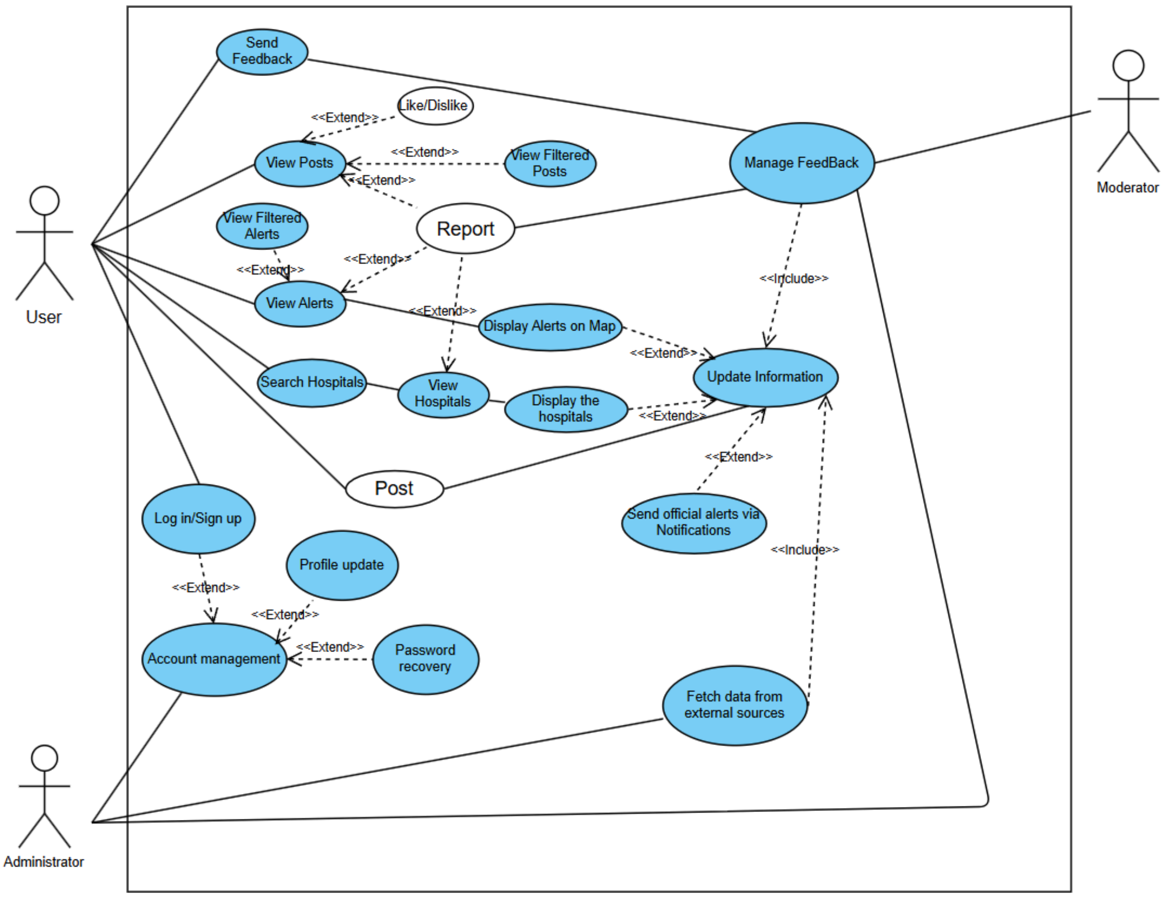
NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

SC2006 - Software Engineering Lab 3 Deliverables

Lab Group	SCEB
Team	Team 1
Members	AKANKSHA MATHUR (U2323265C)
	HU HAN (U2320036H)
	PHUA NAOMI (U2422699H)
	KARTHIK RAJ S/O MOHAN(U2320917J)
	LI YUJIA (U2320574C)
	WANG KE (U2320276E)

Table of Contents

1. Use Case Model

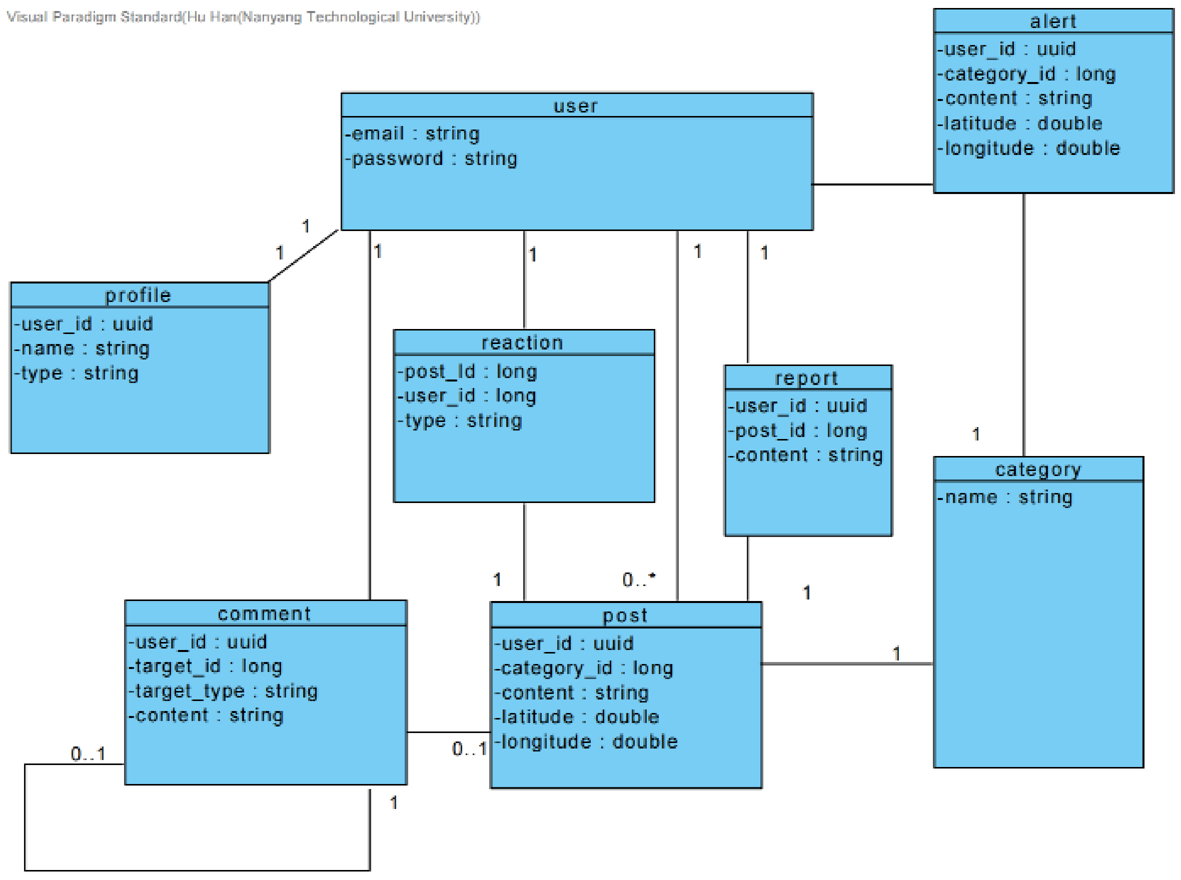


2. Design Model

2.1 Class Diagram

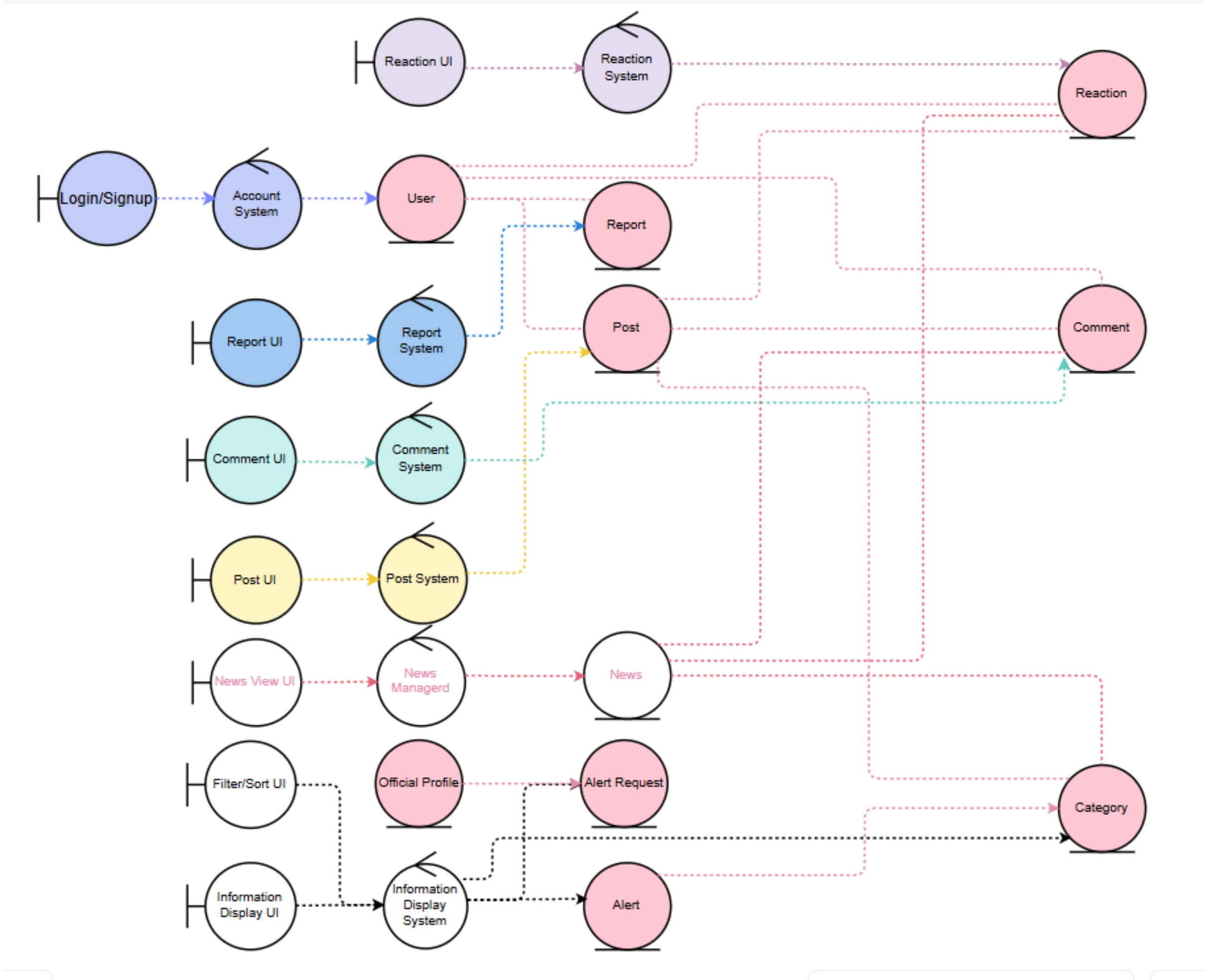
2.1.1 Entity Classes

These entity classes are mapped to the tables in the database, with foreign keys defining associations between them.



