**VALLEY VIEW UNIVERSITY**

**FACULTY OF SCIENCE**

**COMPUTER SCIENCE DEPARTMENT**

****

**A RESEARCH PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE BACHELOR OF SCIENCE (BSC) IN BUSINESS INFORMATION SYSTEM**

**ONLINE HOTEL BOOKING SYSTEM**

**(A CASE STUDY IN KOFORIDUA)**

**BY**

**KORANTENG AARON AYEH**

**ID: B207230030**

**SUPERVISOR:**

**Collinson Collins Agbesi**

**SEPTEMBER,2024**

# DECLARATION

I, Vinoria Abla Kpeglo hereby declare that the project work entitled “Seasonal Trends Analysis of Mobile Phones sales in Ghana” submitted to the Koforidua Technical University, is a record of an original work done by me under the guidance of Dr. Seth Alornyo of Computer Science, Faculty of Applied Science and Technology. This project work is submitted in partial fulfilment of the requirements for the award of the BSC in Business Information Systems. The results embodied in this project have not been submitted to any other University or Institution for the award of any degree or diploma.

**STUDENT NAME** ---------------------------------------

**SIGNATURE:……………………………………….**

**DATE:……………………………………………….**

# SUPERVISOR’S CERTIFICATION

I hereby certify that this project work was supervised in accordance with the University’s guidelines for supervision of project work.

……………………………………………………………… ………………………………………………………… (Supervisor’s Name) (Date)

# ACKNOWLEDGEMENT

I am deeply grateful to Almighty God for granting me the strength, wisdom, and perseverance to complete this work. His guidance has been my cornerstone throughout this journey. I would also like to extend my heartfelt appreciation to my family for their unwavering love, support, and encouragement. Their belief in me has been a constant source of motivation, and without them, this accomplishment would not have been possible. I wish to express my profound gratitude to my supervisor, Dr. Seth Alornyo, for his invaluable guidance, insightful feedback, and patience throughout this research. His expertise has significantly shaped the direction and quality of this work. Special thanks to the faculty and staff of Koforidua Technical University for providing the academic resources and environment that have facilitated my intellectual growth. Their dedication to education and research excellence has inspired me throughout my studies.

# ABSTRACT

This study presents Hotel Nexus, a modern hotel booking platform designed to simplify and enhance the reservation experience for both guests and hotel staff. Hotel Nexus offers a seamless, user-friendly interface enabling customers to reserve rooms anytime and anywhere, with instant access to real-time room availability and secure payment options. By automating and streamlining booking processes, Hotel Nexus minimizes human errors, reduces staff workload, and ensures a smooth, efficient booking journey. Tailored to meet the growing expectations of today’s tech-savvy travelers, Hotel Nexus helps hotels improve operational efficiency and overall guest satisfaction. Analysis suggests that by leveraging such a platform, hotels can adapt to evolving traveler preferences, potentially leading to increased bookings and enhanced customer loyalty. Implementation of Hotel Nexus could be a strategic move for hotels aiming to boost operational efficiency in a competitive hospitality landscape.

**TABLE OF CONTENTS**

[DECLARATION i](#_Toc180672178)

[SUPERVISOR’S CERTIFICATION ii](#_Toc180672179)

[ACKNOWLEDGEMENT iii](#_Toc180672180)

[ABSTRACT iv](#_Toc180672181)

[LIST OF TABLES ix](#_Toc180672182)

[LIST OF FIGURES x](#_Toc180672183)

[CHAPTER ONE 1](#_Toc180672184)

[GENERAL INTRODUCTION 1](#_Toc180672185)

[1.0 Introduction 1](#_Toc180672186)

[1.1 Subject and Field of Study 3](#_Toc180672187)

[1.2 Study Objectives 6](#_Toc180672188)

[1.2.1 Global Objectives 6](#_Toc180672189)

[1.2.2 Specific Objectives 6](#_Toc180672190)

[1.3 Background to the Study 7](#_Toc180672191)

[1.4 Scope of Study 7](#_Toc180672192)

[1.5 Justification of the study 8](#_Toc180672193)

[1.6 Methodology 8](#_Toc180672194)

[1.7 Expected Results of the Study & Possible Use 10](#_Toc180672195)

[1.8 Presentation of Project (Chapter-by-Chapter Summary of Thesis) 13](#_Toc180672196)

[1.9 Study Work Plan (Timelines) 15](#_Toc180672197)

[CHAPTER TWO 17](#_Toc180672198)

[LITERATURE REVIEW 17](#_Toc180672199)

[2.1 Introduction 17](#_Toc180672200)

[2.2 Concept of Seasonal Trends in Consumer Markets 17](#_Toc180672201)

[2.3 Mobile Phone Industry Overview 20](#_Toc180672202)

[2.4 Factors Influencing Mobile Phone Purchases 23](#_Toc180672203)

[2.4.1 Price Sensitivity 23](#_Toc180672204)

[2.4.2 Brand Loyalty and Trust 25](#_Toc180672205)

[2.4.3 Technological Features 28](#_Toc180672206)

[2.4.4 Peer Influence and Social Norms 31](#_Toc180672207)

[2.4.5 Demographic Factors 34](#_Toc180672208)

[2.4.6 Economic and Social Events 37](#_Toc180672209)

[CHAPTER THREE 41](#_Toc180672210)

[METHODOLOGY 41](#_Toc180672211)

[3.1 Type of Research 41](#_Toc180672212)

[3.2 Population 41](#_Toc180672213)

[3.3 Sample 41](#_Toc180672214)

[3.4 Sampling Technique 42](#_Toc180672215)

[3.5 Data Collection Method 42](#_Toc180672216)

[3.6 Type of Data to be Collected 42](#_Toc180672217)

[3.7 Instrument for Data Collection 42](#_Toc180672218)

[3.8 Instrument Validity and Reliability 43](#_Toc180672219)

[3.9 Instrument Structure to Meet Research Objectives 43](#_Toc180672220)

[3.10 Method of Data Analysis 44](#_Toc180672221)

[3.11 Instrument 44](#_Toc180672222)

[CHAPTER FOUR 46](#_Toc180672223)

[DATA PRESENTATION, FINDINGS AND DISCUSSIONS 46](#_Toc180672224)

[4.1 Demographics 46](#_Toc180672225)

[4.1.0 Age Distribution 46](#_Toc180672226)

[4.1.1 Gender Distribution 47](#_Toc180672227)

[4.1.2 Education Levels 48](#_Toc180672228)

[4.1.3 Employment Status 49](#_Toc180672229)

[4.1.4 Monthly Income 50](#_Toc180672230)

[4.2 Scatter Plot Relationships of the Features in the Data 51](#_Toc180672231)

[4.2.1 Sum of Sales Revenue and Sum of Units Sold by Promotion 51](#_Toc180672232)

[4.2.2 Scatter Plot Relationship of Sales against Month and Promotion 52](#_Toc180672233)

[4.2.3 Scatter Plot Relationship between Sales of Revenue by Model and Brand 54](#_Toc180672234)

[4.2.4 Relationship between Sum of Price by Promotion against Units Sold 55](#_Toc180672235)

[4.3 To analyze the relationship between sales revenue and promotion during various seasonal periods. 57](#_Toc180672236)

[4.3.1 Sales on Regular Days 58](#_Toc180672237)

[4.3.2 Black Friday Sales 60](#_Toc180672238)

[4.3.3 New Year Sales 61](#_Toc180672239)

[4.3.4 Valentine’s Day Sales 63](#_Toc180672240)

[4.3.5 Easter Celebration Sales 64](#_Toc180672241)

[4.3.6 Christmas Celebration Sales 65](#_Toc180672242)

[4.4 To identify sales trends across different months and determine peak periods of mobile phone purchases. 66](#_Toc180672243)

[CHAPTER FIVE 68](#_Toc180672244)

[CONCLUSION AND RECOMMENDATIONS 68](#_Toc180672245)

[5.1 Summary of Main Findings 68](#_Toc180672246)

[5.2 Summary of Key Findings 68](#_Toc180672247)

[5.3 Implications of the Study 69](#_Toc180672248)

[5.4 Conclusion 71](#_Toc180672249)

[5.5 Recommendations 72](#_Toc180672250)

[References 74](#_Toc180672251)

[Appendix A 79](#_Toc180672252)

# LIST OF TABLES

[Table 1:Age Distribution of Respondents 45](#_Toc178581525)

[Table 2:Frequency Distribution Table: Gender 46](#_Toc178581526)

[Table 3:Frequency Distribution Table: Highest Level of Education 47](#_Toc178581527)

[Table 4 Frequency Distribution Table: Employment Status 48](#_Toc178581528)

[Table 5:Frequency Distribution Table: Monthly Income Range 49](#_Toc178581529)

# LIST OF FIGURES

[**Figure 1:Sum of Sales against Sum of Units Sold by Promotion** 51](#_Toc180233203)

[**Figure 2:Scatter Plot Relationship of Sales Against Month and Promotion** 53](#_Toc180233204)

[**Figure 3:Sum of Sales Revenue by Model and Brand** 55](#_Toc180233205)

[**Figure 4:Relationship Between Sum of Price against Solid Units** 56](#_Toc180233206)

[**Figure 5:Sales of Revenue Against Count of Brand** 59](#_Toc180233207)

[**Figure 6: No Celebration** 59](#_Toc180233208)

[**Figure 7: Count of Brand against Promotion** 59](#_Toc180233209)

[**Figure 8 New Year Sales** 60](#_Toc180233210)

[**Figure 9:Valentine Sales** 61](#_Toc180233211)

[**Figure 10:Easter Sales** 62](#_Toc180233212)

[**Figure 11:Christmas Sales** 64](#_Toc180233213)

# CHAPTER ONE

# GENERAL INTRODUCTION

## Introduction

The hospitality industry in Ghana, like many others globally, is experiencing shifts in consumer behavior driven by technological advancements and evolving traveler preferences. Hotel Nexus, a modern hotel booking platform, is positioned to address these changes by simplifying and enhancing the reservation experience for both guests and hotel staff. According to Statistic Brian, every year, around 148.3 million travel bookings are made **(Statistics Brian, 2016)**. There are more and more travelers using the Internet to find hotel information for their holiday ¹. The most popular online hotel booking channels are hotel website and online travel agent website. With this knowledge and by offering a seamless, user-friendly interface that enables customers to reserve rooms anytime and anywhere, Hotel Nexus caters to the growing expectations of today’s tech-savvy travelers. The platform provides instant access to real-time room availability and secure payment options, streamlining the booking process and minimizing human errors associated with manual booking methods. As hotels seek to improve operational efficiency and boost guest satisfaction in a competitive landscape, solutions like Hotel Nexus play a critical role. Understanding the dynamics of hotel booking trends, consumer preferences, and the impact of digital platforms on hospitality operations is essential for stakeholders in the industry. Hotel Nexus facilitates business operations by allowing hotels to manage bookings efficiently, enables access to a broader customer base through online channels, and supports hotels in enhancing their service delivery through streamlined processes. Moreover, the adoption of such platforms has become central to how hotels interact with guests, manage reservations, and optimize revenue management ².

