# Dairy Delights



**Session 2023 - 2027**

**Submitted by:**

Abu Tayyab 2023-CS-54

**Supervised by:**

Prof. Maida Shahid

**Course:**

CSC-103 Object Oriented Programming

Department of Computer Science

**University of Engineering and Technology****Lahore Pakistan**

CSC-102 Programming Fundamentals Dairy Delights

Table of Contents

[Dairy Delights 1](#_Toc164807748)

[1. Introduction: 3](#_Toc164807749)

[Design 3](#_Toc164807750)

[OOP Pillars 3](#_Toc164807751)

[2. User Roles and Functionalities: 4](#_Toc164807752)

[i) Admin: 4](#_Toc164807753)

[ii) Worker 4](#_Toc164807754)

[iii) Delivery Boy: 4](#_Toc164807755)

[iv) Customer: 5](#_Toc164807756)

[3. Wire Frames: 7](#_Toc164807757)

[4.Data Structures 9](#_Toc164807758)

[5.Functions prototypes 10](#_Toc164807759)

[6.Flow Chart 14](#_Toc164807760)

[7.Complete Code of Business Application 15](#_Toc164807761)

[8.Weakness in the Business Application 77](#_Toc164807762)

[9.Future Directions 77](#_Toc164807763)

Table of Wire frames

Figure 1Header ................................................................................................................................ 5

Figure 2 Welcome Page .................................................................................................................. 5

Figure 3Sign In page ....................................................................................................................... 6

Figure 5 Sign Up ............................................................................................................................. 6

Figure 4 Admin Menu ..................................................................................................................... 6

Figure 7 Employee Menu ................................................................................................................ 7

Figure 8 Customer Menu ................................................................................................................ 7

Figure 9 Invoice .............................................................................................................................. 7

Figure 11 Flow Chart .................................................................................................................... 13

# 1. Introduction:

Dairy Delights Business Application is designed to manage operations for a dairy products business. This application provides functionalities for four types of users: Admin, Employees, Delivery Boy and Customers. The application offers a user friendly interface allowing efficient management of tasks related to inventory, sales, and customer interactions.

# Design

The whole project consists of 3 sub-projects:

1. **Project Library:** It contains all the classes of the entities in the Business Layer and a Data Layer to manage all entities’ data. The DataBase folder in the Data Layer handles all data using Database connection and SQL queries. Likewise, The File Handling folder handles only the User data. The Data Layer Interfaces folder contains all the Interfaces of the Data Layer Classes. The use of C# language is in full effect in the back-end. This Library makes the base of the Application.
2. **Window Forms Project:** It contains the Front-End of the Application. User-Friendly Window Forms are linked with the Project Library for smooth running of the Application.
3. **Console Application:** It contains a Console App in which the CRUD of the User entity has been performed.

# OOP Pillars

1. **Encapsulation:** The Class Attributes have been made private and accessed through Getters & Setters. The Class Behaviors are public. The Parent Class Attributes are protected.
2. **Association:** The Customer Class has a Ticket Object to show Aggregation between Customer and Ticket classes so that when a customer buys a ticket, it is saved as an object in the customer class. The Ticket Class has an Object of Match class to show which match the ticket is of.

# 2. User Roles and Functionalities:

Dairy Delights supports four user roles: Admin, Employees, Delivery Boy and Customers. Each role has distinct functionalities tailored to their specific needs.

# i) Admin:

1. Add new User to the system.
2. Remove User from the system.
3. View Orders
4. Add More Products
5. Remove Product
6. Update Products
7. Check Review about Products from Customers
8. See Complains/ Suggestions from Worker.
9. Update Details About Worker.
10. Change Password

# ii) Worker

1. Add Products
2. Remove Products
3. Update Products
4. Check Reviews about Products
5. View Pending Orders
6. Write Complains / Suggestions to Admin
7. Update Details of Customer
8. View his details
9. Change Password

# iii) Delivery Boy:

1. View Pending Orders
2. View Order Details
3. Deliver Orders
4. View Personal Details
5. Change Password

# iv) Customer:

1. View Products
2. View Details of Products
3. View Reviews of Products
4. Give Reviews
5. View Own Cart
6. Add Items in Cart
7. Edit Cart
8. Buy Cart
9. View Orders.

(10) Change Password

|  |  |  |
| --- | --- | --- |
| **As a** | **I want to perform** | **So that I can** |
| Admin | Add new employee. | Add a new employee to the system. |
| Remove employee. | Remove employee from the system. |
| View Orders | See All The orders delivered or not |
| Add Product | Add Product in the store |
| Update Product | Update details quantity etc |
| Remove Products | Remove Products from Inventory |
| Check Review | Check Reviews Of Products |
| See complains | See suggestion and problems of Worker |
| Update Users | Update details of Users |
| Change password. | Can change password of any username without even knowing old passwords. |
| Worker | Update Products | Update details of product |
| Add products | Add new products for sale in inventory. |
| Remove products | Remove products form the inventory |
| View Orders | View Pending Orders |
| Check Reviews About Products | Check Reviews form Customers. |
| Remove discount. | Remove discount form the products. |
| Update Customer | Change details of Customer |
| Writes complain to Admin. | Writes the issues he is facing to the admin so that issues can be resolved. |
| Change password. | He can change password but cannot change password of employee. |
| Delivery Boy | View Orders | View Order from the Customers |
| Products Details | See products Details in Order |
| Deliver Orders | Deliver orders to the Customer |
| View His Own Details | See his details Bike number etc |
| Change Password | Change his own password |
| Customer | View products. | See products available for sale. |
| View Reviews | See review of other Customers |
| View Details | View the description and other details about products |
| Give Feedback | Give review about products |
| View Cart | View His Items |
| Edit Cart | Change the cart add or remove things |
| Buy | Buy the products |
| Change password | Change login password |
| Update Details | Update Address and phone number etc |

# 3. Wire Frames:

# 4.Data Structures

string username[100]; string username[100] ;

string role[100]; string complains\_FromEmployees[100];

string cart[100]; string products[100]; string products\_Description[100]; string Customer\_Review\_AboutProducts[100];

int products\_Price[100]; int cart\_Quantity[100]; int products\_Quantity[100]; int productcount; int usercount; int discount; int cart\_Index; int review\_Index; int complain\_Index;

float cart\_Price[100];

# 5.Functions prototypes

void Print\_header(); // for printing the name of app. void print\_Welcome(); // for printing Welcome. void printTankYouCapital(); // printing Thank You. void successful(); // to tell that your instruction is done successfully.

void Unsuccessful(); // to tell that there was some problem which maybe because invalid input. void pressAnyKey(); // for holding screen.

void invalid\_Input(); // for printing that your Input is wrong. void print\_Thanks(); // for printing thanks.

string login\_page(); // it will return the option form login page .

string admin\_menu(); // it will display functions of admin and return which user wants to do. string input\_Name(); // it will take the name of product and return it. string input\_Username(); // it will take username of person and return it. string input\_Password(); // it will take password form user and return ir.

string input\_Role(); // for taking the role which may be customer or employee. string input\_NewPassword(); // to take new password so that it can be replaced with old one.

string input\_NameToViewComplain(); // for asking user to input name so that complain can be traced.

string employee\_menu(); // it will display employee options and return what user wants to do. string input\_Description(); // for writing description for the products.

string input\_OldPassword(); // for inputing old password to aithorize the password change. string input\_NameForReview(); // for user to give review about product.

string input\_UsernameFor\_Complain(); // for inputing username so that complain have name form which it has come. string input\_Complain(); // forwriting the complain.

string customer\_menu(); // fordisplaying the customer options and returning what he wants to do. string get\_ProductName(); // for taking product name from user.

string get\_Opinion\_ForDeletion(); // for displaying thatcustomer wants to delete complete cart or just some items. string input\_NameForDescription(); // forinput product name for description. string input\_NameFor\_Review(); // for input product name for writing review string input\_Review(); // for writing the review. string payment\_option(); // for asking user what will be the payment method.