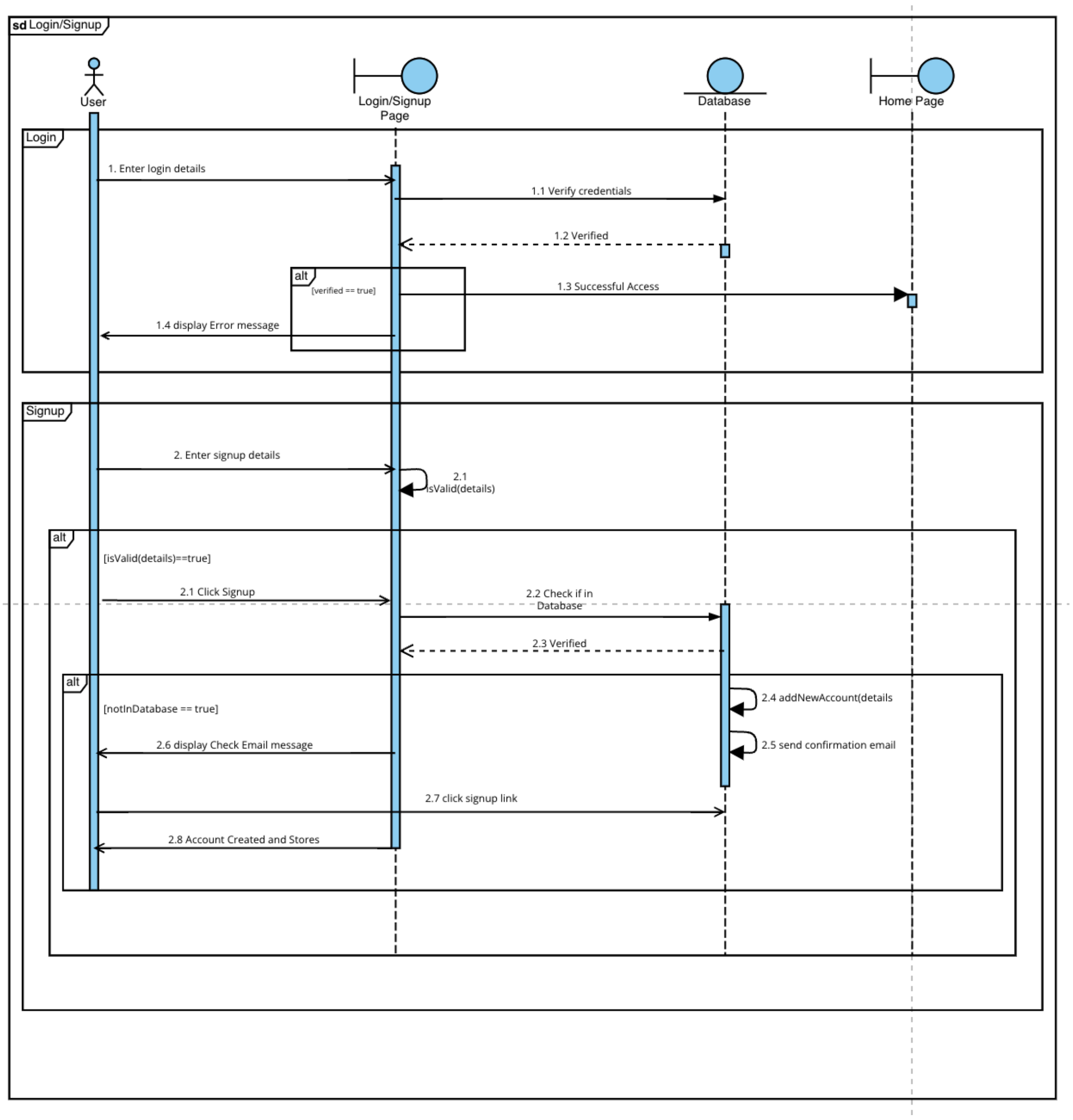
2.1.2 Boundary & Control Classes

The boundary classes refer to the UI components that provide direct interaction to users. The control classes act as services that handle user requests, perform operations on entity objects, and return responses to be displayed on the UI.

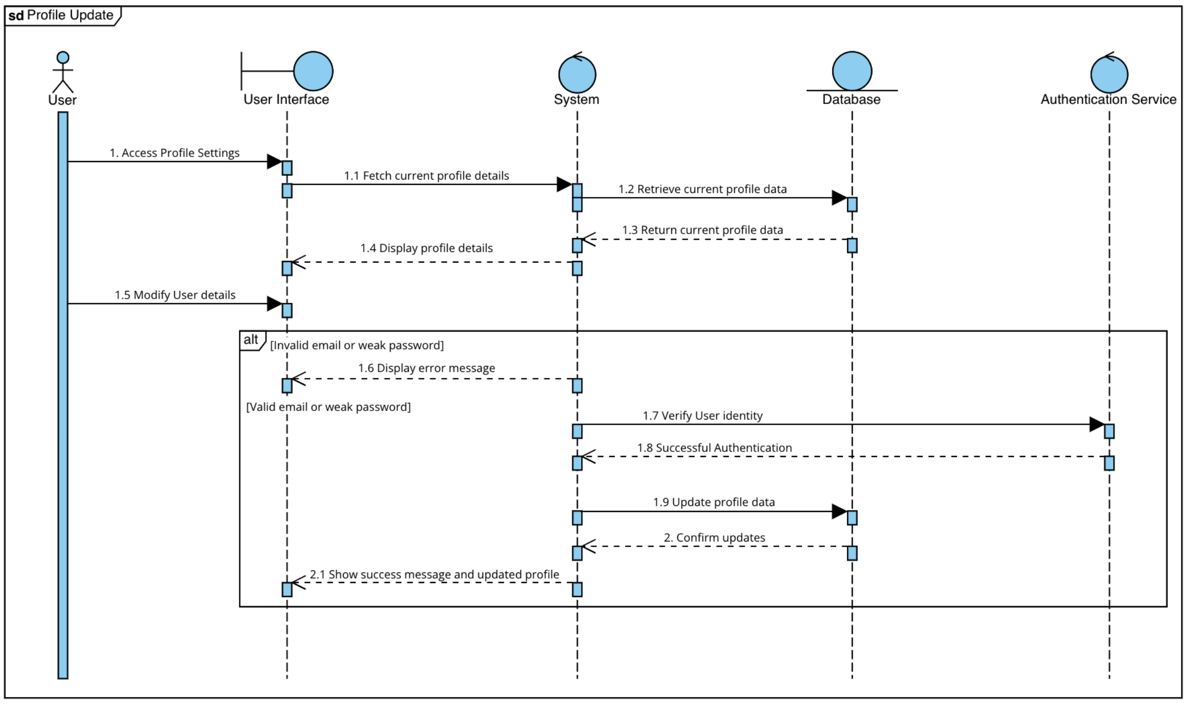

Reaction UI
Reaction System
Reaction
Login/Signup
Account System
User
Report
Report UI
Report System
Post
Comment
Comment UI
Comment System
Post UI
Post System
News View UI
News Managerd
News
Filter/Sort UI
Official Profile
Alert Request
Category
Information Display UI
Information Display System
Alert

2.2 Sequence Diagrams

User Case 1

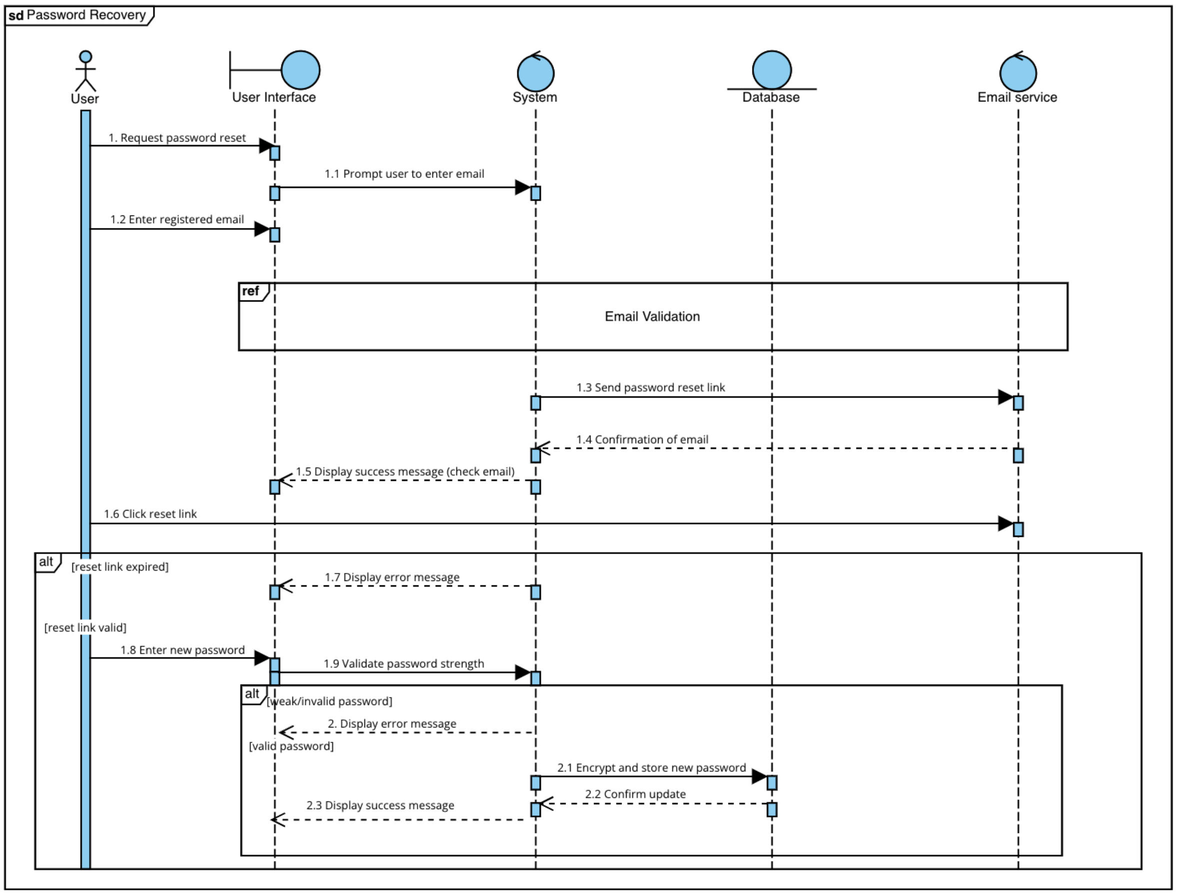


User Case 2



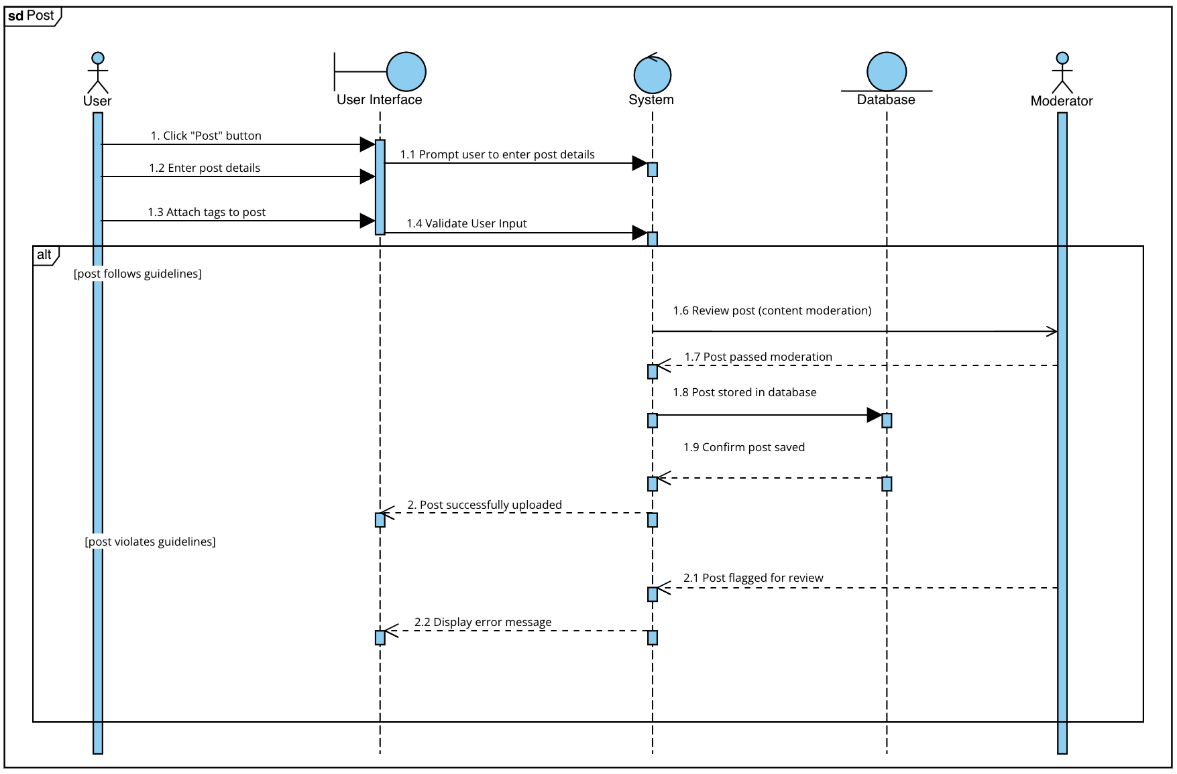
sd Profile Update
User
User Interface
System
Database
Authentication Service
1. Access Profile Settings
1.1 Fetch current profile details
1.2 Retrieve current profile data
1.3 Return current profile data
1.4 Display profile details
1.5 Modify User details
alt
[Invalid email or weak password]
1.6 Display error message
[Valid email or weak password]
1.7 Verify User identity
1.8 Successful Authentication
1.9 Update profile data
2. Confirm updates
2.1 Show success message and updated profile