## 1.1 Subject and Field of Study

This research work falls under the field of Computing, with a specific focus on Information Systems Development for the hospitality and tourism industry. Information and Communication Technologies (ICTs) are a necessary component of business culture (Baggio, 2004). In today’s world using ICTs is no more a distinctive characteristic by itself (Baggio, 2004). Which integrates principles like that of web application development, database management, and human & computer interaction to create a functional, user-friendly hotel booking platform. The study also aligns with Management Information Systems (MIS), as it seeks to enhance operational efficiency, decision-making, and customer service through the deployment of a digital reservation solution.

The hospitality industry in Ghana is navigating a strong recovery following the COVID-19 pandemic. A notable indicator of this rebound is the more than 100% increase in average tourist expenditure, rising from US$1,300 in 2021 to US$2,700 in 2022 ([GNA, 2023](https://gna.org.gh/2023/07/ghanas-tourism-and-hospitality-industry-needs-support-to-grow-dr-joseph-mensah/?utm_source=chatgpt.com)). The accommodation sector alone contributes 31% of total tourism revenues, with the food and beverage segment adding 11%, making hospitality a major driver of Ghana’s tourism economy.

Industry experts; including Dr. Joseph Mensah-Ansah, emphasize that digitalization is key to sustaining this growth, as it enables hotels to efficiently capture client information, streamline operations, and deliver better service. In hotels/hospitality services areas, some functions which can be performed with computers through the aid of management software are: data, yield and reservation management, accounting for guests, forecasting guest demand for reservations and management of guest services (Olifer, 2006) (Measuring Customer Relationship Management in the Hospitality Industry of Some Selected Hotels in Accra, 2013). Hotel Nexus directly addresses this industry need by providing a robust, intuitive, and accessible online booking system that connects guests and hotels seamlessly, supports operational efficiency, and enhances overall customer experience.

## 1.2 Study Objectives

## 1.2.1 Global Objectives

To design and develop an online hotel booking system (Hotel Nexus) that streamlines room reservations, enhances customer convenience, and improves operational efficiency within the hospitality sector in Ghana.

## 1.2.2 Specific Objectives

The specific objectives of the study are to:

1. Develop a web-based booking platform that enables customers to search, view, and reserve hotel rooms in real time.
2. Implement secure user registration and login features for both customers and hotel administrators.
3. Integrate a secure payment gateway that supports multiple payment options, including credit/debit cards and mobile money.
4. Enable automated email or SMS notifications for booking confirmations, cancellations, and reminders.
5. Provide hotel administrators with a dashboard to manage room availability, reservations, and customer records.
6. Ensure the system is responsive and accessible across desktop and mobile devices.
7. Minimize manual intervention in booking processes to reduce errors and staff workload.

## 1.3 Background to the Study

With the hospitality industry plays a significant role in Ghana’s economy, contributing substantially to both tourism revenues and employment. According to the Ghana News Agency (2023), the accommodation sector alone accounts for approximately **31%** of total tourism earnings, with the food and beverage industry contributing **11%**. In recent years, it has been seen that the sector has experienced a resurgence, with average tourist expenditure increasing from US$1,300 in 2021 to US$2,700 in 2022. This growth is driven by rising domestic and international tourism, government initiatives to promote Ghana as a tourist destination, and improvements in infrastructure.

However, despite this progress, some hotels in Ghana still rely on manual or semi-digital reservation processes, which can lead to inefficiencies such as double bookings, delayed confirmations, and limited customer engagement. Guests often need to call or email hotels to check room availability or make reservations, a process that is time-consuming for both customers and hotel staff.

The shift in consumer expectations toward convenience, speed, and 24/7 service fueled by the widespread use of smartphones and internet access has made the adoption of online hotel booking systems increasingly necessary. Globally, digital reservation platforms have been shown to improve operational efficiency, reduce administrative workload, and enhance customer satisfaction. In Ghana, however, the adoption of such systems remains inconsistent, particularly among small- and medium-sized hospitality businesses.

*Hotel Nexus* is proposed as a modern, user-friendly online booking platform tailored to the Ghanaian hospitality landscape. The system aims to bridge the gap between guest expectations and current hotel operations by offering real-time room availability, secure payment options, automated notifications, and an intuitive interface accessible via both desktop and mobile devices. By digitizing the booking process, *Hotel Nexus* not only addresses operational challenges faced by hotels but also aligns with the broader national goal of enhancing the competitiveness of Ghana’s tourism sector in the global market.

## 1.4 Scope of Study

The scope of this study is to cover the design, development, and evaluation of an online hotel booking system, *Hotel Nexus*, tailored to meet the needs of hotels operating within Ghana’s hospitality industry. The system will primarily focus on enabling customers to perform the following tasks:

* Create and manage personal user accounts.
* Search and filter available rooms based on specific criteria such as room type, price range, and occupancy.
* View real-time room availability and booking status.
* Make secure online reservations with instant confirmation.
* Complete payments through integrated payment gateways supporting credit/debit cards and mobile money.
* Receive automated email or SMS notifications for booking confirmations, modifications, and cancellations.

On the administrative side, the system will allow hotel staff to:

* Manage room listings and update availability in real time.
* View and approve incoming bookings.
* Generate invoices and reports for operational and decision-making purposes.
* Monitor customer interactions and maintain booking records securely.

The scope is limited to developing a **web-based platform** optimized for desktop and mobile browsers and is the goto for most tech industry as it is accessible anywhere and by almost any device, most embedded systems, phones and desktop. Sole Mobile application development, while feasible in future iterations, is not part of the current study. Additionally, the system will focus on functionality relevant to small- and medium-sized hotels in Ghana, with features designed for scalability should larger establishments wish to adopt the platform.

## 1.5 Justification of the study

The development of *Hotel Nexus* is justified by the growing demand for efficient, accessible, and user-friendly hotel booking systems in Ghana’s hospitality sector. While the tourism industry is expanding and contributing significantly to the national economy, many hotels particularly small and medium sized establishments still rely on outdated or manual booking processes. These methods often lead to operational inefficiencies such as booking errors, delayed confirmations, and excessive administrative workload, which can negatively impact customer satisfaction and retention.

The introduction of *Hotel Nexus* addresses these challenges by providing:

1. **Enhanced Customer Experience** – Guests can search, book, and pay for rooms online at any time, with instant confirmation and automated notifications, reducing the need for phone or email inquiries.
2. **Operational Efficiency for Hotels** – Automated booking management reduces manual work for staff, minimizes human errors, and improves overall workflow.
3. **Support for Ghana’s Tourism Growth** – By modernizing hotel operations, the system aligns with national efforts to improve service quality in the tourism and hospitality industry, making Ghana more competitive as a travel destination.
4. **Secure and Scalable Solution** – The platform integrates secure payment systems and is designed for scalability, enabling it to serve both small hotels and larger chains in the future.
5. **Alignment with Global Trends** – As more travelers globally turn to online booking platforms, *Hotel Nexus* ensures that Ghanaian hotels remain competitive in an increasingly digital marketplace.

Ultimately, this study contributes to bridging the gap between customer expectations and existing hotel booking systems in Ghana, while supporting the broader goal of enhancing service delivery and economic growth in the hospitality industry.

## 1.6 Methodology

**Research Approach**

This study adopts an applied research approach, combining both theoretical and practical perspectives to design and develop a functional online hotel booking system for the Ghanaian hospitality industry. The project follows a System Development Life Cycle (SDLC) framework, specifically the Agile Development Model, to ensure iterative design, testing, and improvement based on feedback. This model is chosen because it allows flexibility, quick adaptation to changes, and continuous engagement with stakeholders (hotel managers and customers) throughout the development process.

**Data Collection**

In order gather relevant information on the requirements and expectations for Hotel Nexus, questionnaires were used as the primary method of data collection. This method was chosen because it allows the collection of structured responses from a large sample in a short period, ensuring both hotel staff and customers can contribute valuable insights.

**Primary Data**

* **Hotels:** Questionnaires were sent to hotel managers and front desk staff to identify operational challenges in the current booking process, required features, and preferred system functionalities.
* **Customers:** Questionnaires were distributed to past and potential hotel guests to gather opinions on booking preferences, pain points, and essential features for an online booking platform.

Sample of the questionnaire can be found in the appendix A section

**Data Analysis**

After collecting the completed questionnaires, the data will be analyzed using **descriptive statistical methods** such as percentages, frequency counts, and charts to identify patterns in responses. The analysis will focus on:

* Common operational challenges faced by hotels.
* Most requested features by both hotel staff and customers.
* The level of acceptance and readiness for adopting an online booking system.

Findings from the data analysis will directly influence the design decisions for *Hotel Nexus*, ensuring that the final system meets the real needs of its intended users.

**Quantitative Analysis**

The research incorporated a quantitative analysis approach to process and interpret the data collected from the questionnaires. Using Closed ended questions were used to ensure that responses could be easily categorized and analyzed statistically. The data gathered was coded and entered into Microsoft Excel/Google Sheets for processing. Descriptive statistics such as frequencies, percentages, and charts were generated to identify patterns and trends. This approach enabled the researcher to quantify responses and make clear comparisons between the views of hotel staff and customers regarding the booking process.

