Supermarket

Management System

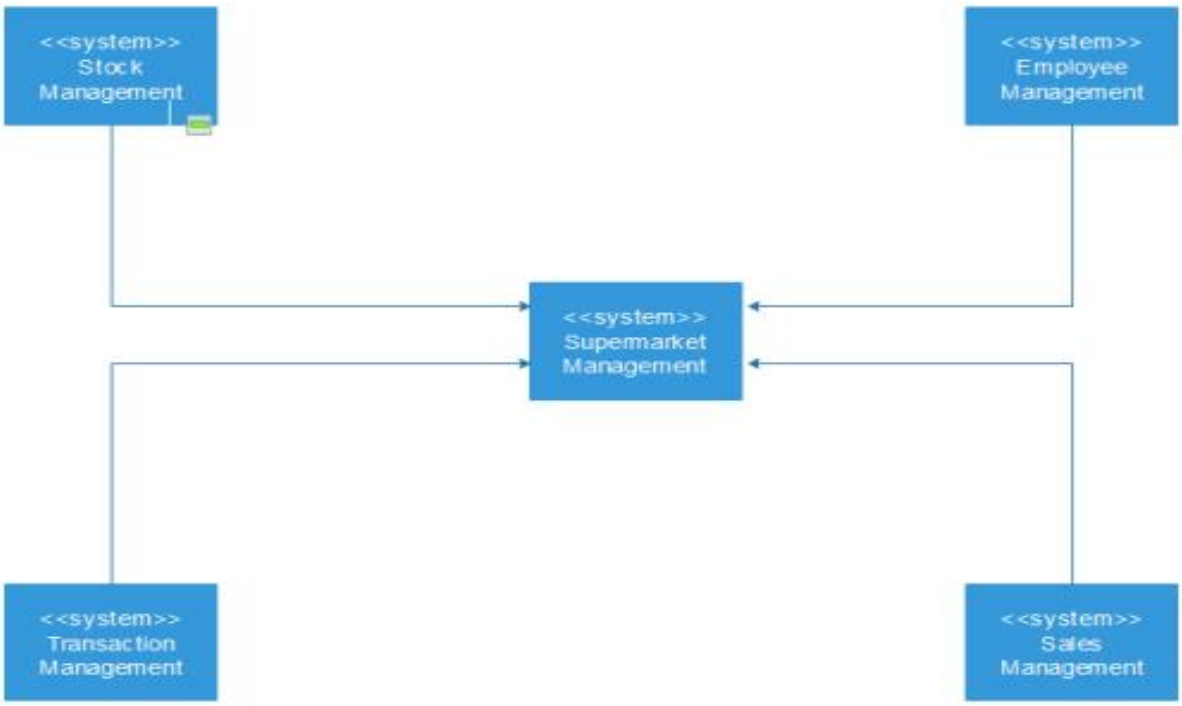
علي يحيي زكريا فهمي 2021170820
احمد سامح احمد مختار 2021170802
فادي مجدي زكي ابراهيم 2021170824
عبدالله احمد شحاته هلال 2021170819
محمد عماد عاطف عبدالحميد غانم 2021170830
احمد طارق محمد كمال 2021170804
كريم فؤاد شهب محمد 2021170826
سامح وليد اسماعيل محسن *2021170812*

TA / Lobna Mady
Team Id: 11 SWE

Introduction

- A supermarket management system designed to help supermarket managers to run their businesses more efficiently. It provides an all-in-one solution that enables them to manage and display items, transactions, sales, budget and employees of the supermarket.

Context Diagram



User requirements

- Management system for a supermarket with login system in which the user can login as an admin or stock controller or salesperson or cashier.

- As an admin

    1. The user can search, view and manage the supermarket items by adding or editing or deleting items.
    2. The user can search, view and manage the supermarket employees' information.
    3. The user can view the supermarket budget and deposit money.
    4. The user can search , view and manage the list of transactions made.

