**“Hotel Reservation Management System for Pilot Testing of the College of Business Management Students in Mindoro State University”**

A non-thesis Project

Presented to the faculty of the

College of Computer Studies

**Mindoro State University**

Calapan City Campus

Masipit, Calapan City, Oriental Mindoro

In Partial Fulfillment

Of the requirements for the Course of

**APPLICATION DEVELOPMENT AND EMERGING TECHNOLOGY**

Maderazo, Justine Rafael B.

Llamoso, Aloha Chloe L.

Tabay, Renz Bryan A.

BSIT III-F3

October 2023

**TABLE OF CONTENTS**

Title page ………………………………………………………………………………………………………………… 1

Table of Contents………………………………………………………………………………………………… 2

**CHAPTER**

1. INTRODUCTION

Project Context

Objectives

Scope and Limitations

Definitions of Terms

1. REQUIREMENTS SPECIFICATION

Hardware and Software Requirements

Functional Requirements

Non-Functional Requirements

Operational Requirement

Performance Requirement

Security Requirement

1. DESIGN AND DEVELOPMENT METHODOLOGIES

System Design

Database Design

Architectural Diagram/Block Diagram

DFD Level 0

UML Use-Case Diagram

Sample Mock Up

Methodology

SDLC Phases

Testing

ISO 25010 Evaluation

Respondents Profile

Likert Scale

Evaluation Instrument

1. DEVELOPMENT, TESTING AND EVALUATION RESULT

System Output

Testing Results

ISO 25010 Evaluation Result

1. CONCLUSIONS AND RECOMMENDATIONS

Conclusion

Recommendation

REFERENCES

APPENDICES

**CHAPTER I**

**INTRODUCTION**

In this chapter, the researchers will discuss the overview of the project, including its project context, objectives, scope, and limitations.

**Project Context**

The 'Hotel Reservation and Management System for Pilot Testing of the College of Business Management Students in Mindoro State University' emerges as a trailblazing initiative propelled by the fusion of educational innovation and real-world application in an era marked by the dynamic landscape of hospitality services and the expanding educational horizons of college students. Offering a full software solution designed especially for instructional purposes inside Mindoro State University's College of Business Management, this project promises to modernize hotel operations and reservation management. The dedication to this project highlights the value of creating a thorough Software Requirements Specification (SRS) document to set the foundation for an innovative system with cutting-edge capabilities. In addition to helping students learn via hands-on experiences, these features will advance the development of effective hotel management software.

In an effort to meet the changing demands of the hospitality sector, the "Hotel Reservation and Management System" project was created. Customers will be able to check room availability, make reservations with ease, and manage bookings effectively thanks to the system's integration of an intuitive web-based platform. With the goal of improving hotel management's operational effectiveness, the system will include a number of modules for reservation administration, room inventory, guest services, invoicing, and reporting. The platform will also have a component solely for instructional reasons, giving students a safe setting in which to comprehend and practice real-world hotel management issues.

This project's relevance stems from its dual function, which includes serving as an essential learning tool for students in addition to meeting the operational needs of the hotel sector. Students at Mindoro State University's College of Business Management can practice hotel management, booking, and guest services by engaging in realistic simulations. Innovative features of the system mean that end users and students will have a better experience. These features include a user-friendly interface that is accessible to both hotel staff and visitors, secure payment channels, individual user profiles, and extensive reporting capabilities.

The 'Hotel Reservation and Management System' project is aimed mainly at students pursuing education in hotel and business management, as well as hotels and hospitality firms looking to optimize their operations. This project meets the needs of two groups of people: industry professionals looking to improve operational efficiency and students hoping to obtain real-world experience in hotel management. It does this by combining practical learning components with advanced management tools.

The 'Hotel Reservation and Management System' project aims to redefine operational excellence in the hotel industry and give students a practical learning environment through its inventive features and educational goals. The technical specifications, design, and minute details of this cutting-edge system will all be covered in depth in this SRS document, which will also provide the groundwork for an integrated strategy that blends operational effectiveness with top-notch instruction in the field of hotel management.

Poór, J., Tóth, É., Besenyei, D. (2019). Hotel Reservation in a Hungarian Therme and Spa Hotel. E-world is characterized by the preference for electronic options, which provides more efficient, faster and more convenient solutions than the traditional way. The emergence of the Internet restructured and made dynamic the tourism sales chain, creating the concept of e-tourism. Online tourism services have made a significant contribution to the growing popularity of individually organized trips by providing convenient and quick information, comparison, and transaction opportunities. At present, the former traditional and new, modern booking options live side by side. Based on hotel booking statistics, in our study we examine the characteristics of direct and indirect, as well as offline and modern online booking. The use of modern information and communication technologies (ICT) and the Internet has led to revolutionary changes in all aspects of life, starting a new marketing reality and changing the relations between market players. The relevance of this in our study is consumers or customers on the online marketers take advantage of the ease of searching and booking. The ideas we got for this project are websites that ensure the variety of product offerings and quick price comparisons for the customers within a small fraction of time which is called “time-saving”. It will benefit for those two parties. The one who made the system and the consumers.

