

Fort Hays State University
CSCI 441 Software Engineering
Project: Concierge Software
Final Report

Team D

John Fritter

Panharith Menh

Omar Omer

Heather Young

Github

Web app: <https://github.com/HeatherJoYoung/concierge-web>

API: <https://github.com/HeatherJoYoung/concierge-api>

0.0 Summary of Changes	5
1.0 Customer Problem Statement	7
1.a Problem Statement	7
1.b Glossary of Terms	11
2.0 Goals, Requirements, and Analysis	12
2.a Business Goals	12
2.b Enumerated Functional Requirements	12
2.c Enumerated Non-Functional Requirements	13
2.d User Interface Requirements	14
3.0 Use Cases	16
3.a Stakeholders	17
3.b Actors and Goals	18
3.c Use Cases	19
3.d System Sequence Diagram	26
4.0 User Interface Specification	30
4.a Preliminary Design	30
4.b User Effort Estimation	35
5.0 System Architecture	36
5.a Identifying Subsystems	36
5.b Architecture Styles	37
5.c Mapping Subsystems to Hardware	38
5.d Connectors and Network Protocols	38
5.e Global Control Flow	38
5.f Hardware Requirements	39
6.0 Analysis and Domain Modeling	39
6.a Conceptual Model	39
i. Concept Definitions	39
ii. Association Definitions	40
iii. Attribute Definitions	41
iv. Traceability Matrix	43
6.b System Operation Contracts	43
6.c Data Model and Persistent Data Storage	46
7.0 Interaction Diagrams	47
7.a Logon	47
7.b Customer Reservations	48
7.c Customer Request Events	48
7.d Administrator Handle Managers	49

7.e Administrator Request Historical Reservations	50
7.f Manager Handle Spa Services	51
7.g Manager Handle Events	52
7.h Manager Handle Reservations	53
8.0 Class Diagram and Interface Specification	54
9.0 Algorithms and Data Structures	58
10.0 User Interface Design and Implementation	58
11.0 Design of Tests	58
Project Management and Plan of Work	64
a. Merging the Contributions from Individual Team Members	64
b. Project Coordination and Progress Report	64
c. Plan of Work	64
d. Breakdown of Responsibilities	66

Credits	
1. Customer Problem Statement	John Fritter
2. Goals, Requirements and Analysis 2.a Business Goals 2.d User Interface Requirements	Panharith Menh
2.b Functional Requirements 2.c Non-Functional Requirements	Heather Young
3. Use Cases a. Stakeholders b. Actors and Goals	Omar Omer
c. Use Cases i. Casual Description ii. Use Case Diagram iii. Traceability Matrix iv. Detailed Description	John Fritter
d. System Sequence Diagrams i. Login ii. Reservations iii. Database iv. Admin Dashboard	Panharith Menh
4. User Interface Specification a. Preliminary Design b. User Effort Estimation	Heather Young
5. System Architecture	Heather Young
6.a Conceptual Model	John Fritter
6.b System Operation Contracts	Panharith Menh
6.c Data Model and Data Storage	Omar Omer
7.0 Interaction Diagrams	Heather Young
8.0 Class Diagram and Interface	John Fritter
9.0 Algorithms and Data Structures	Heather Young

10.0 User Interface Design and Implementation	Heather Young
11.0 Design of Tests	Heather Young
Plan of Work	Omar Omer
References	Heather Young

0.0 Summary of Changes

1. The login page design has been updated
2. The register page design has been updated
3. The table for add/removing services on the admin dashboard has been removed
4. Any reference to functionality having to do with the administrator adding or deleting overall services has been removed from the use cases and diagrams. Since adding a new overall service would require coding updates to the web application and API as well as changes to the database, we realized it is too complex to be handled by a click of a button and some inputs from the UI.

1.0 Customer Problem Statement

1.a Problem Statement

In recent years, the tourism industry has witnessed a significant surge in the demand for all-inclusive resorts. These resorts offer a unique vacation experience where guests can enjoy a wide range of amenities, activities, and services, all included in a single package. The primary appeal of all-inclusive resorts lies in their ability to provide travelers with a hassle-free and comprehensive vacation, allowing them to relax and indulge without constantly worrying about costs or planning intricate details. For many vacationers, convenience is the number one factor in their decision to book a vacation at such a resort. Therefore resort hotels are incentivized to maximize this aspect of their experience in order to appeal to customers.

Sarah's Story

Meet Sarah, a typical vacationer in today's world of travel. She's a busy professional who craves a break from her hectic routine. Amidst her busy schedule, Sarah longed for simplicity and convenience. Planning a vacation shouldn't be a chore, and the idea of having everything neatly bundled in one package appealed to her. The thought of booking her accommodation, meals, drinks, and activities all in one go was like a breath of fresh air. This would save Sarah the time and hassle of seeking out different offerings from different companies, adding costs together to ensure she could stay within her budget, arranging transportation, and checking on the reputation of different companies to ensure she would get a consistent quality of service. By choosing an all-inclusive resort, Sarah's primary goal was to relax and have a stress-free and

memorable vacation filled with a variety of unique experiences. However, as she embarked on her journey, she encountered some challenges that our Concierge Software seeks to resolve.

Once Sarah arrived at the all-inclusive resort, she was greeted with the breathtaking beauty of the surroundings and the promise of a luxurious getaway. However, as she checked in and settled into her room, she couldn't help but notice the absence of something she had come to expect in her everyday life: seamless, user-friendly technology.

While the resort had a traditional concierge desk, it wasn't as accessible as she had hoped. There were often lines of guests waiting to ask questions or seek recommendations, and Sarah had to navigate this queue just to get information about activities or dining options. As someone who valued her vacation time, this felt like an unnecessary waste of precious moments.

The concierge staff, while friendly, had limited availability. They worked set hours, and there were times when Sarah wanted to make dinner reservations, inquire about spa treatments, or sign up for activities outside of these hours. Unfortunately, her requests had to wait until the concierge desk reopened.

The resort's paper-based activity schedules were somewhat outdated, and it was easy for Sarah to miss out on exciting events. She found herself flipping through leaflets and bulletin boards, trying to decipher the schedule, and sometimes missing activities she would have loved to attend.

When it came to dining, Sarah wanted to explore the various restaurants the resort had to offer. However, she often found herself in long lines, especially during peak dining hours, as the process of checking table availability and making reservations was done manually.

In the evenings, Sarah enjoyed live entertainment shows, but she had to rely on word of mouth to discover what was happening and where. It was a hit-or-miss experience, and sometimes she missed out on spectacular performances simply because she hadn't been informed in advance.

All these aspects of her resort experience left room for improvement. Sarah's vacation was precious, and she wanted to make the most of every moment. The existing system relied heavily on manual processes, which resulted in inefficiencies and missed opportunities. The limited accessibility of the concierge desk and the absence of real-time updates made her feel like she was constantly playing catch-up, which was far from the stress-free experience she had envisioned.

When it comes time for Sarah and other guests to choose their next vacation, they will remember this experience, and the existence of our concierge software could easily be the deciding factor in their decision of where to go.

Ethan's Story

Now let's introduce Ethan, the resort employee. Ethan had always been passionate about working in the hospitality industry, and he cherished his role as a concierge at the all-inclusive resort. His primary goal was to ensure that every guest experienced the vacation of their dreams, filled with unforgettable memories. However, he couldn't help but notice that the traditional methods he had to rely on were sometimes a hurdle to providing top-notch service.

As a concierge, Ethan was at the frontline of guest interactions. He often found himself juggling multiple requests and inquiries simultaneously, from booking spa treatments and arranging romantic dinners to providing information about off-site excursions. The resort's

manual reservation system required him to switch between different booking sheets, which could be time-consuming and occasionally resulted in overbooked or double-booked services.

While Ethan was skilled at multitasking, he sometimes wished for a more efficient way to manage guest requests. The traditional system meant that he needed to be physically present at the concierge desk during his working hours. He was aware that many guests preferred to make plans or reservations during the evening hours when they had more leisure time, but Ethan wasn't available then. This disconnect often led to missed opportunities to enhance the guest experience.

Additionally, Ethan's resort had a wide array of services, activities, and dining options, each with its own schedule and availability. Keeping track of all this information and providing guests with accurate and up-to-date details was a demanding task. Ethan frequently had to spend time manually updating bulletin boards, leaflets, and information stands throughout the resort, making sure guests were informed about the latest offerings.

One of the most challenging aspects of Ethan's job was generating reports for resort management. The process involved compiling data from various sources, such as reservation logs and guest feedback forms, and manually creating reports that outlined the resort's service usage and guest preferences. It was a time-consuming endeavor that took him away from his primary responsibility—guest satisfaction.

Despite his dedication, Ethan realized that there was room for improvement in the way he provided assistance to guests and contributed to the resort's overall success. He yearned for a tool that would streamline his tasks, allowing him to be more responsive to guest needs, provide real-time updates, and generate insightful reports effortlessly.

1.b Glossary of Terms

Administrator - An administrator is someone who has top-level access to the application. They have access to the admin dashboard, where they can generate reports and add/remove manager privileges.

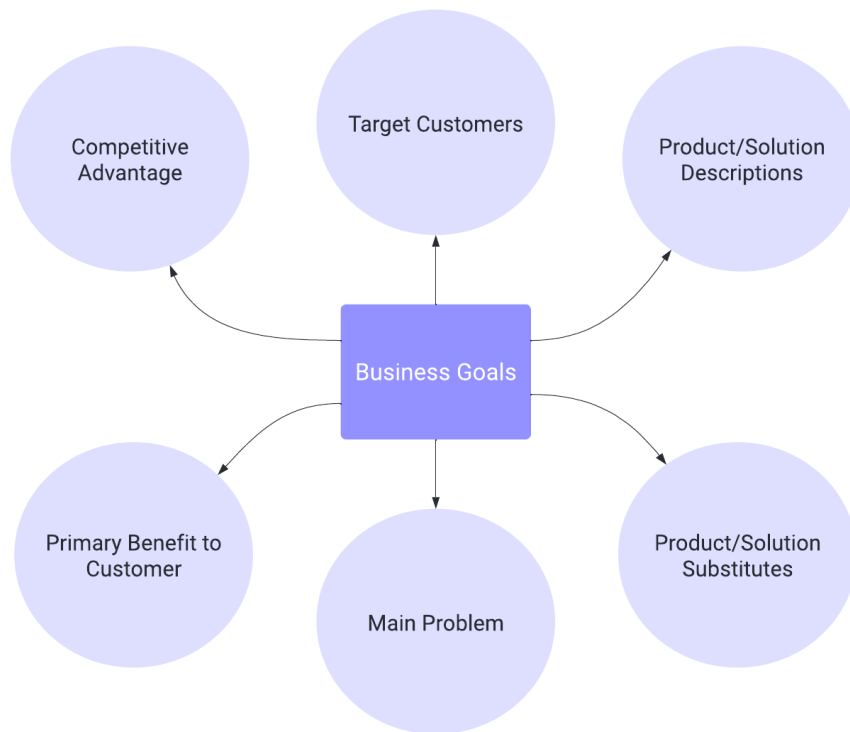
Guest - A guest is someone who has booked a vacation at the resort. They have access to the calendar view and the service-based view which they may use to see available services and sign up for them.

Manager - A manager is someone who is responsible for a particular service, such as a restaurant or spa. They have access to the manager dashboard, where they can update the calendar of events for their service.

Service - A service is a particular area or venue within the parent organization (resort or cruise) that offers activities which a customer can sign up for.

2.0 Goals, Requirements, and Analysis

2.a Business Goals



2.b Enumerated Functional Requirements

Identifier	Priority	Requirement
REQ-1	5	Application must have secure login for customers
REQ-2	5	Application must have secure login for administrators and managers
REQ-3	4	Guest should be able to choose calendar view page
REQ-4	4	Guest should be able to choose service-based view

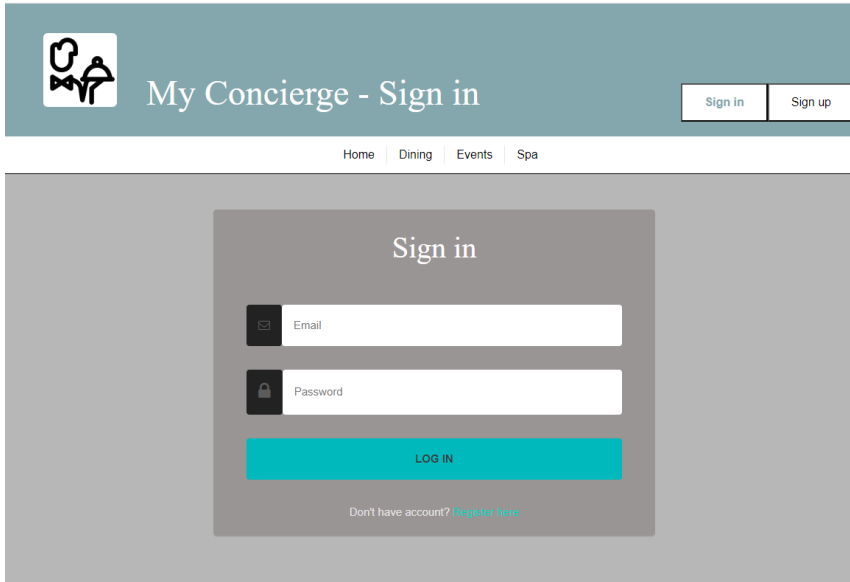
REQ-5	4	Guest should be able to navigate to service-specific home page from calendar view and service-based view
REQ-6	3	Service-specific home pages should include dates and times for upcoming events
REQ-7	5	Service-specific home pages should include a way to sign up for an activity or make a reservation
REQ-8	1	Guests should receive confirmation email and/or text when they sign up for an activity or make a reservation
REQ-9	1	Guests should receive reminder email and/or text prior to an activity or reservation they have signed up for
REQ-10	5	Administrators should be able to access the admin dashboard
REQ-11	5	Administrators should be able to give and remove managers' privileges
REQ-13	5	Managers should have access to the manager dashboard
REQ-14	5	Managers should be able to update the calendar of events for their service, including capacity and assigned staff
REQ-15	3	Administrators and managers should be able to generate usage reports for certain periods of time (a month, a year, etc)

