**A PROPOSED OFFERING OF A HOTEL RESERVATION MANAGEMENT SYSTEM FOR EUROTEL NORTH EDSA**

A Design Document Presented to the

Faculty of Datamex College of Saint Adeline, Inc.

In Partial Fulfillment of the Requirements for the

Degree of Bachelor of Science in Information Technology

Presented By:

Adarayan, Jessie

Anacta, Francis Rico

Bernadas, Marjorie

Pedro, Pamela Anne

## **INTRODUCTION**

Hotels are expected to provide not only comfort but also convenience in their services. One of the main concerns many guests face is the long and sometimes confusing reservation process. At Eurotel North Edsa, many of the booking and front desk tasks are still done manually, which can cause slow transactions, unclear room availability, and mistakes in record-keeping. These problems affect both the efficiency of the admin and the overall satisfaction of the guests.

The Hotel Reservation Management System is created to solve these issues by offering a more organized and dependable way of handling reservations. It allows guests to book their rooms either online or as walk-ins, while admin can easily monitor room status in real time. With this system, check-in and check-out become faster and more orderly, reducing delays and unnecessary waiting time.

Aside from convenience, the system also promotes accuracy and safety. Guest details and booking records are stored in a digital database, making sure that information is properly arranged and secured. This not only improves the service but also lessens the chances of errors that usually happen in manual processes. The system can also automatically prepare reports such as daily occupancy, and guest lists, which help the management make better decisions.

In addition, the system helps improve communication among hotel admin by keeping all the needed information in one place. For example, housekeeping can quickly see which rooms are occupied, available, or under maintenance without waiting for updates from the front desk. This makes hotel operations more coordinated and smoother.

In the future, using this system can also help the hotel grow. By giving guests a quicker and easier booking experience, the hotel can gain repeat customers and be more competitive in the market. At the same time, admin can be more productive since they will spend less time on manual paperwork and focus more on giving quality service.

In conclusion, this project is meant to modernize the daily operations of Eurotel North Edsa. By adopting a digital reservation system, the hotel can improve its image, give a better experience for guests, and build a stronger foundation for future improvements. This system not only addresses current problems but also prepares the hotel for bigger opportunities in the digital age.

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## **SYSTEM ARCHITECTURE**

The Hotel Reservation Management System is built on a three-tier architecture consisting of the Client Side, Application Layer, and Database Layer. This structure ensures organized processing, smooth communication, and secure handling of hotel operations.

**High-Level Components and Interactions**

* Guest Portal – Provides features for guests such as online and walk-in room booking, checking room availability, managing personal booking details, and viewing reservation status.
* Front Desk Dashboard – Allows hotel admin to manage reservations, monitor room occupancy, process check-ins and check-outs, update room availability, and handle payment transactions.
* Admin Panel – Enables administrators to oversee hotel operations, manage admin accounts, configure room rates, generate financial and activity reports, and maintain system security.
* Database – Stores and organizes essential hotel data such as guest details, booking records, room availability, admin information, payment history, and system logs to support efficient and reliable hotel operations.

**Deployment Architecture**

The Hotel Reservation Management System follows a client–server deployment architecture. End users access the system through a standard web browser on their devices (laptops, desktops, or mobile phones). The client-side application consists of HTML, CSS, and JavaScript files, which provide the graphical interface for guests and administrators.

These clients connect to a web server running Apache with PHP, which hosts both the frontend and backend logic. The PHP scripts act as the application layer and expose REST API endpoints that process user requests such as retrieving room availability or submitting bookings.

At the backend, the system uses a MySQL database server to store persistent data including user accounts, rooms, and booking records. The database can be deployed on the same machine as the web server for small-scale setups, or on a dedicated database server for scalability and reliability.

**Communication Protocols and Interfaces**

* HTTP/HTTPS – Communication between the frontend (browser) and backend API (PHP) is done via HTTP(S) requests.
* REST API – The backend exposes REST-style endpoints:
  + GET /get\_rooms.php – fetch available rooms.
  + POST /submit\_booking.php – submit a new booking.
  + (Additional endpoints can include login, manage rooms, manage bookings, etc.)
* **JSON** – Data exchange format between frontend and backend. The API returns JSON responses (e.g., list of rooms, booking confirmation).
* **SQL (MySQL)** – The backend communicates with the database using SQL queries for data storage and retrieval.

## 

## **DATABASE DESIGN**

Database design is the process of structuring and organizing data into a logical and efficient system that supports storage, retrieval, and management. It involves identifying entities, relationships, and constraints to ensure accuracy, consistency, and scalability. A well-designed database improves data integrity, minimizes redundancy, and enhances overall system performance.

**Database Description**

* Guests: Keeps the basic details of guests such as name, contact number, email, address, date of birth, and payment status. It may also store login accounts for those who book online.
* Admin: Saves the information of hotel employees including name, email, password, role (like receptionist or admin), and the date they were added to the system.
* Rooms: Lists all hotel rooms with details like room number, type (single, double, or suite), status (vacant, occupied, or under maintenance), capacity, and price per night.
* Reservations: Connects guests to their booked rooms. It includes booking date, check-in and check-out dates, reservation status (confirmed, pending, or canceled), and payment information.
* Payments: Stores the record of payments made by guests with details such as amount, payment method (over the counter), payment date, and link to the reservation.

**Entity-relationship diagram (ERD)**

The Entity–Relationship Diagram (ERD) of the Hotel Reservation Management System illustrates the logical structure of the database and shows how different entities are related. The ERD provides a clear blueprint for developers to understand how data flows within the system, ensuring accuracy, consistency, and efficiency in handling hotel operation

1. Customer

customer\_id (PK)

name

contact\_no

email

Identification

2. Admin

admin\_id (PK)

name

username

password

role

3. Room

room\_id (PK)

room\_number

type

rate

Availability\_status

4. Reservation

reservation\_id (PK)

customer\_id (FK)

room\_id (FK)

admin\_id (FK)

check\_in

check\_out

status

5. Payment

payment\_id (PK)