- As a stock controller

    1. The user can view the items in the supermarket.
    2. The user can restock items in the supermarket.

- As a salesperson

    1. The user can view the items in the supermarket.
    2. The user can place a sale on items in the supermarket.

- As a cashier

    1. The user can validate the customer credit card then add a transaction and deposit the money.

Functional requirements

1. **Login:**

   Description: Function that allows user to access the system.
   Input: Username and password.
   Input source: Textbox.
   Pre-conditions: username and password validation.
   Post-conditions: verify user authority.
   Output: opens form.

2. **Logout:**

   Description: Function that allows user to sign out from the system.
   Input: Button clicked.
   Input source: Button.
   Pre-conditions:
   Post-conditions:
   Output: redirect to login page.

3. **View/Search Items:**

   Description: Function that display items stored in the database.
   Input: Item id.
   Input source: Combobox.
   Pre-conditions: Item id should be available.
   Post-conditions: Retrieve successfully.
   Output: Items data in a view grid.

4. **Manage items(add/edit/delete):**

   Description: Function that allows the user to insert or edit or delete items in the database.
   Input: Item id.
   Input source: Textbox/Combobox.
   Pre-conditions: (add): Item id should not be used,**(edit/delete):**Item id should be available.
   Post-conditions: No null values.
   Output: MessageBox.

5. **Restock item:**

   Description: Function that user can use to increase item quantity.
   Input: Items id / Count to be added.
   Input source: ComboBox / TextBox.
   Pre-conditions: Item should be available / count should be greater than zero.
   Post-conditions: Database value updated.
   Output: MessageBox**.**

6. **Place a sale:**

   Description: Function that user can use to reduce price of an item.
   Input: Item id/ sale percentage.
   Input source: ComboBox/Textbox.
   Pre-conditions: Item should be available/ sale percentage doesn't exceed 99%.
   Post-conditions: Database value updated successfully.
   Output: MessageBox.

7. **View/Search Employees' Data:**

   Description: Function that display employees' information stored in the database.
   Input: Employee id.
   Input source: Combobox**.**
   Pre-conditions: Employee id should be available.
   Post-conditions: Retrieve successfully.
   Output: employees' data in a view grid.

8. **Manage Employees' Data(add/edit/delete):**

Description: Function that allows the user to insert or edit or delete employees in the database.
Input: Employee id.
Input source: Textbox/Combobox.
Pre-conditions: (add):employee id should not be used,**(edit/delete):**employee id should be available.
Post-conditions: No Null values.
Output: MessageBox.

9. **View Budget:**

Description: Function that allows user to display supermarket budget from the database.
Input: Button clicked.
Input source: Button.
Pre-conditions: user has the authority to use the function.
Post-conditions: Retrieve successfully.
Output: Budget value inside textbox.

10. **deposit money:**

Description: Function that allows users to increase budget value in the database.
Input: Value to be added.
Input source: Textbox.
Pre-conditions: Value should be greater than zero.
Post-conditions: Database value updated successfully.
Output: Messagebox.

11. **Validate customer credit card:**

Description: Function that the user can use to validate the customer's credit card.
Input: Credit card number.
Input source: Textbox.
Pre-conditions: Textbox value is not null.
Post-conditions:
Output: add transaction procedure call.

12.View/Search Transactions:

Description: Function that display Transactions information stored in the database.
Input: Transaction id.
Input source: Combobox.
Pre-conditions: Transaction id should be available.
Post-conditions: Retrieve successfully.
Output: Transactions information in a view grid.

13. **Manage Transactions(add/edit/delete):**

Description: Function that allows the user to insert or edit or delete transactions in the database.
Input: Transaction id.
Input source: Textbox/Combobox.
Pre-conditions: (add): Transaction id should not be used,**(edit/delete):**Transaction id should be available.
Post-conditions: No Null values.
Output: MessageBox.

