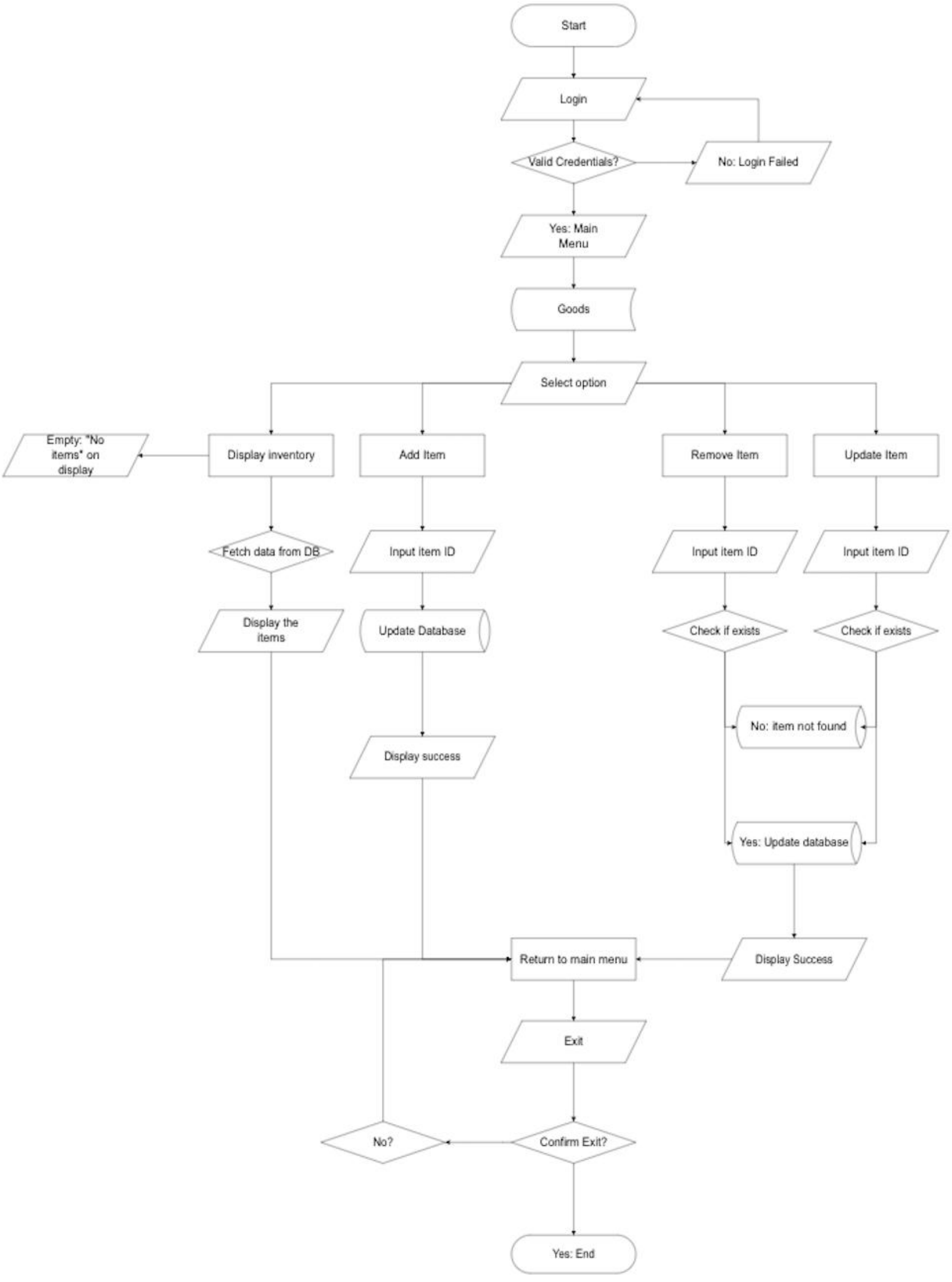
Start
Login
Valid Credentials?
No: Login Failed
Yes: Main Menu
Goods
Select option
Empty: "No items" on display
Display inventory
Add Item
Remove Item
Update Item
Fetch data from DB
Input item ID
Input item ID
Input item ID
Display the items
Update Database
Check if exists
Check if exists
No: item not found
Display success
Yes: Update database
Return to main menu
Display Success
Exit
No?
Confirm Exit?
Yes: End

```c
/* Cdeliv1gp<group 1>.pdf */
#include <stdio.h>
#include <string.h>

#define MAX_PRODUCTS 100

// Function prototypes
void login();
void mainMenu(char names[][30], int quantity[], float price[], int *count);
void addItem(char names[][30], int quantity[], float price[], int *count);
void removeItem(char names[][30], int quantity[], float price[], int *count);
void updateItem(char names[][30], int quantity[], float price[], int count);
void displayInventory(char names[][30], int quantity[], float price[], int count);

int main() {
    char names[MAX_PRODUCTS][30];
    int quantity[MAX_PRODUCTS];
    float price[MAX_PRODUCTS];
    int count = 0;

    login(); // Login before accessing the main menu
    mainMenu(names, quantity, price, &count);

    return 0;
}

void login() {
    char username[20], password[20];
    printf("Enter username: ");
    scanf("%s", username);
    printf("Enter password: ");
    scanf("%s", password);
    printf("Login successful!\n");
}

void mainMenu(char names[][30], int quantity[], float price[], int *count) {
    int option;

    do {
        printf("\nMain Menu:\n");
        printf("1. Add Item\n");
        printf("2. Remove Item\n");
        printf("3. Update Item\n");
        printf("4. Display Inventory\n");
```

```c
        printf("5. Exit\n");
        printf("Choose your option: ");
        scanf("%d", &option);

        switch (option) {
            case 1:
                addItem(names, quantity, price, count);
                break;
            case 2:
                removeItem(names, quantity, price, count);
                break;
            case 3:
                updateItem(names, quantity, price, *count);
                break;
            case 4:
                displayInventory(names, quantity, price, *count);
                break;
            case 5:
                printf("Exiting program.\n");
                break;
            default:
                printf("Invalid option! Please try again.\n");
        }
    } while (option != 5);
}

void addItem(char names[][30], int quantity[], float price[], int *count) {
    if (*count < MAX_PRODUCTS) {
