St. Paul's University

BAR 4102A

IT Business Research Project

Charles Kamiri

BOBITNRB200621

Management Software for Informal Businesses

Table of Contents

1 Introduction

1.1 Background

Informal businesses, also known as micro, small, and medium enterprises (MSMEs), are a critical component of the global economy. These businesses provide employment opportunities and contribute to economic growth in many developing countries.

1.2 Problem Statement

HoIver, most informal businesses face several challenges, such as limited access to financial resources, lack of business skills, and poor record-keeping practices. These challenges hinder the growth and sustainability of these businesses, limiting their potential to contribute to the economy.

1.3 Objectives

To address these challenges, I propose the development of a management software specifically designed for informal businesses. The software aims to provide a simple, affordable, and accessible solution to help these businesses manage their operations effectively. The software will offer features such as inventory management, sales tracking, financial management, and customer relationship management. With these features, informal businesses will be able to keep accurate records, make informed decisions, and improve their profitability.

1.4 Justification

Our proposed management software is designed to be user-friendly, affordable, and accessible to all types of informal businesses, regardless of their level of technical expertise. I believe that this software can have a significant impact on the growth and sustainability of informal businesses, leading to more significant contributions to the economy.

1.5 Limitations

Limited access to technology: Many informal businesses may not have access to the necessary technology, such as computers or smartphones, to use the management software effectively. This could limit the potential reach and impact of the software.

Limited data availability: Informal businesses may not have access to the same level of data as formal businesses, making it more difficult to develop accurate and reliable management software. This could result in incomplete or inaccurate information being used to make business decisions.

Limited financial resources: Many informal businesses operate on very limited budgets, which may make it difficult for them to afford the cost of the software or the hardware required to use it.

Limited technical expertise: Informal business owners may not have the technical expertise required to use the management software effectively, which could limit its usefulness and adoption.

Resistance to change: Some informal businesses may be resistant to adopting new technology or changing their existing business practices, which could limit the adoption and effectiveness of the management software.

Limited scalability: If the management software is not designed to be scalable, it may not be able to handle the growing needs of businesses as they expand, potentially limiting its usefulness over time.

Lack of integration with existing systems: If the management software is not designed to integrate with existing systems or software used by informal businesses, it may not be used effectively or may require additional work to use alongside existing tools.

2 Literature Review

Management software for informal businesses has gained attention in recent years due to its potential to improve business performance. Several studies have explored the use of management software by informal businesses, identifying its benefits and challenges.

A study by Akter and colleagues (2019) examined the adoption of management software among informal businesses in Bangladesh. The study found that the use of software significantly improved business performance, including sales growth and profitability. HoIver, the study also identified several challenges, including the lack of technical expertise and infrastructure.

Similarly, a study by Fuchs and colleagues (2021) examined the impact of a mobile-based management system on microenterprises in Tanzania. The study found that the system improved businesses' ability to manage inventory and finances, leading to increased sales and profitability. The study also highlighted the importance of user-friendly software and training to overcome barriers to adoption.

Another study by Yumusak and colleagues (2021) investigated the factors influencing the adoption of management software by informal businesses in Turkey. The study found that businesses Ire more likely to adopt software if they perceived it to be useful, easy to use, and compatible with their existing systems. The study also highlighted the importance of social networks in promoting software adoption among informal businesses.

The research revieId in this literature review suggests that management software can significantly improve the performance of informal businesses. HoIver, the adoption of software by informal businesses faces several challenges, including technical expertise, infrastructure, user-friendliness, and compatibility with existing systems. Addressing these challenges through training and social networks can help to promote the adoption of management software by informal businesses.

3 Methodology

3.1 Overview

In this section, we describe the methodology that we followed to develop our software system. The methodology is based on the agile software development approach, which emphasizes collaboration, flexibility, and responsiveness to change. We divided the development process into several phases, each of which involved different activities and deliverables.

i Purpose

The purpose of this project is to develop a management software that can help informal businesses to manage their daily operations and grow their business. The software will provide features such as inventory management, sales tracking, expense tracking, customer management, and reporting.

ii Target Audience

The target audience for this software is informal businesses such as small shops, food vendors, and service providers. These businesses often lack the resources and expertise to manage their operations efficiently and could benefit from a simple and easy-to-use management software.

iii Scope

The scope of this project includes developing a web-based management software that can be accessed from any device with an internet connection. The software will include the following features:

- Inventory management: Keep track of products, quantities, and prices.
- Sales tracking: Record sales transactions and generate sales reports.
- Expense tracking: Record expenses and generate expense reports.
- Customer management: Store customer information and purchase history.
- Reporting: Generate reports on sales, expenses, and inventory.