**Sampling**

The study employed a **purposive sampling technique**, targeting respondents who are directly involved in hotel booking processes, either as **hotel staff/managers** or as **hotel guests/customers**.

A total of **60 questionnaires** were distributed:

* **30** to hotel staff and managers from selected hotels within the Eastern Region, Ghana.
* **30** to hotel customers who had booked rooms within the last 12 months.

Out of the 60 questionnaires distributed, **52** were completed and returned, representing an **86.7% response rate**. This high return rate ensured that the sample size was sufficient for meaningful quantitative analysis while capturing diverse perspectives from both stakeholders in the hotel booking process.

|  |  |  |  |
| --- | --- | --- | --- |
| Respondent Category | Questionnaires Distributed | Questionnaires Returned | Response Rate |
| Hotel Staff/Managers | 30 | 27 | 90% |
|  |  |  |  |
| Hotel Customers | 30 | 25 | 83.3% |
| Total | **60** | **52** | **86.7%** |

## 1.7 Expected Results of the Study & Possible Use

The expected results of this study include the successful design, development, and deployment of *Hotel Nexus*, an online hotel booking system that streamlines the reservation process for both customers and hotel administrators. The system is anticipated to deliver the following outcomes:

1. **Improved Booking Efficiency** – Customers will be able to make, modify, and cancel reservations online in real time, reducing delays and reliance on manual processes.
2. **Enhanced Customer Experience** – The platform will provide an intuitive and user-friendly interface accessible across devices, enabling 24/7 access to hotel services.
3. **Secure Transactions** – Integration of a secure payment gateway to facilitate multiple payment methods, including credit/debit cards and mobile money, ensuring safe and reliable transactions.
4. **Operational Optimization** – Hotel administrators will have access to a centralized dashboard for managing room availability, reservations, customer data, and reports, thereby reducing workload and errors.
5. **Automated Communication** – The system will automatically send booking confirmations, reminders, and cancellations via email or SMS, improving customer communication and satisfaction.
6. **Data-Driven Decision Making** – The system will generate analytics and reports that can help hotel management understand customer trends, optimize pricing, and plan promotions.

**Possible Use**

* For **hotels**, the system will serve as an operational tool to manage bookings more effectively and reach a wider customer base through online channels.
* For **customers**, it will be a convenient platform to browse, book, and manage reservations anytime and anywhere.
* For **research purposes**, the findings from the study and the system’s implementation can serve as a reference for further technological innovations in the hospitality industry in Ghana and similar markets.

## 1.8 Presentation of Project (Chapter-by-Chapter Summary of Thesis)

**Chapter One – General Introduction**  
This chapter introduces the project, providing the background to the study, the problem statement, objectives, scope, justification, and significance of the study. It also outlines the research methodology, expected results, work plan, and overall structure of the thesis.

**Chapter Two – Literature Review**  
This chapter reviews existing literature on hotel booking systems, online reservation technologies, and trends in the hospitality industry. It includes an analysis of similar systems, their features, strengths, and weaknesses, and identifies gaps that *Hotel Nexus* intends to fill.

**Chapter Three – Crystallization of the Proposed System**  
This chapter focuses on the conceptualization of *Hotel Nexus*, explaining how the proposed system addresses the identified gaps in existing solutions. It also describes the system objectives, overall architecture, and the intended functionalities to be implemented.

**Chapter Four – Analysis of the Proposed System**  
This chapter presents a detailed requirements analysis, covering both functional and non-functional requirements. It also includes stakeholder analysis, system specifications, and process modeling using tools such as Data Flow Diagrams (DFD) and Use Case Diagrams.

**Chapter Five – Detailed Design of the Proposed System**  
This chapter covers the system’s technical design, including database schema design, architectural diagrams, interface designs, and module specifications. It explains how different system components will interact to achieve the desired functionalities.

**Chapter Six – System Implementation and Testing**  
This chapter describes the technologies and tools used to develop *Hotel Nexus*. It covers the coding process, integration of functionalities such as real-time room availability and payment processing, and outlines the testing procedures, results, and system performance evaluations.

**Chapter Seven – System Documentation**  
This chapter provides comprehensive documentation for *Hotel Nexus*, including user manuals, administrator guides, installation instructions, and maintenance procedures to support system deployment and usage.

**Chapter Eight – Conclusion and Recommendations**  
This chapter summarizes the study, highlights the key achievements, and presents conclusions drawn from the system’s development and implementation. It also offers recommendations for future improvements, scalability, and research in the domain of online hotel booking systems.

**Table 1. 1:** Gantt Chart Outline for Thesis on Hotel Nexus – Online Hotel Booking System

|  |  |  |  |
| --- | --- | --- | --- |
| Activity / Chapter | Month 1 | Month 2 | Month 3 |
| Chapter 1: General Introduction | █████ |  |  |
| Chapter 2: Literature Review | █████ | ███ |  |
| Chapter 3: Crystallization of the Proposed System | ███ | █████ |  |
| Chapter 4: Analysis of the Proposed System |  | █████ | █ |
| Chapter 5: Detailed Design of the Proposed System |  | █████ | ███ |
| Chapter 6: System Implementation and Testing |  | ███ | █████ |
| Chapter 7: System Documentation |  |  | █████ |
| Chapter 8: Conclusion and Recommendations |  |  | ███ |
| Final Editing & Submission |  |  | █████ |

**Legend:**

* █ (Block) indicates weeks of activity.
* Activities may overlap, especially for research, design, and documentation stages.

## 1.9 Study Work Plan (Timelines)

# CHAPTER TWO

# LITERATURE REVIEW

## 2.1 Introduction

The hospitality industry has increasingly embraced digital solutions to streamline operations, improve customer experiences, and maintain competitiveness in a rapidly evolving market (Buhalis & Leung, 2018). One of the most critical components of this transformation is the hotel booking system, which enables customers to search, reserve, and pay for accommodations online without the need for manual processes. These systems provide benefits such as real-time room availability, automated payment processing, and customer data management (Law et al., 2015).

In this section, two existing computer-based hotel management and booking systems are reviewed. The aim is to identify their strengths, weaknesses, and potential gaps, thereby positioning the proposed Hotel Nexus system as a more advanced and adaptable solution to meet modern hospitality needs.

### 2.2 Review of Existing Systems

#### System 1: Online Computerized Hotel Management System (HMS) (Sanni Abubakar Omuya, April 2, 2014.)

Description:  
The Online Computerized Hotel Management System was developed for Satellite Motel, Ilorin, Nigeria, to automate hotel operations. It supports online reservations, real-time availability checks, secure payments, and database-driven record management. The goal was to improve operational efficiency and customer satisfaction by replacing manual processes.

Features:

* User-friendly reservation interface.
* Real-time room availability.
* Automated billing and payment processing.
* Room allocation and tariff management.
* Centralized database for guest and booking details.
* Administrative tools for stock control and reporting.

Development-Tools:  
Java (programming), Internet Information Service (IIS) (web server), and a custom relational database design. Survey data was analyzed using Microsoft Excel.

Strengths:

* Improves booking speed and accuracy.
* Provides 24/7 booking access.
* Reduces paperwork and manual errors.
* Enhances privacy and security.

Weaknesses:

* Designed for a single hotel (lacks scalability).
* Relies heavily on network connectivity.
* No modern payment gateway integrations.

#### System 2: Online Hotel Booking System for Hotel LaVilla (Prarthana, 2017)

Description:  
This is a web-based platform that automates hotel searches, availability checks, reservations, and booking management. It caters to both customers and administrators with dedicated interfaces.

Features:

* User registration and login.
* Hotel search and filtering by location, amenities, and price.
* Real-time availability check.
* Online reservation and cancellation.
* Automated booking confirmation.
* Admin dashboard for hotel management.
* Secure payment integration.

Development Tools:  
PHP (programming), MySQL (database), HTML/CSS/JavaScript (frontend), Apache (server).

Strengths:

* Improves customer convenience through 24/7 access.
* Real-time booking reduces overbooking risks.
* User-friendly interface.
* Secure transactions.

Weaknesses:

* Scalability limitations for multi-hotel integration.
* Security depends on correct payment integration.
* Requires stable internet access for full functionality.

### 2.3 Comparative Analysis of Reviewed Systems

Both systems provide the core functionalities of modern hotel booking platforms, including real-time availability, online payments, and reservation management. However, there are notable differences in their architecture, scalability, and technological adaptability:

* Technology Stack: The HMS uses Java and IIS, making it more suited to enterprise-level customization, while Hotel LaVilla’s PHP and MySQL stack is widely adopted for affordable web development.
* Scalability: Neither system is well-optimized for a multi-hotel ecosystem, limiting their use for hotel chains or large networks.
* Security: Both systems have security provisions, but there is limited documentation on advanced encryption, fraud detection, or integration with modern payment APIs such as PayPal or Stripe.
* Mobile Accessibility: While both claim to be accessible on mobile devices, neither emphasizes mobile-first design principles or dedicated mobile applications, which are increasingly demanded by modern travelers).

### 2.4 Identified Gaps in Current Systems

From the analysis, the following gaps have been identified:

1. Multi-Hotel Scalability: Neither system adequately supports multiple hotels within the same platform.
2. Advanced Payment Integration: Modern, globally accepted payment gateways are absent.
3. Mobile-First Approach: There is little emphasis on mobile-first responsive design or dedicated mobile apps.
4. Data Analytics and Insights: Neither system includes comprehensive analytics for decision-making.

### 2.5 Relevance to Hotel Nexus

The Hotel Nexus system will address these gaps by:

* Providing a multi-hotel booking platform capable of scaling to multiple properties.
* Integrating modern payment solutions with multi-currency support.
* Ensuring mobile-first responsive design for better accessibility.
* Incorporating analytics dashboards for hotel administrators to track performance and customer behavior.

By bridging these gaps, Hotel Nexus aims to create a competitive, future-proof hotel booking solution for the Ghanaian hospitality industry

# CHAPTER THREE

# CRYSTALLIZATION OF THE RESEARCH PROBLEM

### 3.1 Background / History of the Case Study Organization

Looking at the hospitality industry in Ghana there has seen steady growth over the last decade, particularly in urban areas and tourist destinations such as Accra, Kumasi, Cape Coast, and Takoradi. Many hotels in Ghana, especially small to mid-sized establishments, still rely heavily on manual booking systems or basic spreadsheets for reservation management. While these methods are functional for day-to-day operations, they are prone to human error, inefficiency, and delays.

