UNIVERSITY OF RWANDA

COLLEGE OF BUSINESS AND ECONOMICS

SCHOOL OF BUSINESS

BIT DEPARTMENT

LEVEL 2

GROUP 1

COURSE: SYSTEMS ENGINEERING, DATABASE AND JAVA PROGRAMMING

PROJECT PROPOSAL
ON

SUPERMARKET MANAGEMENT SYSTEM

NAMES: HAGENIMANA Diane
REG N^{0}: 222004455

SUBMITTED TO: Dr BUGINGO Emmanuel

Submission date: 08/march/2024

Table of contents

Contents

List of Figures

List of tables

1. Introduction

The Supermarket Management System is an advanced software solution designed to streamline and automate various operations within a supermarket or grocery store. It leverages technology to enhance efficiency, improve customer service, optimize inventory management, and simplify overall operations.

This system serves as a comprehensive tool that integrates multiple functions and modules, providing a centralized platform to manage various aspects of a supermarket. It typically includes features such as point-of-sale (POS) systems, inventory management, purchasing and procurement, sales analytics, customer relationship management (CRM), employee management, and accounting.

The primary objective of a Supermarket Management System is to enhance the overall shopping experience for customers while facilitating smooth operations for supermarket owners and staff. By automating tasks like inventory tracking, stock replenishment, and sales analysis, it helps ensure that the supermarket operates efficiently, reducing costs and minimizing errors.

2. Statement of the problem

1. Outdated Technology: Many supermarket management systems maybe built on outdated technology or software that lacks the necessary features and capabilities to efficiently handle modern retail operations.

3. Limited Scalability: Some systems may struggle to accommodate the growth and expansion of a supermarket chain. As the number of stores increases, the system may become slow, inefficient, or unable to handle the increased volume of data and transactions.
4. Integration Challenges: Integrating various systems and modules within the supermarket management system, such as POS, inventory management, CRM, and accounting, can be complex and prone to errors. Incompatibility between different systems may result in data discrepancies and hinder the flow of information.
5. Inefficient Inventory Management: Poor inventory management can lead to issues like overstocking, stockouts, and inaccurate stock records. Inadequate tracking of product expiration dates, slow inventory turnover, and difficulties in identifying slow-moving items can impact profitability and customer satisfaction.
6. Inaccurate Sales and Analytics: Insufficient or inaccurate sales data and analytics can hinder decision-making, marketing efforts, and understanding customer preferences. Without reliable insights, supermarkets may struggle to identify trends, optimize pricing strategies, and forecast demand accurately.

7. Security Vulnerabilities: Supermarket management systems often handle sensitive customer data, financial information, and inventory details. Inadequate security measures can expose supermarkets to the risk of data breaches, unauthorized access, and potential financial loss.
8. Lack of Mobility and Remote Access: Traditional systems may lack the ability to provide real-time access and management capabilities remotely. This limitation can hinder the ability to monitor and manage operations effectively, especially for supermarket owners or managers who are not physically present at the store.
9. Complex User Interface and Training: Complex or unintuitive user interfaces can make it challenging for employees to learn and navigate the system effectively. Insufficient training and support can hinder user adoption and result in errors or inefficiencies.
10. To address these problems, it is essential to invest in modern supermarket management systems that leverage advanced technologies, offer seamless integration, provide real-time analytics, enhance security, and support scalability. A user-friendly interface and comprehensive training can ensure smooth adoption and optimal utilization of the system by supermark
11. Integration Challenges: Integrating various systems and modules within the supermarket management system, such as POS, inventory management, CRM, and accounting, can be complex and prone to errors. Incompatibility between different systems may result in data discrepancies and hinder the flow of information.
12. Inefficient Inventory Management: Poor inventory management can lead to issues like overstocking, stockouts, and inaccurate stock records. Inadequate tracking of product expiration dates, slow inventory turnover, and difficulties in identifying slow-moving items can impact profitability and customer satisfaction.
13. Inaccurate Sales and Analytics: Insufficient or inaccurate sales data and analytics can hinder decision-making, marketing efforts, and understanding customer preferences. Without reliable insights, supermarkets may struggle to identify trends, optimize pricing strategies, and forecast demand accurately.
14. Security Vulnerabilities: Supermarket management systems often handle sensitive customer data, financial information, and inventory details. Inadequate security measures can expose supermarkets to the risk of data breaches, unauthorized access, and potential financial loss.
15. Lack of Mobility and Remote Access: Traditional systems may lack the ability to provide real-time access and management capabilities remotely. This limitation can hinder the ability to monitor and manage operations effectively, especially for supermarket owners or managers who are not physically present at the store.
16. Complex User Interface and Training: Complex or unintuitive user interfaces can make it challenging for employees to learn and navigate the system effectively. Insufficient training and support can hinder user adoption and result in errors or inefficiencies.

2.1 Description of the existing system

Point-of-Sale (POS) System: Supermarkets utilize a POS system to handle customer transactions, including scanning products, calculating totals, processing payments, and generating receipts. It may include features like barcode scanning, cash registers, and payment terminals.

Inventory Management: Inventory management involves tracking the stock levels of products, monitoring incoming and outgoing shipments, and ensuring adequate stock availability. This process may include manual stock counting, stockroom organization, and product categorization.

Sales Tracking and Reporting: Sales data is collected and recorded manually or through the POS system. Basic reports on daily, weekly, or monthly sales may be generated, detailing revenue, product categories, and sales volumes. However, the depth of analysis and insights may be limited.

Employee Management: Employee scheduling, attendance tracking, and performance management are typically handled manually. This may involve the use of spreadsheets or basic software tools to record work shifts and calculate payroll.

Purchasing and Replenishment: Supermarket owners or managers manually determine inventory needs and generate purchase orders to replenish stock. This process may involve contacting suppliers, negotiating terms, and arranging delivery.

Customer Relationship Management (CRM): Customer information, such as contact details, loyalty programs, and purchase history, may be maintained manually or using basic CRM tools. However, the system may lack automated marketing campaigns or personalized customer interactions.

Accounting and Finance: Financial management processes, including revenue tracking, expense recording, and financial reporting, are typically performed using basic accounting software or spreadsheets. This may involve manual data entry and calculations.

Security Measures: Supermarkets may employ security measures such as surveillance cameras, security personnel, and anti-theft systems to prevent theft and protect customer and employee safety. However, the level of technological integration for security purposes may vary.

While the existing system provides a foundation for managing supermarket operations, it often relies heavily on manual processes, lacks integration between various modules, and offers limited data analysis capabilities. These limitations can lead to inefficiencies, inaccuracies, and challenges in adapting to changing market demands. Upgrading to a more advanced and comprehensive Supermarket Management System can overcome these limitations and improve overall operational efficiency and customer experience.

2.2 Problems of the current system

Manual Processes: Reliance on manual processes for tasks such as inventory management, sales tracking, and employee scheduling results in increased labor requirements, higher error rates, and slower operations.

Data Inaccuracy and Discrepancies: Manual data entry and lack of real-time synchronization between different modules lead to data inaccuracies, discrepancies, and inconsistencies. This hampers decision-making, reporting, and overall data integrity.

Limited Integration: Lack of integration between different systems and modules, such as POS, inventory management, and accounting, leads to inefficient workflows, duplicated efforts, and difficulty in obtaining a comprehensive view of the supermarket's operations.

Inefficient Inventory Management: The current system may lack robust inventory management capabilities, resulting in issues such as stock outs, overstocking, inaccurate stock records, and difficulties in tracking product expiration dates.

Inadequate Sales Analytics: Limited or outdated reporting and analytics capabilities make it challenging to gain insights into sales performance, customer behavior, and product trends. This hampers strategic decision-making, marketing efforts, and identifying opportunities for growth.

Security Vulnerabilities: Insufficient security measures, both physical and digital, pose risks to the supermarket's data, customer information, and financial transactions. This can result in potential data breaches, theft, fraud, and compromised customer trust.

Lack of Scalability: The current system may lack the flexibility and scalability to accommodate the supermarket's growth, new technologies, and changing business requirements. This limits the supermarket's ability to expand, integrate with emerging solutions, and adapt to market dynamics.

Ineffective Customer Relationship Management: Limited or fragmented customer data, inadequate loyalty programs, and underutilization of CRM tools hinder the supermarket's ability to provide personalized customer experiences, targeted marketing campaigns, and effective customer retention strategies.

Addressing these problems requires implementing a modern Supermarket Management System that automates processes, integrates modules, provides real-time data synchronization, offers advanced analytics capabilities, enhances security measures, and supports scalability. Upgrading to such a system can improve operational efficiency, data accuracy, customer satisfaction, and overall profitability for the supermarket.

2.3 How the proposed system will work

The proposed Supermarket Management System aims to address the problems of the current system and improve overall supermarket operations. Here's an overview of how the proposed system will work:

Integration and Centralized Platform: The proposed system will integrate various modules, including POS, inventory management, purchasing, CRM, employee management, and accounting, into a centralized platform. This ensures seamless data flow and eliminates the need for manual data entry and duplicate efforts.

Point-of-Sale (POS) System: The new system will include an advanced POS system equipped with barcode scanning, payment processing, and receipt generation capabilities. It will streamline customer transactions, provide real-time inventory updates, and enable efficient sales tracking.

Inventory Management: The proposed system will offer robust inventory management features. It will automate stock tracking, provide real-time visibility into inventory levels, and generate automated purchase orders based on predefined reorder points. This ensures optimal inventory levels, reduces stockouts, and minimizes carrying costs.

Sales Analytics and Reporting: The new system will provide comprehensive sales analytics and reporting capabilities. It will generate detailed reports on sales performance, customer buying patterns, popular products, and profitability. These insights will support data-driven decision-making, pricing strategies, and targeted marketing campaigns.

Customer Relationship Management (CRM): The proposed system will incorporate CRM functionality to manage customer information, track customer interactions, and support personalized marketing initiatives. It will enable the implementation of loyalty programs, customer segmentation, and effective customer retention strategies.

Employee Management: The new system will automate employee management processes, including scheduling, attendance tracking, and payroll calculations. It will provide a user-friendly interface for managers to efficiently manage staff, track productivity, and generate accurate payroll reports.

Enhanced Security Measures: The proposed system will prioritize data security by implementing robust security measures. This includes access controls, encryption, secure payment processing, and monitoring systems to prevent data breaches and protect sensitive customer and financial information.

Scalability and Integration: The new system will be designed for scalability, allowing it to accommodate the supermarket's growth, multiple locations, and integration with emerging technologies. It will support seamless integration with external systems like electronic payment gateways, CRM tools, and barcode scanners.

Overall, the proposed Supermarket Management System will automate manual processes, provide real-time data insights, enhance efficiency, optimize inventory management, improve customer service, and ensure data security. It will streamline supermarket operations, enhance decision-making, and deliver an enhanced shopping experience for customers.

