**National University of Computer & Emerging Sciences**

**Karachi Campus**



**Stock Management System**

**Project Report**

**Programming Fundamentals**

**Section: BS – CS 1A**

**Group Members:**

**23k-0907 Zubair Ahmed**

**23k-0768 Sufyan Siddiqui**

**23k-0639 Muhammad Talha Saquib**

* INTRODUCTION (AIM OR MOTIVATION)

The aim of the project is to develop a comprehensive solution for efficient inventory management. To address the critical need for an efficient and user-friendly system to track, manage, and analyze inventory data.

Stock management software tracks, manages, and organizes stock levels. The purpose of stock management software is to maintain an optimal stock level, receive new items, and manage warehouse processes such as depositing and withdrawals.

Software for stock management automates what used to be a time-consuming, manual process of counting each item one by one and recording it on paper. Digitising this process not only makes it more accurate, but it also saves valuable time.

Core capabilities include stock optimisation, product identification and tracking, and reorder threshold setting and overstocking warnings.

* BACKGROUND (RESEARCH & PROJECT SELECTION)

To address the challenges faced in inventory management within markets and commercial hot zones such as inefficiencies and inaccuracies inherent in manual inventory tracking, leading to stockouts, overstocking, and lost sales opportunities. Focus on this area stemmed from the need for an effective solution to elevate inventory tracking, minimize stock discrepancies.

* PROJECT SPECIFICATION

The project's specifications include creating a platform that integrates with existing point-of-sale systems, providing real-time updates on stock levels (overstocking/understocking), and generating detailed reports for future references. This specification aims to address the industry's demand for a robust and scalable inventory management system.

* PROBLEM ANALYSIS

The problem analysis revealed the inefficiencies and inaccuracies inherent in manual inventory tracking, leading to stockouts, overstocking, and lost sales opportunities. Inefficient Inventory management: many companies still use outdated software to manage their inventory, which results in wasteful spending and time loss. Manual inventory tracking procedures involve paperwork across spreadsheets and outdated software, which leads to data redundancy.

* SOLUTION DESIGN (PROJECT DETAIL, FUNCTIONALITY,

AND FEATURES)

The solution design encompasses a user-friendly interface, updated data synchronization, and customizable alerts to ensure seamless inventory management. This design aims to provide a comprehensive solution that addresses the industry's pressing needs and enhances operational efficiency.

LIBRARIES:

The code includes several header files for standard input-output operations (windows.h) (stdio.h), string operations (string.h), character handling (ctype.h), (stdlib.h), and console input/output operations (conio.h).

GLOBAL VARIABLES:

net\_transactions store the total value of transactions.

heading\_line is used to store a line read from a file.

Withdraw\_quantity displays how much user withdraws quantity.

Deposit\_quantity displays how many items are deposited.

Status used to provide the status for and tracks the transaction history.

U\_name inputs and stores the username during login.

U\_pass inputs and stores the user password during login.

STRUCTURES:

Product: Contains information about products such as ID, name, quantity, price, and category.

Transactions: Holds transaction details including ID, name, price, quantity, category, and status.

FUNCTION DEFINITIONS

1. void login (int j)

Asks the user to enter a username and password.

Reads username and password from a file (login.txt).

Validates the user's credentials with the stored credentials in the file.

Allows a maximum of 3 login attempts before locking the system temporarily.

1. void printWelcome (void)

Displays a welcome message with an ASCII art logo. It only displays once when the system is first accessed.

1. void menu (void)

Displays a menu of operations to the user.

Accepts user input for the desired operation.

Based on the user's choice, it calls different functions like adding products, withdrawing, searching, displaying transactions, or exiting the program.

1. void input\_item (void)

Allows users to input product details (name, quantity, price, category).

Writes the entered product details to a file (Products.txt).

Handles cases where products with the same name are already present in the file.

1. int product\_check (char name [])

Checks if a product with a given name already exists in the product list file (Products.txt).

1. void display\_menu(void)

Displays all products stored in the file (Products.txt) in a formatted manner.