The selected case study organization, here referred to as XYZ Hotel (a mid-sized hotel located in Accra), currently operates with a hybrid booking process — a combination of manual record-keeping and partial digital tools. This approach has created operational bottlenecks and limited the hotel’s ability to meet the expectations of tech-savvy customers seeking instant online reservations.

### 3.2 Current System Operations

#### 3.2.1 Components / Features of the Current System

The current booking process includes the following:

1. Manual Customer Inquiries: Customers contact the hotel via phone, email, or walk-in to inquire about room availability.
2. Paper-Based Reservation Ledger: Bookings are recorded manually in a physical ledger or Excel spreadsheet.
3. Room Allocation: Rooms are assigned manually based on the ledger and staff availability.
4. Payment Processing: Payments are taken in cash, via bank transfer, or using POS terminals. Online payments are not integrated.
5. Confirmation Process: Confirmation receipts are manually generated and sent via email or printed for collection.
6. Cancellations / Changes: These are handled manually through direct communication with hotel staff, with adjustments made in the ledger.

#### 3.2.2 Process Flow (Manual System)

Textual Process Description:

Customer inquiry → Staff checks availability in ledger/spreadsheet → Reservation entry recorded → Payment collected → Confirmation issued → Booking status updated manually.

### 3.3 Strengths and Weaknesses of the Current System

Strengths:

* Simple to use for staff without advanced technical skills.
* Low cost of operation, as it relies on basic tools like paper and Excel.
* No dependency on high-speed internet connectivity.

Weaknesses:

* No Real-Time Availability: Cannot instantly show updated room status to customers.
* High Risk of Double Booking: Manual entries can result in conflicts.
* Limited Accessibility: Bookings can only be made during working hours.
* Poor Data Analytics: No automated reporting or tracking of booking trends.
* Inadequate Payment Options: No integrated online payment systems.
* High Operational Load: Staff spend significant time on administrative booking tasks.

### 3.4 Comparative Study with Literature

Comparing the current system with existing systems reviewed in Chapter Two reveals significant gaps:

* Unlike the Online Computerized Hotel Management System, XYZ Hotel’s current system lacks automated billing, stock control, and a centralized database.
* Compared to the Online Hotel Booking System for Hotel LaVilla, XYZ Hotel’s system does not offer customer self-service, online cancellation, or secure payment integration.
* While both existing systems in literature allow for real-time room availability checks, XYZ Hotel relies solely on manual ledger reviews, which are slower and prone to error.
* None of the advanced features such as automated notifications, analytics dashboards, or multi-hotel scalability found in modern solutions are present in the current system.

This comparison shows that Hotel Nexus must fill the technological gaps by introducing real-time booking, secure payments, customer self-service, and robust administrative tools to significantly improve operational efficiency and customer satisfaction.

# CHAPTER FOUR

# ANALYSIS OF THE PROPOSED SYSTEM

### 4.0 Overview of the Proposed System

The proposed system, Hotel Nexus, is an advanced, web-based and mobile-responsive hotel booking platform designed to automate and streamline the reservation process for both guests and hotel administrators. The system will provide real-time room availability checks, secure payment processing, automated booking confirmations, and a comprehensive administrative dashboard for hotel management.

Hotel Nexus will address the inefficiencies and limitations of the current manual and semi-manual booking methods by introducing a unified platform where guests can register, search, filter, reserve, modify, and cancel bookings independently, while hotel staff manage operations through a secure back-office portal. The system will be built to ensure high availability, data security, and scalability to support future expansions.

#### Functional Requirements

Functional requirements define what the system must do:

1. User Registration and Authentication – Guests and administrators can securely register and log in.
2. Room Search and Filtering – Search based on room type, availability, occupancy, and price range.
3. Real-Time Availability Check – Instant updates on room status to prevent double booking.
4. Booking and Reservation Management – Ability to reserve, modify, and cancel bookings online.
5. Payment Processing – Integration with secure payment gateways (mobile money, card payments, etc.).
6. Automated Notifications – Email and SMS booking confirmations, reminders, and updates.
7. Administrative Dashboard – Tools for staff to manage bookings, room details, customer records, and reports.
8. Reporting and Analytics – Generate occupancy rates, booking trends, and revenue reports.

#### Non-Functional Requirements

These specify how the system should operate:

1. Performance – System should process booking requests within 3 seconds.
2. Availability – 99.9% uptime to support 24/7 online reservations.
3. Scalability – Support multiple hotels and expansion of features.
4. Security – Encrypted transactions, secure authentication, and role-based access control.
5. Usability – User-friendly interface optimized for desktop and mobile devices.
6. Maintainability – Modular code structure for easy updates and maintenance.

### 4.1 Major Features / Components of the Proposed System

* Guest Portal: Self-service interface for customers to browse, book, and manage reservations.
* Admin Portal: Secure dashboard for managing rooms, pricing, bookings, and reports.
* Search and Filter Module: Advanced search options for faster room discovery.
* Payment Integration: Multiple payment methods including credit/debit cards, mobile money, and PayPal.
* Notification System: Automated booking confirmations and reminders.
* Analytics Module: Visual representation of booking trends and revenue statistics.

### 4.2 Benefits / Advantages of the Proposed System

* Increased Efficiency: Reduces manual workload for staff through automation.
* 24/7 Accessibility: Guests can make bookings anytime, anywhere.
* Reduced Errors: Real-time updates eliminate double booking risks.
* Enhanced Customer Experience: Easy navigation, instant confirmation, and multiple payment options.
* Improved Decision-Making: Data analytics support strategic planning and marketing.
* Secure Transactions: Protects customer data and payments.

### 4.3 System Context Diagram of the Proposed System

# CHAPTER FIVE

## DETAILED DESIGN OF THE PROPOSED SYSTEM

# 5.0 Functional Processes of the Proposed System (Hotel Nexus)

This chapter describes the core functional processes of Hotel Nexus: booking, payment, notifications, admin management, and reporting. Each major process is broken down into algorithmic steps and visual models to illustrate data and control flow between users and subsystems.

## 5.1 Algorithms / Flowchart of the System

### 5.1.1 High-level Booking Algorithm (pseudocode)

1. Customer browses rooms and selects dates

2. System checks room availability for selected dates

3. If rooms available:

a. Customer fills booking form and selects payment method

b. System creates a provisional reservation (holds room for short time)

c. System sends payment request to Payment Gateway

d. If payment confirmed:

i. Mark reservation as confirmed

ii. Send confirmation notification

iii. Update room availability

iv. Log transaction and update admin dashboard

e. Else (payment failed):

i. Release provisional reservation

ii. Notify customer of failure and prompt retry

4. Else (no rooms available):

a. Offer alternative dates or room types

b. Allow customer to join waitlist or be notified when room is free

### 5.1.2 Mermaid Flowchart (Booking)

flowchart TD

A[Customer selects dates & room] --> B{Availability?}

B -- Yes --> C[Create provisional reservation]

C --> D[Collect payment info]

D --> E[Send payment request to Payment Gateway]

E --> F{Payment status}

F -- Confirmed --> G[Mark reservation confirmed]

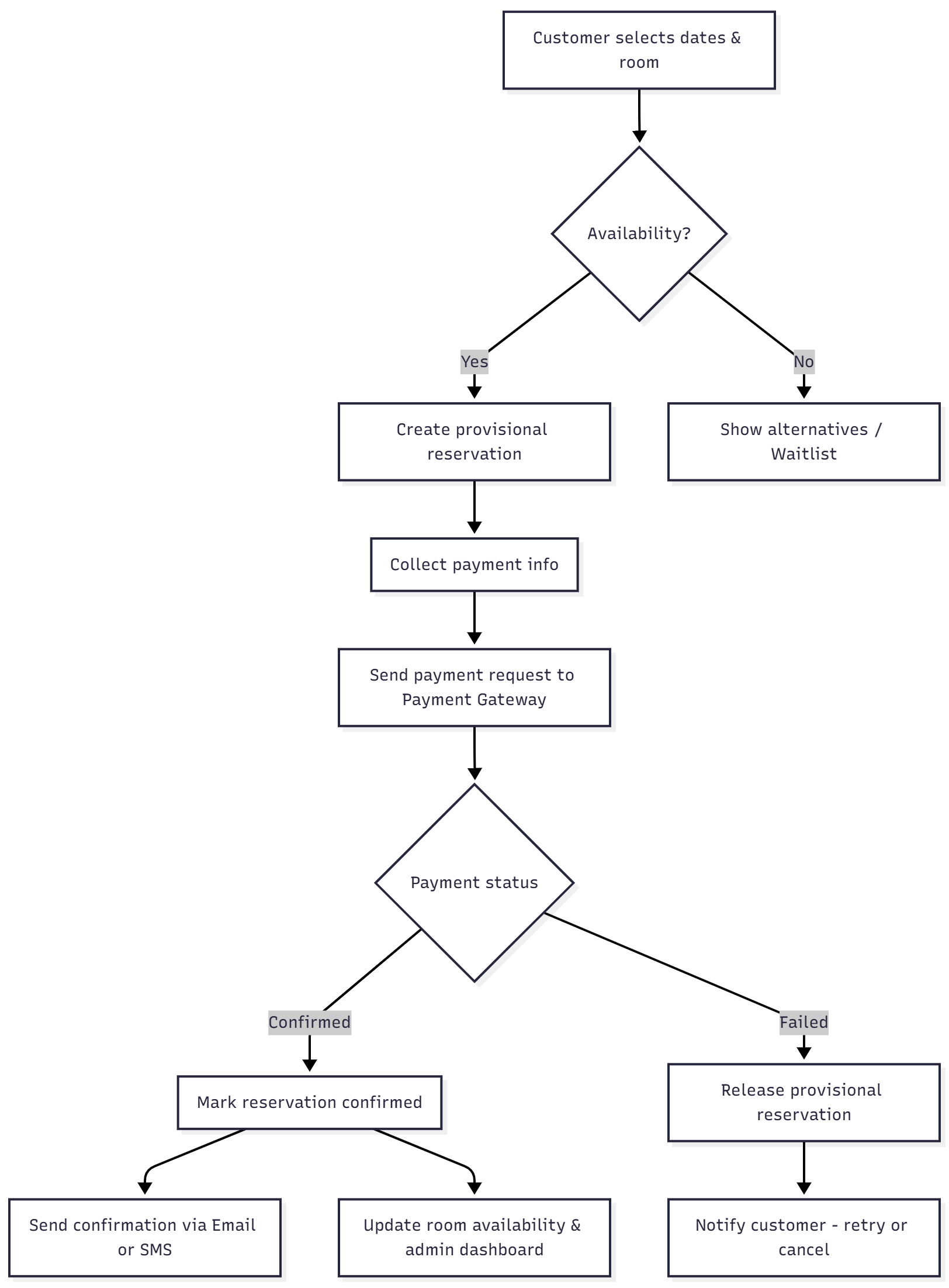
G --> H[Send confirmation (Email/SMS)]