// it will take things name and pasword and return what is this its role means customer or employee. string sign\_In(string user, string pass, string username[], string password[], string roleArray[]);

// it will take a username and return the role of that username.

string user\_Behind\_Username(string user, string username[], string role[], int count);

// it will take username , password, role and then verifies all the things and then enter user in data base.

bool sign\_Up(string user, string pass, string role, string username[], string password[],

string roleArray[], int count);

// it will add new employee to the data base. bool add\_Employee(string user, string pass, string username[], string password[],

string roleArray[], int count);

// it will remove employee form data base.

bool remove\_Employee(string user, string username[], string password[], string

roleArray[], int count);

// it will allow admin to change anyone password withour khnowing their password. bool change\_Password\_ForAdmin(string user, string new\_password, string username[], string password[], int count);

// it will print all the products that are available right now. bool print\_Products(string products[], int products\_Quantity[], int products\_Price[], int &productcount);

// it is for that new productscan be added to the store.

bool add\_NewProducts(string name, int quantity, int price, string products[], int

products\_Quantity[], int products\_Price[], int &productcount);

// this function will allow user to buy more stock for his shop.

bool buy\_MoreStock(string name, int quantity, string products[], int products\_Quantity[], int products\_Price[], int &productcount);

// so that user entered is valid for exanple it cannot be admin. bool validUser(string user);

// so that password must be higher or equal to 3 letters. bool validPassword(string pass);

// to validate the role of user. bool validRole(string role);

// for removings products from the data base.

bool remove\_Product(string name, string products[], int products\_Quantity[], int

products\_Price[], int &productcount);

// for changing the pass word it will need old password in order to change it.

bool change\_Password(string user, string old\_Password, string new\_password, string

username[], string password[], int count);

// to see the complain form employee.

bool see\_ComplainFromEmployee(string name, string username[], string

complains\_FromEmployees[], int usercount);

// so that description can be given about the products. bool give\_DiscriptionAboutProducts(string name, string discription, string products[],

string product\_Description[], int &productcount);

// to validate the username so that is cannot be admin or have spelling mistakes. bool valid\_Username(string user, string username[], string role[], int count);

// for user to give review about product whichh he liked or may be ot liked.

bool review\_OfProduct(string name, string products[], string customer\_Review\_AboutProduct[], int productcount);

// for employee to give what problems he is facing, bool give\_Complains(string user, string complain, string username[], string

complains\_FromEmployees[], int usercount);

// for the customer that it can only change the password of user not any employee to validate this thing this function is used. bool validRolecustomer(string role);

// for the user to add product to cart and then pay only when heis done shopping.

bool added\_InCart(string name, int quantity, string products[], int products\_Quantity[],

string cart[], int cart\_Quantity[], int &cart\_index, int product\_index);

// for empying the whole cart.

bool Delete\_CompleteCart(string cart[], int cart\_Quantity[], int &cart\_Index);

// for printig what is in the cart so far

bool print\_Items\_InCart(string cart[], int cart\_Quantity[], int cart\_Index);

// for deleting some productsfrom the cart.

bool Delete\_ProductFromCart(string name, string cart[], int cart\_Quantity[], int &cart\_Index);

// so that username must be present in data base or it will be invalid.

bool valid\_Username\_Customer(string user, string username[], string role[], int count); // view details about a specific product.

bool View\_DescriptionOfProduct(string name, string products[], string products\_Description[], int productcount);

// for user to give review about a product. bool give\_ReviewAboutProducts(string name, string review, string products[], string Customer\_Review\_AboutProducts[], int productcount);

// to check only card or cash is input byuser nothing else. bool Validate\_Payment\_Option(string payment\_method);

// at the end to creating the invoie/bill of shopping. bool print\_Invoice(float result, int discount, string payment\_Method, float final\_Price);

// will update the existing price of product bool update\_PriceOfProduct(string name, int price, string products[], int

products\_Price[], int productcount);

int get\_percentage\_of\_Discount(); // for taking percentage of discount.

int remove\_Discount(int &discount); // for removing the discount. int input\_Productquantity(); // for taking the quantity of product. int input\_ProductPrice(); // for entering the price of product. int available\_Discount(int discount); // to see how much discount is currently availabe.

// it will check if product is already preset and user bought more then increase the quantity which was in cart.

int already\_PresentThenUpdate\_Quantity(string name, int quality, string products[], int products\_Quantity[], string cart[], int cart\_Quantity[], int productcount, int cart\_Index);

int Get\_QuantityForCart(); // to ask user how much he wants to buy.

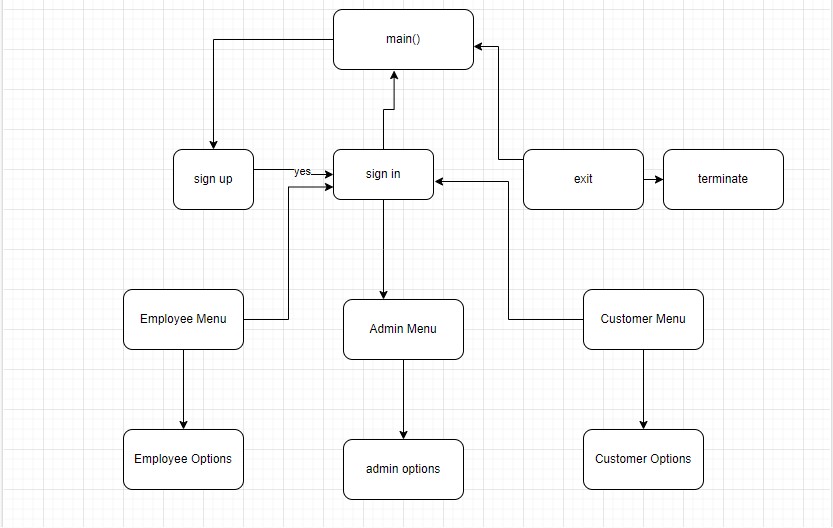
// for checkingout means the products in cart are ready to bought.

float checkoutCart(string cart[], int cart\_Quantity[], string products[], int products\_Quantity[], int product\_price[], float cart\_Price[], int &cart\_Index, int productcount);

float Total\_Amount(float result, int discount, string payment\_Method); // for

calculating the whole price after discount and tax.

# 6.Flow Chart



***Figure 9 Flow Chart***

# 7.Complete Code of Business Application

#include <iostream>

#include <conio.h>

#include <windows.h> #include <string>

using namespace std;

void Print\_header(); // for printing the name of app. void print\_Welcome(); // for printing Welcome. void printTankYouCapital(); // printing Thank You. void successful(); // to tell that your instruction is done successfully.

void Unsuccessful(); // to tell that there was some problem which maybe because invalid input.

void pressAnyKey(); // for holding screen.

void invalid\_Input(); // for printing that your Input is wrong. void print\_Thanks(); // for printing thanks.

string login\_page(); // it will return the option form login page .

string admin\_menu(); // it will display functions of admin and return which user wants to do.

string input\_Name(); // it will take the name of product and return it. string input\_Username(); // it will take username of person and return it. string input\_Password(); // it will take password form user and return ir. string input\_Role(); // for taking the role which may be customer or employee. string input\_NewPassword(); // to take new password so that it can be replaced with old one.

string input\_NameToViewComplain(); // for asking user to input name so that complain can be traced.

string employee\_menu(); // it will display employee options and return what user wants to do. string input\_Description(); // for writing description for the products.

string input\_OldPassword(); // for inputing old password to aithorize the password change. string input\_NameForReview(); // for user to give review about product.

string input\_UsernameFor\_Complain(); // for inputing username so that complain have name form which it has come. string input\_Complain(); // forwriting the complain.

string customer\_menu(); // fordisplaying the customer options and returning what he wants to do. string get\_ProductName(); // for taking product name from user.

string get\_Opinion\_ForDeletion(); // for displaying thatcustomer wants to delete complete cart or just some items. string input\_NameForDescription(); // forinput product name for description. string input\_NameFor\_Review(); // for input product name for writing review string input\_Review(); // for writing the review. string payment\_option(); // for asking user what will be the payment method.