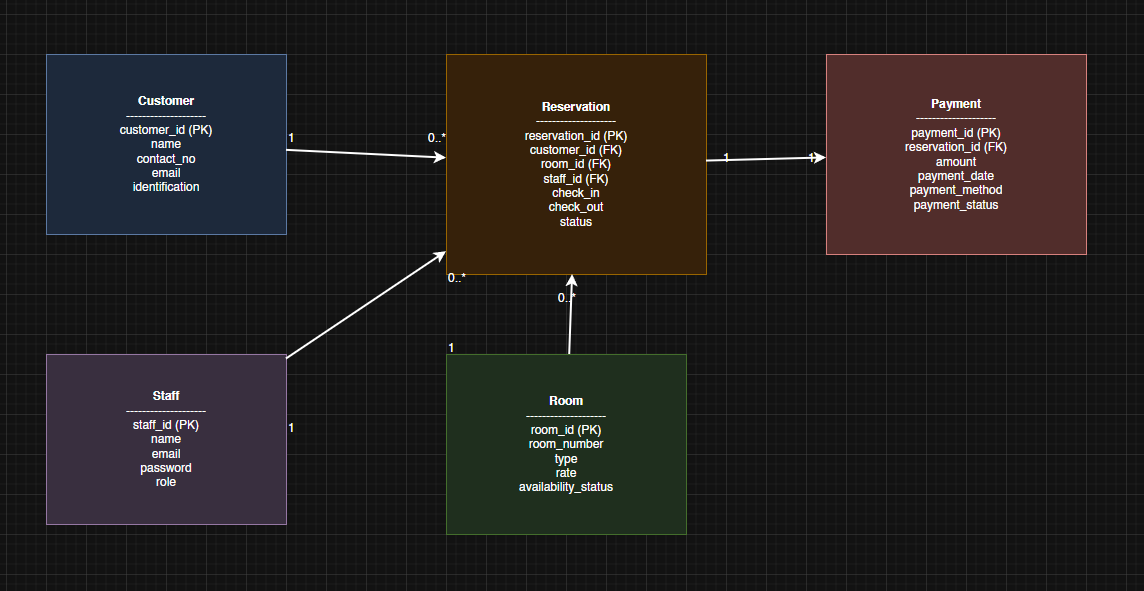
reservation\_id (FK)

amount

payment\_date

payment\_method

payment\_status



*Image 1: Entity-Relationship Diagram*

The Entity–Relationship Diagram (ERD) of the Hotel Reservation Management System illustrates the logical structure of the database and how different entities are related to each other. Key entities include:

* Customer – stores guest details such as name, contact number, email, and identification.
* Reservation – records booking details including check-in/check-out dates, status, and references to customers, rooms, and admin.
* Room – contains information about room numbers, types, rates, and availability.
* Payment – holds transaction details linked to reservations such as amount, method, and payment status.
* Admin – manages user accounts and roles, and is associated with reservations they handle.

The relationships are defined as follows:

* One Customer can make many Reservations.
* One Room can be associated with many Reservations.
* Each Reservation has exactly one Payment.
* One Admin member can manage many Reservations.

Through this ERD, developers gain a clear blueprint of how data flows within the system, ensuring accuracy, consistency, and efficiency in handling hotel operations.

**Data Normalization Techniques**

The database design for the Hotel Reservation Management System applies the principles of Third Normal Form (3NF) to maintain consistency, minimize redundancy, and make data easier to manage.

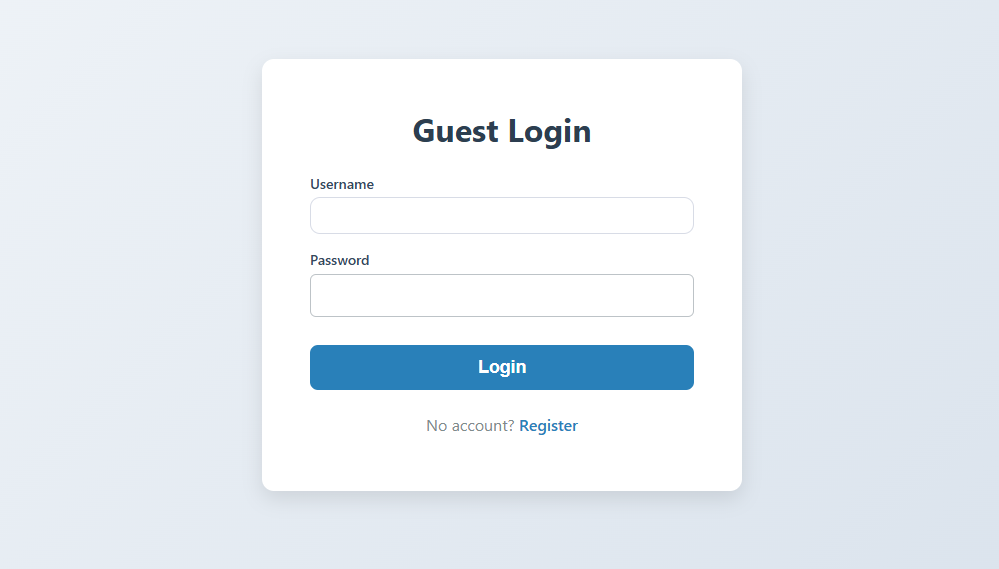
1. First Normal Form (1NF): All fields are kept atomic, meaning each contains only one value. For example, guest information such as full\_name, email, and contact\_number are stored separately instead of being grouped in a single field.
2. Second Normal Form (2NF): Every non-key attribute depends entirely on the primary key of the table or collection. For instance, each record in guests, admin, rooms, and reservations are identified by a unique ID (guestId, adminId, roomId, reservationId), preventing partial dependencies.
3. Third Normal Form (3NF): Transitive dependencies are removed by using references instead of repeating data. For example, reservation records only store references (guest\_ref, room\_ref, admin\_ref) rather than duplicating all guest or admin details. Similarly, payment records are linked to a reservation through reservation\_ref, ensuring updates are consistent and do not require changes across multiple tables.

