Building a Modern Hotel Management Application with Database System

Course ID.: CPE241 Database Systems

Members:

Chawit Pimapansri
ID: 65070503411 <chawitpimapansri@gmail.com>
Sorawit Tonpitak
ID: 65070503438 <rocbot01@gmail.com>
Nichaporn Manachaiprasert
ID: 65070503446 <patty.nichapornn@gmail.com>
Thanakit Chokbunsuwan
ID: 65070503448 <tchokbunsuwan@gmail.com>
Yuil Tripathee
ID: 65070503480 <yuil.trip@mail.kmutt.ac.th>

Submitted To:

Department of Computer Engineering
in partial fulfillment of the requirements for the course of CPE241 Database Systems.

Instructor:

Asst. Prof. Dr. Phond Phunchongharn
Department of Computer Engineering

KMUTT
May, 2024

ABSTRACT

We started this project to understand and apply database systems into a real world scenario. Hotel management system is one of the popular application projects being used in universities around the world to let students exercise their database design and implementation skills. We started with learning through the domain knowledge. However, things did not go as planned and we had to revise our ER model throughout our journey which we have documented as part of this report. Finally, a web application was developed (using Node.JS + MongoDB stack combination) and improvised so that multiple stakeholders can interact with our application. Each kind of end user has their own permission levels to different resources depending on the stake of their interaction with a hotel environment.

Keywords: database, hotel management, ER model

TABLE OF CONTENTS

List of Tables

List of Figures

CHAPTER 1

BACKGROUND

1.1 Domain Analysis for the project

With the intensified competition in hotel industry, their strive to improve their operational management has led to extensive use of "online hotel management system". These systems are dedicated to addressing hotel management issues in a more practical, well-organized, quick way of processing the service from hotel both nearby and remote customers with a more friendly GUI oriented experience. [1]

1.1.a Stakeholders and their concerns

We distinguish primary and secondary stakeholders connected to this hotel organization. This helps us understand the domain of hotel better and do not mix features from other domain such as enterprise management. This process will follow to draw out entities, their attributes and relationships.

1.1.a.1 Primary stakeholders

We are especially going to deal with these stakeholders for the context of our application.

Admin They are the apex decision makers in the organization. They can be hotel owner, highest level of management or the IT staff being outsourced by the highest decision making authority.

Management Staffs working on reception, dining, and other management related areas.

Ground staff Staffs working on gardening, housekeeping, security. Management and ground staffs are supposed to have different level of abstraction when it comes to data access.

Guest The end customer. They can be at the hotel for just dining, book room or attend conference. In such case guest entity can be a person or an organization that

can book hotel facility for social gathering such as corporate parties, academic conferences, hackathons and much more.

1.1.a.2 Secondary stakeholders

Although they have some influential stake at the hotel industry, we are going to cut their concerns off as our product would be more close to ERP (Enterprise Resource Planning) tools by addressing their feature requests rather than hotel management application.

Legal team They oversee legal matters in the organization.

Audit team Also known as auditors or charter accountant, they look after balance sheets, taxation and other fiscal affairs.

Human resources They manage staff operations such as recruiting, training, performance assessment and so on.

Supply chain This includes supply chain operations related to interior design, kitchen ingredients, gifts, and decoration being brought by business in a routine basis.

Procurement This is part of occasional purchases rather than regular supply chain issue. This can cover damage control for accidents, damages to property & their necessary replacement or even upgrade of hotel infrastructure.

1.2 Existing Solutions in the market

Some of the prominent hotel operating system (or more popularly known as property management system) include the following as of 2024: [2]

Oracle Hospitality OPERA Cloud Services They are a cloud-based, mobile-enabled platform for next-generation hotel management. Based on OPERA, the leading enterprise solution suite for the hospitality industry, OPERA Cloud offers an intuitive user interface, comprehensive functionality for all areas of hotel management, secure data storage and hundreds of key partner interfaces to meet the needs of hotels of all types and sizes. Oracle Hospitality is a proud AHLA Platinum Partner.

SkyTouch Technology This company has provided the most widely-used hotel PMS software to solve all property management needs since 2013. Hoteliers seek our cloud-based software with 24/7 support for our easy-to-use interface abilities. Our products efficiently manage rates & availability, distribution & booking, guest experience, and more.

Cloudbeds It is a leading platform serving thousands of properties in more than 150 countries worldwide.

RoomRacoon It is an innovate all-in-one Hotel Management System for independent hotels, B7Bs, hotels, apartments, and vacation rentals. RoomRaccoon is one of the most connected solutions on the market and offers over 400 integration with business solutions such as Lightspeed, Sage, Booking.com and Expedia.

WebRezPro WebRezPro is a secure, one-stop-shop cloud property management system that streamlines operations for independent lodgings of all shapes and sizes. The automated, easy-to-use system saves time and money while boosting direct bookings and improving guest experience.

Beside academic version, hotel management systems are also popular choice among IT and engineering institutes to give hands-on experience to student regarding the software engineering concepts like object-oriented software engineering and database management. [1][3]

CHAPTER 2

INTRODUCTION

2.1 Project Design Brief

2.1.a Project Description

We aim to digitize hotel property management system. In comparison to existing market offerings, we also attempt to provide unique value proposition by being cost effective and lean than others. This helps businesses guarantee easy management and supervision, less workload, and consistency on both information and business domains.

2.1.b Project Goals

Adequate record keeping This is achieved by understanding and implementing database design for Hotel System using ER diagram.