16.4 Describe other alternatives

1. Custom-Built Software: Supermarkets may choose to develop their own custom software solution tailored to their unique needs. This option allows for complete control and customization of features, workflows, and integrations. However, it requires significant investment in development resources and ongoing maintenance.

2. Mobile Apps: Supermarkets can leverage mobile applications that provide functionalities such as mobile ordering, loyalty program management, digital coupons, and personalized offers. Mobile apps can enhance the shopping experience for customers, provide real-time updates, and allow for mobile payment options.

3. Integrated Retail Management Systems: Some companies provide comprehensive retail management systems that combine supermarket management with other retail sectors, such as e-commerce, supply chain management, and omnichannel sales. These integrated systems offer end-to-end solutions for managing the entire retail operation.

4. Hybrid Approaches: Supermarkets can also adopt a hybrid approach by combining multiple systems and technologies. This involves integrating different software solutions and hardware components to create a customized ecosystem that meets specific supermarket needs.

   So, the choice of alternative depends on factors such as budget, scalability requirements, customization needs, technological capabilities, and integration with existing systems. Supermarkets should carefully evaluate the available options and select the solution that best aligns with their goals and operational requirements.

17. Software requirement

1. Inventory Management: The software should provide robust inventory management capabilities, including real-time stock tracking, automated stock replenishment, stock movement tracking, and alerts for low stock levels. It should support categorization, product variants, and integration with barcode scanners for efficient stock management.

2. Sales Analytics and Reporting: The system should offer advanced reporting and analytics features to generate comprehensive sales reports, analyze sales performance, identify trends, and provide insights into customer behavior. Customizable reporting options, data visualization, and drill-down capabilities are beneficial for data analysis.

3. Customer Relationship Management (CRM): The software should include CRM functionality to manage customer information, track customer interactions, and support personalized marketing initiatives. It should allow for the creation of customer profiles, loyalty programs, targeted promotions, and customer segmentation.

4. Employee Management: The system should provide features for employee scheduling, attendance tracking, and performance management. It should allow managers to easily assign shifts, track employee attendance, calculate payroll, and generate reports for payroll processing.

5. Security Measures: The system should prioritize data security by implementing robust security measures, including user access controls, data encryption, regular backups, and compliance with data protection regulations. It should also provide secure payment processing options to protect customer financial information.

3.1 User Requirements

1. Point-of-Sale (POS) Functionality:

Fast and reliable transaction processing

Support for multiple payment methods

Barcode scanning and receipt generation

Integration with electronic payment gateways.

2. Inventory Management:

Real-time tracking of stock levels

Automated stock replenishment based on predefined thresholds

Product categorization and organization

Alerts for low stock, expired items, or overstocking

3. Customer Relationship Management (CRM):

Customer information management and tracking

Loyalty program management

Personalized marketing initiatives and targeted promotions

Customer segmentation and behavior analysis

4. Employee Management:

Employee scheduling and shift management

Attendance tracking and timekeeping

Payroll calculation and reporting

Communication and collaboration features for staff

5. Integration and Scalability:

Integration with external systems like accounting software, CRM tools, and electronic payment gateways

Scalability to support multiple supermarket locations and future growth

Flexibility to adapt to changing business needs and workflows

6. Security and Data Protection:

User access controls and permissions

Data encryption and secure storage

Regular backups and disaster recovery measures

Compliance with data protection regulations

7. Training and Support:

Comprehensive training materials and resources

Ongoing technical support and assistance

User documentation and help guides

. Reporting and Compliance:

Generation of financial reports, sales tax reports, and audit trails

Compliance with regulatory requirements and industry standards

To gather user requirements effectively, it is important to engage with supermarket owners, managers, cashiers, inventory controllers, and other relevant staff members. Conducting interviews, workshops, and surveys can help elicit their needs and expectations, ensuring that the Supermarket Management System project meets their specific requirements and contributes to improved efficiency and effectiveness in supermarket management.

3.1.1 Functional requirement

1. Inventory Management

Track and manage stock levels of products.

Support barcode scanning and automatic updating of inventory.

Provide alerts for low stock levels and automatic reordering.

Handle product categorization and classification.

Allow for stocktaking and reconciliation.

2. Customer Management:

Maintain a database of customer information.

Manage loyalty programs and customer rewards.

Capture and analyze customer purchase history.

Provide customer registration and login functionality.

Generate customer reports and analytics.

3. Product Pricing and Promotions:

Set and manage pricing for products.

Support dynamic pricing and discounts.

Create and manage promotional campaigns.

Schedule and automate promotions.

Handle price adjustments and special pricing for specific customers.

4. Supplier Management:

Maintain a database of supplier information.

Manage supplier contracts and agreements.

Track deliveries and handle purchase orders.

Generate reports on supplier performance and product quality.

Support communication and collaboration with suppliers.

5. Employee Management:

Manage employee profiles and roles.

Handle employee scheduling and shift management.
Track attendance and hours worked.
Calculate and manage payroll.
Provide secure access controls and permissions.

6. Online Ordering and Delivery:

Enable online ordering and delivery services.

Support integration with e-commerce platforms.

Manage delivery schedules and logistics.

Track order status and provide real-time updates.

Handle customer feedback and ratings.

3.1.2 Non-function requirement

Portability requirement

refers to the ability of the system to be easily transferred and deployed across different environments and platforms. It ensures that the system can be installed, executed, and maintained in various hardware and software configurations without significant modifications or dependencies.

Reliability requirement

The system should have a high level of availability, minimizing downtime and ensuring continuous operation during peak hours.

It should be capable of handling system failures, such as network disruptions or power outages, with minimal impact on ongoing transactions.

Usability requirement

The user interface should be intuitive, easy to navigate, and require minimal training for store employees to use efficiently.

The system should provide clear and concise error messages to assist users in resolving issues.

Accessibility features should be implemented to accommodate users with disabilities.

The system should support multiple languages if the supermarket operates in a multicultural environment.

Space requirement

Refers to the physical area or floor space needed to accommodate the various components and functionalities of the system.
Storage Areas: Depending on the size of the supermarket, storage space is required for backroom inventory storage, including perishable and non-perishable goods. This space should be organized and accessible for efficient stock management.

Equipment and Machinery: Allocate space for equipment like refrigeration units, freezers, industrial ovens, and other machinery required for stocking, storing, and preparing food products.

Staff Amenities: Supermarkets often provide dedicated spaces for staff break rooms, restrooms, lockers, and other amenities. Ensure adequate space is allocated to cater to the needs of the store's employees

Customer Amenities: Plan for additional space for customer amenities such as restrooms, seating areas, shopping carts, and baskets that help the customer to comfort.

Organization requirement

Implementation requirement

Refer to the necessary steps, components, and considerations involved in deploying and integrating the system into the supermarket's operations. The following are some implementation requirements for supermarkets.
Software Development: Develop or acquire the software application that will serve as the core of the supermarket management system. This includes designing and building features such as inventory management, sales tracking, customer relationship management, pricing and promotions, and reporting functionalities.

Hardware Infrastructure: Identify the hardware components required to support the system, such as servers, computers, barcode scanners, receipt printers, cash registers, and other devices. Ensure compatibility and scalability to meet the supermarket's current and future needs.

Network Setup: Establish a reliable network infrastructure to connect all the system components. This includes configuring local area networks (LANs), internet connectivity, wireless access points, and data security measures to protect sensitive information.

User Training: Provide comprehensive training sessions to educate supermarket staff on how to effectively use the management system. This includes training cashiers, inventory managers, customer service representatives, and other relevant personnel on system functionalities, data entry, reporting.

Testing and Quality Assurance: Conduct thorough testing to ensure the system functions correctly, meets performance expectations, and integrates seamlessly with other systems. Identify and fix any bugs or issues before the system goes live.

Change Management: Develop a change management plan to smoothly transition employees and processes to the new system. Communicate the benefits and changes associated with the implementation, address concerns, and provide ongoing support to facilitate a successful adoption.

External environment requirement

refer to the external factors and considerations that may impact the implementation, operation, and success of the system within the broader environment.

Regulatory Compliance: Ensure that the supermarket management system adheres to relevant laws, regulations, and industry standards. This includes compliance with data protection and privacy laws, food safety regulations, pricing regulations, and any other applicable legal requirements.

Integration with Suppliers and Vendors: Consider the need for integration with suppliers and vendors to facilitate seamless ordering, inventory management, and procurement processes. This may involve establishing electronic data interchange (EDI) connections or using standardized protocols to exchange data with external partners.

Payment Processors and Financial Institutions: Integrate the supermarket management system with payment processors and financial institutions to enable secure and efficient payment transactions. This may involve working with banks, credit card processors, and mobile payment providers to ensure smooth payment processing and reconciliation.

Customer Relationship Management: Consider the need to integrate the supermarket management system with customer relationship management (CRM) tools or loyalty program platforms. This integration can help manage customer data, track customer preferences, and provide personalized shopping experiences.

E-commerce and Online Marketplaces: If the supermarket has an online presence or participates in online marketplaces, ensure that the management system can integrate with e-commerce platforms. This integration facilitates inventory synchronization, order processing, and seamless customer experiences across online and offline channels.

Technology and Infrastructure: Consider the technological infrastructure available in the external environment. This includes factors such as internet connectivity, mobile network coverage, availability of hardware components, and compatibility with existing technologies used by customers and partners.

Competitive Landscape: Analyze the competitive landscape of the supermarket industry in the specific market where the system will be deployed. Understand the capabilities and features of competing supermarkets' management systems to ensure that the implemented system provides competitive advantages and meets customer expectations.

Customer Expectations and Trends: Stay updated on evolving customer expectations and industry trends in supermarket management systems. This includes factors such as increasing demand for personalized experiences, online ordering and delivery options, mobile app integration, and sustainability initiatives. Align the system's features and functionalities with these customer expectations to remain competitive.

Scalability and Growth Potential: Consider the scalability of the management system to accommodate potential future growth of the supermarket. Ensure that the system can handle increased transaction volumes, additional store locations, and potential expansions into new markets.

Social and Environmental Responsibility: Factor in the growing emphasis on social and environmental responsibility in the external environment. Consider integrating features that support sustainability initiatives, waste reduction, responsible sourcing, and community engagement, aligning with the supermarket's corporate social responsibility goals.

By considering these external environment requirements, the supermarket management system project can be designed and implemented to effectively operate within its broader context, meet customer expectations, comply with regulations, and leverage external opportunities for success.

Privacy requirement

Involve ensuring the protection and responsible handling of customer and employee data.

Data Protection: Implement measures to protect sensitive information collected by the system, such as customer names, addresses, contact details, payment information, and purchase history. Utilize encryption, secure transmission protocols, and access controls to safeguard data against unauthorized access or disclosure.

Data Minimization: Only collect and store the necessary data required for supermarket operations and customer service. Minimize the retention period of personal data to the shortest time necessary and regularly review and delete outdated or unnecessary data.

