**Software Requirements Specification**

**for**

**FreshMart**

**Version 1.0 approved**

**Prepared by Mohsin Dodhiya, Mohmadhusen Khimani, Ramiz Ghada**

**RK University, Rajkot**

**28/11/2024**

### 

### 

### 

**Table of Contents**

**Table of Contents** [**ii**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.23ckvvd)

**Revision History ii**

**1. Introduction** [**1**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.ihv636)

1.1 Purpose [1](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.32hioqz)

1.2 Document Conventions [1](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.1hmsyys)

1.3 Intended Audience and Reading Suggestions [1](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.41mghml)

1.4 Project Scope [1](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.2grqrue)

1.5 References [1](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.vx1227)

**2. Overall Description** [**2**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.3fwokq0)

2.1 Product Perspective [2](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.1v1yuxt)

2.2 Product Features [2](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.4f1mdlm)

2.3 User Classes and Characteristics [2](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.2u6wntf)

2.4 Operating Environment [2](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.19c6y18)

2.5 Design and Implementation Constraints [2](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.3tbugp1)

2.6 User Documentation [2](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.28h4qwu)

2.7 Assumptions and Dependencies [3](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.nmf14n)

**3. System Features** [**3**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.37m2jsg)

3.1 System Feature 1 [3](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.1mrcu09)

3.2 System Feature 2 (and so on) [4](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.46r0co2)

**4. External Interface Requirements** [**4**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.3j2qqm3)

4.1 User Interfaces [4](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.1y810tw)

4.2 Hardware Interfaces [4](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.2lwamvv)

4.3 Software Interfaces [4](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.111kx3o)

4.4 Communications Interfaces [4](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.3l18frh)

**5. Other Nonfunctional Requirements** [**5**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.206ipza)

5.1 Performance Requirements [5](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.2bn6wsx)

5.2 Safety Requirements [5](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.4k668n3)

5.3 Security Requirements [5](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.2zbgiuw)

5.4 Software Quality Attributes [5](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.1egqt2p)

**6. Other Requirements** [**5**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.3ygebqi)

**Appendix A: Glossary** [**5**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.2dlolyb)

**Appendix B: Analysis Models** [**6**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.sqyw64)

**Appendix C: Issues List** [**6**](https://docs.google.com/document/d/1Sd476VSK5UtsZPYuUkiknF4ehDgtch2x/edit#heading=h.3cqmetx)

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

### **1. Introduction**

#### **1.1 Purpose**

*The purpose of this SRS is to provide a detailed description of the Fresh Mart Supermarket Desktop Application, which will be developed in C# .NET. The document will serve as a guide for developers, designers, and stakeholders.*

#### **1.2 Document Conventions**

#### *This document adheres to standard SRS conventions and highlights all requirements, such as functional and non-functional requirements, assumptions, and dependencies.*

**1.3 Intended Audience and Reading Suggestions**

*The SRS is intended for developers, project managers, and system administrators. The document begins with an overview, followed by detailed descriptions of system features and external interface requirements.*

#### **1.4 Project Scope**

*The Fresh Mart Supermarket Desktop Application will allow store administrators to manage inventory, process sales, track orders, and generate reports. The system will include functionalities such as product management, sales transactions, and user account management.*

**1.5 References**

* *FreshMart Project Report by Mohmadhusen Ahmadbhai Khimani, RK University, Nov 2024. Details the system overview and technical requirements for FreshMart.*
* *SRS Template Document provided by RK University Faculty. Used as a standard framework for this SRS.*
* *YouTube Tutorial on a similar supermarket management system project*

**2. Overall Description**

**2.1 Product Perspective**

*The FreshMart system is a standalone desktop application developed in C# .NET. It serves as a robust inventory and sales management solution, replacing manual or basic electronic systems commonly used in similar supermarkets.*