Reduced fraud incidence We make sure transaction holds consistency and integrity, compliant with the selected database program (relational database should have ACID compliant and non-relational comply with CAP by design).

Data-level security Implemented using role based access control by design.

Reduced operational costs Having a documented data dictionary can help the software scale and port to other branches.

2.1.c Project Scope

Business premises: Our client has a hotel at **only 1 location**. The scope is constrained in order to boost software development turnaround period such that rapid feedback can be received from hospitality sector for customization rather than building a complex monolith application and making huge expensive changes later on.

2.1.c.1 In-scope Functions

We deduce functional requirements of our system by being centered on in-scope functions. [4]

1. Guest information
2. Staffs (management and ground staff)
3. Ground staff information
4. Task ticket for Ground Staff
5. Job category and position
6. Customer reservation (booking)

   Check in and Check out

   Seasons discount

   Invoicing (could go through service package)
7. Room (physical status)
8. Room again (features and booking status)

   Room Rate Code
9. Room types (bedroom, conference, banquet)
10. Services (catering and others)

2.1.c.2 Out-scope Functions

Some of the non-functional requirement were eased up because it does not particularly address the scope of neither database design concept nor core hotel business functions:

Software Infrastructure Security: Within this project only basic authentication and RBAC (Role Based Access Control) will be exercised in the database design & implementation phase. Security issues related to server infrastructure, networking setup and human factors were dismissed as part of this project's requirements.

Relevant industrial compliance: We will only be focusing on quality of software and standards associated to it. Regulatory issues compliant with hospitality industry will be implemented as client's customization needs following the completion of required functional requirement.

2.2 Team member's responsibilities

All members in the group had considerably equal amount of stake when it comes to deciding relational model of the database, normalization and transaction. However, tasks other than database design and implementation were split by respective member's area of expertise. Also, each members individually designed couple of complex query reading and/or manipulating the data from more than two relations.

COMMON tasks include database design using ER model, implementation in both SQL and NoSQL platforms with individual bench-marking, review, & performance evaluation. We also designed form and reports query together using DML and Mongo API together.

3411 and 3438 performed back-end integration, translate SQL output to a from that can be rendered on user's end (JSON for web and mobile app platforms), apart from designing databases.

3446 and 3448 designed forms and reports & implemented on front-end stack using NextJS, apart from database related works.

3480 handled literature review, business domain analysis, full stack integration and documenting the project, apart from database design and implementation.

2.3 Expected project deliverable

By the completion of this project, the development team is committed to deliver the following resources associated with this project:

1. Project Report
2. ER Model Diagram
3. Data dictionary
4. User Interface (Form and Report Design)
5. Major User Views
6. Source codes (for application and related SQL query)

CHAPTER 3

DATABASE DESIGN

3.1 User and User Roles

This is crucial to maintain database security via RBAC (Role-Based Access Control).