User Case 3



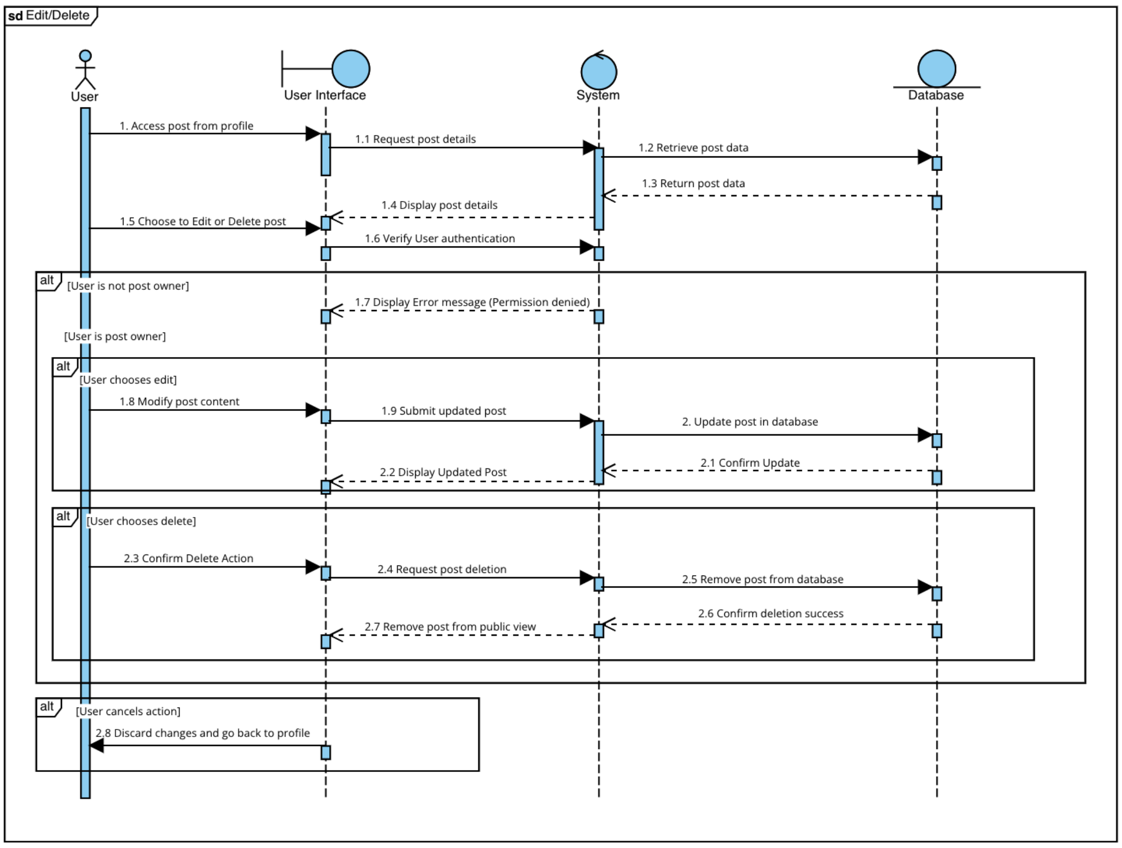
sd Password Recovery
User
User Interface
System
Database
Email service
1. Request password reset
1.1 Prompt user to enter email
1.2 Enter registered email
ref
Email Validation
1.3 Send password reset link
1.4 Confirmation of email
1.5 Display success message (check email)
1.6 Click reset link
alt
[reset link expired]
1.7 Display error message
[reset link valid]
1.8 Enter new password
1.9 Validate password strength
alt
[weak/invalid password]
2. Display error message
[valid password]
2.1 Encrypt and store new password
2.2 Confirm update
2.3 Display success message

User Case 4



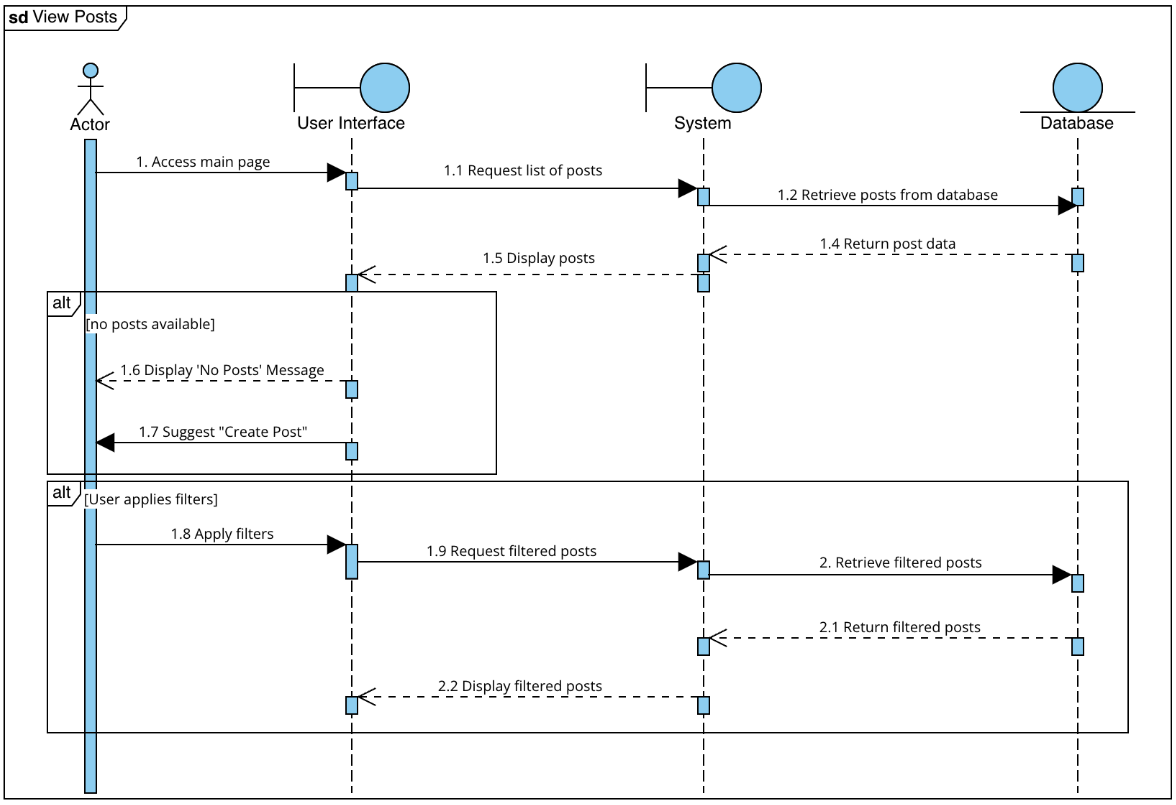
sd Post
User
User Interface
System
Database
Moderator
1. Click "Post" button
1.1 Prompt user to enter post details
1.2 Enter post details
1.3 Attach tags to post
1.4 Validate User Input
alt
[post follows guidelines]
1.6 Review post (content moderation)
1.7 Post passed moderation
1.8 Post stored in database
1.9 Confirm post saved
2. Post successfully uploaded
[post violates guidelines]
2.1 Post flagged for review
2.2 Display error message

User Case 5

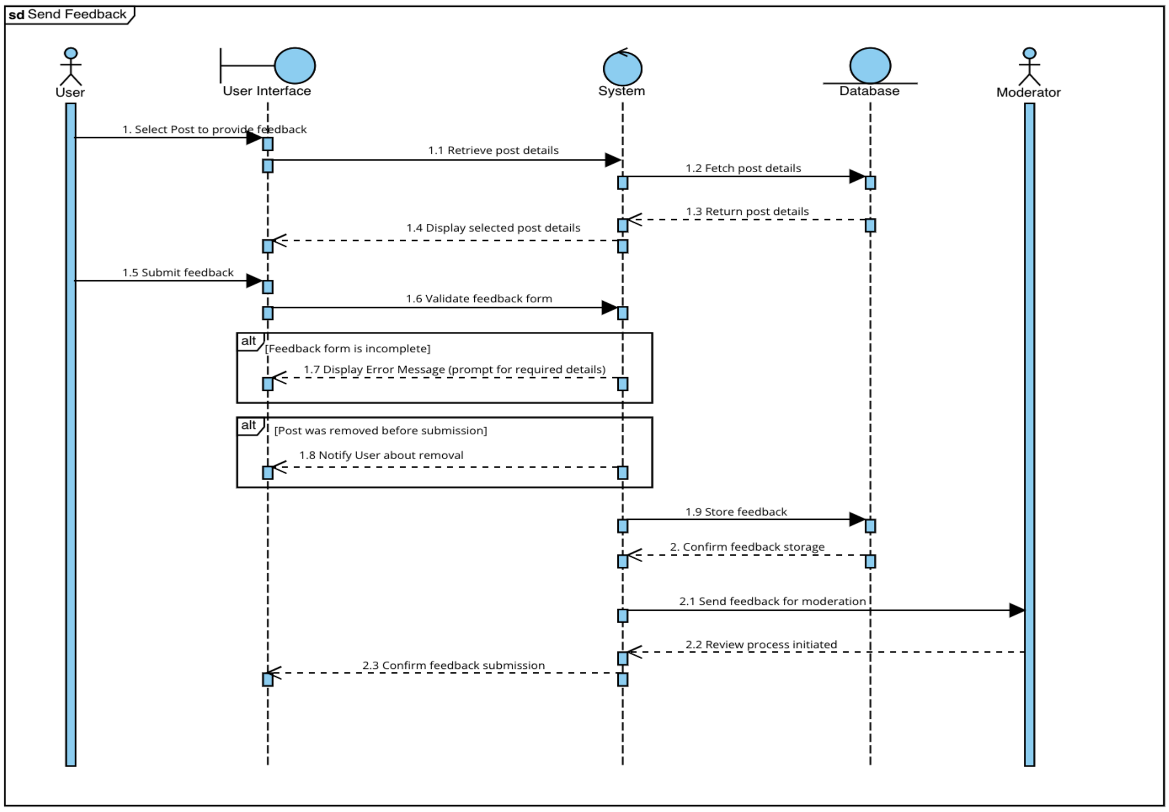


sd Edit/Delete
User
User Interface
System
Database
1. Access post from profile
1.1 Request post details
1.2 Retrieve post data
1.3 Return post data
1.4 Display post details
1.5 Choose to Edit or Delete post
1.6 Verify User authentication
alt
[User is not post owner]
1.7 Display Error message (Permission denied)
[User is post owner]
alt
[User chooses edit]
1.8 Modify post content
1.9 Submit updated post
2. Update post in database
2.1 Confirm Update
2.2 Display Updated Post
alt
[User chooses delete]
2.3 Confirm Delete Action
2.4 Request post deletion
2.5 Remove post from database
2.6 Confirm deletion success
2.7 Remove post from public view
alt
[User cancels action]
2.8 Discard changes and go back to profile