3.2 Requirements Gathering

To ensure that the software meets the needs of the target audience, the following data will be collected:

i User Needs

To understand the needs of the target audience, the following data will be collected:

- Demographic information: Age, gender, location, and occupation of the target audience.
- Business information: Types of products or services offered, current challenges faced by the business, and current methods of managing operations.

The data will be collected through surveys and interviews with informal business owners.

ii User Behavior

To understand how the target audience currently manages their operations and how they may use the software, the following data will be collected:

- Inventory data: Types of products, quantities, and prices.
- Sales data: Transactions, customer information, and payment methods.
- Expense data: Types of expenses, amounts, and frequency.
- Customer data: Information on customer demographics, purchase history, and loyalty.

The data will be collected through the software itself, which will track and store the relevant information.

iii Performance Metrics

To ensure that the software is meeting the needs of the target audience and improving their operations, the following performance metrics will be collected:

- User engagement: Number of users, frequency of use, and time spent on the software.
- Software performance: Speed, reliability, and availability of the software.
- Business performance: Improvement in sales, reduction in expenses, and growth of the business.

The data will be collected through the software itself and analyzed using data analysis tools.

iv Tools

To collect and analyze the data, the following tools will be used:

- Survey and interview tools: Google Forms and Zoom.
- Data analysis tools: Excel and Tableau.

Google Forms and Zoom will be used to conduct surveys and interviews with informal business owners. The data collected will be analyzed using Excel and Tableau, which will help identify patterns and insights from the data.

3.3 Architectural Framework

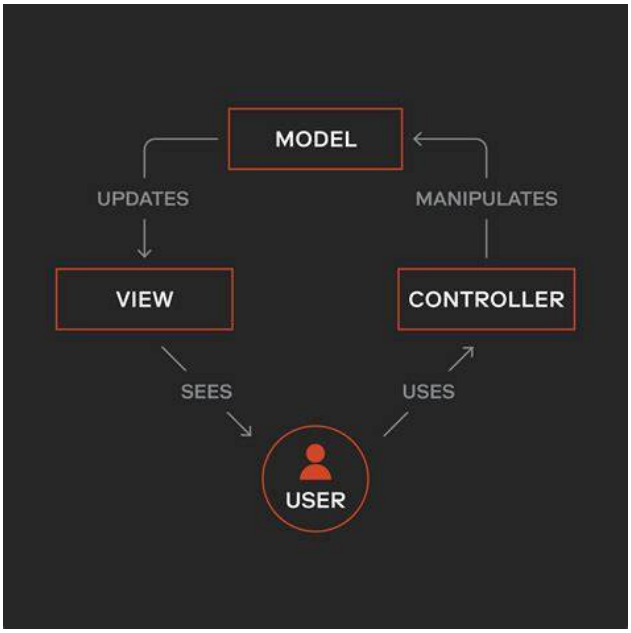
A model-driven approach will be utilized to design the software system's architecture. I will use UML (Unified Modeling Language) to create high-level diagrams that capture the system's structure and behavior. I will also use design patterns to guide our design decisions and ensure that our software system is scalable, maintainable, and extensible.

The architectural design consists of several components, including the user interface, application logic, and database. The user interface will be designed using a responsive design approach, which will ensure that the software system is accessible and usable across different devices and platforms. The application logic will be designed using a modular approach, which will allow for the breaking down the system into smaller, more manageable components. The database will be designed using a relational database management system (RDBMS), which will provide a flexible and efficient way to store and retrieve data.

The software architecture of our system is designed to provide a scalable and extensible framework for building complex software applications. The architecture is based on a layered approach, with each layer providing a specific set of functionality and services.

The architectural framework of our system is based on the Model-View-Controller (MVC) design pattern. This design pattern separates the application logic into three distinct components:

- **Model**: This component represents the data and the business logic of the application.
- **View**: This component represents the user interface of the application.
- **Controller**: This component acts as the intermediary between the Model and the View, and handles user input and business logic.