**2.2 Product Features**

1. ***Product Management***
   * *Add Products: Allows admins to add new products with details such as name, description, SKU, price, stock quantity, and category. Additional fields may include supplier information, product images, and reorder levels.*
   * *Edit Products: Enables admins to update product information, such as adjusting prices, updating descriptions, modifying stock levels, or changing the product status (e.g., active or discontinued).*
   * *Remove Products: Allows admins to delete products that are no longer in stock or offered. To maintain historical sales data, products are often archived rather than permanently deleted.*
   * *Product Categorization and Filtering: Supports categorizing products by type, brand, or department, with search and filtering options to facilitate faster retrieval.*
2. ***Sales Processing***
   * *Sales Cart Management: Provides a point-of-sale (POS) interface for adding items to a customer’s shopping cart. Cashiers can search for products by name, SKU, or barcode, and easily add items to the cart.*
   * *Discount and Promotion Handling: Supports applying discounts or promotions to individual items or the entire cart, either by entering discount codes or applying preset promotional pricing.*
   * *Payment Processing: Offers multiple payment options, such as cash, card, or digital payment methods, and allows split payments if needed. The system ensures accurate calculation of tax and total costs.*
   * *Receipt Generation: Automatically generates and prints a detailed receipt after a successful transaction, displaying itemized purchases, subtotal, taxes, discounts, and total amount paid. Digital receipts can also be sent via email if enabled.*
   * *Refund and Return Handling: Supports processing refunds or exchanges by locating the original sale record and reversing or adjusting the transaction accordingly.*
3. ***Order Tracking***
   * *Sales History Maintenance: Records details of each transaction, including date and time, items sold, quantity, price, payment method, and cashier involved. Each transaction is stored with a unique transaction ID for easy lookup.*
   * *Search and Filtering for Past Transactions: Enables search by date, transaction ID, product SKU, or cashier ID, allowing admins or authorized staff to quickly locate specific transactions.*
   * *Order Status Monitoring: Helps track the fulfillment status of any pending or special orders, such as backorders or customer reservations, and notifies staff when these orders are completed.*
4. ***User Management***
   * *Role-Based Access Control (RBAC): Enables secure management of access to various parts of the system. Admins can assign user roles (e.g., Admin, Cashier, Inventory Manager) with specific permissions according to the required functionality.*
   * *User Account Management: Allows admins to create, edit, and deactivate user accounts as needed. Each account can include the user’s name, role, login credentials, and contact information.*
   * *Activity Logging and Monitoring: Tracks actions performed by each user to maintain accountability, including login history, inventory changes, and sales transactions processed by each cashier.*
5. ***Reporting***
   * *Sales Reports: Provides insights into sales performance by generating reports based on custom timeframes (e.g., daily, weekly, monthly, yearly). Reports detail sales volume, total revenue, average transaction value, and peak sales hours.*
   * *Inventory Reports: Tracks stock levels, reorder points, and product movement trends. Low-stock reports help admins proactively manage restocking.*
   * *Staff Performance Reports: Analyzes cashier and sales staff productivity by monitoring the volume and value of transactions handled by each team member.*
   * *Customer Purchase Trends: Offers insights into popular products or categories, allowing store admins to tailor inventory and promotions based on customer preferences.*
   * *Export and Sharing Options: Allows reports to be exported in formats like CSV or PDF, supporting easy sharing and integration with other analytics or accounting tools.*

**2.3 User Classes and Characteristics**

***1. Store Admins***

* *Role & Responsibilities:  
  Store Admins oversee the entire application, including managing inventory, processing sales, and overseeing cashier activity.*
* *Frequency of Use:  
  Daily; Admins regularly log in to monitor transactions, update product details, review sales reports, and manage cashier access.*

***2. Cashiers***

* *Role & Responsibilities:  
  Cashiers handle daily sales transactions, manage product checkout, and print receipts.*
* *Frequency of Use:  
  Constant use during shifts; Cashiers access the system throughout the day to complete sales transactions.*
* *Subset of Functions:  
  Limited to the sales processing and checkout functions, including adding products to the sales cart and completing transactions.*
* *Security/Privilege Level:  
  Limited access; cashiers have restricted privileges that prevent them from accessing inventory management, user management, and system settings.*

**2.4 Operating Environment**

1. ***Hardware Platform:****The software will operate on standard desktop computers equipped with barcode scanners, receipt printers, and cash drawers. The minimum hardware requirements include:*
   * *Processor: Intel Core i3 or equivalent*
   * *RAM: 4GB or more*
   * *Storage: 500MB free space for application installation*
   * *I/O Ports: USB or Bluetooth for barcode scanners and receipt printers*
2. ***Operating System:****Windows 10 or higher is the recommended platform, as the application is developed using the C# .NET framework. Cross-platform support may be achievable through Mono, allowing limited compatibility with Linux and macOS if necessary.*
3. ***Other Required Software:***
   * *Database Management System (DBMS): SQL Server or MySQL for managing product, sales, and user data.*
   * *.NET Framework: The application requires the .NET runtime environment, specifically .NET 5.0 or newer, for optimal performance and compatibility.*
   * *Email System: Integration with SMTP for sending end-of-day sales summaries to admins.*
4. ***Network Requirements:****The application can function in an offline mode for daily operations, with internet access only required for email reporting and remote database backups.*
5. ***Peripheral Compatibility:****The application must support common peripherals such as barcode scanners for efficient product entry and receipt printers for customer transactions.*

