NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

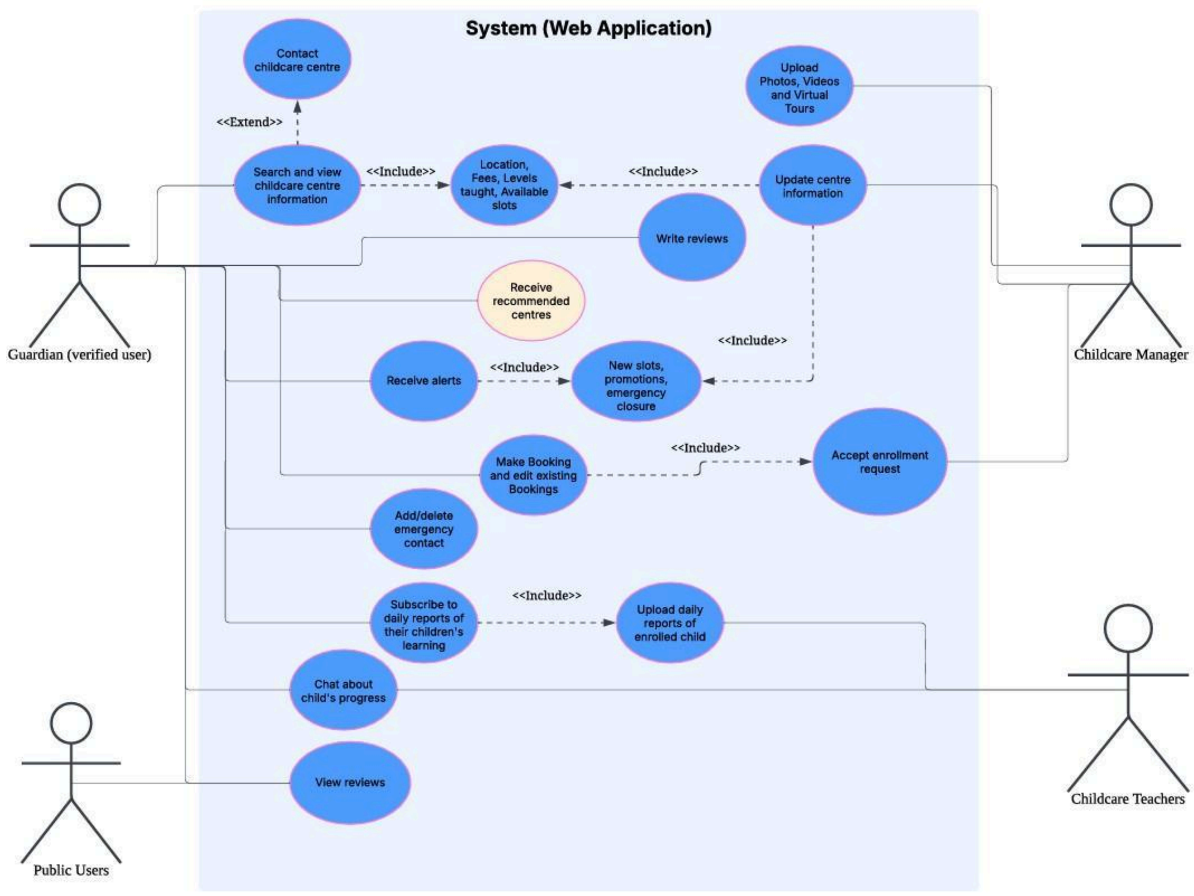
SC2006 SOFTWARE ENGINEERING

Lab 3 Deliverables

Lab Group	FDAB
Group	2
App	Carescout
Members	Agarwal Dhruv Kris Khor Hai Xiang Liu Chunyi Sridhar Abhinav Vaghela Nikunj Wong Shi Hui Cheryl

1. Use Case Model

1.1 Complete Use Case Diagram


System (Web Application)
Contact childcare centre
<<Extend>>
Search and view childcare centre information
<<Include>>
Location, Fees, Levels taught, Available slots
<<Include>>
Update centre information
Upload Photos, Videos and Virtual Tours
Write reviews
Receive recommended centres
<<Include>>
Receive alerts
<<Include>>
New slots, promotions, emergency closure
Make Booking and edit existing Bookings
<<Include>>
Accept enrollment request
Add/delete emergency contact
Subscribe to daily reports of their children's learning
<<Include>>
Upload daily reports of enrolled child
Chat about child's progress
View reviews
Guardian (verified user)
Childcare Manager
Childcare Teachers
Public Users

1.2 Use Case Descriptions

Use Case ID:	UC-01		
Use Case Name:	Guardian log-in and sign-up		
Created By:		Last Updated By:	
Date Created:	2025-02-05	Date Last Updated:	2025-02-12

Actor:	Website, Guardian
Description:	The Guardian must log in or sign up to access the website’s features.
Preconditions:	The Guardian is not already logged in. The device has an active internet connection.
Postconditions:	If login/sign-up is successful, the Guardian is directed to the main page. If unsuccessful, the Guardian is prompted to retry or reset their credentials.

Priority:	1
Frequency of Use:	Everytime the guardian launches the website
Flow of Events:	S1. Guardian launches the website. S2. The website requests the Guardian to log in or sign up. S3. To sign up, guardian enters: - Name
	- Email address
	- Phone number - Password S4. The website sends a verification email to the Guardian. S5. If the guardian has an existing account, they enter their username and password. S6. The website verifies the credentials. S6. The website directs the guardian to the main page.

Altern ative Flows :	**AF-S3: If email or phone number is already in use** 1. The website displays "Email or phone number is already in use. Please try another or log into your existing account." 2. The website goes back to S2. **AF-S4: If email verification is not received** 1. The guardian can request a resend verification email option. 2. If still unsuccessful, they are prompted to check spam folders to contact support. **AF-S5: If email or password is incorrect**

	1. The website displays "Incorrect login credentials, Please try again". 2. The website goes back to S2.
Exceptions:	**No Internet Connection:** The website displays an error message and suggests reconnecting. **Account Locked:** After multiple failed login attempts, the website temporarily locks the account and prompts the Guardian to reset their password.
Includes:	Email verification Password recovery
Special Requirements:	Passwords must be stored using industrystandard encryption (e.g., bcrypt, Argon2) and meet security standards (minimum length, special characters, etc).. The system must comply with data privacy laws

	(e.g., GDPR, CCPA) regarding user information.
Assumptions:	Guardian has a valid email and phone number. Guardian will check their email for verification.
Notes and Issues:	Consider **multi-factor authentication (MFA)** for enhanced security.

Use Case ID:	UC-02		
Use Case Name:	Search and reserve a childcare centre slot		
Created By:		Last Updated By:	
Date Created:	2025-02-05	Date Last Updated:	2025-02-12

Actor:	Website, Guardian and Childcare Manager
Description:	The guardian wants to search for a childcare centre based on their preferences and make reservations and/or edit existing reservations. Childcare centre manager will review and approve the request.
Preconditions:	Guardian must have an active account and is logged in Childcare centre must have available slots for reservation.
Postconditions:	Reservation is recorded in the system Childcare centre’s booking schedule is updated Guardian receives a confirmation email.

Priority:	2
Frequency of Use:	Varies with different guardians

Flow of Events:	S1. Guardian enters search criteria (location, fees, level etc.) S2. Website displays a list of recommended childcare centres based on user preferences, past searches and reviews. S3. Guardian clicks on a childcare center ‘node’, and the information of the childcare center is displayed. S4. Guardian clicks on the “Book Now” button. S5. Website displays date and time slots. S6. Guardian selects a date and time. S7. Childcare centre receives the request, and the admin will accept or reject the booking. S8. Child is successfully enrolled into the childcare centre. S9. Website sends confirmation (email) to the guardian, including date and time details of the slot. S10. Guardian clicks on the “confirm” button upon checking details are correct.

Alternative Flows:	**AF-S1: No results for chosen criteria** 1. Website displays “No results. Please try searching again.” **AF-S4: There are no available slots** 1. Display error message “There are no slots available. Please try another outlet/centre.” 2. Website brings the guardian back to S1. **AF-S6: If the childcare centre has not responded in 3 working days** 1. The website will notify the centre of the new reservation every working day.
	2. If there is still no response after 3 working days, the website automatically cancels the booking. 3. App sends the guardian a notification “Sorry, please try another centre.” **AF-S9: Reservation details are incorrect** 1. Guardian clicks on “Incorrect reservation” within the confirmation email sent to them. 2. App will cancel their booking automatically. 3. Guardian will be brought to S4 through the button.
Exceptions:	**System downtime:** If the system is under maintenance, the app informs the Guardian and prevents booking.

