Feasibility Study

for

PID-27 - Smart POS (Point of Sales) application

Prepared by Group 11

Mentor - Dr. Sapumal Ahangama

Group Members

Kobinarth Panchalingam - 200307C

R. A. T. C. Kumara - 200321M

Sanjula Kumarasinghe - 200323V

TABLE OF CONTENTS

Index Number	Contribution
200307C	• Introduction • Financial Feasibility • Technical Feasibility • References
200321M	• Introduction • Risk Feasibility • Social/Legal Feasibility • Considerations
200323V	• Introduction • Technical Feasibility • Resource and Time Feasibility

1.Introduction

1.1 Overview of the Project

The Smart POS application aims to develop a modern and feature-rich Point of Sales (POS) system for businesses to efficiently manage sales transactions, inventory, and customer information. The proposed system will consist of both a mobile application and a web application to cater to different user needs. The mobile application will turn a smartphone into a hand-held POS system, enabling sales staff to process transactions on the go. The web application will provide a comprehensive platform for managers and administrators to monitor sales data, manage inventory, and analyze business performance. And customers will have an online portal to order and purchase goods.

Input:

- Product details such as descriptions, pricing, variants (sizes, colors), and images entered by administrators through the web application.
- Sales transaction information, including product quantities, payment methods, and customer details, input by sales staff using the mobile application.
- Customer orders placed through the online portal by customers.

Output:

- Receipts generated after each transaction, including a breakdown of the purchased items and the total amount, displayed on the mobile application for customers and sales staff.
- Real-time updates on inventory levels and stock alerts for low items, shown on the web application for administrators and managers.
- Reports and analytics on sales trends, best-selling items, employee performance, and other key metrics, accessible on the web application for managerial decision-making.
- Order confirmations and delivery details are sent to customers through the online portal after placing their orders.

1.2 Objectives of the Project

- Design and implement a user-friendly application for efficient sales processing.
- Automate inventory management processes.
- Create an online portal to purchase goods.
- Create a Comprehensive Web Application for Managers and Administrators.

1.3 The Need for the Project

Traditional POS systems often lack mobility and comprehensive analytics, limiting businesses' ability to make data-driven decisions. By automating sales processes, optimizing inventory management, and offering real-time data insights, the Smart POS system aims to enhance business efficiency and decision-making. With a focus on data security, scalability, and cost-effectiveness, the Smart POS system is designed to empower businesses, foster growth, and adapt to technological trends, ensuring their sustained success in today's dynamic market.

1.4 Scope of the Project.

- Sales Staff: Access the application for processing sales transactions, scanning barcodes/QR codes, and generating receipts.
- Managers/Administrators: Utilize the web application to monitor sales trends, manage inventory, employees and generate reports.
- Customers: Engage with the online portal to purchase goods.

1.5 Deliverables.

- A web-based software system for managing sales transactions and for managing employees by administrators.
- A mobile application for staff members to manage sales transactions and for managing the inventory as well.
- An online portal for customers to purchase goods.

1.6 Overview of Existing Systems and Technology

Square POS [1] is a popular mobile-based Point of Sale system that enables businesses to process payments, track sales, and manage inventory. It offers a user-friendly mobile app for sales staff and a web-based dashboard for administrators to monitor sales data and inventory. Square POS uses secure payment processing technologies and provides real-time analytics to help businesses make data-driven decisions. It is widely used in various industries, including retail, foodservice, and small businesses.

Techniques/Tools/Resources/Approaches for System Implementation:

The Smart POS application utilizes a modern technology stack to provide a seamless and user-friendly experience. With React [2] and React Native [3] for frontend development, Spring Boot [4] for the backend, GraphQL for APIs [5], and PostgreSQL [6] as the database. For secure and seamless payment processing, we will integrate a reliable payment gateway API, such as PayPal, Stripe, or Square, to handle transactions securely. To implement barcode and QR code scanning functionality, we will utilize barcode scanning libraries and APIs like ZXing (Zebra Crossing) or Google Vision API for mobile platforms.

2. Feasibility Study

2.1 Financial Feasibility

Costs associated with the project can be categorized into two major categories, development costs and operational costs.

Development Cost

Since the project will be using open-source frameworks, and freely available tools for development, cost associated with development, or the system will be minimal.

For testing and demonstration purposes Render [6] can be used for hosting backend service, Netlify [7] for hosting React frontend website and Postgres database can be hosted in Azure [8] using free student account credits. All these solutions are free of charge.

The development cost for the React Native mobile app encompasses the creation of a dynamic and user-friendly application that can seamlessly run on both iOS and Android platforms. React Native, a versatile and efficient framework, enables the development of a single codebase that serves as the foundation for both platforms, resulting in cost savings and streamlined development. Utilizing Expo Go for testing the React Native app ensures a streamlined and cost-effective development approach. Expo Go [10] simplifies the deployment process, enabling seamless distribution across iOS and Android devices. It is also free of charge.