Sun-n-Sand Hotel Mumbai, India (2019) Sun-n-Sand, Mumbai is India’s first 5-star beachfront hotel. The hotel has been a popular backdrop in several Indian movies and many Bollywood stars is their frequent patrons. However, the biggest challenge they faced was not knowing how to increase their Average Daily Rate’s or the (ADR) and when to increase or decrease their rates in a systematic manner. The hotel started using Aiosell in mid-2019. Aiosell’s automated dynamic pricing ensured that the rates were not only benchmarked against their key competition hotels, but were also changing

constantly with changing market demands and patterns. AI based pricing algorithm, that took hotel’s lead time, seasonality, day of the week, time of the booking and several such dynamic factors ensured that rates were updated 30- 40 times/day automatically, thereby ensuring yield optimization. Moreover, the hotel management & operations team was free from their day-to-day tasks of managing the channel management system and could focus all their energies on strategic expansion and guest experience. Just like in our system, if we can manage to make the system for its price by seasonal peak and maintain the prices of every room by giving discounts to loyal customers. We should know how to increase and decrease the rates because we will need a good understanding of baseline rates and how they change throughout the year before we can start improving the rates. Looking at rates can give you a good idea of demand at your hotel, indicating which days of the week and times of the year you're likely to see the highest demand for rooms. This can help you determine your pricing strategy, know when to promote special packages, and help with revenue management.

**Objectives of the Project**

The researchers aim for the ‘Hotel reservation and management system for pilot testing project’ are to identify and address technical and user experience issues, ensuring the system aligns with hotel requirements, improves operational efficiency, and enhances guest experience. Specifically, this project aims:

1. To enable customers to make online reservations seamlessly.
2. To ensure real-time availability of rooms and accurate booking information.
3. To provide a user-friendly interface for reservation management.
4. To track room availability, including different room types and their respective prices.
5. To implement an efficient inventory management system to avoid overbooking.
6. To maintain a database of customer profiles and preferences.
7. To implement a feedback system to gather guest reviews and improve services.
8. To generate accurate bills for guests, including room charges, additional services, and taxes.
9. To generate reports on occupancy rates, revenue, customer demographics, etc.
10. To ensure data security and compliance with relevant data protection regulations.
11. To implement secure login systems and access controls for staff members.
12. To integrate with other hotel systems like accounting, CRM, and point of sale (POS) systems.
13. To automate routine tasks, such as check-in/check-out procedures, to improve staff productivity.
14. To provide role-based access controls to restrict staff access to sensitive information.
15. To enable guests to make reservations, check-in, and access services through their smartphone, tablets, and PC.
16. To use customer feedback to continuously improve service quality.

**Scope and Limitations**

The study aims to develop a hotel reservation and management system for Pilot Testing of the College of Business and Management Students in Mindoro State University.

The hotel reservation and management system for pilot testing aim to evaluate its functionality, reliability, and user-friendliness in a real-world setting before a full-scale implementation. It offers a simple user interface, room inventory, check-in and check-out process, message confirmation, accounts for staff and management, and reservation management. Through the web application or mobile application, the project intends to enhance the overall experience of guests and efficiency of hotel operations.

There are restrictions to be considered in hotel reservation and management system project. This only be for College of Business and Management Department at Mindoro State University. Primarily confined to a pilot testing phase within an educational environment, potentially limiting its scalability to larger, real-world hospitality settings.

**Definition of Terms**

**Pilot Testing:** The initial phase of testing a system, product, or software within a limited environment to evaluate its performance, usability, and functionality before broader implementation.

**Educational Innovation:** The application of new methods, approaches, or technologies in an educational setting to enhance learning experiences or improve teaching methods.

**Cutting-Edge Capabilities**: Advanced and innovative features or functionalities that are at the forefront of technological development and provide a competitive edge.

**Reservation Administration:** The process of managing and overseeing the booking and reservation procedures for hotel rooms or services.

**Inventory Management:** The control and oversight of available resources or goods, such as hotel room availability and allocation.

**Guest Services:** Services provided to guests during their stay, including assistance, amenities, and any other customer-oriented services.

**Invoicing:** The process of generating bills or invoices for services provided, including room charges, additional services, and taxes.

**Occupancy Rates:** The percentage of available rooms or accommodations that are occupied during a specific period.

**Yield Optimization:** Maximizing revenues through dynamic pricing based on market demands, optimizing rates in real-time.

**Data Security:** The protection of digital information against unauthorized access or data breaches through various security measures and protocols.

**Point of Sale (POS) Systems:** Software or hardware used for transactions in hospitality settings, managing sales and payments.

**User Feedback:** Information or comments provided by users or guests to improve services and enhance customer experience.

**Scalability:** The capability of a system or software to handle increased workload or growth without compromising performance or function.

**CHAPTER II**

**REQUIREMENTS AND SPECIFICATIONS**

Functional Requirements define the functions and processes that the software must perform to meet the needs of its users. It provides a detailed description of the behavior of the system in response to inputs, outputs, and expected results. This level of detail is necessary to ensure that the system has a strong foundation for its performance.