Includes:	Information such as Location, Fees, Levels taught, Available slots. Childcare manager approval process.
Special Requirements:	The app should handle high traffic (up to 10,000 concurrent users) efficiently to avoid booking failures.
Assumptions:	Childcare centres actively update availability. Guardians have valid payment methods (if prepayment is required to confirm reservation).
Notes and Issues:	Consider adding a **waitlist feature** for fully booked centres. Introduce an **auto-accept feature** for certain bookings if enabled by the childcare centre manager.

Use Case ID:	UC-03		
Use Case Name:	Leaving a review (only verified users)		
Created By:		Last Updated By:	
Date Created:	2025-02-05	Date Last Updated:	2025-02-12

Actor:	Website, Guardian
Description:	Guardians are able to rate the service provided by the childcare centre after their child has attended it.
Preconditions:	The Guardian must have an active, logged-in account. The child’s reservation must be completed before a review can be submitted.
Postconditions:	Review is posted and can be viewed by other users.
Priority:	7
Frequency of Use:	Only after their child has attended the centres service.
Flow of Events:	S1. Guardian navigates to the centre page and clicks the “Write a Review” button. S2. Website prompts the guardian to enter a rating (1-5 stars) and write their review in words.
	S3. Guardian writes a review and clicks on “Submit”. S4. Website will verify if the guardian has been logged into an account. S5. Website displays a confirmation message “Post your review?” S6. Upon clicking “Yes”, the review will be published.

Alternative Flows:	**AF-S3: Incomplete review** 1. Display error message “Rating has not been given” or “Description is empty”. 2. Bring guardians back to S2 to continue filling up their review. **AF-S4: User is not logged in/verified** 1. Website displays the message “Please log in first”. 2. Website displays a page for guardians to log in, then brings them back to S2.
Exceptions:	**Fake reviews:** The system may flag and investigate reports of fake reviews.
Includes:	Rating system (1-5 stars). Text-based reviews.
Special Requirements:	Reviews must be moderated for offensive language.
Assumptions:	Guardians will leave honest and constructive reviews.
Notes and Issues:	Consider adding a **“Reply to Review”** feature for childcare managers.

Use Case ID:	UC-04

Use Case Name:	Receiving notifications from the app		
Created By:		Last Updated By:	
Date Created:	2025-02-05	Date Last Updated:	2025-02-12

Actor:	Website, Guardian
Description:	Guardians can receive and customize the notifications to their preference.
Preconditions:	Guardian must have notifications enabled on their device.
Postconditions:	Guardian successfully receives and views the notification.
Priority:	3
Frequency of Use:	One-time setup, but customizations still can be changed anytime. Notifications will remain ongoing.
Flow of Events:	S1. Guardian navigates to notification settings and customize notifications they would like to receive. S2. Website displays a list of past and new notifications. S3. If guardian has notifications on, they will receive alerts for:

	- New slots opened - Pickup times - Promotions or discounts - Emergency closures - Daily report notifications (if subscribed - UC6)
Alternative Flows:	**AF-S3: System fails to send notification** 1. Website logs the failure and re-attempts to send the notification.
Exceptions:	**Guardian disabled notifications:** The website must report this setting and prompt them to turn it on should they wish to receive notifications.
Includes:	Notification preferences for different event types.
Special Requirements:	The website should comply with mobile OS notification policies (Android/iOS).
Assumptions:	Guardians will check their notifications.
Notes and Issues:	Consider allowing **SMS/email notifications** for critical updates.

Use Case ID:	UC-05

Use Case Name:	Subscribing to daily reports of child		
Created By:		Last Updated By:	
Date Created:	2025-02-05	Date Last Updated:	2025-02-12

Actor:	Website, Guardian and Childcare Providers
Description:	Guardians can receive a newsletter about their child when their child is in the childcare centre.
Preconditions:	Guardian has an active booking with the childcare center. Childcare providers must offer daily reports.
Postconditions:	If subscribed, the parent would receive daily reports. If unsubscribed, reports will no longer be sent to the parent. Guardian can change their subscription status anytime.
Priority:	4

Frequency of Use:	Once after every complete reservation.
Flow of Events:	S1. Guardian navigates to the child's enrolled childcare center profile. S2. Guardian selects the "Subscribe to Daily Reports" option. S3. App confirms the subscription and saves the preference. S4. The childcare provider will upload daily reports, after which the app will send a notification to the guardian. S5. Guardian can view and download reports from the app.
Alternative Flows:	**AF-S2: Childcare centre does not provide subscription service** 1. App displays the message “Feature unavailable for this centre. Please contact them via the ‘Chat’ function. **AF-S4: Childcare centre fails to provide the report** 1. App sends reminders to the childcare centre admin to upload daily reports as soon as possible. 2. App sends notification to guardians informing them of delayed reports.

	3. Once a report is out for that child, the guardians will be notified and proceed to S5.
Exceptions:	**System error:** When notifications are unable to be sent, log the error in the system and retry in set intervals. **Invalid number/email:** System should verify contact details upon subscription and prompt to update incorrect information.
Includes:	Subscription management module.
Special Requirements:	The system must comply with data privacy regulations as mandated by government agencies.
Assumptions:	Childcare centres update reports daily.
	Guardians check their notifications regularly.
Notes and Issues:	Expanding to **monthly/weekly** reports.

Use Case ID:	UC-06		
Use Case Name:	Chat Function with Teachers		
Created By:		Last Updated By:	

Date Created:	2025-02-05	Date Last Updated:	2025-02-12

Actor:	Guardian and Childcare Teachers
Description:	Guardians and teachers keep an open line of communication through the website.
Preconditions:	Guardian is logged into the website. Message from guardian is successfully sent and received.
Postconditions:	Guardians can continue chatting with the childcare provider as long as their child is enrolled. Previous conversations remain accessible for reference.
Priority:	5

Frequency of Use:	Depends on either party.
Flow of Events:	S1. Guardian navigates to the "Chat" section. S2. Website displays a list of childcare providers the guardian has confirmed bookings with.
	S3. Guardian selects the provider and sends a message. S4. Childcare provider responds, and the conversation continues.

Alternative Flows:	**AF-S3: Message delivery fails** 1. Website displays an error message and prompts the guardian to retry. 2. If the issue persists, the guardian is advised to check their internet connection or contact support.
Exceptions:	**Untimely response:** Send automated reminders after a set inactivity period. **Account deactivated:** Notify and provide instructions to reactivate or transfer communication access.
Includes:	Chat notification module. Message storage system.
Special Requirements:	Messages should be stored securely for future reference. Chat logs must comply with data protection regulations as mandated by government agencies.
Assumptions:	Both parties respond in a timely manner. Both parties will use the chat responsibly.
Notes and Issues:	Future updates may include **calling options**.

Use Case ID:	UC-07

Use Case Name:	Adding Emergency Contacts		
Created By:		Last Updated By:	
Date Created:	2025-02-05	Date Last Updated:	2025-02-12

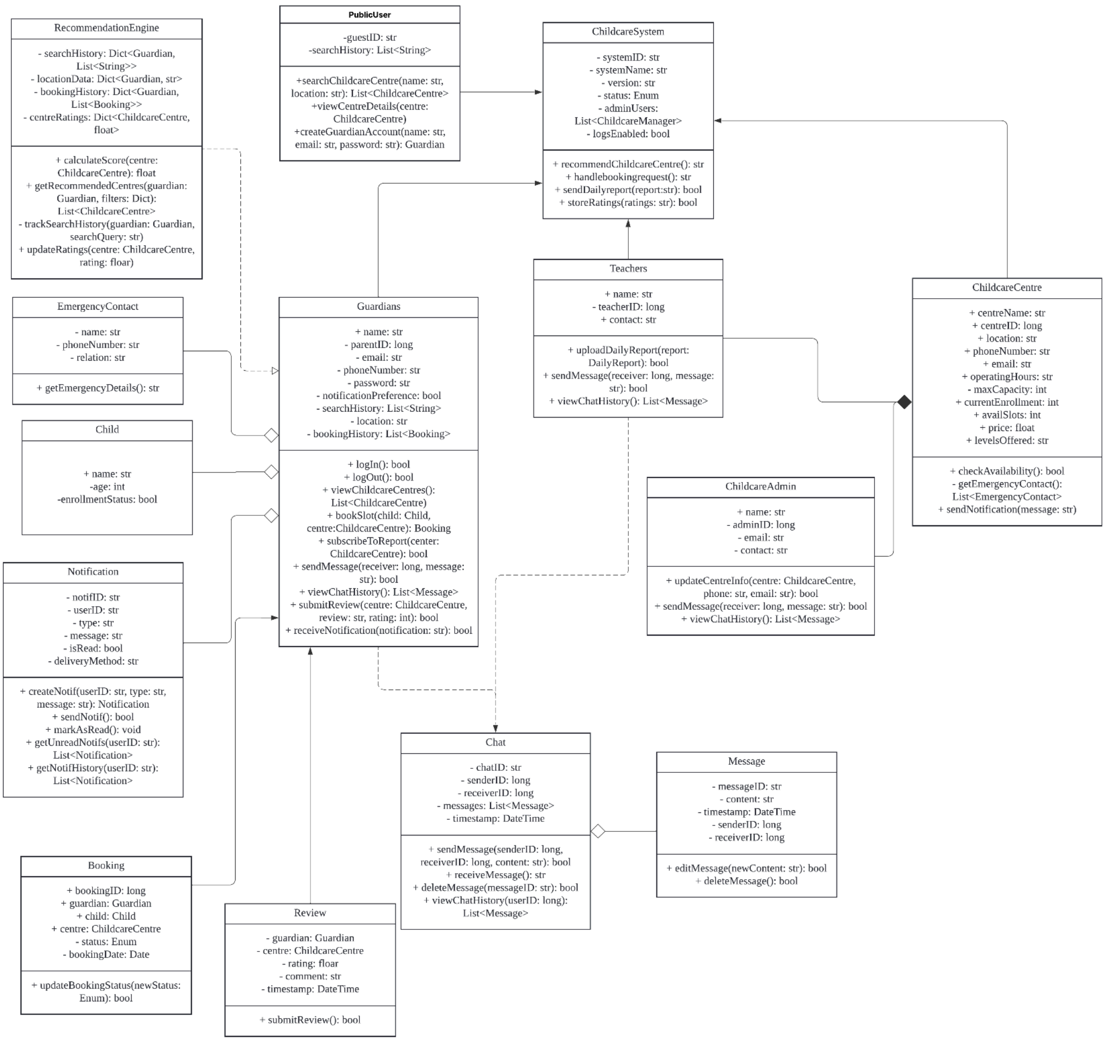
Actor:	Website, Guardian, Childcare Provider
Description:	Guardians can add their emergency contact in case something happens to their child. The childcare centre can update the parents on what has happened and the condition of their child.
Preconditions:	Guardian must be logged into the application. Guardian must have an active profile for their child.
Postconditions:	In case of an emergency where the guardian is uncontactable, the childcare provider will notify the listed contacts. Emergency contacts will receive real-time updates about the child's condition.
Priority:	6