G --> I[Update room availability & admin dashboard]

F -- Failed --> J[Release provisional reservation]

J --> K[Notify customer -> retry or cancel]

B -- No --> L[Show alternatives / Waitlist]



## 5.2 Process Models

### 5.2.1 System Context Diagram (Text + Mermaid)

External entities: Customer, Hotel Admin/Staff, Payment Gateway, Notification Service

Mermaid flowchart:

flowchart LR

Customer["Customer/Guest"] -->|"Booking Request / Payment"| HotelNexus["Hotel Nexus System"]

HotelAdmin["Hotel Admin/Staff"] -->|"Room Updates / Manage Bookings"| HotelNexus

HotelNexus -->|"Payment Request"| PaymentGateway["Payment Gateway"]

PaymentGateway -->|"Payment Confirmation/Failure"| HotelNexus

HotelNexus -->|"Send Notifications"| NotificationSvc["Notification Service (Email/SMS)"]

HotelNexus -->|"Reports / Booking Data"| HotelAdmin

ThirdParty["3rd-Party OTAs"] -->|"Channel Bookings / Sync"| HotelNexus

### 5.2.2 Data Flow Diagrams (DFD)

#### DFD Level 0 (Context-level) — single process

flowchart TD

Customer -->|Booking Request| HotelNexus((Hotel Nexus))

HotelNexus -->|Confirmation / Invoice| Customer

HotelAdmin -->|Room Data / Manage| HotelNexus

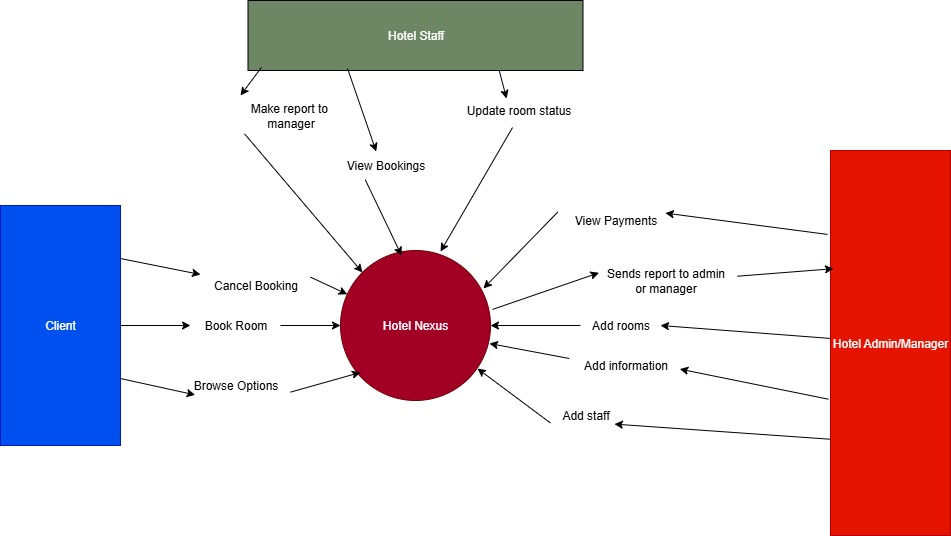
HotelNexus -->|Reports / Stats| HotelAdmin

HotelNexus -->|Payment Request| PaymentGateway

PaymentGateway -->|Payment Result| HotelNexus

HotelNexus -->|Notification Request| NotificationService

NotificationService -->|Deliver Message| Customer



#### DFD Level 1 — major subprocesses

The **Hotel Nexus System** serves as the central hub connecting three main external entities:

1. **Client (Customer)**
   * **Inputs:**
     + Browse room options
     + Book a room
     + Cancel booking
   * **Outputs (from the system):**
     + View booking confirmation and available room information
2. **Hotel Staff**
   * **Inputs:**
     + Update room status (e.g., available, occupied, under maintenance)
     + Make reports to the manager
   * **Outputs (from the system):**
     + View bookings
3. **Hotel Admin / Manager**
   * **Inputs:**
     + Add rooms, staff, and general hotel information
     + Receive system-generated reports from Hotel Nexus
   * **Outputs (from the system):**
     + View payments and overall system analytics

#### DFD Level 2 — expand Booking & Payment

Customer --> A[Submit booking details]

A --> B[Validate input & check availability]

B -->|available| C[Create provisional reservation]

C --> D[Generate payment request]

D --> PaymentGateway

PaymentGateway --> E[Process payment]

E -->|success| F[Confirm reservation]

F --> G[Send confirmation (email/sms)]

E -->|failure| H[Cancel provisional reservation]

H --> I[Notify customer]

F --> J[Update admin reports & ledger]

### 5.2.3 Decision Tree (Booking vs Waitlist)

A simple Mermaid decision tree (use graph LR):

Start[Start Booking] --> Check[Check Availability]

Check -- Available --> Pay[Proceed to Payment]

Check -- Not Available --> Offer[Offer Alternatives]

Offer --> Waitlist[Join Waitlist?]

Waitlist -- Yes --> SaveWaitlist[Save contact & notify]

Waitlist -- No --> End[End]

### 5.2.4 UML Sequence Diagram (Customer booking)

sequenceDiagram

participant Customer

participant WebApp as HotelNexus

participant DB as Database

participant PG as PaymentGateway

participant NS as NotificationService

participant Admin

Customer->>WebApp: Search rooms (dates)

WebApp->>DB: Query availability

DB-->>WebApp: Return availability

alt rooms available

WebApp->>WebApp: Create provisional reservation (hold)

WebApp->>Customer: Collect payment details

Customer->>PG: Submit payment (via WebApp proxy)

PG-->>WebApp: Payment success/fail

alt success

WebApp->>DB: Mark reservation confirmed

WebApp->>NS: Send confirmation (email/sms)

NS-->>Customer: Deliver message

WebApp->>Admin: Update dashboard

else failure

WebApp->>DB: Release provisional reservation

WebApp->>Customer: Notify payment failure

end

else no rooms

WebApp->>Customer: Offer alternatives/waitlist

end

## 5.3 Data Models

### 5.3.1 Entity Relationship Overview

Primary entities:

* User (customer)
* Hotel
* Room
* Reservation
* Payment
* Notification
* Admin/Staff
* RoomType
* Amenity

#### ERD (textual description)

* A User can have many Reservations.
* A Hotel has many Rooms.
* A Room belongs to a RoomType and can have many Reservations (over time).
* A Reservation has one Payment record (or multiple for partial payments).
* Notification logs are linked to Reservation and User.

(If you want a visual ERD, use an ERD tool or Mermaid class-like graph; I'll provide a simple Mermaid relation graph.)

Mermaid simple ER-like graph:

erDiagram

USER ||--o{ RESERVATION : makes

HOTEL ||--o{ ROOM : contains

ROOM ||--o{ RESERVATION : booked\_for

RESERVATION ||--|| PAYMENT : has

USER ||--o{ NOTIFICATION : receives

ROOM }|--|| ROOMTYPE : is\_of

### 5.3.2 Data Dictionary (sample)

Provide selected fields and descriptions:

#### • Client

* client\_id (PK): INT
* first\_name: VARCHAR(50)
* middle\_name: VARCHAR(50)
* last\_name: VARCHAR(50)
* gender: ENUM('male', 'female', 'other')
* email: VARCHAR(100)
* phone: VARCHAR(15)
* password: VARCHAR(255)

#### • Rooms

* room\_id (PK): INT
* room\_number: VARCHAR(20)
* type: VARCHAR(50)
* status: ENUM('available', 'occupied', 'maintenance')
* price: DECIMAL(10,2)
* description: TEXT
* amenities: ARRAY(VARCHAR)
* capacity: INT
* rating: DECIMAL(2,1)
* services: ARRAY(VARCHAR)
* image: ARRAY(IMAGE\_URL)

#### Booking

* booking\_id (PK): INT
* client\_id (FK): INT → references Client.client\_id
* room\_id (FK): INT → references Rooms.room\_id
* check\_in: DATE
* check\_out: DATE
* status: ENUM('booked', 'checked\_in', 'checked\_out', 'cancelled')
* created\_at: DATETIME
* Payment
  + payment\_id (PK): INT
  + reservation\_id (FK): INT
  + amount: DECIMAL(10,2)
  + status: ENUM('pending','completed','failed','refunded')
  + provider\_ref: VARCHAR(255)
  + paid\_at: TIMESTAMP

### 5.3.3 Database Schema & Tables (sample SANITY)

**Client**

export const client = defineType({

    name: "client",

    title: "Client",

    type: "document",

    fields: [

        defineField({

            name: "first\_name",

            type: "string",

        }),

        defineField({

            name: "middle\_name",

            type: "string"

        }),

        defineField({

            name: "last\_name",

            type: "string"

        }),

        defineField({

            name: "gender",

            type: "string"

        }),

        defineField({

            name: "email",

            type: "string"

        }),

        defineField({

            name: "phone",

            type: "string"

        }),

        defineField({

            name: "password",

            type: "string"

        })