        printf("Enter product name: ");
        scanf("%s", names[*count]);
        printf("Enter quantity: ");
        scanf("%d", &quantity[*count]);
        printf("Enter price: ");
        scanf("%f", &price[*count]);
        (*count)++;
        printf("Item added successfully!\n");
    } else {
        printf("Inventory full! Cannot add more items.\n");
    }
}
```

```c

void removeItem(char names[][30], int quantity[], float price[], int *count) {
    if (*count == 0) {
        printf("No items to remove.\n");
        return;
    }

    char nameToRemove[30];
    int found = -1;
    printf("Enter the name of the product to remove: ");
    scanf("%s", nameToRemove);

    for (int i = 0; i < *count; i++) {
        if (strcmp(names[i], nameToRemove) == 0) {
            found = i;
            break;
        }
    }

    if (found != -1) {
        for (int j = found; j < *count - 1; j++) {
            strcpy(names[j], names[j + 1]);
            quantity[j] = quantity[j + 1];
            price[j] = price[j + 1];
        }
        (*count)--;
        printf("Item removed successfully!\n");
    } else {
        printf("Item not found.\n");
    }
}

void updateItem(char names[][30], int quantity[], float price[], int count) {
    if (count == 0) {
        printf("No items to update.\n");
        return;
    }

    char nameToUpdate[30];
    int found = -1;
    printf("Enter the name of the product to update: ");
    scanf("%s", nameToUpdate);
```

```c
    char nameToUpdate[30];
    int found = -1;
    printf("Enter the name of the product to update: ");
    scanf("%s", nameToUpdate);

    for (int i = 0; i < count; i++) {
        if (strcmp(names[i], nameToUpdate) == 0) {
            found = i;
            break;
        }
    }

    if (found != -1) {
        printf("Enter new quantity: ");
        scanf("%d", &quantity[found]);
        printf("Enter new price: ");
        scanf("%f", &price[found]);
        printf("Item updated successfully!\n");
    } else {
        printf("Item not found.\n");
    }
}

void displayInventory(char names[][30], int quantity[], float price[], int count) {
    if (count == 0) {
        printf("No items in inventory.\n");
        return;
    }

    printf("\nInventory List:\n");
    printf("-----------------------------------------------------\n");
    printf("No.\tName\t\tQuantity\tPrice\n");
    for (int i = 0; i < count; i++) {
        printf("%d\t%s\t\t%d\t\t%.2f\n", i+1, names[i], quantity[i], price[i]);
    }
    printf("-----------------------------------------------------\n\n");
}