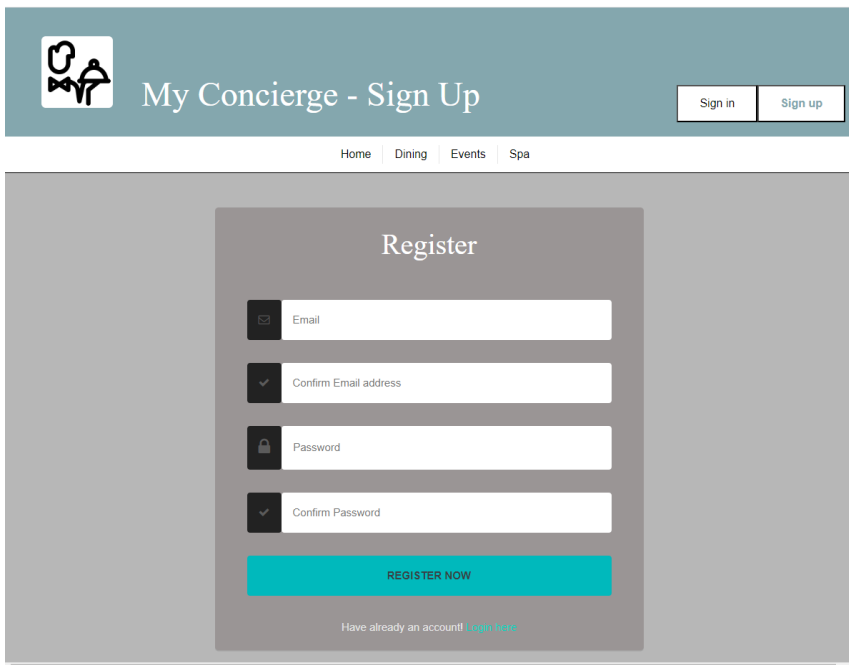
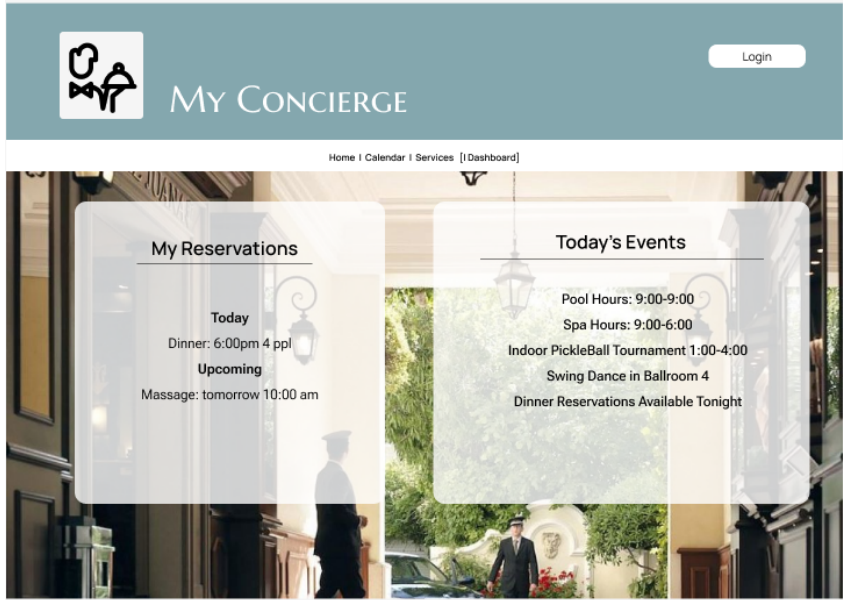
2.c Enumerated Non-Functional Requirements

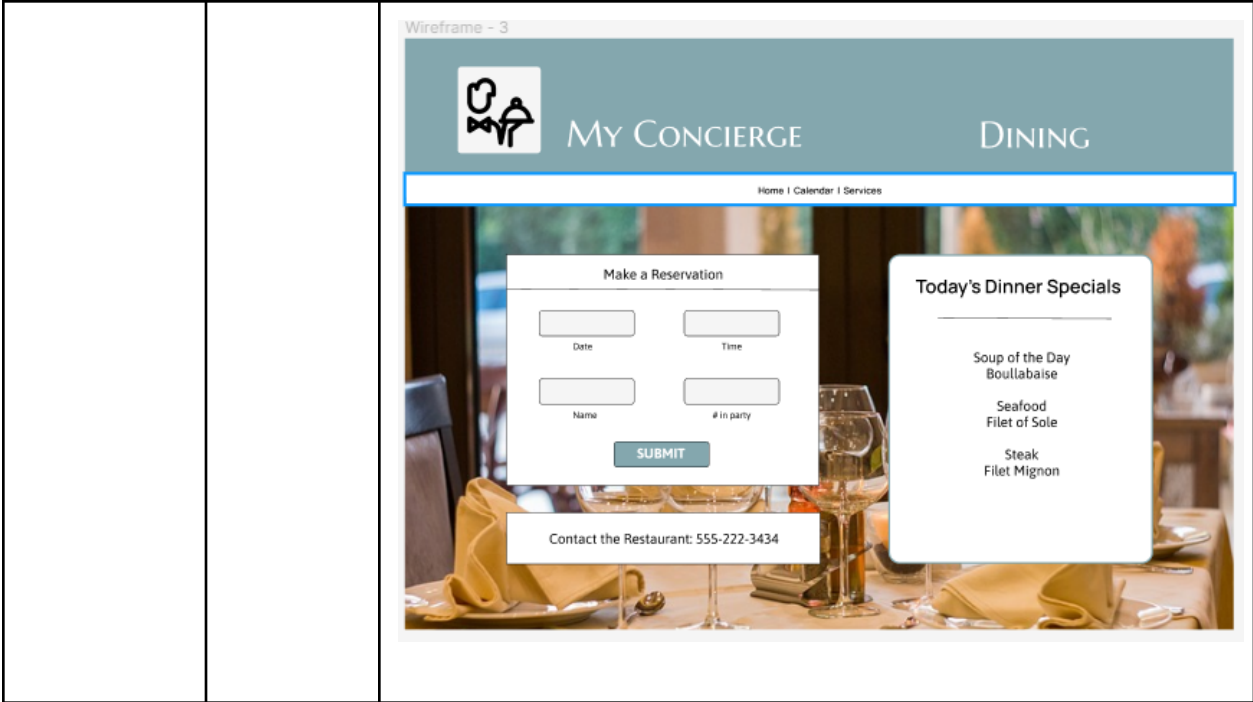
Identifier	Priority	Requirement
REQ-16	5	Application must run in the major browsers and on web and mobile platforms
REQ-17	4	Application should be modular and scalable
REQ-18	4	Application must be intuitive and easy to use
REQ-19	3	The privileged admin/manager views should be protected from inadvertent or malicious access

REQ-20	2	Site access privileges for customers should correlate to reservation dates.
REQ-21	2	Proper cyber security protocols should be followed to reduce risks of malicious access to the system in general
REQ-22	2	Application should be resilient against points of failure at any particular component

2.d User Interface Requirements

Identifier	Priority	Requirement
REQ-23	10	GUI must have a login screen/main screen for the user to enter account information.
REQ-24	10	GUI must have the “Forgot password” option in case users forget their infomation.
REQ-25	10	GUI must also have a register option in case users are new to the application or website.
REQ-26	7	<p style="text-align: center;">GUI Login Screen</p> 
REQ-27	7	GUI Sign up Screen

		
<p>REQ-28</p>	<p>8</p>	<p>GUI home/main screen</p> 
<p>REQ-29</p>	<p>8</p>	<p>GUI for scheduling/reservation</p>



3.a Stakeholders

In the dynamic landscape of the modern tourism industry, the demand for all-inclusive resorts has surged significantly. These resorts promise travelers a seamless and hassle-free vacation experience, catering to the desire for convenience and relaxation. To meet these expectations, the Concierge Software project aims to bridge the gap between the guests' expectations and the resort's services, offering a comprehensive solution for planning, booking, and enjoying all that the resort has to offer.

Let's explore the various stakeholders involved in the Concierge Software project, each with their distinct roles and interests:

- ❖ **Guests (End Users) :** As the primary users, guests seek a user-friendly Concierge Software that simplifies vacation planning, providing easy access to resort information and seamless booking, enhancing their overall vacation experience.
- ❖ **Resort Management :** Resort managers prioritize guest satisfaction and operational efficiency. They rely on the software for decision-making, reporting, and optimizing resource allocation to streamline resort operations.
- ❖ **Concierge Staff :** Front-line concierge staff, require the software to reduce manual tasks, provide real-time updates on reservations and activities, and enhance guest engagement, ensuring exceptional service delivery.

- ❖ **Administrators** : Administrators manage the software's configuration and maintenance. Their responsibilities include generating reports, managing services, and assigning tasks.

3.b Actors and Goals

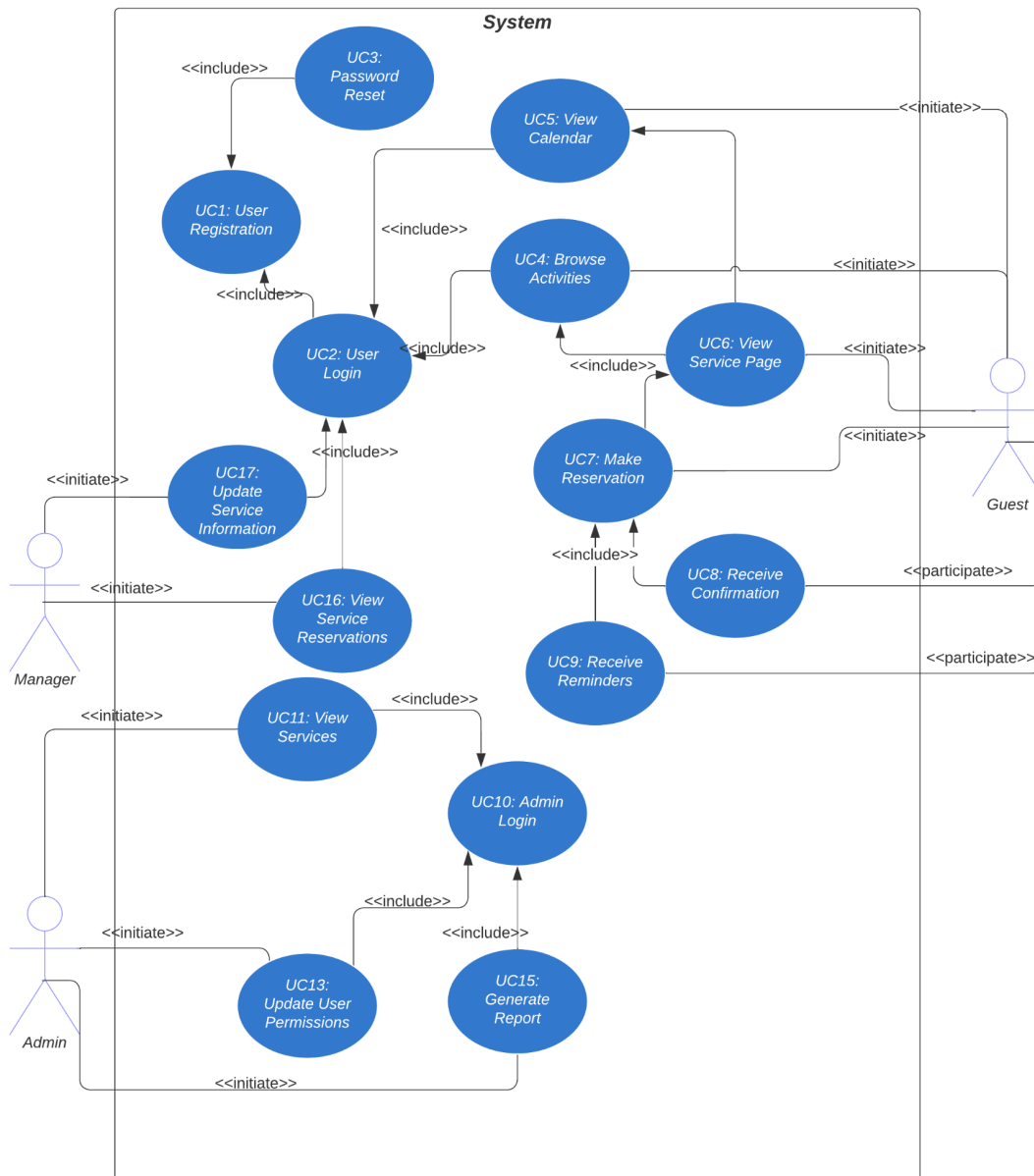
Actor	Goal	Use Case
Users	<ul style="list-style-type: none"> → Register an account → Login to user account → Reset password → Browse services → View a calendar of activities → Make reservations for activities, dining, or services → Receive reminders for scheduled activities 	UC1 UC2 UC3 UC4 UC5 UC6 UC7 UC8 UC9
Managers	<ul style="list-style-type: none"> → Login securely → Browse available activities and services → Update service or activity information → Assign staff 	UC10 UC16 UC17
Administrators	<ul style="list-style-type: none"> → Log in securely → Update user permissions and roles → Generate usage reports for specific periods 	UC10 UC12 UC13 UC14 UC15
Database	<ul style="list-style-type: none"> → Store users information → Store activities or services → Store reservations for users → Store staff information → Store administrators information 	UC1 UC17 UC12 UC13 UC14