**2.5 Design and Implementation Constraints**

1. ***Corporate and Regulatory Policies:***
   * *Compliance with local data privacy laws (e.g., GDPR for user data protection) is essential, especially for storing customer information and transaction data.*
   * *User data should be securely stored and transmitted, adhering to best practices in data encryption.*
2. ***Hardware Limitations:****The system relies on barcode scanners and receipt printers; thus, hardware compatibility checks and integration testing are necessary. Hardware limitations such as processing speed and memory capacity could impact the software's ability to handle high transaction volumes and inventory size.*
3. ***Software and Technology Requirements:***
   * *Database System: The application is designed to work with SQL Server or MySQL. Compatibility with other database systems will require additional testing and configuration adjustments.*
   * *Development Language and Framework: The system is built on C# .NET, so it must remain compatible with the .NET environment for code maintenance, updates, and debugging.*
   * *User Interface Standards: The UI must follow standard Windows application guidelines to ensure a consistent user experience across devices and future .NET updates.*
4. ***Communication Protocols:***
   * *Internet connectivity is needed for email reporting and updates; thus, the application should support basic internet protocols like SMTP and HTTPS for secure communication.*
5. ***Security Considerations:***
   * *User authentication should employ password hashing and possibly multi-factor authentication (MFA) for admin access.*
   * *Role-based access control should restrict user privileges according to their roles (Admin, Cashier, IT Support).*
6. ***Programming Standards and Conventions:***
   * *The codebase must follow .NET coding conventions for readability and maintainability.*
   * *Documentation standards should be adhered to, making it easier for future developers to understand and modify the code as needed.*

**2.6 User Documentation**

1. ***User Manuals:*** *A detailed user manual will be provided to help Admins and Cashiers understand system functionalities and operational workflows. This manual will cover topics such as product management, sales processing, report generation, and troubleshooting.*
2. ***Online Help and Tooltips:****Built-in tooltips will guide users through key functions, reducing the learning curve. An online help section with FAQs and quick reference guides will be accessible directly within the application.*
3. ***Video Tutorials:*** *Video tutorials covering basic tasks, such as adding products, processing sales, generating reports, and navigating the dashboard, will be available. These tutorials will support new users and reduce training time.*
4. ***Technical Documentation for IT Support:****A separate technical document for IT support will outline system setup, backup procedures, troubleshooting common issues, and performing regular maintenance tasks. This documentation will ensure that IT staff can efficiently manage the system and address potential technical issues.*
5. ***Formats and Delivery:***
   * *User Manual and Quick Reference Guides: Provided as downloadable PDFs within the application or accessible through an internal network.*
   * *Online Help: HTML-based help files integrated within the application.*
   * *Video Tutorials: Hosted on a designated platform (e.g., YouTube or private server) and accessible via the help menu.*

*This documentation aims to provide comprehensive support for all user roles, enhancing ease of use, reducing operational errors, and ensuring smooth software deployment and adoption.*

### **2.7 Assumptions and Dependencies**

1. ***Operating Environment:***
   * *Assumed that the system will operate on Windows OS in a supermarket environment with reliable access to desktop computers, barcode scanners, and receipt printers. Any deviation, such as using different hardware or operating systems, may require additional compatibility adjustments.*
2. ***Database Management****:*
   * *Assume the system will use SQL Server as its primary database. The application is dependent on SQL Server compatibility for optimal performance. A switch to another database could necessitate significant code modification.*
3. ***Internet and Network Access:***
   * *FreshMart relies on intermittent internet connectivity for tasks like report emails. It is assumed that network access will be available when required. If access is limited, some features may become unavailable or require modification.*
4. *Budget and Timeline Constraints:*
   * *The project must be completed within specified academic timelines and budget constraints. Significant changes in these constraints could limit the functionality or features included in the final system.*

## **3. System Features**

### **3.1 Product Management**

### *The Product Management module enables the Admin to manage the supermarket's inventory, including adding, updating, and removing products.*

### ***Description and Priority:*** *This is a high-priority feature since accurate inventory is essential for daily operations, customer satisfaction, and preventing stockouts.*

### ***Sub-Features:***

### ***Add New Product:*** *Admins can add new products to the inventory, specifying details such as product name, price, category, SKU, supplier, quantity, and expiration date (if applicable).*

### *Functional Requirements:*

### *REQ-1: The system must allow Admins to add a product with all relevant details.*

### *REQ-2: The system must validate product data (e.g., price format, positive quantity).*

### *REQ-3: SKU numbers should be unique to prevent duplication.*

### ***Edit Product Information:*** *Enables updating product details, such as price adjustments, stock quantities, or product re-categorization.*

### *Functional Requirements:*

### *REQ-4: The system must allow Admins to modify product attributes.*

### *REQ-5: Updates must reflect immediately in the inventory to ensure accuracy.*

### ***Remove Product:*** *Allows Admins to remove outdated or discontinued items from inventory.*

### *Functional Requirements:*

### *REQ-6: The system must allow Admins to delete products securely.*

### *REQ-7: Deleted items must be archived or removed based on system settings for data retention.*

### ***Stimulus/Response Sequences:***

### *Stimulus: Admin chooses to add a new product and inputs product details.*

### *Response: The system validates inputs, adds the product to the database, and confirms addition with an on-screen message.*