```

Sample Output for this code :
Enter username: administrator
Enter password: admin12345
Login successful!

Main Menu:
1. Add Item
2. Remove Item
3. Update Item
4. Display Inventory
5. Exit
Choose your option: 1

Enter product name: Apples
Enter quantity: 50
Enter price: 24
Item added successfully!

Main Menu:
1. Add Item
2. Remove Item
3. Update Item
4. Display Inventory
5. Exit
Choose your option: 1

Enter product name: Bananas
Enter quantity: 30
Enter price: 12
Item added successfully!

Main Menu:
1. Add Item
2. Remove Item
3. Update Item
4. Display Inventory
5. Exit
Choose your option: 4

Inventory List:

No. Name Quantity Price
1 Apples 50 24
2 Bananas 30 12

Main Menu:
1. Add Item
2. Remove Item
3. Update Item
4. Display Inventory
5. Exit
Choose your option: 3

Enter the name of the product to update: Bananas
Enter new quantity: 40
Enter new price: 20
Item updated successfully!

Main Menu:
1. Add Item
2. Remove Item
3. Update Item
4. Display Inventory
5. Exit
Choose your option: 4

Inventory List:

No.	Name	Quantity	Price
1	Apples	50	24
2	Bananas	40	20

Main Menu:
1. Add Item
2. Remove Item
3. Update Item
4. Display Inventory
5. Exit
Choose your option: 2

Enter the name of the product to remove: Apples
Item removed successfully!

Main Menu:
1. Add Item
2. Remove Item

3. Update Item
4. Display Inventory
5. Exit
Choose your option: 4

Inventory List:

No.	Name	Quantity	Price
1	Bananas	40	20

Main Menu:
1. Add Item
2. Remove Item
3. Update Item
4. Display Inventory
5. Exit
Choose your option: 5

Exiting program.

Inventory Management System Project (Week 4 Group 1)

This is an Inventory Management System (IMS) which is a simple C program designed to handle basic inventory tasks like adding, removing, updating, and displaying items. The program uses arrays to store item names, quantities, and prices, with functions managing the various operations.

Analysis of Implementation

Arrays:

Arrays are used to store names, quantities, and prices of products.

Loops: Used in the main menu to allow repeated user input. For loops are used to iterate through arrays to find, update, or display items, and the do-while loop manages the main menu operations.

Functions: Functions modularize the code, improving readability and reusability. They operate directly on the arrays using pointers where necessary, like modifying the count variable.

Strings: Strings are used to store and compare product names for easy identification. String handling is done using the strcmp function for comparisons and strcpy for assignments, crucial for identifying and updating products.

The role GenAI played in our code:

- Our GenAI tool, in this case, ChatGPT, suggested using #define to avoid overusing function names and instead assigned numerical values to functions, making the code more readable.
- getInput() was suggested instead of the repetitive use of printf and scanf throughout the code. So, getInput() is used in place of every printf followed by scanf.
- It made the item search, and therefore the remove and update functions, much faster. By using a single loop instead of several separate loops, it ensures more efficient logic.

The role GenAI played in our flowchart:

Based on our new code ChatGPT suggested to include the following to our flowchart:

1. Login Section:
   - Login → "Valid Credentials?"
   - Yes: Main Menu
   - No: "Login Failed" → Loop back to "Login"
2. Main Menu Section:
   - Select Option → "Valid option?"
   - No: "Invalid option!" → Return to "Main Menu"
3. Display Inventory Section:
   - Display Inventory → "Inventory Empty?"
   - Yes: "No items to display" → Return to "Main Menu"
   - No: Display items → Return to "Main Menu"
4. Add/Remove/Update Section:
   - Input Item ID → "Item Exists?"
   - No: "Item not found" → Return to "Main Menu"
   - Yes: "Update Database" or "Remove Item" → "Success" → Return to "Main Menu"
5. Exit Section:
   - Exit? → "Confirm Exit?"
   - Yes: End
   - No: Return to Main Menu

Fair Contribution Sheet

• Group Number/Name 1/Alireza Eftekhar

Enter average fair contribution scores here.

• Member name Mark

1. Fatima Zafar 10

2. Marvo Amini 10

3.

• Name and signature of each member:

Signature ARE ..... Date 23/09/24

Signature Marvo Amini ..... Date 23.09.24

Signature Fatima Zafar ..... Date 23/09/24

Fair Contribution Sheet

• Group Number/Name.....1/Fatima Zafar.....

Enter average fair contribution scores here.

• Member name Mark

1 Marvo Amini 10

2 Alireza Eftekhar 10

3.....

• Name and signature of each member:

Signature Date.23/09/2024

Signature .....Date.23.09.24.....

Signature .....Date.23/09/24.....

Fair Contribution Sheet

• Group Number/Name.....1/Marvo Amini.....

Enter average fair contribution scores here.

• Member name Mark

1.....Fatima Zafar.....10.....

2.....Alireza Eftekhar.....10.....

3.....

• Name and signature of each member:

Signature .....Marvo Amini.....Date.....23.09.24.....

Signature .....Fatima Zafar.....Date.....23/09/24.....

Signature .....ARE.....Date.....23/09/24.....