**Hardware Requirements**

Hardware Requirements refer to the representation of the hardware used by the system. Table 1 below presents the hardware requirements to be used by the project.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Hardware** | **Functions** | **Specifications** | | **Unit** |
| **Minimum** | **Recommended** |
| Processor | It's used to tackle the challenging tasks of building a solid, secure, and dependable system. This ensures we have the computing muscle to keep our system running smoothly and reliably. | Intel(R)Core (TM)i37020U CPU | Intel Core i3 or higher version | 1 |
| RAM | It acts as a temporary memory storage for the system. It enables the computer to quickly access the website's data and process it efficiently, providing a faster and smoother browsing experience. | 6GB | 8GB | 2 |
| SSD | For faster performance of the system |  |  | 3 |
| High-resolution Display | Facilitate spacious workspace for coding and design work. |  | Full HD or higher. | 4 |
| Internet Connection | A reliable internet connection to enable seamless access to cloud storage, version control, and various web development tools. |  |  | 5 |
| Android Smartphone | It is used if the customer wants to access the system conveniently and for seamless booking a room they want. |  |  | 6 |

**Table 1: Software Requirements**

**Software Requirements**

Software Specifications refer to the representation of the software used by the system. Table 2 below presents the software specifications to be used by the project.

|  |  |
| --- | --- |
| Software Used | Description |
| Operating System | The researchers chose the Windows 10 operating system because it will suit and can support the system. |
| Web Development Server | The laragon provides a graphical interface for SQL which is the phpMyAdmin that helps to maintain data in a relational database. |
| Visual Studio Code | Visual Studio Code is a code editor that supports development operations like debugging, task running, and version control. |
| CodeIgniter 4 | CodeIgniter 4 is the framework being employed for the creation of the backend of the system. It is a set of tools for constructing websites using PHP. |
| Frontend Development | Vue.js is a programming language that we use to create a visually appealing and user-friendly graphical interface for front-end coding. |
| Version Control | GitHub is a web-based platform built around the Git version control system. It enhances Git by providing a graphical interface, web-based hosting for repositories, and features like issue tracking, pull requests, and code review. GitHub makes collaboration more accessible, enabling teams to work together efficiently, whether they are in the same office or distributed globally. |

**Table 2: Software Requirements**

**Functional Requirements**

This part enumerates the operations and activities that the system must perform. Table 3 represents the descriptions of data requirements, process requirements, and output requirements.

|  |  |
| --- | --- |
| **ID NO** | **Requirement Description** |
| **FR1** | **Data Requirements** |
| **Admin Account** | |
| 1.1 | The administrator should be able to provide the needed information to register an account. |
| 1.2 | The admin can manage user accounts. |
| 1.3 | The admin can add/edit/remove/update the room details such as room type, rates, and availability. |
| 1.4 | The admin can be able to monitor and update room status. |
| **User Account** | |
| 1.5 | User should be able to create an account with their personal information to access the webpage. |
| 1.6 | Users should be able to log in and log out of their accounts securely. |
| 1.7 | Users have option to reset their password in case they forget their login credentials. |
| 1.8 | Users must be able to book a room for specific dates, providing necessary details like check-in and check-out dates, and number of guests. |
| 1.9 | Users can view/and confirm their reservation before making the final booking |
| 2.0 | The user can give suggestions through feedback |
| **FR2** | **Process Requirements** |
| **Admin Account** | |
| 2.1 | The system must accept the registration of the new user. |
| 2.2 | The system must accept all the reservation from customers and process them immediately. |
| 2.3 | The system must send notifications email to the client with all transactions. |
| 2.4 | The system must send a confirmation email whenever the booking/reservation is placed. |
|  | The system must send payment confirmation to the customers. |
| **User Account** | |
| 2.5 | The system must receive an email notification of the previous transaction. |
| 2.6 | The system can view and track their reservation in real-time. |
| **Staff Account** | |
| 2.7 | The system must receive an email notification of the previous transaction and track the check-in and out of the guests. |
| **FR3** | **Output Requirements** |
| **Admin Account** | |
| 2.8 | The system must have an intuitive Home page for easy management of reservation. |
| 2.9 | The system generates payment receipt. |
| **User Account** | |
| 3.0 | The system should view all the transaction history. |

**Table 3: Functional Requirements**

**Non-Functional Requirements**

These are requirements that pertain to behavior properties that a system must have. It defines how a system is supposed to be or its system properties. It contains the following:

**Operational Requirement**

Table 4 represents the requirement description that will specify the operating environment(s) in which the system must perform and how these might change over time.

|  |  |
| --- | --- |
| ID No. | Requirement Description |
| 1.1 | The system is compatible with any operating system |
| 1.2 | The system has an intuitive user interface to facilitate easy navigation for both customers and hotel staff. |

**Table 4: Operational Requirement**

**Performance Requirement**

Table 5 represents the requirement description that will emphasize the response time, capacity, and reliability of the system.