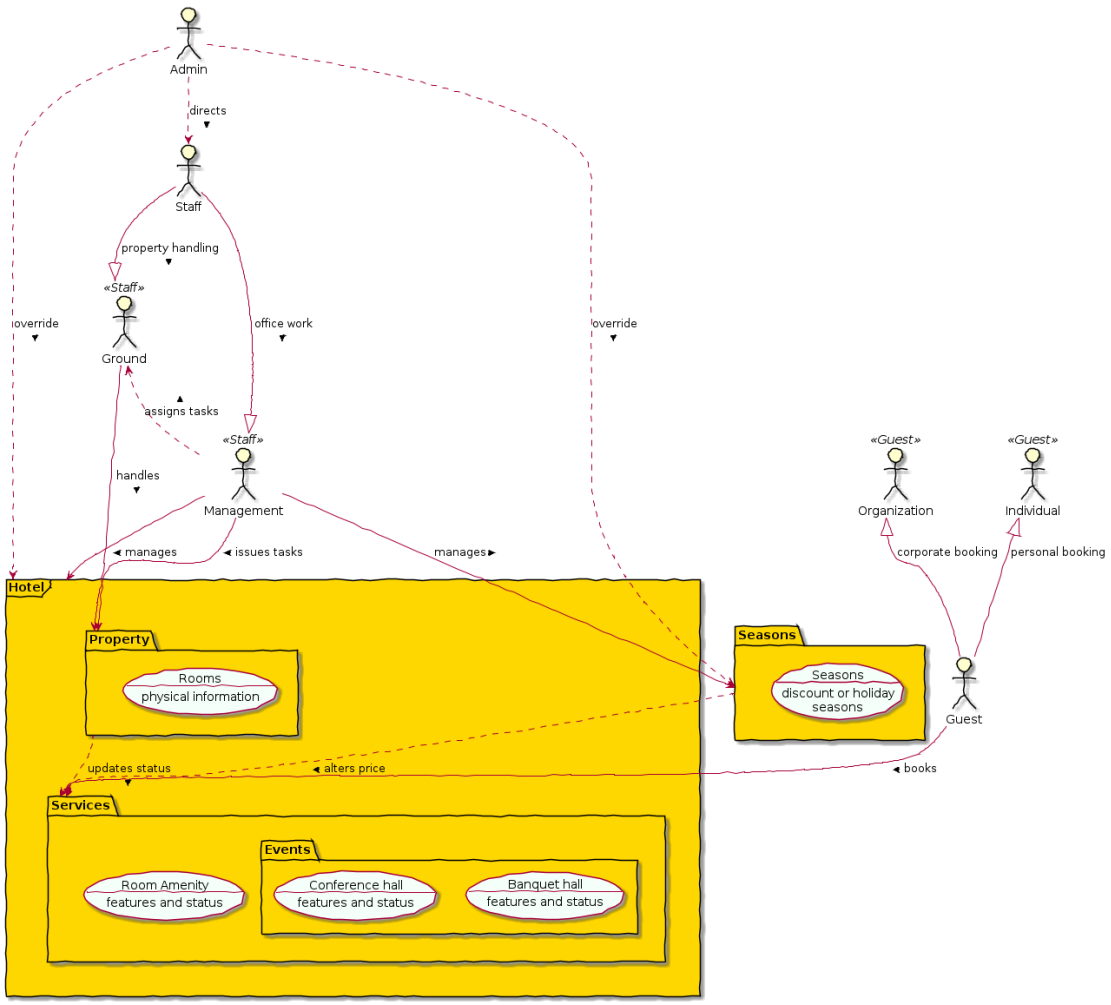

Figure 3.1: Use case diagram to represent actors and entities

As from the premise in chapter 1, we have three major user types of our applications:

admin, **staff** and **guest**. **Staff** is further separated into **management** and **ground** staff. Likewise, **guest** is also separated such that they can guest entity can represent **individual** or an **organizations**. This comes handy since guests reserving these properties would like to taxes on their expenses in various fashions. Therefore, we have following major user roles:

1. Admin
2. Management <staff>
3. Ground <staff>
4. Individual <guest>
5. Organization <guest>

3.1.a User Roles Design for RBAC

For reference, we use A for Add, V for View, E for Edit, and D for Delete to represent user roles.

Relation	Admin (DBA)	Management <staff>	Ground <staff>	Guest (both)
Hotel-Basic-Info	V, A, E, D	V	V	V
Facility-Info	V, A, E, D	V, A, E	V	V
Property-Info	V, A, E, D	V, A, E	V	V
Service-Info	V, A, E, D	V, A, E	-	V
Staff-Info	V, A, E, D	V (self)	V (self)	-
Mgmt-Staff-Info	V, A, E, D	V	-	-
Contact-Ticket	V, A, E, D	V, E	-	V, A (self)
Ground-Staff-Info	V, A, E, D	V	V (self)	-
Ground-Task-Info	V, A, E, D	V, A, E	V (self)	-
Ground-Task-Ticket	V, A, E, D	V, A, E, D	V, E (self)	-
Guest-Info	V, A, E, D	V	-	V, A, E, D (
Seasons-Discount-Info	V, A, E, D	V	-	V
Reservations	V, A, E, D	V, A, E, D	-	V, A (self)
Invoice	V, A, E, D	V, A, E	-	V, A (self)

Table 3.1: Major user roles

For abstraction, both individual and guests representing organization are hold under same entity. This can be customized later based on customer feedback. Also, we would like to merge upper management to Admin access and use Management <staff> entity for field level management. This is for not allowing all levels of managerial positions have Admin level access. However, in small hotels upper management usually has data access up to admin level in absence of owner.

3.1.b Relation Design

3.2 ER Model Diagram

ER model represents business needs to remember in order to perform business operations. [5] Among different diagramming convention techniques used to represent ER model of the database, we selected UML class diagram. We prefer this over originally accepted Chen's notation considering its extensive ability to describe multiplicity between the relationships. [6]

The final version of ER model is presented in Figure 3.3. We have separated areas of concerns such as staff management, business management and property management into different packages. This helps development teams to split tasks on specific domains and develop each packages individually.