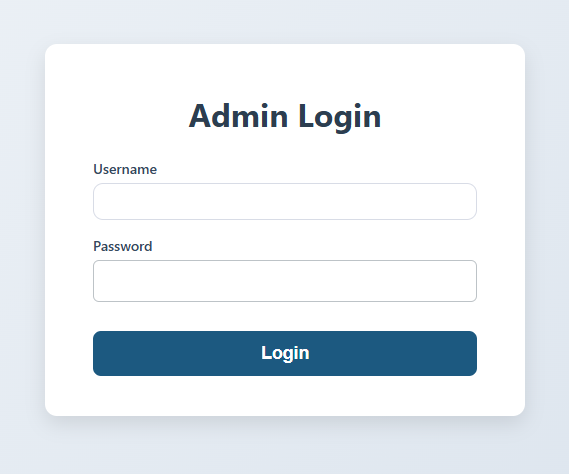
## 

## 

## **USER INTERFACE DESIGN**

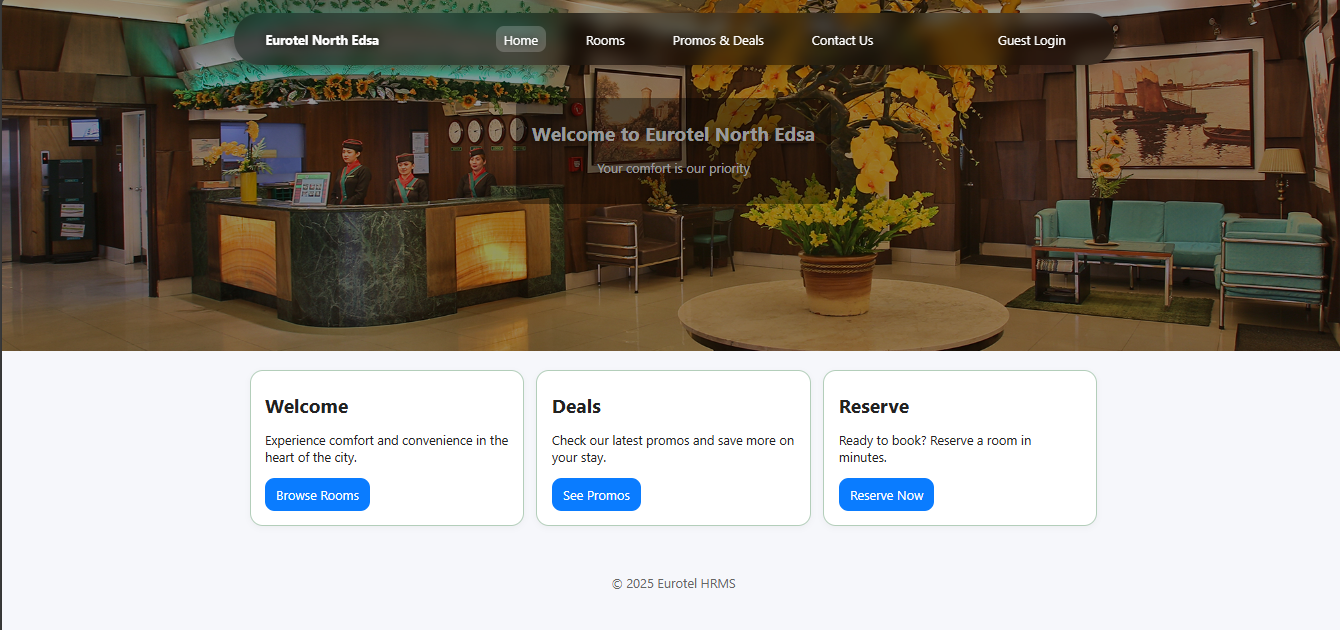
This section describes the system’s web-based interface, which is designed to be simple, responsive, and user-friendly. It provides clear navigation through forms, buttons, and menus, ensuring accessibility across devices. The interface supports both guests and admin by streamlining reservations, records management, and overall interaction with the system.