// it will take things name and pasword and return what is this its role means customer or employee. string sign\_In(string user, string pass, string username[], string password[], string roleArray[]);

// it will take a username and return the role of that username.

string user\_Behind\_Username(string user, string username[], string role[], int count);

// it will take username , password, role and then verifies all the things and then enter user in data base.

bool sign\_Up(string user, string pass, string role, string username[], string password[],

string roleArray[], int count);

// it will add new employee to the data base.

bool add\_Employee(string user, string pass, string username[], string password[],

string roleArray[], int count);

// it will remove employee form data base.

bool remove\_Employee(string user, string username[], string password[], string

roleArray[], int count);

// it will allow admin to change anyone password withour khnowing their password. bool change\_Password\_ForAdmin(string user, string new\_password, string username[], string password[], int count);

// it will print all the products that are available right now. bool print\_Products(string products[], int products\_Quantity[], int products\_Price[], int &productcount);

// it is for that new productscan be added to the store.

bool add\_NewProducts(string name, int quantity, int price, string products[], int

products\_Quantity[], int products\_Price[], int &productcount);

// this function will allow user to buy more stock for his shop.

bool buy\_MoreStock(string name, int quantity, string products[], int products\_Quantity[], int products\_Price[], int &productcount);

// so that user entered is valid for exanple it cannot be admin. bool validUser(string user);

// so that password must be higher or equal to 3 letters.

bool validPassword(string pass);

// to validate the role of user. bool validRole(string role);

// for removings products from the data base. bool remove\_Product(string name, string products[], int products\_Quantity[], int

products\_Price[], int &productcount);

// for changing the pass word it will need old password in order to change it.

bool change\_Password(string user, string old\_Password, string new\_password, string

username[], string password[], int count);

// to see the complain form employee.

bool see\_ComplainFromEmployee(string name, string username[], string

complains\_FromEmployees[], int usercount);

// so that description can be given about the products. bool give\_DiscriptionAboutProducts(string name, string discription, string products[],

string product\_Description[], int &productcount);

// to validate the username so that is cannot be admin or have spelling mistakes. bool valid\_Username(string user, string username[], string role[], int count);

// for user to give review about product whichh he liked or may be ot liked.

bool review\_OfProduct(string name, string products[], string customer\_Review\_AboutProduct[], int productcount);

// for employee to give what problems he is facing, bool give\_Complains(string user, string complain, string username[], string

complains\_FromEmployees[], int usercount);

// for the customer that it can only change the password of user not any employee to validate this thing this function is used. bool validRolecustomer(string role);

// for the user to add product to cart and then pay only when heis done shopping.

bool added\_InCart(string name, int quantity, string products[], int products\_Quantity[],

string cart[], int cart\_Quantity[], int &cart\_index, int product\_index);

// for empying the whole cart.

bool Delete\_CompleteCart(string cart[], int cart\_Quantity[], int &cart\_Index);

// for printig what is in the cart so far bool print\_Items\_InCart(string cart[], int cart\_Quantity[], int cart\_Index);

// for deleting some productsfrom the cart.

bool Delete\_ProductFromCart(string name, string cart[], int cart\_Quantity[], int &cart\_Index);

// so that username must be present in data base or it will be invalid.

bool valid\_Username\_Customer(string user, string username[], string role[], int count);

// view details about a specific product.

bool View\_DescriptionOfProduct(string name, string products[], string products\_Description[], int productcount);

// for user to give review about a product. bool give\_ReviewAboutProducts(string name, string review, string products[], string Customer\_Review\_AboutProducts[], int productcount);

// to check only card or cash is input byuser nothing else. bool Validate\_Payment\_Option(string payment\_method);

// at the end to creating the invoie/bill of shopping. bool print\_Invoice(float result, int discount, string payment\_Method, float final\_Price);

// will update the existing price of product bool update\_PriceOfProduct(string name, int price, string products[], int

products\_Price[], int productcount);

int get\_percentage\_of\_Discount(); // for taking percentage of discount.

int remove\_Discount(int &discount); // for removing the discount. int input\_Productquantity(); // for taking the quantity of product. int input\_ProductPrice(); // for entering the price of product. int available\_Discount(int discount); // to see how much discount is currently availabe. // it will check if product is already preset and user bought more then increase the quantity which was in cart.

int already\_PresentThenUpdate\_Quantity(string name, int quality, string products[], int products\_Quantity[], string cart[], int cart\_Quantity[], int productcount, int cart\_Index);

int Get\_QuantityForCart(); // to ask user how much he wants to buy.

// for checkingout means the products in cart are ready to bought.

float checkoutCart(string cart[], int cart\_Quantity[], string products[], int products\_Quantity[], int product\_price[], float cart\_Price[], int &cart\_Index, int productcount);

float Total\_Amount(float result, int discount, string payment\_Method); // for

calculating the whole price after discount and tax.

main()

{

string username[100] = {"tayyab", "ali", "ahmad"};

// it is an array of usernames.

string password[100] = {"1234", "1234", "1234"};

// it is parrallel to username and it will store password.

string role[100] = {"admin", "employee", "customer"};

// it is parrallel to username and it will store roles of users.

string complains\_FromEmployees[100];

// array for storing complain form employees.

string cart[100];

// cart which will have all the products user wants to buy.

string products[100] = {"milk", "yougurt", "chedder", "mozerella"}; // for storing the names of products.

string products\_Description[100] = {"Milk is very good for bones", "Sweet Yougurt which is gluten free", "Type of Cheese which is best for Sandwiches", "Type of Cheese which is best for Pizza"}; // array of description.

string Customer\_Review\_AboutProducts[100];

// array for storing reviews form user.

int products\_Price[100] = {200, 300, 1500, 2000}; // this array will store price of products.

int cart\_Quantity[100]; // for quantity of products which are in cart right now.

int products\_Quantity[100] = {80, 20, 8, 10}; // how much stock is awailabe of products. int productcount = 4; // how much products are available.

int usercount = 3; // how much users have ben entere so far.

int discount = 0; // right now discount available int cart\_Index = 0; // how many items in cart.

int review\_Index = 0; // how many reviews available.

int complain\_Index = 0; // how many complains available.

float cart\_Price[100]; // total price of products in cart.

while (true) // it is infinite loop it will keep repeating untill user wants to exit.

{ system("cls"); // for clearing the screen.

Print\_header(); // for printing Header.

print\_Welcome(); // for printing welcome.

string login\_value;

login\_value = login\_page(); // it will tell which option is selected by user.

if (login\_value == "1") // if will execute only when 1 is selected by user.

{

string user, pass, person\_role; // variable daclaration

system("cls");

Print\_header(); // agian header will print

user = input\_Username(); // gather info pass = input\_Password();

person\_role = sign\_In(user, pass, username, password, role); // pass to function

which will return a value which will be admin or employee or customer.

if (person\_role == "admin") // it will true only when role id admin.

{ while (true) // again infinite loop.

{

system("cls");

string admin\_option; // variable daclaration

admin\_option = admin\_menu();

if (admin\_option == "1") // it will execute when user wants to do work on

option 1

{

bool check; // variable daclaration bool check\_employee;

bool validate;

string user\_admin, pass\_admin, role\_admin; system("cls");

Print\_header(); // agian header will print

user\_admin = input\_Username(); validate = validUser(user\_admin); // to check the user is valid. if (validate)

{

pass\_admin = input\_Password(); validate = validPassword(pass\_admin); // then check password

validity.

if (validate)

{

role\_admin = input\_Role(); // then check role validity.

validate = validRole(role\_admin);

if (validate)

{

check = sign\_Up(user\_admin, pass\_admin, role\_admin, username, password, role, usercount); // if itis true then user have been entered in data base.

if (check) // if true the add one to numbers of users in data base.

{

usercount++; pressAnyKey();

} else {

pressAnyKey();

} } else {

Unsuccessful(); // it means that input was not right and action cannoy be done.

continue;

} } else

{

pressAnyKey(); // agian input mistake.

continue;

}

}

else // agian action cannot be done because of invalid input.

{

pressAnyKey(); continue;

}

}

else if (admin\_option == "2") // in this section a user can be removed.

{

system("cls");

Print\_header(); // agian header will print bool check; string remove\_employee\_user;

remove\_employee\_user = input\_Username();

check = remove\_Employee(remove\_employee\_user, username,

password, role, usercount);

if (check)

{ successful(); usercount--; getch();

} else { invalid\_Input();

getch();

}

}

else if (admin\_option == "3") // this set the discount.

{

system("cls");

Print\_header(); // agian header will print bool check; discount = get\_percentage\_of\_Discount();

if (discount >= 100)

{

discount = 0; getch();

} else {

successful();

} continue;

}

else if (admin\_option == "4") // this will remove the discount.