Regular Audits and Assessments: Conduct periodic privacy audits and assessments to evaluate the effectiveness of privacy controls and procedures. Regularly review and update privacy policies and practices to align with evolving privacy regulations and industry standards.

Third-Party Data Processors: If the supermarket management system utilizes third-party vendors or service providers, ensure they adhere to appropriate privacy and security standards.

Access Controls and User Permissions: Implement strict access controls to ensure that only authorized personnel have access to customer data. Use role-based access controls (RBAC) to limit data access based on job responsibilities and enforce strong password policies.

Safety requirement

involve ensuring the security and safety of both customers and employees within the supermarket environment.

Data Security: Implement robust data security measures to protect sensitive customer information, employee data, and other confidential data stored within the management system.

System Reliability: Design and develop the management system to ensure high availability and reliability. Implement appropriate backup and disaster recovery mechanisms to minimize system downtime and ensure continuous operation, especially during critical periods such as peak shopping hours.

User Authentication and Authorization: Implement strong user authentication mechanisms to ensure that only authorized personnel can access and operate the management system. Use secure login methods, such as passwords, and enforce strict user access controls based on roles and responsibilities.

Physical Security: Consider physical security measures to protect the hardware infrastructure and sensitive areas within the supermarket. This may include surveillance cameras, access control systems, alarms, and physical barriers to prevent unauthorized access.

Emergency Preparedness: Develop and implement emergency preparedness plans, including evacuation procedures, in the event of fire, natural disasters, or other emergencies. Ensure that the management system can support emergency notifications and communication to quickly alert employees and customers during critical situations.

Equipment Safety: Ensure that any equipment, such as cash registers, barcode scanners, or other devices integrated with the management system, meet safety standards and do not pose any hazards to users. Regularly inspect and maintain the equipment to ensure safe operation.

Network Security: Implement robust network security measures to protect the supermarket management system from external threats. This includes using firewalls, intrusion detection systems, and antivirus software to detect and mitigate potential security breaches

User Training: Provide comprehensive training to supermarket employees on the safe and secure operation of the management system. This includes educating them about potential security risks, data handling practices, and emergency procedures. Regularly update training materials to address new security threats or system updates.

Incident Response: Develop an incident response plan to address any safety or security incidents related to the management system. This plan should outline procedures for reporting incidents, investigating security breaches, and implementing corrective actions to prevent future occurrences.

3.2 System requirements

3.2.1 Minimum End-user Hardware Requirements

User-Friendly Interface: The system should have an intuitive and user-friendly interface that allows supermarket staff to navigate and perform tasks efficiently. The interface should be visually appealing, organized, and provide clear instructions and feedback to users.

Inventory Management: The system should provide inventory management capabilities, allowing users to track stock levels, manage product categories, receive and manage supplier orders, conduct stocktaking, and generate reports on inventory status.

Sales Reporting and Analytics: The system should offer reporting and analytics features that enable users to generate sales reports, analyze sales trends, monitor performance, and make data-driven decisions.

Multi-Platform Access: Consider the ability for users to access the system from multiple platforms, such as desktop computers, tablets, or mobile devices. This can enhance flexibility and allow staff to perform tasks on the go or from different locations within the supermarket.

Training and Support: Provide comprehensive training materials, user manuals, and ongoing support to ensure that end users are comfortable and proficient in using the system. This can include training sessions, user guides ,etc…

3.2.2 Minimum End-user Software Requirements

This deals on the software and hardware specifications necessary to run the system effectively.

Operating System: Identify the supported operating systems for running the supermarket management system. Common options include Windows, macOS, and Linux distributions. Specify the minimum version requirements for each supported operating system.

Web Browser Compatibility: If the management system has a web-based interface, ensure compatibility with popular web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari. Specify the minimum browser versions supported by the system.

Mobile Device Compatibility: If the system includes mobile applications for end users, specify the supported mobile platforms, such as iOS and Android. Provide the minimum supported operating system versions for each platform.

System Resources: Determine the minimum system resource requirements, such as CPU, RAM, and storage space, necessary to run the management system smoothly. Specify the minimum requirements based on the complexity and scale of the system

.

Database Requirements: Identify the supported database management systems (DBMS) for the system. Specify the required version and configuration details for the DBMS, such as MySQL, PostgreSQL, Oracle, or Microsoft SQL Server.

Networking Requirements: Specify the networking requirements for the system, such as internet connectivity, LAN connectivity, and network protocols. This includes specifying the minimum network bandwidth required for efficient system operation.

Integration with Peripheral Devices: Identify any peripheral devices, such as barcode scanners, receipt printers, or cash registers, that need to be integrated with the supermarket management system. Specify the compatibility requirements and any additional software drivers or APIs needed for proper integration.

User Account Management: Specify the requirements for user account management, such as username and password policies, password complexity, and account expiration policies. Ensure that proper security measures are in place to protect user accounts and sensitive data.

3.3 Software specifications

System Overview: Provide an overview of the software, its purpose, and its intended users. Describe the primary goals and objectives of the system and its intended benefits for the supermarket.

Functional Requirements: Specify the system's functional requirements by describing the core features and functionalities it should provide. This may include point of sale (POS), inventory management, sales reporting, customer management, pricing, promotions, and other relevant modules.

User Roles and Permissions: Define the different user roles within the system and their associated permissions. Specify the actions and functionalities each role can access to ensure appropriate access controls and data security.

User Interface: Describe the user interface design guidelines, including the layout, navigation, and visual elements. Specify any branding or style guidelines to maintain consistency with the supermarket's overall branding.

Data Management: Outline the data management requirements, including how customer data, product information, and inventory data will be stored, retrieved, and updated within the system. Specify any integration needs with external systems or databases.

18. DATA FLOW DIAGRAM (LEVEL 0, LEVEL 1**)**

4.1 DFD LEVEL 0

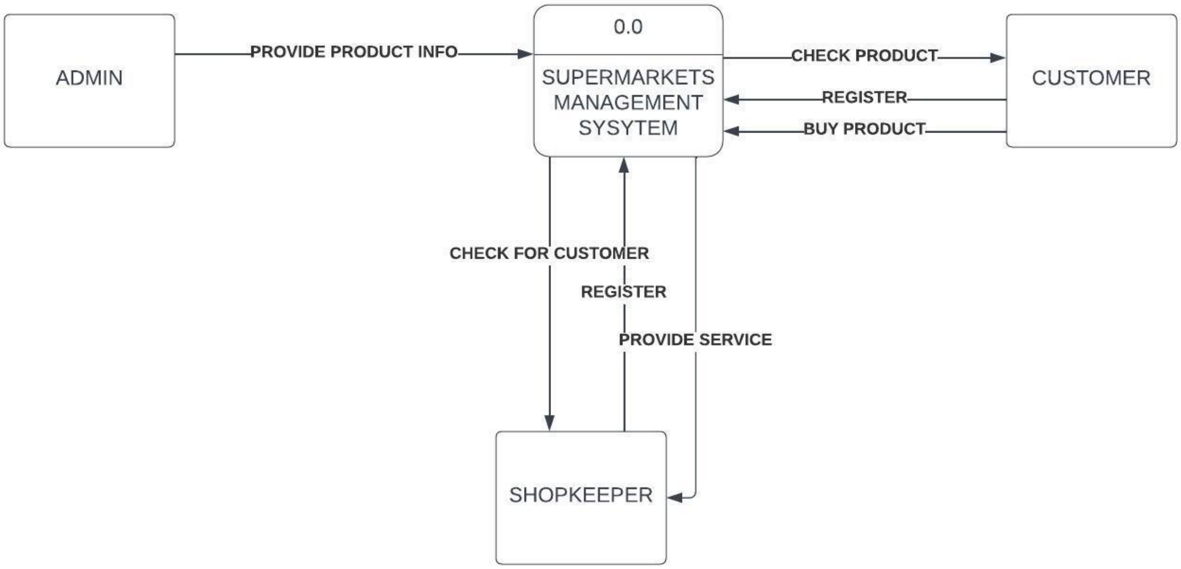


Figure 1

This figure defines the boundary between the system ,or part of a system, and its environment, showing the entities that interact with it, this diagram is a high level view of a system.

4.2 DFD LEVEL 1

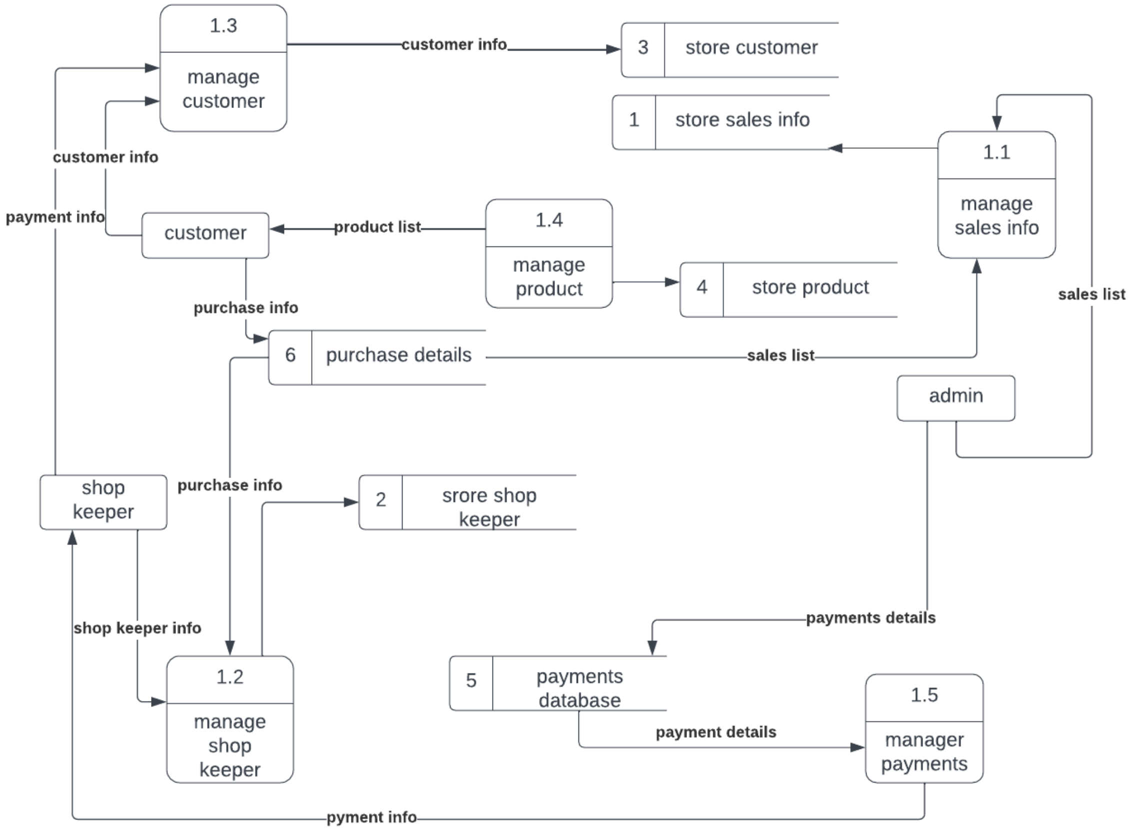


Figure 2

This diagram shows the whole system is represented as a single process. A level 1 DFD notates each of the main sub-processes that together form the complete system.