Frequency of Use:	Once, unless there is a change of contact from one guardian to another or if the existing contact changes number.
Flow of Events:	S1. Guardian navigates to the "Emergency Contacts" section in their profile settings. S2. Website displays any current emergency contacts. S3. Guardian selects the "Add New Contact" button and enters emergency contact details (Name, Phone Number, Relationship). S4. Website verifies the contact information format and saves it if accurate. It then displays a confirmation message on screen.
Alternative Flows:	**AF-S3: Invalid contact information** 1. Website displays an error message “Incorrect details. Please check and try again.” 2. Guardian is brought back to S2. **AF-S3: Maximum number of contacts have been added.** 1. Website displays “Maximum number of contacts reached.” 2. Guardian is brought back to S2, where they can choose to delete any contacts.

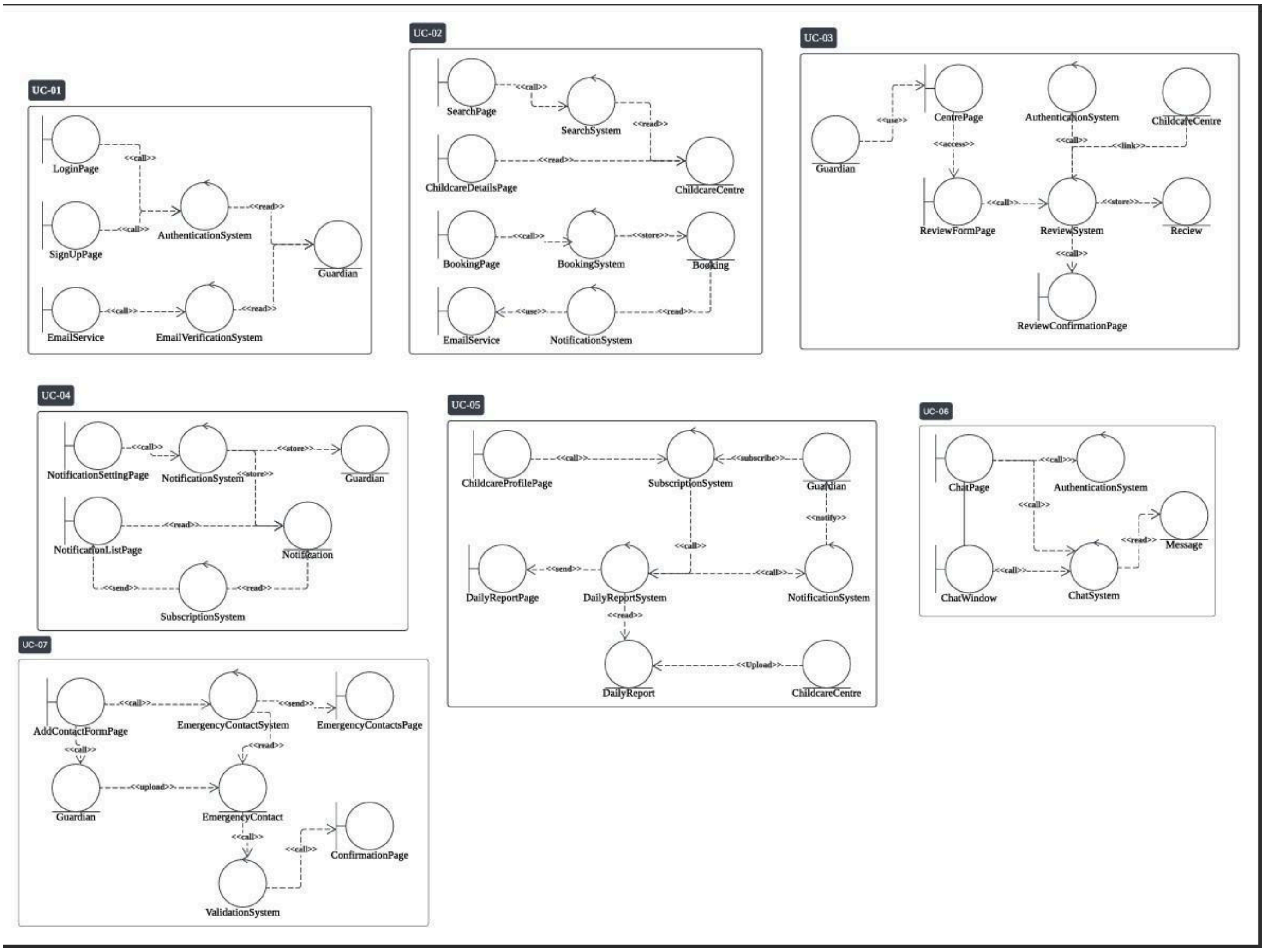
Exceptions:	**Outdated/unreachable contact:** System prompts guardian to confirm contact details periodically. **System error:** When guardian is unable to update contact details, implement data recovery mechanisms and provide support access for manual updates.
Includes:	Contact verification system.
	Emergency notification system.
Special Requirements:	Personal user data of emergency contacts must not be shared with third parties without explicit consent.
Assumptions:	Guardians have other trusted contacts.
Notes and Issues:	Consider adding an option for guardians to set **priority levels** for listed emergency contacts.