### 

### **3.2 Sales Processing**

### *The Sales Processing module allows cashiers to scan products, manage carts, apply discounts, process payments, and print receipts. This is crucial for handling customer transactions accurately and efficiently.*

### ***Description and Priority:*** *This is a high-priority feature, as it directly impacts customer experience and revenue generation.*

### ***Sub-Features:***

### *Product Scanning and Cart Management: Cashiers can scan product barcodes to add items to the cart, adjust quantities, and view the current cart total.*

### *Functional Requirements:*

### *REQ-8: The system must support barcode scanning to add products to the cart.*

### *REQ-9: Cashiers must be able to adjust quantities within the cart.*

### *REQ-10: Real-time updates to cart totals must be displayed.*

### *Apply Discounts and Taxes: Enables authorized users to apply discounts, calculate applicable taxes, and update the total.*

### *Functional Requirements:*

### *REQ-11: Admins can set discount rules and tax rates.*

### *REQ-12: Discounts and taxes must update the cart total accurately.*

### *Payment Processing and Receipt Generation: Supports various payment methods (cash, card, etc.), completes transactions, and generates receipts for customers.*

### *Functional Requirements:*

### *REQ-13: The system must support cash and card payments.*

### *REQ-14: Receipts must display transaction details, including product list, subtotal, discounts, taxes, and total amount.*

### ***Stimulus/Response Sequences:***

### *Stimulus: Cashier scans products and proceeds to checkout.*

### *Response: The system calculates the total, processes payment, and prints a receipt.*

### **3.3 Order Management**

### *The Order Management feature tracks all completed transactions and maintains an order history.*

### ***Description and Priority:*** *Medium priority; this feature supports backend management and customer service functions.*

### ***Sub-Features:***

### *Order History Review: Admins can view and filter past transactions based on criteria such as date, total amount, or customer.*

### *Functional Requirements:*

### *REQ-15: The system must store transaction details in an accessible order history.*

### *REQ-16: Admins can filter and sort order history based on specific parameters.*

### *Order Lookup and Returns: Enables searching specific orders and processing product returns if needed.*

### *Functional Requirements:*

### *REQ-17: The system must allow order searches using unique order IDs.*

### *REQ-18: Returns processing should automatically update inventory and reflect in sales reports.*

### ***Stimulus/Response Sequences:***

### *Stimulus: Admin searches for a specific order to review details.*

### *Response: The system displays order information and allows further actions (e.g., refund)*

### **3.4 Reporting**

### *The Reporting module generates periodic reports (daily, weekly, and monthly) that summarize sales, inventory changes, and user activity. This feature provides insights for informed decision-making and operational efficiency.*

### ***Description and Priority:*** *High priority for managerial oversight, financial planning, and inventory control.*

### ***Sub-Features:***

### *Sales Reports: Generate detailed reports on daily, weekly, and monthly sales, showing revenue, number of transactions, and peak hours.*

### *Functional Requirements:*

### *REQ-23: The system must generate sales reports showing revenue, most sold products, and transaction counts.*

### *REQ-24: Admins can select date ranges for custom reports.*

### *Inventory Reports: Summarizes stock levels, low-stock alerts, and restocking recommendations.*

### *Functional Requirements:*

### *REQ-25: The system must generate inventory status reports, including low-stock alerts.*

### *REQ-26: Inventory reports should highlight stock levels for high-demand products.*

### *User Activity Reports: Tracks user logins, logout times, and activities for auditing purposes.*

### *Functional Requirements:*

### *REQ-27: The system must log user actions with timestamps for audit purposes.*

### *REQ-28: Activity reports should be filterable by user role and date.*

### ***Stimulus/Response Sequences:***

### *Stimulus: Admin selects a date range to generate a sales report.*

### *Response: The system compiles data and presents the report, ready for export or printing.*

### 

### **4. External Interface Requirements**

#### **4.1 User Interface**

* *Login Screen: Secure login for staff members.*
* *Dashboard: Displays sales data, inventory levels, and user options.*
* *Product Management Interface: A form for adding, editing, and deleting products.*
* *Sales Screen: Allows cashiers to scan items, view cart contents, and process payments.*
* *Reports Screen: Displays sales and inventory reports.*

#### **4.2 Hardware Interface**

*Compatible with desktop computers, barcode scanners, receipt printers, and cash drawers.*

