**Project Requirements and Objectives**

**Objectives**

The goal of this project is to develop a simple **console-based Inventory Management System** that allows users to efficiently manage product stock. The system should provide an intuitive interface for users to **add, update, view, and remove products** from the inventory.

**Functional Requirements (What the system should do)**

1. **Add New Products**
   * Users can add products with the following details:
     + Name
     + Price
     + Stock quantity
2. **Update Stock Levels**
   * Users can increase stock when restocking products.
   * Users can decrease stock when products are sold.
3. **View Products**
   * Users can list all available products, showing:
     + Product Name
     + Price
     + Stock Quantity
4. **Remove Products**
   * Users can delete a product from the inventory when it is no longer needed.
5. **Error Handling**
   * Prevent invalid input (e.g., negative stock values, empty product names).
   * Notify the user when attempting to update or remove a non-existing product.

**Non-Functional Requirements (How the system should perform)**

1. **Usability**
   * The system should be easy to use with clear prompts and instructions.
   * Provide meaningful error messages and confirmations.
2. **Performance**
   * The system should respond quickly to user input.
3. **Scalability**
   * The system should allow for an increasing number of products without performance degradation.
4. **Reliability**
   * Ensure data consistency when updating stock levels.
   * Prevent accidental data loss or incorrect product modifications.
5. **Maintainability**
   * The code should be well-structured, easy to read, and documented for future improvements.

Here's a high-level design outline for your **Inventory Management System** in **C#**, including a flowchart and task breakdown.

**Flowchart**

I'll generate a flowchart that represents the main interactions of the system.

**Process Flow:**

1. Start
2. Display Main Menu
3. User selects an action:
   * **Add Product** → Enter product details → Save to inventory
   * **Update Stock** → Choose product → Increase/Decrease stock
   * **View Products** → Display all products
   * **Remove Product** → Choose product → Delete from inventory
4. Return to the Main Menu or Exit.

Here's the flowchart for your **Inventory Management System in C#**. It visually represents the key processes and decisions. Let me know if you need any modifications!



**Task Breakdown & Code Components**

1. **Project Setup**
   * Create a new C# Console Application.
   * Set up a Product class to store product details.
2. **Main Menu Display**
   * Use a while loop to show the menu options.
   * Read user input and call the appropriate function.
3. **Adding a Product**
   * Prompt the user for **name, price, and stock quantity**.
   * Store the product in a **List** or a dictionary.
4. **Updating Stock**
   * Search for a product by name or ID.
   * Increase or decrease stock based on user input.
5. **Viewing Products**
   * Loop through the product list and display details.
6. **Removing a Product**
   * Search for a product by name or ID.
   * Remove it from the list.
7. **Error Handling**
   * Prevent invalid inputs (e.g., negative stock values, empty product names).
   * Handle cases where the product doesn’t exist.

**Code:**  
  
using System;

using System.Collections.Generic;

class Product

{

public string Name { get; set; }

public double Price { get; set; }

public int Stock { get; set; }

public Product(string name, double price, int stock)

{

Name = name;

Price = price;

Stock = stock;

}

}

class Program

{

static List<Product> inventory = new List<Product>();

static void Main()

{

while (true)

{

Console.WriteLine("\nInventory Management System");

Console.WriteLine("1. Add Product");

Console.WriteLine("2. Update Stock");

Console.WriteLine("3. View Products");

Console.WriteLine("4. Remove Product");

Console.WriteLine("5. Exit");

Console.Write("Choose an option: ");

string choice = Console.ReadLine();

switch (choice)

{

case "1": AddProduct(); break;

case "2": UpdateStock(); break;

case "3": ViewProducts(); break;

case "4": RemoveProduct(); break;

case "5": return;

default: Console.WriteLine("Invalid choice. Try again."); break;

}

}

}

static void AddProduct()

{

Console.Write("Enter product name: ");

string name = Console.ReadLine();

Console.Write("Enter product price: ");

double price = Convert.ToDouble(Console.ReadLine());

Console.Write("Enter stock quantity: ");

int stock = Convert.ToInt32(Console.ReadLine());

inventory.Add(new Product(name, price, stock));

Console.WriteLine("Product added successfully!");

}

static void UpdateStock()

{

Console.Write("Enter product name to update: ");

string name = Console.ReadLine();

Product product = inventory.Find(p => p.Name.ToLower() == name.ToLower());

if (product != null)

{

Console.Write("Enter new stock quantity: ");

product.Stock = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Stock updated successfully!");

}

else

{

Console.WriteLine("Product not found!");

}

}

static void ViewProducts()

{

Console.WriteLine("\nCurrent Inventory:");

foreach (var product in inventory)

{

Console.WriteLine($"Name: {product.Name}, Price: {product.Price:C}, Stock: {product.Stock}");

}

}

static void RemoveProduct()

{

Console.Write("Enter product name to remove: ");

string name = Console.ReadLine();

Product product = inventory.Find(p => p.Name.ToLower() == name.ToLower());

if (product != null)

{

inventory.Remove(product);

Console.WriteLine("Product removed successfully!");

}

else

{

Console.WriteLine("Product not found!");

}

}

}