5. Feasibility study

5.1 Technical Feasibility

Hardware and Infrastructure: Evaluate the supermarket's existing hardware infrastructure to ensure it can support the proposed management system. Consider factors such as server capacity, network bandwidth, storage capacity, and compatibility with the required peripherals and devices.

Software and Technology: Assess the compatibility of the proposed system with the supermarket's existing software ecosystem. Determine if the required software components, databases, operating systems, programming languages, and development tools are compatible and readily available.

Integration with Existing Systems: Identify any existing systems or software applications used in the supermarket that need to be integrated with the management system. Evaluate the feasibility of integrating data, functionality, and communication between the systems, ensuring compatibility and seamless information flow.

Scalability and Performance: Consider the scalability requirements of the management system to accommodate future growth, increased transaction volumes, and additional store locations. Assess whether the proposed technical infrastructure can handle anticipated growth and deliver the expected performance levels.

Data Management and Security: Evaluate the technical feasibility of securely managing and storing sensitive data, such as customer information, transaction records, and inventory data. Ensure that proper data backup, security measures, and data privacy regulations can be implemented.

Technical Expertise and Resources: Assess the availability of technical expertise within the supermarket or the ability to acquire and manage external resources if needed. Evaluate whether the required skills, such as software development, system administration, and technical support, are available to successfully implement and maintain the system.

Cost and Return on Investment (ROI): Consider the financial feasibility of the technical requirements. Evaluate the costs associated with hardware upgrades, software licenses, system maintenance, and ongoing technical support. Assess the potential return on investment based on increased efficiency, improved operations, and customer satisfaction.

Time Constraints: Assess whether the proposed system can be developed, implemented, and deployed within the desired timeframe. Consider factors such as system complexity, resource availability, and any dependencies or constraints that may impact the project timeline.

5.2 Financial Feasibility

Cost Estimation: Determine the total cost of developing and implementing the supermarket management system. This includes costs associated with software development, hardware infrastructure, licensing fees, data migration, training, and project management.

Return on Investment (ROI): Assess the potential benefits and returns that the supermarket management system can deliver. Consider factors such as increased operational efficiency, improved inventory management, reduced manual errors, enhanced customer experience, and cost savings over the long term. Calculate the estimated ROI based on these factors.

Cost Reduction and Savings: Identify potential cost reductions and savings that the management system can bring to the supermarket. For example, automation of inventory management can reduce overstocking or stockouts, resulting in cost savings. The system may also streamline processes and reduce labor costs by optimizing staff allocation and reducing manual data entry.

Increased Revenue and Sales: Evaluate the potential impact of the management system on revenue generation and sales growth. Consider how improved inventory management, targeted promotions, and better customer insights can contribute to increased sales and profitability.

Cost of Maintenance and Support: Estimate the ongoing costs associated with maintaining and supporting the system after implementation. This includes software updates, technical support, hardware maintenance, and any licensing or subscription fees.

Payback Period: Determine the time it will take for the supermarket to recover the initial investment in the management system. Calculate the payback period by considering the initial costs and the expected savings or increased revenue generated over time.

Cash Flow Analysis: Conduct a cash flow analysis to assess the inflows and outflows of funds throughout the project's lifecycle. Consider the timing of costs, revenue streams, and any financing options. Analyze the net cash flow to ensure the project remains financially viable.

Financing Options: Explore different financing options available to the supermarket for the implementation of the management system. This may include self-funding, loans, leasing options, or partnerships. Assess the financial implications and risks associated with each option.

Risk Assessment: Identify and evaluate the financial risks associated with the project, such as cost overruns, changes in technology, market dynamics, or unexpected challenges during implementation. Develop risk mitigation strategies to minimize the financial impact of these risks.

Cost-Benefit Analysis: Conduct a comprehensive cost-benefit analysis by comparing the estimated costs and benefits of the management system. Evaluate the financial impact of the project by quantifying both tangible and intangible benefits against the costs incurred.

By conducting a thorough financial feasibility assessment, stakeholders can determine the economic viability of implementing the supermarket management system. This assessment helps in making informed decisions, securing necessary funding, and ensuring the long-term financial sustainability of the project.

5.3 Market Feasibility

Market Analysis: Conduct market research to understand the current market trends, customer preferences, and industry dynamics. Identify the target market segment for the supermarket management system, such as small, medium, or large supermarkets, and assess the market size and potential demand.

Customer Needs and Pain Points: Identify the specific needs, challenges, and pain points faced by supermarkets in managing their operations. Understand the areas where the system can provide value, such as improving inventory management, streamlining checkout processes, enhancing customer service, or providing data analytics for decision-making.

Competitive Analysis: Analyze the competitive landscape to identify existing solutions available in the market. Assess their features, strengths, weaknesses, pricing, and customer satisfaction levels. Identify opportunities for differentiation and how the proposed management system can provide unique value compared to competitors.

Unique Selling Proposition (USP): Define the unique selling proposition of the supermarket management system. Determine how it stands out from existing solutions in the market, whether through advanced features, user-friendly interfaces, integration capabilities, scalability, or cost-effectiveness.

Target Market Segmentation: Define the specific target market segments for the management system. Consider factors such as supermarket size, industry verticals, geographical locations, and customer preferences. Tailor the system's features and marketing strategies to address the unique needs of each segment.

Market Potential and Demand: Estimate the potential market size and demand for the supermarket management system. Consider factors such as the number of supermarkets in the target market, market growth rates, and potential adoption rates based on customer needs and willingness to invest in such a system.

Pricing Strategy: Determine the appropriate pricing strategy for the management system. Consider factors such as the value provided, competitive pricing, pricing models (e.g., subscription-based, one-time licensing), and the pricing preferences of the target market.

Marketing and Distribution Channels: Develop a marketing strategy to promote the supermarket management system. Identify the most effective marketing channels to reach the target market, such as trade shows, digital marketing, industry publications, or partnerships with industry influencers. Determine the appropriate distribution channels for delivering and supporting the system.

Regulatory and Compliance Factors: Consider any industry-specific regulations or compliance requirements that may impact the adoption or implementation of the system in the target market. Ensure that the system meets relevant standards and regulatory obligations, such as data privacy or security regulations.

Customer Feedback and Validation: Gather feedback and validation from potential customers or stakeholders in the supermarket industry. Conduct surveys, interviews, or pilot tests to assess the level of interest, feedback on system features, and willingness to adopt the system.

By conducting a thorough market feasibility assessment, stakeholders can determine if there is a viable market for the supermarket management system project. This assessment helps in understanding customer needs, positioning the system effectively, and developing a targeted marketing and sales strategy to capture the market share.

.

5.4 Economic Feasibility

Economic feasibility is an evaluation of the financial viability and profitability of a project or system. When assessing the economic feasibility of a supermarket management system, several factors need to be considered.

1. Cost-Benefit Analysis: Determine the costs associated with implementing and maintaining the supermarket management system, including software licenses, hardware, training, and ongoing support.
2. Return on Investment (ROI): Calculate the projected ROI by estimating the financial gains or savings resulting from the implementation of the system. Consider factors like increased sales, reduced stockouts, optimized inventory levels, improved employee productivity, and potential cost savings in areas like labor and accounting.
3. Payback Period: Determine how long it will take to recover the initial investment made in the supermarket management system.
4. Operational Cost Savings: Evaluate the potential reduction in operational costs due to automation and streamlining of processes. For example, the system may minimize manual errors, optimize inventory levels, reduce wastage, and enhance resource allocation, resulting in cost savings over time.
5. Revenue Generation: Assess the system's potential to generate additional revenue. This can include features such as customer loyalty programs, targeted marketing campaigns, and improved sales tracking and analysis, which can lead to increased customer satisfaction and loyalty.
6. Scalability and Growth: Consider the system's ability to accommodate the supermarket's growth and expansion plans. A scalable system allows for the addition of new stores, increased transaction volumes, and the integration of new technologies, ensuring long-term economic feasibility.
7. Risk Assessment: Identify and evaluate potential risks that may impact the economic feasibility of the supermarket management system.
8. Timeframe for Implementation: Consider the time and resources required for implementing the system. Delays or excessive implementation costs can affect the economic feasibility of the project.
   To sum up, the stakeholders can make an informed decision regarding the economic feasibility of implementing a supermarket management system. It is essential to consider both short-term benefits and long-term value to determine whether the investment aligns with the supermarket's financial goals and objectives.

5.5 Legal and Regulatory Feasibility

Compliance with Data Protection and Privacy Laws: Evaluate whether the system adheres to data protection and privacy laws, such as the General Data Protection Regulation (GDPR) or other local regulations. Ensure that customer information, including personal data, is handled securely and with proper consent.

Payment Card Industry Data Security Standard (PCI DSS) Compliance: If the system handles credit card transactions, it must comply with PCI DSS requirements to ensure the security of cardholder data. Verify that the system encrypts sensitive payment information and follows industry best practices for secure payment processing.

Intellectual Property Rights: Assess whether the system and its components, including software, have appropriate licenses or permissions. Avoid infringing on intellectual property rights by using licensed software or ensuring that custom-developed software complies with copyright laws.

Employment Laws and Labor Regulations: Consider employment laws and labor regulations relevant to the implementation of the system. Ensure compliance with regulations related to working hours, wages, benefits, and employee rights. This includes adherence to equal employment opportunity laws and fair labor practices.

Consumer Protection Laws: Verify that the system complies with consumer protection laws, such as accurate pricing, clear product labeling, and fair business practices. Ensure that any loyalty programs or promotions offered through the system adhere to applicable regulations.

Health and Safety Regulations: Assess whether the system and its operations comply with health and safety regulations. This includes considerations such as proper storage and handling of perishable goods, compliance with food safety standards, and employee safety protocols.

Local and National Regulations: Evaluate compliance with specific local or national regulations relevant to the supermarket industry. This can include licensing requirements, zoning regulations, waste management regulations, and other applicable laws.

Accessibility Compliance: Ensure that the system and any associated customer interfaces are accessible to individuals with disabilities, complying with accessibility standards and guidelines such as the Web Content Accessibility Guidelines (WCAG).

Legal Agreements and Contracts: Review any legal agreements or contracts associated with the system, such as software licenses, service level agreements, or vendor contracts. Ensure that the terms and conditions are in compliance with relevant laws and regulations and protect the supermarket's interests.

It is crucial to involve legal experts or consultants specializing in relevant areas to assess the legal and regulatory feasibility of implementing a supermarket management system. Compliance with applicable laws and regulations mitigates legal risks, protects the supermarket's reputation, and ensures a smooth and lawful operation of the system.