2. Design Model

2.1 Class Diagram of Entity Classes

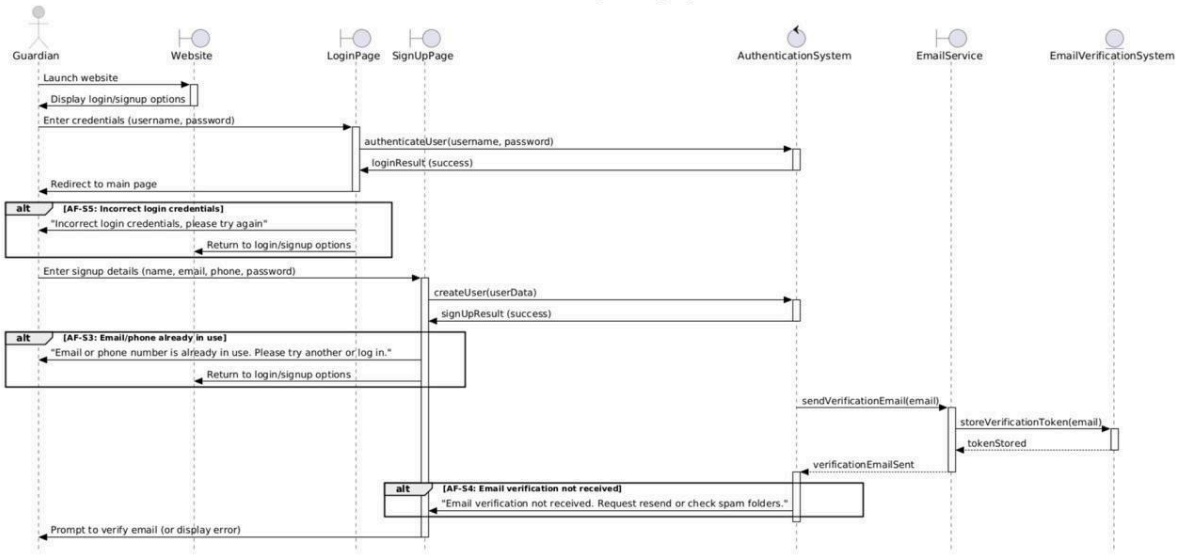

RecommendationEngine
- searchHistory: Dict<Guardian, List<String>>
- locationData: Dict<Guardian, str>
- bookingHistory: Dict<Guardian, List<Booking>>
- centreRatings: Dict<ChildcareCentre, float>
+ calculateScore(centre: ChildcareCentre): float
+ getRecommendedCentres(guardian: Guardian, filters: Dict): List<ChildcareCentre>
- trackSearchHistory(guardian: Guardian, searchQuery: str)
+ updateRatings(centre: ChildcareCentre, rating: floar)
PublicUser
-guestID: str
-searchHistory: List<String>
+searchChildcareCentre(name: str, location: str): List<ChildcareCentre>
+viewCentreDetails(centre: ChildcareCentre)
+createGuardianAccount(name: str, email: str, password: str): Guardian
ChildcareSystem
- systemID: str
- systemName: str
- version: str
- status: Enum
- adminUsers: List<ChildcareManager>
- logsEnabled: bool
+ recommendChildcareCentre(): str
+ handlebookingrequest(): str
+ sendDailyreport(report:str): bool
+ storeRatings(ratings: str): bool
EmergencyContact
- name: str
- phoneNumber: str
- relation: str
+ getEmergencyDetails(): str
Guardians
+ name: str
- parentID: long
- email: str
- phoneNumber: str
- password: str
- notificationPreference: bool
- searchHistory: List<String>
- location: str
- bookingHistory: List<Booking>
+ logIn(): bool
+ logOut(): bool
+ viewChildcareCentres(): List<ChildcareCentre)
+ bookSlot(child: Child, centre:ChildcareCentre): Booking
+ subscribeToReport(center: ChildcareCentre): bool
+ sendMessage(receiver: long, message: str): bool
+ viewChatHistory(): List<Message>
+ submitReview(centre: ChildcareCentre, review: str, rating: int): bool
+ receiveNotification(notification: str): bool
Teachers
+ name: str
- teacherID: long
+ contact: str
+ uploadDailyReport(report: DailyReport): bool
+ sendMessage(receiver: long, message: str): bool
+ viewChatHistory(): List<Message>
ChildcareCentre
+ centreName: str
+ centreID: long
+ location: str
+ phoneNumber: str
+ email: str
+ operatingHours: str
- maxCapacity: int
+ currentEnrollment: int
+ availSlots: int
+ price: float
+ levelsOffered: str
+ checkAvailability(): bool
- getEmergencyContact(): List<EmergencyContact>
+ sendNotification(message: str)
Child
+ name: str
-age: int
-enrollmentStatus: bool
ChildcareAdmin
+ name: str
- adminID: long
- email: str
- contact: str
+ updateCentreInfo(centre: ChildcareCentre, phone: str, email: str): bool
+ sendMessage(receiver: long, message: str): bool
+ viewChatHistory(): List<Message>
Notification
- notifID: str
- userID: str
- type: str
- message: str
- isRead: bool
- deliveryMethod: str
+ createNotif(userID: str, type: str, message: str): Notification
+ sendNotif(): bool
+ markAsRead(): void
+ getUnreadNotifs(userID: str): List<Notification>
+ getNotifHistory(userID: str): List<Notification>
Chat
- chatID: str
- senderID: long
- receiverID: long
- messages: List<Message>
- timestamp: DateTime
+ sendMessage(senderID: long, receiverID: long, content: str): bool
+ receiveMessage(): str
+ deleteMessage(messageID: str): bool
+ viewChatHistory(userID: long): List<Message>
Message
- messageID: str
- content: str
- timestamp: DateTime
- senderID: long
- receiverID: long
+ editMessage(newContent: str): bool
+ deleteMessage(): bool
Booking
+ bookingID: long
+ guardian: Guardian
+ child: Child
+ centre: ChildcareCentre
- status: Enum
- bookingDate: Date
+ updateBookingStatus(newStatus: Enum): bool
Review
- guardian: Guardian
- centre: ChildcareCentre
- rating: floar
- comment: str
- timestamp: DateTime
+ submitReview(): bool