#### **4.3 Communication Interface**

*Email notifications for end-of-day sales reports.*

*Integration with external payment processors if needed.*

#### **4.4 Software Interface**

*Uses a database management system (e.g., SQL Server or MySQL).*

*Built with C# .NET framework.*

*Runs on Windows OS with support for other platforms through Mono.*

## **5. Other Nonfunctional Requirements**

### **5.1 Performance Requirements**

1. ***Response Time:***
   * *The application should respond to most user interactions (e.g., adding a product to the cart, completing a sale) within 2 seconds to maintain a smooth user experience at checkout.*
   * *Data queries, such as product lookups and inventory checks, must be processed within 3 seconds under normal conditions.*
2. ***Transaction Throughput:***
   * *The system must support up to 100 concurrent transactions per minute during peak hours without noticeable degradation in performance.*
   * *Daily transaction capacity should accommodate at least 10,000 transactions to meet the needs of high-traffic supermarkets.*
3. ***Scalability:***
   * *The application must support scalability, allowing it to handle an increased number of products (up to 100,000 SKUs) and transactions as business demands grow.*
   * *System architecture should allow for expansion in storage and processing power without significant code refactoring.*
4. ***Availability:***
   * *The system must have an uptime of 99.5%, particularly during business hours, to ensure minimal disruption to supermarket operations.*
   * *Scheduled maintenance downtime should be kept to non-operational hours or low-traffic times to avoid impact on day-to-day operations.*

### **5.2 Safety Requirements**

1. ***Data Backup and Recovery:***
   * *The application must perform daily automatic backups to prevent data loss in the event of system failure. Backups should include transactional data, inventory records, and user activity logs.*
   * *A disaster recovery plan should be implemented to restore data from the latest backup within 2 hours of a critical failure.*
2. ***Error Handling and Logging:***
   * *Errors, such as failed transactions or database connection issues, must trigger detailed logging and alert the Admin.*
   * *Critical errors should be logged with details including timestamp, affected components, and error messages to support troubleshooting and audit trails.*
3. ***Transaction Integrity:***
   * *The system must ensure transaction accuracy, particularly during sales processing, to prevent discrepancies between sales records and inventory.*
   * *Transactions must be atomic (all-or-nothing), ensuring that either the entire transaction completes successfully, or no changes are made, preserving data integrity.*
4. ***User Safety:***
   * *The application should follow user interface best practices, such as clear navigation and prompts, to reduce the risk of user errors during transactions.*
   * *Physical safety considerations, such as system placement away from cash drawers or any dangerous hardware parts, should be implemented to protect user well-being.*

### **5.3 Security Requirements**

1. ***Authentication and Access Control:***
   * *The system must require secure login credentials, with password hashing for data security. Multi-factor authentication (MFA) may be required for Admin access.*
   * *Role-based access control (RBAC) must be enforced, limiting access based on user roles (e.g., Cashier, Admin) to prevent unauthorized data access or changes.*
2. ***Data Encryption:***
   * *Sensitive data, including user credentials, transaction history, and customer information, must be encrypted at rest (using AES-256) and in transit (using TLS/SSL) to ensure data security and compliance with data protection regulations.*
   * *Backup files should also be encrypted and securely stored to prevent unauthorized access.*
3. ***Audit Logging:***
   * *The system must maintain an audit log of all critical actions, such as product additions, deletions, user logins, and access to financial reports.*
   * *Audit logs must include details like timestamp, user ID, action performed, and any changes to data to support security and compliance audits.*
4. ***Compliance with Data Protection Regulations:***
   * *FreshMart must comply with relevant local data privacy laws, such as GDPR, to ensure the protection of customer and transaction data.*
   * *Access to customer information should be restricted to only necessary personnel, and data retention policies should align with regulatory requirements.*

### **5.4 Software Quality Attributes**

1. ***Usability:***
   * *The user interface must be intuitive and user-friendly, with consistent navigation and layout across different screens (e.g., Sales, Product Management, Reports).*
   * *The system should require minimal training, especially for Cashiers, with clear labels, buttons, and error messages that enhance the user experience.*
2. ***Reliability:***
   * *The application should maintain high reliability, especially during peak transaction times, to ensure smooth operation.*
   * *Key functions, such as product scanning, checkout, and receipt generation, must perform consistently without errors during normal use.*
3. ***Maintainability:***
   * *The codebase should be modular, using well-defined classes and functions, making it easy to update, debug, or enhance specific features as needed.*
   * *Documentation standards must be followed, with detailed comments and structured function names to facilitate code understanding for future developers.*
4. ***Portability:***
   * *While primarily designed for Windows, the system should be compatible with other operating systems (Linux, macOS) through Mono or similar frameworks if required.*
   * *The database structure should be portable across different SQL-based DBMS (e.g., MySQL, PostgreSQL) with minimal changes.*
5. ***Extensibility:***
   * *The application architecture should allow for new features to be added in future updates without significant redesigns.*
   * *For example, modules for loyalty programs, discounts, or integration with third-party e-commerce platforms should be easy to add or integrate into the system.*
6. ***Interoperability:***
   * *The system should support interaction with other supermarket hardware and software systems, such as barcode scanners, receipt printers, and payment gateways, using standard APIs.*
   * *Data exports, such as reports or inventory data, should be compatible with common file formats (CSV, PDF) for easy integration with external software tools.*