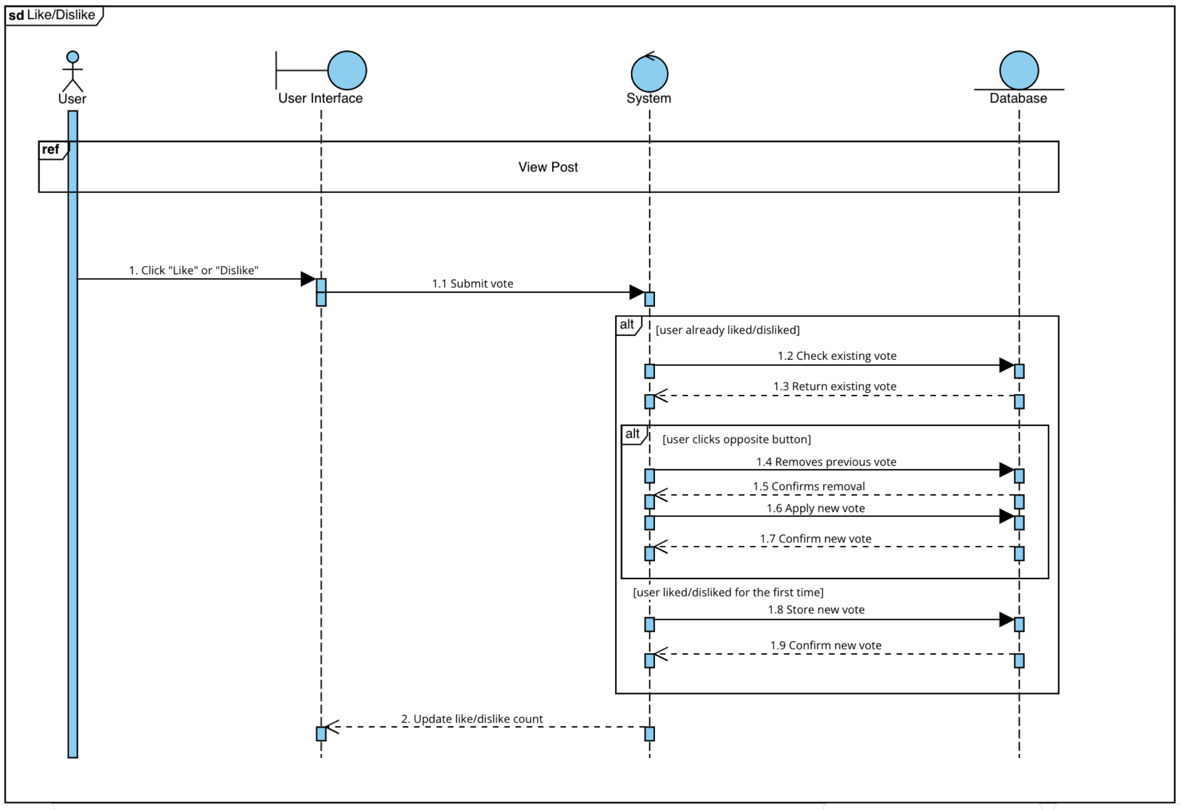
User Case 6


sd View Posts
Actor
User Interface
System
Database
1. Access main page
1.1 Request list of posts
1.2 Retrieve posts from database
1.4 Return post data
1.5 Display posts
alt
[no posts available]
1.6 Display 'No Posts' Message
1.7 Suggest "Create Post"
alt
[User applies filters]
1.8 Apply filters
1.9 Request filtered posts
2. Retrieve filtered posts
2.1 Return filtered posts
2.2 Display filtered posts

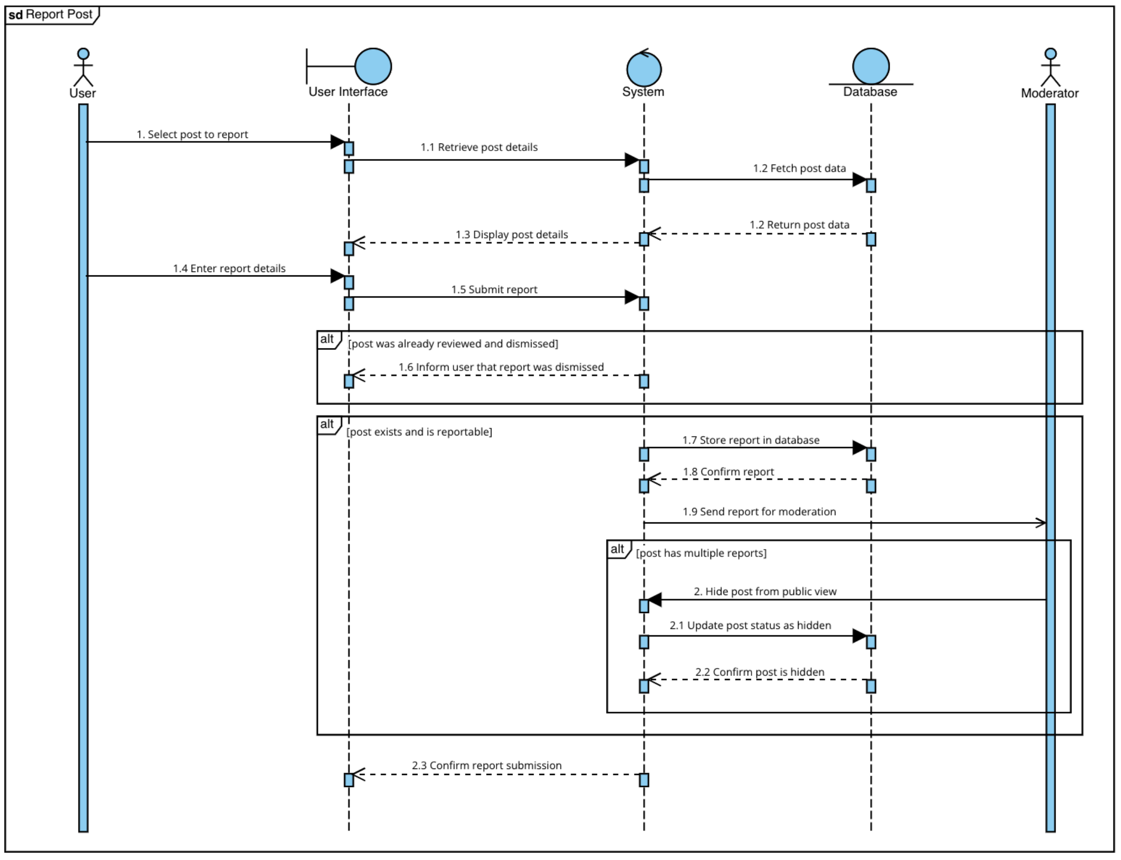
User Case 7


sd Send Feedback
User
User Interface
System
Database
Moderator
1. Select Post to provide feedback
1.1 Retrieve post details
1.2 Fetch post details
1.3 Return post details
1.4 Display selected post details
1.5 Submit feedback
1.6 Validate feedback form
alt
[Feedback form is incomplete]
1.7 Display Error Message (prompt for required details)
alt
[Post was removed before submission]
1.8 Notify User about removal
1.9 Store feedback
2. Confirm feedback storage
2.1 Send feedback for moderation
2.2 Review process initiated
2.3 Confirm feedback submission

User Case 8

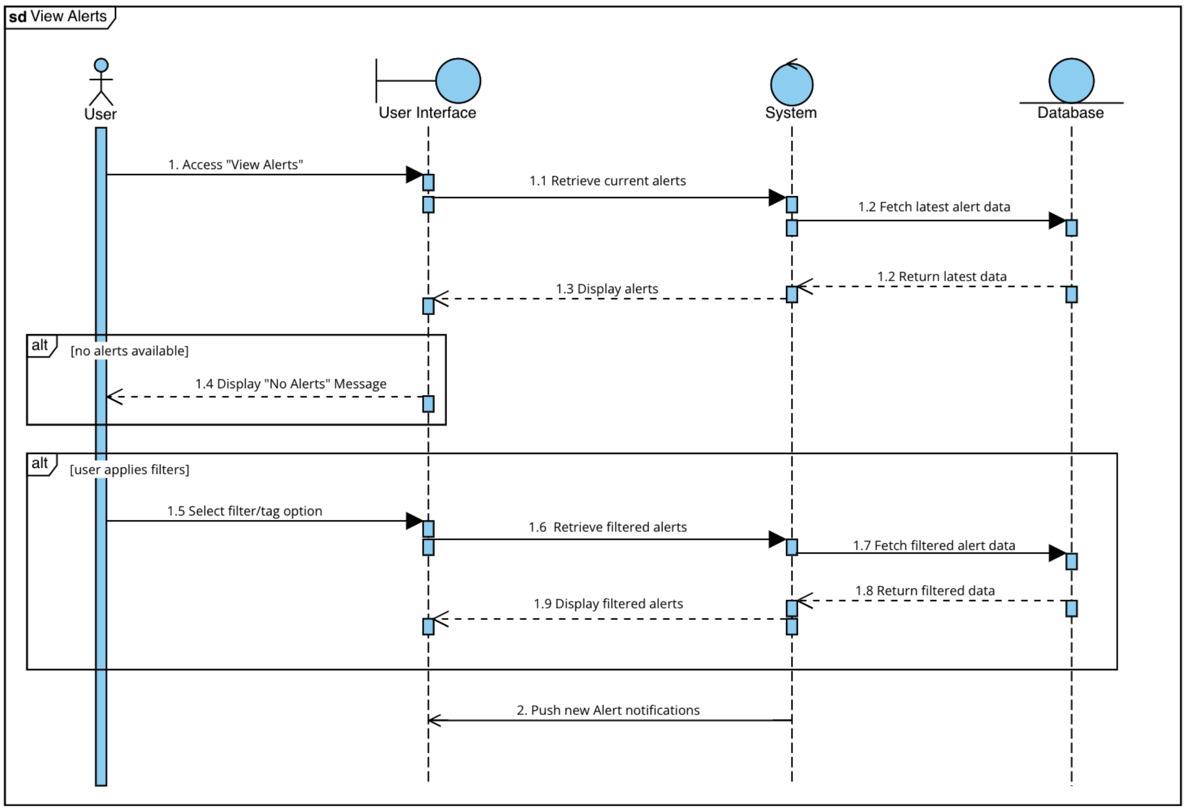


User Case 9

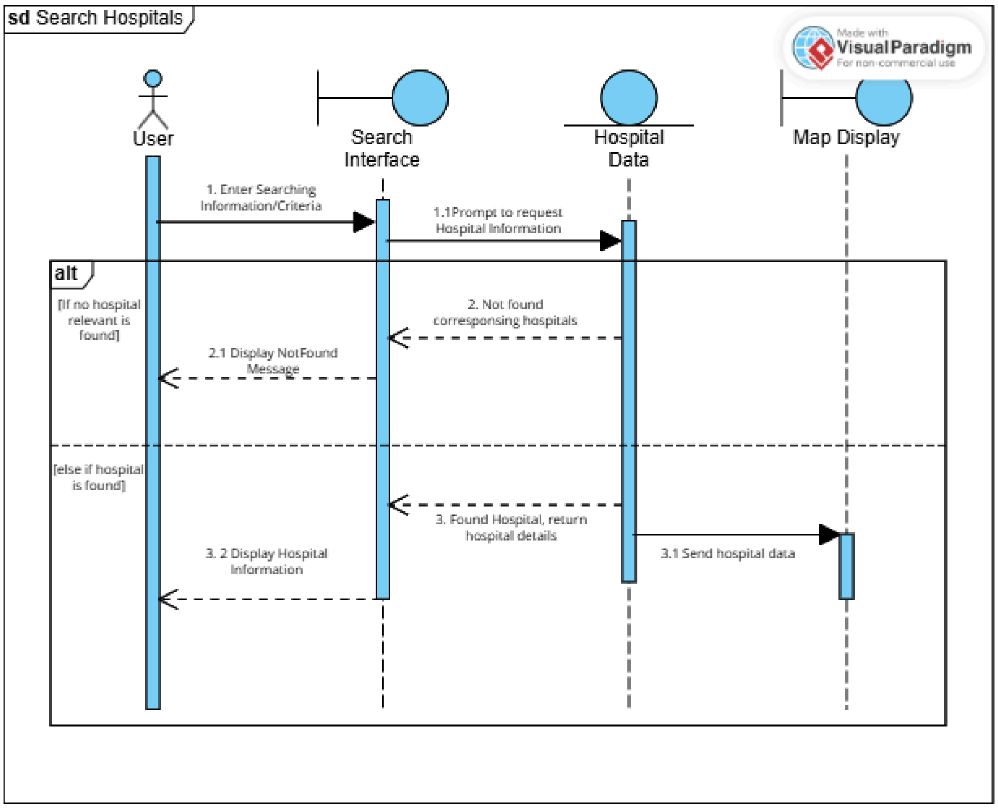


sd Report Post
User
User Interface
System
Database
Moderator
1. Select post to report
1.1 Retrieve post details
1.2 Fetch post data
1.2 Return post data
1.3 Display post details
1.4 Enter report details
1.5 Submit report
alt
[post was already reviewed and dismissed]
1.6 Inform user that report was dismissed
alt
[post exists and is reportable]
1.7 Store report in database
1.8 Confirm report
1.9 Send report for moderation
alt
[post has multiple reports]
2. Hide post from public view
2.1 Update post status as hidden
2.2 Confirm post is hidden
2.3 Confirm report submission