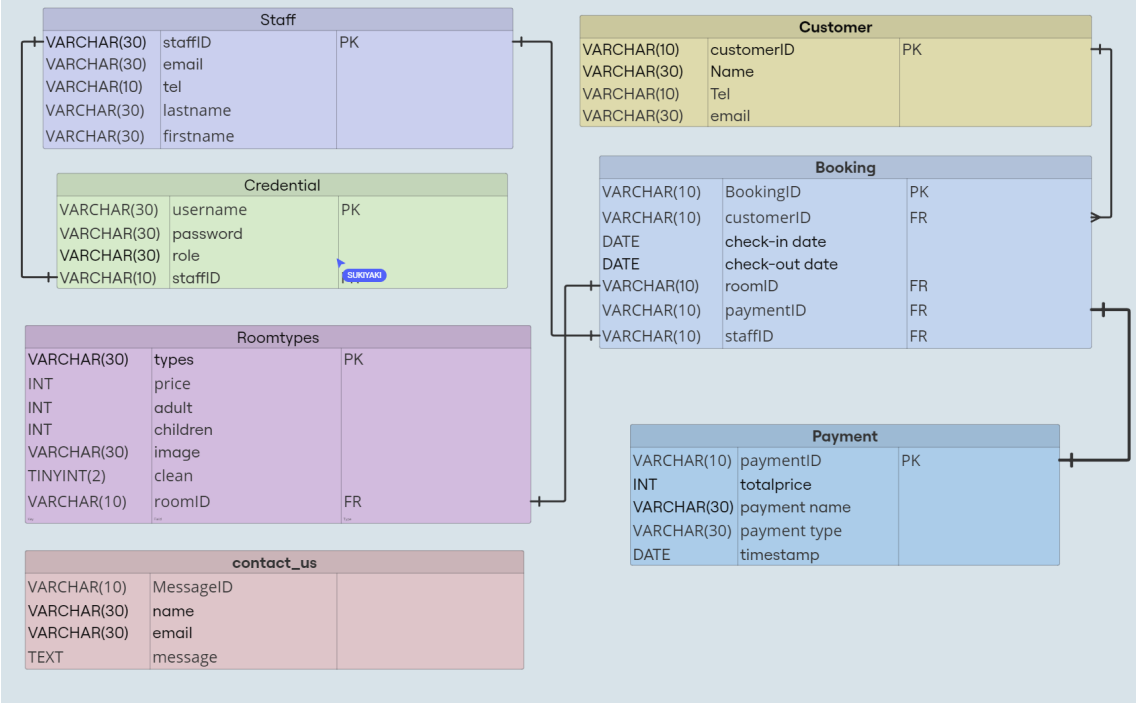

Figure 3.2: Initial draft for ER model diagram

3.3 Data Dictionary

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.2: Basic Format for Data Dictionary

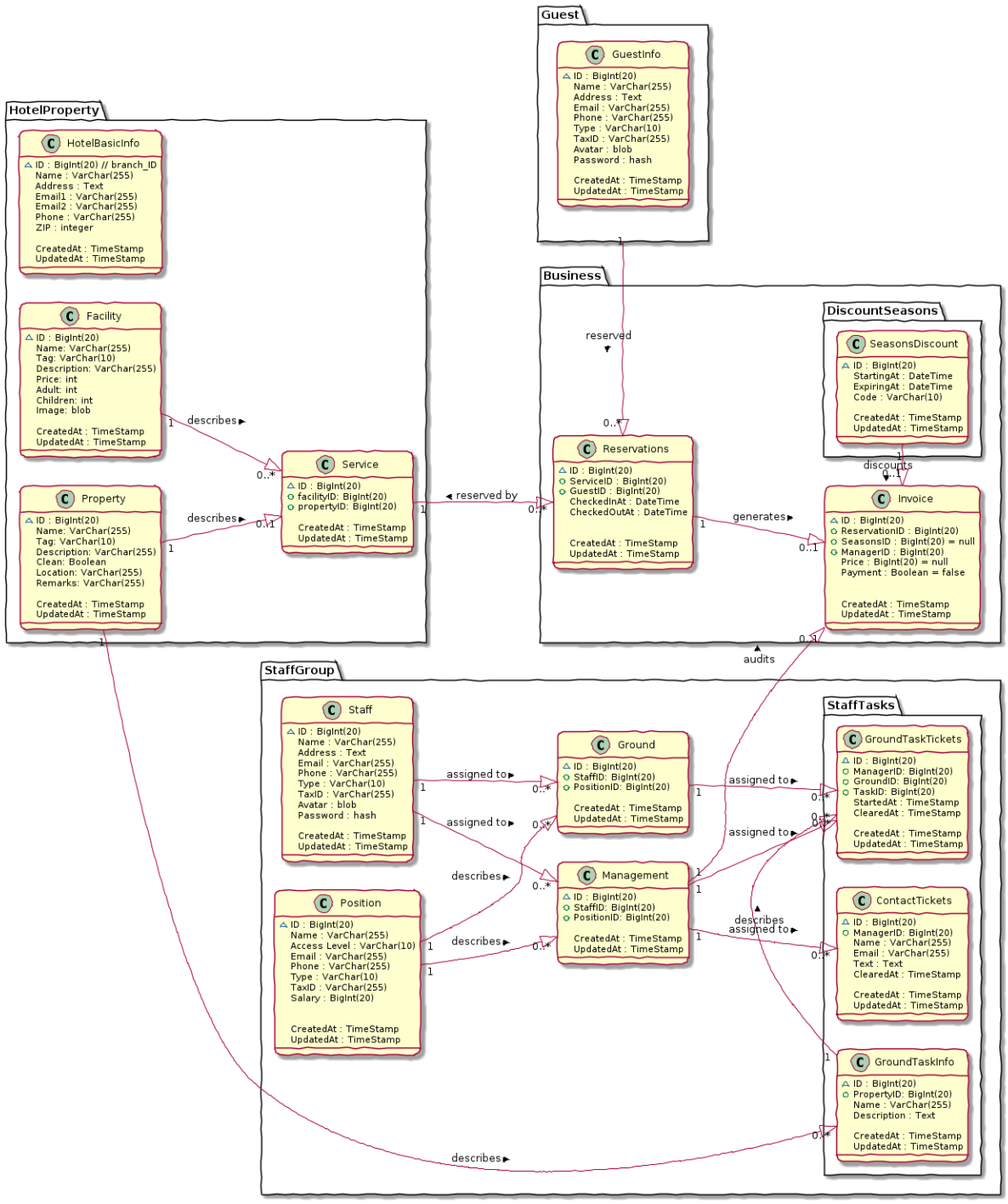

Figure 3.3: Final version for ER model diagram

3.3.a Guest Information

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
Name	VARCHAR	255	NOT NULL	
Address	TEXT		NOT NULL	
Email	VARCHAR	255	NOT NULL	
Phone	VARCHAR	255	NOT NULL	
Type	VARCHAR	10	NOT NULL	Individual or Organization
TaxID	VARCHAR	255		
Avatar	BLOB			
Password	HASH		NOT NULL	
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.3: Guest Info Table

3.3.b Hotel Property

3.3.b.1 Hotel Basic Information

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
Name	VARCHAR	255	NOT NULL	
Address	TEXT		NOT NULL	
Email1	VARCHAR	255	NOT NULL	
Email2	VARCHAR	255	NOT NULL	
Phone	VARCHAR	255	NOT NULL	
ZIP	INT	10		
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.4: Hotel Basic Information Table

3.3.b.2 Facility: Hotel Service Offerings

Now, this is the service offerings category. Another relation `Service` is associated which takes features of this relation and map them with individual rooms. Take this is type of Room (say Deluxe) that guests can book.