|  |  |
| --- | --- |
| ID No. | Requirement Description |
| 1.3 | Each web page must load within 2 seconds |
| 1.4 | The user can track their reservation 98% of the time without failure. |
| 1.5 | Any user reserves a room in the system. |
| 1.6 | The system must have the capacity to handle a minimum of 10 reservation request per minute. |
| 1.7 | The system must be able to handle a minimum of 100 concurrent users without performance degradation. |

**Table 5: Performance Requirement**

**Security Requirement**

Table 6 represents the requirement description that will address issues with security, such as who has access to the system's data and must have the ability to protect data from disruption or data loss.

|  |  |
| --- | --- |
| ID No. | Requirement Description |
| 1.8 | The user’s information including the payment information must be encrypted |
| 1.9 | Only the admin can manage the user accounts |
| 2.0 | The users must be logged in before they reserved a room. |

**Table 6: Security Requirement**

**CHAPTER III**

**DESIGN AND DEVELOPMENT METHODOLOGIES**

**System Design**

The development team has meticulously devised a robust plan for the implementation of the Hotel Reservation system, facilitating customers in seamlessly booking rooms and obtaining information with greater ease. Furthermore, the system ensures swift and efficient processing of transactions. Hosted on a website, this system mandates an internet connection for both administrators and customers to access its functionalities. Registered customers can conveniently log in through the website to avail themselves of the system's features.

**Database design**

Making a good database for the hotel reservation system is super important for building the app. It helps people know how things are organized and how to use the app right. The design includes different tables with specific details, like names and types of information, to make sure everything is clear and organized. This design also shows how different parts are connected.

For the database, the developers chose MySQL. It helps connect different tables, making it easier to handle how things relate to each other. The main goal is to design the database so that it fits perfectly with what a hotel reservation system needs.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size** | **Default** | **Description** |
| user\_id | int |  | Not Null | User ID |
| username | varchar | 50 | Not Null | User Name |
| password | varchar | 50 | Not Null | User Password |
| token | varchar | 100 | Not Null | User token |
| status | varchar | 10 | Not Null | User Status |
| role | varchar | 10 | Not Null | User Role |

**Table 7. Fields for User**

Table 7 above contains the field name, data type, size, default, and description of the field in the User table. Here, the user\_id is the Primary Key (PK).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size** | **Default** | **Description** |
| id | int |  | Not Null | Room id |
| roomName | varchar | 50 | Not Null | Room name |
| roomType | varchar | 50 | Not Null | Room type |
| ac | varchar | 10 | Not Null | Room Aircon |
| food | varchar | 50 | Not Null | Room food |
| cancelCharge | varchar | 50 | Not Null | Room cancels charge |
| numGuest | int |  | Not Null | Room number of guest |
| rent | int |  | Not Null | Room rent |
| desciption | varchar | 100 | Not Null | Room description |
| status | varchar | 50 | Not Null | Room status |

**Table 8. Fields for Room**

Table 8 above contains the field name, data type, size, default, and description of the field in the Room table. Here, the id is the Primary Key (PK).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size** | **Default** | **Description** |
| id | int |  | Not Null | Pending booking id |
| user\_id | int |  | Not Null | User id (FK) |
| roomName | varchar | 50 | Not Null | User room name |
| Time | timestamp |  | Current\_timestamp(6) | Time created |
| arrivalDate | date |  | Not Null | Arrival date |
| depatureDate | date |  | Not Null | Departure date |
| guest | int |  | Not Null | Guest |
| rent | varchar | 50 | Not Null | Room rent |
| amenities | json |  | Null | Amenities |

**Table 9. Fields for Pending Booking**

Table 9 above contains the field name, data type, size, default, and description of the field in the Pending booking table. Here, the id is the Primary Key (PK) and the user\_id is a foreign key(FK).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size** | **Default** | **Description** |
| Payment\_id | int |  | Not Null | Payment id |
| user\_id | int |  | Not Null | User id (FK) |
| booking\_id | int |  | Not Null | Booking id(FK) |
| Transaction\_id | int |  | Not Null | Transaction id |
| arrivalDate | date |  | Not Null | Arrival date |
| depatureDate | date |  | Not Null | Departure date |
| total | varchar | 50 | Not Null | Total amount |
| datePaid | datetime |  | Current\_timestamp | Date of payment |

**Table 10. Fields for Payment**

Table 10 above contains the field name, data type, size, default, and description of the field in the Payment table. Here, the payment\_id is the Primary Key (PK) while the user\_id and booking\_id are the foreign key(FK).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size** | **Default** | **Description** |
| employee\_id | int |  | Not Null | Employee id |
| name | varchar |  | Not Null | Employee name |
| username | varchar | 50 | Not Null | Employee username |
| email | varchar | 50 | Not null | Employee email |
| password | varchar | 150 | Not Null | Employee password |
| date\_joined | date | 50 | Not Null | Date joined |
| phone\_num | varchar | 13 | Not Null | Employee phone/mobile number |
| role | varchar | 50 | Not Null | Employee role |