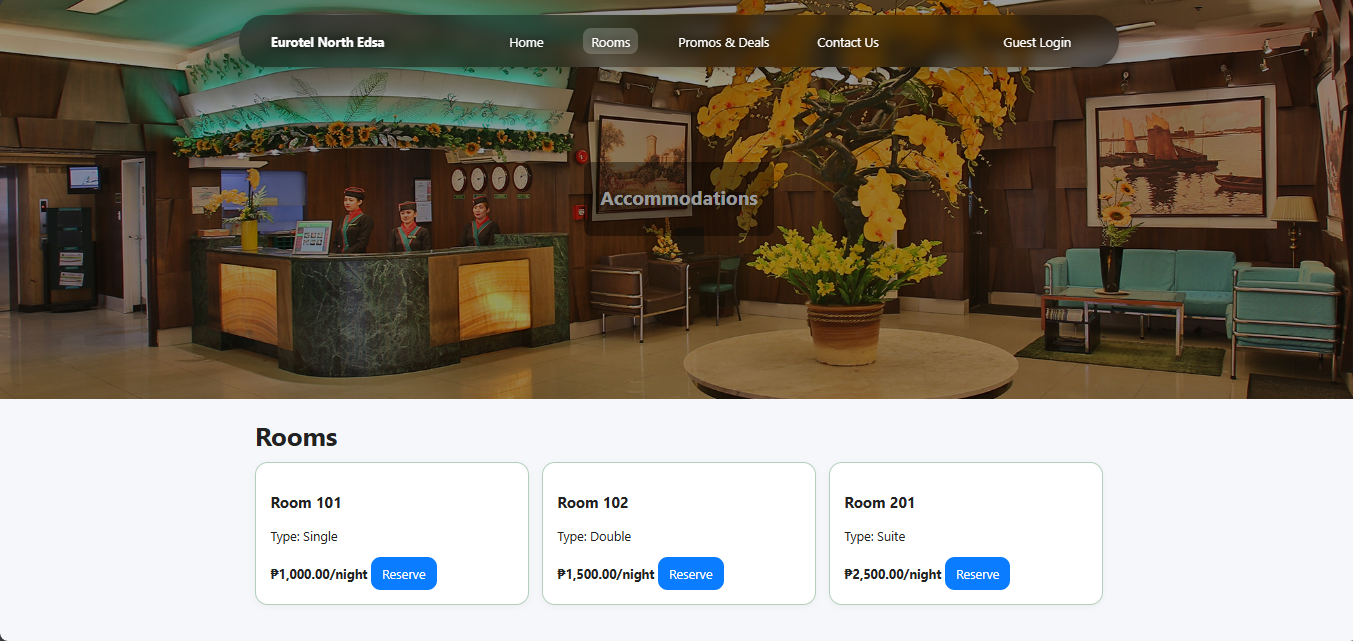


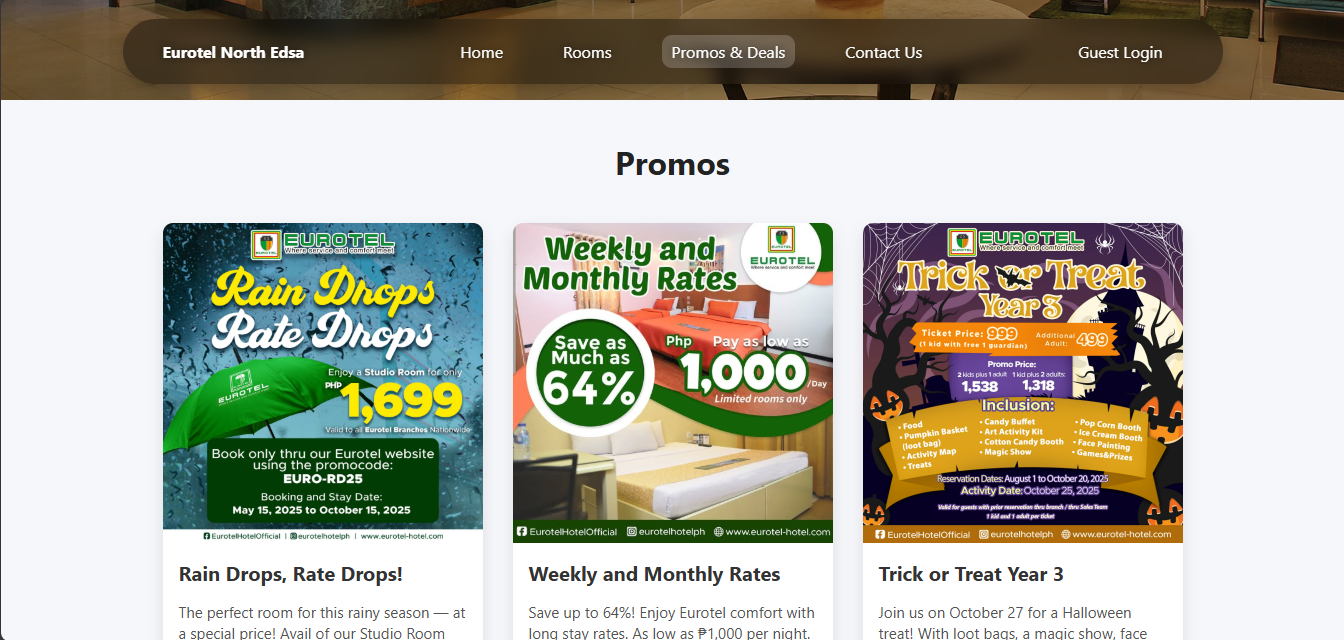


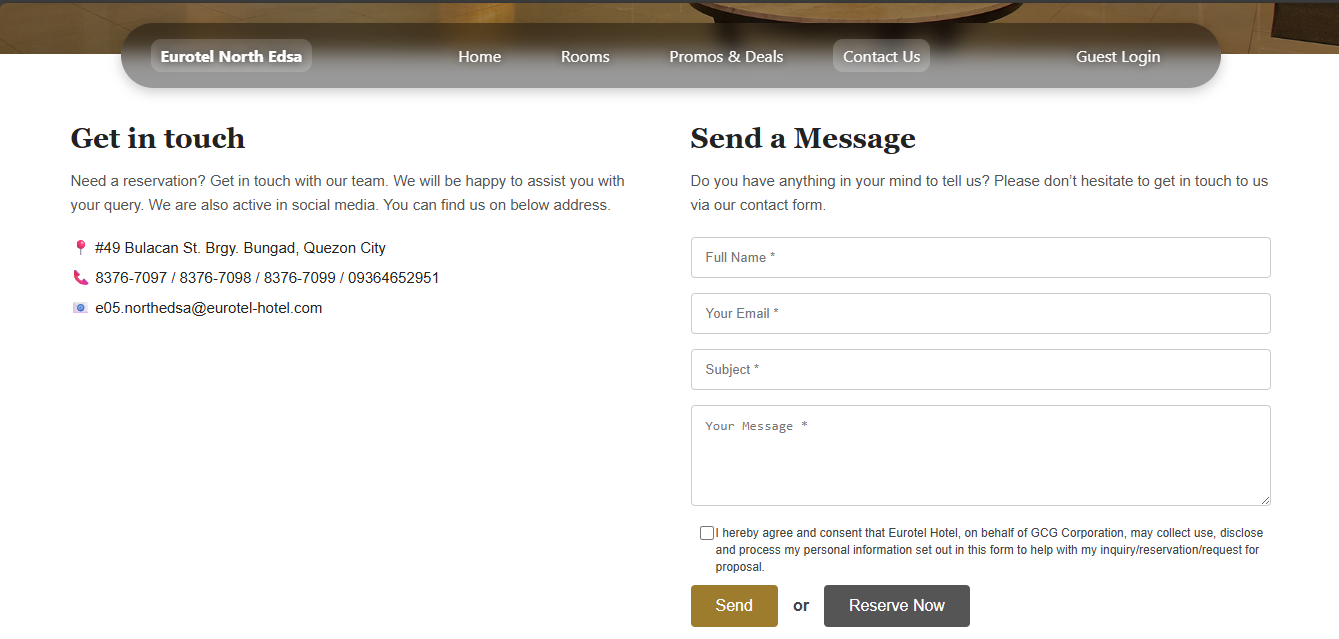
*Image 2. Log-In Form*

The Log-In Form provides two login portals: Guest Login for users to create and track reservations, and Admin Login for administrators to manage and approve bookings. This ensures secure access and proper role-based control.