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
Name	VARCHAR	255	NOT NULL	
Tag	VARCHAR	10	NOT NULL	
Description	VARCHAR	255	NOT NULL	
Price	INT		NOT NULL	
Adult	INT		NOT NULL	
Children	INT		NOT NULL	
Image	BLOB			
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.5: Hotel Facilities Table

3.3.b.3 Property: Elements of Hotel Premises

This can be rooms, hallway or any entity within the compound. Ground staff handle them. However, only few components in compound such as room and hallway can get Service status. We will see that in below subsection.

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
Name	VARCHAR	255	NOT NULL	
Tag	VARCHAR	10	NOT NULL	type (room, banquet)
Description	VARCHAR	255	NOT NULL	
Clean	BOOLEAN		NOT NULL	
Location	VARCHAR	255	NOT NULL	
Remarks	VARCHAR	255	NOT NULL	
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.6: Hotel Property Table

3.3.b.4 Service

Now, this is the individual entity guests can use for reservation.

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
facilityID	BIGINT	20	FK	
propertyID	BIGINT	20	FK	
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.7: Hotel Service Table

3.3.c Business Context

3.3.c.1 Reservations

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
ServiceID	BIGINT	20	FK	
GuestID	BIGINT	20	FK	
CheckedInAt	DATETIME			
CheckedOutAt	DATETIME			
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.8: Reservations Table

3.3.c.2 Seasons Discount

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
StartingAt	DATETIME			
ExpiringAt	DATETIME			
Code	VARCHAR	10	NOT NULL	
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.9: Seasons Discount Table

3.3.c.3 Invoice

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
ReservationsID	BIGINT	20	FK	
SeasonsID	BIGINT	20	FK, NULL	
ManagerID	BIGINT	20	FK	
Price	BIGINT	20		
Payment	BOOLEAN		DEFAULT = FALSE	
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.10: Invoice Table

3.3.d Staff Operations

3.3.d.1 Staff

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
Name	VARCHAR	255	NOT NULL	
Address	TEXT		NOT NULL	
Email	VARCHAR	255	NOT NULL	
Phone	VARCHAR	255	NOT NULL	
Type	VARCHAR	10	NOT NULL	Internship status
TaxID	VARCHAR	255		Personal ID
Avatar	BLOB			
Password	HASH		NOT NULL	
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.11: Staff Info Table

3.3.d.2 Position

This is a metadata table for the list of positions in hotel.

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
Name	VARCHAR	255	NOT NULL	
Access Level	VARCHAR	10	NOT NULL	
Email	VARCHAR	255	NOT NULL	
Phone	VARCHAR	255	NOT NULL	
Type	VARCHAR	10	NOT NULL	
TaxID	VARCHAR	255		Tax Code
Salary	BIGINT	20		
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.12: Position Info Table

3.3.d.3 Ground Staff

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
StaffID	BIGINT	20	FK	
PositionID	BIGINT	20	FK	
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.13: Ground Staff Info Table

3.3.d.4 Management Staff

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
StaffID	BIGINT	20	FK	
PositionID	BIGINT	20	FK	
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.14: Management Staff Info Table

3.3.d.5 Staff Tasks

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
ManagerID	BIGINT	20	FK	
GroundID	BIGINT	20	FK	
TaskID	BIGINT	20	FK	
StartedAt	TIMESTAMP			
ClearedAt	TIMESTAMP			
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.15: Ground Tasks Ticket Table

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
PropertyID	BIGINT	20	FK	
Name	VARCHAR	255	NOT NULL	
Description	TEXT			
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.16: Ground Tasks Information Table

This table is also a metadata table for assigning tasks.

Field Name	Data type	Field size	Constraints	Description
ID	BIGINT	20	PK	Auto-increment
ManagerID	BIGINT	20	FK	
Name	VARCHAR	255	NOT NULL	
Email	VARCHAR	255	NOT NULL	
Text	TEXT			
ClearedAt	TIMESTAMP			
CreatedAt	TIMESTAMP	AUTO	NOT NULL	Auto
UpdatedAt	TIMESTAMP	AUTO	NOT NULL	Auto

Table 3.17: Contact Tickets Table

CHAPTER 4

USER INTERFACE DESIGN

4.1 Complex Transaction Form

4.1.a Registration Form (for `staff`)

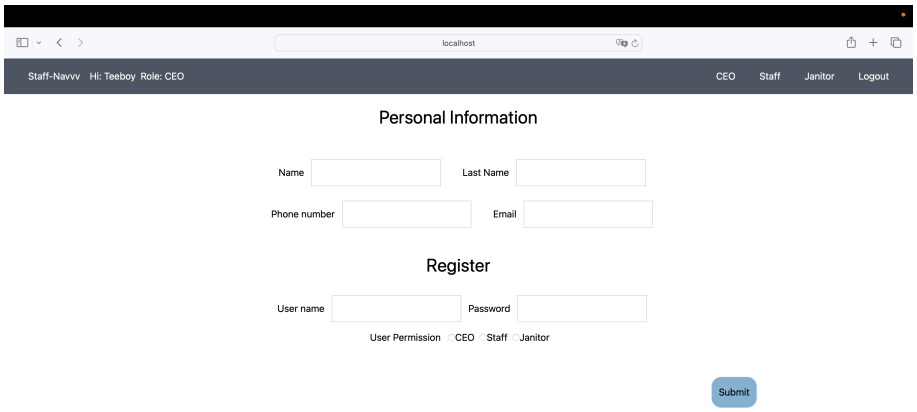

Figure 4.1: Registration form

It is capable of creating both staff and users through the same UI.
Relations involved is based on guest or staff:

Set 1 Guest only

Set 2 Staff, Position, Ground, Manager

4.1.b Booking form (for `user`)

Relations involved: Reservations, Service, GuestInfo

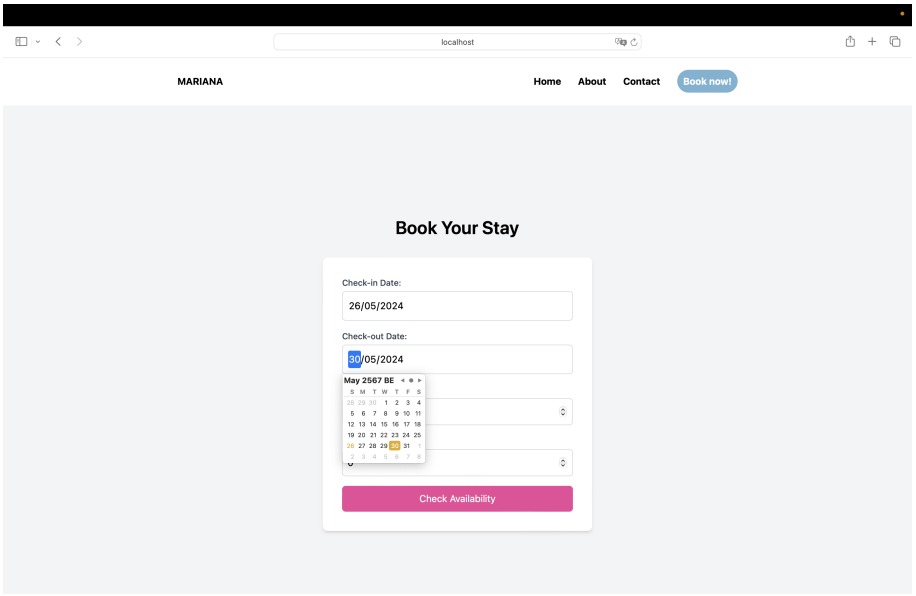

Figure 4.2: Booking form

4.1.c Housekeeping form

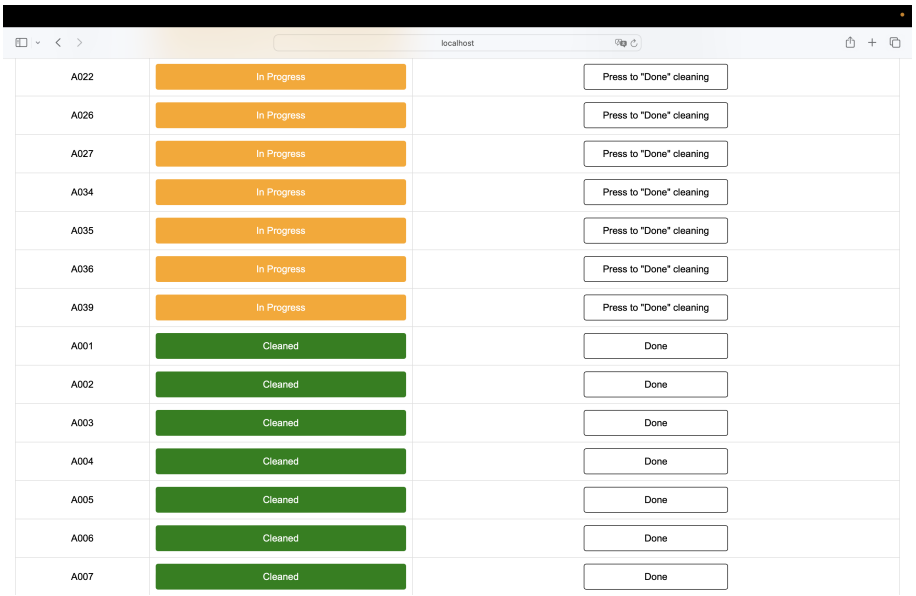

Figure 4.3: Housekeeping form

The housekeeping form also serves bidirectionally as a report.