3.c Use Cases

i. Casual Description

- UC1: User Registration
 - Description: Allows users to register an account in the system.
 - Responds to Requirements: REQ-1, REQ-2, REQ-25, REQ-27
- UC2: User Login
 - Description: Enables users to securely log in to their accounts.
 - Responds to Requirements: REQ-1, REQ-2, REQ-23, REQ-26
- UC3: Password Reset
 - Description: Allows users to reset a forgotten password.
 - Responds to Requirements: REQ-1, REQ-2, REQ-24
- UC4: Get Information on Services and Activities
 - Description: Permits guests to gain information about services and activities available at the resort.
 - Responds to Requirements: REQ-3, REQ-4, REQ-5, REQ-6, REQ-28, REQ-29
- UC7: Make Reservation
 - Description: Enables guests to reserve activities, dining, or services.
 - Responds to Requirements: REQ-7, REQ-8, REQ-9
- UC10: Admin Login and Dashboard
 - Description: Enables concierge staff to securely log in to their accounts and see the Admin dashboard.
 - Responds to Requirements: REQ-2, REQ-10, REQ-23
- UC12: Admin Management Functions
 - Description: Allows administrators to change user permissions and roles
 - Responds to Requirements: REQ-11
- UC14: Generate Reports
 - Description: Enables administrators to generate usage reports for specific periods.
 - Responds to Requirements: REQ-15
- UC16: Manage Service
 - Description: Allows managers to manage reservations for their specific service and to update information about their service.
 - Responds to Requirements: REQ-13, REQ-14

ii. Use Case Diagram



iii. Traceability Matrix

Requirement	PW	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11	UC12	UC13	UC14	UC15	UC16	UC17
REQ-1	5	X	X	X														
REQ-2	5	X	X	X							X							
REQ-3	4					X												
REQ-4	4				X													
REQ-5	4						X											
REQ-6	3						X											
REQ-7	5							X										
REQ-8	1								X									
REQ-9	1									X								
REQ-10	5										X							
REQ-11	5												X					
REQ-13	5																X	
REQ-14	5																	X
REQ-15	3															X		
REQ-16	5																	
REQ-17	4																	
REQ-18	4																	
REQ-19	3																	
REQ-20	2																	
REQ-21	2																	
REQ-22	2																	
REQ-23	5		X								X							
REQ-24	5			X														
REQ-25	5	X																
REQ-26	2		X															
REQ-27	2	X																
REQ-28	3				X													
REQ-29	3					X												
Max PW		5	5	5	4	4	4	5	1	1	5	5	3	5	3	3	5	5
Total PW		17	17	15	7	7	7	5	1	1	10	5	3	5	3	3	5	5

iv. Fully-Dressed Description

UC1: User Registration
<p>Preconditions:</p> <ul style="list-style-type: none">● The guest is not yet registered in the system.● The guest is using a web or mobile platform with internet access.● The guest is on the login screen. <p>Postconditions:</p> <ul style="list-style-type: none">● The guest is successfully registered in the system and logged in.
<p>Flow of Events:</p> <ul style="list-style-type: none">● The guest opens the application or website.● The guest clicks on the "Register" or "Sign Up" button.● The system displays the User Registration form, prompting the guest to provide the following information:<ul style="list-style-type: none">○ First name○ Last name○ Email address○ Password○ Confirm password○ Additional optional information (e.g., contact number, date of birth)● The guest fills in the required fields.● The guest clicks the "Submit" or "Register" button.● The system validates the provided information:<ul style="list-style-type: none">● Ensures that the email address is unique (not already registered).● Checks that the password meets security requirements (e.g., minimum length, complexity).● If validation fails, the system displays appropriate error messages and prompts the guest to correct the information.● If validation succeeds, the system creates a new user account for the guest and logs them in automatically.● The system displays a welcome message to the guest and redirects them to the home/main screen. <p>Alternative Flows:</p> <ul style="list-style-type: none">● If, during registration, the guest enters an email address that is already registered in the system, the system will display an error message asking the guest to use a different email address. <p>Exceptional Flows:</p> <ul style="list-style-type: none">● If there are technical issues or network problems during the registration process, the system will display an error message and ask the guest to try again later.

UC7: Make a Reservation**Preconditions:**

- The guest is registered and logged in the system.
- The guest is using a web or mobile platform with internet access.
- The guest is on the home screen.

Postconditions:

- The guest has successfully made a reservation for one of the services.

Flow of Events:

- From the home page, the guest navigates to the Services page.
- The guest clicks on a service and navigates to the service home page.
- On the service home page is a reservation form, prompting the guest to provide the following information:
 - Date
 - Time
 - Guest Name
 - Service-specific info (e.g., # in party for Dining, Service for Spa, and Event for Special Events)
- The guest fills in the required fields.
- The guest clicks the "Submit" button.
- The system validates the provided information:
- If validation fails, the system displays appropriate error messages and prompts the guest to correct the information.
- If validation succeeds, the system creates a new reservation record and saves it to the database.
- The system displays a success message.

Alternative Flows:

- The guest can also navigate from the main home page to the calendar page.
- On the calendar, they can click any activity link, which will take them to the appropriate service home page. From there the flow is the same as above.

Exceptional Flows:

- If there are technical issues or network problems during the reservation process, the system will display an error message and ask the guest to try again later.

UC13: Update User Permissions

Preconditions:

- The administrator is using a web or mobile platform with internet access.
- The administrator is on the home page.

Postconditions:

- The administrator has successfully made changes to user permissions.

Flow of Events:

- The administrator navigates to the Admin Dashboard.
- In the Managers table, the administrator clicks on the plus icon to add a new manager.
- In the empty row, the administrator can enter the manager information:
 - Name
 - Department
- The administrator clicks on the plus icon to add the manager..
- The system validates the provided information.
- If validation fails, the system displays appropriate error messages and prompts the guest to correct the information.
- If validation succeeds, the system adds manager permissions for the employee.

Alternative Flows:

- To remove manager permissions, the administrator clicks on the red X in the row to the right of the manager whose permissions are being removed.
- The rest of the flow is the same as above.

Exceptional Flows:

- If there are technical issues or network problems during the process of saving the new information to the database, the system will display an error message.

UC14: Generate Reports

Preconditions:

- The administrator is using a web or mobile platform with internet access.
- The administrator is on the home page.

Postconditions:

- The administrator has successfully downloaded a report.

Flow of Events:

- The administrator navigates to the Admin Dashboard.
- The administrator clicks on the Download Report button beneath the graph.
- The displayed data is downloaded as a csv file.

Alternative Flows:

- None.

Exceptional Flows:

- If there are technical issues or network problems during the process of downloading the report, the system will display an error message.

3.d System Sequence Diagram

i. Use Case - Login/Register

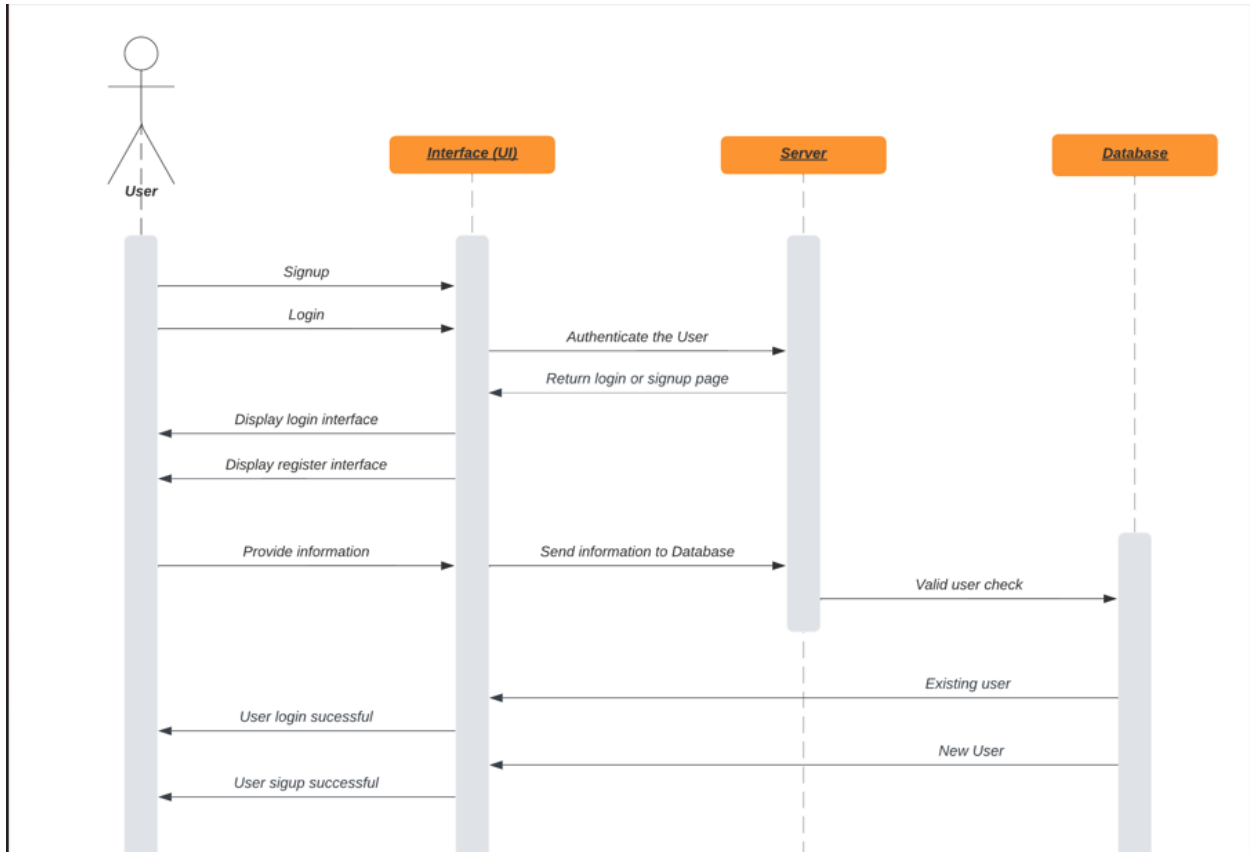


Fig1: The above diagram is for Use Case User Login/Register diagram. The users sign up their account into the system, and the accounts are validated in the server. If the credentials are valid, the data will be stored in the database. With the existing accounts, the server checks for the database and confirms or verifies the account.

ii. Use Case - Booking/Reservation

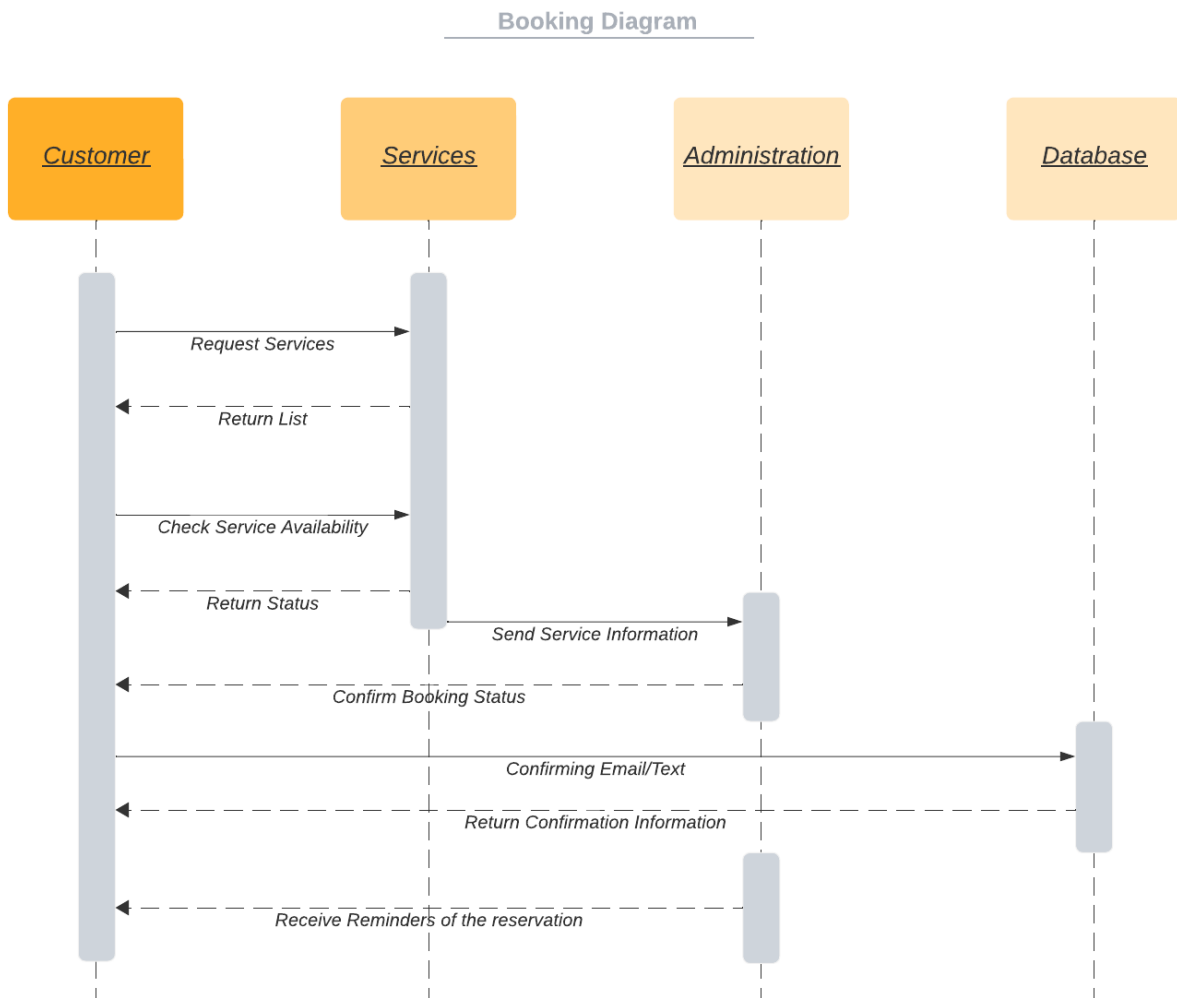


Fig 2: The above diagram is Booking/Reservation. Users/Customers can make a reservation for the hotel. They can look up the available room and activities in the hotel. They could then send their transaction information to the hotel to confirm the reservation. Finally, they would receive a receipt stating they have booked the reservation.

