

CS5310 SRS

Group 3

November 15, 2024

Contents

1	Introduction	1
1.1	Purpose	1
1.2	Scope	1
1.3	Definitions, Acronyms, and Abbreviations	2
1.3.1	Acronyms	2
1.3.2	Definitions	2
1.4	References	2
1.5	Overview	2
2	Overall Description	3
2.1	Product Perspective	3
2.1.1	System Interfaces	3
2.1.2	User Interfaces	4
2.1.3	Communications Interfaces	6
2.2	Product Functions	6
2.3	User Characteristics	6
2.3.1	Guest	6
2.3.2	Account Owner	6
2.3.3	Home Owner	6
2.3.4	Neighborhood Watch	7
2.3.5	Home Owner's Association Member (HOA Member)	7
2.4	Constraints	7
2.5	Assumptions and Dependencies	7
3	Specific Requirements	7
3.1	External Interface Requirements	7
3.1.1	User Interfaces	7
3.1.2	Hardware Interfaces	11
3.1.3	Software Interfaces	13
3.1.4	Communication Interfaces	13
3.2	Functional Requirements	13
3.2.1	Feature: Account Creation	13
3.2.2	Feature: Guest Account Creation	15
3.2.3	Feature: Account Modification	16
3.2.4	Feature: Account Deletion	17
3.2.5	Feature: Anonymous Settings	18
3.2.6	Feature: Neighborhood Watch Flash Alert	19
3.2.7	Feature: Neighborhood Watch Application	19
3.2.8	Feature: User Login	21
3.2.9	Feature: User Logout	21
3.2.10	Feature: User Settings	21
3.2.11	Feature: Block Users	22
3.2.12	Feature: Create Forum Post	22
3.2.13	Feature: View Forum Post	23
3.2.14	Feature: Edit Forum Post	23
3.2.15	Feature: Reply To Forum Post	24
3.2.16	Feature: Delete Forum Post	24
3.2.17	Feature: Activity Reporting	24
3.2.18	Feature: Delete Activity Report	25
3.2.19	Feature: Edit Activity Report	26
3.2.20	Feature: View Activity Report	27
3.2.21	Feature: Create Calendar Event	27

3.2.22	Feature: View Calendar Event	28
3.2.23	Feature: Edit Calendar Event	29
3.2.24	Feature: Delete Calendar Event	30
3.2.25	Feature: Register Pet	30
3.2.26	Feature: Modify Pet Registration	31
3.2.27	Feature: Delete Pet Registration	32
3.2.28	Feature: Create Missing Pet Report	32
3.2.29	Feature: Edit Missing Pet Report	33
3.2.30	Feature: Close Missing Pet Report	34
3.3	Non-Functional Requirements	34
3.3.1	Usability	34
3.3.2	Robustness	34
3.3.3	Reliability	35
3.3.4	Performance	35
3.3.5	Maintainability	36
3.3.6	Availability	36
3.3.7	Security	36
3.3.8	Portability	37
3.4	Design Constraints	38
Appendix A - Lofi Diagrams		39
A.1	Home Page Lo-Fi Diagrams	39
A.2	Logged-in User Page Lo-Fi Diagram	39
A.3	Reports Page Lo-Fi Diagram	39
A.4	Sign-Up Page Lo-Fi Diagram	40
A.5	Settings Page/Tab Lo-Fi Diagram	41
Appendix B - Use Cases		42
B.1	Account Creation	42
B.2	Account Administration	45
B.3	Account Actions	48
B.4	Neighborhood Watch	50
B.5	User Forum	52
B.6	Event Calendar	54
B.7	Activity Reports	56
B.8	Pet Registration	62
Appendix C - Class Diagrams		66
C.1	Homeowner Association Account	66
C.2	Neighborhood Watch Account	67
C.3	Homeowner Account	68
C.4	Owner Account	69
C.5	Guest Account	70

1 Introduction

This document is the Software Requirements Specification (SRS) used as the primary source by which to describe the breadth and depth of the homeowners association (HOA) Neighborhood Watch application. The application shall be developed as a cross-platform web-based and mobile application. The community's primary concern is to provide residents and guests with an informative, secure, and centralized source of communication built upon a tiered system specific to the user type. All requirements defined within this document shall be reviewed and approved by the board of directors of the HOA, with the president of the HOA having ultimate authority over any disputes and/or modifications to these requirements herein. <https://www.overleaf.com/project/6402731ec80c1724fe674d0a>