Cost item	Projected cost / year (in LKR)
Salaries and Wages	0.00
Programming Equipment	0.00
Software Licenses	0.00
Software Utilities and Tools	0.00
Web hosting • Netlify	0.00
App publishing • Expo Go	0.00
Database • Azure flexible server - PostgreSQL	0.00
	0.00

Operational Costs

The Smart POS system encompasses a range of operational aspects that contribute to its seamless functioning and continued excellence. These include hosting, domain name registration, app publishing, hardware and software maintenance, user support, data backups, and comprehensive security measures. These components collectively ensure optimal performance, reliability, and user satisfaction.

Hosting and Infrastructure
Hosting web applications and databases constitutes a significant operational cost. The expenditure varies based on the company's specific requirements and desired performance levels. While options like Hostinger, Wix, and BlueHost offer accessible domain names and hosting services, the costs are influenced by performance needs and available pricing plans. Embracing the flexible "pay-as-you-go" pricing models offered by reputable cloud service providers is a strategic approach, catering to both immediate needs and future system growth.

Mobile App Publishing
For the Android platform, the Smart POS mobile app can be made accessible through the Google Play Store. This avenue demands an initial registration fee and subsequent annual fees for app publication. It is worth noting that this route offers unparalleled convenience, ensuring easy installation and ongoing maintenance. In contrast, an alternative approach involves sharing the APK among users. While this mitigates some costs, it introduces trade-offs in terms of user experience and convenience.

By thoughtfully considering these operational aspects and making informed decisions, the Smart POS system ensures a robust, user-friendly, and cost-conscious environment for both administrators and end-users.

2.2 Technical Feasibility

The Smart POS consist of a web application and mobile application. The main technologies, frameworks and tools that arc associated with the project are,

System Components	Technologies/ Tools
Backend	• Spring Boot
Frontend	• ReactJS • React-Native (mobile-app) • Redux
Database	• PostgreSQL
IDEs	• VS code • IntelliJ IDEA • Android Studio

Hosting Services	• Neltify • Expo Go • Render
Diagram tools	• Lucidchart • Draw.io • Figma
APIs	• PayPal • react-native-vision-camera
Version Control Management	• Git

The chosen technologies for the Smart POS system demonstrate a strong foundation for technical feasibility, ensuring efficient development, scalability, and a seamless user experience.

Spring Boot, a robust Java-based framework, empowers the backend with rapid development capabilities and enriched with GraphQL integration for seamless data querying, streamlining data processing and communication.

ReactJS and React-Native, coupled with Redux, facilitate dynamic and responsive user interfaces for both web and mobile platforms, enhancing user engagement.

PostgreSQL serves as a reliable database solution, supporting data integrity and real-time updates crucial for sales and inventory management.

The integration of PayPal API [11] and react-native-vision-camera [12] enriches the system with secure payment processing and barcode scanning functionalities, respectively.

Version control through Git promotes collaborative development, while the use of popular IDEs like VS Code, IntelliJ IDEA, and Android Studio ensures efficient code creation.

Hosting services such as Netlify, Expo Go, and Render facilitate effortless deployment, while diagram tools like Lucidchart, Draw.io, and Figma aid in clear visualization and design. This harmonious blend of technologies fosters a technically feasible environment for crafting a feature-rich, scalable, and user-centric Smart POS system.

2.3 Resource and Time Feasibility

Resource Feasibility

Resources which are required for the proposed POS system.

- Computers, mobile phones.
- A team for development
- Programming tools (Free)
  - Visual Studio code
  - Android Studio
  - IntelliJ IDEA

So, the project has the required resources.

Time Feasibility

Project can be completely developed within the given 15 weeks' time frame.

2.4 Risk Feasibility

Database or server failure

Database failures can be minimized using backups. Even better is having backups of backups. To reduce the probability of a system failure, selecting a server with the appropriate size relative to system operations would be beneficial.

Staff Training and Change Management

Introducing new software may require staff training and change management efforts to ensure smooth adoption. Resistance to change or inadequate training can hinder the software's effectiveness.

User Acceptance

User acceptance is critical for the success of the software. If the software does not meet user expectations or is challenging to use, employees may resist adoption, leading to operational disruptions. Involving end-users in the development process through user testing and feedback can help address this risk.

Market and Technological Changes

The retail industry is continually evolving, and technology advances rapidly. The software should be flexible and adaptable to accommodate future market trends and technological advancements.

Data Security and Privacy

Supermarket software deals with sensitive data, including customer information, financial records, and inventory details. A data breach can result in financial losses, legal liabilities, and damage to the supermarket's reputation. Robust security measures, encryption protocols, and regular security audits are essential to safeguard data.

Scope Creep

The risk of scope creep arises when the project's objectives and requirements keep expanding beyond the initial plan. This can lead to budget overruns, missed deadlines, and compromised quality. Proper project scope management and change control procedures are necessary to mitigate this risk.