iii. Use Case - Database/Storage

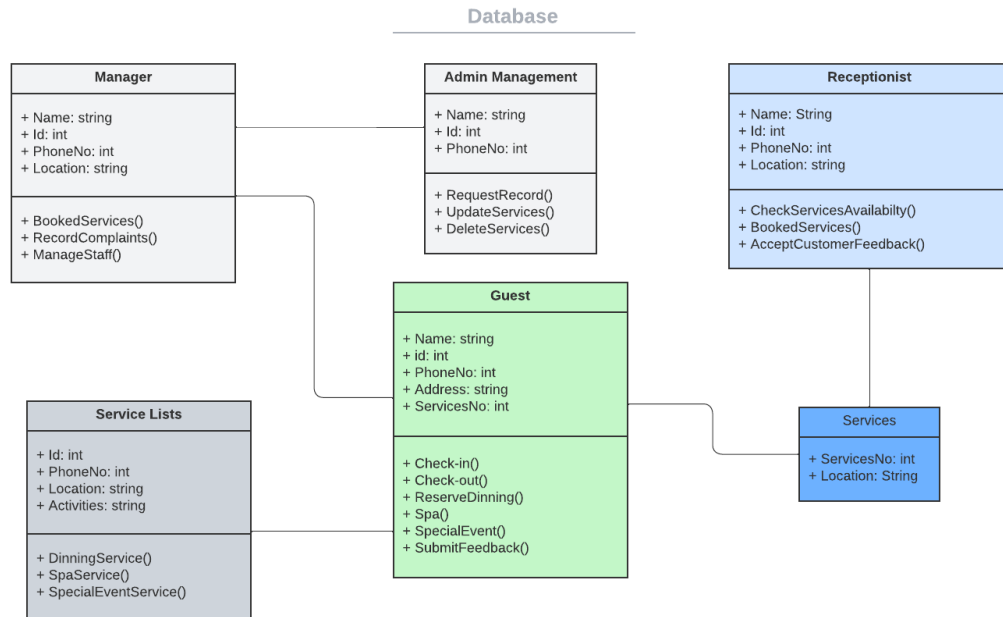


Fig-3: The above diagram shows a data/storage map. We store all the required data within the Guest. Each attribute would connect to other boxes with its decimated values such as Rooms, Bill, Food Items, and Manager. Time processing is also different depending on value boxes. The Transaction process would take time as it needs to confirm the process between both parties, Customers and Admin.

iv. Use Case - Admin Dashboard

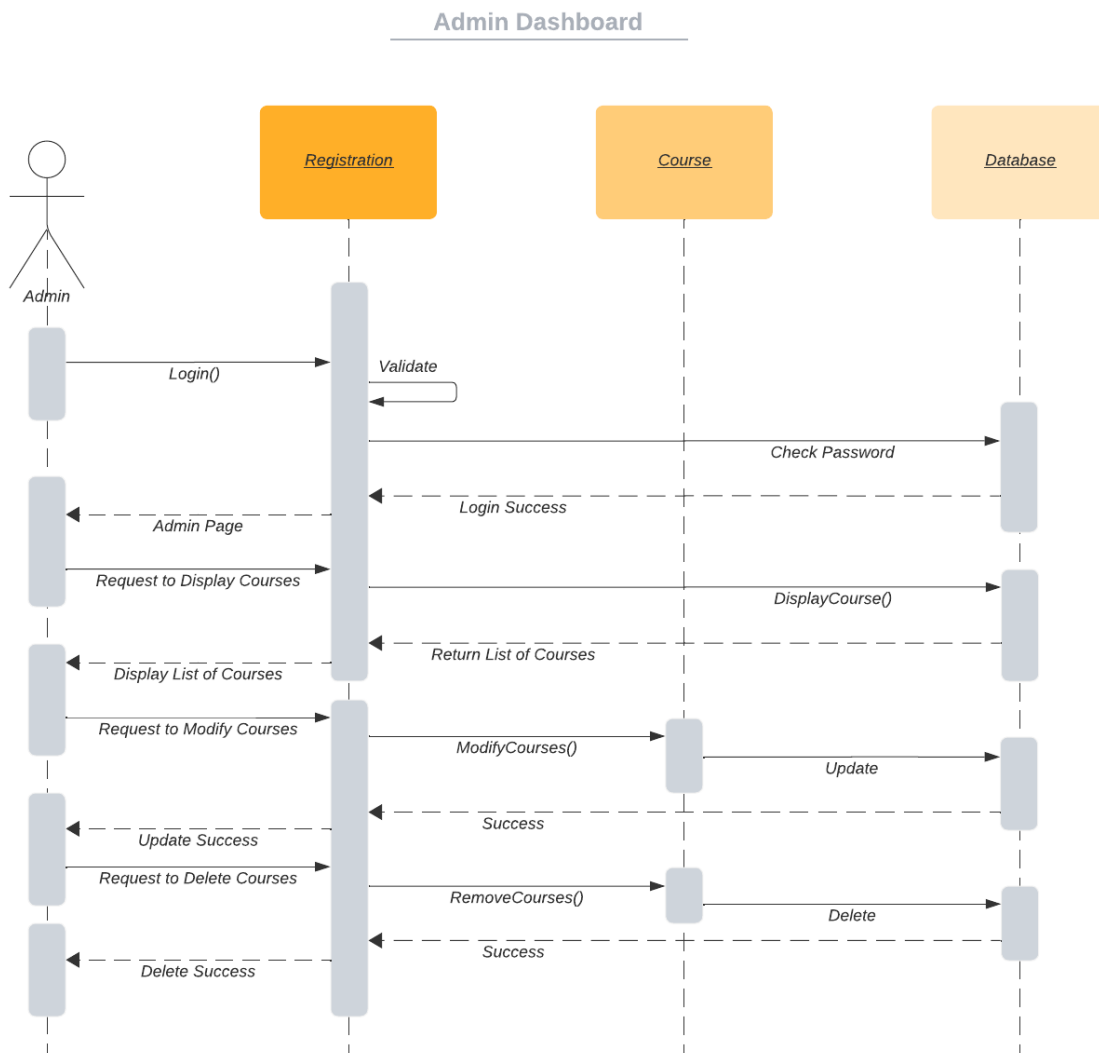


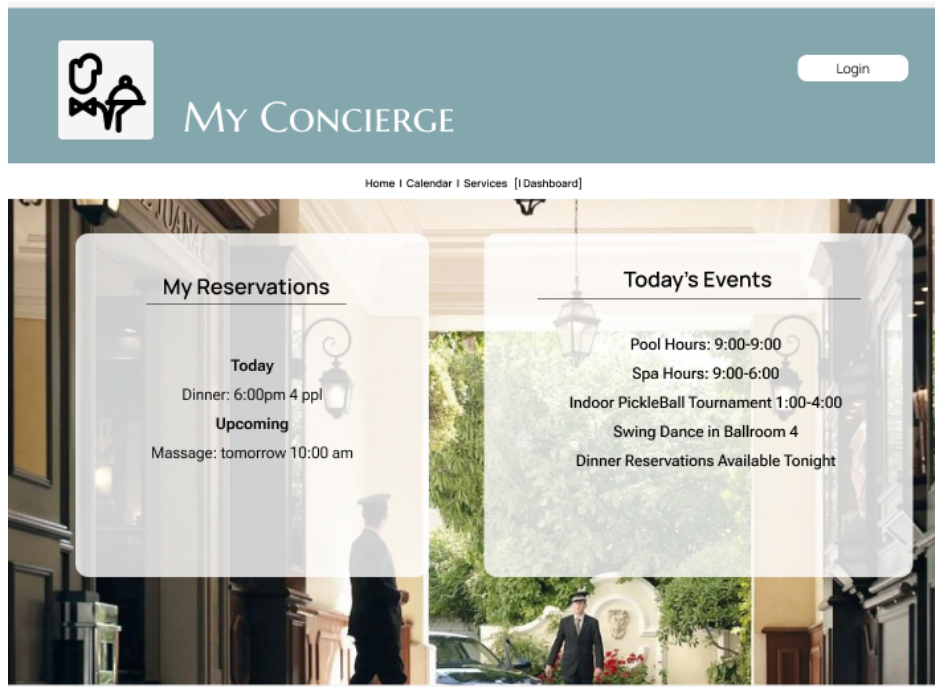
Fig-5: The above diagram shows the admin commands/dashboard process. Admin must sign in for confirmation. They can view the current services and managers list. They would then need to request if they want to modify/update either of these.

4.0 User Interface Specification

4.a Preliminary Design


Guest Interfaces

Home page



Calendar View

Wireframe - 6




MY CONCIERGE

Home | Calendar | Services

January 2024						
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
	1	2	3 Double Tennis 2:00 - 4:00	4 Pineapple Golf Pro 1:00 - 2:00	5	6
7	8 Salsa Dancing 6:00 - 8:00	9	10	11	12	13 La Scala Opera 7:00 - 12:00
14	15 Salsa Dancing 6:00 - 8:00	16	17 Double Tennis 2:00 - 4:00	18	19 Homeparty Talk 3:00 - 4:00	20
21	22 Salsa Dancing 6:00 - 8:00	23	24	25	26	27 Local Night Club Pop 8:00 - 12:00
28	29 Salsa Dancing 6:00 - 8:00	30	31			

Services View


Wireframe - 2




MY CONCIERGE

Home | Calendar | Services


Services



Dining



Spa

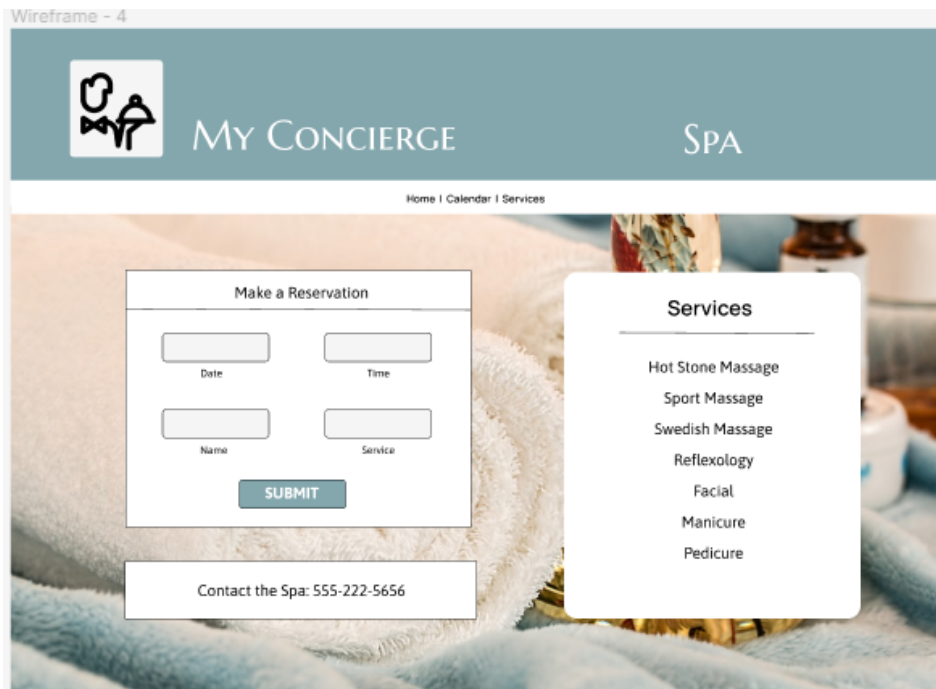


Special Events

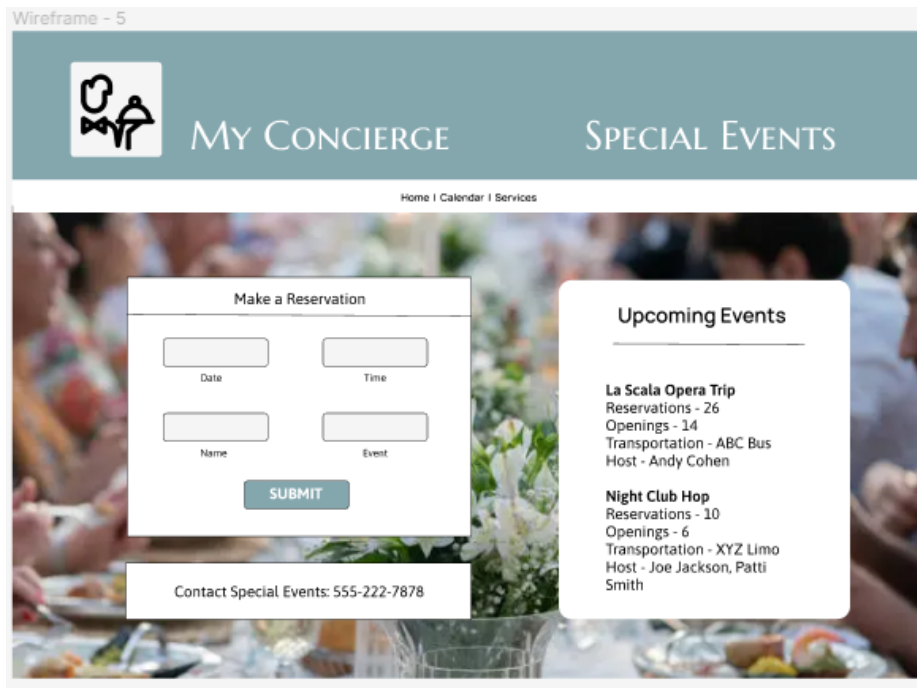
Dining Home



Spa Home

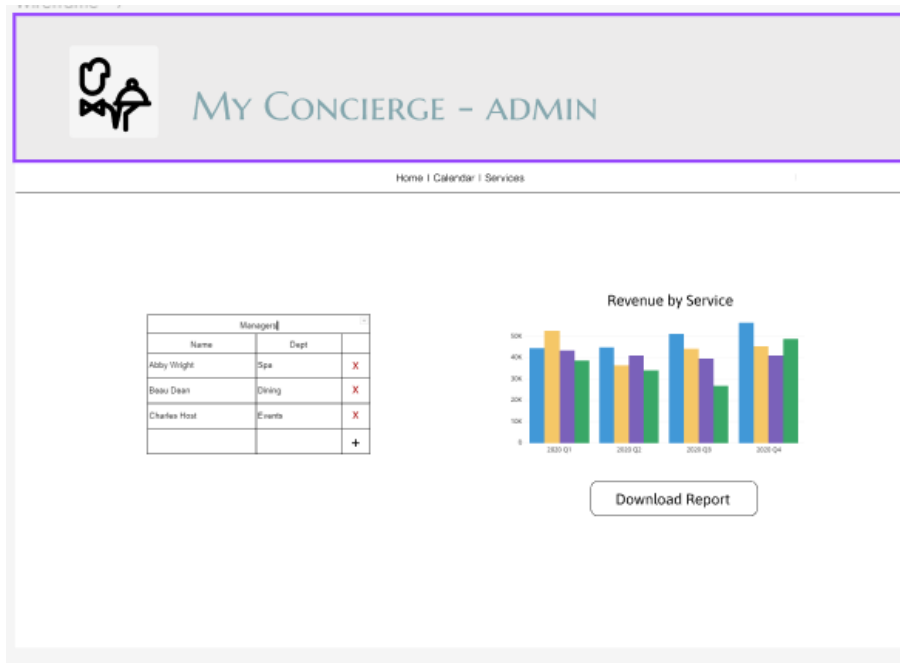


Special Events Home




Management Interfaces

Administrator's Dashboard



Dining Management

Wireframe - 8



MY CONCIERGE - ADMIN

Home | Calendar | Services | Dining Home

Dining Management

Reservations					
Date	Time	Client	Party #	Update	Delete
1/2/23	6:00	Jane Doe	4	Update	Delete
	6:30	John Smith	2	Update	Delete
	7:00	Mary Jackson	4	Update	Delete
1/3/23	7:00	Don Johnson	10	Update	Delete
	6:00	Jane Barnaby	2	Update	Delete
	7:00	Joe Jackson	5	Update	Delete
	7:30	Tom Barnaby	12	Update	Delete
	8:00	Kelly Greene	2	Update	Delete