Relations involved: Ground, Management, Ground Task Info, Ground Task Tickets

4.1.d Room form

Relations involved: Facility, Property, Service

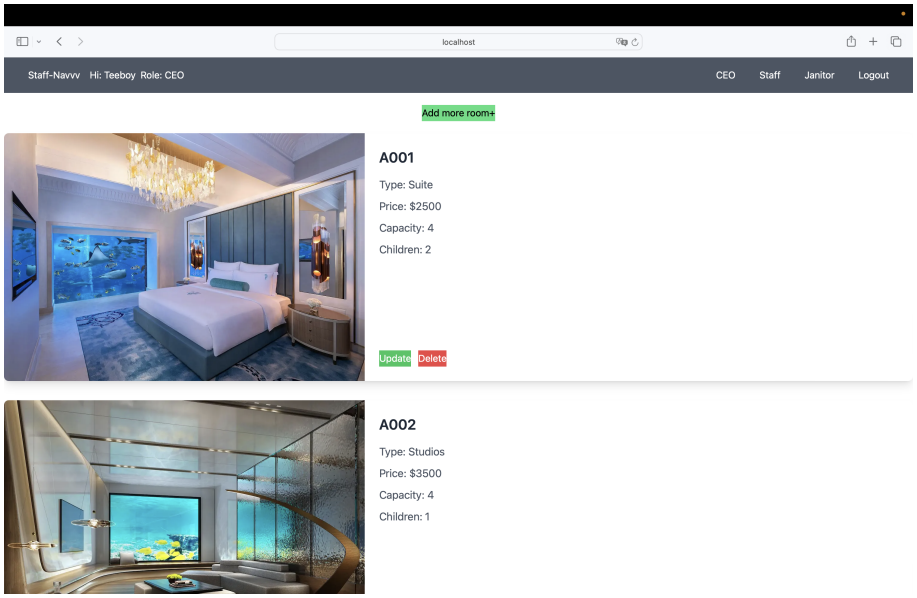

Figure 4.4: Room form

4.2 Advanced Analysis Report

4.2.a Summary Report

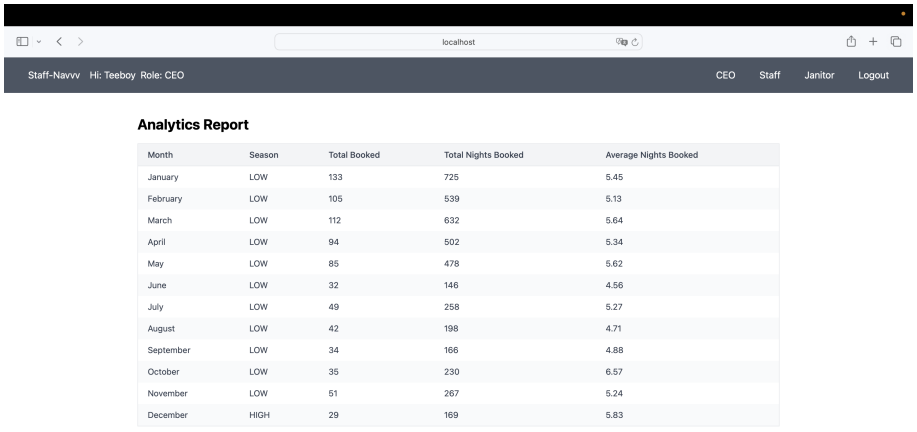

Figure 4.5: Summary report

Relations involved: Invoice, Seasons Discount, Reservations

4.2.b Company Report

Relations involved: Invoice, Seasons Discount

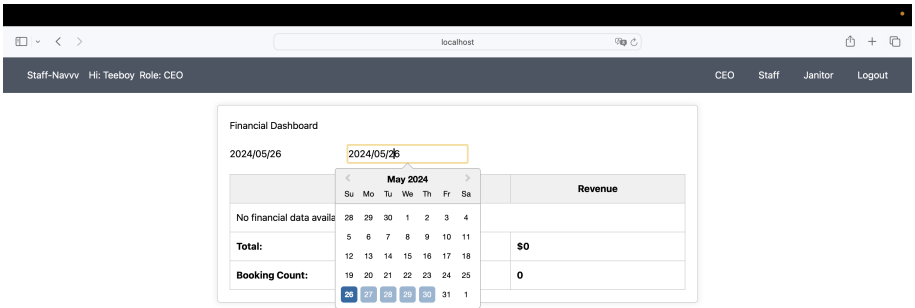

Figure 4.6: Company report

Viewing invoice against seasons discount allows us to see the correlation between revenue when the discount is provided.

CHAPTER 5

APP IMPLEMENTATION

5.1 Infrastructure Setup

Using common three tier architecture by isolating the presentation tier, logic tier and the data tier, we allow development teams to create flexible and reusable applications. The three-tier setup can also be modified to n-tier or multi-tier architecture to accommodate other parts of client-server application such as CDN, media-server and others. [7] On top of that, the modular development process has helped us integrate applications with multiple code bases, exploiting the best features from each of its programming ecosystem. Here, we are benefited with comfort that back-end, front-end and database can be replaced in real-time as long as our business logic holds integrity.

5.1.a Data tier

We improved our database's ER model progressing through the different stages of project.

Development Version Prioritizing development speed and resilience to changes, we opt for MongoDB to ideate and test on our relations. This streamlined speed on the product side.

Prodution Version As our ER model mature over certain development period, we opt to move for PostgreSQL relational database.

5.1.b Logic tier

Both production and development version uses RESTful services with the standard HTTP specifications for authentication and data transfer.

Development Version We applied Node.JS for rapid prototyping.

Prodution Version As our API are stable, we moved to Actix (based on rust programming language). We do not need extra runtime now and a single lightweight binary is much more performant than the development version.

5.1.c Presentation tier

For design of the elements, both production and development version utilizes TailwindCSS and custom components created on top of it. For prototyping User Interface (and basic UX), Figma came into use in our team.

Development Version We applied Next.JS as our team was familiar with it.

Prodution Version We moved to using faster and lighter JS library (SvelteKit) with WebSocket for uninterrupted data changes. Now, our application is updated accross the client nodes in real time without bothering any load to the database or back-end side.