2.2 Key Boundary Classes and control classes

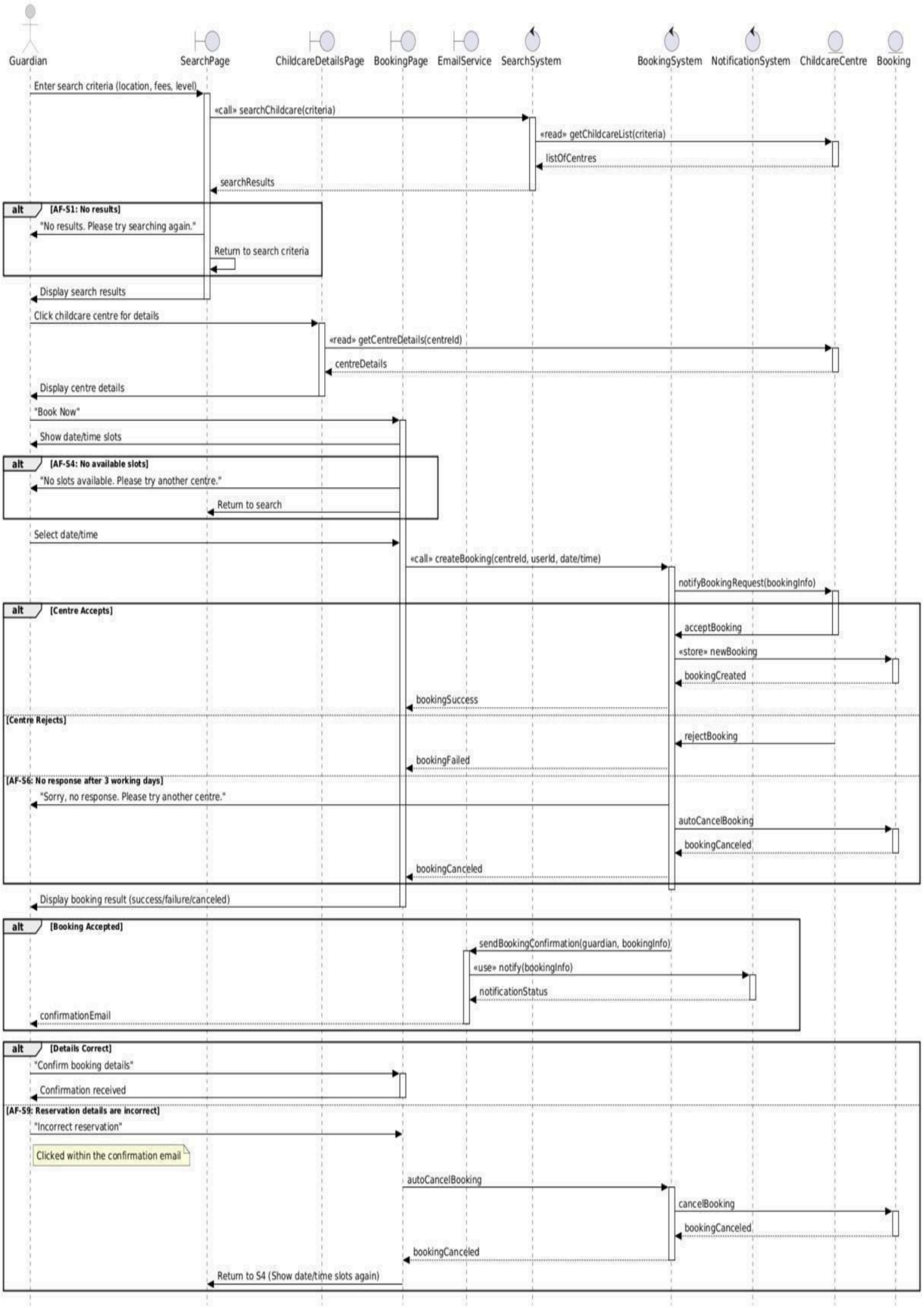


2.3 Sequence diagrams of some use cases

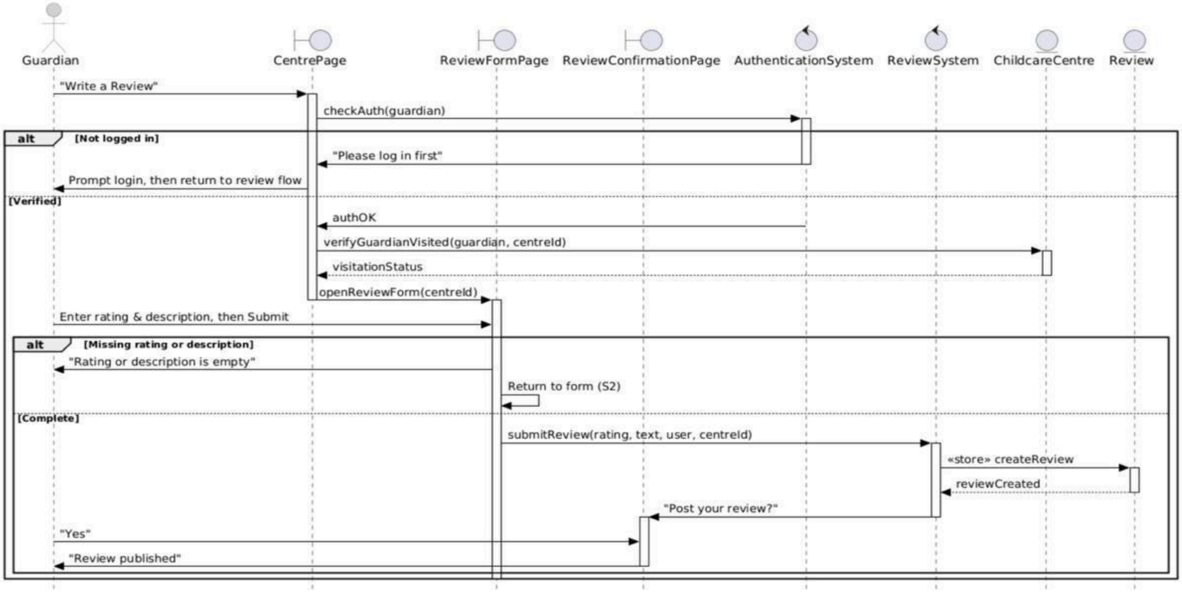
UC-01: Guardian Log-In and Sign-Up


Guardian
Website
LoginPage
SignUpPage
AuthenticationSystem
EmailService
EmailVerificationSystem
Launch website
Display login/signup options
Enter credentials (username, password)
authenticateUser(username, password)
loginResult (success)
Redirect to main page
alt
[AF-S5: Incorrect login credentials]
"Incorrect login credentials, please try again"
Return to login/signup options
Enter signup details (name, email, phone, password)
createUser(userData)
signUpResult (success)
alt
[AF-S3: Email/phone already in use]
"Email or phone number is already in use. Please try another or log in."
Return to login/signup options
sendVerificationEmail(email)
storeVerificationToken(email)
tokenStored
verificationEmailSent
alt
[AF-S4: Email verification not received]
"Email verification not received. Request resend or check spam folders."
Prompt to verify email (or display error)

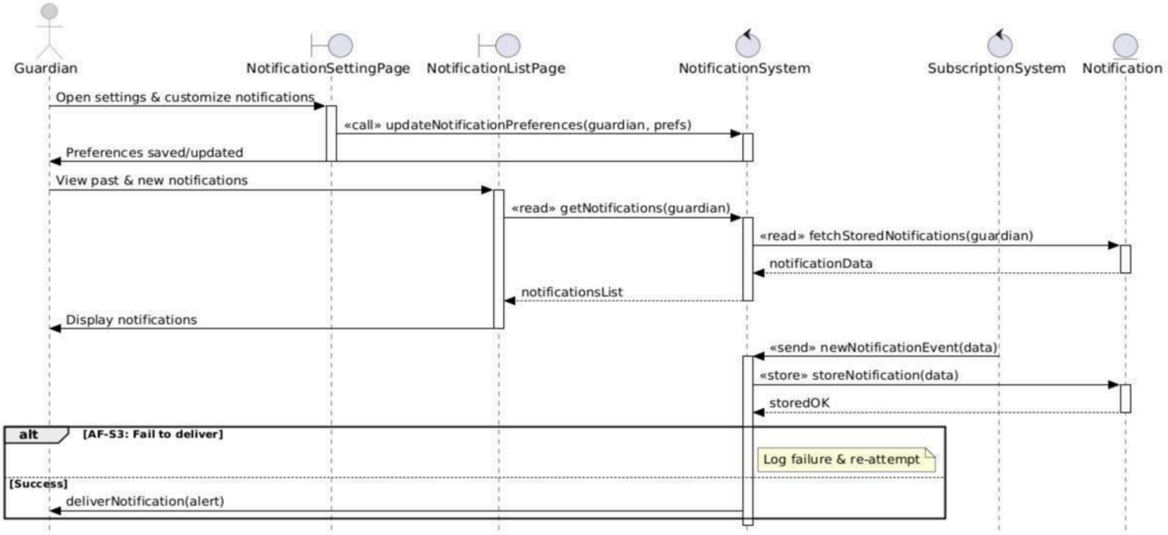
UC-02: Search and Reserve a Childcare Centre Slot


Guardian
SearchPage
ChildcareDetailsPage
BookingPage
EmailService
SearchSystem
BookingSystem
NotificationSystem
ChildcareCentre
Booking
Enter search criteria (location, fees, level)
«call» searchChildcare(criteria)
«read» getChildcareList(criteria)
listOfCentres
searchResults
alt
[AF-S1: No results]
"No results. Please try searching again."
Return to search criteria
Display search results
Click childcare centre for details
«read» getCentreDetails(centreId)
centreDetails
Display centre details
"Book Now"
Show date/time slots
alt
[AF-S4: No available slots]
"No slots available. Please try another centre."
Return to search
Select date/time
«call» createBooking(centreId, userId, date/time)
notifyBookingRequest(bookingInfo)
alt
[Centre Accepts]
acceptBooking
«store» newBooking
bookingCreated
bookingSuccess
[Centre Rejects]
rejectBooking
bookingFailed
[AF-S6: No response after 3 working days]
"Sorry, no response. Please try another centre."
autoCancelBooking
bookingCanceled
bookingCanceled
Display booking result (success/failure/canceled)
alt
[Booking Accepted]
sendBookingConfirmation(guardian, bookingInfo)
«use» notify(bookingInfo)
notificationStatus
confirmationEmail
alt
[Details Correct]
"Confirm booking details"
Confirmation received
[AF-S9: Reservation details are incorrect]
"Incorrect reservation"
Clicked within the confirmation email
autoCancelBooking
cancelBooking
bookingCanceled
bookingCanceled
Return to S4 (Show date/time slots again)

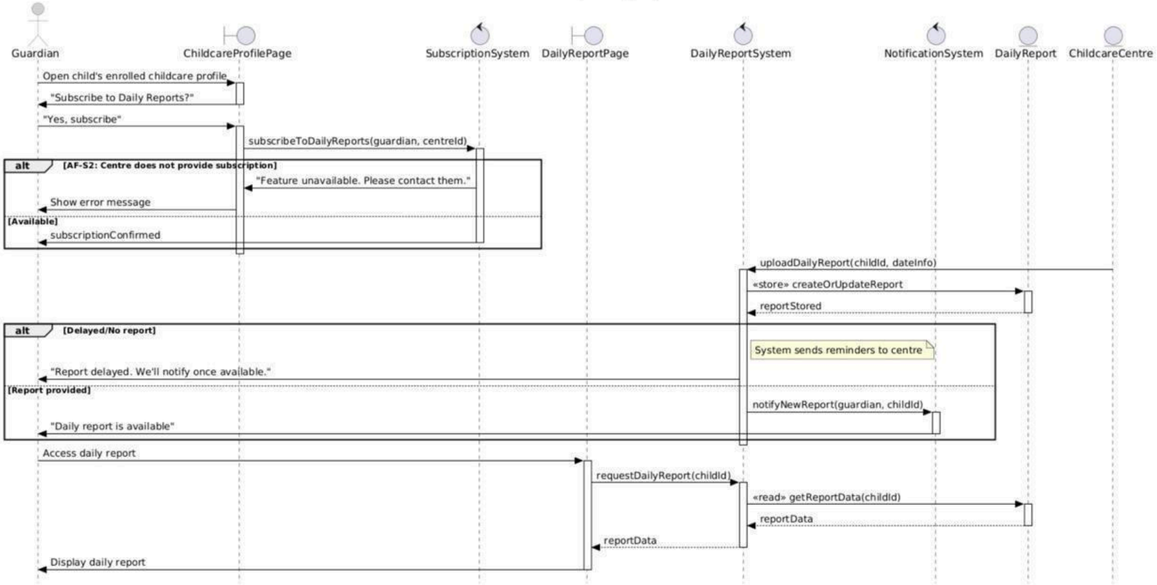
UC-03: Leaving a Review (Only Verified Users)