5.6 Operational Feasibility

This refers to the assessment of whether a proposed project or system can be effectively implemented and integrated into the existing operations of an organization.

Compatibility with Existing Systems and Infrastructure: Assess whether the system can seamlessly integrate with the supermarket's current infrastructure, including hardware, software, and network systems.

User Acceptance and Training: Evaluate the system's user-friendliness and whether it meets the needs of various stakeholders, such as cashiers, store managers, inventory personnel, and customer service representatives. Consider the system's ease of use, intuitiveness, and the training required for employees to effectively utilize and navigate the system.

Scalability and Growth: Consider whether the system can accommodate the supermarket's growth plans and handle increased transaction volumes. Assess its ability to handle multiple store locations, support expanding product catalogs, and integrate additional features or functionalities as the business evolves.

Resource Availability: Evaluate the availability of resources, such as hardware, software, and IT personnel, required to implement and maintain the system effectively. Assess whether the supermarket has the necessary resources and infrastructure to support the system and meet ongoing maintenance requirements.

Change Management: Assess the organization's readiness and ability to manage the changes associated with implementing the system. Determine whether the necessary change management processes, communication strategies, and employee support mechanisms are in place to ensure a smooth transition.

Risk Assessment: Identify and assess potential risks associated with implementing the system, such as system downtime, data security vulnerabilities, or operational disruptions. Develop risk mitigation strategies and contingency plans to address these potential risks.

Stakeholder Engagement: Involve key stakeholders, including employees, managers, and customers, in the evaluation and decision-making process. Seek their input and feedback to ensure that the system meets their needs and expectation.

6. Description of the selected Process Mode

The selected development model for a supermarket management system can vary depending on the specific requirements, project scope, and preferences of the development team. One commonly used development model is the "Agile" methodology.

The Agile development model is an iterative and flexible approach that emphasizes collaboration, adaptability, and customer involvement throughout the development process. It enables the development team to respond quickly to changing requirements and deliver incremental functionality in short development cycles called "sprints."

Here is a more detailed description of the Agile model:

Iterative and Incremental Development: Agile follows an iterative and incremental approach. Instead of developing the entire software solution in one go, it divides the development process into smaller iterations or timeboxed cycles called sprints. Each sprint typically lasts from one to four weeks and results in a potentially shippable product increment.

Customer Collaboration: Agile emphasizes collaboration between the development team and the customer or end-users. Customers are involved in the process from the beginning and provide continuous feedback, ensuring that the developed solution meets their needs and expectations. This collaboration fosters a customer-centric approach to development.

Adaptive Planning: Agile recognizes that requirements can change during the development process. It promotes adaptive planning, where requirements are not fully defined upfront but evolve and are refined as the project progresses. The development team collaboratively works with the customer to prioritize requirements and make adjustments based on feedback and changing business needs.

Cross-Functional Teams: Agile encourages the formation of cross-functional teams that comprise members with diverse skills, including developers, testers, designers, and business analysts. These teams work together closely, fostering collaboration, shared responsibility, and effective communication.

Continuous Delivery: Agile aims to deliver working software increments at the end of each sprint. This allows stakeholders to see tangible progress, provide feedback, and make necessary adjustments early in the development process. Continuous delivery enables faster time-to-market and facilitates the early identification of issues or areas for improvement.

Emphasis on Communication: Agile places a strong emphasis on communication within the development team and with stakeholders. Regular meetings, such as daily stand-ups, sprint planning, and retrospective sessions, ensure effective communication, address challenges, and foster transparency.

Continuous Improvement: Agile promotes a culture of continuous improvement. The development team conducts regular retrospectives at the end of each sprint to reflect on successes, challenges, and areas for improvement. Lessons learned from each iteration are used to enhance future sprints and refine the development process.

The Agile model offers numerous benefits, including increased customer satisfaction, faster time-to-market, early and continuous delivery of value, adaptability to changing requirements, and improved team collaboration. It is particularly well-suited for projects where requirements are not fully known upfront or are subject to change, allowing for greater flexibility and responsiveness throughout the development lifecycle.

6.1 Cause of your selection

Changing Business Requirements: Traditional software development models, such as the Waterfall model, assume that requirements are fully understood and fixed upfront. However, in practice, business requirements often change or evolve during the development process. The Agile model addresses this challenge by embracing change and allowing for flexibility in requirements.

Customer-Centric Approach: The Agile model recognizes the importance of actively involving customers or end-users throughout the development process. Traditional models often have limited customer involvement, resulting in a disconnect between the developed solution and the actual needs of the users. Agile promotes collaboration and frequent customer feedback to ensure the software solution aligns with customer expectations.

Increasing Complexity and Uncertainty: Modern software projects often involve complex systems, evolving technologies, and uncertain market conditions. Traditional models with rigid planning and documentation may struggle to cope with such complexity and uncertainty. Agile provides a more adaptive and iterative approach that allows for incremental development and quick adaptation to changing circumstances.

Desire for Faster Time-to-Market: In today's competitive market, businesses often seek to deliver software solutions quickly to gain a competitive edge. Traditional models may have longer development cycles, leading to delayed time-to-market. Agile's iterative and incremental approach enables faster delivery of working software increments, allowing businesses to respond more quickly to market demands.

Team Collaboration and Communication: Traditional models often have a hierarchical structure and limited communication channels between team members. Agile promotes cross-functional teams, collaboration, and effective communication among team members. This collaborative environment fosters better understanding, knowledge sharing, and collective ownership of the project's success.

Empirical and Data-Driven Decision Making: Agile emphasizes the use of empirical data and feedback to guide decision making. It encourages frequent inspection and adaptation based on real-time information. This empirical approach enables teams to make informed decisions, identify potential risks or issues early on, and adjust their strategies accordingly.

Continuous Improvement and Learning: Agile promotes a culture of continuous improvement and learning. It encourages teams to reflect on their work, identify areas for improvement, and implement changes iteratively. By embracing a mindset of continuous learning and adaptation, Agile teams can enhance their effectiveness and deliver higher-quality software.

Industry Best Practices: Over time, industry practitioners and thought leaders identified limitations and challenges with traditional models, leading to the emergence of Agile principles and methodologies. Agile draws upon these collective experiences, lessons learned, and best practices to provide a more effective and responsive software development approach.

Overall, the causes behind the Agile model revolve around the need to address the changing landscape of software development, adapt to evolving requirements, improve customer satisfaction, enhance team collaboration and communication, and deliver software solutions more efficiently in a rapidly changing business environment.

6. Conclusion

By summing up, the implementation of a supermarket management system is a significant project that offers numerous benefits for supermarkets and grocery stores. By adopting a comprehensive software solution, supermarkets can streamline their operations, enhance efficiency, and improve overall management processes.

The integration of features such as inventory management, point of sale, customer relationship management, employee management, sales analytics, and supplier management enables supermarkets to optimize their daily activities. These functionalities allow for accurate inventory tracking, smooth customer transactions, effective customer relationship management, streamlined employee scheduling and payroll, data-driven decisionmaking, and efficient supplier coordination.

The selected process mode, such as the Continuous Improvement approach, ensures ongoing enhancement of the supermarket management system. It fosters a culture of continuous learning, innovation, and efficiency, enabling supermarkets to adapt to changing market demands, make data-driven decisions, and stay competitive in the retail industry.

The Agile development model, with its iterative and customer-centric approach, is well-suited for the implementation of a supermarket management system. It allows for flexibility in responding to changing requirements, promotes collaboration between the development team and stakeholders, and facilitates early and continuous delivery of valuable software increments.

Overall, the implementation of a supermarket management system as a project brings numerous advantages. It enhances operational efficiency, reduces costs, improves customer service, enables accurate inventory management, and enhances decision-making capabilities. By embracing technology and adopting a comprehensive software solution, supermarkets can position themselves for success in a competitive market, meet the evolving needs of their customers, and achieve sustainable growth.

7. References

Dictionary

1. Longman

2. Oxford

3. Wikipedia

Books

1. Class notes by BUGINGO Emmanuel
2. Supermarkets management practices Author Ghazi Makammel Hossain
3. Supermarkets business plan template Author:Meir Liraz
4. Object oriented analysis and design using UML Author:Mahesh P. Matha
5. Retailing management by Michel Levy and Barton
6. Supermarkets management system by Alfred V.Aho

Websites

http://www.Amazon.com

http://studylib.net

www.google.com

http://www.studocu.com

www.softwareadvice.com

DATABASE OF THE SYSTEM

2.1INTRODUCTION:

In this chapter we will be describing database of the system tables inside that database and the way those tables were created in, addition on that in this chapter we will show table views from original tables, the relationship between created tables that are in this database. In database of this system be ready to look at operation used on some table in this database so that you can know what to and what not to do on given entity. Note that to develop this system database there are some material which were used so that we can get on final output of system as was described in chapter one “system analyses”. Among these the important one is xampp saver MySQL, so now let together navigate this database system.

```
- Generation Time: Feb 19, 2024 at 09:08 PM

-- Server version: 10.4.25-MariaDB

-- PHP Version: 7.4.30


SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";

START TRANSACTION;

SET time_zone = "+00:00";



/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;

/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;

/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;

/*!40101 SET NAMES utf8mb4 */;


--

-- Database: `hagenimana_diane_sms`

--


DELIMITER $$

--

-- Procedures
```

--

User table database structure

This is table is table that will be only created by admin and will hold other system users apart from admin it will give them email and password that they will use to login.