{

system("cls");

Print\_header(); // agian header will print int remove; remove = remove\_Discount(discount);

if (remove > discount)

{

getch();

} else

{

discount = discount - remove;

successful();

} continue;

}

else if (admin\_option == "5") // this will print the available products.

{

system("cls");

Print\_header(); // agian header will print bool check; string name; int price;

name = input\_Name(); price = input\_ProductPrice(); if (price == 65535)

{

pressAnyKey();

continue;

}

if (price != 65535)

{

check = update\_PriceOfProduct(name, price,products,products\_Price,productcount);

if (check)

{

successful();

} else {

pressAnyKey();

} continue;

}

}

else if (admin\_option == "6") // for adding new products.

{

system("cls");

Print\_header(); // agian header will print bool check; string name; int quantity; int price;

name = input\_Name(); quantity = input\_Productquantity(); if (quantity == 65535)

{

pressAnyKey(); continue;

}

price = input\_ProductPrice(); if (price == 65535)

{

pressAnyKey(); continue;

}

if (price != 65535)

{

check = add\_NewProducts(name, quantity, price, products,

products\_Quantity, products\_Price, productcount);

if (check)

{

successful();

} else {

pressAnyKey();

} continue;

}

}

else if (admin\_option == "7") // for buying new stock.

{

system("cls");

Print\_header(); // agian header will print bool check; string name; int quantity;

name = input\_Name(); quantity = input\_Productquantity();

if(quantity != 65535)

{

check = buy\_MoreStock(name, quantity, products,

products\_Quantity, products\_Price, productcount); // for buying more stock; if (check)

{

successful();

} else {

pressAnyKey();

}

continue;

} else

{

pressAnyKey(); continue;

}

}

else if (admin\_option == "8") // this section will remove products.

{

system("cls");

Print\_header(); // agian header will print bool check;

string name;

name = input\_Name();

check = remove\_Product(name, products, products\_Quantity,

products\_Price, productcount);

if (check)

{

successful();

} else

{ pressAnyKey();

} continue;

}

else if (admin\_option == "9") // to view complain from employee.

{

system("cls");

Print\_header(); // agian header will print bool check;

string name;

name = input\_NameToViewComplain();

check = see\_ComplainFromEmployee(name, username,

complains\_FromEmployees, usercount);

if (check)

{

pressAnyKey();

} else

{

pressAnyKey();

} continue;

}

else if (admin\_option == "10")

{

system("cls");

Print\_header(); // agian header will print bool check;

string name; name = input\_NameForReview();

check = review\_OfProduct(name, products, Customer\_Review\_AboutProducts, productcount);

if (check)

{

pressAnyKey();

} else

{

pressAnyKey();

} continue;

}

else if (admin\_option == "11") // to change the password

{

system("cls");

Print\_header(); // agian header will print bool check;

bool valid;

string user;

string new\_password;

user = input\_Username(); new\_password = input\_NewPassword();

valid = validPassword(new\_password); if (valid)

{

check = change\_Password\_ForAdmin(user, new\_password,

username, password, usercount);

if (check)

{

successful();

} else {

pressAnyKey();

} continue;

} else

{

pressAnyKey(); continue;

}

}

else if (admin\_option == "12") // to exit the loop

{ break;

} else {

invalid\_Input(); continue;

}

}

}

else if (person\_role == "employee") // it will execute only when role employee

is return

{

while (true) // infinite loop for asking the employee what he wants to do untill

he wants to exit the loop

{

string employee\_Option;

employee\_Option = employee\_menu(); // it will print the employee menu and return which option is returned

if (employee\_Option == "1") // execute only when option 1 is selected

{

system("cls");

Print\_header(); // agian header will print bool check; check = print\_Products(products, products\_Quantity, products\_Price,

productcount);

pressAnyKey(); continue;

}

else if (employee\_Option == "2")

{

system("cls");

Print\_header(); // agian header will print bool check; string name; int quantity; int price;

name = input\_Name(); quantity = input\_Productquantity(); if (quantity == 65535)

{

pressAnyKey(); continue;

}

price = input\_ProductPrice(); if (price == 65535)

{

pressAnyKey(); continue;

}

if (price != 65535)

{

check = add\_NewProducts(name, quantity, price, products,

products\_Quantity, products\_Price, productcount);

if (check)

{

successful();

} else {

pressAnyKey();

} continue;

}

}

else if (employee\_Option == "3")

{

system("cls");

Print\_header(); // agian header will print bool check;

string name;

name = input\_Name();

check = remove\_Product(name, products, products\_Quantity,

products\_Price, productcount);

if (check)

{

successful();

} else

{

pressAnyKey();

}

continue;

}

else if (employee\_Option == "4")

{

system("cls");

Print\_header(); // agian header will print bool check; string name; int price;

name = input\_Name(); price = input\_ProductPrice(); if (price == 65535)

{

pressAnyKey(); continue;

}

if (price != 65535)

{

check = update\_PriceOfProduct(name, price,products,products\_Price,productcount);

if (check)

{

successful();

} else {

pressAnyKey();

} continue;

}

}

else if (employee\_Option == "5")

{

system("cls");

Print\_header(); // agian header will print

int result;

result = available\_Discount(discount); getch(); continue;

}

else if (employee\_Option == "6")

{

system("cls");

Print\_header(); // agian header will print bool check; string description;

string name;

name = input\_Name();

description = input\_Description();

check = give\_DiscriptionAboutProducts(name, description, products,

products\_Description, productcount);

if (check)

{

successful();

} else

{

pressAnyKey();

} continue;

}

else if (employee\_Option == "7")

{

system("cls");

Print\_header(); // agian header will print int remove; remove = remove\_Discount(discount);

if (remove > discount)

{

pressAnyKey();

}

else

{

discount = discount - remove;

successful();

} continue;

}

else if (employee\_Option == "8")

{

system("cls");

Print\_header(); // agian header will print bool check;

string name; name = input\_NameForReview();

check = review\_OfProduct(name, products, Customer\_Review\_AboutProducts, productcount);

if (check)

{

pressAnyKey();

} else

{

pressAnyKey();

} continue;

}

else if (employee\_Option == "9")

{

system("cls");

Print\_header(); // agian header will print bool check; string name;

string complain;

name = input\_UsernameFor\_Complain(); complain = input\_Complain();

check = give\_Complains(name, complain, username,

complains\_FromEmployees, usercount);

if (check)

{

successful();

} else

{

pressAnyKey();

} continue;

}

else if (employee\_Option == "10")

{

system("cls");

Print\_header(); // agian header will print bool check; bool valid\_password; bool valid\_username;

bool valid; string user; string person; string old\_Password;

string new\_password;

user = input\_Username(); valid\_username = valid\_Username(user, username, role, usercount);

if (valid\_username)

{

person = user\_Behind\_Username(user, username, role, usercount);

if (person == "employee")

{

old\_Password = input\_OldPassword();

new\_password = input\_NewPassword(); valid\_password = validPassword(new\_password); if (valid\_password)

{

check = change\_Password(user, old\_Password, new\_password,

username, password, usercount);

if (check)

{

successful();

} else {

pressAnyKey();

}

continue;

} else

{

Unsuccessful(); continue;

}

}

else if (person == "customer")

{

new\_password = input\_NewPassword();

valid = validPassword(new\_password); if (valid)

{

check = change\_Password\_ForAdmin(user, new\_password,

username, password, usercount); if (check)

{

successful();

} else

{

Unsuccessful();

} continue; } else

{

Unsuccessful();

}

continue;

} } else

{ pressAnyKey(); continue;

}

}

else if (employee\_Option == "11")

{ break; } else { invalid\_Input(); continue;

}

}

}

else if (person\_role == "customer")

{ while (true) {

system("cls");

Print\_header(); // agian header will print string customer\_Option;

customer\_Option = customer\_menu();

if (customer\_Option == "1")

{

system("cls");

Print\_header(); // agian header will print

bool check; check = print\_Products(products, products\_Quantity, products\_Price,

productcount);

getch(); continue;

}

else if (customer\_Option == "2")

{

system("cls");

Print\_header(); // agian header will print bool check; bool added\_toCart; int already\_Present; string name; int quantity;

check = print\_Products(products, products\_Quantity, products\_Price,

productcount);

name = get\_ProductName(); quantity = Get\_QuantityForCart();

if (quantity != 65535)

{

already\_Present = already\_PresentThenUpdate\_Quantity(name,

quantity, products, products\_Quantity, cart, cart\_Quantity, productcount, cart\_Index); if (already\_Present == 1)

