Направените промени са върху проект №12 Shop-Storage-System. На базата на изработената система са направени следните две функционални промени:

# 1.Добавяне на известия при намалено количество , отнасящо се до всеки вид стока.

За реализирането на новата функционалност са добавени следните методи:

* transactionsToolStripMenuItem във formAdminDashboard е менюто, отговорно за транзакциите в системата. При кликане върху “Transactions” се генерира събитие. Отваря се формата Transactions. Там може да бъде направен избор от падащото меню между „Sales“ и “Purchases”. Този избор предизвиква събитие, а методът, който отговаря на това събитие е cmbTransactionType\_SelectedIndexChanged във формата за транзакции. Типът транзакция се приема като текст. Вика се DisplayTransactionByType метод, който визуализира данните за вида транзакция в таблица. Тези данни се извличат от БД на системата.
* При кликане върху бутона „Show All“ възниква събитие във формата за транзакциите (formTransactions). Методът btnAll\_Click регистрира събитието и прави връзка с БД. Импортират се в DataGridView данните за всички транзакции от съответната категория заедно с информацията за продукта/продуктите, включени в съответната транзакция.
* При кликане върху бутона „Clear“ възниква събитие във формата за транзакциите (formTransactions). Методът btnClear\_Click регистрира събитието и прави връзка с БД, където се премахват от БД данните за всички транзакции от съответната категория.
* На база транзакциите (брой и вид на транзакцията) системата извлича данни от БД, предназначени за потребителя, импортира ги в DataGridView, което съдържа цялата информация за продуктите в намалено количество. Съдържанието на лейбъла над DataGridView-то се мени в зависимост от това дали има продукти в намалено количество, или няма продукти в намалено количество. Методът ShowLowQuantity извлича продуктите в намалено количество от базата данни и ги визуализира в DataGridView-то. Ако редовете на DataGridView-то са 0 на брой (т.е. няма продукти в намалено количество), лейбълът ще има съдържание „No products in low quantity!“ и DataGridView-то ще остане незапълнено, иначе, данните за продуктите в БД ще се визуализират в DataGridView и лейбълът ще има съдържание „Products in low quantity“.
* Методът purchaseToolStripMenuItem\_Click в кода на потребителската форма е отговорен за събитието, което настъпва след кликане върху “Purchase” – отваря се форма за покупко-продажба Purchase and Sales. При затваряне на формата за покупко-продажба се генерира събитие, методът Purchase\_FormClosed изпълнява това събитие и вика метода ShowLowQuantity, който извлича данните за продуктите в намалено количество от БД и ги визуализира в DataGridView. Същата процедура важи и при кликане върху „Sales“. Методът salesToolStripMenuItem\_Click в кода на потребителската форма е отговорен за събитието, което настъпва след кликане върху “Sales” – отваря се форма за покупко-продажба Purchase and Sales. При затваряне на формата за покупко-продажба се генерира събитие, методът Sales\_FormClosed изпълнява това събитие и вика метода ShowLowQuantity, който извлича данните за продуктите в намалено количество от БД и ги визуализира в DataGridView.
* В кода на формата за покупко-продажба formPurchaseSales if-else if конструкцията отразява дали транзакцията е покупка или продажба. Ако транзакцията е покупка, следва, че количеството на продукта/продуктите се увеличава. Това е от съществено значение и се прави връзка с БД, където по ID на продукта/продуктите и по брой закупен/и продукт/и се увеличава количеството на последните чрез метода IncreaseProduct, имплементиран в ProductData. Ако транзакцията е продажба, обаче, следва, че количеството на дадения продукт/продукти намалява. Дейността на програмата по отношение на случая „покупка“ е аналогична – методът DecreaseProduct, имплементиран в ProductData съответно по ID на продукта/продуктите и по продадено количество от тях намалява количеството им в БД.

За по-лесна ориентация относно количество на продуктите в Инвентара:

* Методът dgvProducts\_CellFormatting, имплементиран във formInventory, е отговорен за събитието, настъпило след кликане върху бутона “Show All” във формата за инвентара. Методът оцветява в червен сигнализиращ цвят редовете в DataGridView-то на тези продукти, чието количество е по-малко от 20 броя. Иначе, ако количеството на същите е над 19 броя, няма опасения за недостиг на продукти и редовете на DataGridView-то се оцветяват в бяло.

Във формата за покупко-продажба formPurchaseSales е разписана if-else конструкция, която проверява дали количеството на дадена стока не е отрицателно число, тъй като базата данни не приема отрицателни числа. Ако е така, се изпраща съобщение за невалидно количество „Invalid quantity!“, ако не, се изчислява крайната цена в зависимост от количеството на дадената стока.

Количеството на продуктите е от съществено значение за коректната и точна работа на системата, затова, няма как клиентът да добави отрицателно число като количество на продукт.

2.Добавяне на форма за справка за извършени поръчки в определен от потребителя времеви интервал с информация за бройки, сума ,средна цена за количество , средна стойност на данък, средна стойност на отстъпка.