    ],

   preview: {

    select: {

        first\_name: "first\_name",

        last\_name: "last\_name",

    },

    prepare(selection) {

        const { first\_name, last\_name } = selection;

        return {

            title: `${first\_name} ${last\_name}`,

            subtitle: selection.gender,

        };

    }

   }

})

**ROOMS**

export const rooms = defineType({

  name: "rooms",

  title: "Rooms",

  type: "document",

  icon: () => "🛏️",

  fields: [

        defineField({

            name: "image",

            title: "Image",

            type: "array",

            of: [{ type: "image" }],

            options: { hotspot: true },

        }),

        defineField({

            name: "room\_number",

            title: "Room Number",

            type: "string",

        }),

        defineField({

            name: "type",

            title: "Type",

            type: "string",

        }),

        defineField({

            name: "status",

            title: "Status",

            type: "string",

            options: {  list: [

                { title: 'Available', value: 'available' },

                { title: 'Occupied', value: 'occupied' },

                { title: 'Maintenance', value: 'maintenance' }, ]

            },

        }),

        defineField({

            name: "price",

            title: "Price",

            type: "number",

        }),

        defineField({

            name: "description",

            title: "Description",

            type: "text",

        }),

        defineField({

            name: "amenities",

            title: "Amenities",

            type: "array",

            of: [{ type: "string" }],

            description: "List of amenities available in the room",

        }),

        defineField({

            name: "capacity",

            title: "Capacity",

            type: "number",

            description: "Maximum number of guests allowed",

        }),

        defineField({

            name: "rating",

            title: "Rating",

            type: "number",

            description: "Average customer rating",

            options: { min: 1, max: 5, step: 0.1 },

        }),

        defineField({

            name: "services",

            title: "Services",

            description: "Additional services available with the room (e.g., room service, laundry, wifi, parking, coffee etc)",

            type: "array",

            of: [{ type: "string" }],

            description: "Additional services available with the room",

        }),

    ]

});

**Bookings/Reservations**

import {defineType, defineField} from "sanity";

export const booking = defineType({

    name: "booking",

    title: "Booking",

    type: "document",

    icon: () => "📅",

    fields: [

        defineField({

            name: "client",

            title: "Client",

            type: "reference",

            to: [{type: "client"}],

        }),

        defineField({

            name: "room",

            title: "Room",

            type: "reference",

            to: [{type: "rooms"}],

        }),

        defineField({

            name: "check\_in",

            title: "Check-In Date",

            type: "date",

            options: {dateFormat: "YYYY-MM-DD"},

        }),

        defineField({

            name: "check\_out",

            title: "Check-Out Date",

            type: "date",

            options: {dateFormat: "YYYY-MM-DD"},

        }),

        defineField({

            name: "status",

            title: "Status",

            type: "string",

            options: {  list: [

                {title: 'Booked', value: 'booked'},

                {title: 'Checked In', value: 'checked\_in'},

                {title: 'Checked Out', value: 'checked\_out'},

                {title: 'Cancelled', value: 'cancelled'},

            ]},

        }),

        defineField({

            name: "created\_at",

            title: "Created At",

            type: "datetime",

            options: {dateFormat: "YYYY-MM-DD", timeFormat: "HH:mm"},

            initialValue: (new Date()).toISOString(),

        })

    ],

});

**Payment**

export const payments = defineType({

  name: "payments",

  title: "Payments",

  type: "document",

  icon: () => "💳",

  fields: [

    defineField({

      name: "reservation",

      title: "Reservation",

      type: "reference",

      to: [{ type: "booking" }], // references reservation (booking)

    }),

    defineField({

      name: "amount",

      title: "Amount",

      type: "number",

      description: "Total amount paid for the booking",

      validation: (Rule) => Rule.required().positive(),

    }),

    defineField({

      name: "status",

      title: "Payment Status",

      type: "string",

      options: {

        list: [

          { title: "Pending", value: "pending" },

          { title: "Paid", value: "paid" },

          { title: "Failed", value: "failed" },

          { title: "Refunded", value: "refunded" },

        ],

        layout: "dropdown",

      },

    }),

    defineField({

      name: "provider\_ref",

      title: "Provider Reference",

      type: "string",

      description: "Transaction reference from the payment provider",

    }),

    defineField({

      name: "paid\_at",

      title: "Paid At",

      type: "datetime",

      description: "Date and time when payment was completed",

      options: { dateFormat: "YYYY-MM-DD", timeFormat: "HH:mm" },

    }),

  ],

  preview: {

    select: {

      title: "provider\_ref",

      subtitle: "status",

    },

    prepare(selection) {

      const { title, subtitle } = selection;

      return {

        title: title || "Payment Record",

        subtitle: `Status: ${subtitle}`,

      };

    },

  },

});

### 5.3.4 Table Relationship Diagram

Use the ERD above; you can also convert the SQL foreign keys into a visual diagram in any ERD tool. The primary relationships are:

* users -> reservations (1-to-many)
* hotels -> rooms (1-to-many)
* rooms -> reservations (1-to-many)
* reservations -> payments (1-to-1 or 1-to-many)

## 5.4 Model the Views (UI snapshots / designs)

Provide sample view descriptions and low-fidelity mockups (textual):

1. Home/Search Page
   * Search bar with datepicker (check-in, check-out), guest count, and location filter.
   * Featured hotels and promotional cards.
2. Search Results
   * List of rooms with images, ratings, price per night, availability badge, and booking CTA.
3. Room Details
   * Gallery, amenities list, cancellation policy, price breakdown, and “Book Now” button.
4. Booking Flow
   * Stepper: Guest details → Payment → Confirmation.
5. Admin Dashboard
   * Occupancy chart, recent bookings list, revenue summary, and room management controls.

## 5.5 Interactivity & Module Coupling (Collaboration, Sequence, State Diagrams, IPO, HIPO)

### 5.5.1 Collaboration (Component Interaction Summary)

* Frontend UI interacts with Backend API endpoints.
* Backend orchestrates: Availability service, Reservation service, Payment service, Notification service, and Reporting service.
* Storage: Database + logs.
* External services: Payment Gateway.

### 5.5.2 Class Sequencing (high-level)

Provide components/classes:

* UserController, BookingController, PaymentController, NotificationService, RoomController, AdminController, ReportService, Database.

Sequence: User -> BookingController -> RoomService -> BookingService -> PaymentController -> PaymentGateway -> BookingService -> NotificationService.

### 5.5.3 State Diagram (Reservation lifecycle)

stateDiagram-v2

[\*] --> Provisional

Provisional --> Confirmed : Payment Success

Provisional --> Cancelled : User Cancel / Payment Fail

Confirmed --> CheckedIn : On arrival

CheckedIn --> CheckedOut : On checkout

Confirmed --> NoShow : Missed arrival

NoShow --> Cancelled

CheckedOut --> [\*]

### 5.5.4 IPO (Input-Process-Output) for Booking Module

* Input: Customer search params, booking form, payment details.
* Process: Validate input → check availability → create provisional reservation → process payment → confirm reservation → notify user.
* Output: Booking confirmation, updated room availability, payment receipt.

### 5.5.5 HIPO (Hierarchy plus Input-Process-Output) Chart (top-level)

* Top: Hotel Nexus
  + Booking Module
    - Search & Availability (IPO)
    - Reservation Management (IPO)
    - Payment Processing (IPO)
  + Admin Module
    - Room Management
    - Reporting
  + Notification Module
    - Email/SMS delivery
  + User Management
    - Registration, Authentication

# CHAPTER SIX

## SYSTEM IMPLEMENTATION AND TESTING

### 6.0 Implementation Overview

* The tools, technologies, and frameworks used.
* The development environment setup.
* The step-by-step implementation process, following your design in Chapter 5.

With the implementation of the Hotel Nexus system was carried out using modern web technologies to ensure scalability, security, and ease of use. The system was developed as a full-stack web application using Next.js for the frontend, Sanity CMS for backend content and database management, and Tailwind CSS for UI styling. The platform was hosted on Vercel, providing continuous deployment and integration with GitHub.

This project followed a modular development approach, where each major features (such as authentication, booking management, payment integration, and dashboard analytics) were implemented and tested independently before integration into the final system.

### **6.1 Step-by-Step Implementation Process**

1. **Database & Backend Setup**
   * Created schemas in Sanity (client, staff, rooms, bookings, payments).
   * Configured API routes to communicate with the Sanity client using GROQ queries.
2. **Frontend Development**
   * Built React components for Login, Dashboard, Room Management, and Bookings.
   * Implemented state management using React Hooks (useState, useEffect).
   * Designed a responsive UI using TailwindCSS.
3. **Authentication and User Management**
   * Added sign-up and login functionality for both Client and Staff users.
   * Integrated role-based dashboards.
4. **Booking Workflow**
   * Developed booking form to select room type, dates, and confirm reservations.
   * Connected frontend with Sanity to save bookings.
5. **Payment Integration**
   * Implemented mock or live payment using a payment gateway (e.g., Paystack/Stripe).
   * Stored payment data in payments schema.
6. **Testing & Debugging**
   * Performed unit and integration testing.
   * Fixed UI bugs and tested booking logic.
7. **Deployment**
   * Deployed to Vercel.
   * Connected Sanity project via API tokens and ensured proper environment variable configuration.