Key Stats


Today's Dinner Reservations
100

Availability
40

Update Menu

Spa Management

Wireframe - 9



MY CONCIERGE - ADMIN

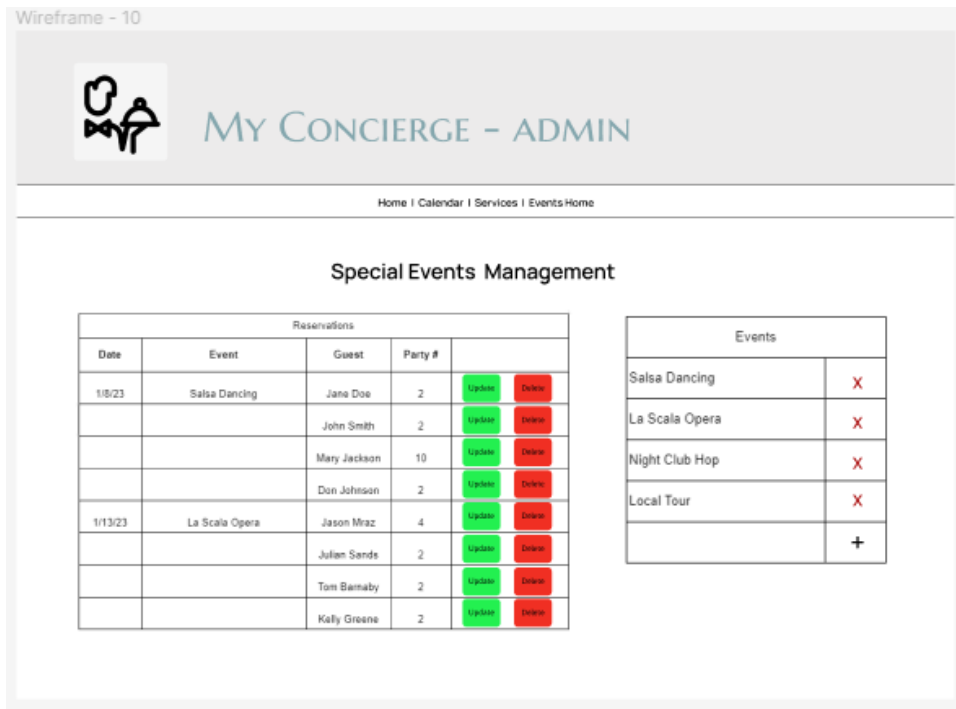
Home | Calendar | Services | Spa Home

Spa Management

Reservations						
Date	Time	Duration (mins)	Service	Client	Staff Member	
1/2/23	11:00	60	swedish massage	Jane Doe	Nancy Drew	Update Delete
	11:00	90	hot stone massage	John Smith	Sally Good	Update Delete
	1:00	30	pedicure	Mary Jackson	Shari Samson	Update Delete
1/3/23	2:30	60	sport massage	Don Johnson	Tom Jones	Update Delete
	3:00	30	reflexology	Jane Barnaby	Drew Chapman	Update Delete
	3:30	60	hot stone massage	Jane Barnaby	Nancy Drew	Update Delete
	3:00	90	sport massage	Tom Barnaby	Sally Good	Update Delete
	4:00	60	swedish massage	Kelly Greene	Tom Jones	Update Delete

Services	
Hot Stone Massage	X
Sport Massage	X
Swedish Massage	X
Reflexology	X
Facial	X
Manicure	X
Pedicure	X
	+

Special Events Management



4.b User Effort Estimation

Usage Scenario	Clicks	Keystrokes
Guest Interface		
Register	5	16-40
Login	3	16-40
Home page	1	0
Check Reservations	0	0
Calendar View	1	0
Services View	1	0
Dining Home	2	0
Spa Home	2	0
Events Home	2	0

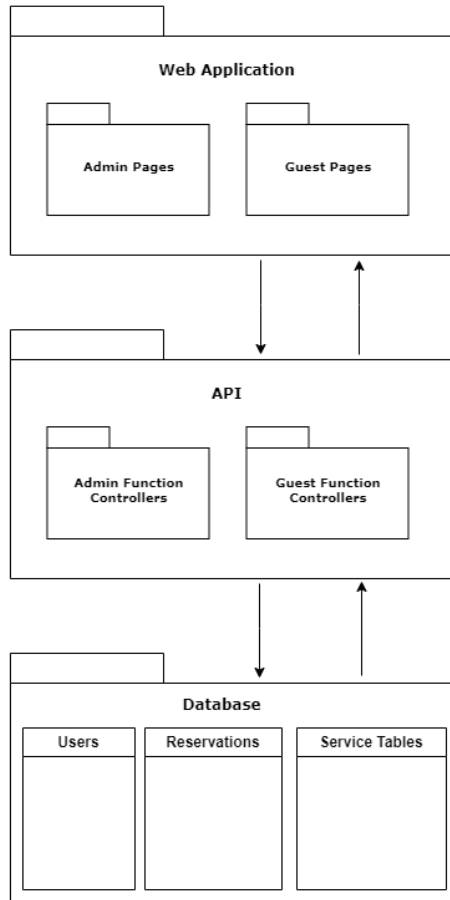
Make Reservation	5	20-50
Contact Modal	3	0
Management Interface		
Login	3	16-40
Admin Dashboard	1	0
Update Managers	2	0
Manage Services	2	0
Download Report	2	0
Mgmt Dashboard	1	0
View Reservations	2	0
Delete Reservations	1	0
Add Reservations	2	20-50
Update Service Details	2	10-30

5.0 System Architecture

5.a Identifying Subsystems

Our project includes a web application, API, and database. It can further be divided into subsystems related to administrative views and functions on the one hand and guest views and functions on the other hand. As the diagram below shows, the guests' user interface sends requests to the controllers in the API that relate to the guest-specific functions. The API communicates with the database to retrieve and update information in the appropriate tables. Likewise, the administrative user interface communicates with the controllers in the API that

relate to the administrative functions. The API in turn communicates with the database to make updates and retrieve information from the appropriate tables.



5.b Architecture Styles

Our project is based on the client/server architecture. Our web application contains the user interface design for users, both administrators and guests, and implements the front-end logic that allows users to communicate with the server and ultimately post data to and get data from the database. The web application sends requests to the server which contains the API and

database and receives responses back from the API. The API in turn communicates with the database to create new records, update existing records, and retrieve data.

5.c Mapping Subsystems to Hardware

The server runs on a master computer which contains the database, serves up the web application, serves up the API and handles requests between the client-side web application running in the user's browser and the API. Administrators and managers have desktop computers through which they can pull up the concierge web application in their browsers. Guests can access the web application via a browser on their phones or tablets.

5.d Connectors and Network Protocols

The cornerstone of our system is a REST API that will communicate with the client-side web application running in user's browsers via HTTP POST and GET requests. The front-end application will be built with Vue using a Vite server, which will handle performing the requests for the front end. The API is a Node application utilizing Express for handling HTTP requests. In addition, we are using MS SQL Server for the database. The database and the API are hosted on the same machine and the API will communicate with the database utilizing SQL queries.

5.e Global Control Flow

Our Concierge application is an event-driven system. Users can generate actions in different order and independently of each other. Once registered and logged in, a guest can browse for information and make a reservation for any of the available services. Independent of what actions guests are performing, managers can update reservations for their respective service and manage the service-related information that gets displayed to guests. Likewise, administrators can update managers' permission and manage what services are available.

5.f Hardware Requirements

Our front-end application can be run on any device that has access to the internet and a browser. The initial layout design we are implementing is for desktop, as our customer is the resort administration and managers. At the same time, we intend our UI to be responsive so that it can be viewed on tablets and phones, because the resort guests who use the Concierge will likely be using phones and tablets. The server, the API and the database are built to be hosted on a Windows machine with Windows 10 or higher OS.

6.0 Analysis and Domain Modeling

6.a Conceptual Model

In this section, we present the Conceptual Model for our Concierge Software project. The Conceptual Model forms the foundational framework for our software's architecture, encompassing key concepts, associations, attributes, and a traceability matrix. It provides a clear understanding of the domain within which our software operates, facilitating alignment with user requirements and guiding the software's development.

i. Concept Definitions

Responsibility Description	Type (D - doing; K - knowing)	Concept Name
Contains detailed information about an available activity.	K	Service
Contains detailed information about a specific guest's reservation of a certain service.	K	Reservation
Contains reservation information to be sent to a guest as confirmation or reminder.	K	Notification

Contains detailed account information for any human user of the system.	K	User Account
Collect guest information to facilitate registration, login, and creating reservations.	D	Web App
Collect manager input to update service information.	D	Web App
Enable admins to update service information, edit reservations, and update User Accounts.	D	Web App
Deliver notifications to guests	D	Web App
Provide interface for guests to view service information and user account information.	D	Web App
Provide interface for managers to view reservations.	D	Web App
Provide interface for admins to view all service information, user information, and usage reports.	D	Web App
Store and retrieve user information, service information, and usage history in the database	D	API

ii. Association Definitions

Concept Pair	Association Definition	Association Name
Web App \Leftrightarrow User Account	Create user accounts and verify them for login and permissions	create/verify
Web App \Leftrightarrow Reservation	Create reservations and display them	create/reserve
Web App \Leftrightarrow Notification	Deliver notifications to the user	deliver
Web App \Leftrightarrow Service	Create, update, and display service information	Service query
User Account \Leftrightarrow API	Store and fetch user information from the database	DB query
Service \Leftrightarrow API	Store and fetch service information from	DB query

	the database	
Reservation ⇔ API	Fetch user and service information from the database	DB query
Notification ⇔ API	Fetch user and service information from the database	DB query
Reservation ⇔ Notification	Generate notifications	generate
Web App ⇔ API	Retrieve usage data for reports	report

iii. Attribute Definitions

Concept	Attribute	Description
WebApp	serviceView	A user interface element responsible for displaying service information.
	calendarView	A user interface element that provides guests with a calendar view of available activities.
	loginPage	A user interface element where users can log in to their accounts securely.
	adminDash	A user interface element that serves as the dashboard for administrators, allowing them to manage services and user accounts.
	mngrView	A user interface element that provides managers with a specific view to oversee their respective services.
User Account	userLogin	A unique string used to identify a user.
	userName	The name associated with a

		user's account.
	userRole	The role or permissions assigned to a user, such as guest, manager, or administrator.
	guestUntil	The date until which a guest is given access to the WebApp.
	userEmail	The email address associated with a user's account.
	password	The secure credential used for user authentication.
Service	servName	The name or title of a specific service.
	servSchedule	The dates and times during which a service is available.
	servOwner	The manager responsible for overseeing a particular service.
Reservation	resDate	The date and time for a reservation.
	resOwner	The user who initiated the reservation.
API	dbConnection	The connection information and configuration details required for the API to connect to the database.
Notification	userName	The login of the account associated with a particular notification.
	reminderTime	The time at which a reminder is scheduled to be sent to a guest.

iv. Traceability Matrix

Use Case	PW	Domain Concepts					
		WebApp	API	Service	Reservation	Notification	User Account
UC1	17	X	X				X
UC2	17	X	X				X
UC3	15	X	X				X
UC4	7	X	X	X			
UC5	7	X	X	X			
UC6	7	X	X	X			
UC7	5	X			X		
UC8	1	X				X	
UC9	1	X				X	
UC10	10	X	X				X
UC11	5	X	X	X			
UC13	5	X	X				X
UC14	3	X	X				X
UC15	3	X	X				
UC16	5	X	X	X	X		
UC17	5	X	X	X			

6.b System Operation Contracts

Operation	Registration
Use Case:	UC-1
Pre-Conditions:	<ul style="list-style-type: none"> The user input all the required information needed to create a new account. The system will check whether the given information already existed or not.
Post-Conditions:	<ul style="list-style-type: none"> The account is registered into the system or software. The browser will ask whether to save the account.

Operation	View Services and Activities
Use Case:	UC-4
Pre-Conditions:	<ul style="list-style-type: none"> • The users can either use login account/guest account to view services • The users can choose to view any services and activities such as Dining, Spa, and Special Events.
Post-Conditions:	<ul style="list-style-type: none"> • The users/customers can see all the given services on the website, it can be viewed either by login account/guest account.

Operation	Making Reservation
Use Case:	UC-7
Pre-Conditions:	<ul style="list-style-type: none"> • The users can choose any services and activities before making a reservation. They can add and delete services. • The calendar will pop out for users to pick a specific date when the reservation is finished.
Post-Conditions:	<ul style="list-style-type: none"> • The reservation will be saved for both the admin and users. • The admin will send notification to remind the users of the reservation either by email account, user login account on the website, or any other form of communication.

