S./

GENESIS – Learning Outcome & Mini-project Summary Report

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **To be approved By** | **Remarks/Revision Details** |
| 1 | 15/07/21 | Arvindan P  (99004440) |  |  |  |
| 2 | 15/07/21 | Chinmayi B C  (99004441) |  |  |  |
| 3 | 15/07/21 | G Sri Nishanth  (99004442) |  |  |  |
| 4 | 15/07/21 | Shravya K N  (99004443) |  |  |  |

**Document History**

Table of Contents

[MINIPROJECT [TEAM]- Supermarket Management System Using C ++ 4](#_Toc77275695)

[MODULE/S 4](#_Toc77275696)

[TOPIC AND SUBTOPICS 4](#_Toc77275697)

[OBJECTIVES AND REQUIREMENTS 4](#_Toc77275698)

[High Level Requirements 4](#_Toc77275699)

[Table 1: High Level Requirements 4](#_Toc77275700)

[Low Level Requirements 5](#_Toc77275701)

[Table 2: Low Level Requirements 5](#_Toc77275702)

[DESIGN 6](#_Toc77275703)

[Structural diagrams 6](#_Toc77275704)

[Figure 1: Low Level System Diagram for Product 6](#_Toc77275705)

[Figure 2: Low Level System Diagram for Bill 6](#_Toc77275706)

[Behavioral diagram 7](#_Toc77275707)

[Figure 3: High Level Use Case Behavioral diagram 7](#_Toc77275708)

[Figure 4: Activity diagram for product 8](#_Toc77275709)

[Figure 5: Use case diagram for billing 9](#_Toc77275710)

[TEST PLAN 9](#_Toc77275711)

[High level test plan 9](#_Toc77275712)

[Table 3: High level test plan 9](#_Toc77275713)

[Low level test plan 10](#_Toc77275714)

[Table 4: Low level test plan 10](#_Toc77275715)

[IMPLEMENTATION SUMMARY 10](#_Toc77275716)

[OUTPUT SUMMARY 11](#_Toc77275717)

[Figure 6: Main menu 11](#_Toc77275718)

[Figure 7: Forgot password 11](#_Toc77275719)

[GIT LINK 12](#_Toc77275720)

[GIT DASHBOARD 12](#_Toc77275721)

[Figure 8: Git dashboard 12](#_Toc77275722)

[INDIVIDUAL CONTRIBUTION AND HIGHLIGHTS 12](#_Toc77275723)

[Table 5: INDIVIDUAL CONTRIBUTION 12](#_Toc77275724)

[SUMMARY 13](#_Toc77275725)

[CHALLENGES FACED AND HOW WERE THEY OVERCOME 13](#_Toc77275726)

# 

# MINIPROJECT [TEAM]- Supermarket Management System Using C ++

## MODULE/S

SDLC, Essentials of C++, Intermediate C++

### TOPIC AND SUBTOPICS

* Classes and objects
* Inheritance
* Constructors and destructors
* File handling
* Operator overloading
* reinterpret\_cast
* const keyword usage
* Mutil file system
* Makefile
* Requirements analysis, SWOT analysis, 4W 1H
* Test plan

## OBJECTIVES AND REQUIREMENTS

1. The objective of the “Supermarket Management System “is to produce the monitoring the supermarket.

2. The project Supermarket Management System is developed with the objective of making the system reliable, easier, fast, and more informative

3. The admin can create a product, display all the products, display all the price list, modify any product and can also delete any product.

4. The user can place the order of their interest with all the available products in the market.

4. The goal of this phase is to develop the internal logic of each of the modules identified during the system design.

### High Level Requirements

|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Status** |
| H1 | User class with required features | Implemented |
| H2 | Customer class with required features | Implemented |
| H3 | File handling of User login details | Implemented |
| H4 | Product class with required features inherits from User class | Implemented |
| H5 | Transactions/Billing using class Bill which inherits from Customer and Product class | Implemented |
| H6 | Display of the menu and working of all the features | Implemented |

### Table 1: High Level Requirements

### Low Level Requirements

|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Status** |
| H1\_L1 | User has own username and password maintained in the .txt file | Implemented |
| H1\_L2 | Adding User function in User class | Implemented |
| H1\_L3 | Verifying User function in User class | Implemented |
| H2\_L1 | Reading the customer details Customer class | Implemented |
| H2\_L2 | Displaying Customer details function in Customer class | Implemented |
| H3\_L1 | Maintain the user.txt file | Implemented |
| H4\_L1 | Creating Product function in Product class | Implemented |
| H4\_L2 | Showing the Product function in Product class | Implemented |
| H4\_L3 | Displaying all Product details function in Product class | Implemented |
| H4\_L4 | Displaying specific Product details function in Product class | Implemented |
| H4\_L5 | Price list of Product function in Product class | Implemented |
| H4\_L6 | Modify Product function in Product class | Implemented |
| H4\_L7 | Admin menu function in Product class | Implemented |
| H4\_L8 | Deleting Product function in Product class | Implemented |
| H5\_L1 | Placing order function in class Bill | Implemented |