**Table 11. Fields for Employee**

Table 11 above contains the field name, data type, size, default, and description of the field in the Employee table. Here, the employee\_id is the Primary Key (PK).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size** | **Default** | **Description** |
| id | int |  | Not Null | booking id |
| user\_id | int |  | Not Null | User id(FK) |
| name | varchar | 50 | Not Null | User name |
| time | timestamp |  | Current\_timestamp(6) | Time created |
| arrivalDate | date |  | Not Null | Arrival date |
| depatureDate | date |  | Not Null | Departure date |
| guest | int |  | Not Null | Guest |
| rent | varchar | 50 | Not Null | Room rent |
| amenities | json |  | Null | Amenities |

**Table 12. Fields for Booking**

Table 12 above contains the field name, data type, size, default, and description of the field in the Booking table. Here, the id is the Primary Key (PK) while the user\_id is a foreign key (FK).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size** | **Default** | **Description** |
| amenities\_id | int |  | Not Null | Amenities id |
| name | varchar | 50 | Not Null | Amenities name |
| price | int |  | Not Null | Amenities price |

**Table 13. Fields for Amenities**

Table 13 above contains the field name, data type, size, default, and description of the field in the Amenities table. Here, the amenities\_id is the Primary Key (PK).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Size** | **Default** | **Description** |
| id | int |  | Not Null | Cancel Id |
| User\_id | int |  | Not Null | User id (FK) |
| Booking\_id | int |  | Not Null | Booking id (FK) |
| reason | varchar | 100 | Not Null | Reason why they cancel the book. |

**Table** **14 Fields of Cancellations**

Table 14 above contains the field name, data type, size, default, and description of the field in the Cancellation table. Here, the id is the Primary Key (PK) and the user\_id and booking\_id are the foreign key (FK).

**Architectural Diagram/ Block Diagram**

In this section, system architecture was designed to define the flow and behavior of the system’s functionalities to execute its high-quality performance. This covers the formal illustration and description of the project structure.

A screenshot of a computer

Description automatically generated

**Figure 1. System Architecture of Hotel Reservation Management System**

In Figure 1, the system architecture of the hotel reservation management system development is depicted, illustrating the flow and functionality of the system. The researchers emphasize the necessity of an internet connection for accessing the hotel reservation features by guests, staff, and administrators to facilitate the smooth execution of intended transactions.

**Data Flow Diagram Level 0**

This section shows the Data Flow Diagram Level 0 which is commonly known as an exploded view of the context diagram that shows the detailed process of how the project works.

A computer screen shot of a diagram

Description automatically generated

**Figure 2. Data Flow Diagram Level 0**

In figure 2, diagram 0 illustrates the step-by-step process of the system from the administrative perspective. It outlines various functions available to the admin, such as user notification, logging into the system, and making bookings. The admin has the flexibility to choose and utilize any of these functions based on their specific requirements within the system.

**UML Use-case Diagram**

In this section, the use case diagram serves as a concise overview, presenting the primary functions and visually representing the connections among the system, admin, and users. It provides insight into how these entities interact, facilitating a comprehensive understanding of the project's functionality for researchers as they identify and structure the various aspects of the system.