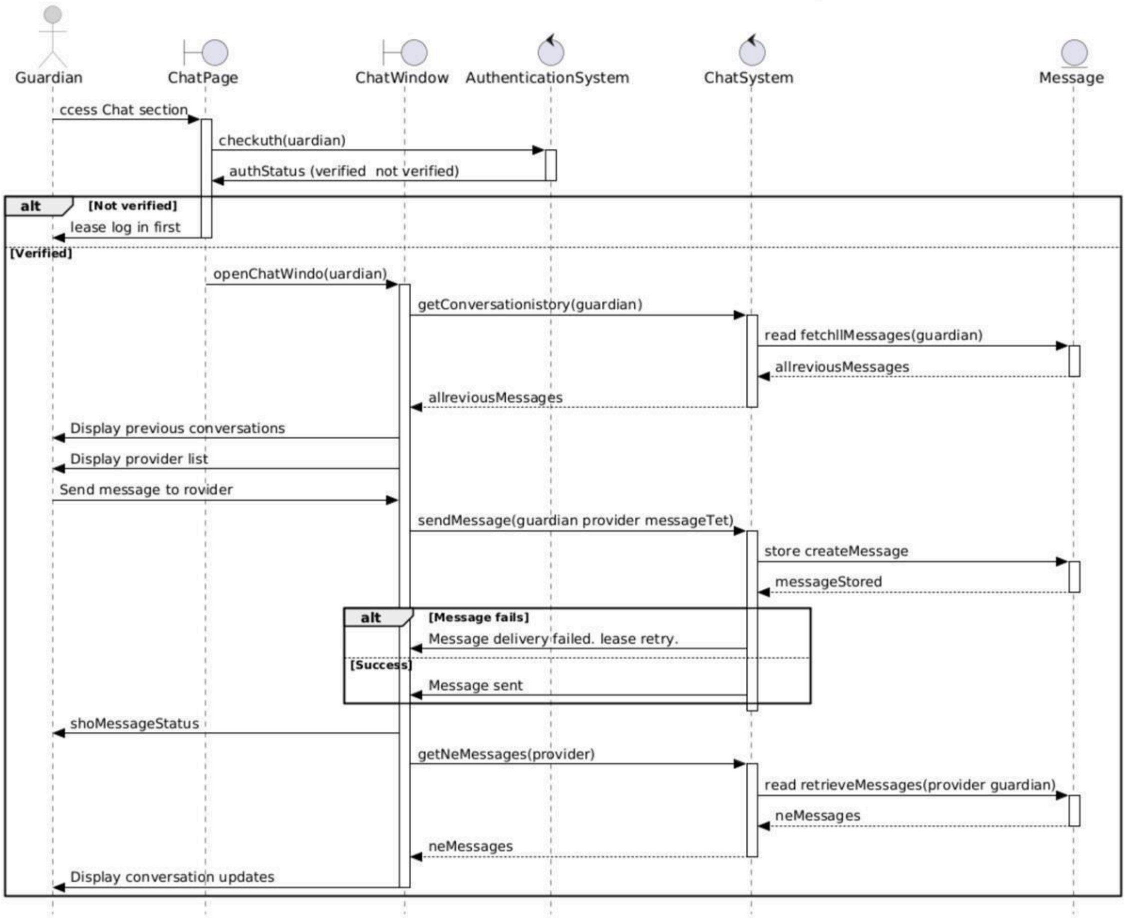
UC-04: Receiving Notifications

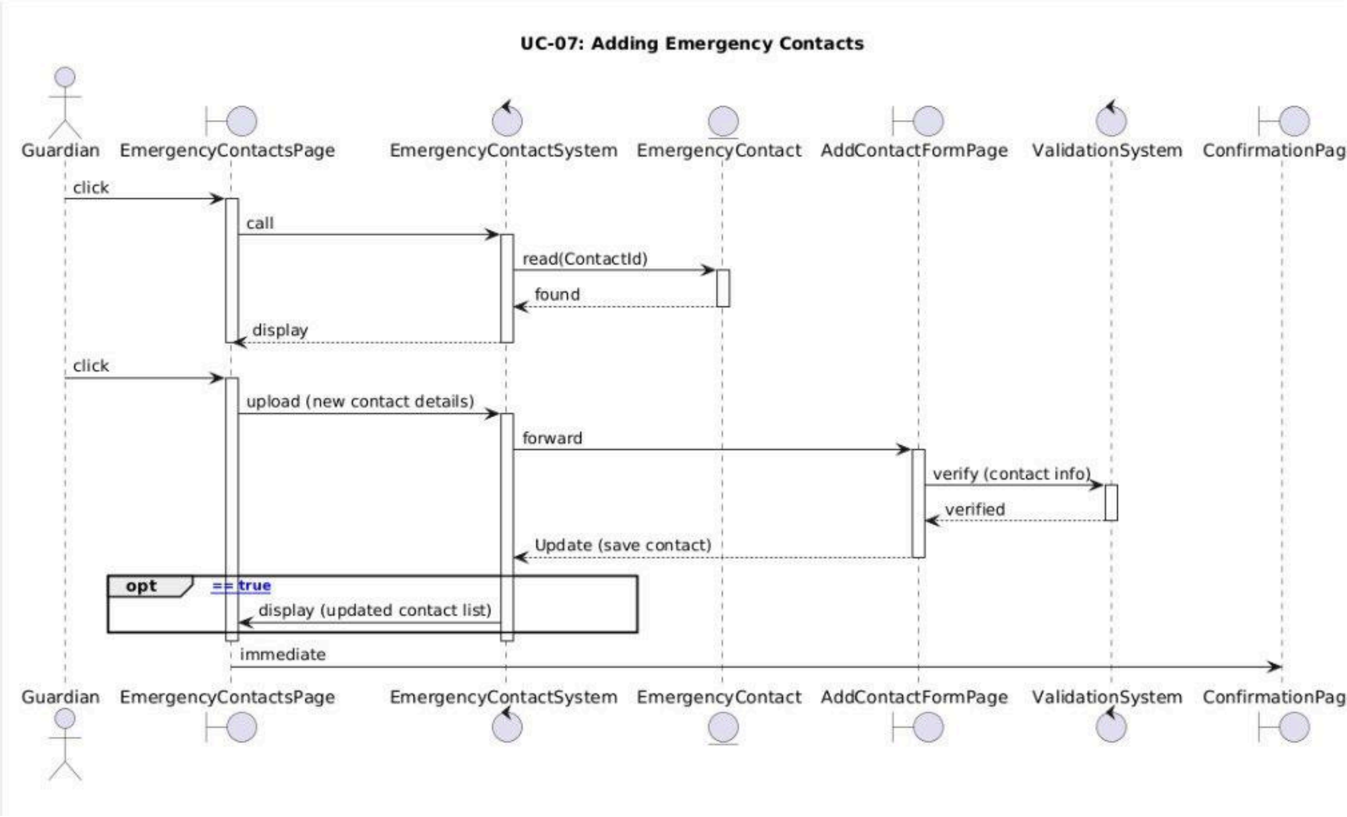

Guardian
NotificationSettingPage
NotificationListPage
NotificationSystem
SubscriptionSystem
Notification
Open settings & customize notifications
«call» updateNotificationPreferences(guardian, prefs)
Preferences saved/updated
View past & new notifications
«read» getNotifications(guardian)
«read» fetchStoredNotifications(guardian)
notificationData
notificationsList
Display notifications
«send» newNotificationEvent(data)
«store» storeNotification(data)
storedOK
alt
[AF-S3: Fail to deliver]
Log failure & re-attempt
[Success]
deliverNotification(alert)

UC-05: Subscribing to Daily Reports


Guardian
ChildcareProfilePage
SubscriptionSystem
DailyReportPage
DailyReportSystem
NotificationSystem
DailyReport
ChildcareCentre
Open child's enrolled childcare profile
"Subscribe to Daily Reports?"
"Yes, subscribe"
subscribeToDailyReports(guardian, centreId)
alt
[AF-S2: Centre does not provide subscription]
"Feature unavailable. Please contact them."
Show error message
[Available]
subscriptionConfirmed
uploadDailyReport(childId, dateInfo)
«store» createOrUpdateReport
reportStored
alt
[Delayed/No report]
System sends reminders to centre
"Report delayed. We'll notify once available."
[Report provided]
notifyNewReport(guardian, childId)
"Daily report is available"
Access daily report
requestDailyReport(childId)
«read» getReportData(childId)
reportData
reportData
Display daily report