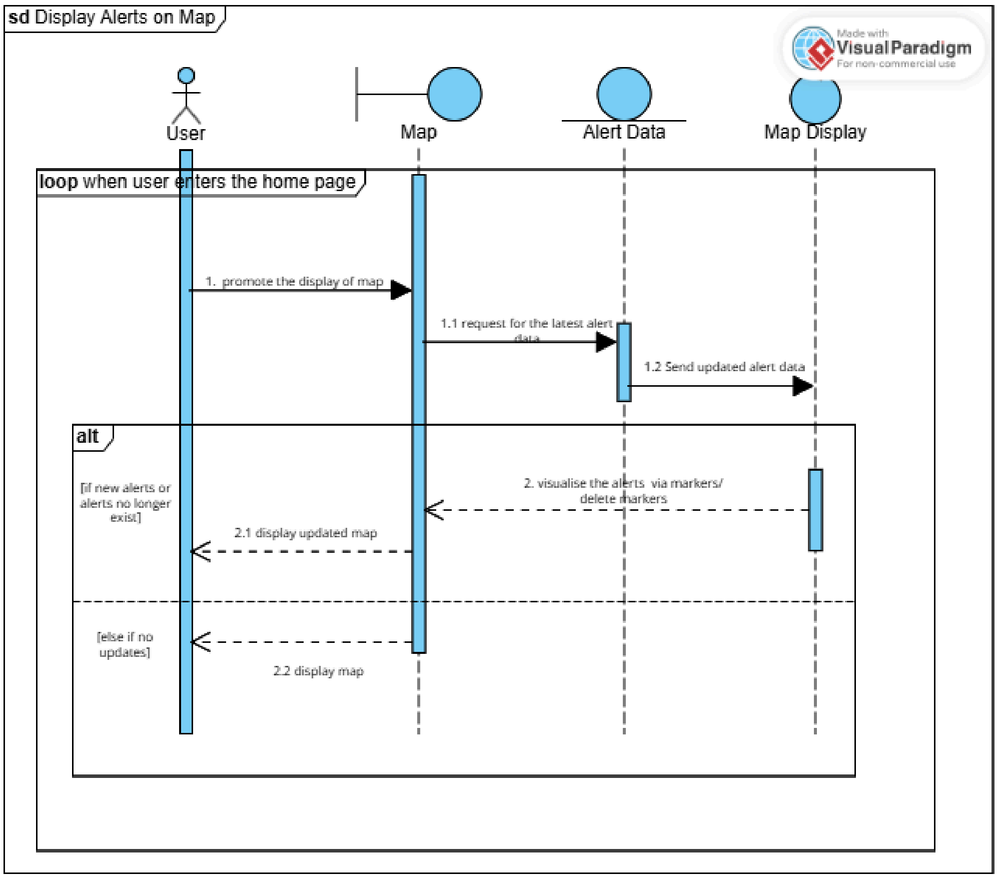
User Case 10


sd View Alerts
User
User Interface
System
Database
1. Access "View Alerts"
1.1 Retrieve current alerts
1.2 Fetch latest alert data
1.2 Return latest data
1.3 Display alerts
alt
[no alerts available]
1.4 Display "No Alerts" Message
alt
[user applies filters]
1.5 Select filter/tag option
1.6 Retrieve filtered alerts
1.7 Fetch filtered alert data
1.8 Return filtered data
1.9 Display filtered alerts
2. Push new Alert notifications

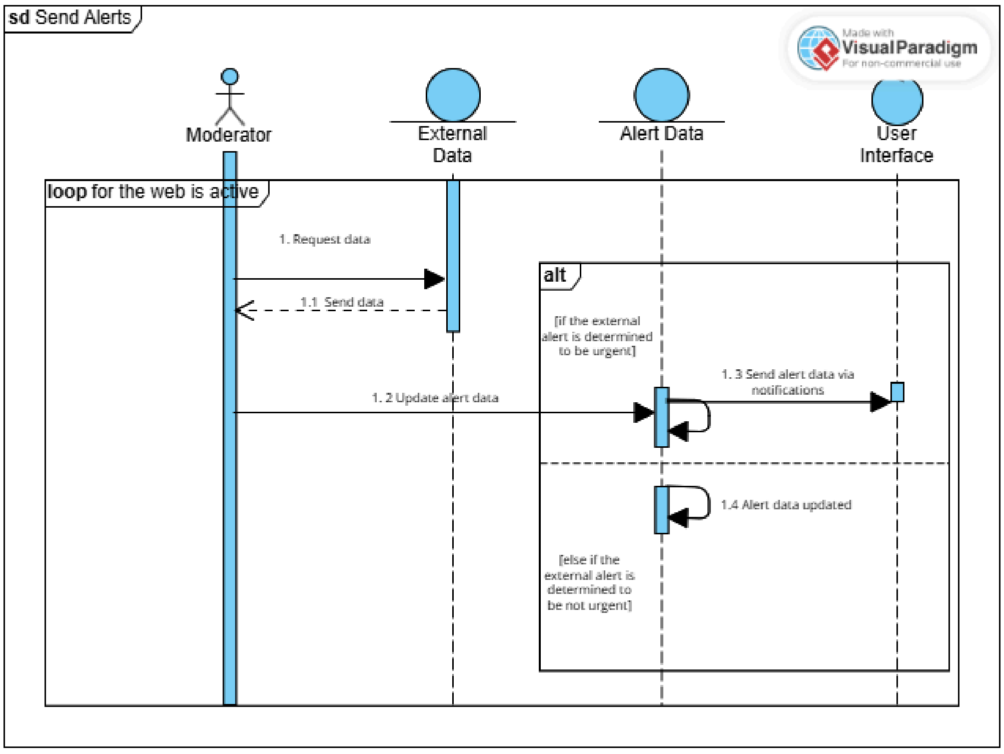
User case 11


sd Search Hospitals
Made with Visual Paradigm For non-commercial use
User
Search Interface
Hospital Data
Map Display
1. Enter Searching Information/Criteria
1.1Prompt to request Hospital Information
alt
[If no hospital relevant is found]
2. Not found corresponsing hospitals
2.1 Display NotFound Message
[else if hospital is found]
3. Found Hospital, return hospital details
3.1 Send hospital data
3. 2 Display Hospital Information

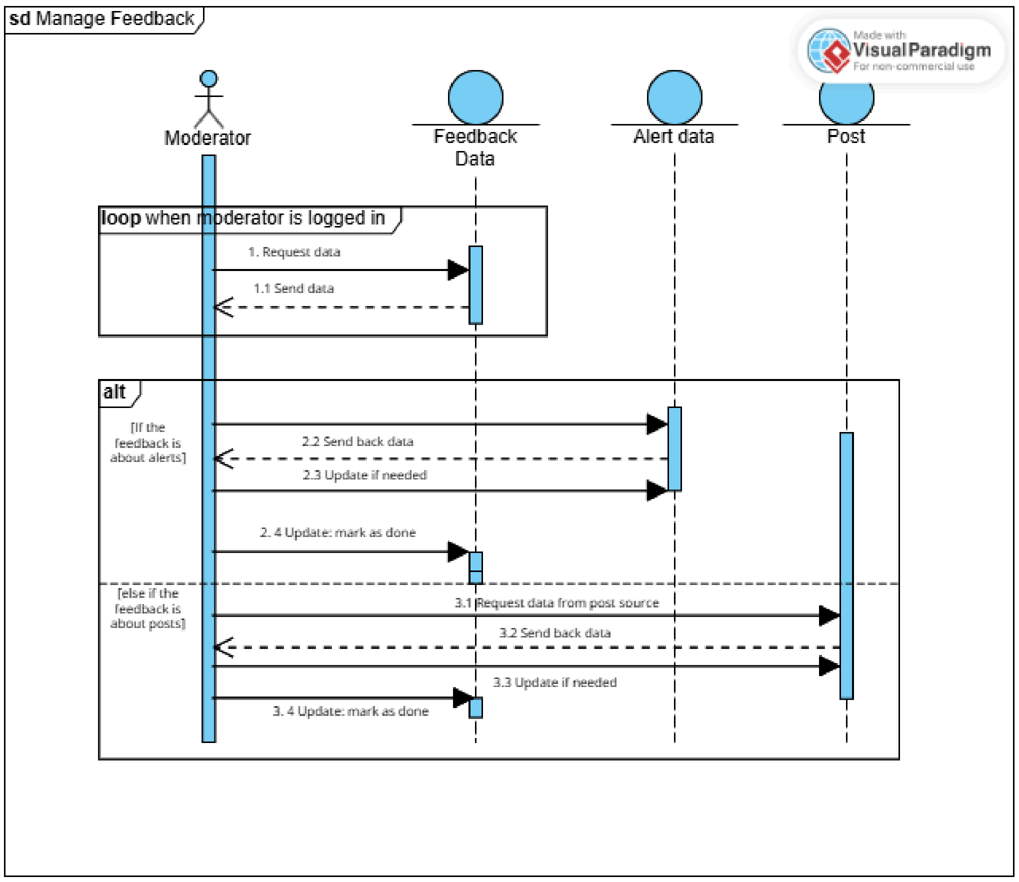
User case 12


sd Display Alerts on Map
User
Map
Alert Data
Map Display
loop when user enters the home page
1. promote the display of map
1.1 request for the latest alert data
1.2 Send updated alert data
alt
[if new alerts or alerts no longer exist]
2. visualise the alerts via markers/ delete markers
2.1 display updated map
[else if no updates]
2.2 display map