**

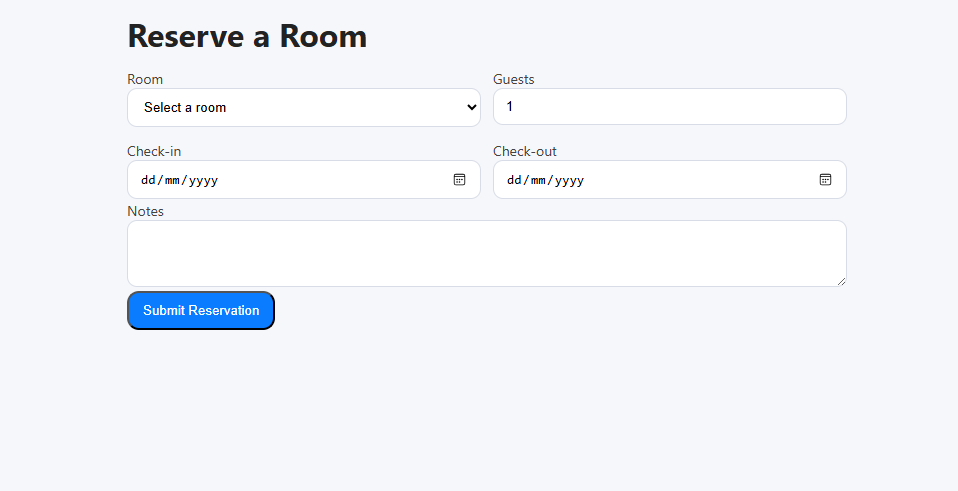
**

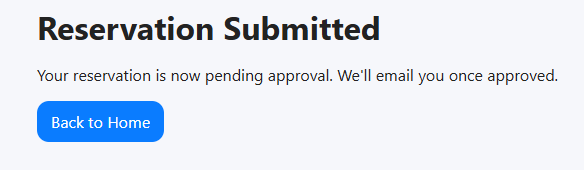
**

**

*Image 3. Dashboard*

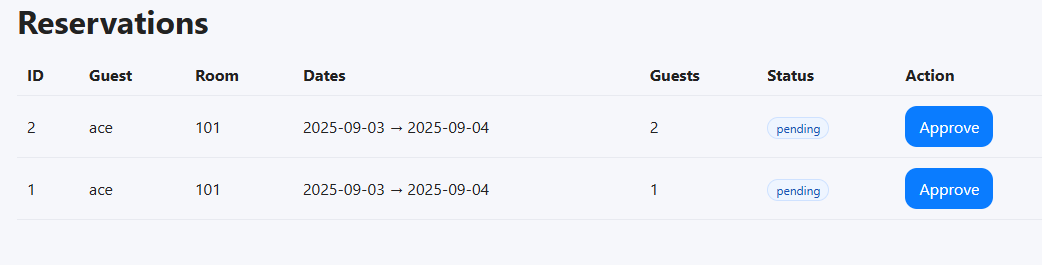
The Dashboard serves as the main interface of the system where guests can access different features. It displays available rooms with corresponding rates, ongoing promos and deals, and a quick reservation form. It also provides contact details and a message form for customer inquiries.

**

**

*Image 4. Reservation Form*

The Reservation Form allows guests to book a room by filling out necessary details such as check-in date, check-out date, and number of guests. Once the form is completed, the system records the request and forwards it for approval.



*Image 5. Managing Room Reservation*

The Managing Room Reservation displays essential details such as guest name, room number, reservation dates, number of guests, and booking status. From this interface, the administrator can approve or update reservations to ensure proper room allocation and avoid conflicts or double bookings.

*It***COMPONENT DESIGN**

The component design explains the system’s structure by breaking it down into different parts or modules. Each component has its own role, inputs, outputs, and connections with other components. This helps keep the system organized, easy to maintain, and ready for future improvements. The design also gives clear descriptions of the user interface, database, business logic, and external connections. By defining the role of each component, the system can be developed step by step, tested separately, and combined smoothly into one working system.

**Key Components and Modules**

* **Reservation Module –** Allows guests to book rooms in advance, check room availability, and receive booking confirmations.
* **Walk-in Booking Module –** Handles on-the-spot reservations at the front desk, updating room status instantly to avoid conflicts.
* **Room Management Module –** Tracks room status in real time (available, occupied, or under maintenance) to support efficient operations.
* **Authentication and Access Control –** Ensures secure login for admin and administrators with role-based access to system functions.
* **Payment Module –** Manages walk-in payments made over the counter. Records guest payments, verifies transactions, and links them with booking details for accurate financial tracking.
* **Reporting Module –** Generates summaries of reservations, occupancy rates, and transaction records to assist management in decision-making.

**Interface Specifications**

The Hotel Reservation Management System (HRMS) makes use of REST APIs to support communication between the client-side interface and the server. These APIs are responsible for handling important functions such as online and walk-in reservation processing, room availability updates, and guest record management. To ensure secure access, an Authentication API is implemented, which manages user login, session control, and role-based permissions for admin and administrators. All data exchanged between the client, server, and database uses JSON format, making the system consistent, lightweight, and platform-independent. This design allows the HRMS to remain scalable, secure, and adaptable for future improvements or integration with third-party services.