A screenshot of a computer

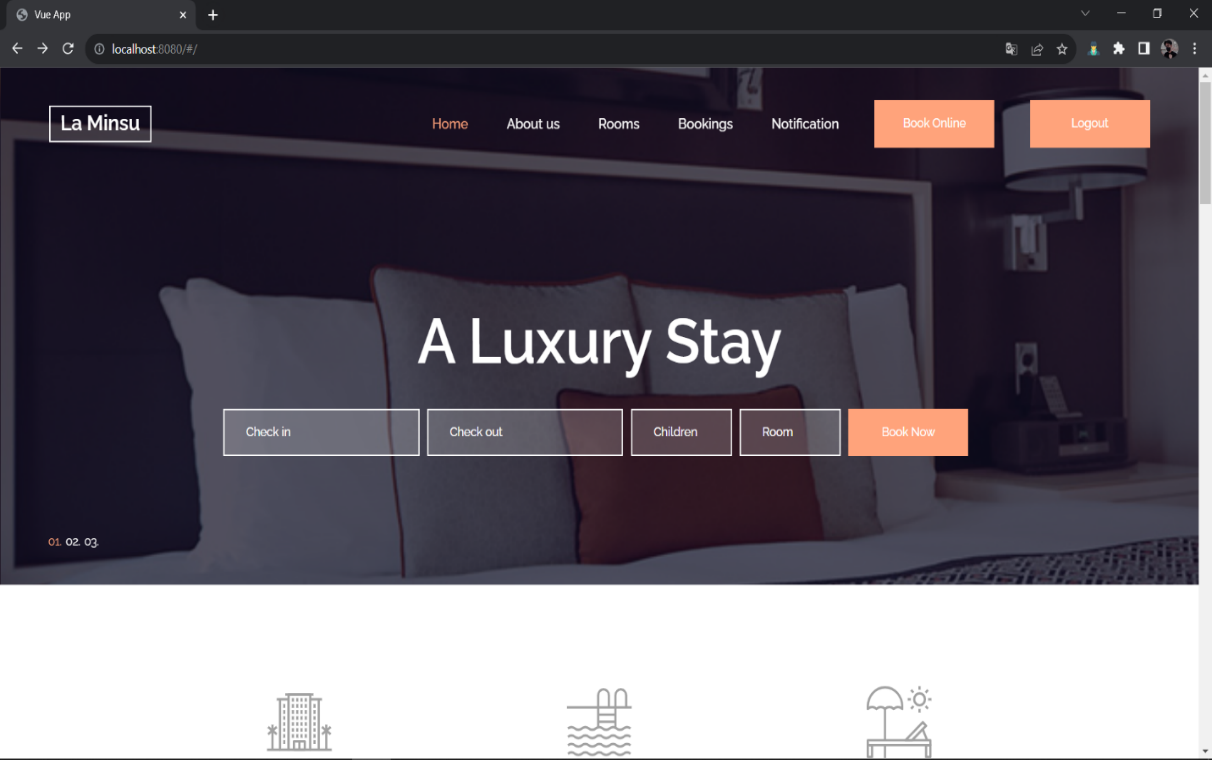
Description automatically generated

**Figure 3. UML Use-case Diagram**

In Figure 3, you can see the different jobs that Admin, Staff, and users have in the system. The picture shows what each of them does and how they work together to make the system function properly. It's like a map that helps you understand who does what in the system.

**Sample Mock-up**

A sample mock-up is like a picture of a website before it's made. It shows how the website should look and what it should do. People use it to make sure the design is good, find any issues, and make sure the website works the way users want it to. Here are the pictures of how regular users and administrators will see and use the website.



**Figure 4. User Interface**

A screenshot of a computer

Description automatically generated

**Figure 5. Admin Interface**

**Development Method**

The "Hotel Reservation Management System for College of Business Management in MinSU" will be made using a process called the System Development Life Cycle (SDLC). This process helps ensure that we create a good and effective system. For this project, we're using a method called Iterative and Incremental Model, which means we can make things better as we go along in the development. The whole process has four steps: first, we collect what we need for the system; then, we design and build it; next, we test to make sure it works well, and finally, we put it into use. Following these steps will help us make a system that works well for everyone.

**Requirements Gathering**

**Testing**

**Implementation**

**Design & Development**

**Testing**

**Implementation**

**Design & Development**

Build 1

Build 2

**Testing**

**Implementation**

**Design & Development**

Build 3

**Planning**

**Maintenance**

**Figure 6. SDLC Iterative and Incremental Model**

The research went through different steps:

1. **Planning**: First, the researchers figured out what the problem was and how to solve it. They talked to people, observed things, and planned for the project.
2. **Requirements Gathering**: Next, they gathered more information and details needed for the project. They figured out what the system had to do and what it shouldn't do.
3. **Design**: Then, they started designing the computer and software parts of the system. They carefully planned and tried different things to make sure the system would look and work the way people wanted it to.
4. **Development**: After that, they started writing the code that makes the system work. They used tools like Visual Studio Text Editor, PhpMyAdmin, and CodeIgniter 4 to create and store the information the system needed.
5. **Testing**: They tested the system to make sure it worked correctly before letting people use it. This was like practice to catch and fix any problems.
6. **Implementation**: Once everything worked well, they put the project into action. They looked at how people used it and made improvements to make it even better.
7. **Maintenance**: After the system was up and running, they made sure it kept working well. They also fixed any issues that popped up over time to keep the system running smoothly.

**Gantt Chart**

In this section, the Gantt Chart is presented to show the plans and schedules of the project timeline. All the development stages up to the completion of the project were documented in this chart. This helps the researchers to know the deadlines needed to accomplish and show breakthroughs in various tasks.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Task Name** | Task Date | | | | | | | | | | | | |
| Oct | | | | Nov | | | | | Dec | | | | |
| Week 1 | Week 2 | Week 3 | Week 4 | Week 1 | Week 2 | Week 3 | Week 4 | Week 1 | | Week 2 | Week 3 | Week 4 | |
| **1.Planning** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.1 Conduct an interview |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.2 Define project objectives |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.3 Define project plan |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.4 Approval of project plan |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **2.Requirements Gathering** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.1 Data Collection |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.2 Functional |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.3 Non-Functional |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **3.Design** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 3.1 Frontend software design |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **4.Development** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 4.1 Back-end coding |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **5.Testing** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 5.1 Functionality testing |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 5.2 User interface testing |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **6.Implementation** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **7.Maintenance** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 7.1 Project monitoring |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 7.2 Resolve system errors |  |  |  |  |  |  |  |  |  | |  |  |  | |