Note: the created user will not have same ability and right in the system

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `DeleteOldOrders` (IN `deleteDate` DATE)   BEGIN

   DELETE FROM `Order` WHERE OrderDate < deleteDate;

END$$


CREATE DEFINER=`root`@`localhost` PROCEDURE `GetCustomersWithMultipleOrders` ()   BEGIN

   SELECT C.CustomerID, C.FirstName, C.LastName

   FROM Customer C

   WHERE (SELECT COUNT(*) FROM `Order` O WHERE O.CustomerID = C.CustomerID) > 3;

END$$


CREATE DEFINER=`root`@`localhost` PROCEDURE `GetProductInfo` ()   BEGIN

   SELECT * FROM Product;

END$$


CREATE DEFINER=`root`@`localhost` PROCEDURE `InsertProduct` (IN `productName` VARCHAR(255),
IN `productPrice` DECIMAL(10,2), IN `productQuantity` INT)   BEGIN

   INSERT INTO Product (Name, Price, Quantity) VALUES (productName, productPrice,
productQuantity);

END$$


CREATE DEFINER=`root`@`localhost` PROCEDURE `UpdateProduct` (IN `productId` INT, IN
`newProductName` VARCHAR(255), IN `newProductPrice` DECIMAL(10,2))   BEGIN

   UPDATE Product SET Name = newProductName, Price = newProductPrice WHERE ProductID =
productId;

END$$
```

```
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `admin`
--
```

SQL

In this sub unit we will be describing techniques especially SQL queries used to create, delete tables as well as inserting and deleting data in the tables in that database.

1. created database is “sms”
2. queries to create tables and their relationship

```
CREATE TABLE `admin` (
  `email` varchar(40) NOT NULL,
  `password` varchar(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `admin`
--
```

QUERIES TO INSERT DATA INTO TABLES

```
INSERT INTO `admin` (`email`, `password`) VALUES
('supermarket@gmail.com', '000');

-- --------------------------------------------------------
```

```
--
-- Table structure for table `customer`
--

CREATE TABLE `customer` (
  `CustomerID` int(11) NOT NULL,
  `FirstName` varchar(50) NOT NULL,
  `LastName` varchar(50) NOT NULL,
  `Email` varchar(100) DEFAULT NULL,
  `password` varchar(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `customer`
--

INSERT INTO `customer` (`CustomerID`, `FirstName`, `LastName`, `Email`, `password`) VALUES
(1, 'John', 'Doe', 'john.doe@example.com', '000'),
(2, 'Jane', 'Smith', 'jane.smith@example.com', '1234'),
(11, 'kamugi', 'kahu', 'kamu@gmail.com', '1234');

--
-- Triggers `customer`
--
DELIMITER $$
CREATE TRIGGER `AfterDeleteCustomer` AFTER DELETE ON `customer` FOR EACH ROW BEGIN



END
```

```
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Stand-in structure for view `customerswithmultipleorders`
-- (See below for the actual view)
--
CREATE TABLE `customerswithmultipleorders` (
`CustomerID` int(11)
,`FirstName` varchar(50)
,`LastName` varchar(50)
);

-- --------------------------------------------------------

--
-- Stand-in structure for view `insertproduct`
-- (See below for the actual view)
--
CREATE TABLE `insertproduct` (
`ProductID` int(11)
,`Name` varchar(255)
,`Price` decimal(10,2)
,`Quantity` int(11)
);

-- --------------------------------------------------------
```

```
--
-- Table structure for table `order`
--

CREATE TABLE `order` (
  `OrderID` int(11) NOT NULL,
  `CustomerID` int(11) DEFAULT NULL,
  `OrderDate` date NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `order`
--

INSERT INTO `order` (`OrderID`, `CustomerID`, `OrderDate`) VALUES
(1, 1, '2023-09-10'),
(2, 2, '2023-09-11');

-- --------------------------------------------------------

--
-- Table structure for table `orders`
--

CREATE TABLE `orders` (
  `OrderID` int(11) NOT NULL,
  `CustomerID` int(11) NOT NULL,
  `ProductID` int(11) NOT NULL,
  `OrderDate` varchar(20) NOT NULL,
  `status` varchar(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `orders`
--

INSERT INTO `orders` (`OrderID`, `CustomerID`, `ProductID`, `OrderDate`, `status`) VALUES
(1, 11, 1, '1/1/2024', 'pending'),
(2, 2, 2, '1/1/2024', 'pending');

-- --------------------------------------------------------

--
-- Table structure for table `product`
--

CREATE TABLE `product` (
  `ProductID` int(11) NOT NULL,
  `Name` varchar(255) NOT NULL,
  `Price` decimal(10,2) NOT NULL,
  `Quantity` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `product`
--
```

```
INSERT INTO `product` (`ProductID`, `Name`, `Price`, `Quantity`) VALUES
(1, 'bread', '1200.00', 22),
(2, 'mandazi', '5000.00', 15),
(3, 'B Juice', '7000.00', 16);

--
-- Triggers `product`
--
DELIMITER $$
CREATE TRIGGER `AfterInsertProduct` AFTER INSERT ON `product` FOR EACH ROW BEGIN


END
$$
DELIMITER ;
DELIMITER $$
CREATE TRIGGER `AfterUpdateProduct` AFTER UPDATE ON `product` FOR EACH ROW BEGIN


END
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Stand-in structure for view `productinfo`
-- (See below for the actual view)
--
```

```
CREATE TABLE `productinfo` (

`ProductID` int(11)

,`Name` varchar(255)

,`Price` decimal(10,2)

);


-- --------------------------------------------------------


--

-- Stand-in structure for view `updatecustomer`

-- (See below for the actual view)

--

CREATE TABLE `updatecustomer` (

`CustomerID` int(11)

,`FirstName` varchar(50)

,`LastName` varchar(50)

);


-- --------------------------------------------------------


--

-- Table structure for table `users`

--


CREATE TABLE `users` (

  `id` int(11) NOT NULL,

  `email` varchar(39) NOT NULL,

  `password` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--

-- Dumping data for table `users`

--


INSERT INTO `users` (`id`, `email`, `password`) VALUES

(1, 'admin@gmail.com', 'admin');


-- --------------------------------------------------------


--

-- Structure for view `customerswithmultipleorders`

--

DROP TABLE IF EXISTS `customerswithmultipleorders`;


CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW
`customerswithmultipleorders`  AS SELECT `c`.`CustomerID` AS `CustomerID`, `c`.`FirstName` AS
`FirstName`, `c`.`LastName` AS `LastName` FROM `customer` AS `c` WHERE (select count(0) from
`order` `o` where `o`.`CustomerID` = `c`.`CustomerID`) > 33  ;


-- --------------------------------------------------------


--

-- Structure for view `insertproduct`

--

DROP TABLE IF EXISTS `insertproduct`;


CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW
`insertproduct`  AS SELECT `product`.`ProductID` AS `ProductID`, `product`.`Name` AS `Name`,
`product`.`Price` AS `Price`, `product`.`Quantity` AS `Quantity` FROM `product``product`  ;
```

```
-- --------------------------------------------------------


--

-- Structure for view `productinfo`

--

DROP TABLE IF EXISTS `productinfo`;


CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW
`productinfo`  AS SELECT `product`.`ProductID` AS `ProductID`, `product`.`Name` AS `Name`,
`product`.`Price` AS `Price` FROM `product``product`  ;


-- --------------------------------------------------------


--

-- Structure for view `updatecustomer`

--

DROP TABLE IF EXISTS `updatecustomer`;


CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW
`updatecustomer`  AS SELECT `customer`.`CustomerID` AS `CustomerID`, `customer`.`FirstName` AS
`FirstName`, `customer`.`LastName` AS `LastName` FROM `customer``customer`  ;


--

-- Indexes for dumped tables

--


--

-- Indexes for table `customer`

--
```

```
ALTER TABLE `customer`
  ADD PRIMARY KEY (`CustomerID`),
  ADD UNIQUE KEY `Email` (`Email`);

--
-- Indexes for table `order`
--
ALTER TABLE `order`
  ADD PRIMARY KEY (`OrderID`),
  ADD KEY `CustomerID` (`CustomerID`);

--
-- Indexes for table `orders`
--
ALTER TABLE `orders`
  ADD PRIMARY KEY (`OrderID`);

--
-- Indexes for table `product`
--
ALTER TABLE `product`
  ADD PRIMARY KEY (`ProductID`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `customer`
```

```
--

ALTER TABLE `customer`

  MODIFY `CustomerID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=12;


--

-- AUTO_INCREMENT for table `order`

--

ALTER TABLE `order`

  MODIFY `OrderID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;


--

-- AUTO_INCREMENT for table `product`

--

ALTER TABLE `product`

  MODIFY `ProductID` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;


--

-- Constraints for dumped tables

--


--

-- Constraints for table `order`

--

ALTER TABLE `order`

  ADD CONSTRAINT `order_ibfk_1` FOREIGN KEY (`CustomerID`) REFERENCES `customer`
(`CustomerID`);

COMMIT;


/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
```

```
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;

/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

By concluding on this database we have achieved more on this system using MySQL and the system in terms of keeping data is working well with different operation we have settled on it in some tables like revoking some activities, triggers are created in this database and relationship of much of these tables in this database are there to link them one by one. So this will lead us to creating the link between data store which is database and the final user of the system which will be focusing more on user interface and connectivity.

JAVA PROGRAMMING

1 introduction

In this chapter I will be describing how powerful generalpurpose programming language was used to create the analyzed system. Under this chapter I will undergo full detail of how everything will function together with database that have been describe above and how it cope with full analyzed system.

Tools used to develop this system in java programming:

Eclipse IDE: an integrated development environment used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. It is the second-most-popular IDE for Java development, and, until 2016, was the most popular.

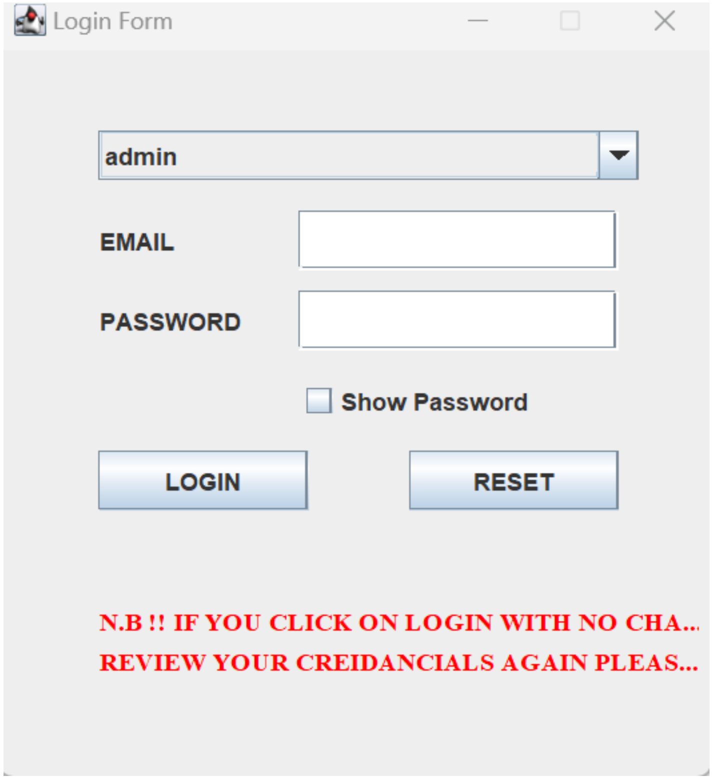
JAR stands for Java Archive. It's a file format based on the popular ZIP file format and is used for aggregating many files into one. Although JAR can be used as a general archiving tool, the primary motivation for its development was so that Java applets and their requisite components. MySQL Connectors. MySQL provides standards-based drivers for JDBC, ODBC, and .Net enabling developers to build database applications in their language

Forms description

So let look together how the system will function one by one from the beginning up to the end.