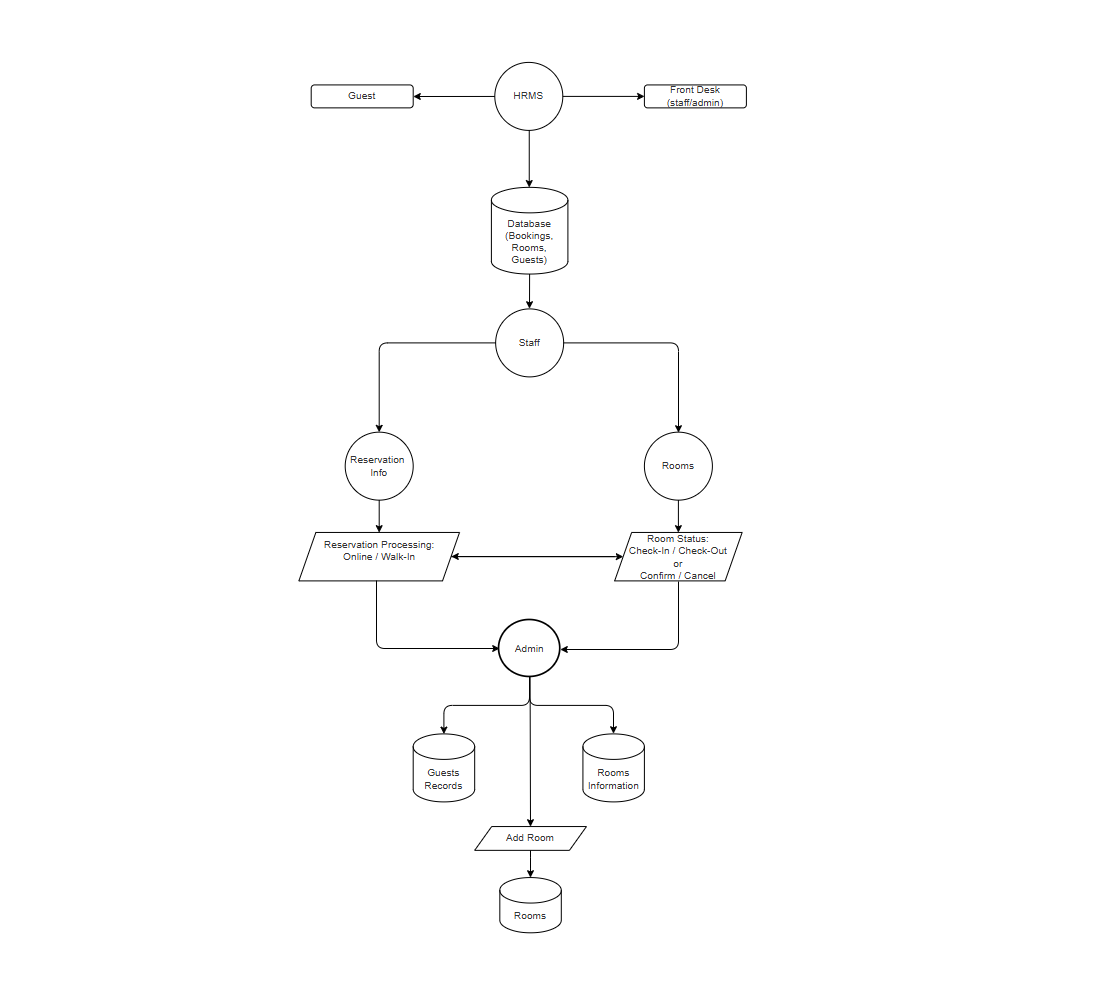
**Dependency Management**

The Hotel Reservation Management System relies on different software tools, frameworks, and services to operate efficiently. It uses a database management system for storing guest and reservation records, web frameworks for backend processes, and front-end libraries for delivering a user-friendly interface. External APIs, such as payment gateways and email services, are also integrated to support secure transactions and notifications. To manage these dependencies, package managers are used to handle installation, updates, and compatibility. With proper dependency management, the system remains stable, secure, and easy to maintain as technologies continue to evolve

## **DATA FLOW DIAGRAM**

In this study, Data Flow Diagrams (DFDs) are utilized to illustrate how information moves throughout the system. These diagrams highlight the relationships between data sources, processes, and outputs, making it easier to visualize how information is collected, processed, and stored.

**Level 2 Data Flow Diagram**

The Level 2 DFD expands the Hotel Reservation Management System into its main sub-processes. Guests can make reservations online or as walk-ins, with booking details stored in the database. The Reservation Processing Module handles these requests, generating confirmations and updating guest records. The Room Management Module allows admin and administrators to manage room availability, check-ins, check-outs, cancellations, and confirmations. Both modules rely on the database to provide accurate and up-to-date information on bookings and room status. Administrators also have access to add or update room records and generate reports, ensuring efficient monitoring of hotel operations and service flow.

*Image 7. Level 2 Data Flow Diagram*

## **SECURITY DESIGN**

The system is developed with strict security measures to protect the confidentiality, integrity, and availability of all data it handles. This includes sensitive details such as guest information, booking records, payment transactions, and system activity logs. Security practices follow the principle of least-privilege access, accountability through audit trails, and safeguards against unauthorized use. Moreover, the system complies with data privacy regulations and hotel policies, especially in relation to guest consent, data retention, and access control.

**Authentication and Authorization Mechanisms**

The Hotel Reservation Management System implements authentication and authorization mechanisms to ensure that only legitimate users can access the system and its resources. Authentication is handled through secure login credentials, such as unique usernames and encrypted passwords, which verify the identity of hotel admin and administrators. Additional methods like two-factor authentication may also be applied for stronger security.

Once authenticated, authorization defines the level of access each user is permitted. For example, administrators can manage system settings, generate reports, handle reservations and check-ins. This role-based access control prevents unauthorized actions and protects sensitive information such as customer records and financial transactions. Together, these mechanisms strengthen the security of the system and help maintain data integrity and confidentiality.

**Data Encryption and Protection Measures**

The Hotel Reservation Management System incorporates strong data encryption and protection measures to safeguard sensitive information such as guest details, reservation records, and payment transactions. The system uses SSL/TLS protocols to secure communication between the client and server, ensuring that data transmitted over the network cannot be easily intercepted. Stored data, including customer information and financial records, is encrypted in the database to prevent unauthorized access.

In addition, role-based access control is implemented so that only authorized admin can access specific system functions, minimizing the risk of data misuse. Regular security patches, backups, and monitoring tools are also applied to protect against breaches, data loss, and system failures. By combining these encryption methods with strict protection measures, the Hotel Reservation Management System ensures that customer trust, privacy, and security are always maintained.

## **PERFORMANCE DESIGN**

The Hotel Reservation Management System is built to provide fast, reliable, and secure operations for both guests and admin. Its main goal is to make the reservation process more convenient by reducing delays during booking, check-in, and check-out. The system updates room availability in real time to avoid double bookings and ensure accuracy in reservations. By automating routine tasks such as handling guest records, confirming payments, and generating reports, the system helps admin work more efficiently and focus on customer service. At the same time, it protects sensitive guest details by using secure logins, data storage, and controlled access for authorized admin only. With these features, the system is expected to improve hotel operations while giving guests a smoother booking experience.

**Strategies for Optimizing System Performance**

* **Load Balancing:** When there are many users at the same time, the system will distribute requests across different servers. This helps avoid slowdowns and ensures that online reservations and front desk operations continue to run smoothly even during busy hours.
* **Database Optimization:** The database will be arranged and managed in a way that improves speed and efficiency. This includes using indexes, creating effective query patterns, and organizing data properly. In some cases, certain data can be duplicated (denormalized) to make searches and reports faster.
* **Scalability:** The system is designed to handle growth by supporting both vertical scaling (adding more power to the server) and horizontal scaling (adding more servers). This makes sure the system can still work efficiently as the number of bookings and users increases in the future.
* **Asynchronous Processing:** Tasks that are not urgent, like generating reports or sending confirmation emails, will be handled in the background. This way, important actions such as reservations and check-ins remain quick and uninterrupted.
* **Resource Monitoring:** The system will regularly track the use of resources such as CPU, memory, and network activity. Monitoring these factors helps identify possible issues early and ensures the system continues to perform well.

**Performance Testing Plan**

To ensure that the Hotel Reservation Management System works smoothly and can handle the daily operations of Eurotel North Edsa Hotel, different performance tests will be conducted:

* **Load Testing** – The system will be tested by simulating multiple guests making reservations, checking room availability, and processing payments at the same time. This will help measure how fast and stable the system is during peak hours.
* **Stress Testing** – The system will be pushed beyond its normal capacity, such as handling an unusually high number of bookings, to check if it can still function properly and recover in case of overload.

## 

## 

## 

## 

## 

## 

## 

## 

## **ERROR HANDLING AND LOGGING**

This section explains how the system manages errors to maintain stability and reliability. Issues will be detected and handled without disrupting core functions such as room reservations, check-in/check-out, and user authentication. When errors occur, the system will record detailed logs for administrators while displaying simple, user-friendly messages to guide admin or guests in resolving the issue.