**Table 11. Gantt Chart**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Task Name** | Task Date | | | | | | | | | | | | |
| Oct | | | | Nov | | | | | Dec | | | | |
| Week 1 | Week 2 | Week 3 | Week 4 | Week 1 | Week 2 | Week 3 | Week 4 | Week 1 | | Week 2 | Week 3 | Week 4 | |
| **1.Planning** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.1 Conduct an interview |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.2 Define project objectives |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.3 Define project plan |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.4 Approval of project plan |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **2.Requirements Gathering** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.1 Data Collection |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.2 Functional |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.3 Non-Functional |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **3.Design** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 3.1 Frontend software design |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **4.Development** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 4.1 Back-end coding |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **5.Testing** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 5.1 Functionality testing |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 5.2 User interface testing |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **6.Implementation** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **7.Maintenance** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 7.1 Project monitoring |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 7.2 Resolve system errors |  |  |  |  |  |  |  |  |  | |  |  |  | |

**Table 12. Gantt Chart (Maderazo, Justine Rafael B.)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Task Name** | Task Date | | | | | | | | | | | | |
| Oct | | | | Nov | | | | | Dec | | | | |
| Week 1 | Week 2 | Week 3 | Week 4 | Week 1 | Week 2 | Week 3 | Week 4 | Week 1 | | Week 2 | Week 3 | Week 4 | |
| **1.Planning** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.1 Conduct an interview |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.2 Define project objectives |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.3 Define project plan |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.4 Approval of project plan |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **2.Requirements Gathering** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.1 Data Collection |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.2 Functional |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.3 Non-Functional |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **3.Design** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 3.1 Frontend software design |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **4.Development** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 4.1 Back-end coding |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **5.Testing** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 5.1 Functionality testing |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 5.2 User interface testing |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **6.Implementation** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **7.Maintenance** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 7.1 Project monitoring |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 7.2 Resolve system errors |  |  |  |  |  |  |  |  |  | |  |  |  | |

**Table 13. Gantt Chart (Llamoso, Aloha Chloe L.)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Task Name** | Task Date | | | | | | | | | | | | |
| Oct | | | | Nov | | | | | Dec | | | | |
| Week 1 | Week 2 | Week 3 | Week 4 | Week 1 | Week 2 | Week 3 | Week 4 | Week 1 | | Week 2 | Week 3 | Week 4 | |
| **1.Planning** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.1 Conduct an interview |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.2 Define project objectives |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.3 Define project plan |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 1.4 Approval of project plan |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **2.Requirements Gathering** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.1 Data Collection |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.2 Functional |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 2.3 Non-Functional |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **3.Design** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 3.1 Frontend software design |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **4.Development** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 4.1 Back-end coding |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **5.Testing** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 5.1 Functionality testing |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 5.2 User interface testing |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **6.Implementation** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| **7.Maintenance** |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 7.1 Project monitoring |  |  |  |  |  |  |  |  |  | |  |  |  | |
| 7.2 Resolve system errors |  |  |  |  |  |  |  |  |  | |  |  |  | |

**Table 14. Gantt Chart (Tabay, Renz Bryan A.)**

**Legend:** - Completed/ Done

Table 11 shows the whole process of developing the hotel reservation management system. It displayed the various tasks and marks as completed oats certain date. Researchers will be kept informed of the progress of the development which will help them not to miss out on steps and differentiate tasks from the amount of time took to complete them.

Table 12, 13, and 14 show how each of the researchers do the part of their system. This illustrates the task they’ll do in working on the project until its completion.

**Testing and Evaluation**

This part talks about checking and reviewing the software as it's being made. The developers must do tests on the system to see what it can and can't do. This helps them find any problems before the software is officially used. The tests need to cover everything from how it's designed to how well it works and other things it needs to support. The results of these tests help the developers see how well the system is doing and if it's meeting the goals of the project. The developers considered the following:

1. **Unit Testing**- It checks each small part of the program to make sure it works the way it's supposed to. Usually, the people who created the program (developers) do this testing because they know how the inside of the program is built. The goal is to make sure that every little piece of the program does its job correctly, like individual functions or parts, to ensure the whole program works as it should.
2. **Component Testing**- This step is about checking each part of the system, like building blocks, to see if they work by themselves. We do this before putting them together. It helps find any problems in each part and makes sure they don't cause issues when combined. We want to catch any mistakes early on and make sure everything works well together in the end.
3. **System Testing**- checking the entire computer program to make sure everything works together correctly. It tests if each part of the program does what it's supposed to and if they share information accurately. This testing is crucial to make sure the program meets all its needs, works as expected, and fits well into its intended use. We usually do this after checking each part separately and making sure they work well together.