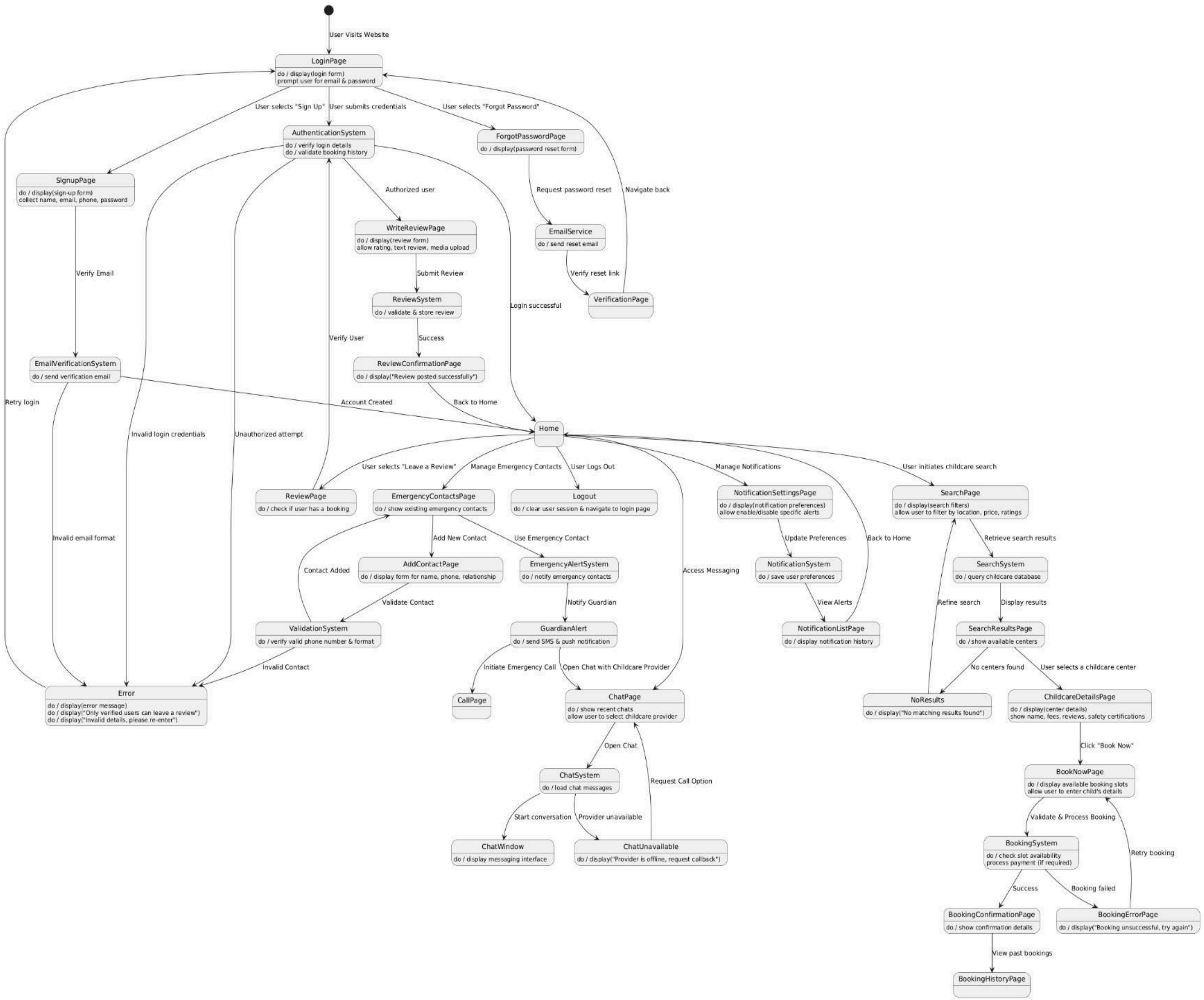
UC-06: Chat Function with Teachers (with Conversation History)




UC-07: Adding Emergency Contacts
Guardian
EmergencyContactsPage
EmergencyContactSystem
EmergencyContact
AddContactFormPage
ValidationSystem
ConfirmationPage
click
call
read(ContactId)
found
display
click
upload (new contact details)
forward
verify (contact info)
verified
Update (save contact)
opt
== true
display (updated contact list)
immediate
Guardian
EmergencyContactsPage
EmergencyContactSystem
EmergencyContact
AddContactFormPage
ValidationSystem
ConfirmationPage

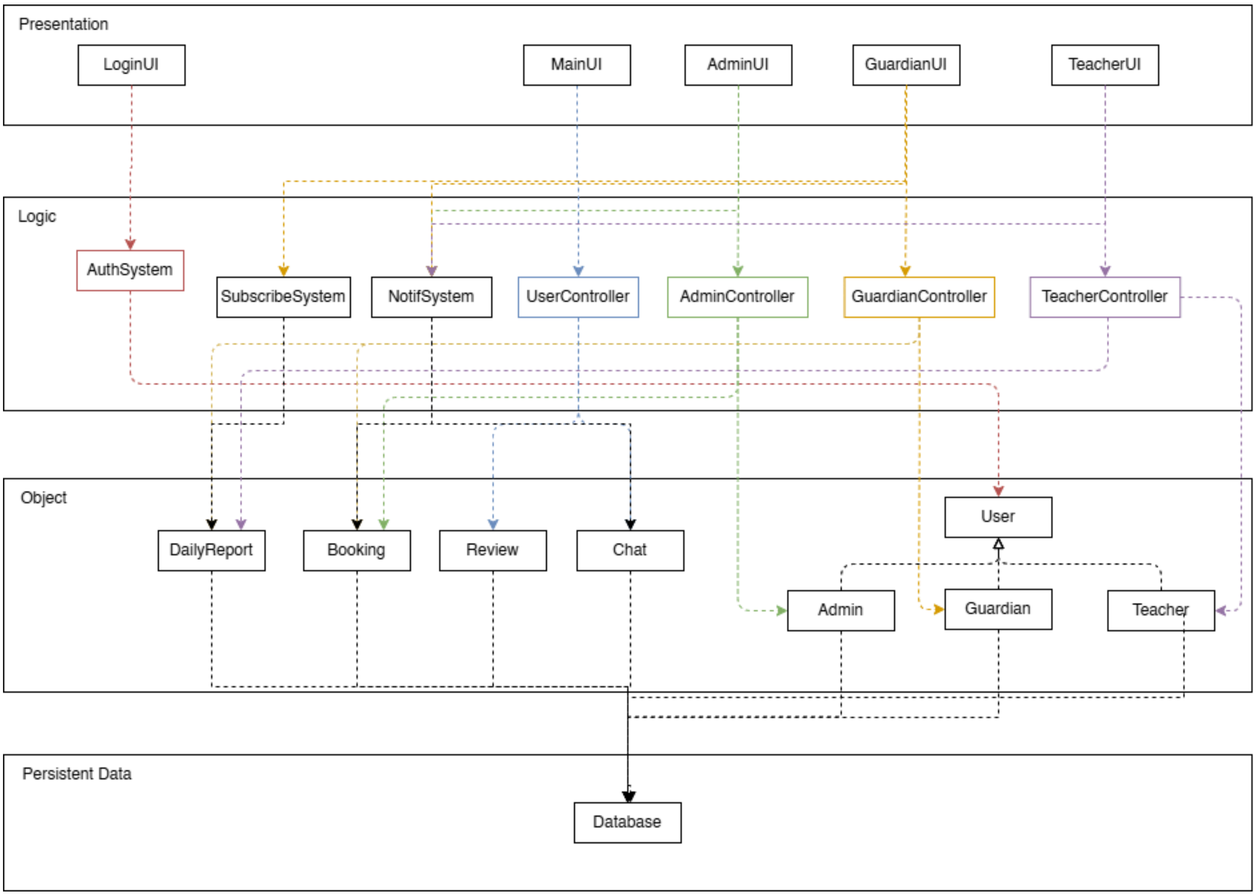
2.3 Initial Dialog Map

Initial Dialog Map for CareScout Website


User Visits Website
LoginPage
do / display(login form)
prompt user for email & password
User selects "Sign Up"
User submits credentials
User selects "Forgot Password"
AuthenticationSystem
do / verify login details
do / validate booking history
ForgotPasswordPage
do / display(password reset form)
SignupPage
do / display(sign-up form)
collect name, email, phone, password
Authorized user
Request password reset
Navigate back
WriteReviewPage
do / display(review form)
allow rating, text review, media upload
EmailService
do / send reset email
Verify Email
Submit Review
Verify reset link
ReviewSystem
do / validate & store review
VerificationPage
Login successful
Verify User
Success
EmailVerificationSystem
do / send verification email
ReviewConfirmationPage
do / display("Review posted successfully")
Retry login
Account Created
Back to Home
Home
Invalid login credentials
Unauthorized attempt
User selects "Leave a Review"
Manage Emergency Contacts
User Logs Out
Manage Notifications
User initiates childcare search
ReviewPage
do / check if user has a booking
EmergencyContactsPage
do / show existing emergency contacts
Logout
do / clear user session & navigate to login page
NotificationSettingsPage
do / display(notification preferences)
allow enable/disable specific alerts
SearchPage
do / display(search filters)
allow user to filter by location, price, ratings
Invalid email format
Add New Contact
Use Emergency Contact
Update Preferences
Back to Home
Retrieve search results
Contact Added
AddContactPage
do / display form for name, phone, relationship
EmergencyAlertSystem
do / notify emergency contacts
Access Messaging
NotificationSystem
do / save user preferences
SearchSystem
do / query childcare database
Validate Contact
Notify Guardian
View Alerts
Refine search
Display results
ValidationSystem
do / verify valid phone number & format
GuardianAlert
do / send SMS & push notification
NotificationListPage
do / display notification history
SearchResultsPage
do / show available centers
Invalid Contact
Initiate Emergency Call
Open Chat with Childcare Provider
No centers found
User selects a childcare center
Error
do / display(error message)
do / display("Only verified users can leave a review")
do / display("Invalid details, please re-enter")
CallPage
ChatPage
do / show recent chats
allow user to select childcare provider
NoResults
do / display("No matching results found")
ChildcareDetailsPage
do / display(center details)
show name, fees, reviews, safety certifications
Open Chat
Click "Book Now"
Request Call Option
ChatSystem
do / load chat messages
BookNowPage
do / display available booking slots
allow user to enter child's details
Start conversation
Provider unavailable
Validate & Process Booking
ChatWindow
do / display messaging interface
ChatUnavailable
do / display("Provider is offline, request callback")
BookingSystem
do / check slot availability
process payment (if required)
Retry booking
Success
Booking failed
BookingConfirmationPage
do / show confirmation details
BookingErrorPage
do / display("Booking unsuccessful, try again")
View past bookings
BookingHistoryPage