## **6. Other Requirements**

### **6.1 Database Requirements**

* ***Database Type:****The system will use a relational database (SQL Server or MySQL) to store structured data, including product details, transaction records, user information, and audit logs.*
* ***Data Integrity:****The database must enforce data integrity constraints (e.g., primary keys, foreign keys) to prevent duplicate or inconsistent records.*
* ***Backup and Recovery:****Scheduled backups are required to protect against data loss. The database should support incremental and full backups, stored securely with encryption.*
* ***Storage Capacity:****The database must handle a minimum of 100,000 SKUs and at least 500,000 transactions annually, with scalability options to expand storage as needed.*

### **6.2 Internationalization Requirements**

* ***Language Support:****Initially, the application will be in English, but the system should be designed for easy adaptation to other languages in future versions.*
* ***Currency Formatting:****The application should display currency based on the locale, with options for currency symbol placement, decimal notation, and thousand separators.*
* ***Date and Time Formats:****The system must support flexible date and time formats according to regional standards, allowing customization in settings.*

### **6.3 Legal Requirements**

* ***Data Protection Compliance:****The application must comply with local data privacy regulations, such as the GDPR, especially concerning customer data handling, retention, and deletion.*
* ***Tax Compliance:****The system must accommodate regional tax regulations, allowing configuration of tax rates and categories in compliance with legal requirements.*
* ***Sales and Financial Reporting Compliance:****Sales records and financial reports generated by the system must comply with local financial reporting standards to ensure accurate tax and accounting practices.*

### **6.4 Reuse Objectives**

* ***Modular Code Design:****The code should be structured in reusable modules (e.g., inventory management, sales processing, reporting) to facilitate reuse in other retail applications or product variants.*
* ***Standardized Components:****Common interface elements and backend components should follow industry standards to ensure compatibility and reusability across similar systems.*
* ***API Compatibility:****Where applicable, the system should use RESTful APIs for features like payment processing and inventory syncing, making them adaptable to other applications.*

### **6.5 Other Requirements**

* ***Environmental Constraints:****The application must operate reliably in environments such as supermarkets where there may be dust, noise, and heavy foot traffic, with requirements for durable hardware compatibility.*
* ***Maintenance and Support:****The system must include built-in diagnostics for troubleshooting, with logs accessible by IT Support for issue resolution.*
* ***Accessibility:****The user interface should support accessibility standards, with considerations for color contrast, readable fonts, and screen-reader compatibility to aid visually impaired users.*

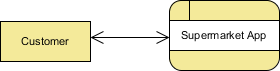
## **Appendix A: Glossary**

* ***Admin:*** *User with full access to all system functions, including inventory management and reporting.*
* ***CRUD:*** *Acronym for Create, Read, Update, Delete operations in databases.*
* ***POS (Point of Sale):*** *The location and system where transactions are processed in a retail environment.*
* ***SKU (Stock Keeping Unit):*** *A unique identifier for each product in the inventory.*
* ***UI (User Interface):*** *The graphical interface that users interact with.*

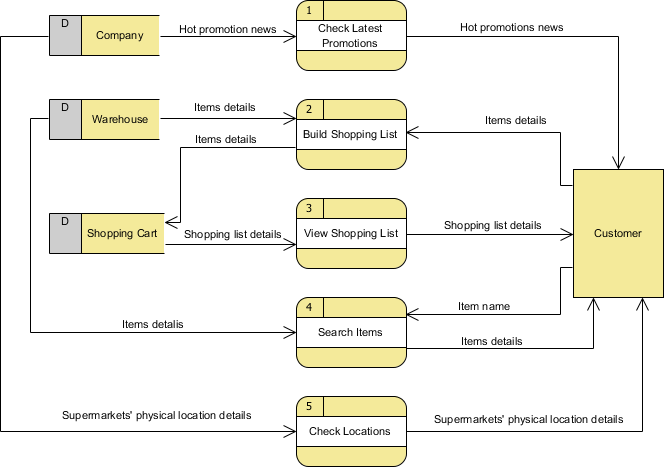
## **Appendix B: Analysis Models**

*The following analysis models provide a visual representation of system components and data flow, aiding developers in understanding the system’s structure and interactions:*

1. ***Data Flow Diagrams (DFD):***
   * ***Level 0 DFD:*** *Provides a high-level overview of the system’s main functions, such as inventory management, sales processing, and reporting.*