**CHAPTER IV**

**DEVELOPMENT, TESTING AND EVALUATION RESULT**

**Presentation of the System Output**

**User Interface(UI)**

**A screenshot of a computer

Description automatically generated**

**Figure 1.2 User Log In Form**

**Figure 1.2 shows the login forms of the customer wherein she/he will be directed to the website. If you haven’t yet registered, you can create account and you’ll be directed to the registration form.**

**A screenshot of a computer

Description automatically generated**

**Figure 1.3 User Registration Form**

**Figure 1.3 shows the registration form wherein the user can register their account to login in the website.**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a hotel room

Description automatically generatedFigure 1.4 Homepage**

**Figure 1.4 shows the homepage of the website where the user can see the information about the hotel and what services it may offer. If you click the button book now you will be directed to the booking page. The user also sees the available rooms that they can book.**

**A screenshot of a hotel room

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Figure 1.5 Booking Page**

**Figure 1.5 shows the booking page of the website where the user can create their reservation. They also see the rooms that are available in the hotel to make them easily reserved for the room they wanted.**

**A screenshot of a website

Description automatically generated**

**Figure 1.6 Bookings**

**Figure 1.6 shows the details of the reservation that the user made and they can cancel their reservation if they want to.**

**Admin Dashboard (UI)**

**A screenshot of a computer

Description automatically generated**

**Figure 1.7 Admin Dashboard Page**

**Figure 1.7 shows the admin dashboard page where the admin can see the booking that the user made. In this page the admin can see all the transactions. Admin can also make a change in the rooms.**

**A screenshot of a computer

Description automatically generated**

**Figure 1.8 Booking Dashboard**

**Figure 1.8 shows all the booking information. Admin can see who are the guests and what room did they avail and also the time of their arrival and departure.**

**A screenshot of a computer

Description automatically generated**

**Figure 1.9 Pending Booking**

**Figure 1.9 shows the request of the guests that were booked in the hotel.**

**A screenshot of a computer

Description automatically generated**

**Figure 2.0 Cancellation**

**Figure 2.0 shows all the cancelled booking and the information of the guest and also the reason why they cancel their booking.**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generatedFigure 2.1 Amenities**

**Figure 2.1 shows all the amenities available in the hotel. The admin can make changes about the prices and amenities that are available. The admin can add amenities that the hotel have.**

**A screenshot of a computer

Description automatically generated**

**Figure 2.2 Payment Module**

**Figure 2.2 shows the payment receipt which contains the information of the customers and the amount that they need to pay after their stay in the hotel. When the user click the generate invoice button it will print the payment receipt.**

**A screenshot of a computer

Description automatically generated**

**Figure 2.3 Payment receipt**

**Figure 2.3 shows the pdf form of the payment receipt of the customers.**

**Chapter V**

**CONCLUSION AND RECOMMENDATION**

**Conclusion**

1. This project makes it super easy for customers to book a hotel room. You can reserve a room online without any trouble, making it smooth and simple for guests.
2. The system makes sure that rooms are always ready to book right away and provides accurate information. This helps the hotel run better and keeps guests happy.
3. The project is making the booking system easy for everyone to use. It's all about making things positive for both customers and the hotel staff.
4. The system keeps track of which rooms are available, what types they are, and how much they cost. This helps the hotel manage rooms and set the right prices.
5. The project solves the problem of having too many bookings by being smart about managing the rooms. This makes the hotel run smoother with fewer mistakes.
6. The system keeps information on customers and what they like. This helps the hotel give guests a personal touch and encourages them to come back.
7. Guests can share their thoughts on the hotel through a feedback system. This helps the hotel get better by learning from what guests have to say.
8. The project makes sure that guest bills are accurate, covering room charges, additional services, and taxes, which helps with financial transparency.
9. The system will create helpful reports on things like how full the hotel is, how much money it's making, and details about the guests. This helps with decision-making and planning.
10. Keeping data safe and following rules ensures that guests' information is secure. This makes guests trust the hotel more.
11. Addressing data security and following regulations makes sure that sensitive guest information is protected and builds trust.
12. The project works well with other systems the hotel uses. This makes it easy to manage everything about the hotel in one place.
13. The system does some jobs automatically, like checking guests in and out. This helps the staff focus on more important tasks to make guests happy.
14. Only certain people can see private information with role-based access. This keeps everything safe and private.
15. Guests can use their phones, tablets, or computers to book, check-in, and get services. This makes it easy and convenient for guests and keeps up with what they like.
16. The project is committed to using customer feedback to get better all the time. This shows a focus on making things better and growing within the hotel management system.

**Recommendation**

The effective implementation of the "Hotel Reservation and Management System for Pilot Testing Project" relies on key recommendations. These include prioritizing a user-friendly interface and establishing real-time testing with a feedback loop for swift issue resolution. It is crucial to provide comprehensive staff training on accurate data entry, system efficiency, and security protocols, while continuously updating measures to comply with regulations and protect guest information. Strategies for active guest engagement, ongoing monitoring of room availability, and seamless integration with other hotel systems are also vital. Optimizing for mobile devices to meet guest expectations and regularly conducting reporting and analysis for informed decision-making are important aspects. Clearly defining staff roles, providing continuous training, and regularly reviewing automated processes contribute to a smooth and efficient system. Implementation of these recommendations ensures the success of the system, offering a seamless experience for guests and optimizing hotel operations.

**REFERENCES:**

**Poór, J., Tóth, É., Besenyei, D. (2019). Reservation System for Citystate Hotel.** [**www.coursehero.com/file/150516394/Reservation-System-for-Citystate-Hotel-Revision-1docx/**](http://www.coursehero.com/file/150516394/Reservation-System-for-Citystate-Hotel-Revision-1docx/)