Risk description	**Likelihood of occurrence**	**Impact on project**	**Mitigation policy**
Database or server failure	Medium	High	Maintaining a backup database
Staff Training and Change Management	Medium	High	Handle user friendly interfaces and interacting with users during implementation
User Acceptance	Low	High	Handle user friendly interfaces and interacting with users during implementation
Market and Technological Changes	High	Medium	Engage experienced developers and conduct thorough testing throughout the development process
Data Security and Privacy	Low	High	Implement security Measures.
Scope Creep	Medium	High	Define a clear and detailed project scope from start.

2.5 Social/Legal Feasibility

Social Feasibility

Social feasibility assesses whether the proposed Smart POS application aligns with the needs and expectations of the supermarket's customers, employees, and other stakeholders.

Customer Acceptance: Evaluate whether customers are open to adopting new technology and using the Smart POS application. Conduct surveys or focus groups to gauge customer interest and feedback.

Employee Training and Acceptance: Consider the readiness of supermarket staff to use the Smart POS application. Assess whether they are comfortable with technology and if proper training can be provided to ensure smooth adoption.

User Experience: Ensure that the application's user interface and features are intuitive and user-friendly for both customers and employees. A positive user experience is crucial for successful adoption.

Accessibility and Inclusivity: Assess whether the Smart POS application is accessible to all customers, including those with disabilities or those less familiar with technology.

Cultural Considerations: Consider any cultural factors that may impact the acceptance of the application in different regions or among diverse customer demographics.

Legal Feasibility

The Smart POS application's legal viability is determined by determining whether it complies with all applicable laws, rules, and standards.

Data Privacy and Security: Ensure that the application collects, processes, and stores customer and employee data in compliance with data protection laws (e.g., GDPR, CCPA).

Payment Regulations: Confirm that the application meets the necessary payment processing regulations to handle financial transactions securely and legally.

Licensing and Intellectual Property: Verify that the application and any third-party components used are appropriately licensed and that there are no copyright or patent infringement issues.

Consumer Protection Laws: Ensure that the application provides accurate information about products, pricing, and terms, adhering to consumer protection laws.

Contractual Obligations: Review existing contracts with vendors or service providers to ensure that the development and implementation of the Smart POS application do not violate any contractual terms.

Accessibility Compliance: Ensure the application meets accessibility standards to accommodate individuals with disabilities, in accordance with relevant accessibility laws.

3. Considerations

Security

- To guarantee data integrity and shield the system from various sorts of hacks, security must be maintained.
- The username and password will be used to authenticate users. The username will be the user's email address.
- With many degrees of access that can be identified by the system using the given username and password without the need for an additional data column, the System will keep with the least privilege principle.

Ease of use

This system is designed to be utilized by anyone, even those with less computer literacy, for both mobile and web apps. Therefore, creating an interface would be one of the project's main concerns.

Availability

This system must be accessible 24 hours. Therefore, the system must carefully manage all resources and pick a hosting space that satisfies all criteria to guarantee maximum system availability. Load balancing should be properly implemented because daytime utilization of the system predominates, and nighttime usage is limited.

Usability

The system's interfaces must be appropriate for all of its users, who fall into a variety of age groups and educational levels and use it. These web and mobile applications are designed to use simpler language and pay greater attention to visual display features. Therefore, even those with poorer reading comprehension or less familiarity with the English language can use these programs. People who are less familiar with new technologies might not comprehend why or under what conditions they would receive notifications, why they would need to disclose their key, etc. These issues will be fixed by the applications' precise descriptions, which are required.

4. References

[1] Square Point of Sale. Square, Inc. [Online]. Available: https://squareup.com/us/en/point-of-sale. (Accessed: August 7, 2023).

[2] "React,". React - A JavaScript library for building user interfaces. [Online]. Available: https://react.dev/reference/react. (Accessed: August 7, 2023).

[3] "React Native,". React Native - A framework for building native apps using React. [Online]. Available: https://reactnative.dev/docs/getting-started. (Accessed: August 7, 2023).

[4] "Spring Boot,". Spring Boot Documentation. [Online]. Available: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#getting-started. (Accessed: August 7, 2023).

[5] "Introduction to GraphQL." A query language for API. [Online]. Available: https://graphql.org/learn/. (Accessed: August 7, 2023).

[6] "PostgreSQL". PostgreSQL documentation. [Online]. Available: https://www.postgresql.org/docs/. (Accessed: August 7, 2023).

[7] "Render". Pricing plan on Render. [Online]. Available: https://render.com/pricing. (Accessed: August 7 , 2023).

[8] "Netlify". Get started with Netlify. [Online]. Available: https://docs.netlify.com/get-started/. (Accessed: August 7, 2023).

[9] "Azure". Azure for students. [Online]. Available: https://azure.microsoft.com/en-us/free/students. (Accessed: August 7, 2023).

[10] "Expo Go". Expo Documentation. [Online]. Available: https://docs.expo.dev/get-started/expo-go/. (Accessed: August 7, 2023).

[11] "Get started". PayPal API reference. [Online]. Available: https://developer.paypal.com/api/rest . (Accessed: August 7, 2023).

[12] "Get started | Vision Camera". react-native-vision-camera API reference. [Online]. Available: https://www.react-native-vision-camera.com/docs/guides. (Accessed: August 7, 2023).