CHAPTER 6

FINDINGS AND CONCLUSION

6.1 Evaluation of Outcomes

We have split our outcome evaluation to access both technical requirements of this database project and then learning outcomes of the CPE241 course.

6.1.a Technical Outcomes

Database Design This is tested from E2E testing in order to find if the database design is actually functional and perform the desired functionally completely. Later, we normalized the relational structure followed by another end to end testing.

Adequate permission levels Proper RBAC metadata is created and imported prior to any data operations via our software applications.

Forms and Reports We have created transaction forms such that at least 3 tables are modified on one action. However, simple actions also exists within the application. Aggregate functions were used for calculating average bookings per month and net profit.

Application development A complete three tier web application is developed to work hand in hand with the designed database. The unit testing was done thoroughly to check if the desired functions are handled properly.

6.1.b Learning Outcomes: Self Assessment

Here is the summary to the very least:

1. ER model was designed and improved progressively over time as our team's domain knowledge improved. It is prepared to best of our abilities and knowledge.
2. Data dictionary and user permission level is included in the report.
3. We developed a functional web application to interact with the database.

6.2 Conclusion

We have explored and a hotel management system applying the best known software development practices using modern database concept.

REFERENCES

[1] W.P.S.W. Weerasinghe, K.D.M.I. Jayathilaka, W.V.C. Prasadi, M.D.K.M. Goonetilleke, Dilshan De Silva, and Piyumika Samarasekara. Research on hotel management system. *International Journal of Engineering and Management Research*, 12:218–226, 10 2022. doi: 10.31033/ijemr.12.5.27.

[2] G2. Best hotel management software, 2024. URL `https://www.g2.com/categories/hotel-management`. [Accessed 26-05-2024].

[3] C.P.U Jayawardena, T.I.S Senaweera, G.S Weeratunga, K.A.P.M Perera, Dilshan De Silva, and S. Vidhanaarachchi. Development of an online hotel reservation system in sri lanka using cutting-edge technologies. *International Journal of Engineering and Management Research*, 12:209–217, 10 2022. doi: 10.31033/ijemr.12.5.26.

[4] R Fulton and R Vandermolen. Chapter 4: Requirements - writing requirements. In *Airborne Electronic Hardware Design Assurance: A Practitioner's Guide to RTCA/DO-254*, chapter 4, pages 89–93. CRC Press, 2017. ISBN 9781351831420.

[5] Sikha Bagui and Richard Walsh Earp. *Database Design Using Entity-Relationship Diagrams*. Auerbach Publications, 2022. ISBN 978-1-032-01718-1.

[6] Peter Chen. Entity-relationship modeling: Historical events, future trends, and lessons learned. In Manfred Broy and Ernst Denert, editors, *Software Pioneers*, pages 296–310. Springer Berlin Heidelberg. ISBN 978-3-642-63970-8 978-3-642-59412-0. doi: 10.1007/978-3-642-59412-0_17. URL `https://link.springer.com/10.1007/978-3-642-59412-0_17`.

[7] Mark Richards. *Fundamentals of Software Architecture: An Engineering Approach*. O'Reilly Media, 1st edition, 2020. ISBN 978-1492043454.

APPENDIX A

(Brainstorming notes - project briefing)

GitHub: https://github.com/TeeThanakit/mariana-hotel

Project name: Hotel Management System

Potential features: Hotel Management, Reservation, Check-in/Checkout, Charges and Invoicing, House Cleaning, Room Rate Code, Seasons, Setup

OUR SOFTWARE STACKS

Data Layer Postgres (database) (use diesel, sqlx, or SeaORM), Redis or MongoDB for dev. period

Logic Layer: Actix-web on Rust (production), Node.JS on Express (development version)

Presentation Layer: Tailwind CSS (add some custom components that work with SvelteKit), Svelte (with Typescript and SvelteKit) for interactivity

This is just initial one.

DEVELOPMENT INSTRUCTIONS

1. Choose a topic (already selected above).
2. Clearly define the detail and scope of the project.
3. Define major user views (user roles) and user permission levels (Note that your project must have at least three separate roles for users).
4. Design your database and represent the design with an ER diagram with primary key, foreign key, and multiplicity. (Note that your project must have at least 7 tables so that each person in your group must have at least 1 complex transaction form with modification in 2 or more tables) and 1 advanced analytic report to (using at least three tables with aggregation functions) in the final project).

   Example of complex transaction form: The form must be able to modify at least 3 tables with one action (one click).

Example of Analysis Report: The report must use at least one aggregate function. (e.g., Count, Sum, Avg., Min., and Max) and be used for decision making.

5. Select appropriate data types for each attribute and generate a data dictionary (showing field names, constraints, and data types) for your database

6. Develop a web/mobile application to connect with the designed database. (Note that each person in your group must have at least 1 complex transaction form (with modification in 2 or more tables) and 1 advanced analytic report to (using at least three tables with aggregate functions) in the final project).

TBD for later changes on plans.

GRADING CRITERIA:

This is the criteria for the final project grading:

1. Database design (30%): The ER model or database model must be designed correctly for the topic selected.

2. Report (20%): The detail and scope of the project, user permission level (Insert, Delete, Update, View) with each table for each user role, and data dictionary must be presented clearly and correctly.

3. Web or Mobile Application (30%): The form and analysis report represented in the developed application must be performed completely and correctly.

4. Presentation (20%): Make your presentation interesting, clear to the point, within a time limit (10 minutes), and correct language usage.