совместимости |
| D | Требование к структуре данных |
| F | Функциональное требование |
| U | Требование к пользовательскому интерфейсу |

**4.1 Требования к архитектуре**

**А01.** Система должна быть реализована в виде настольного приложения.

**4.2 Требования к структуре данных**

**D01.** Данные о чеках должны храниться в JSON-файле с расширением \*.json.

**D01.01.** Формат JSON-файла должен соответствовать следующей схеме:

[

{

"Date": "2022-02-17T00:00:00",

"ChequeBody": "Tomato - 3 units - 3 $\nChicken - 7 units - 7 $",

"Cost": 58.0,

"DiscountedCost": 35.96,

"Benefit": 22.04

}

]

**4.3 Функциональные требования**

**F01.** В системе должен быть реализован список чеков.

**F01.01.** Каждый элемент должен иметь следующие обязательные параметры:

* Дата;
* Тело чека;
* Стоимость покупки без учета скидки;
* Стоимость покупки с учетом скидки;
* Выгода.

**F01.02.** Тело чека состоит из информации о продукте:

* Название продукта;
* Количество продуктов;
* Цена продукта;

**F02.** В системе должна присутствовать функция добавления элементов в список.

**F03.** В системе должна присутствовать функция удаления элементов из списка.

**F04.** В системе должна присутствовать функция поиска элементов по диапазону дат.

**F05.** В системе должна присутствовать функция сохранения списка элементов в файл (**D01**).

**F06.** В системе должна присутствовать функция загрузки списка элементов из файла (**D01**).

**4.4 Требования к пользовательскому интерфейсу**

**U01.** Система должна иметь графический интерфейс пользователя.

**U02.** Данные должны быть представлены в табличном виде.

**U03.** В системе должна быть реализована система обработки ошибок.

**4.5 Требования к программному обеспечению**

**C01.** Система должна работать на операционной системе Windows 10. Работоспособность на других выпусках и версиях не гарантируется.

**C02.** На рабочей станции должен быть установлен .NET Framework версии 4.8.

**C03.** Процессор: процессор с частотой 1 гигагерц (ГГц) или выше

**C04.** ОЗУ: 1 ГБ для 32-разрядных систем или 2 ГБ для 64-разрядных систем

**C05.**Место на жестком диске: 16 ГБ для 32-разрядных систем или 20 ГБ для 64-разрядных систем