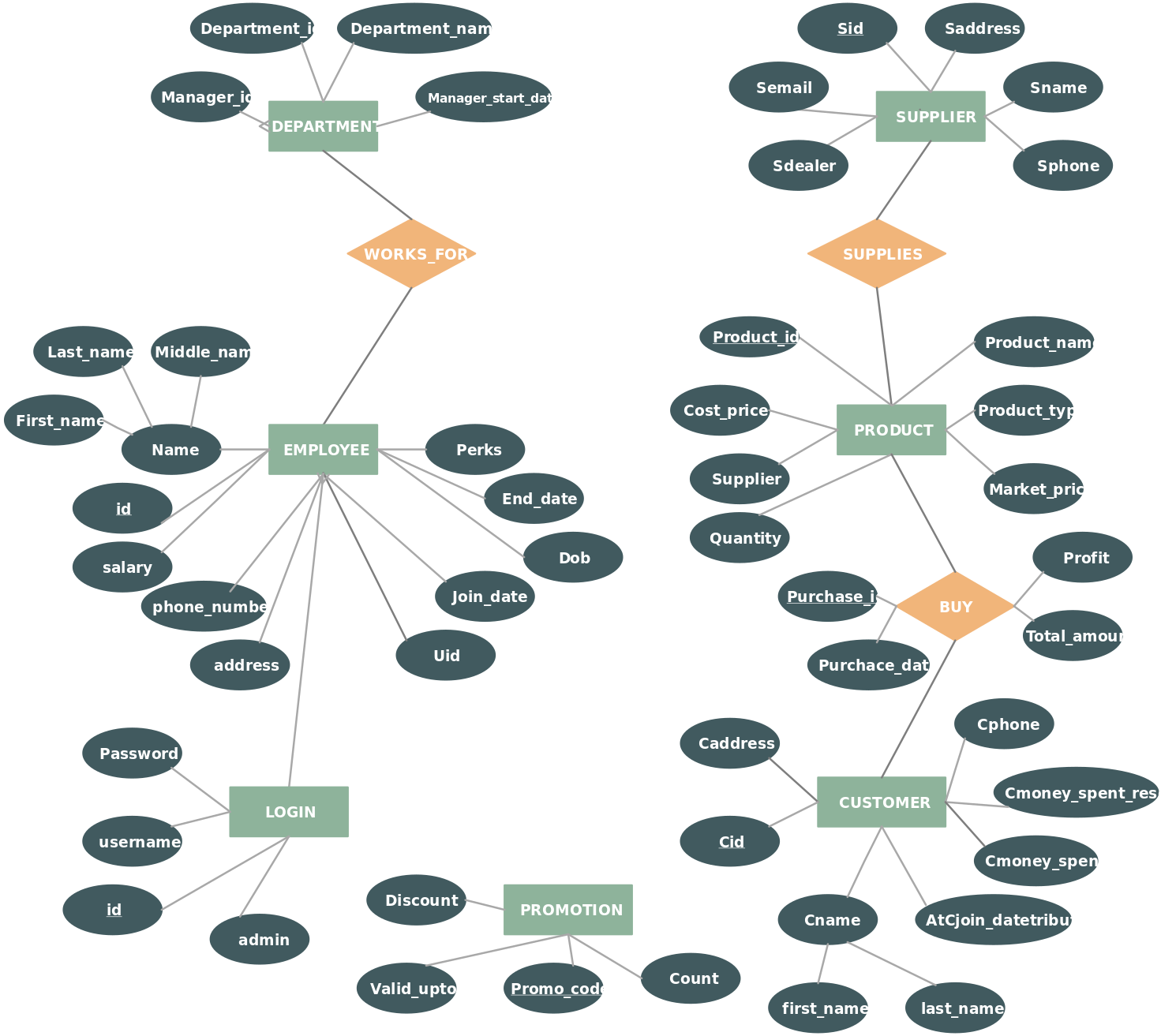
******

* + ***Level 1 DFD:*** *Breaks down the primary processes, showing data flows between user actions (e.g., login, product search) and the database.*