{

added\_toCart = added\_InCart(name, quantity, products,

products\_Quantity, cart, cart\_Quantity, cart\_Index, productcount); if (added\_toCart)

{

successful();

} else {

pressAnyKey();

}

continue;

} } else

{

pressAnyKey();

} continue;

}

else if (customer\_Option == "3")

{

system("cls");

Print\_header(); // agian header will print string option;

option = get\_Opinion\_ForDeletion();

if (option == "1")

{

bool check;

check = Delete\_CompleteCart(cart, cart\_Quantity, cart\_Index); if (check)

{

successful();

} else

{

Unsuccessful();

} continue;

}

else if (option == "2")

{

bool check; bool remove; string name; check = print\_Items\_InCart(cart, cart\_Quantity, cart\_Index); if(check)

{

name = input\_Name();

remove = Delete\_ProductFromCart(name, cart, cart\_Quantity,

cart\_Index);

if (remove)

{

successful();

} else {

pressAnyKey();

}

continue;

} else {

pressAnyKey();

} } continue;

}

else if (customer\_Option == "4")

{

system("cls");

Print\_header(); // agian header will print bool check;

check = print\_Items\_InCart(cart, cart\_Quantity, cart\_Index); if (check)

{

pressAnyKey();

} else

{

pressAnyKey();

} continue; }

else if (customer\_Option == "5")

{

system("cls");

Print\_header(); // agian header will print bool check;

string name;

name = input\_NameForDescription();

check = View\_DescriptionOfProduct(name, products,

products\_Description, productcount);

if (check)

{

successful();

} else

{

pressAnyKey();

} continue;

}

else if (customer\_Option == "6")

{

system("cls");

Print\_header(); // agian header will print bool check; string name;

string review;

name = input\_NameFor\_Review(); review = input\_Review();

check = give\_ReviewAboutProducts(name, review, products, Customer\_Review\_AboutProducts, productcount);

if (check)

{

successful(); } else {

pressAnyKey();

} continue;

}

else if (customer\_Option == "7")

{

system("cls");

Print\_header(); // agian header will print int result;

result = available\_Discount(discount); pressAnyKey(); continue;

}

else if (customer\_Option == "8")

{

system("cls");

Print\_header(); // agian header will print bool check; bool valid\_password; bool valid\_username; string user; string old\_Password;

string new\_password;

user = input\_Username(); valid\_username = valid\_Username(user, username, role, usercount);

if (valid\_username)

{

old\_Password = input\_OldPassword(); valid\_username = valid\_Username\_Customer(user, username, role,

usercount);

if (valid\_username)

{

new\_password = input\_NewPassword(); valid\_password = validPassword(new\_password); if (valid\_password)

{

check = change\_Password(user, old\_Password, new\_password,

username, password, usercount);

if (check)

{

successful();

} else {

pressAnyKey();

}

continue;

} else {

pressAnyKey(); continue;

} } else

{

Unsuccessful(); continue;

} } else

{

pressAnyKey(); continue;

}

}

else if (customer\_Option == "9")

{

system("cls");

Print\_header(); // agian header will print float result;

bool check;

string payment\_Method;

payment\_Method = payment\_option();

result = checkoutCart(cart, cart\_Quantity, products, products\_Quantity,

products\_Price, cart\_Price, cart\_Index, productcount);

check = Validate\_Payment\_Option(payment\_Method); if (check)

{

float final\_Price; bool invoice; final\_Price = Total\_Amount(result, discount, payment\_Method);

invoice = print\_Invoice(result, discount, payment\_Method,

final\_Price);

if (invoice)

{

bool check;

check = Delete\_CompleteCart(cart, cart\_Quantity, cart\_Index); printTankYouCapital();

getch(); continue;

} else

{

Unsuccessful(); continue;

} } else {

invalid\_Input(); continue;

}

}

else if (customer\_Option == "10")

{ break; } else { invalid\_Input(); continue;

}

} } else { getch(); continue;

}

}

else if (login\_value == "2")

{ while (true) { system("cls"); Print\_header(); bool check; bool check\_employee;

bool validate; string user\_admin, pass\_admin, role\_admin;

user\_admin = input\_Username();

validate = validUser(user\_admin);

if (validate)

{

pass\_admin = input\_Password();

validate = validPassword(pass\_admin); if (validate)

{

role\_admin = input\_Role(); validate = validRolecustomer(role\_admin);

if (validate)

{

check = sign\_Up(user\_admin, pass\_admin, role\_admin, username,

password, role, usercount);

if (check)

{

usercount++; pressAnyKey();

break;

} } else

{

pressAnyKey();

break;

}

} else

{

pressAnyKey();

break;

}

} else

{

pressAnyKey();

break;

}

}

}

else if (login\_value == "3")

{

print\_Thanks();

break;

}

else {

invalid\_Input(); continue;

}

}

}

void Print\_header()

{

cout << "\033[1;36m"; // Set text color to cyan cout << " \_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_

\_\_\_\_\_\_\_ \_ \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ " << endl; cout << "\033[1;32m"; // Set text color to green

cout << " ( \_\_ \\ ( \_\_\_ )\\\_\_ \_\_/( \_\_\_\_ )|\\ /| ( \_\_ \\ ( \_\_\_\_

\\( \\ \\\_\_ \_\_/( \_\_\_\_ \\|\\ /|\\\_\_ \_\_/( \_\_\_\_ \\" << endl; cout << "\033[1;36m"; // Set text color to cyan

cout << " | ( \\ )| ( ) | ) ( | ( )|( \\ / ) | ( \\ )| ( \\/| ( ) (

| ( \\/| ) ( | ) ( | ( \\/" << endl;

cout << "\033[1;35m"; // Set text color to magenta

cout << " | | ) || (\_\_\_) | | | | (\_\_\_\_)| \\ (\_) / | | ) || (\_\_ | | |

| | | | (\_\_\_) | | | | (\_\_\_\_\_" << endl; cout << "\033[1;33m"; // Set text color to yellow

cout << " | | | || \_\_\_ | | | | \_\_) \\ / | | | || \_\_) | | | | |

| \_\_\_\_ | \_\_\_ | | | (\_\_\_\_\_ )" << endl; cout << "\033[1;31m"; // Set text color to red

cout << " | | ) || ( ) | | | | (\\ ( ) ( | | ) || ( | | | | | |

\\\_ )| ( ) | | | ) |" << endl;

cout << "\033[1;33m"; // Set text color to yellow

cout << " | (\_\_/ )| ) ( |\_\_\_) (\_\_\_| ) \\ \\\_\_ | | | (\_\_/ )| (\_\_\_\_/\\|

(\_\_\_\_/\\\_\_\_) (\_\_\_| (\_\_\_) || ) ( | | | /\\\_\_\_\_) |" << endl; cout << "\033[1;35m"; // Set text color to magenta cout << " (\_\_\_\_\_\_/ |/ \\|\\\_\_\_\_\_\_\_/|/ \\\_\_/ \\\_/ (\_\_\_\_\_\_/ (\_\_\_\_\_\_\_/(\_\_\_\_\_\_\_/\\\_\_\_\_\_\_\_/(\_\_\_\_\_\_\_)|/ \\| )\_( \\\_\_\_\_\_\_\_)" << endl; cout << endl;

cout << "\033[1;36m"; // Set text color to cyan

cout << " THE DAIRY SHOP"

<< endl; cout << "\033[0m"; // Reset text color to default cout << endl;

}

void print\_Welcome()

{ cout << " \033[1;35m"; // Set text color to magenta cout << " \_\_ \_\_ .\_\_ " << endl; cout << " \033[1;33m"; // Set text color to yellow cout << " / \\ / \\ \_\_\_\_ | | \_\_\_\_ \_\_\_\_ \_\_\_\_\_ \_\_\_\_ " <<

endl;

cout << " \033[1;36m"; // Set text color to cyan cout << " \\ \\/\\/ // \_\_ \\| | \_/ \_\_\_\\/ \_ \\ / \\\_/ \_\_ \\ " << endl; cout << " \033[1;31m"; // Set text color to red cout << " \\ /\\ \_\_\_/| |\_\\ \\\_\_( <\_> ) Y Y \\ \_\_\_/ " << endl; cout << " \033[1;32m"; // Set text color to green cout << " \\\_\_/\\ / \\\_\_\_ >\_\_\_\_/\\\_\_\_ >\_\_\_\_/|\_\_|\_| /\\\_\_\_ >"