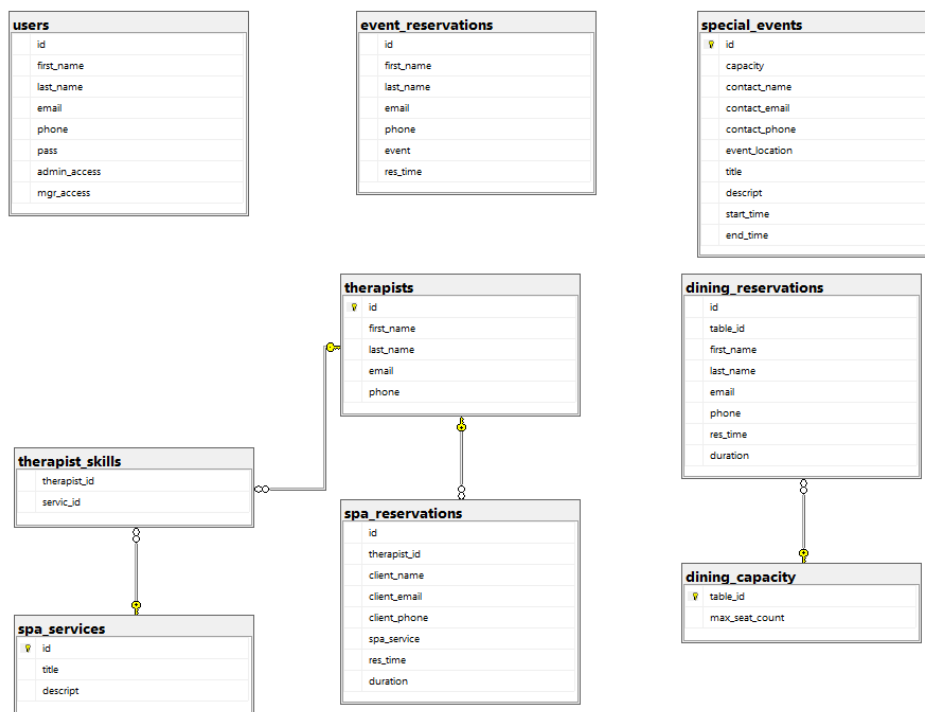
Operation	Admin Check Up
Use Case:	UC-10
Pre-Conditions:	<ul style="list-style-type: none"> • Admins must log in as the employee for check-in, it is to keep track of the activities. • The system will check all the existing info of the admin to ensure security for the website server.
Post-Conditions:	<ul style="list-style-type: none"> • The admin login information is accepted and successfully correct.

	<ul style="list-style-type: none"> • All login time and date will be saved to ensure security for the server.
--	--

Operation	Admin Commands
Use Case:	UC-14
Pre-Conditions:	<ul style="list-style-type: none"> • It will enable the admin to generate reports for specific time periods.
Post-Conditions:	<ul style="list-style-type: none"> • All the activities done by the admin will be saved to ensure security for the server. • The report can be shared to relevant users depending on circumstances.

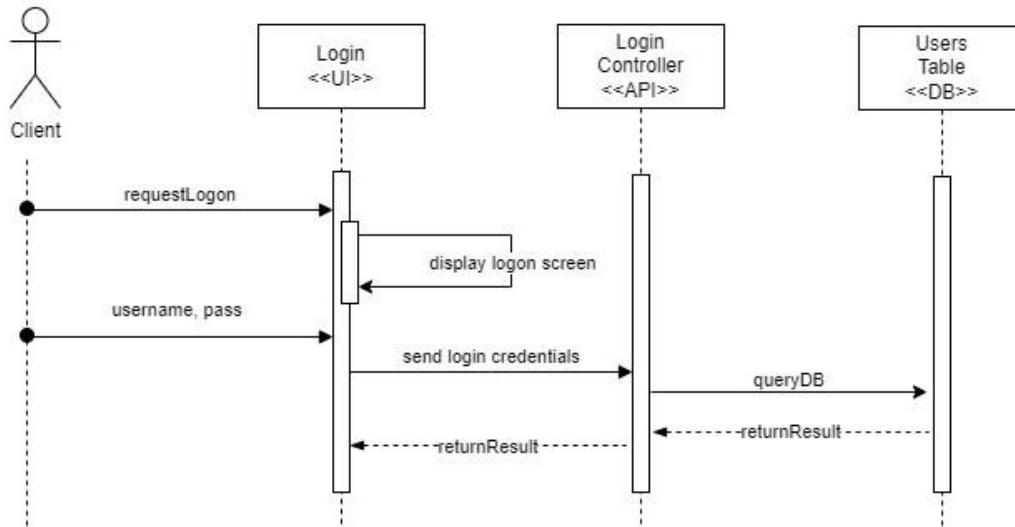
6.c Data Model and Persistent Data Storage

Our concierge software heavily relies on the use of a database management system. The database schema is designed to accurately represent user data, event and service information, and reservations. We use Microsoft SQL Server, a relational database system, to manage this data. SQL queries are used to retrieve data from the database, ensuring that users always access the latest and most accurate information in our software. This approach guarantees the real-time reflection of current services or events that concierge staff might add.