1. ADMIN LOGIN:

This button is dedicated for system admin to login so that he/she can do more in the system, like adding customer or department and many more

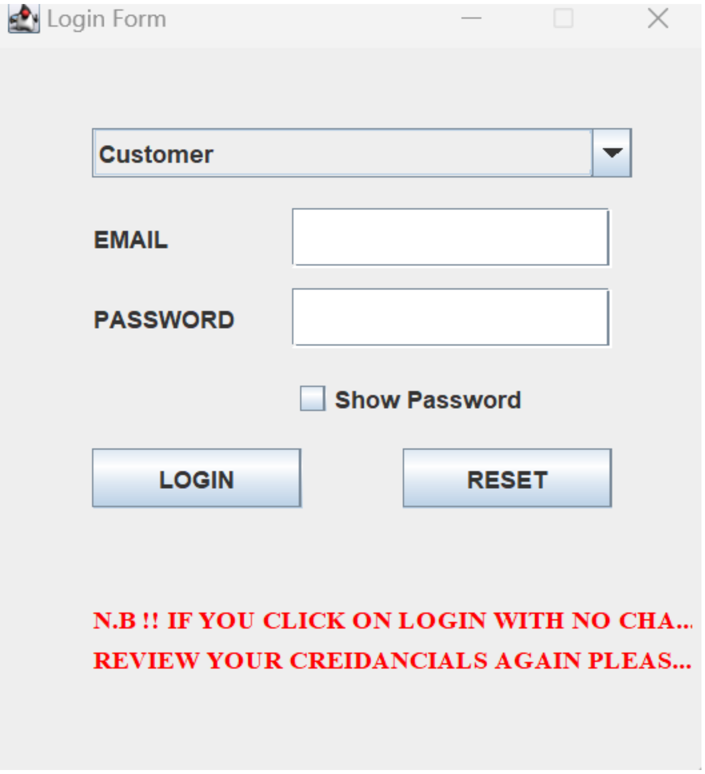


N.B !! IF YOU CLICK ON LOGIN WITH NO CHA...
REVIEW YOUR CREIDANCIALS AGAIN PLEAS...

For admin to continue his/her tasks he will need to login using created and allowed username and password otherwise no access will be given.

2. CUSTOMER LOGIN

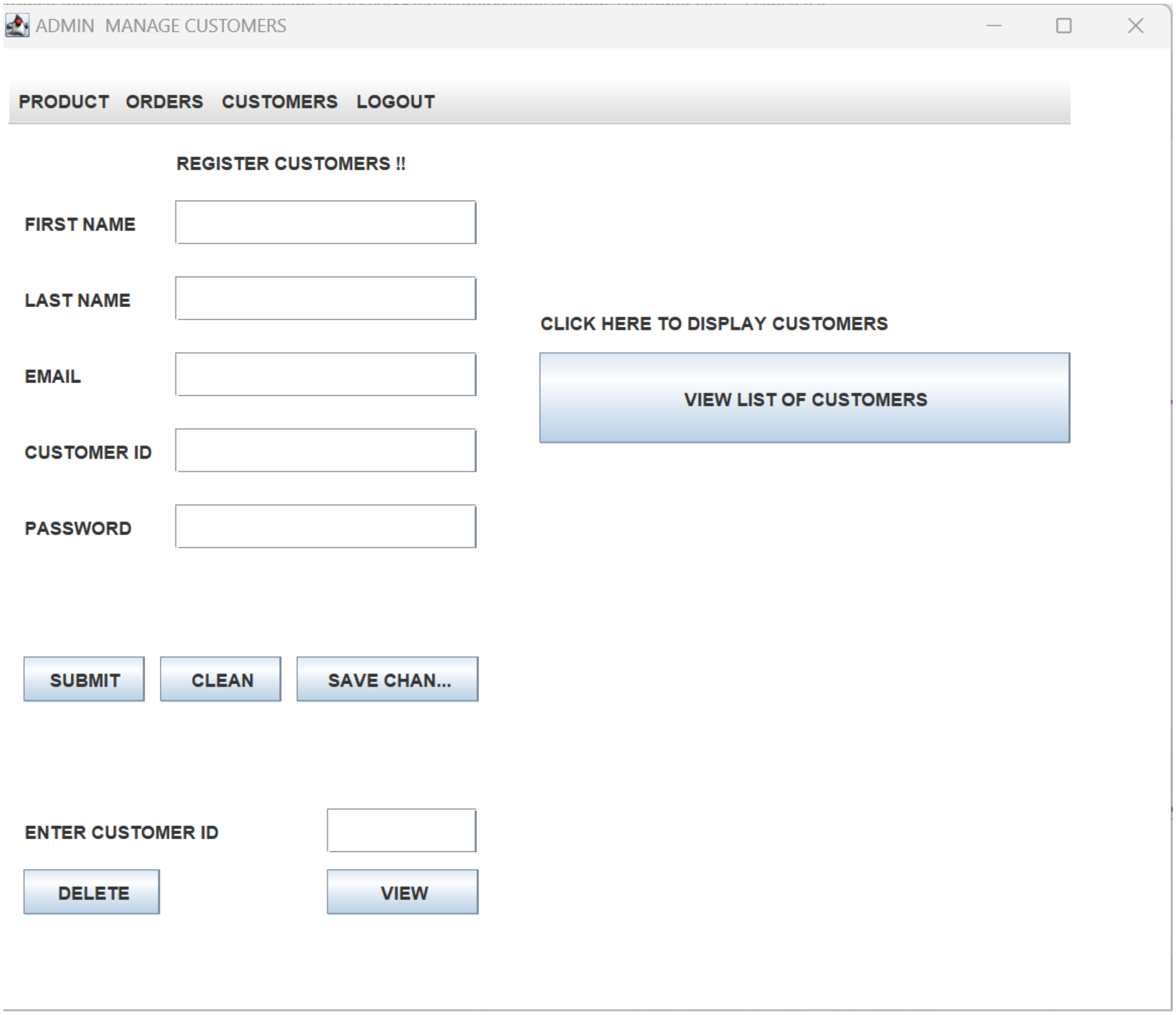
This is where customer is able to enter the system by only making order



N.B !! IF YOU CLICK ON LOGIN WITH NO CHA...
REVIEW YOUR CREIDANCIALS AGAIN PLEAS...

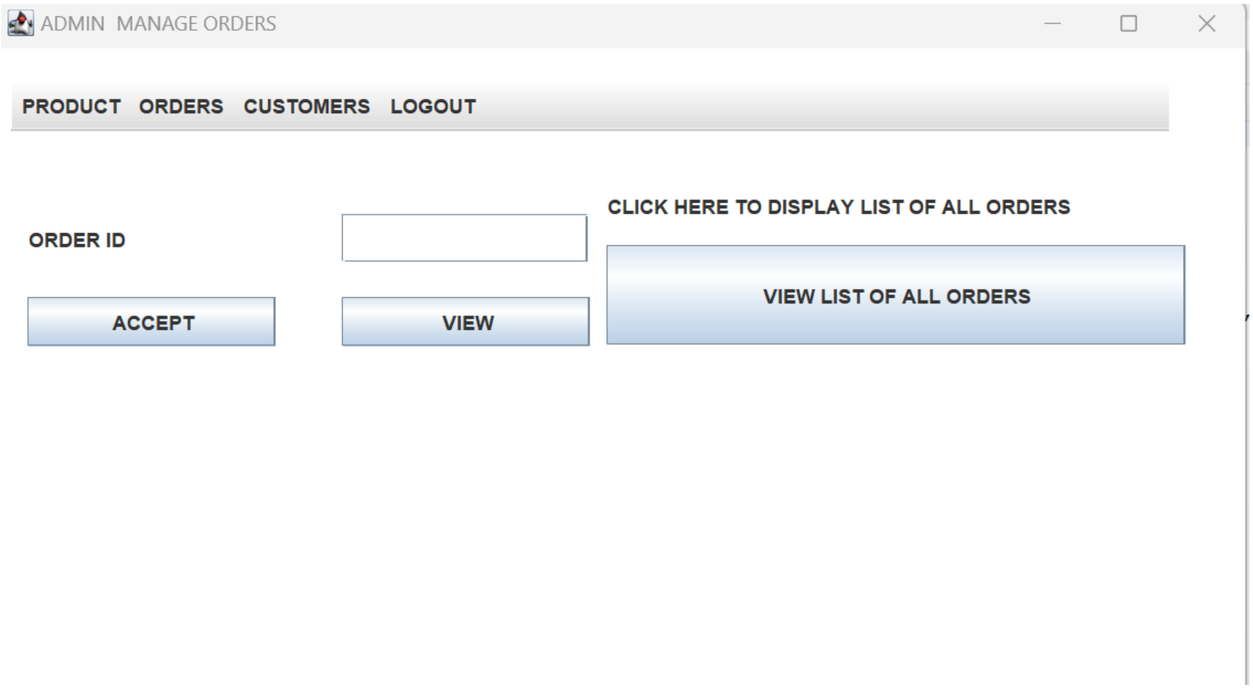
3.ADMIN MANAGER CUSTOMER

This where Admin is able to update , delete ,insert the customer



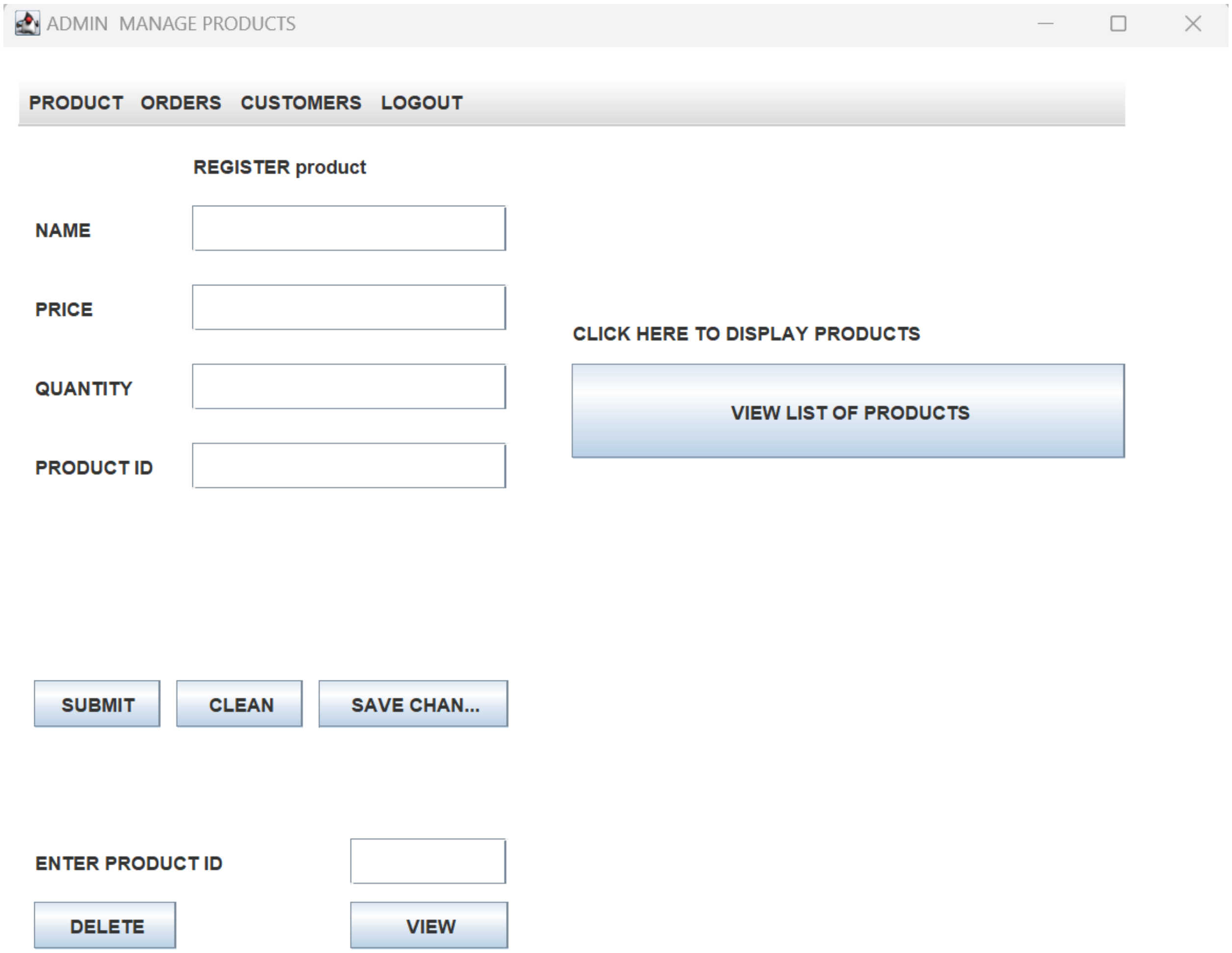
4.ADMIN MANAGE ORDERS

This is where admin is able to receive or reject the orders according to what we have in our store.