<< endl;

cout << " \033[1;34m"; // Set text color to blue cout << " \\/ \\/ \\/ \\/ \\/ " << endl; cout << " \033[0m"; // Reset text color to default cout << endl;

}

void printTankYouCapital()

{

cout << endl;

cout << " \033[1;31m"; // Set text color to red for the title cout << " \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_ \_ \_\_\_\_\_\_\_ " << endl;

Sleep(50);

cout << " \\\_\_ \_\_/|\\ /|( \_\_\_ )( ( /|| \\ /\\ |\\ /|( \_\_\_ )|\\ /|"; Sleep(50); cout << endl; Sleep(50);

cout << " ) ( | ) ( || ( ) || \\ ( || \\ / / ( \\ / )| ( ) || ) ( |";

Sleep(50); cout << endl; Sleep(50); cout << " | | | (\_\_\_) || (\_\_\_) || \\ | || (\_/ / \\ (\_) / | | | || | | |";

Sleep(50); cout << endl; Sleep(50);

cout << " | | | \_\_\_ || \_\_\_ || (\\ \\) || \_ ( \\ / | | | || | | |";

Sleep(50); cout << endl; Sleep(50);

cout << " | | | ( ) || ( ) || | \\ || ( \\ \\ ) ( | | | || | | |";

Sleep(50); cout << endl; Sleep(50);

cout << " | | | ) ( || ) ( || ) \\ || / \\ \\ | | | (\_\_\_) || (\_\_\_) |"; Sleep(50); cout << endl; Sleep(50);

cout << " )\_( |/ \\||/ \\||/ )\_)|\_/ \\/ \\\_/

(\_\_\_\_\_\_\_)(\_\_\_\_\_\_\_)"; Sleep(50); cout << endl; Sleep(50);

cout << " \033[0m"; // Reset text color to default }

void successful()

{

cout << "Task is completed Successfully" << endl; getch(); }

void Unsuccessful()

{

cout << "Task is not completed Successfully" << endl; getch(); }

void print\_Thanks()

{

cout << "Thank You :)" << endl;

} string login\_page()

{

string login\_value; cout<<"\033[;33m"; cout << "LOGIN PAGE :" << endl; cout << endl; cout << "1. Sign In " << endl; cout << "2. Sign Up " << endl; cout << "3. Exit " << endl; cout << endl; cout << "Enter Your Option here: ";

getline(cin>>ws, login\_value);

// cin >> login\_value;

return login\_value;

cout<<"\033[0m";

} string input\_Username()

{ string user;

cout << "Enter username: ";

// cin >> user;

getline(cin>>ws, user);

return user; } string input\_Password()

{ string pass; cout << "Enter password: "; // cin >> pass; getline(cin>>ws,pass);

return pass; }

string input\_Role()

{

string role\_admin;

cout << "Enter Role: "; // cin >> role\_admin;

getline(cin>>ws,role\_admin);

return role\_admin;

}

string input\_NewPassword()

{ string pass;

cout << "Enter New Password: ";

// cin >> pass;

getline(cin>>ws,pass);

return pass; }

bool validUser(string user)

{

bool containsSpace;

containsSpace = true;

for (int index = 0; index < user.length(); index++)

{

if (user[index] == ' ')

{

containsSpace = false; cout<<"User name cannot contain spaces"<<endl; break; // exit the loop if a space is found

}

}

return containsSpace;

}

bool validPassword(string pass)

{ bool pass\_admin;

if (pass.length() == 4)

{

pass\_admin = true;

} else

{

cout << "Password Must contain 4 letters" << endl; pass\_admin = false;

}

return pass\_admin;

}

bool validRolecustomer(string role)

{

bool Role;

Role = false; if (role == "customer")

{

Role = true;

}

if (!(Role))

{

cout << "You Can only Signup as \"customer\" if you want to sign up as employee

contact your admin" << endl;

}

return Role;

}

bool validRole(string role)

{

bool Role;

Role = true;

return Role;

} bool sign\_Up(string user, string pass, string role, string username[], string password[],

string roleArray[], int count)

{

bool check = false;

for (int index = 0; index < count; index++)

{

if (user == username[index])

{

check = true;

}

}

if (check)

{

cout << "Username already exists"; getch();

}

if(!(check))

{

if (role == "employee" || role == "customer")

{ check = false;

} else {

check = true; cout << "Enter a valid role (employee/customer)" << endl;

}

if (!check)

{

username[count] = user;

password[count] = pass; roleArray[count] = role;

count++; cout<<"Successfully Signed Up"<<endl;

}

}

return !check;

} string sign\_In(string user, string pass, string username[], string password[], string

roleArray[])

{ int index; string result;

bool check = true;

for (index = 0; index < 100; index++)

{

if (user == username[index] && pass == password[index])

{

cout << "Valid" << endl; cout << "Press any key to continue"; getch(); result = roleArray[index];

check = false;

} } if (check)

{

cout << "Invalid username or password" << endl; result = "invalid";

}

return result;

}

string admin\_menu()

{ system("cls");

system("color 09");

Print\_header();

string option; cout << "\033[1;33m";

cout << " Admin Menu" <<

endl;

cout << endl;

cout << "1. Add New Employee" << endl; cout << "2.Remove Employee" << endl; cout << "3.Set discount" << endl; cout << "4.Remove discount" << endl; cout << "5.Update Price" << endl; cout << "6.Add More Products" << endl; cout << "7.Buy More Stock" << endl; cout << "8.remove product" << endl; cout << "9.See Complains from Employees" << endl; cout << "10.Check review About Product From User" << endl; cout << "11.change password" << endl; cout << "12. Logout" << endl; cout << "Enter Your Option Here:"; getline(cin>>ws,option);

return option; cout << "\033[0m"; // Reset text color to default

} bool remove\_Employee(string user, string username[], string password[], string

roleArray[], int count)

{

int user\_index, pass\_index;

bool check = false;

for (int index = 0; index <= count; index++)

{

if (username[index] == user)

{

user\_index = index; check = true; break;

} } if (check)

{

for (int index = user\_index; index < count - 2; index++)

{

username[index] = username[index + 1]; password[index] = password[index + 1];

roleArray[index] = roleArray[index + 1];

}

}

return check;

}

int remove\_Discount(int &discount)

{

int discount\_percentage;

string ans;

cout << "Right Now Discount offered is :" << discount << " %" << endl; cout << "Enter how much discount You Want to Remove : " << endl; getline(cin>>ws,ans);

if (isdigit(ans[0]) && ans[0] != '0') // Check if the input is a valid positive integer

{

discount\_percentage = stoi(ans);

} else

{

cerr << "Error: Invalid input. Please enter a valid positive integer." << endl; discount\_percentage = 65535;

}

return discount\_percentage;

}

int get\_percentage\_of\_Discount()

{

int discount\_percentage;

string ans;

cout << "Enter how much flat discount You Want to Offer: " << endl; getline(cin>>ws,ans);

if (isdigit(ans[0]) && ans[0] != '0') // Check if the input is a valid positive integer

{

discount\_percentage = stoi(ans);

} else

{

cerr << "Error: Invalid input. Please enter a valid positive integer." << endl; discount\_percentage = 65535;

}

return discount\_percentage;

}

bool make\_Employee\_Array(string username[], string role[], string employee[])

{

bool check;

check = false; int employee\_index = 0;

for (int index = 0; index < 100; index++)

{

if (role[index] == "employee")

{

employee[employee\_index] = username[index]; check = true;

employee\_index++;

}

}

return check;

} bool print\_Products(string products[], int products\_Quantity[], int products\_Price[], int &productcount)

{

bool check;

cout << endl; cout << "Name:" << endl; for (int index = 0; index < productcount; index++)

{

cout << index + 1 << ". " << products[index]; cout << endl;

}

cout << endl;

cout << "Quantity: " << endl; for (int index = 0; index < productcount; index++)

{

cout << index + 1 << ". " << products\_Quantity[index] << " kg."; cout << endl;

}

cout << endl;

cout << "Price: " << endl; for (int index = 0; index < productcount; index++)

{

cout << index + 1 << ". " << products\_Price[index] << " Rs."; cout << endl;