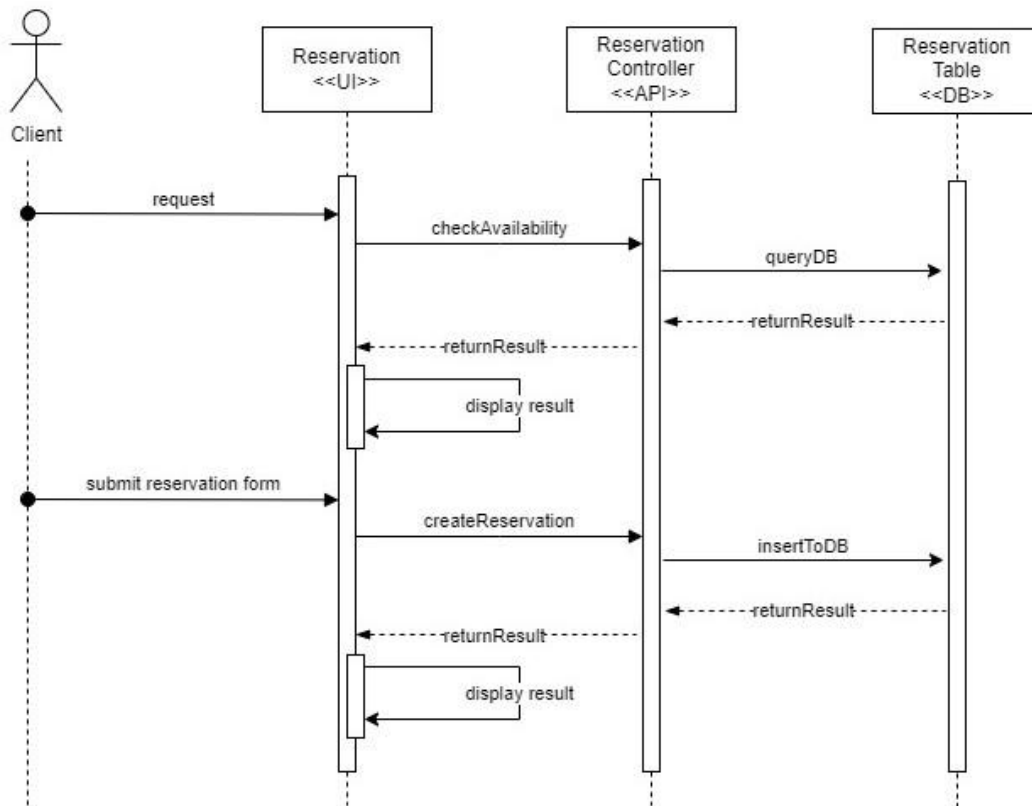


7.0 Interaction Diagrams

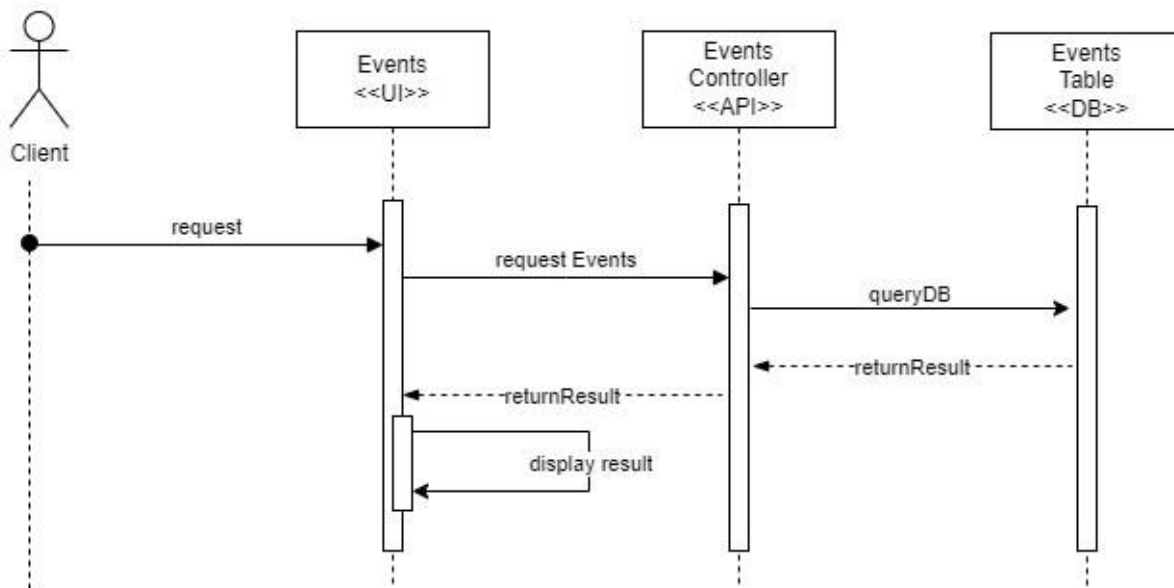
7.a Logon



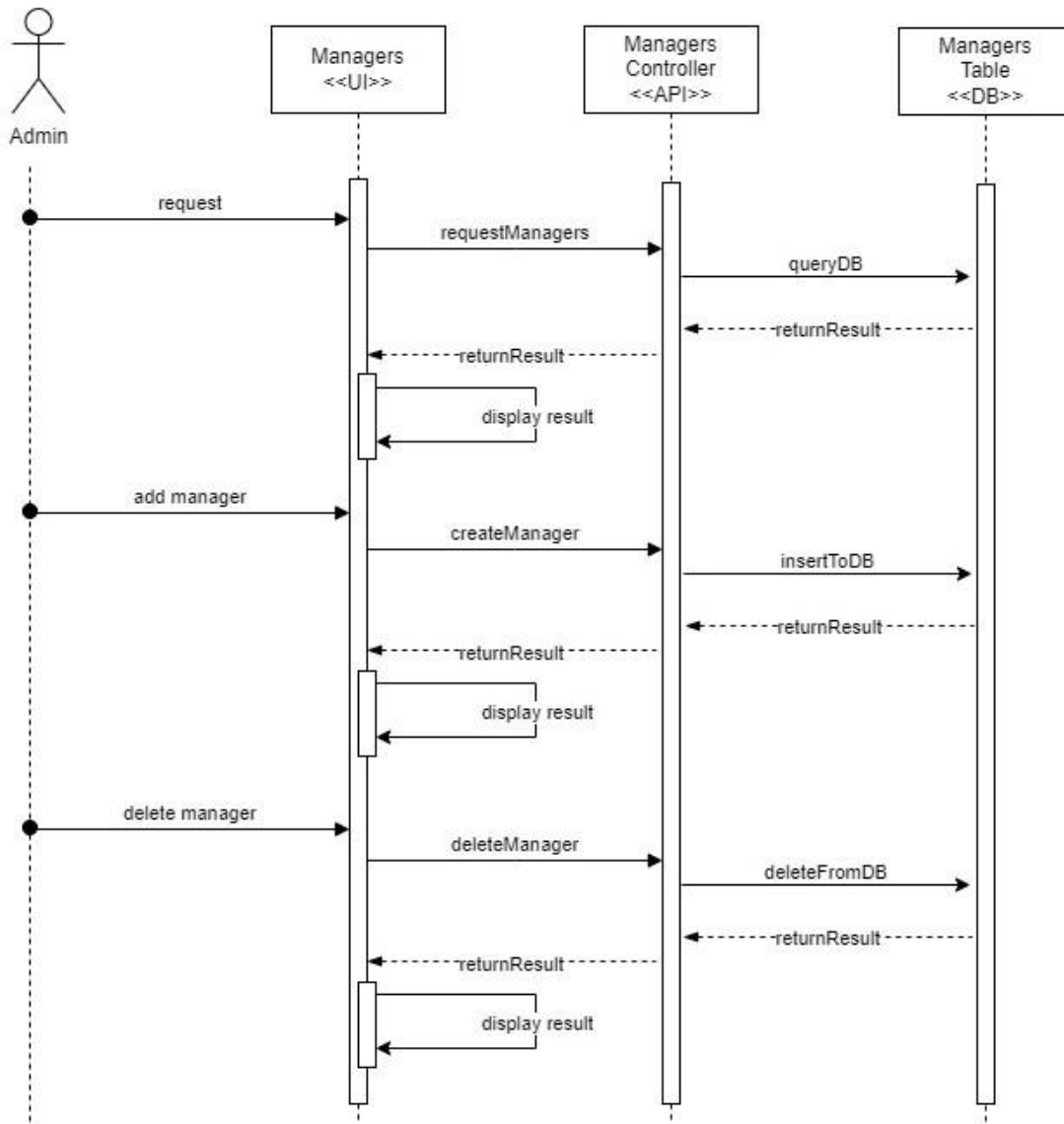
7.b Customer Reservations



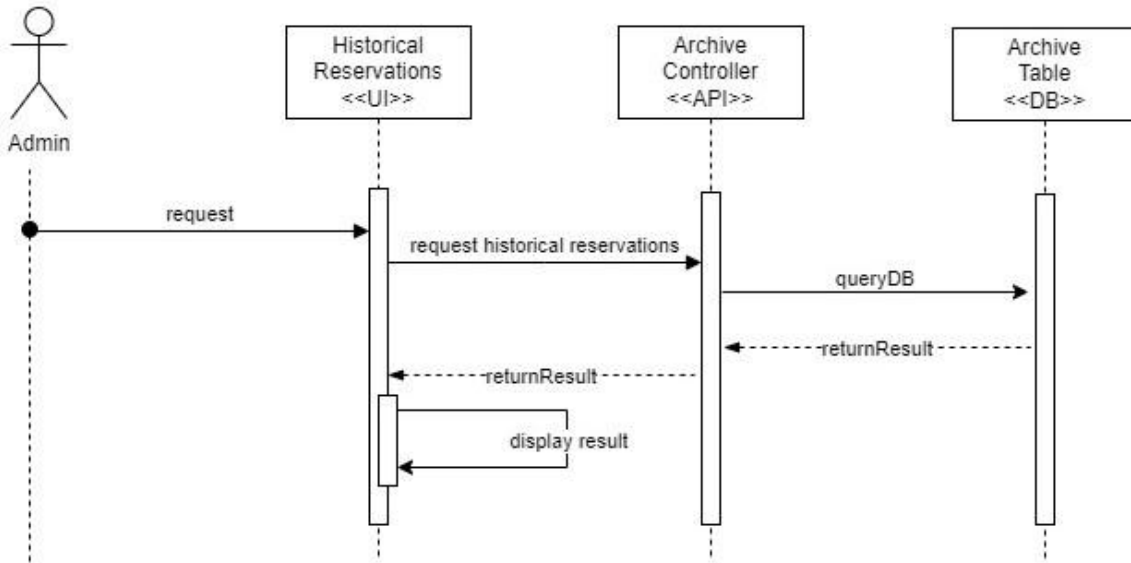
7.c Customer Request Events



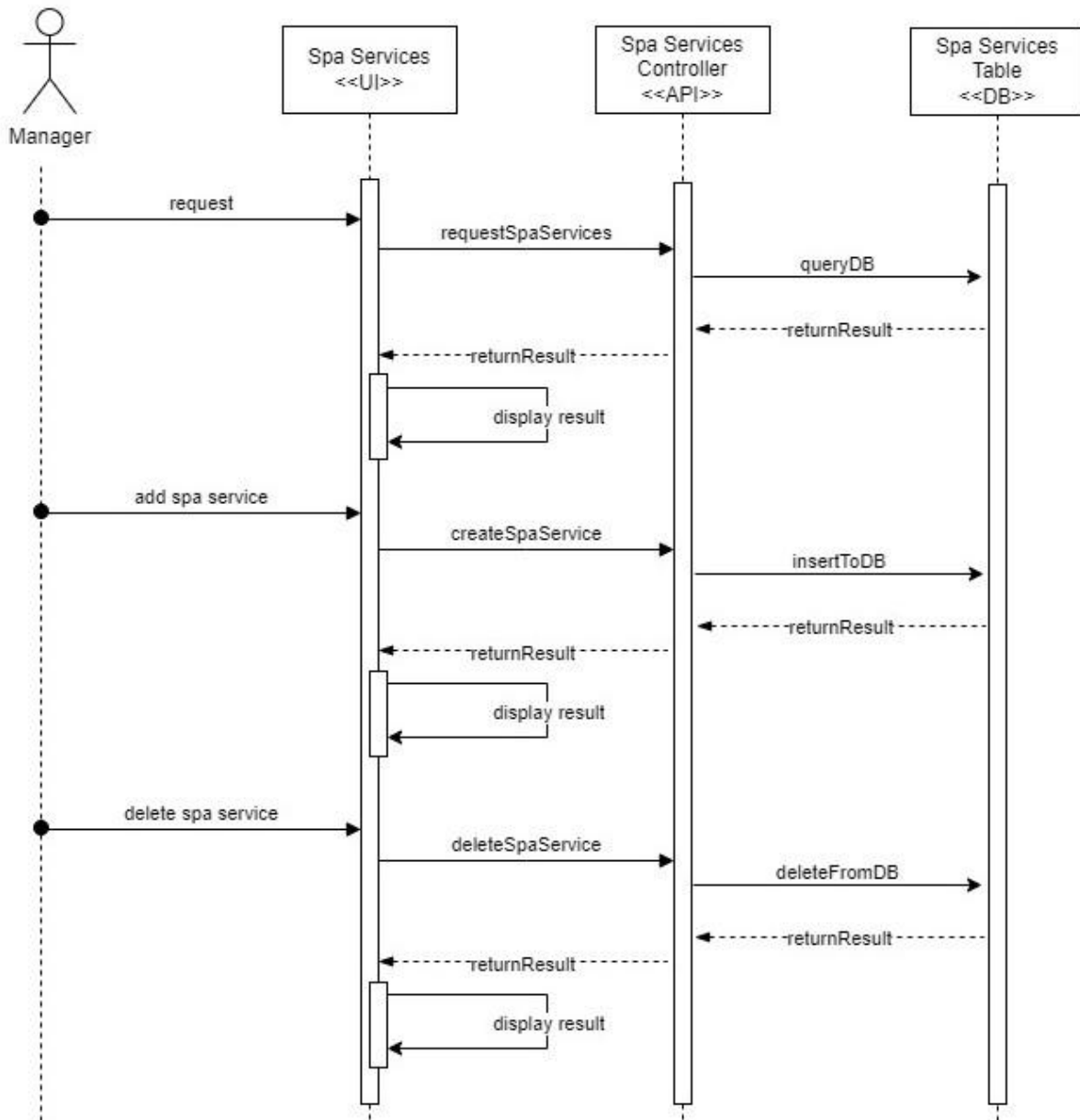
7.d Administrator Handle Managers



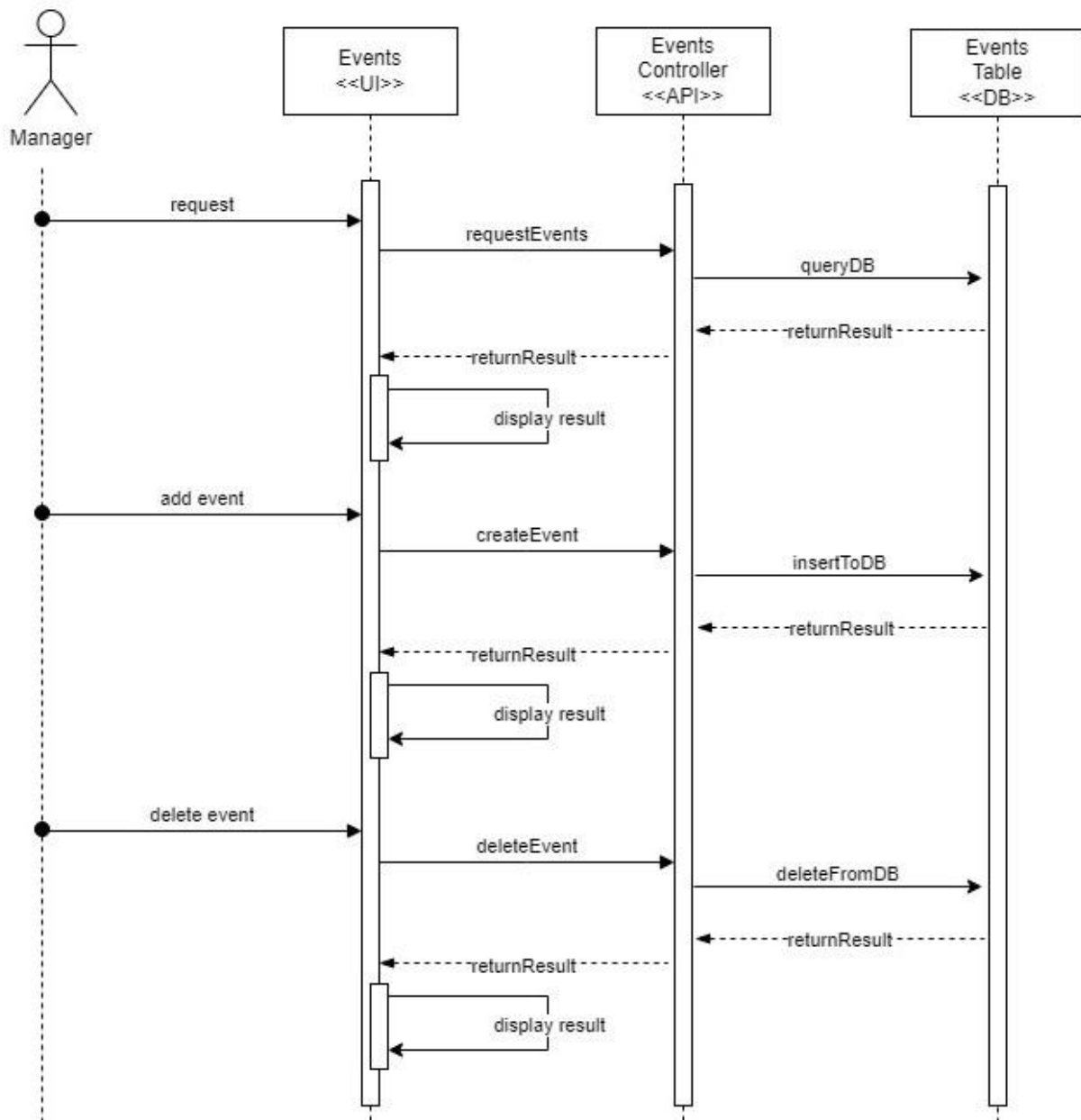
7.e Administrator Request Historical Reservations