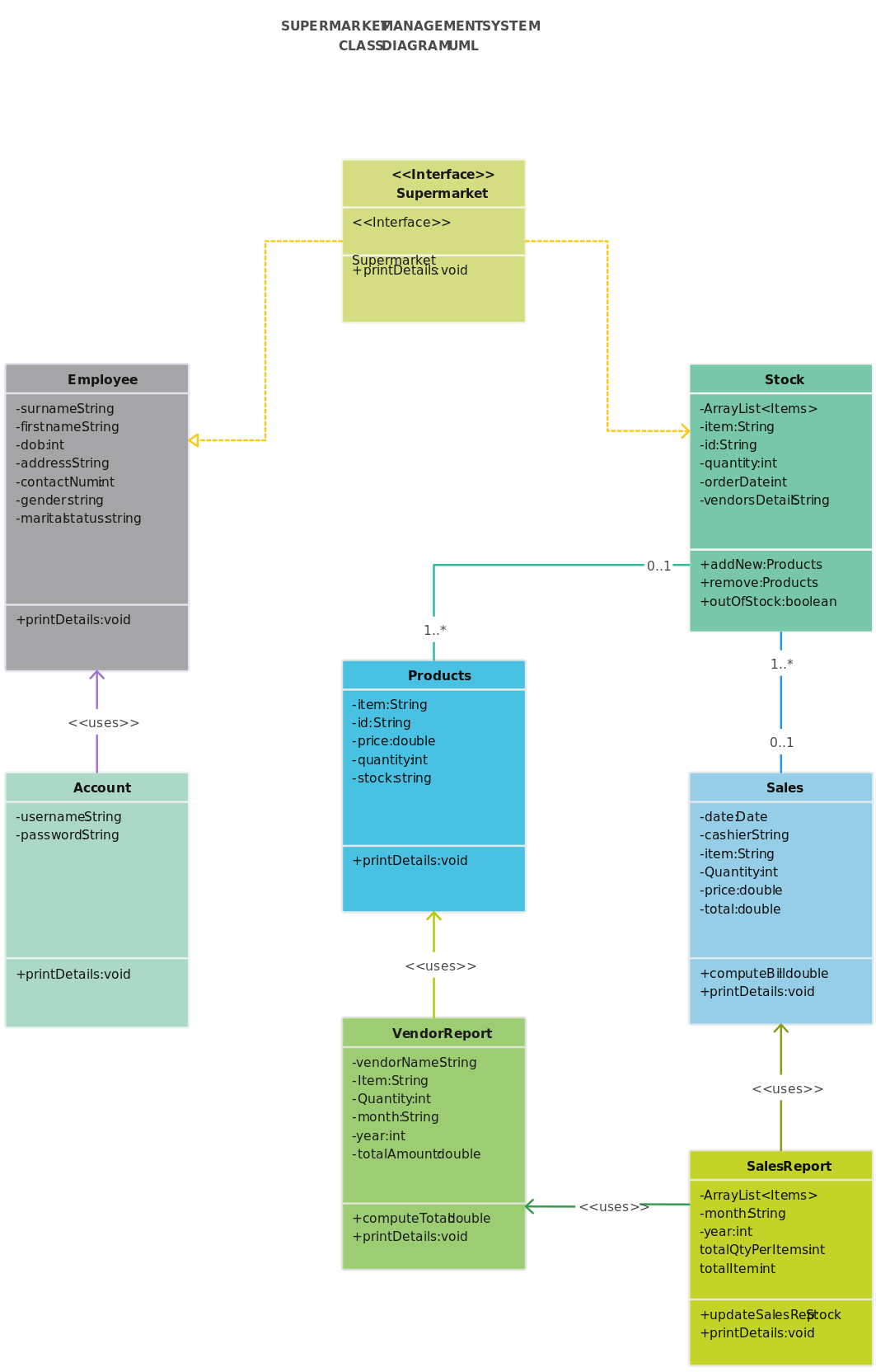
**

* + ***Level 2 DFD:*** *Explores sub-processes within each primary function, like sales transaction processing, which includes product scanning, cart total calculation, and receipt generation.*

1. ***Entity-Relationship Diagram (ERD):***
   * *Illustrates relationships between key entities, such as Products, Orders, Users, and Transactions, defining primary keys, foreign keys, and associations to aid in database design.*

**

1. ***Class Diagrams (if object-oriented):***
   * *Defines the main classes (e.g., Product, Order, User), their attributes, and methods, as well as associations, inheritance, and aggregation relationships.*

**

**Appendix C: Issues List**

This is a dynamic list that captures any unresolved issues, decisions, or pending information related to the requirements.

| **Issue ID** | **Description** | **Status** | **Priority** | **Owner** |
| --- | --- | --- | --- | --- |
| 1 | Finalize the tax calculation formula | Open | High | Ramiz Ghada |
| 2 | Decision on Report generated | Pending Decision | High | Husen Khimani |
| 3 | Confirmation of hardware compatibility | In Progress | Low | Ramiz Ghada |
| 4 | Determine cloud storage provider for backups | Open | Medium | Mohsin Dodhiya |
| 5 | Add support for optional third-party payment gateway | Pending Decision | Low | Mohsin Dodhiya |

**7. Conclusion**

*The development of the FreshMart Supermarket Management desktop application has aimed to deliver a user-friendly, efficient, and cost-effective solution for managing supermarket operations. A conscious effort was made to utilize the best available tools, techniques, and resources to meet the expectations of modern retail management. Although certain limitations exist, such as the need for local server setup and regular database backups, the flexibility and scalability of the system make it adaptable for future growth and improvements. Despite challenges during the development process, the system offers a reliable platform that meets the fundamental needs of a retail management solution.*

**\*\*\*\*\***