}

check = true; return check;

}

string input\_Name()

{

string name;

cout << "Enter the Product Name: "; getline(cin>>ws,name);

return name;

}

int input\_Productquantity()

{ int quantity;

string ans;

cout << "Enter the Quantity of the Product: "; getline(cin>>ws,ans);

if (isdigit(ans[0]) && ans[0] != '0') // Check if the input is a valid positive integer

{

quantity = stoi(ans);

} else

{

cerr << "Error: Invalid input. Please enter a valid positive integer." << endl; quantity = 65535;

}

return quantity;

}

int input\_ProductPrice()

{ int price;

string ans;

cout << "Enter the Price of the Product: "; getline(cin>>ws,ans);

if (isdigit(ans[0]) && ans[0] != '0') // Check if the input is a valid positive integer

{

price = stoi(ans);

}

else

{

cerr << "Error: Invalid input. Please enter a valid positive integer." << endl; price = 65535;

}

return price;

} bool add\_NewProducts(string name, int quantity, int price, string products[], int

products\_Quantity[], int products\_Price[], int &productcount)

{

bool check;

check = true; for(int index = 0; index < productcount; index++)

{

if(name == products[index])

{

check = false; cout<<"Product already exist"<<endl; break;

} } if(check)

{

products[productcount] = name; products\_Quantity[productcount] = quantity; products\_Price[productcount] = price; productcount++;

}

return check;

}

bool buy\_MoreStock(string name, int quantity, string products[], int products\_Quantity[], int products\_Price[], int &productcount)

{ bool check; check = false;

for (int index = 0; index < productcount; index++)

{

if (name == products[index])

{

products\_Quantity[index] += quantity;

check = true; break;

} }

if(!(check))

{

cout<<"Product Not Fount"<<endl;

}

return check;

} bool remove\_Product(string name, string products[], int products\_Quantity[], int

products\_Price[], int &productcount)

{ bool check; int name\_index; check = false; for (int index = 0; index < productcount; index++)

{

if (name == products[index])

{

name\_index = index; check = true; break;

} } if(check)

{

for (int index = name\_index; index < productcount - 2; index++)

{

products[index] = products[index + 1];

products\_Quantity[index] = products\_Quantity[index + 1]; products\_Price[index] = products\_Price[index + 1];

}

productcount--;

}

if(!(check))

{

cout<<"Product Not Found"<<endl;

}

return check;

}

bool change\_Password\_ForAdmin(string user, string new\_password, string username[], string password[], int count)

{ bool check; int user\_index; check = false; for (int index = 0; index < count; index++)

{

if (user == username[index])

{

user\_index = index; check = true; break;

} } if(check)

{

password[user\_index] = new\_password;

}

if(!(check))

{

cout<<"Username Not Found"<<endl;

}

return check;

}

string employee\_menu()

{ system("cls"); Print\_header(); string option; cout << "\033[1;34m";

cout << " Employee Menu" <<

endl;

cout << endl; cout << "1.View products" << endl; cout << "2.Add products" << endl; cout << "3.Remove products" << endl; cout << "4.Update Price" << endl; cout << "5.View available discount" << endl; cout << "6.Write Description about the Product" << endl; cout << "7.Remove Discount" << endl; cout << "8.See Review About Product From User" << endl; cout << "9.Write complains for the Admin" << endl; cout << "10.Change Password" << endl; cout << "11.Logout" << endl;

cout << "Enter Your Option Here: ";

// cin >> option;

getline(cin>>ws,option);

return option; cout << "\033[0m"; // Reset text color to default

} int available\_Discount(int discount)

{

cout << "right Now Available Discount is :" << discount << " %" << endl; return 0; } bool give\_DiscriptionAboutProducts(string name, string discription, string products[],

string product\_Description[], int &productcount)

{ bool check; int name\_index; check = false;

for (int index = 0; index < productcount; index++)

{ if (name == products[index])

{

name\_index = index; check = true; break;

} } if(check)

{

product\_Description[name\_index] = discription;

} if(!(check))

{

cout<<"Product Not Found"<<endl;

}

return check;

}

string input\_Description()

{ string result; cout << "Enter The Description of the product: "; getline(cin>>ws,result);

return result;

}

bool valid\_Username(string user, string username[], string role[], int count)

{ bool check; int user\_index; string result; check = false; for (int index = 0; index < count; index++)

{

if (user == username[index])

{

user\_index = index; check = true;

break;

} } if(check)

{

result = role[user\_index];

if (result == "admin")

{

cout<<"Cannot Change password of admin"; check = false; return check;

} }

if(!(check))

{

cout<<"Invalid username"<<endl;

}

return check;

}

string customer\_menu()

{ system("cls"); Print\_header(); string option; cout << "\033[1;32m";

cout << " Customer Menu" <<

endl;

cout << endl;

cout << "1.View products" << endl; cout << "2.Add to cart" << endl; cout << "3.Edit Cart" << endl; cout << "4.View Cart" << endl;

cout << "5.See information about a product" << endl; cout << "6.Give Review about a Product" << endl; cout << "7.View discounts" << endl; cout << "8.Change password" << endl;

cout << "9.Checkout" << endl; cout << "10. Logout" << endl;

cout << "Enter Your Option Here:";

getline(cin>>ws,option);

return option; cout << "\033[0m"; // Reset text color to default

}

string get\_ProductName()

{

string result; cout << "Enter the Product Name you Want to buy: ";

getline(cin>>ws,result);

return result;

} bool added\_InCart(string name, int quantity, string products[], int products\_Quantity[],

string cart[], int cart\_Quantity[], int &cart\_index, int product\_index)

{ bool check; int name\_index; check = false; for (int index = 0; index < product\_index; index++)

{

if (name == products[index])

{

name\_index = index; check = true; break;

} } if(check)

{

if (products\_Quantity[name\_index] >= quantity)

{

if (check)

{

cart[cart\_index] = name;

cart\_Quantity[cart\_index] = quantity;

cart\_index++;

} } else {

check = false;

} } else

{

cout<<"Product Not Found"<<endl;

}

return check;

}

string get\_Opinion\_ForDeletion()

{ string option;

cout << "1.Delete The Complete Cart:" << endl; cout << "2.Delete Some Items" << endl; getline(cin>>ws,option);

return option;

}

bool Delete\_CompleteCart(string cart[], int cart\_Quantity[], int &cart\_Index)

{ bool check; check = false; for (int index = 0; index < cart\_Index; index++)

{

cart[index] = "nothing"; cart\_Quantity[index] = 0;

check = true;

}

cart\_Index = 0; return check;

}

int Get\_QuantityForCart()

{

string input;

int result;

cout << "Enter How much You want to Buy:";

getline(cin>>ws,input);

if (isdigit(input[0]) && input[0] != '0') // Check if the input is a valid positive integer

{

result = stoi(input);

} else

{

cout << "Error: Invalid input. Please enter a valid positive integer." << endl; result = 65535;

}

return result;

}

bool print\_Items\_InCart(string cart[], int cart\_Quantity[], int cart\_Index)

{ bool check; check = false; for (int index = 0; index < cart\_Index; index++)

{

cout << "Product Name: " << cart[index] << endl; cout << "Product Price: " << cart\_Quantity[index] << endl; cout << endl; check = true; if (cart[index] == "noting")

{

check = true; break;

} }

if (!(check))

{

cout << "Your Cart is Empty:" << endl;

check = false;

}

return check;

} bool Delete\_ProductFromCart(string name, string cart[], int cart\_Quantity[], int &cart\_Index)

{ bool check; int name\_index; check = false; for (int index = 0; index < cart\_Index; index++)

{

if (name == cart[index])

{

name\_index = index; check = true; break;

} } if(check)

{

for (int index = name\_index; index < cart\_Index - 1; index++)

{

cart[index] = cart[index + 1]; cart\_Quantity[index] = cart\_Quantity[index + 1];

} if (check)

{

cart\_Index--;

} } else

{

cout<<"product Not Fount"<<endl;

}

return check;

}

int already\_PresentThenUpdate\_Quantity(string name, int quality, string products[], int products\_Quantity[], string cart[], int cart\_Quantity[], int productcount, int cart\_Index)

{ bool check; int result;

int index\_Already\_PresentItem; int index\_products;

int total\_Quantity;

check = false;

result = 1;

for (int index = 0; index < cart\_Index; index++)