1.1 Purpose

The purpose of this document is to meticulously define and document stakeholder requirements to mitigate any ambiguities that may arise throughout the software development life cycle (SDLC) of the HOA Neighborhood Watch application. This document is binding upon approval of the HOA board of directors, and any alterations, modifications, and/or adjustments to functional, non-functional, and/or domain-specific requirements must have the approval of the HOA board of directors, with the ultimate authority for resolution of potential conflicts mediated by the president of the HOA. This document shall be version controlled with any modifications requiring documented prior approval and configuration management and is intended to be used as a stakeholder reference, as well as the primary source by which all planning, development, test, integration, and deployment will be based throughout the entire SDLC.

1.2 Scope

The primary objective in developing this application is to have an affordable, accessible, and centralized source of communication specific to the community accessible via a secure, user-friendly web-based, as well as both iOS and Android compatible mobile platforms. The application shall utilize a tiered user system such that HOA members/residents and their guests are enabled access to the application. Guest users will have access to community events and a map of the community. HOA members will be granted the user access to the application expanding the capabilities by providing access to the community forum.

This application is intended to promote community growth through the accessibility of guest users to general community information, promote a heightened sense of family-oriented community engagement, and most of all provide a means of dissemination of information specific to the safety and well-being of residents and guests through a community alert system to facilitate heightened community awareness and safety. In addition to the dissemination of information amongst HOA members/users, the application must have the ability to notify and report suspicious activity and/or active crimes to notify emergency services such as fire, medical, and local law enforcement. The alert system notifications shall be individually configurable on every accessible platform, and optional anonymity for both internal community alerts as well as emergency services shall be available to any user.

The application itself shall be built upon a tiered system enabling additional functionality enabling privileged user accounts the ability to approve new and/or delete existing users, as well as the ability to forum postings not in compliance with community standards. The specific language(s) to be used in front-end and back-end development, as well as the DBMS used for the archival and/or querying of data have yet to be determined and will be dependent upon the complexity of the application upon initial completion of the review and approval of functional, non-functional, and domain-specific requirements, but shall be finalized and documented before beginning the first iteration of software development.

1.3 Definitions, Acronyms, and Abbreviations

1.3.1 Acronyms

SRS	Software Requirements Specification
HOA	Homeowners Association
SDLC	Software Development Life-cycle
iOS	iPhone Operating System
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
GPS	Global Positioning System
API	Application Programming Interface
MB	Mega Byte

1.3.2 Definitions

Functional Requirements	Defines the behavior of a designed feature specific to a function performed by the system
Non-functional Requirements	Describes the general properties and/or attributes of the system
Domain	a thought, activity, or interest over which someone has control, influence, or rights
Neighborhood	A geographic location bounded by a set of addresses defined such as by an HOA charter, mapped zone, or bounded description of roads.
User	Generic term for any person utilizing the application
Homeowner	A user who owns a home in the associated neighborhood
Neighborhood Watch Member	A user who owns a home in the associated neighborhood and is approved by peers for elevated status in community management.
Home Owner's Association Member	A user who owns a home in the associated neighborhood and is a member of the homeowner's association.
Guest	A user who does not own a home in the associated neighborhood.

1.4 References

- [1] "IS-GPS-200," 22 August 2022. [Online]. [Accessed 2 May 2023].
- [2] "MongoDB," MongoDB, Inc., [Online]. [Accessed 2 May 2023].
- [3] "Simple Mail Transfer Protocol," April 2021. [Online]. [Accessed 2 May 2023].
- [4] 4-1986[R2022], "American Standard Code for Information Interchange," [Accessed 2 May 2023]
- [5] G. Alexander, "REST API (RESTful API)," TechTarget, [Online]. [Accessed 02 May 2023].