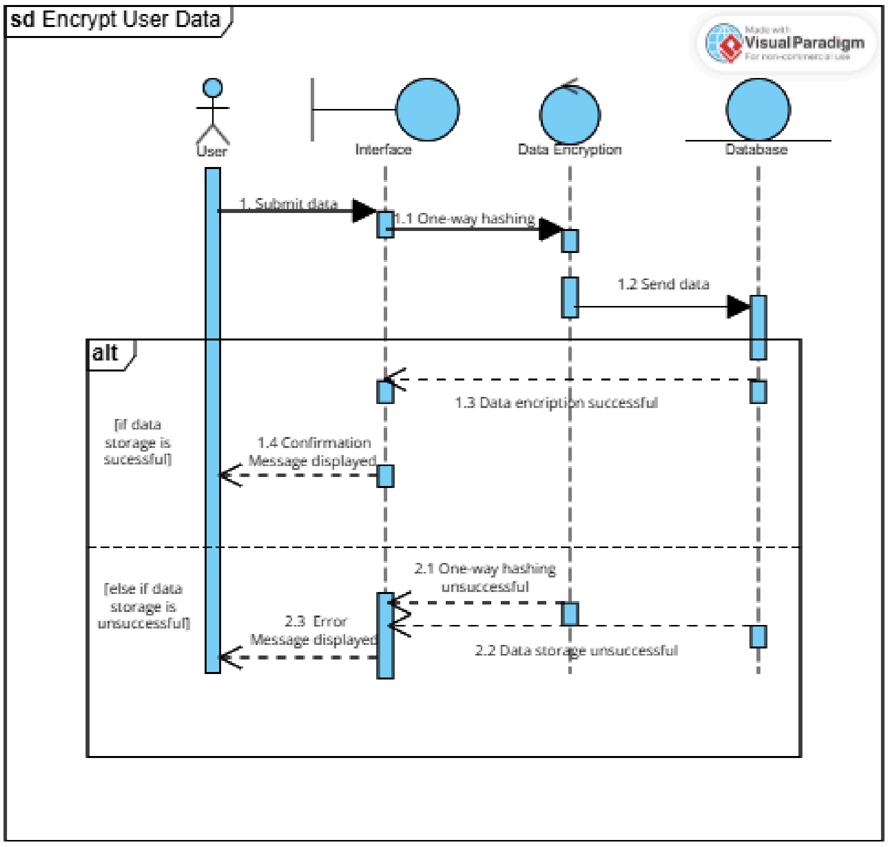
User case 13


sd Send Alerts
Moderator
External Data
Alert Data
User Interface
loop for the web is active
1. Request data
1.1 Send data
1. 2 Update alert data
alt
[if the external alert is determined to be urgent]
1. 3 Send alert data via notifications
1.4 Alert data updated
[else if the external alert is determined to be not urgent]

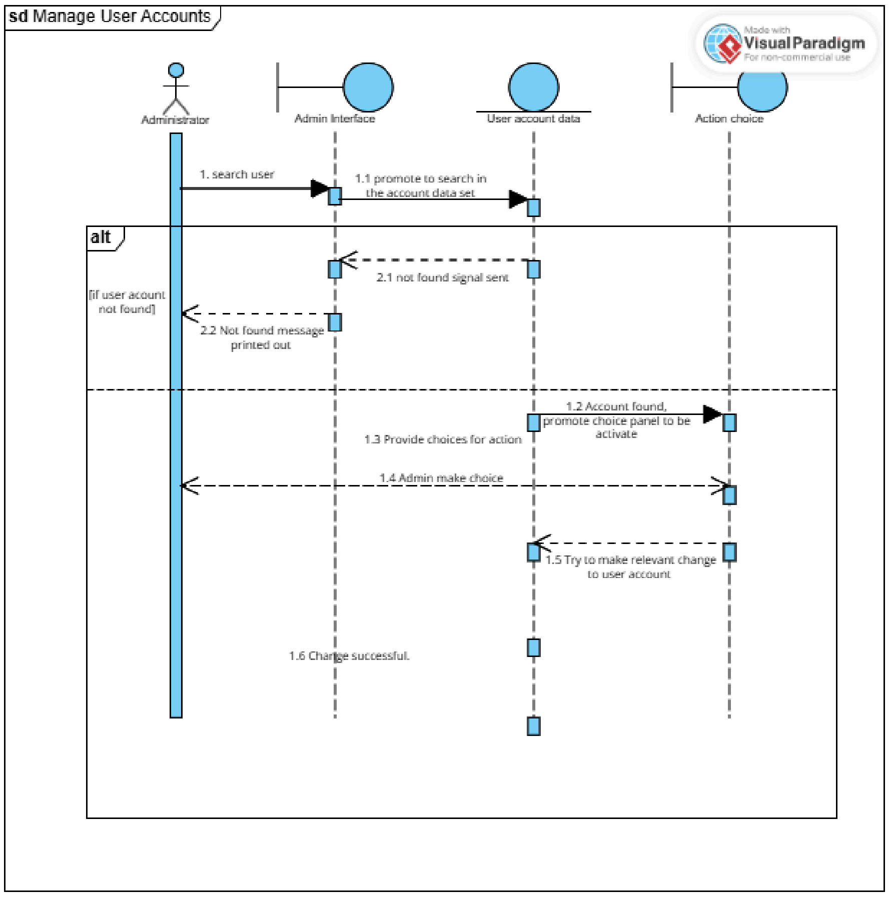
User case 14


sd Manage Feedback
Made with Visual Paradigm For non-commercial use
Moderator
Feedback Data
Alert data
Post
loop when moderator is logged in
1. Request data
1.1 Send data
alt
[If the feedback is about alerts]
2.2 Send back data
2.3 Update if needed
2. 4 Update: mark as done
[else if the feedback is about posts]
3.1 Request data from post source
3.2 Send back data
3.3 Update if needed
3. 4 Update: mark as done

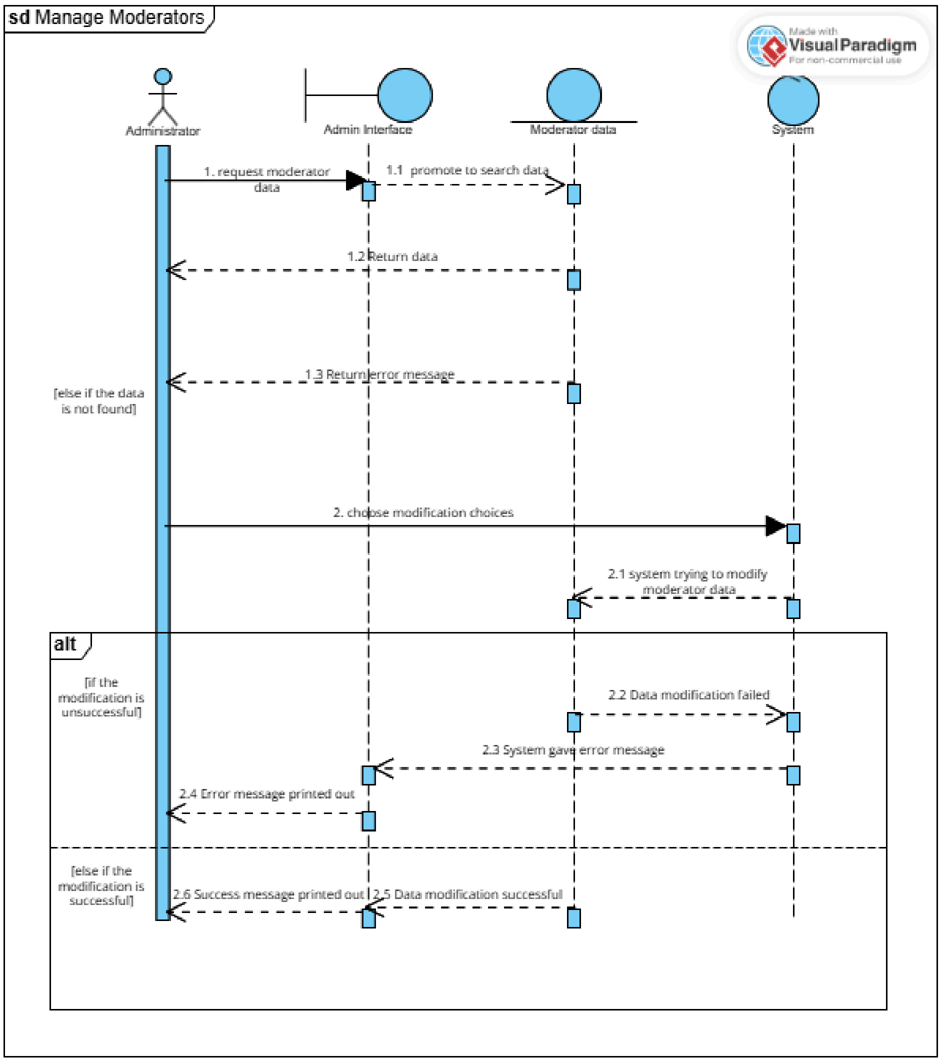
User case 15


sd Encrypt User Data
Made with Visual Paradigm For non-commercial use
User
Interface
Data Encryption
Database
1. Submit data
1.1 One-way hashing
1.2 Send data
alt
1.3 Data encription successful
[if data storage is sucessful]
1.4 Confirmation Message displayed
2.1 One-way hashing unsuccessful
[else if data storage is unsuccessful]
2.3 Error Message displayed
2.2 Data storage unsuccessful

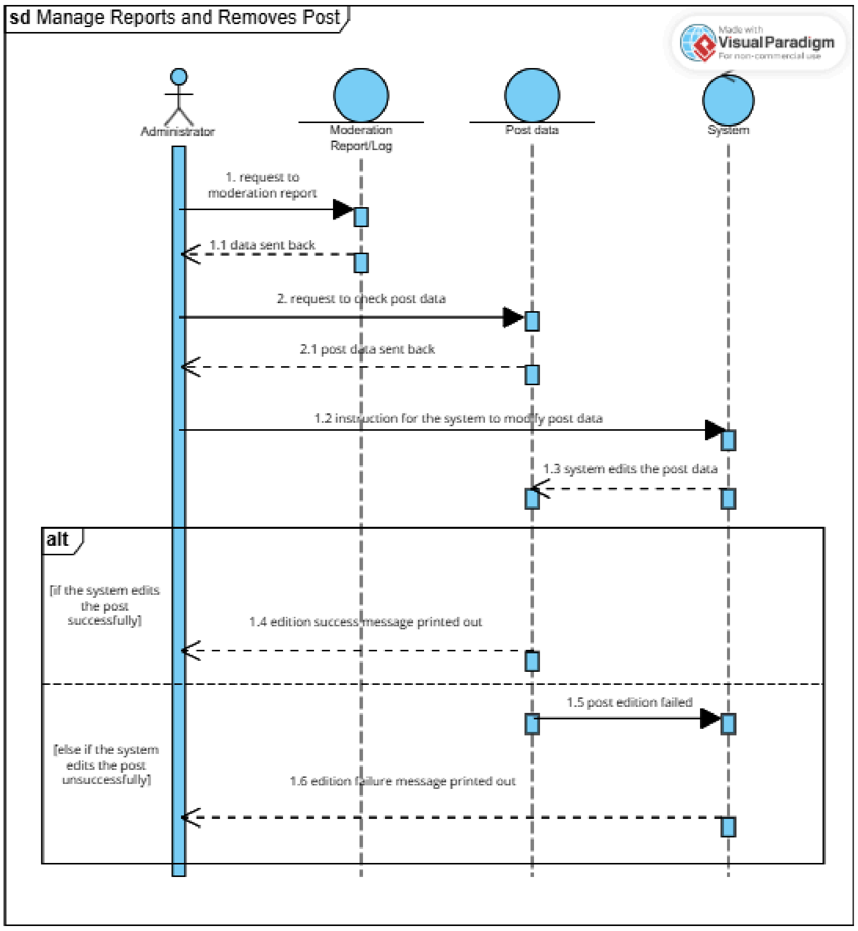
User case 16


sd Manage User Accounts
Made with Visual Paradigm For non-commercial use
Administrator
Admin Interface
User account data
Action choice
1. search user
1.1 promote to search in the account data set
alt
2.1 not found signal sent
[if user acount not found]
2.2 Not found message printed out
1.2 Account found, promote choice panel to be activate
1.3 Provide choices for action
1.4 Admin make choice
1.5 Try to make relevant change to user account
1.6 Change successful.