### **6.2 Detailed Semantics with Code Snippets**

// signUpHandler.js

Example Code Snippet

import { NextResponse } from "next/server";

import { client } from "@/sanity/lib/client";

import { encryptPassword } from "@/server/functions/encrypt";

export async function POST(request) {

    const formData = await request.json();

    const { first\_name, middle\_name, last\_name, gender, email, phone, password } = formData;

    // If client.fetch is not available, import sanityClient from the correct location

    // import { sanityClient } from "@/sanity/schema/client";

    // const existingUser = await sanityClient.fetch(...);

    // If client supports fetch:

    const existingUser = await client.fetch(

        `\*[\_type == "client" && (email == $email || phone == $phone)][0]`,

        { email, phone }

        );

    if (existingUser) {

        return NextResponse.json({

            type: "fail",

            success: false,

            message: "Email or Phone number already in use"

        }, { status: 400 })

    }

    if(!password || password.length < 6) {

        return NextResponse.json({

            type: "fail",

            success: false,

            message: "Password must be at least 6 characters long"

        }, { status: 400 });

    }

    const passwordHash = await encryptPassword(password);

    const newDoc = {

        \_type: "client",

        first\_name: first\_name,

        middle\_name: middle\_name,

        last\_name: last\_name,

        gender: gender,

        email: email,

        phone: phone,

        password: passwordHash,

    }

    try {

        const createdDoc = await client.create(newDoc);

        return NextResponse.json({

            type: "success",

            success: true,

            message: "Account created successfully",

            data: createdDoc

        }, { status: 201 });

    }catch (error) {

        console.error("Error creating document:", error);

        return NextResponse.json({

            type: "fail",

            success: false,

            message: "Error creating account",

            location: "signup\_route"

        }, { status: 500 });

    }

}

Example snippet (UI logic):

"use client"

import React from "react";

import { Building2, User, Mail, Phone, Eye, EyeOff } from "lucide-react";

import Image from "next/image";

import { set } from "sanity";

import { useRouter } from "next/navigation";

export default function SignupPage() {

  const [showPassword, setShowPassword] = React.useState(false);

  const [showConfirmPassword, setShowConfirmPassword] = React.useState(false);

  const [formData, setFormData] = React.useState({

    first\_name: "",

    middle\_name: "",

    last\_name: "",

    gender: "",

    email: "",

    phone: "",

    password: "",

    confirmPassword: "",

  });

  const [error, setError] = React.useState({});

  const [loading, setLoading] = React.useState(false);

  const [modal, setModal] = React.useState({

    isOpen: false,

    title: "",

    message: "",

  });

  const router = useRouter();

  // Controlled input handler

  const handleChange = (e) => {

    const { name, value, type, checked } = e.target;

    setFormData((prev) => ({

      ...prev,

      [name]: type === "checkbox" ? checked : value,

    }));

  };

  // Validation function

  const handleValidation = () => {

    const newError = {};

    const { password, confirmPassword, email, first\_name, last\_name, phone, gender } = formData;

    if (!first\_name) newError.first\_name = "First name is required.";

    if (!last\_name) newError.last\_name = "Last name is required.";

    if (!email) newError.email = "Email is required.";

    else {

      const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

      if (!emailRegex.test(email)) newError.email = "Please enter a valid email address.";

    }

    if (!phone) newError.phone = "Phone number is required.";

    else {

      // Ghana network operator indicators

      const operatorIndicators = [

        "020", "050", "024", "025", "055", "059", "026", "057", "056"

      ];

      // Remove non-digit characters

      const phoneDigits = phone.replace(/\D/g, "");

      const prefix = phoneDigits.substring(0, 3);

      if (

        phoneDigits.length !== 10 ||

        !operatorIndicators.includes(prefix)

      ) {

        newError.phone = "Phone number must be 10 digits and start with a valid network operator code.";

      }

    }

    if (!gender) newError.gender = "Gender is required.";

    if (!password) newError.password = "Password is required.";

    if (!confirmPassword) newError.confirmPassword = "Please confirm your password.";

    if (password && confirmPassword && password !== confirmPassword)

      newError.confirmPassword = "Passwords do not match.";

    setError(newError);

    return Object.keys(newError).length === 0;

  };

  const handleSignup = async (e) => {

    e.preventDefault();

    if (!handleValidation()) return;

    setLoading(true);

    try {

      const response = await fetch("/api/signup", {

        method: "POST",

        headers: {

          "Content-Type": "application/json",

        },

        body: JSON.stringify(formData),

      });

      let data;

      try {

        data = await response.json();

      } catch (jsonError) {

        throw new Error("Invalid server response");

      }

      if (data && data.success) {

        setModal({

          isOpen: true,

          title: "Success",

          message: "Account created successfully! Redirecting...",

          isSuccess: true,

        });

        setLoading(false);

        setTimeout(() => {

          setModal({ isOpen: false, title: "", message: "" });

          router.push("/"); // Change to your login route

        }, 1000);

      } else {

        setModal({

          isOpen: true,

          title: "Error",

          message: data?.message || "Signup failed.",

          isSuccess: false,

        });

        setLoading(false);

      }

    } catch (error) {

      console.error("Error signing up:", error);

      setModal({

        isOpen: true,

        title: "Error",

        message: "An error occurred while signing up. Please try again.",

        isSuccess: false,

      });

      setLoading(false);

    }

  };

  return (

    <div className="min-h-screen bg-gradient-to-br from-rose-500 to-rose-950 text-sm font-sans">

      {modal.isOpen && (

        <div className="fixed inset-0 bg-black/50 flex items-center justify-center z-50">

          <div className="bg-white rounded-lg shadow-lg p-6 max-w-sm w-full text-center">

            <h2 className="text-xl font-semibold mb-4">{modal.title}</h2>

            <p className="mb-6">{modal.message}</p>

            {modal.isSuccess ? (

              <div className="flex flex-col items-center justify-center">

                <span className="w-8 h-8 border-4 border-rose-500 border-t-transparent border-b-transparent rounded-full animate-spin mb-2"></span>

                <span className="text-rose-500 font-semibold">Redirecting...</span>

              </div>

            ) : (

              <div className="flex justify-end mt-4">

                <button

                  className="bg-red-500 px-4 py-2 text-white font-semibold rounded-sm w-auto"

                  onClick={() =>

                    setModal({ isOpen: false, title: "", message: "" })

                  }

                >

                  Ok

                </button>

              </div>

            )}

          </div>

        </div>

      )}

      <div className="container mx-auto px-4 py-8">

        <div className="grid lg:grid-cols-2 gap-8 items-center max-w-6xl mx-auto">

          {/\* Left Side - Hotel Image and Branding \*/}

          <div className="hidden lg:block">

            <div className="text-center mb-8">

              <div className="inline-flex items-center justify-center w-20 h-20 bg-rose-950 rounded-xl mb-6">

                <Building2 className="w-10 h-10 text-white" />

              </div>

              <h1 className="text-4xl font-bold text-white mb-2">Hotel Nexus</h1>

              <p className="text-xl text-gray-600">Hotel Management System</p>

            </div>

            <div className="relative rounded-2xl overflow-hidden shadow-2xl">

              <Image

                src="/images/hotel-lobby.png"

                alt="Luxury hotel lobby"

                width={600}

                height={400}

                className="w-full h-[400px] object-cover"

              />

              <div className="absolute inset-0 bg-gradient-to-t from-black/50 to-transparent" />

              <div className="absolute bottom-6 left-6 text-white">

                <h3 className="text-2xl font-bold mb-2">Join Hotel Nexus Today</h3>

                <p className="text-lg opacity-90">

                  Streamline your hotel operations with our comprehensive management system

                </p>

              </div>

            </div>

          </div>

          {/\* Right Side - Signup Form \*/}

          <div className="w-full max-w-md mx-auto lg:mx-0">

            {/\* Mobile Branding \*/}

            <div className="text-center mb-8 lg:hidden">

              <div className="inline-flex items-center justify-center w-16 h-16 bg-blue-600 rounded-xl mb-4">

                <Building2 className="w-8 h-8 text-white" />

              </div>

              <h1 className="text-3xl font-bold text-gray-900">HotelPro</h1>

              <p className="text-gray-600 mt-2">Hotel Management System</p>

            </div>

            {/\* Signup Card \*/}

            <div className="bg-white rounded-xl shadow-xl p-6">

              <div className="space-y-1 pb-6 text-center">

                <h2 className="text-2xl font-semibold">Create Account</h2>

                <p className="text-gray-600">Join HotelPro to manage your hotel efficiently</p>

              </div>

              <form className="space-y-4" onSubmit={handleSignup}>

                {/\* Name Fields \*/}

                <div className="grid grid-cols-2 gap-4">

                  <div className="space-y-2">

                    <label htmlFor="firstName" className="text-sm font-medium text-gray-700">

                      First Name \*

                    </label>

                    <div className="relative">

                      <User className="absolute left-3 top-1/2 transform -translate-y-1/2 text-gray-400 w-4 h-4" />

                      <input

                        id="firstName"

                        name="first\_name"

                        type="text"

                        placeholder="John"

                        className="pl-10 h-12 w-full border border-gray-200 rounded-lg focus:border-blue-500 focus:ring focus:ring-blue-200"

                        required

                        value={formData.first\_name}

                        onChange={handleChange}

                      />

                    </div>

                    {error.first\_name && <p className="text-rose-600 text-sm mt-1">{error.first\_name}</p>}

                  </div>

                  <div className="space-y-2">

                    <label htmlFor="lastName" className="text-sm font-medium text-gray-700">

                      Last Name \*

                    </label>

                    <div className="relative">

                      <User className="absolute left-3 top-1/2 transform -translate-y-1/2 text-gray-400 w-4 h-4" />

                      <input

                        id="lastName"

                        name="last\_name"

                        type="text"

                        placeholder="Doe"

                        className="pl-10 h-12 w-full border border-gray-200 rounded-lg focus:border-blue-500 focus:ring focus:ring-blue-200"

                        required

                        value={formData.last\_name}

                        onChange={handleChange}

                      />

                    </div>

                    {error.last\_name && <p className="text-rose-600 text-sm mt-1">{error.last\_name}</p>}

                  </div>

                </div>

                {/\* Middle Name \*/}

                <div className="space-y-2">

                  <label htmlFor="middleName" className="text-sm font-medium text-gray-700">

                    Middle Name <span className="text-gray-400">(Optional)</span>

                  </label>

                  <div className="relative">

                    <User className="absolute left-3 top-1/2 transform -translate-y-1/2 text-gray-400 w-4 h-4" />

                    <input

                      id="middleName"

                      name="middle\_name"

                      type="text"

                      placeholder="Middle name"

                      className="pl-10 h-12 w-full border border-gray-200 rounded-lg focus:border-blue-500 focus:ring focus:ring-blue-200"

                      value={formData.middle\_name}

                      onChange={handleChange}

                    />

                  </div>

                </div>

                {/\* Gender \*/}

                <div className="space-y-3">

                  <label className="text-sm font-medium text-gray-700">Gender \*</label>

                  <div className="flex space-x-6">

                    <label className="flex items-center space-x-2 cursor-pointer">

                      <input

                        type="radio"