За добавянето на зададената функционалност в администраторския изглед е добавена нова форма за показване на справки.

Във първоначалната администраторска форма е добавен нов ToolStripMenuItem с името Reference:

* При натискането му се изпълнява методът referenceToolStripMenuItem\_Click , който визуализира прозорец за справки.

Формата за справки е с името formOrderReference. В нея потребителя може да проследи в табличен вид:

* Transaction Id
* Type
* Dealer/Customer
* Description
* Total price
* Date
* Tax
* Discount
* Paid Amount
* Added by ID
* Added by Name

Справките могат да се направят в два варианта :

* За целия период на работа.
* За специфичен период след избор на конкретни дати.

След натискане на бутон за визуализация на таблицата съществуват три варианта за проследяването на информацията , които могат да се комбинират:

* Проследяване по тип – All, Purchase или Sales.
* Проследяване , от какъв профил е направена транзакцията.
* Проследяване , какъв е потребителят Dealer/Customer.

Визуализира се таблица , в която е показана информацията , както и текстови кутии с данни за :

* Обща стойност на направените поръчки.
* Средна стойност на всяка поръчка.
* Средна стойност на данъка.
* Средна стойност на отстъпката.
* Общ брой на артикулите.

Към статистиката са добавени две диаграми , чийто обхват включва:

* Диаграма показваща , какво е отношението между Purchase и Sales.
* Диаграма показваща , в какъв ценови диапазон са направени поръчките.

2.1. Съдържание на добавената форма.

В добавения прозорец се съдържат следните полета :

* mainDT от тип DataTable – използвано за извличане на данни от базата данни към таблицата за визуализация
* dv от тип DataView – използвано за филтриране на данните в таблицата

Създадени са и полета , които се използват при визуализиране на статистическите диаграми:

* numOfPurchases
* numOfSales
* numOfOrders0to50
* numOfOrders50to200
* numOfOrders200to500
* numOfOrders500to1000
* numOfOrdersMoreThan1000

2.1.Процеси при взаимодействие

1.Първоначално зареждане на формата:

* Възниква събитие за зареждане , което се изпълнява от метода formOrderReference\_Load.
* При визуализиране на формата първоначално диалоговия прозорец е на цел екран
* radioButtonAll е отбелязан.
* Показани са полетата за отбелязване на период , като и двете сочат текуща дата.
* ComboBoxes са с отбелязване по подразбиране „All“.

2. Избор на радио-бутон „All“

* Използва се методът radioButtonAll\_CheckedChanged.
* При отбелязване на радио-бутон radioButtonAll , полетата за времеви диапазон са неактивни.
* При промяна на избора те се активират.

3.Избор на времеви диапазон

* При избор за промяна на времеви диапазон от първото поле се активира и второто , като първоначално му е зададена днешна дата.

3.Натискане на бутона „Show“

* При натискане на бутон „Show“ се изпълнява метода btnShow\_Click.
* В него се прави проверка , кой от радио-бутон е избран от потребителя.
* Ако е избран радио-бутон различен от този по подразбиране се визуализират данни за определения времеви диапазон, в противен случай се показва цялата налична информация по направените транзакции.
* Всички ComboBoxes са активни.
* Визуализира се таблица с заявената информация.

4.Филтриране на данните в таблицата.

* Използва се методът FilterData.
* Потребителят може да избере по-какъв начин желае да сортира информацията спрямо направения избор от comboboxes.
* Информацията се филтрира ,изчисляват се Total и се съставят диаграми за статистическата извадка.

5.Съставяне на таблица DataGridView.

* Използва се методът RenameGrid().
* На всяка колона е заявено име , което ориентира потребителя какво вижда на екрана.

6.Представяне на статистика от извлечените данни.

* Използва се методът CalculateTotals.
* В колоната Description се показват както името на стоката , така и нейното количество.
* От колоната Total в таблицата се изчислява общата цена , която се записва в текстовата кутия Price.
* От същата колона се изчислява и средната цена за всеки продукт , като стойността се получава , при разделянето и на бройката на всички стоки от колоната Description.
* Аналогично е и при изчисляването на средна стойност на данък и отстъпка.
* В последната текстова кутия от Totals е изчислен общия брой на всички стоки.

За съставяне на статистически диаграми важат следните зависимости:

Диаграма за цена на направените поръчки е разделена по следния начин:

* Ако цената е до 50п.е. се сумират колко такива поръчки са направени.
* Ако цената е над 50п.е. и до 200п.е.
* Ако цената е над 200п.е. и до 500п.е.
* Ако цената е над 500п.е. и до 1000п.е.
* Ако цената е над 1000п.е.

Диаграма на отношението Sales/Purchase:

* Изчисляват се по отделно колко Sales и колко Purchase .
* Визуализира се диаграма на отношението.

За визуализацията на диаграмите отговарят методите :

* DrawTypesChart
* DrawPriceChart