### Table 2: Low Level Requirements

## DESIGN

### Structural diagrams

Graphical user interface

Description automatically generated

### Figure 1: Low Level System Diagram for Product

**Timeline

Description automatically generated**

### Figure 2: Low Level System Diagram for Bill

### Behavioral diagram

Diagram, schematic

Description automatically generated

### Figure 3: High Level Use Case Behavioral diagram

**Diagram

Description automatically generated**

### Figure 4: Activity diagram for product

**Diagram

Description automatically generated**

### Figure 5: Use case diagram for billing

## TEST PLAN

### High level test plan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Exp I/P** | **Exp O/P** | **Status** | **Type of Test** |
| H\_01 | Maintaining the User Login credentials in .txt file | username and password | .txt file has all user data | Passed | Requirement Based |
| H\_02 | Maintaining product details in .dat file | product class input | .dat has all product details | Passed | Requirement Based |
| H\_03 | Printing the bill | User and Product class | generation of bill | Passed | Requirement Based |
| H\_04 | Display of menu for user | Inputting the choice in product class | Display of menu | Passed | Requirement Based |
| H\_05 | Fetching the product details based on product id | product ID | Display of product details | Passed | Requirement Based |

### Table 3: High level test plan

### Low level test plan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Exp I/P** | **Exp O/P** | **Status** | **Type of Test** |
| L\_01 | Displaying the price list | Choosing the price list option in admin menu | Display of price list | Passed | Requirement Based |
| L\_02 | Modifying the product available in .dat file | product id | Printing "Record updated" | Passed | Requirement Based |
| L\_03 | Deleting product available in .dat file | product id | Delete product from .dat file | Passed | Requirement Based |
| L\_04 | Display of admin menu for various functions | option in integer format | Going to corresponding function | Passed | Requirement Based |
| L\_05 | Modifying the poduct not availble in .dat file | invalid product id | Printing "Record not found" | Passed | Scenario Based |
| L\_06 | Invalid choice for admin menu | invalid choice in integer format | Printing "Invalid Choice" | Passed | Boundary Based |
| L\_07 | Deleting invalid product from .dat file | invalid product id | Printing "Record not found" | Passed | Scenario Based |
| L\_08 | Creating new product | Valid product ID | Product is created | Passed | Requirement based |
| L\_09 | Creating new product | Invalid product ID | Prints “Incorrect product ID” | Passed | Scenario Based |
| L\_10 | Displaying specific product | Valid product ID | Product is created | Passed | Requirement based |
| L\_11 | Displaying specific product | Invalid product ID | Prints “Incorrect product ID” | Passed | Scenario Based |

### Table 4: Low level test plan

## IMPLEMENTATION SUMMARY

In the implementation part, there are four classes which User, Customer, Product and Bill. In User class it maintains the data of user by maintaining username and password in the user.txt file. In the Customer class which inherits from user, reading and displaying the customer data is made. In the product class, data has been maintained using .dat file where user can maintain the product data by providing id, name, price, and discount values. All the features like modify, display, create, delete products, add user are available in admin menu. Billing Class inherits from User and Product class to print the bill for specific user.

### OUTPUT SUMMARY

Text

Description automatically generated

### Figure 6: Main menu

Text

Description automatically generated

### Figure 7: Forgot password

### GIT LINK

<https://github.com/99004440-Arvindan/EMBEDDED_MINI_PROJECT>

### GIT DASHBOARD

A picture containing text, screenshot, computer, computer

Description automatically generated

### Figure 8: Git dashboard

## INDIVIDUAL CONTRIBUTION AND HIGHLIGHTS

|  |  |  |
| --- | --- | --- |
| PS Number | Name | Contribution |
| 99004440 | Arvindan P | Worked on User class, UML Diagrams, Test Plan, SWOT Analysis, PPT |
| 99004441 | Chinmayi B C | Worked on Product class, UML Diagrams, 4W’s and 1H, Test Plan, Report |
| 99004442 | G Sri Nishanth | Worked on Billing class, UML Diagrams, Introduction and Features, Test Plan, PPT |
| 99004443 | Shravya K N | Worked on Product class, UML Diagrams, High and Low level Requirements, Test Plan, Report |

### Table 5: INDIVIDUAL CONTRIBUTION

### SUMMARY

After combining everyone’s work we are able to develop *supermarket management system*. This helps shopkeeper to keep track of products, organize efficiently and supervise effectively. Hence he/she can give better service to the customer. It provides multiple options to the shopkeeper such as add new product, view all products or specific product, modify any product, delete any product. Based on these details customer/shopkeeper can find product details accurately.

### CHALLENGES FACED AND HOW WERE THEY OVERCOME

* Had problem in saving the information in .dat and .txt file format. Used reinterpret\_cast to solve the issue.
* Dealing with the billing for each user. Used inheritance from both product and user to resolve.