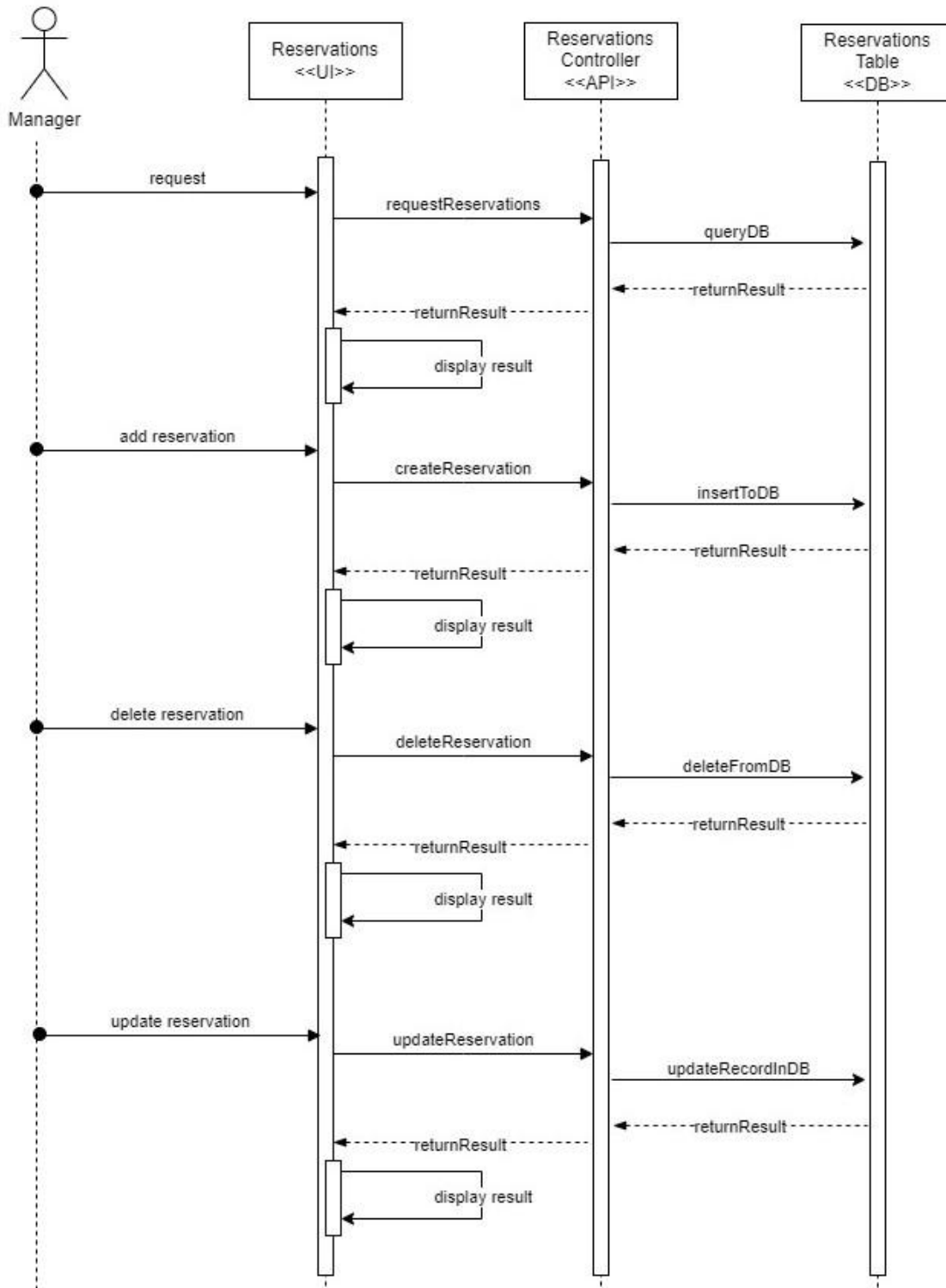
7.f Manager Handle Spa Services



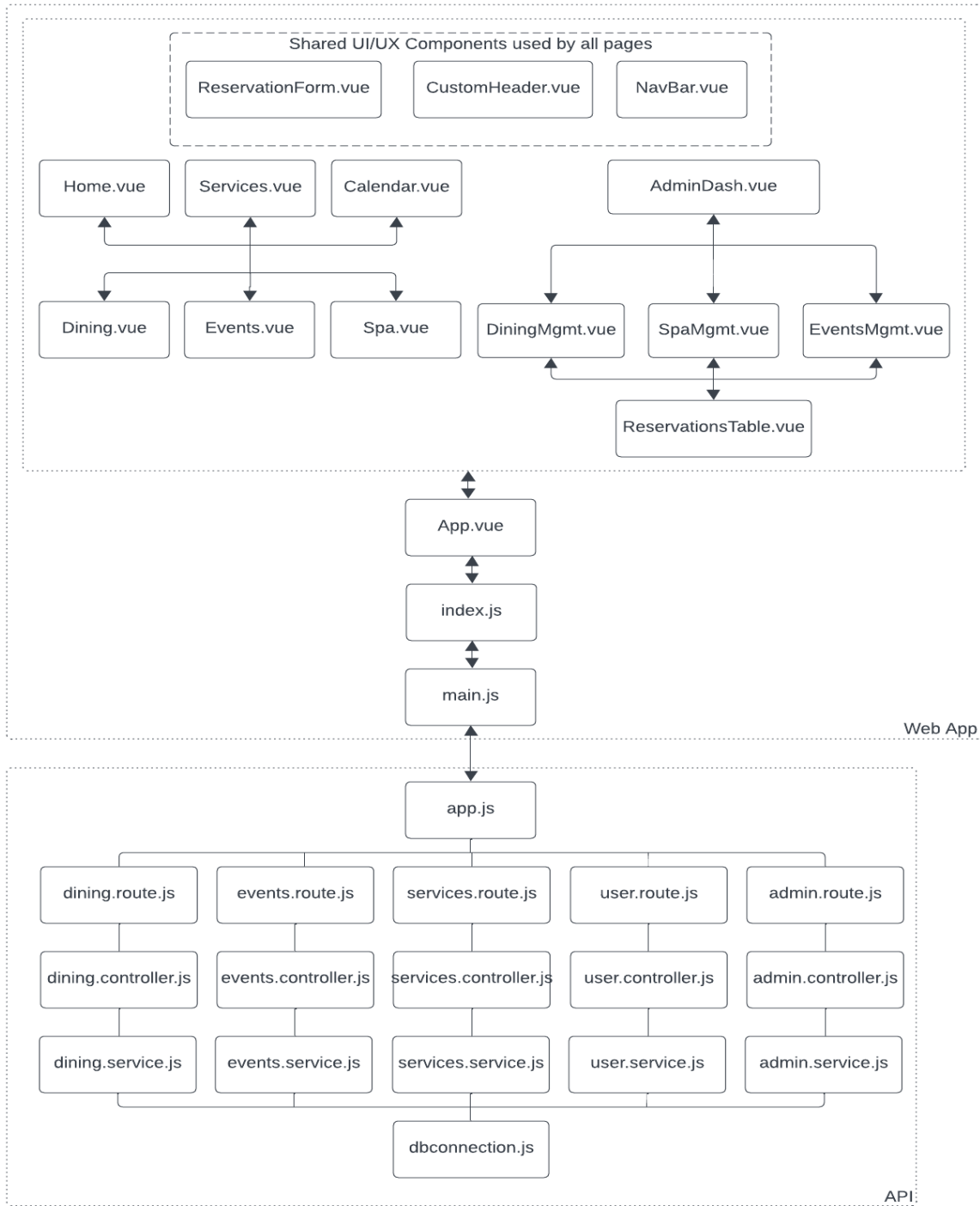
7.g Manager Handle Events



7.h Manager Handle Reservations



8.0 Class Diagram and Interface Specification



We have built our Web Application using Vue 3 and the backend API using Node. Both of these are javascript frameworks and as such we are not using classes or interfaces. The diagram above offers a high-level overview of our software project's component architecture, illustrating the interaction between the client-facing Web App and the API.

Web App(Vue 3):

- The Web App is the client-facing component of the system, developed using Vue 3.
- It encompasses a suite of user interface pages and shared components that collectively form the frontend of the application.

main.js:

- main.js serves as the entry point for the Vue.js application, initializing essential configurations and connections.

index.js:

- index.js acts as an intermediary between the main.js component and the root App.vue component in the Vue.js application.
- It contains the endpoints used by the router to direct http requests to specific components.

App.vue:

- App.vue acts as the root component of our Vue app, serving as a container for all other Vue components within the Web App.

Home.vue, Services.vue, Calendar.vue:

- These components represent distinct user-facing pages, accessible by guests.
- Home.vue is the system's homepage, while Services.vue and Calendar.vue provide guests with access to service and event-related information.

Dining.vue, Events.vue, Spa.vue:

- These components are dedicated to specific services, offering users detailed information and reservation capabilities.

AdminDash.vue:

- AdminDash.vue is tailored for administrators, equipping them with tools for managing service-related information.

DiningMgmt.vue, SpaMgmt.vue, EventsMgmt.vue:

- These components are integral to the administrative dashboard, enabling administrators to manage and update data for dining, spa, and event services.

ReservationsTable.vue:

- ReservationsTable.vue is a shared component used by administrators for an efficient overview and management of service reservations.

ReservationForm.vue, NavBar.vue, CustomHeader.vue:

- These are shared components used by other pages in the Web App to display UI/UX elements.

API (Node.js):

- The API component forms the backend infrastructure of the system, operating on Node.js.
- It serves as the intermediary between the frontend and the database, managing data requests and responses.

app.js:

- app.js is the entry point for the Node.js API, initializing the server and defining routes for incoming requests.

API Routes (dining.route.js, events.route.js, services.route.js, user.route.js, admin.route.js):

- These route components define endpoints for distinct system functionalities.
- They map incoming HTTP requests to corresponding actions within the controllers.

Controller Layer (.controller.js):

- The controllers encapsulate the core business logic of the application.
- They receive and process requests from the API routes, coordinating data manipulation and interaction with the service layer.

Service Layer (.service.js):

- The service layer abstracts data operations, providing methods for consistent interaction with the database.
- It interfaces with the database through the database connection, ensuring efficient data handling.

dbconnection.js:

- dbconnection.js is responsible for establishing and managing the database connection.
- It serves as the vital link between the service layer and the database, enabling critical CRUD (Create, Read, Update, Delete) operations.

9.0 Algorithms and Data Structures

The most complex data structure we are using in our project is an array of objects, which is not really that complex. The data populating all of the tables on the administrative and management pages and the data populating the informational displays on the public facing pages and the management pages will all be in this format. The data will be stored on relational databases in MySQL Server.

Additionally, we are not using any complicated, custom or mathematical algorithms.

10.0 User Interface Design and Implementation

For the most part, other than minor styling changes, we will be implementing the design as described in Section 4.0. To increase ease of use, we are considering consolidating the Calendar page, Services page and main Home page into one page. This would allow resort guest users to more quickly make a reservation or find information about schedules and services. For development purposes, we are implementing the calendar and service icon list as separate components. Once we have that functionality implemented, which is the top priority, we will work on the redesign of the Home page to accommodate adding in those components there.

11.0 Design of Tests

The following are the test cases to be used for unit testing:

- TC - 1: Tests login functionality and accuracy
- TC - 2: Tests administrator ability to add managers
- TC - 3: Tests administrator ability to delete managers
- TC - 4: Tests administrator ability to add services
- TC - 5: Tests administrator ability to delete services
- TC - 6: Tests viewability of statistics
- TC - 7: Tests administrator ability to download statistical report

- TC - 8: Tests manager ability to add reservations
- TC - 9: Tests manager ability to update reservations
- TC - 10: Tests manager ability to delete reservations
- TC - 11: Tests spa manager ability to add services
- TC - 12: Tests spa manager ability to delete services
- TC - 13: Tests events manager ability to add events
- TC - 14: Tests events manager ability to delete events
- TC - 15: Tests user ability to make a reservation

<p>Test Case Identifier: TC-1 Use Case Tested: UC-2 Pass/Fail Criteria: The test will pass if the user is able to successfully log in to the system. The test will fail if the user inputs an invalid username or password.</p>	
Test Procedure:	Expected Result
User inputs valid username and password	Login succeeds
User inputs invalid username or password	Login fails

<p>Test Case Identifier: TC-2 Use Case Tested: UC-11, UC-12 Pass/Fail Criteria: The test will pass if a user with administrator privileges successfully adds a manager. The test will fail if the user without administrator privileges tries to add a manager.</p>	
Test Procedure:	Expected Result
User with admin privileges tries to add a manager	Manager is successfully added and saved to database
User without admin privilege tries to add a manager	Request to add manager fails

<p>Test Case Identifier: TC-3 Use Case Tested: UC-11, UC-12 Pass/Fail Criteria: The test will pass if a user with administrator privileges successfully deletes a manager. The test will fail if the user without administrator privileges tries to delete a manager.</p>	
Test Procedure:	Expected Result

User with admin privileges tries to delete a manager	Manager is successfully deleted and saved to database
User without admin privilege tries to delete a manager	Request to delete manager fails

<p>Test Case Identifier: TC-4 Use Case Tested: UC-11, UC-12 Pass/Fail Criteria: The test will pass if a user with administrator privileges successfully adds a service. The test will fail if the user without administrator privileges tries to add a service.</p>	
Test Procedure:	Expected Result
User with admin privileges tries to add a service	Service is successfully added and saved to database
User without admin privilege tries to add a service	Request to add service fails

<p>Test Case Identifier: TC-5 Use Case Tested: UC-11, UC-12 Pass/Fail Criteria: The test will pass if a user with administrator privileges successfully deletes a service. The test will fail if the user without administrator privileges tries to delete a service.</p>	
Test Procedure:	Expected Result
User with admin privileges tries to delete a service	Service is successfully deleted and saved to database
User without admin privilege tries to delete a service	Request to delete service fails

<p>Test Case Identifier: TC-6 Use Case Tested: UC-14 Pass/Fail Criteria: The test will pass if the admin page successfully retrieves and displays statistics. The test will fail if the admin page cannot retrieve or display service-related statistics.</p>	
Test Procedure:	Expected Result

SQL server and API running, and the web app makes the call to GET service statistics	The statistics are successfully retrieved and displayed
A mock false response from the API is implemented by the test for the call to GET service statistics	An error message regarding the failed request is displayed

Test Case Identifier: TC-7
Use Case Tested: UC-14
Pass/Fail Criteria: The test will pass if a user with administrator privileges downloads a report. The test will fail if the user without administrator privileges tries to download a report.

Test Procedure:	Expected Result
User with admin privileges tries to download a statistics report	The report is successfully downloaded
User with admin privileges tries to download a statistics report when no stats are available	The report is not downloaded. Error message displays.
User without admin privilege tries to download a statistics report	The report is not downloaded

Test Case Identifier: TC-8
Use Case Tested: UC-16
Pass/Fail Criteria: The test will pass if a user with manager privileges is able to add a reservation for their service. The test will fail if a user with manager privileges tries to add a reservation for a service that is not theirs.

Test Procedure:	Expected Result
User with manager privileges tries to add a reservation for their service	Reservation is successfully added and saved to database
User with manager privilege tries to add a service that is not theirs	Reservation is not added. Error message is logged to console.

Test Case Identifier: TC-9
Use Case Tested: UC-16
Pass/Fail Criteria: The test will pass if a user with manager privileges is able to update a

reservation for their service. The test will fail if a user with manager privileges tries to update a reservation for a service that is not theirs.	
Test Procedure:	Expected Result
User with manager privileges tries to update a reservation for their service	Reservation is successfully updated
User with manager privilege tries to update a service that is not theirs	Reservation is not updated. Error message is logged to console.

Test Case Identifier: TC-10 Use Case Tested: UC-16 Pass/Fail Criteria: The test will pass if a user with manager privileges is able to delete a reservation for their service. The test will fail if a user with manager privileges tries to delete a reservation for a service that is not theirs.	
Test Procedure:	Expected Result
User with manager privileges tries to delete a reservation for their service	Reservation is successfully deleted
User with manager privilege tries to delete a service that is not theirs	Reservation is not deleted. Error message is logged to console.

Test Case Identifier: TC-11 Use Case Tested: UC-16 Pass/Fail Criteria: The test will pass if a user with spa manager privileges successfully adds a service. The test will fail if the user without spa manager privileges tries to add a service.	
Test Procedure:	Expected Result
User with spa manager privileges tries to add a service	Service is successfully added and saved to database
User without spa manager privileges tries to add a service	Request to add service fails

Test Case Identifier: TC-12 Use Case Tested: UC-16 Pass/Fail Criteria: The test will pass if a user with spa manager privileges successfully deletes a service. The test will fail if the user without spa manager privileges tries to delete a	
--	--

service.	
Test Procedure:	Expected Result
User with spa manager privileges tries to delete a service	Service is successfully deleted and saved to database
User without spa manager privileges tries to delete a service	Request to delete service fails

Test Case Identifier: TC-13 Use Case Tested: UC-16 Pass/Fail Criteria: The test will pass if a user with event manager privileges successfully adds an event. The test will fail if the user without event manager privileges tries to add an event.	
Test Procedure:	Expected Result
User with event manager privileges tries to add an event	Event is successfully added and saved to database
User without event manager privileges tries to add an event	Request to add event fails

Test Case Identifier: TC-14 Use Case Tested: UC-16 Pass/Fail Criteria: The test will pass if a user with event manager privileges successfully deletes an event. The test will fail if the user without event manager privileges tries to delete an event.	
Test Procedure:	Expected Result
User with event manager privileges tries to delete a event	Event is successfully deleted and saved to database
User without event manager privileges tries to delete an event	Request to delete event fails

Test Case Identifier: TC-15 Use Case Tested: UC-7 Pass/Fail Criteria: The test will pass if a user logged in as a resort guest is able to make a reservation. The test will fail if a user logged in as a resort guest tries to make a reservation with incomplete information.	
--	--

Test Procedure:	Expected Result
User logged in as a resort guest submit a registration form that is properly filled in	Reservation is successfully added and saved to database
User logged in as a resort guest submits a registration form that is not properly filled in	Reservation is not added. Error message is displayed

Project Management and Plan of Work

a. Merging the Contributions from Individual Team Members

The team leader, Heather Young, created a formatted draft of the finished report. This included the outline of the finished product with placeholder text for the unfinished sections. The sections to be completed were divided up among the team members. After everyone completed their sections, Heather reviewed the document and made any necessary changes needed for cohesion.

b. Project Coordination and Progress Report

None of the use cases have been fully implemented yet. The framework for the project has been set up, including a Node.js API, a Vue web application, and an MSSQL database with all the necessary tables. This week, each team member took on one of the pages for the web application, worked on that code and submitted PRs for the work to be merged.

c. History of Work

From September 25th to October 2nd, the team continued the work on mock-ups of the user interface and began the coding of these mock-ups, along with submitting the first part of Report 2. This phase marked the transition from design to implementation.

During the week of October 2nd to 9th, our focus shifted to the system architecture. We started working on class diagrams, created all required application pages, and established the necessary links between them. This period was important for laying down the foundation of our application.

From October 9th to 16th, we continued on class diagrams and the user interface, while also starting to develop important functionalities of the application. This phase was important for integrating the design and functional aspects of the project.

In the following weeks, from October 16th to November 3rd, we worked on the system architecture and design, continued enhancing the app functionality, and prepared for the first demo to show the initial functionality in our app.

The period from November 3rd to 13th saw continued coding and debugging efforts, improvements in the user interface, and the application programming interface.

From November 13th to 20th, we maintained focus on coding, debugging, while also adding extra features to the application as needed. The final stretch from November 20th to today was dedicated to preparing and submitting Demo 2, and working on some unit tests for our application.

Overall, the project progressed smoothly and largely as planned. From the initial stages of designing the user interface to the final presentation of Demo 2, each step was completed on schedule.

Key Accomplishments

- Successful transition from UI design to coding implementation.