1. int assignID (void)

Assigns a unique ID to a new product based on the existing products' IDs in the file (Products.txt).

1. void search (void)

Provides options to search for products either by ID or category.

1. void search\_by\_ID (void)

Search for a product by its ID in the product list file (Products.txt).

1. void search\_by\_category (void)

Searches for products by their category in the product list file (Products.txt).

1. void withdraw(void)

Allows users to withdraw products by either ID or name.

1. void withdraw\_by\_ID (void)

Withdraws products based on their ID from the product list file (Products.txt).

1. void withdraw\_by\_name (void)

Withdraws products based on their name from the product list file (Products.txt).

1. void Transactionshistory (void)

Returns a brief rundown of all finances, stock purchase, restock.

1. Void Bill ()

Display bill or exit.

TRANSACTION HISTORY

Records transaction history in a file ("Transactions.txt").

Updates transactions for every deposit or withdrawal action.

FILE HANDLING

The code uses file operations (fopen, fclose, fprintf, fscanf, fgets) to read from and write to.

It reads user credentials from login.txt and product information from Products.txt.

Temporary files (temp.txt) are used for intermediate operations like updating or modifying the product list.

CONTROL STRUCTURES AND LOOPS

The code employs if-else statements and do-while loops for decision-making, error handling, and user interaction.

do-while loops are used in multiple places for user input validation.

while loops are used for file reading and processing.

BUILT-IN STRING FUNCTIONS

1. strcpy: Used to copy strings.
2. strcmp: Used for string comparison.
3. Strncmp: string comparison for a certain length

POINTERS

File Pointers: FILE \*fptr, temp, file, checker

USER INPUT HANDLING

scanf is used to receive input from the user.

gets is used (not recommended due to security concerns) for username input.

Getch () is used for password input.

MISCELLANEOUS

Increment/Decrement Operators: Used for counting, indexing, etc.

printf and puts for outputting messages to the console.

system("cls"): Used to clear the console screen.

exit (0): Used to terminate the program.

fflush(stdin): Used to flush the input buffer.

The code contains ASCII art for the welcome message and clear screen functionality (system("cls")).

* IMPLEMENTATION & TESTING

The implementation and testing phase will involve rigorous quality assurance to guarantee the system's reliability and accuracy. This phase is crucial to ensure that the developed solution meets the industry's standards and effectively addresses the identified challenges in inventory management.

* PROJECT BREAKDOWN STRUCTURE (WORKLOAD DISTRIBUTION WITH TIMELINE)

The project breakdown structure outlines the workload distribution among team members, with a detailed timeline for each milestone. This structured approach ensures that tasks are allocated efficiently, and progress is monitored effectively to meet project deadlines.

Zubair: Framework (Week 1-3) Structure Defining (Week 4)

Sufyan: Code implementation (Week 1-4), Testing/Implementation (Week 4), Filing (Week 3)

Talha: Features + Readability + User Interface (Week 1-4)

* RESULTS (OUTPUTS & SCREENSHOTS)

Login

A computer screen with blue text

Description automatically generated

Add ProductA screenshot of a computer

Description automatically generated

Searching an item (by name)

A screenshot of a computer

Description automatically generated

Searching an item by idA screenshot of a computer

Description automatically generated

Displaying All ProductsA screen shot of a computer

Description automatically generated

Withdrawing an itemA screenshot of a computer program

Description automatically generated

Displaying Transaction historyA screenshot of a computer program

Description automatically generated

Alert On OverstockingA screen shot of a computer program

Description automatically generated

Results (Displaying Menu) after withdrawingA screenshot of a computer program

Description automatically generated

Transaction HistoryA screenshot of a computer program

Description automatically generated

Transaction billA screenshot of a computer screen

Description automatically generated

* CONCLUSION (SUMMARY)

In conclusion, this project aims to revolutionize inventory management within the retail sector by offering a robust, scalable, and intuitive solution to address the industry's pressing needs. The comprehensive approach, from research and project selection to solution design and implementation, underscores the commitment to delivering a transformative solution for inventory management.