**Error Codes and Message**

* Invalid admin credentials: Will alert when the user enters incorrect or missing fields such as username, or password. The system will alert the user to check and correct the error.
* Email is already registered: Triggered when guests try to sign up using an email address that already exists in the database.
* Reservation ID not found: The reservation you're looking for does not exist. Please check the ID or contact support.
* Room number already exists: Indicates when a guest or admin is trying to add a new room to the system using a room number that is already present in the database. This prevents duplicates and ensures each room can be uniquely identified.
* Required fields cannot be empty: Occurs when the user submits a form without filling out mandatory fields. Such as, Full Name, Room Type, or Check-In/Check-Out.

**Third-Party Integrations**

To improve performance, scalability, and user experience, the Hotel Reservation Management System integrates several third-party tools.

* **API:** Serves as the main backend storage, managing structured data such as guest information, reservations, room records, and transaction logs. It is considered an external dependency since the system relies on it for data persistence.
* **PHP (Backend):** Powers the system behind the scenes by handling tasks like processing reservations, managing user accounts, and communicating with the database. Its flexibility makes it a reliable choice for building web-based applications.
* **CSS (UI Design):** Shapes the look and feel of the application by providing responsive layouts, clean styles, and visually appealing elements. It helps create a smoother and more user-friendly experience without changing how the system works.
* **PHPMailer (Email Handling)** PHPMailer is integrated into the backend to handle email functionalities within the Hotel Reservation Management System. It enables the system to send automated emails such as:
* Booking confirmations to guests
* Password reset and account verification emails
* Notifications regarding reservation updates or cancellations

**Integration Points and Data Exchange Formats**

|  |  |  |
| --- | --- | --- |
| **Service** | **Integration Point** | **Data Exchange Format** |
| MySQL Workbench | The PHP backend communicates with the MySQL database using SQL queries to store and retrieve structured data such as reservations and guest information. | SQL (Structured Query Language) |
| PHP | Acts as the core backend processing unit, receiving input from the UI, validating data, handling business logic, and communicating with the database and email services. | PHP data structures, HTTP requests/responses (JSON format) |
| PHPMAILER | Integrated within the PHP backend to send automated emails triggered by system events (booking confirmation, password reset, etc.). | MIME-formatted emails via SMTP protocol (supports HTML and plain text) |
| CSS | Provides styling for the frontend UI, enhancing user experience by making the application responsive and visually appealing. | CSS stylesheets applied to HTML elements |

## 

## **Deployment Plan**

## The Hotel Reservation Management System will be deployed in a step-by-step approach to make sure it functions properly before being released for actual use. Development will focus on building and improving system features in a controlled setup, allowing changes to be applied and reviewed safely. After this, the system will go through a testing phase where both functional and non-functional requirements will be checked to confirm that everything works as intended. Once testing is complete, the system will move to a staging environment, which is designed to imitate the real hotel operations. This phase acts as the final checkpoint where the team can validate performance, security, and usability before the system goes live. The last stage is production, where the system will officially be deployed. To support smooth updates, this setup will automate testing, building, and deployment, which reduces errors, minimizes downtime, and ensures that improvements can be delivered consistently.

**Hardware and Software Requirements for Deployment**

Hardware:

* Minimum intel core i5 10th gen - The device that will be used for the system to work. A maximum of 3 devices for the department is needed.

Software:

* Visual Studio Code - To create dynamic, interactive, and engaging user experiences on the computer laboratory management system. It is a versatile, client-side/server-side scripting language that works alongside HTML and CSS to provide functionality and interactivity.

**Configuration Management and Version Control Procedures**

**Version Control:**

The current system development does not use formal version control tools like Git. Instead, versioning is managed manually by keeping backup copies of project files at different stages or dates to track changes. For future improvements, adopting a version control system is recommended to enhance collaboration and change management.

**PHPMailer and SMTP Setup:**

PHPMailer is utilized to send automated emails such as booking confirmations and password resets. It requires configuration of SMTP server settings including host, port, authentication credentials, and encryption protocol to enable secure email transmission. These settings are stored separately from the main codebase to allow for easy updates without modifying source code.

**Manual Configuration:**

Configuration files such as database credentials and SMTP details are maintained outside the main codebase (e.g., in environment files or configuration scripts) to facilitate secure and flexible management.

**Backup Procedures:**

Regular manual backups of files and databases are conducted to prevent data loss and allow for recovery in case of failures.

**MAINTENANCE SUPPORT**

To keep the Hotel Reservation Management System running smoothly, a proactive maintenance and support plan will be applied. Routine system checks will be carried out to ensure that performance remains stable, the database is optimized, and all features are working properly. Preventive maintenance will also be scheduled to minimize possible errors and avoid system downtime that could affect hotel operations. The system will include monitoring tools to track performance in real time, detect unusual activities, and send alerts whenever issues arise. Logs will be maintained to help identify the root cause of problems and guide faster troubleshooting. In addition, a dedicated support process will be established to address user concerns, technical problems, and service requests, making sure that both admin and guests experience reliable and uninterrupted service.

**Procedures for Handling Software Updates, Patches, and Bug Fixes**

* Minor Fixes – Small corrections, such as resolving booking errors or patching security issues, will be applied immediately after quick testing to avoid disrupting hotel operations.
* Planned Major Updates – Bigger improvements, like adding new reservation features, optimizing system speed, or upgrading the database, will be scheduled and tested thoroughly before deployment.
* Testing Before Release – All updates will be checked in a controlled environment to ensure they function correctly and do not cause conflicts with existing features.
* Version Control – Every change will be tracked to maintain consistency and to make it easier to identify the source of problems if they occur.

**Support Framework**

* User Support **–** A helpdesk system will be provided for hotel admin to report issues and request assistance. Concerns will be prioritized depending on urgency to make sure critical problems are resolved quickly.
* Training and References **–** Manuals and simple training sessions will be given to admin so they can use the system effectively and easily adapt to new updates or added features.
* Data Backup and Recovery **–** Regular backups of guest records, room availability, and transaction details will be scheduled to prevent data loss. Recovery steps will be in place to restore operations if unexpected failures occur.
* System Security **–** Routine checks will be conducted to protect sensitive guest information and maintain system reliability. Security updates will also be applied to reduce risks of breaches.