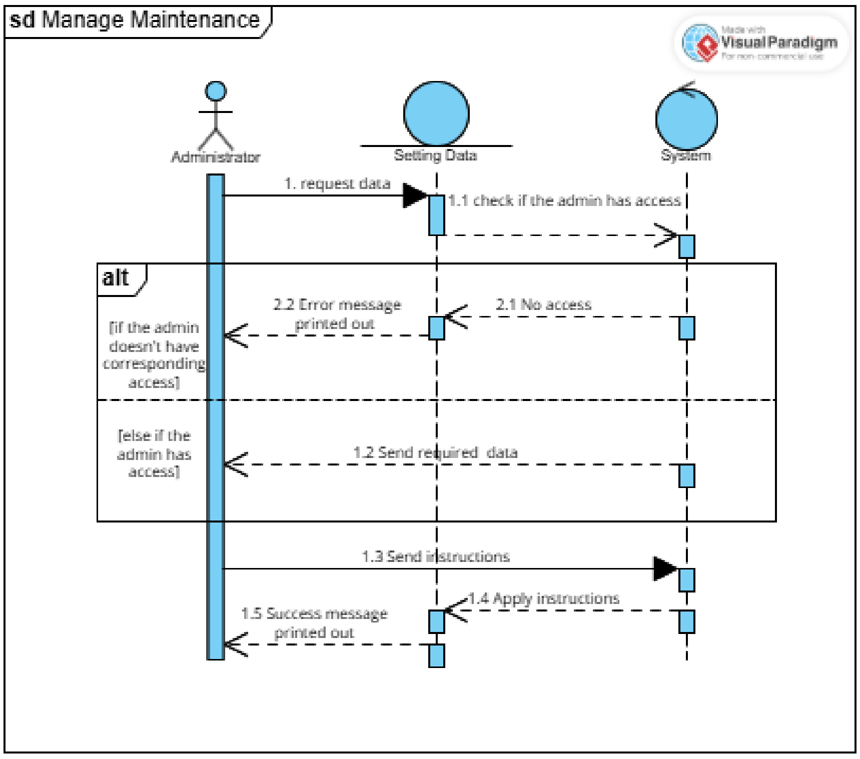
User case 17


sd Manage Moderators
Made with Visual Paradigm For non-commercial use
Administrator
Admin Interface
Moderator data
System
1. request moderator data
1.1 promote to search data
1.2 Return data
1.3 Return error message
[else if the data is not found]
2. choose modification choices
2.1 system trying to modify moderator data
alt
[if the modification is unsuccessful]
2.2 Data modification failed
2.3 System gave error message
2.4 Error message printed out
[else if the modification is successful]
2.6 Success message printed out
2.5 Data modification successful

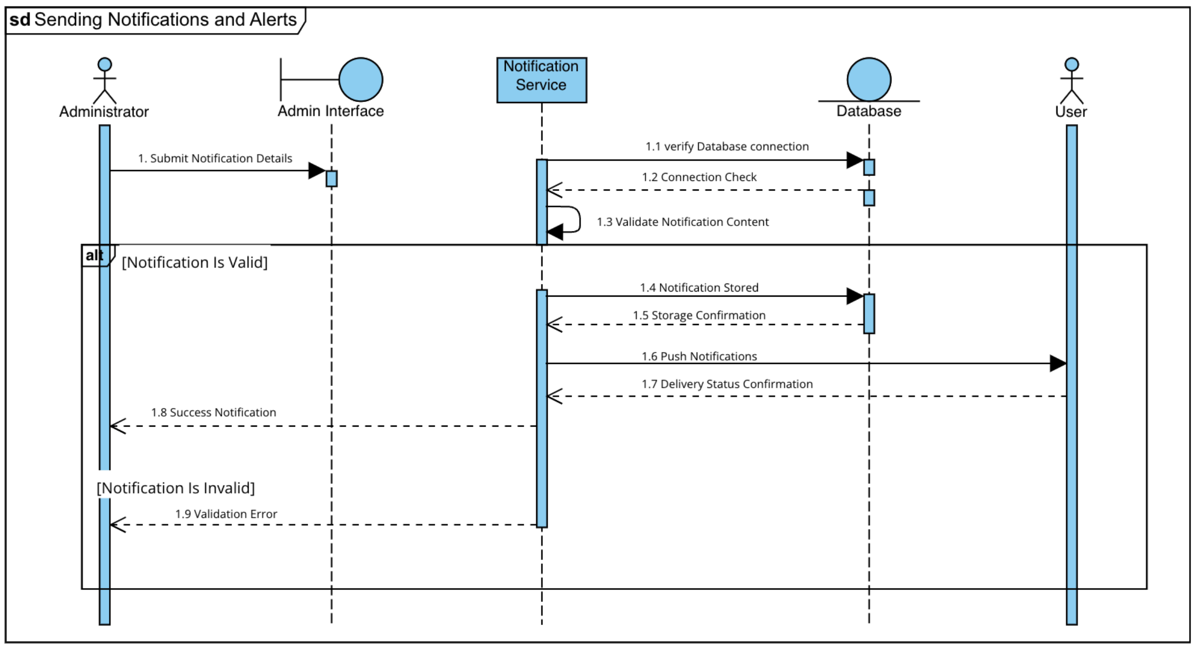
User case 18


sd Manage Reports and Removes Post
Made with Visual Paradigm For non-commercial use
Administrator
Moderation Report/Log
Post data
System
1. request to moderation report
1.1 data sent back
2. request to check post data
2.1 post data sent back
1.2 instruction for the system to modify post data
1.3 system edits the post data
alt
[if the system edits the post successfully]
1.4 edition success message printed out
1.5 post edition failed
[else if the system edits the post unsuccessfully]
1.6 edition failure message printed out

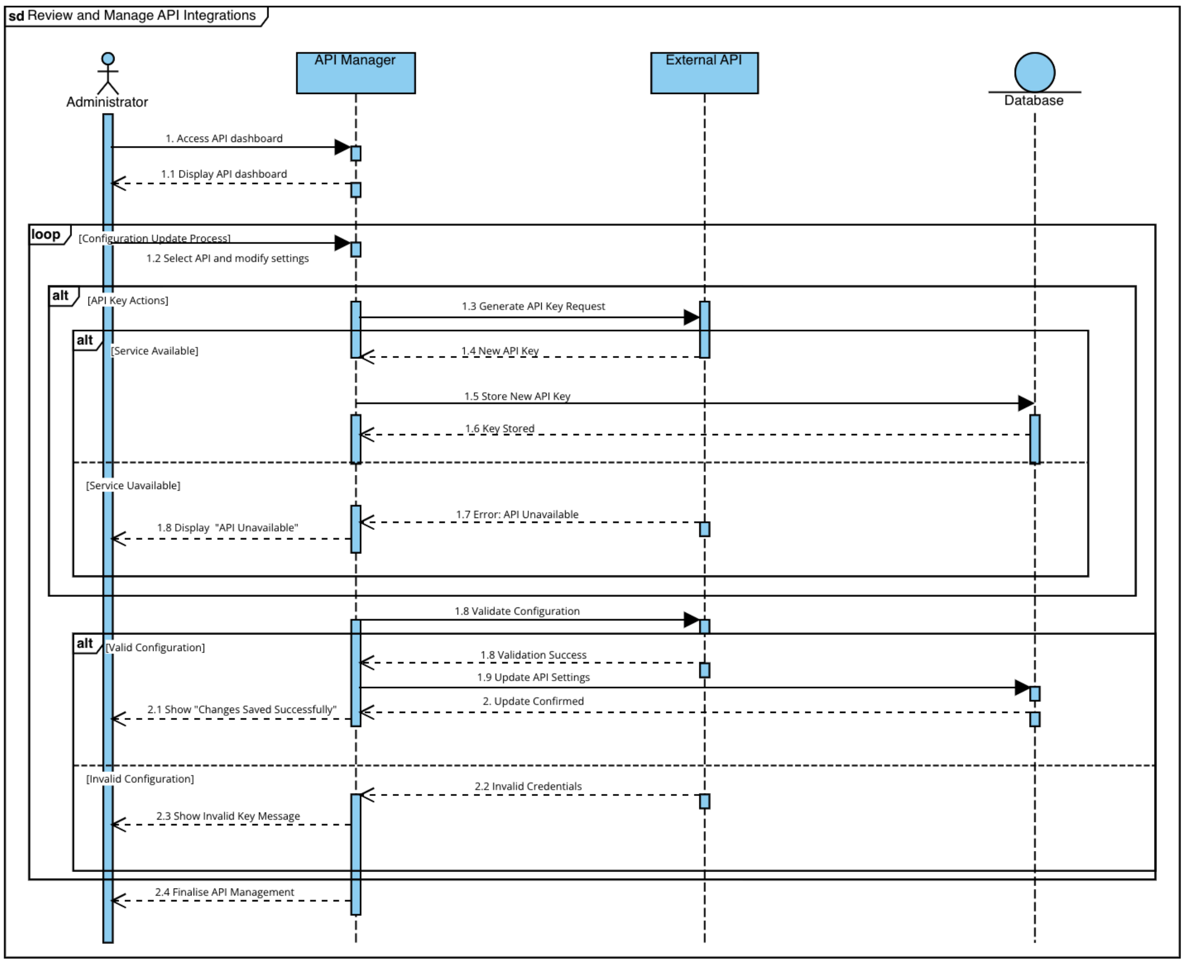
User case 19


sd Manage Maintenance
Made with Visual Paradigm For non-commercial use
Administrator
Setting Data
System
1. request data
1.1 check if the admin has access
alt
[if the admin doesn't have corresponding access]
2.2 Error message printed out
2.1 No access
[else if the admin has access]
1.2 Send required data
1.3 Send instructions
1.4 Apply instructions
1.5 Success message printed out

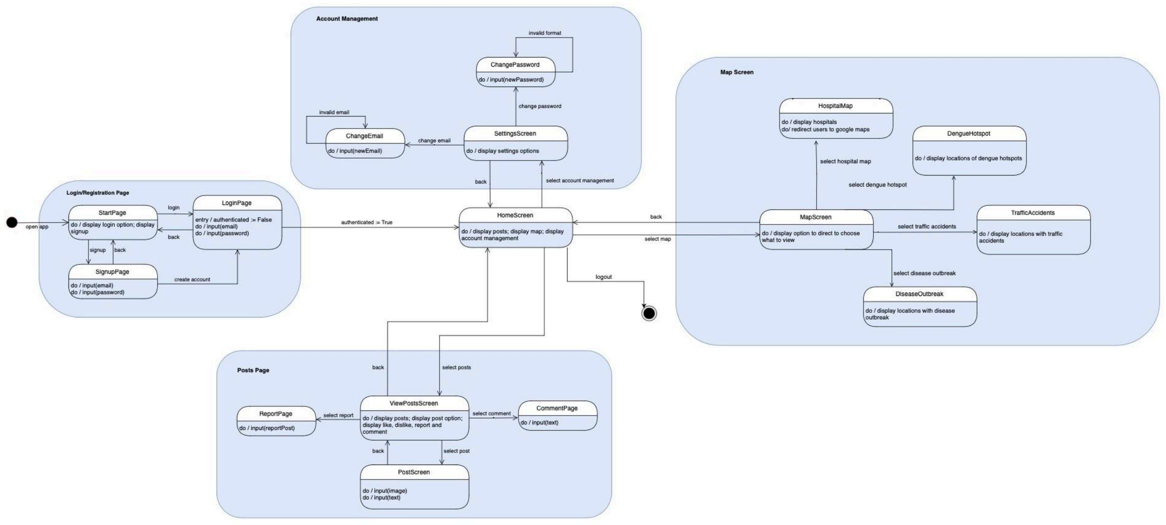
User Case 20


sd Sending Notifications and Alerts
Administrator
Admin Interface
Notification Service
Database
User
1. Submit Notification Details
1.1 verify Database connection
1.2 Connection Check
1.3 Validate Notification Content
alt
[Notification Is Valid]
1.4 Notification Stored
1.5 Storage Confirmation
1.6 Push Notifications
1.7 Delivery Status Confirmation
1.8 Success Notification
[Notification Is Invalid]
1.9 Validation Error