3. System Architecture

The architecture of *Care Scout* follows a **layered design pattern**, promoting modularity, maintainability, and separation of concerns. It is composed of four primary layers:



3.1. Presentation Layer

This layer handles user interactions and consists of UI components tailored for each user role. Examples include:

- LoginUI, MainUI – Shared navigation and login access points.
- GuardianUI – Interfaces for booking, reviewing, and managing emergency contacts.
- TeacherUI – Used to upload daily reports and respond to chats.
- AdminUI – For managing system-level users and data if needed.

These UI components route user actions to the appropriate system controllers in the Logic Layer.

3.2. Logic Layer

This layer contains the core application logic. It includes:

- Controllers such as AuthController, BookingController, NotificationController, ChatController, etc.
- Each controller coordinates between UI actions and backend processing (data fetching, validation, business rules).

This layer abstracts the complexity of the Object Layer and ensures business logic is centralized and reusable.

3.3. Object Layer

This layer contains the entity classes that represent the core domain of the application, such as:

- User (with roles like Guardian, ChildcareProvider, Admin)
- Booking, DailyReport, EmergencyContact, Review, ChatMessage

These classes encapsulate application data and are manipulated by the Logic Layer.

3.4. Persistent Data Layer

This is the database layer responsible for storing and retrieving persistent data. It includes all entity tables and their relationships:

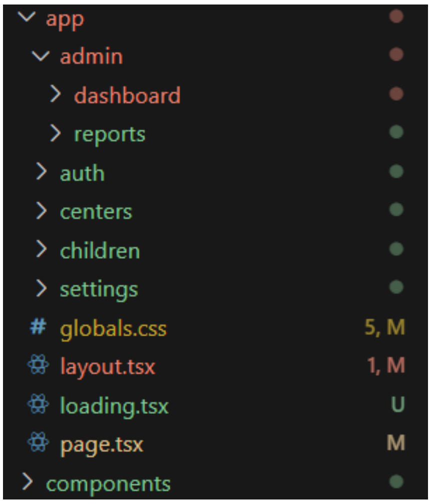
- Guardians, Bookings, Childcare Centres, Reviews, Emergency Contacts, Reports, Messages

This layer is accessed via ORM (Object Relational Mapping) or direct queries, depending on the implementation.

4. Application Skeleton

Please refer to the source code uploaded in the github repository for the application skeleton

4.1 Frontend



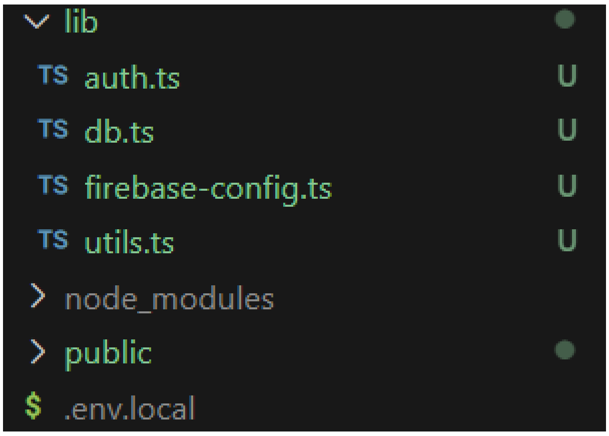
```
∨ app                  ●
  ∨ admin              ●
    > dashboard        ●
    > reports          ●
  > auth               ●
  > centers            ●
  > children           ●
  > settings           ●
  # globals.css     5, M
    layout.tsx      1, M
    loading.tsx        U
    page.tsx           M
> components           ●
```

The frontend for the web app is built in React.js, with user interfaces grouped by feature areas: Admin handles dashboards and reports; Auth covers login, registration, and role-based access; Centers manages center details like address and capacity; Children handles child profiles, attendance, and activities; and Settings provides system preferences and account configuration. Each feature area contains sub-screens or sub-components, keeping the UI organized.

An App.tsx or page.tsx file serves as the main entry point, setting up global providers and routing. A central globals.css file maintains a consistent design, and a layout.tsx outlines shared elements such as headers, footers, and side navigation. Loading.tsx manages loading states and user feedback during data retrieval or routing transitions.

Common UI elements (e.g., buttons, modals, form inputs) reside in a components folder, while global states (like authentication or theme preferences) are managed via React Context in a contexts folder. This modular approach keeps the application maintainable, scalable, and easier to collaborate on.

4.2 Backend


```
lib
  auth.ts               U
  db.ts                 U
  firebase-config.ts    U
  utils.ts              U
node_modules
public
.env.local
```

The childcare center’s backend is organized into distinct modules.

The **Models** directory contains business objects representing the core data structures.

Services directory holds logic for accessing, modifying, or manipulating those models.

Controllers act as an abstraction layer, interfacing between the Services.

Routers defines the REST API endpoints for communicating with the frontend. A dedicated file (e.g., database.py) manages the connection to the database that stores all of the entities, ensuring data consistency across the application.

Additionally, the backend integrates a Firebase component that handles user authentication, real-time data synchronization, and cloud messaging. This integration enhances security and scalability while providing streamlined management of user sessions and notifications.

5. Appendix

5.1 Persistent Data Design

Database Overview
Carescout uses **Firebase Firestore**, a NoSQL cloud-based database that organizes data into collections and documents instead of traditional relational tables.

Key Collections and Fields

- **users**: Stores user details such as UID, name, email, and role (guardian, provider, or admin).

- **childcare_centres**: Contains data like name, location, fees, education levels offered, and slot availability.

- **bookings**: Tracks reservations with details including booking ID, guardian ID, centre ID, date, and status.

- **reviews**: Includes guardian reviews linked to childcare centres, with a rating and written feedback.

- **daily_reports**: Used by providers to upload daily summaries about children, including notes and media attachments.

- **emergency_contacts**: Stores names, phone numbers, and relationships of alternate contacts for each guardian.

- **messages**: Supports chat between guardians and providers with sender, receiver, timestamps, and message content.

Data Access
All reads and writes are handled securely using Firebase SDKs. Firebase Security Rules restrict access based on the user’s authentication state and role.

5.2 Access Control

User Roles and Permissions

- **Guardians** can search for centres, make and manage bookings, leave reviews, subscribe to daily reports, use the chat function, and manage emergency contacts.

- **Childcare Providers** can upload daily reports, chat with guardians, and view bookings relevant to their centre.

- **Admins** have elevated privileges and can monitor and manage all users and data in the system.

Authentication

Authentication is managed via Firebase Authentication, which supports secure email/password sign-ins. Upon login, the system retrieves the user's role from the users collection.

Authorization

- In the frontend, pages and components are shown or hidden based on the user's role.

- In the backend (API routes), each request is validated using Firebase ID tokens.

- Firestore Security Rules are used to restrict data access, ensuring users can only read and write the documents they own or are allowed to access.

5.3 Tech Stack

Frontend Technologies

- **Next.js** is used for server-side rendering and page routing.

- **React** enables a dynamic and component-based UI.

- **TypeScript** ensures type safety in the codebase.

- **Tailwind CSS** is used for utility-first styling.

- **shadcn/ui** provides accessible and customizable UI components.

- **PostCSS** enables transformation of CSS with modern tooling.

Backend Technologies

- **Next.js API Routes** are used to define server-side logic within the same codebase.
- **Node.js** serves as the runtime environment for executing backend code.

Database and Storage

- **Firebase Firestore** is used for storing structured data across collections.
- **Firebase Auth** handles user authentication.
- **Firebase Storage** is used to store uploaded images and files (e.g., daily reports).