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

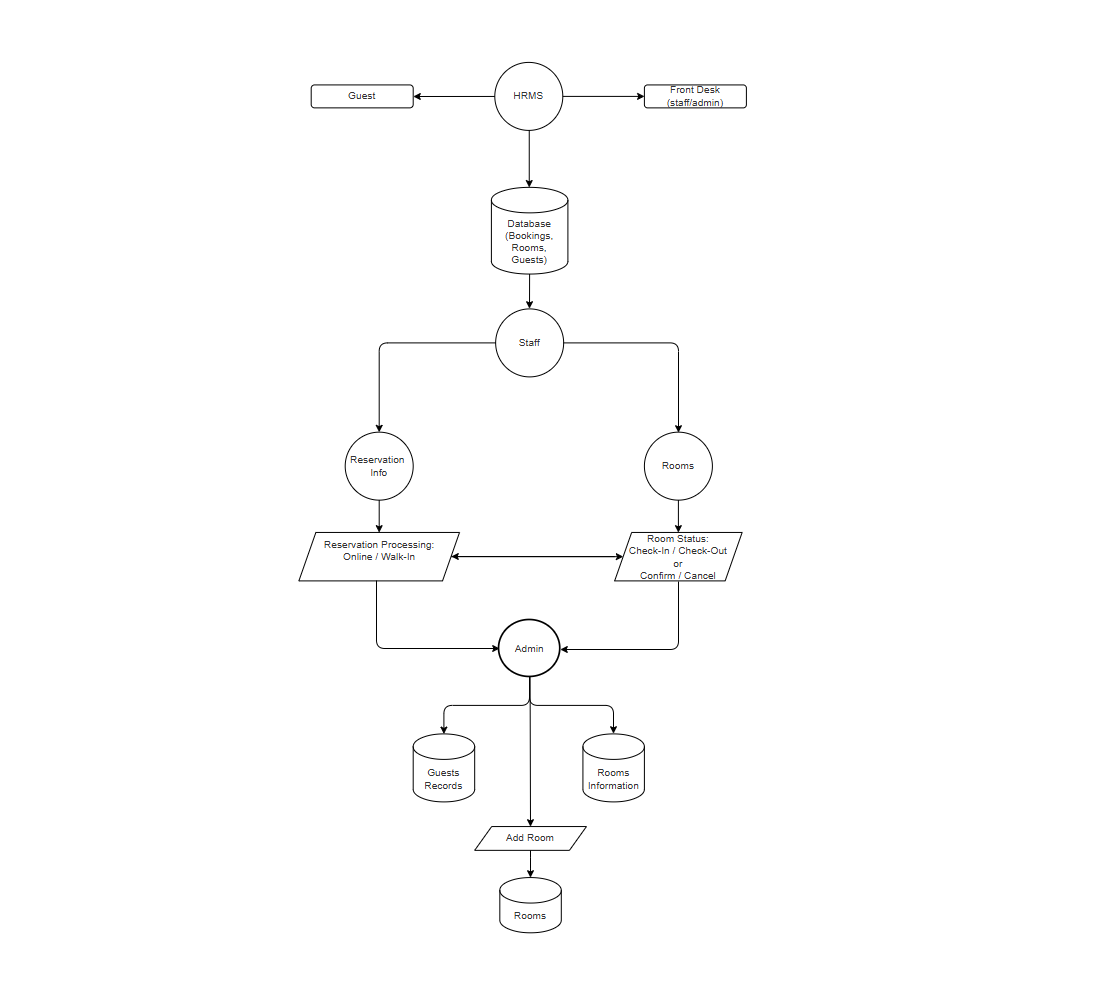
## **REVISION HISTORY**

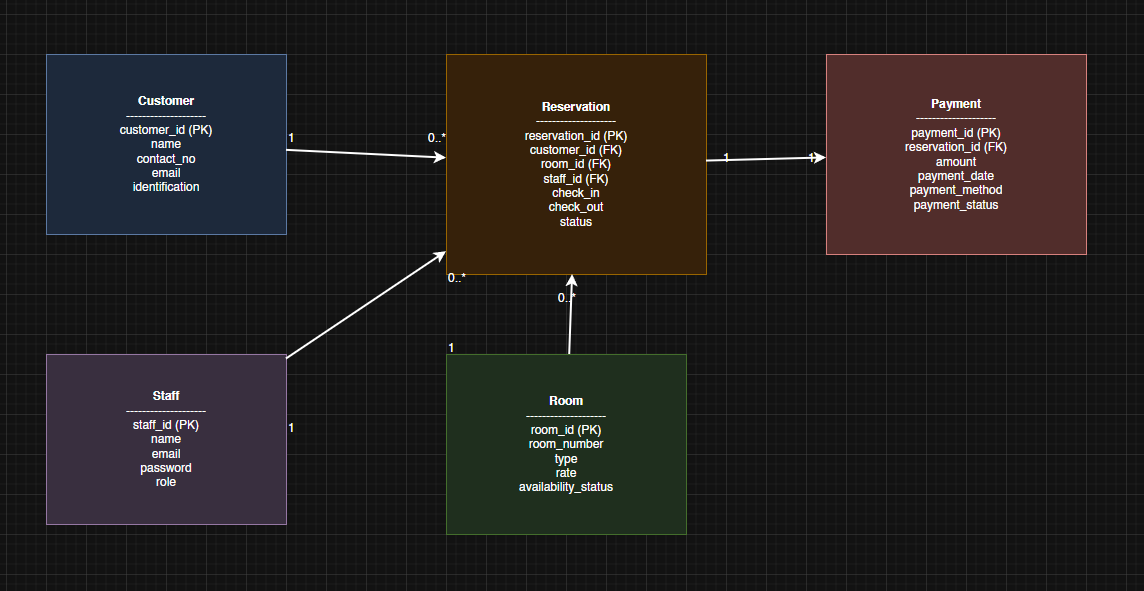
This section documents all changes made to the design document over time. Each revision entry includes the version number, date of update, and a summary of modifications. It ensures proper tracking of progress, accountability, and clarity in the evolution of the system’s design.

|  |  |
| --- | --- |
| DATE | CHANGES MADE |
| August 18, 2025 | Prepared the documents |
| August 19, 2025 | Gather some information or details for the paper |
| August 20, 2025 | Update the UI design |
| August 21, 2025 | Create data flow diagrams |
| August 23, 2025 | Add the information in the paper |
| August 26, 2025 | Reviewed the content of the paper |
| August 30, 2025 | Revision of paper and Update of System |

## **APPENDIX**

The Appendix section contains additional materials that support the main content of this document. It may include diagrams, charts, and other references that provide further explanation of the system but are not part of the core discussion. Examples include database designs, data flow diagrams that help illustrate how the system works.

**Appendix A - Supporting Diagrams**

*Image A.1: Data Flow Diagram of the system*

*Image A.2: Entity-relationship diagram (ERD) of the database schema*