User Case 21

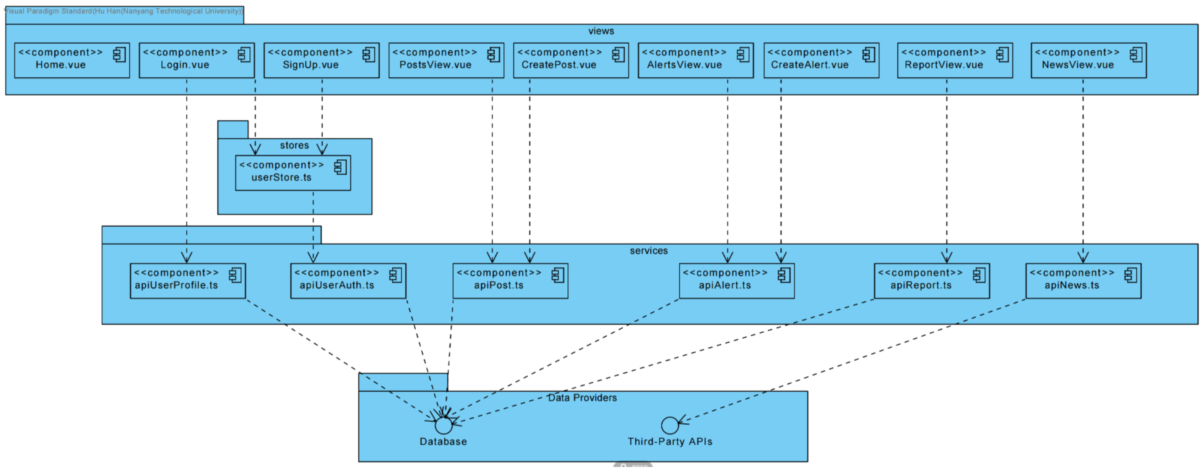

sd Review and Manage API Integrations
Administrator
API Manager
External API
Database
1. Access API dashboard
1.1 Display API dashboard
loop
[Configuration Update Process]
1.2 Select API and modify settings
alt
[API Key Actions]
1.3 Generate API Key Request
alt
[Service Available]
1.4 New API Key
1.5 Store New API Key
1.6 Key Stored
[Service Uavailable]
1.7 Error: API Unavailable
1.8 Display "API Unavailable"
1.8 Validate Configuration
alt
[Valid Configuration]
1.8 Validation Success
1.9 Update API Settings
2. Update Confirmed
2.1 Show "Changes Saved Successfully"
[Invalid Configuration]
2.2 Invalid Credentials
2.3 Show Invalid Key Message
2.4 Finalise API Management

2.3 Dialog Map



3. System Architecture

Our system adopts the MVC architecture, separating Models, Views and Controllers into different layers to improve extensibility.


Visual Paradigm Standard(Hu Han(Nanyang Technological University))
views
<<component>> Home.vue
<<component>> Login.vue
<<component>> SignUp.vue
<<component>> PostsView.vue
<<component>> CreatePost.vue
<<component>> AlertsView.vue
<<component>> CreateAlert.vue
<<component>> ReportView.vue
<<component>> NewsView.vue
stores
<<component>> userStore.ts
services
<<component>> apiUserProfile.ts
<<component>> apiUserAuth.ts
<<component>> apiPost.ts
<<component>> apiAlert.ts
<<component>> apiReport.ts
<<component>> apiNews.ts
Data Providers
Database
Third-Party APIs

Here are some essential components in each layer:

3.1 Views layer

Web UI for users to interact with the system.
Some main views:

1. Home: The main entrance of the application. Provide navigation to all functions.
2. Login: For registered users to log in to their account.
3. SignUp: For new users to register. Fields for essential information (email).
4. PostsView: For users to view blog posts and their geographical distribution.
5. CreatePost: For users to create a new post with categorical tags and locational information.
6. AlertsView: For users to get real-time first-hand alerts from official organizations.
7. CreateAlert: Only for accounts with specific identity to release alerts in a certain category in a specific location.
8. ReportView: Only for accounts with specific identity to view reports and take actions.
9. NewsView: For users to browse health and safety related news and search for nearby health facilities.

3.2 Services (Controllers) layer

Accept user inputs and send requests to the database to fetch required data, followed by data verification and processing if necessary.
Some main services:

1. apiUserProfile: Fetches user profile information depending on given user ID.
2. apiUserAuth: Deals with user login and signup.
    a. userStore: A middle layer between apiUserAuth.ts and views for maintaining user login information in browser's local storage.
3. apiPost: Deals with post retrieval, creation, deletion, edition, reaction, comments, reports and blocking (needs identity).

4. apiAlert: Deals with alerts retrieval, creation (needs identity) and withdrawal (needs identity)
5. apiReport: Deals with report retrieval,
6. apiNews: Retrieves information from third-party APIs.

3.3 Data Providers layer:

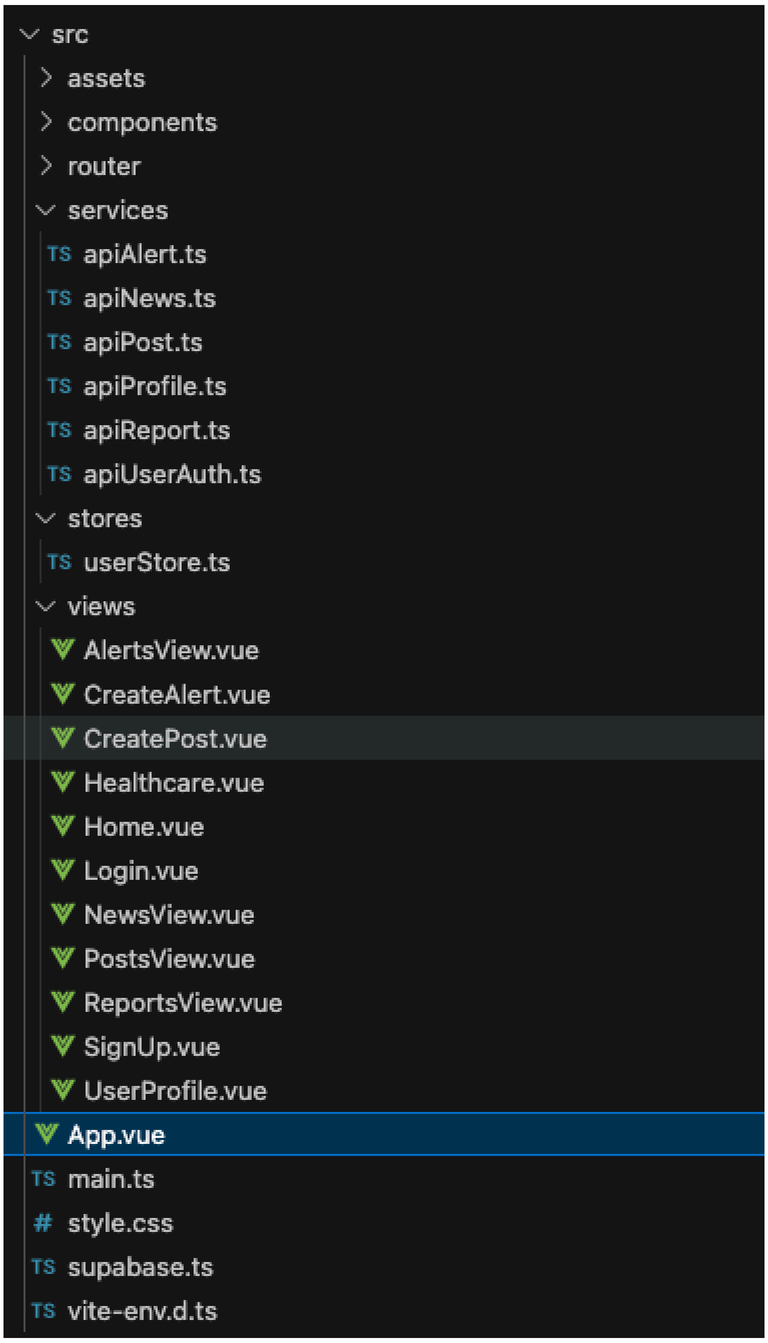
1. Database (Models): Persistent storage of core data the system deals with.
2. Third-Party APIs: Integrates with external data services for real-time news, weather alerts, or other relevant information.

4. Application Skeleton

We use Vue.js as our frontend framework, and Supabase as our database, which also provides simple ORM (Object Relational Mapping) APIs.

4.1 Frontend: Vue.js

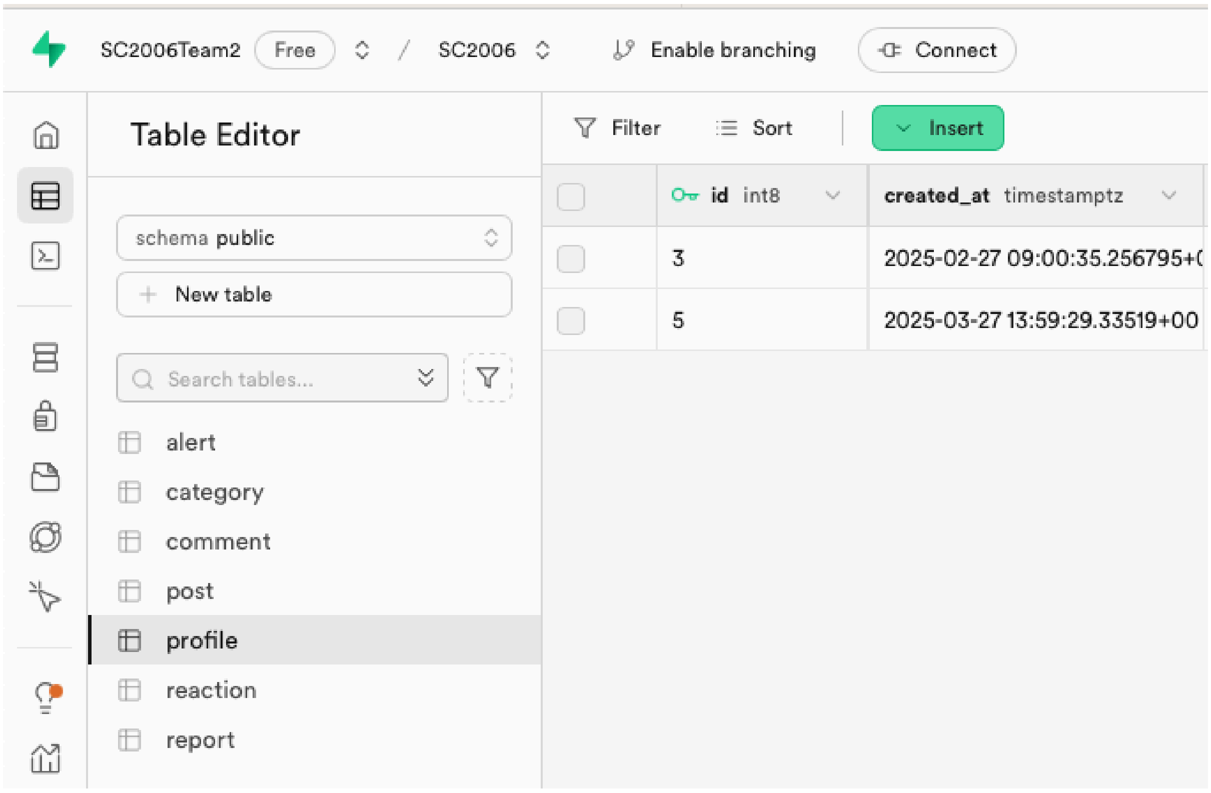
The source code is separated into different packages indicating different layers.



```
src
  assets
  components
  router
  services
    apiAlert.ts
    apiNews.ts
    apiPost.ts
    apiProfile.ts
    apiReport.ts
    apiUserAuth.ts
  stores
    userStore.ts
  views
    AlertsView.vue
    CreateAlert.vue
    CreatePost.vue
    Healthcare.vue
    Home.vue
    Login.vue
    NewsView.vue
    PostsView.vue
    ReportsView.vue
    SignUp.vue
    UserProfile.vue
  App.vue
  main.ts
  style.css
  supabase.ts
  vite-env.d.ts
```

4.2 Database: Supabase

Supabase is a PostgreSQL service that provides easy table definition, user authentication and Role Level Security (RLS) control.



alert
category
comment
post
profile
reaction
report

id int8	created_at timestamptz
3	2025-02-27 09:00:35.256795+(
5	2025-03-27 13:59:29.33519+00