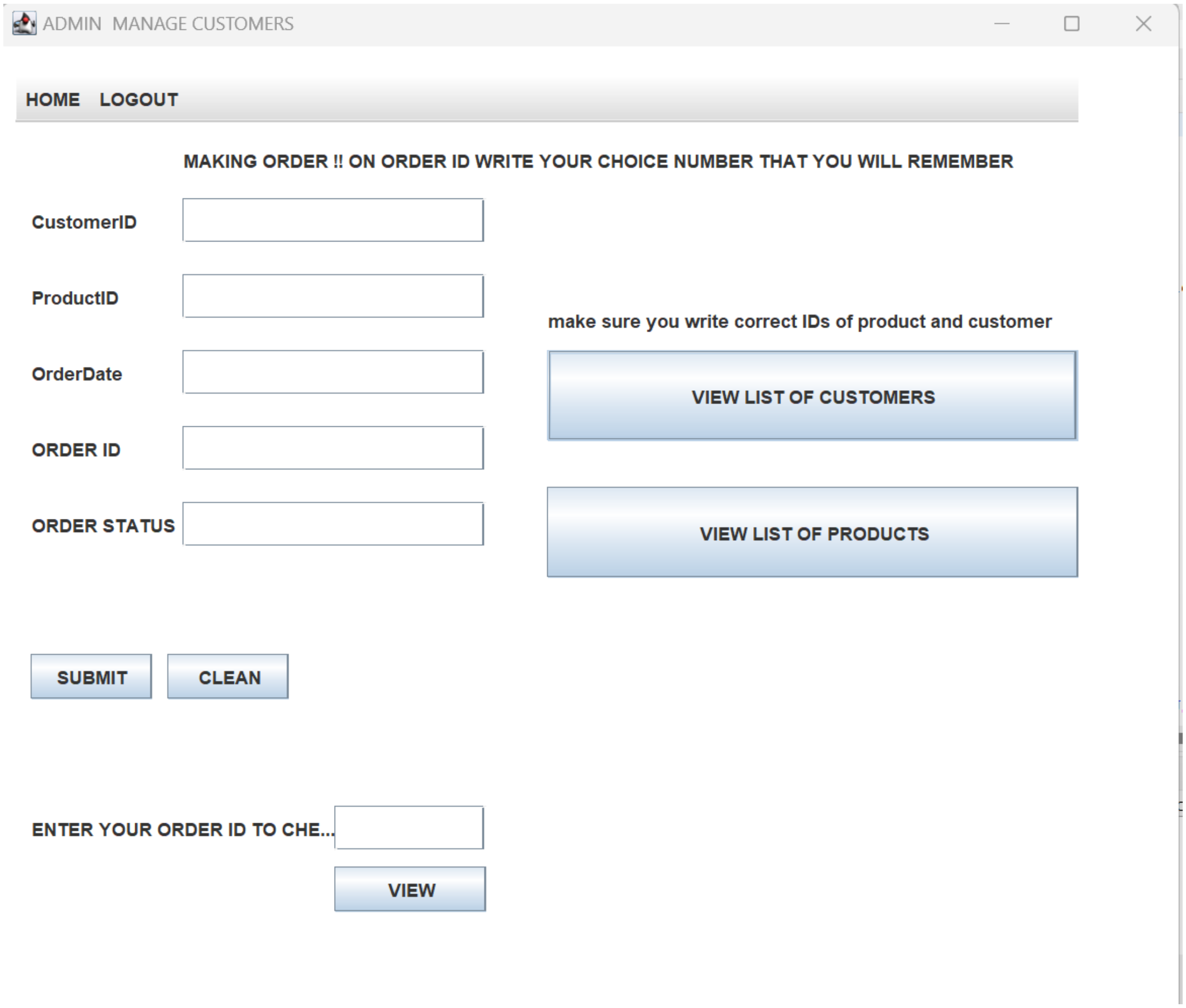
5. ADMIN MANAGE PRODUCTS

This where admin is able to update, delete, insert, the price of the product , the new product which we have in our store.



6. CUSTOMER MAKING ORDER

This is where customer are able to make order in our supermarket according to what we have in our store.

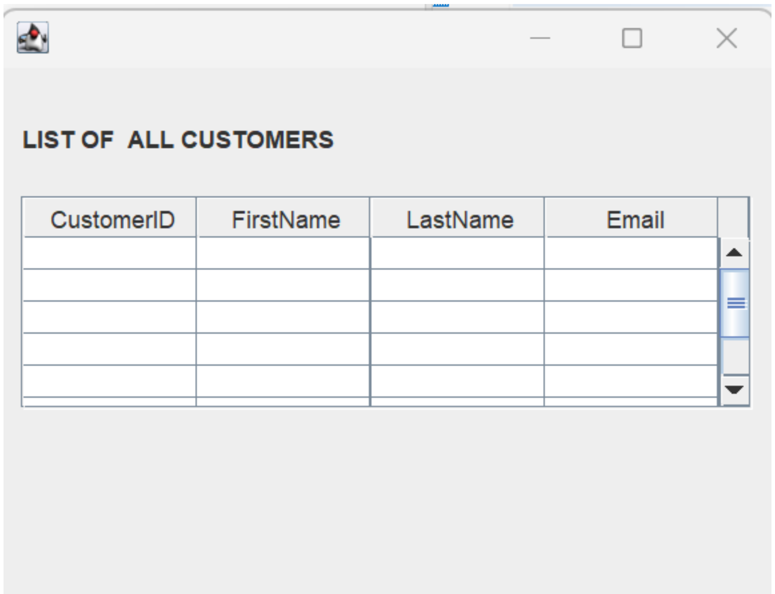


MAKING ORDER !! ON ORDER ID WRITE YOUR CHOICE NUMBER THAT YOU WILL REMEMBER

make sure you write correct IDs of product and customer

7. LIST OF ALL CUSTOMERS

This is the list where the information of our customers are recorded

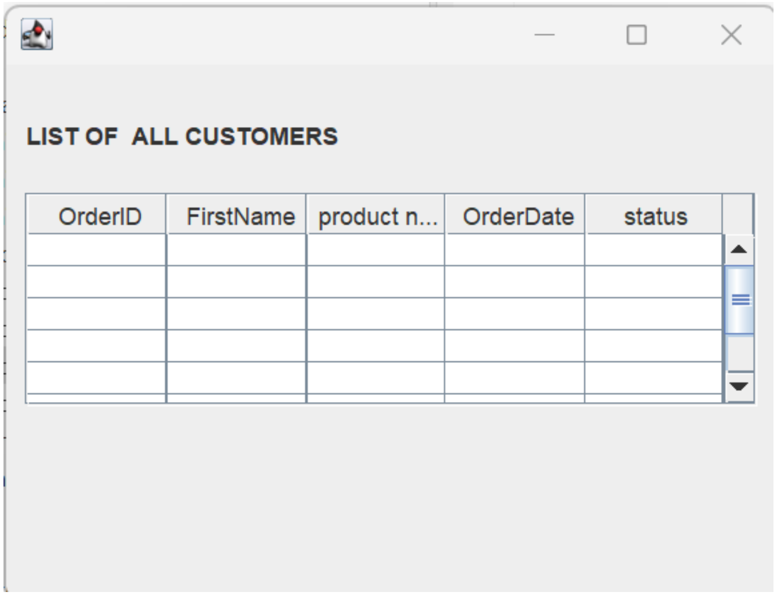


LIST OF ALL CUSTOMERS

CustomerID	FirstName	LastName	Email

8. LIST OF ORDERS

This where we the order made by our customer inother to give them what they want.

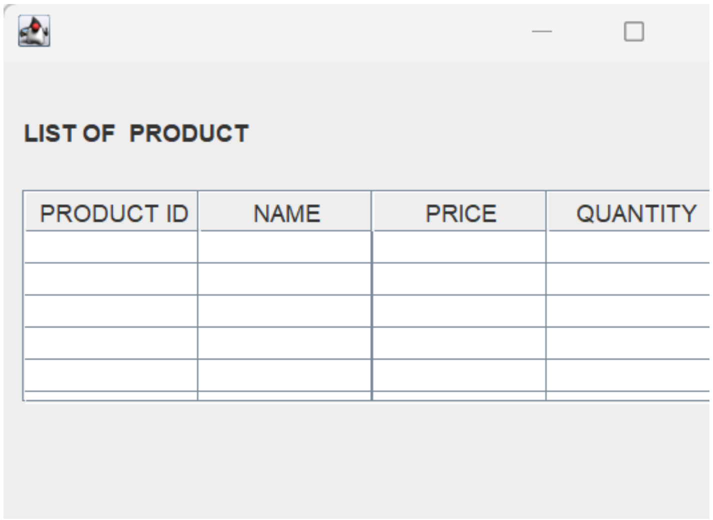


LIST OF ALL CUSTOMERS

OrderID	FirstName	product n...	OrderDate	status

9. LIST OF ALL PRODUCT

This is the form which show all information about the product we have in the store such as price, name,…….



LIST OF PRODUCT

PRODUCT ID	NAME	PRICE	QUANTITY

CONCLUSION:

By concluding this chapter concerns with java programming especially in my developed system, we can say that I have final product that I was expecting to have it, the manipulation of data is going well the design is there with special appearance, but there much to go on and that need to be improved will be gained from external view apart from system developer.