                        name="gender"

                        value="M"

                        className="form-radio text-blue-600"

                        required

                        checked={formData.gender === "M"}

                        onChange={handleChange}

                      />

                      <span className="text-sm">Male</span>

                    </label>

                    <label className="flex items-center space-x-2 cursor-pointer">

                      <input

                        type="radio"

                        name="gender"

                        value="F"

                        className="form-radio text-blue-600"

                        checked={formData.gender === "F"}

                        onChange={handleChange}

                      />

                      <span className="text-sm">Female</span>

                    </label>

                  </div>

                  {error.gender && <p className="text-rose-600 text-sm mt-1">{error.gender}</p>}

                </div>

                {/\* Email \*/}

                <div className="space-y-2">

                  <label htmlFor="email" className="text-sm font-medium text-gray-700">

                    Email Address \*

                  </label>

                  <div className="relative">

                    <Mail className="absolute left-3 top-1/2 transform -translate-y-1/2 text-gray-400 w-4 h-4" />

                    <input

                      id="email"

                      name="email"

                      type="email"

                      placeholder="john.doe@example.com"

                      className="pl-10 h-12 w-full border border-gray-200 rounded-lg focus:border-blue-500 focus:ring focus:ring-blue-200"

                      required

                      value={formData.email}

                      onChange={handleChange}

                    />

                  </div>

                  {error.email && <p className="text-rose-600 text-sm mt-1">{error.email}</p>}

                </div>

                {/\* Phone \*/}

                <div className="space-y-2">

                  <label htmlFor="phone" className="text-sm font-medium text-gray-700">

                    Phone Number \*

                  </label>

                  <div className="relative">

                    <Phone className="absolute left-3 top-1/2 transform -translate-y-1/2 text-gray-400 w-4 h-4" />

                    <input

                      id="phone"

                      name="phone"

                      type="tel"

                      placeholder="+1 (555) 123-4567"

                      className="pl-10 h-12 w-full border border-gray-200 rounded-lg focus:border-blue-500 focus:ring focus:ring-blue-200"

                      required

                      value={formData.phone}

                      onChange={handleChange}

                    />

                  </div>

                  {error.phone && <p className="text-rose-600 text-sm mt-1">{error.phone}</p>}

                </div>

                {/\* Password \*/}

                <div className="space-y-2">

                  <label htmlFor="password" className="text-sm font-medium text-gray-700">

                    Password \*

                  </label>

                  <div className="relative">

                    <input

                      id="password"

                      name="password"

                      type={showPassword ? "text" : "password"}

                      placeholder="Create a strong password"

                      className="pl-3 pr-10 h-12 w-full border border-gray-200 rounded-lg focus:border-rose-500 focus:ring focus:ring-rose-200"

                      required

                      value={formData.password}

                      onChange={handleChange}

                    />

                    <button

                      type="button"

                      className="absolute inset-y-0 right-0 pr-3 flex items-center text-gray-400 hover:text-gray-600"

                      onClick={() => setShowPassword(!showPassword)}

                    >

                      {showPassword ? (

                        <EyeOff className="h-5 w-5" />

                      ) : (

                        <Eye className="h-5 w-5" />

                      )}

                    </button>

                  </div>

                  {error.password && <p className="text-rose-600 text-sm mt-1">{error.password}</p>}

                </div>

                {/\* Confirm Password \*/}

                <div className="space-y-2">

                  <label htmlFor="confirmPassword" className="text-sm font-medium text-gray-700">

                    Confirm Password \*

                  </label>

                  <div className="relative">

                    <input

                      id="confirmPassword"

                      name="confirmPassword"

                      type={showConfirmPassword ? "text" : "password"}

                      placeholder="Confirm your password"

                      className="pl-3 pr-10 h-12 w-full border border-gray-200 rounded-lg focus:border-rose-500 focus:ring focus:ring-rose-200"

                      required

                      value={formData.confirmPassword}

                      onChange={handleChange}

                    />

                    <button

                      type="button"

                      className="absolute inset-y-0 right-0 pr-3 flex items-center text-gray-400 hover:text-gray-600"

                      onClick={() => setShowConfirmPassword(!showConfirmPassword)}

                    >

                      {showConfirmPassword ? (

                        <EyeOff className="h-5 w-5" />

                      ) : (

                        <Eye className="h-5 w-5" />

                      )}

                    </button>

                  </div>

                  {error.confirmPassword && <p className="text-rose-600 text-sm mt-1">{error.confirmPassword}</p>}

                </div>

                {/\* Terms and Conditions \*/}

                <div className="flex items-start space-x-2">

                  <input

                    id="terms"

                    name="terms"

                    type="checkbox"

                    required

                    checked={!!formData.terms}

                    onChange={handleChange}

                  />

                  <label htmlFor="terms" className="text-sm text-gray-600 leading-relaxed">

                    I agree to the{" "}

                    <button type="button" className="text-rose-600 hover:text-rose-800 font-medium">

                      Terms of Service

                    </button>{" "}

                    and{" "}

                    <button type="button" className="text-rose-600 hover:text-rose-800 font-medium">

                      Privacy Policy

                    </button>

                  </label>

                </div>

                {/\* Create Account Button \*/}

                <button

                  type="submit"

                  className="w-full h-12 bg-rose-600 hover:bg-rose-800 text-white font-medium rounded-lg transition-colors flex items-center justify-center"

                  disabled={loading}

                >

                  {loading ? (

                    <>

                      <span className="w-5 h-5 border-2 border-white border-t-transparent border-b-transparent rounded-full animate-spin mr-2"></span>

                      Creating Account

                    </>

                  ) : (

                    "Create Account"

                  )}

                </button>

              </form>

              {/\* Sign In Link \*/}

              <div className="text-center pt-4 border-t border-gray-200 mt-6">

                <p className="text-sm text-gray-600">

                  Already have an account?{" "}

                  <button className="text-blue-600 hover:text-blue-800 font-medium">Sign in here</button>

                </p>

              </div>

            </div>

          </div>

        </div>

      </div>

    </div>

  );

}

### **6.3 Purpose of Implementation**

The purpose of this implementation section is to provide sufficient technical information for future developers or researchers to replicate the Hotel Nexus system. Every module, data model, and process has been documented to ensure clarity and transparency. This reproducibility ensures the project can be extended or adapted for other hospitality management systems.

### **6.4 Testing**

#### **6.4.1 Testing Strategy**

Testing followed a bottom-up approach. Each function was first tested independently (unit testing), followed by integration testing to confirm the proper flow between frontend and backend. Finally, user acceptance testing was conducted with selected hotel staff and clients to evaluate usability, functionality, and system reliability.

#### **6.4.2 Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Function | Input | Expected Output | Actual Output | Result |
| TC01 | Login | Correct email/password | Redirect to dashboard | Redirect successful | Pass |
| TC02 | Booking | Select room, check-in/out | Booking confirmation | Booking confirmed | Pass |
| TC03 | Payment | Valid payment details | Payment success & record saved | Payment recorded | Pass |
| TC04 | Cancel Booking | Select booking → cancel | Booking removed | Booking marked “cancelled” | Pass |

#### **6.4.3 Test Results and Observations**

All major functions of the system were tested and yielded successful results. The login, booking, and payment modules worked seamlessly. Minor interface bugs were detected during early testing and resolved. The system now performs efficiently with minimal latency and supports simultaneous user operations.

**­CHAPTER SEVEN**

**SYSTEM DOCUMENTATION**

## REFERENCES

1. Factors Influencing Online Hotel Booking Intention: A Conceptual Framework from Stimulus-Organism-Response Perspective Asilah Emir, Hazwani Halim, Asyikin Hedre, Dahlan Abdullah, Azila Azmi, Saiful Bahri Mohd Kamal.
2. Hotel Trends in 2025: What Ghanaian Hoteliers Should Expect. Amapolley Kabenla Annor. Feb 19.
3. (Statistics Brian, 2016) - Statistic Brian. (2016). Travel bookings statistics.
4. GNU-https://gna.org.gh/2023/07/ghanas-tourism-and-hospitality-industry-needs-support-to-grow-dr-joseph-mensah/ by Daniel Adu Darko – 2023
5. R. Baggio, “Information and Communication Technologies in the Hospitality Industry: The Italian Case,” e-Review of Tourism Research (eRTR), Vol. 2, No. 5, pp. 108-114, 2004
6. Adopting Electronic Business in Ghana: Story of the Hospitality Industry Nana Yaw Asabere, Vida Doku. Simonov KusiSarpong, David Oppong
7. N.Y. Asabere and V. Doku “Measuring Customer Relationship Management in the Hospitality Industry of Some Selected Hotels in Accra, Ghana: The Role of ICTs”, International Journal of Application or Innovation in Engineering and Management (IJAIEM), Vol. 2, Issue 3, pp. 19-28, 2013.
8. N. Olifer and V. Olifer, “Computer Networks, Principles, Technologies and Protocol for Network Design,” John Wiley and Sons Ltd, England, 2006.
9. Online computerized Hotel Management System Ogirima, Sanni Abubakar Omuya, Awode, Tolulope Reuben, Adeosun, Olajide Olusegun Department of Computer Science & Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria

# Appendix A

This questionnaire is part of a research project titled "Design and Implementation of an Online Hotel Booking System – Hotel Nexus", conducted by Koranteng Aaron Ayeh, a student of the Computer Science Department at Koforidua Technical University. The project is being submitted in partial fulfilment of the requirements for the award of the BSc in [Your Program Name].

The purpose of this questionnaire is to gather data to understand the challenges, requirements, and preferences of both hotel staff and hotel customers regarding hotel booking processes in Ghana. The findings will guide the design and development of Hotel Nexus, an online booking platform aimed at enhancing efficiency and customer satisfaction in the hospitality industry.

Your responses will remain confidential and will be used solely for academic purposes. Kindly answer all questions honestly and tick the appro  
Thank you for your participation.

## SECTION 1: Respondent Demographics

## SECTION 2: For Hotel Staff/Managers

## SECTION 3: For Hotel Guests/Customers

## SECTION 4: General Feedback