{

if (name == cart[index])

{

index\_Already\_PresentItem = index;

check = true; break;

} } if (check)

{

for (int index = 0; index < productcount; index++)

{

if (name == products[index])

{

index\_products = index;

break;

}

}

total\_Quantity = cart\_Quantity[index\_Already\_PresentItem] + quality;

if (total\_Quantity <= products\_Quantity[index\_products])

{ result = 0;

cart\_Quantity[index\_Already\_PresentItem] += quality;

} else {

result = 65535;

}

}

return result;

}

bool valid\_Username\_Customer(string user, string username[], string role[], int count)

{ bool check; int user\_index; string result; check = true; for (int index = 0; index < count; index++)

{

if (user == username[index])

{

user\_index = index;

break;

}

}

result = role[user\_index];

if (result == "admin")

{

cout<<"You Cannot change password of an Admin"<<endl;

check = false;

}

if (result == "employee")

{

cout<<"You Cannot change password of an employee"<<endl; check = false;

}

return check;

}

bool View\_DescriptionOfProduct(string name, string products[], string products\_Description[], int productcount)

{ bool check; check = false; int description\_index;

for (int index = 0; index < productcount; index++) {

if (name == products[index])

{

description\_index = index;

check = true; break;

} } if (check)

{

cout << "The Description of " << name << " is follows: " << endl; cout << products\_Description[description\_index] << endl;

} else

{

cout<<"Product Not Fount"<<endl;

}

return check;

}

string input\_NameForDescription()

{

string name;

cout << "Enter the Product Name about which you want to see description: "; getline(cin>>ws,name); return name;

}

string input\_NameFor\_Review()

{

string name;

cout << "Enter the Product Name about which you want to give Review: "; getline(cin>>ws,name); return name;

}

string input\_Review()

{ string result;

cout << "Enter The Review of the product: ";

getline(cin>>ws,result);

return result;

} bool give\_ReviewAboutProducts(string name, string review, string products[], string Customer\_Review\_AboutProducts[], int productcount)

{ bool check; int name\_index; check = false; for (int index = 0; index < productcount; index++)

{

if (name == products[index])

{

name\_index = index; check = true; break;

} } if(check)

{

Customer\_Review\_AboutProducts[name\_index] = review;

} else

{

cout<<"Product Not Found"<<endl;

}

return check;

} bool change\_Password(string user, string old\_Password, string new\_password, string

username[], string password[], int count)

{ bool check; int user\_index; check = false; for (int index = 0; index < count; index++)

{

if (user == username[index] && password[index] == old\_Password)

{

user\_index = index; check = true; break;

} } if(check)

{

password[user\_index] = new\_password;

}

if(!(check))

{

cout<<"Invalid Old Pasword"<<endl;

}

return check;

}

float checkoutCart(string cart[], int cart\_Quantity[], string products[], int products\_Quantity[], int product\_price[], float cart\_Price[], int &cart\_Index, int productcount)

{

float result = 0;

for (int i = 0; i < cart\_Index; i++)

{

for (int loop = 0; loop < productcount; loop++)

{

if (products[loop] == cart[i])

{

products\_Quantity[loop] -= cart\_Quantity[i]; cart\_Price[i] = cart\_Quantity[i] \* product\_price[loop]; result += cart\_Price[i];

}

}

}

return result;

}

string payment\_option()

{

string option;

cout << "How Would You Like to Pay (\"Card\" or \"Cash\"): "; getline(cin>>ws,option);

return option;

}

bool Validate\_Payment\_Option(string payment\_method)

{ bool check; check = false; if (payment\_method == "card" || payment\_method == "Card")

{

check = true;

}

else if (payment\_method == "cash" || payment\_method == "Cash")

{

check = true;

}

return check;

}

float Total\_Amount(float result, int discount, string payment\_Method)

{

float final\_Price;

if (payment\_Method == "card" || payment\_Method == "Card")

{

final\_Price = result - (result \* discount / 100); final\_Price = final\_Price + final\_Price \* 0.05;

}

else if (payment\_Method == "cash" || payment\_Method == "Cash")

{

final\_Price = result - (result \* discount / 100); final\_Price = final\_Price + final\_Price \* 0.18;

}

return final\_Price;

}

bool print\_Invoice(float result, int discount, string payment\_Method, float final\_Price) { bool check;

system("cls");

check = true; cout << endl; cout << endl; cout << endl; cout << endl; cout << endl; cout << " DAIRY - DELIGHTS"

<< endl;

cout << " Items Total Price: "

<< result << " $" << endl; cout << " Currently Discount Available

" << discount << " %" << endl;

cout << " Payment Method "

<< payment\_Method << endl;

cout << " Final Price "

<< final\_Price << " $" << endl; cout << endl;

}

string user\_Behind\_Username(string user, string username[], string role[], int count)

{ int user\_index; bool check;

string result;

check = false;

result = "invalid";

for (int index = 0; index < count; index++)

{

if (user == username[index])

{

user\_index = index; check = true; break;

} } if(check)

{

result = role[user\_index];

} else

{

cout<<"Invalid Username"<<endl;

}

return result;

}

string input\_OldPassword()

{ string pass; cout << "Enter Old password: "; getline(cin>>ws,pass);

return pass;

}

string input\_NameForReview()

{

string name;

cout << "Enter the Product Name about which you want to see Review: "; getline(cin>>ws,name);

return name;

}

bool review\_OfProduct(string name, string products[], string customer\_Review\_AboutProduct[], int productcount)

{ bool check; check = false; int review\_index; for (int index = 0; index < productcount; index++)

{

if (name == products[index])

{

review\_index = index;

check = true; break;

} } if (check)

{

cout << "The Review of " << name << " is follows: " << endl; cout << customer\_Review\_AboutProduct[review\_index] << endl;

}

if(!(check))

{

cout<<"Product Not Found"<<endl;

}

return check;

}

string input\_UsernameFor\_Complain()

{

string name;

cout << "Enter the Valid Username for submitting Complain: "; getline(cin>>ws,name);

return name;

}

string input\_Complain()

{

string result; cout << "Enter The issues You are facing: "; getline(cin>>ws,result);

return result;

} bool give\_Complains(string user, string complain, string username[], string

complains\_FromEmployees[], int usercount)

{ bool check; int name\_index; check = false; for (int index = 0; index < usercount; index++)

{

if (user == username[index])

{

name\_index = index; check = true; break;

} } if(check)

{

complains\_FromEmployees[name\_index] = complain;

}

if(!(check))

{

cout<<"Username not Found"<<endl;

}

return check;

}

string input\_NameToViewComplain()

{

string name;

cout << "Enter the username whose complain you want to review "; getline(cin>>ws,name);

return name;

}

bool see\_ComplainFromEmployee(string name, string username[], string

complains\_FromEmployees[], int usercount)

{ bool check; check = false; int complain\_index; for (int index = 0; index < usercount; index++)

{

if (name == username[index])

{

complain\_index = index;

check = true; break;

}

}

if(!(check))

{

cout<<"No complain Found Againts: "<<name<<endl; cout<<"Either Username is wrong or "<<name<<" is not an Employee"<<endl;

} if (check)

{

cout << "The Complain of " << name << " is follows: " << endl; cout << complains\_FromEmployees[complain\_index] << endl;

}

return check;

}

void pressAnyKey()

{

cout << endl; cout << "Press Any Key To Continue :)" << endl; getch(); }

void invalid\_Input()

{

cout << endl;

cout << "Invalid Input!" << endl; cout << "Press Any Key To Continue :)" << endl; getch();

}

bool update\_PriceOfProduct(string name, int price, string products[], int

products\_Price[], int productcount)

{

bool check;

check = false; int product\_index; for(int index = 0; index < productcount; index++)

{

if(name == products[index])

{

check = true;

product\_index = index;

break;

} } if(check)

{

products\_Price[product\_index] = price;

} if(!(check))

{

cout<<"Product not fount"<<endl;

}

return check;

}

# 8.Weakness in the Business Application

1.Using parallel arrays.

2. Complex Nesting of loops and ifs statements.

3.Readability is not very good.

4.Commenting is not very good.

# 9.Future Directions

1. I will make it much more beautiful 2. Code will be more easy to understand.

1. Use different data structures.
2. I will do a lot better commenting.