Non-Functional requirements

- **System Requirements:**

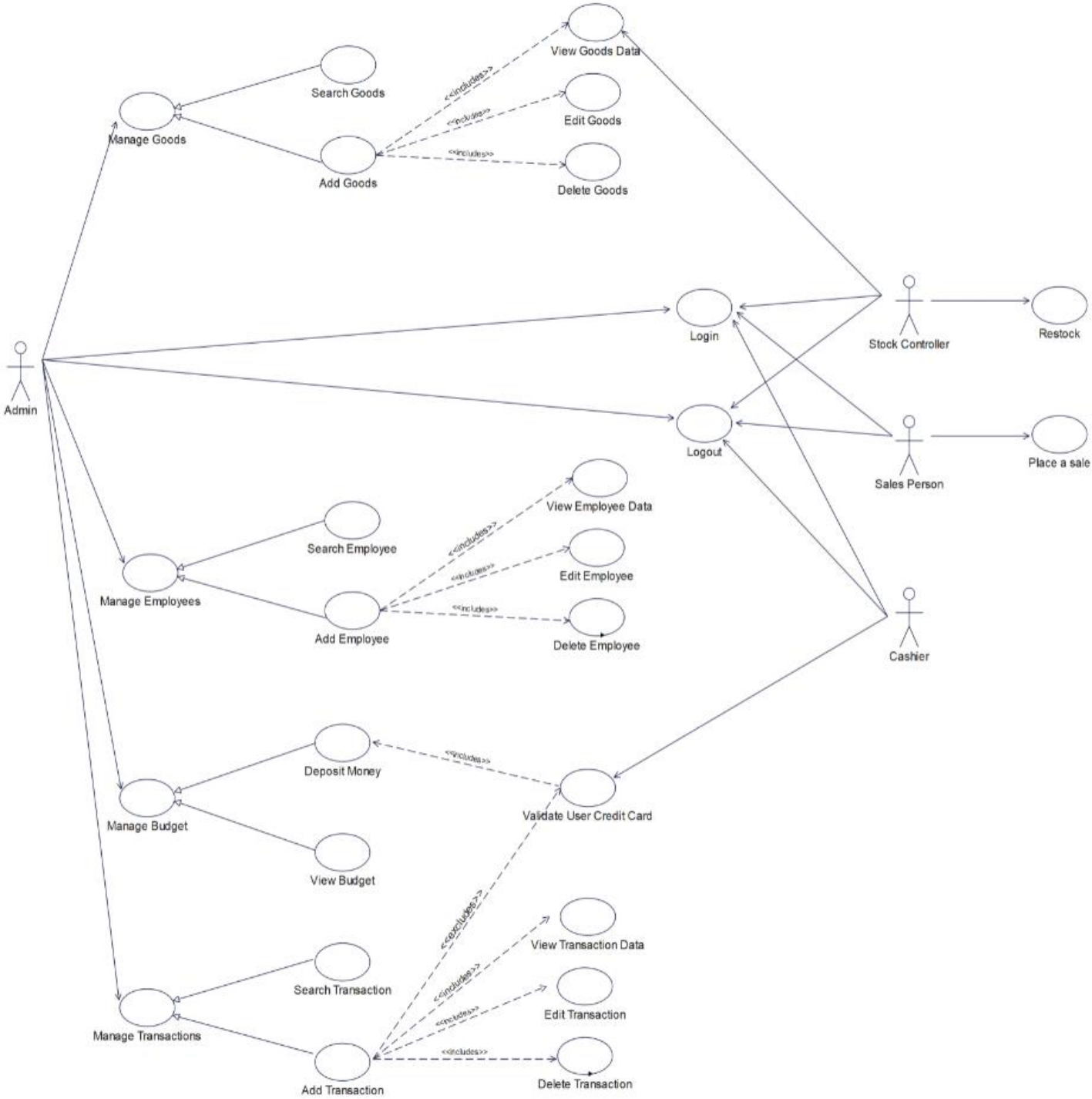
  1. **operating system:** windows 7.
  2. **Processor type:** Intel core i3.
  3. **Storage requirements:** 2GB.

- **Technology Requirements:**

  1. Visual studio C# windows form (.net Framework).
  2. Oracle Database
  3. SQL developer

- **Reliable** to handle unexpected conditions and errors.
- **Safe** from unauthorized access and malware attacks
- **Usable** so it is easily used by users.
- **Plan driven Development method.**
- **Response/Execution Time should not exceed 5 seconds.**

Use Case diagram



Sequence diagram

- Stock controller view goods use case as sequence diagram.