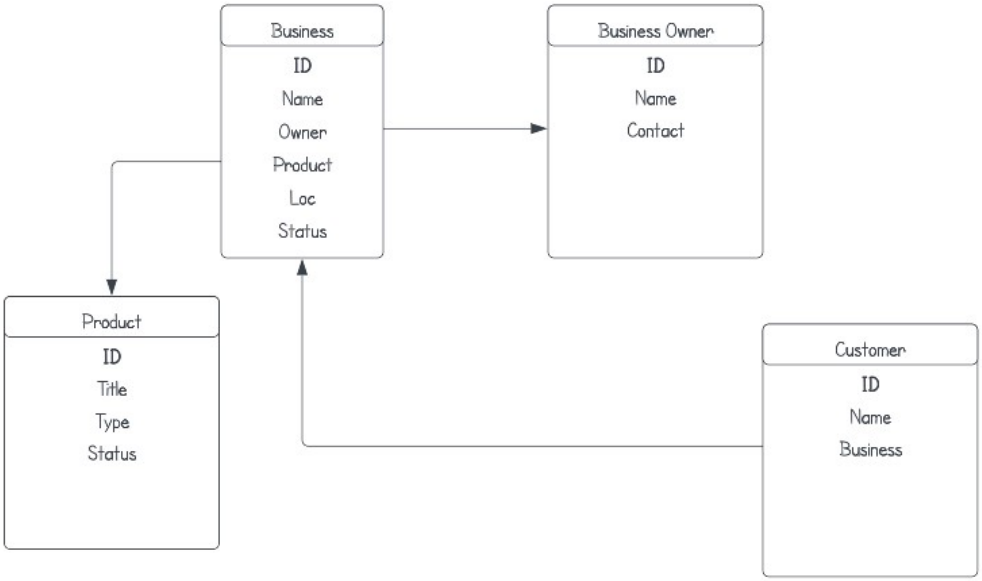
By separating these components, we can achieve greater modularity, maintainability, and flexibility in our codebase.

3.4 Component Overview

Our system consists of several components, each providing a specific set of functionality:

i Database Layer

The database layer is responsible for managing the persistence of data in our system. We use a relational database management system (RDBMS) to store and retrieve data, and we use an Object-Relational Mapping (ORM) framework to map the database schema to our application's data model.



ii Business Logic Layer

The business logic layer is responsible for implementing the core functionality of our application. This layer contains the Model component of the MVC architecture, and is responsible for managing the data and the application logic.

iii Presentation Layer

The presentation layer is responsible for providing the user interface of our application. This layer contains the View and Controller components of the MVC architecture, and is responsible for rendering the data and handling user input.

iv Integration Layer

The integration layer is responsible for integrating our system with other external systems and services. This layer provides interfaces for communicating with other systems, and handles data transformation and mapping between our system and external systems.

v Infrastructure Layer

The infrastructure layer is responsible for providing the underlying infrastructure and services required to run our system. This layer includes components such as web servers, load balancers, and network infrastructure.

3.5 Implementation

The third phase of our development process is implementation. I will use a test-driven development (TDD) approach to implement the software system's components. TDD involves writing automated tests for each component before writing the code for the component. This approach ensures that the software system is reliable, maintainable, and had good test coverage.

I will employ several programming languages and frameworks to implement the software system's components. The user interface will be implemented using the React framework which compiles to HTML, CSS and JavaScript, and the application logic to be implemented using Java and the Java Spring Boot web framework. The database will be implemented using PostgreSQL, a popular open-source RDBMS.

3.6 Testing

The fourth phase of our development process is testing. I will use various testing techniques such as unit testing, integration testing, and acceptance testing to test the software system's components. We also used manual testing to ensure that the software system was user-friendly and met the stakeholders' needs and requirements.

A continuous integration and continuous deployment (CI/CD) approach to automate the testing and deployment process will be implemented. The use of GitHub Actions to automate the testing and deployment process, will allow for quick detection and fixation of any issues that may arise during the development process will also be used.

3.7 Maintenance

The final phase of the development process is maintenance. I will use various techniques such as bug tracking, performance monitoring, and user feedback to maintain the software system. I will also use a DevOps approach to ensure that the software system is reliable, scalable, and secure. I will utilize cloud hosting services such as Amazon Web Services (AWS) and Google Cloud Platform (GCP) to deploy and manage the software system.

Overall, the methodology will ensure and guarantee the delivery of a high-quality software system that meets the stakeholders' needs and requirements. I will follow a collaborative and flexible approach that will allow for quick response to changes and ensure that the software system is reliable, maintainable, and scalable.

4 References

[1] Akter, S., Islam, S., & Ahsan, M. (2019). Adoption of management software among microenterprises in Bangladesh: An empirical study. Information Development, 35(2), 265-280.

[2] Fuchs, C., Huesing, T., & Schäfer, M. (2021). Mobile-based management systems for informal businesses in Tanzania: Effects on business performance and adoption barriers. Journal of Business Research, 129, 496-507.

[3] Yumusak, I. G., Coskun, Y., & Ozkan-Ozen, Y. D. (2021). Factors influencing adoption of management software among small businesses: The case of Turkey. Journal of Enterprise Information Management, 34(1), 27-45.