1.5 Overview

The remainder of this document will describe in immense detail the system from the product perspective including but not limited to architecture, interfaces, constraints, and limitations, as well as the functional aspects of the product. The functional perspective will emphasize in the behavior of the system in detail specific to the features defined by the stakeholders. The system requirements will emphasize the behavior of the system at a higher level which will generally be the aggregate of a set of features that accomplish a task or tasks defined in the requirements. In addition to the various types and levels of requirements, this document shall also contain a variety of diagrams, tables, and figures depicting visual representations of processes specific to the behavior of the system. Furthermore, details specific to system dependencies, constraints, limitations, and quality (i.e., reliability, availability, security, maintainability, portability, extensibility, usability, etc...), as well as cost, schedule, and performance metrics will also be included.

2 Overall Description

This section provides a description of the system in terms of product perspective, product functions, user characteristics, constraints, assumptions and dependencies, and the apportioning of requirements.

2.1 Product Perspective

The application will be developed to operate on mobile phones and as web apps accessible via browser. This will require the application to be compatible with multiple Operating Systems, and web browsers. It will also utilize other system services, so it will require the ability to communicate with other applications and services on the host device.

2.1.1 System Interfaces

This section outlines the high-level system interfaces for this program, their functionalities, and descriptions.

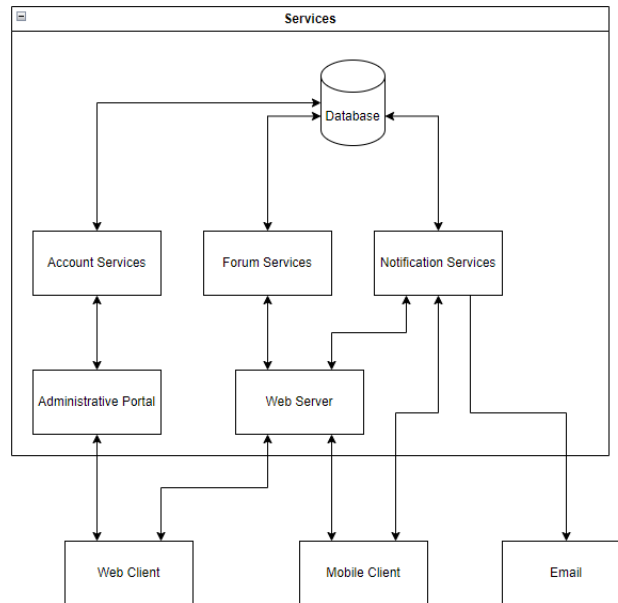


Figure 1: System Block Diagram

The architecture of the system provisions client interfaces, server interfaces, and data interfaces. Client interfaces include portions of the system that are directly touched by end users including web applications, mobile applications, and email. Server interfaces include portions of the system which ingest, distribute, and process data. Data interfaces include portions of the system that store and retrieve information.

2.1.1.1 Database This is a data interface that provides long-term storage and retrieval of account information, events, alerts, and forum activity.

2.1.1.2 Account Services This server interface facilitates the back-end functions of account management including account creation, modification, deletion, and information retrieval.

2.1.1.3 Forum Services This server interface facilitates the back-end functions of the user forum, calendar, and activity reporting.

2.1.1.4 Notification Services This server interface facilitates the back-end functions related to the distribution of notifications such as flash alerts and activity reports.

2.1.1.5 Administrative Portal This server interface facilitates the front-end functions of administrative tasks for account management.

2.1.1.6 Web Server This server interface facilitates front-end features for distributing web pages and receiving user input.

2.1.2 User Interfaces

This section outlines the various user interfaces to be developed for this program. A listing of interfaces is provided along with their logical characteristics of the system functions.

2.1.2.1 Activity Reporting The user interface for activity reporting must include the following:

- Drop-down menu to select activity level
- Text box to capture the user's description of the activity

2.1.2.2 Alerts

Web Client Alerts Web client alerts are displayed on a sidebar within a list containing alerts from top to bottom in chronological order. The form of the alert is as follows:

[Time of Alert] - [Alert Level] - [Alert Details]

Mobile Push Alerts Mobile client alerts are displayed in a list containing alerts from top to bottom in chronological order. The form of the alert is as follows:

[Time of Alert] - [Alert Level] - [Alert Details]

Email Alerts This is a text-only alert message sent to users. The form of the alert is the following:

[Alert Level]

THERE HAS BEEN AN ALERT ISSUED FOR [community name] AT THE TIME [Time of Alert] WITH A SEVERITY OF [Alert Level]. THE DETAILS OF THIS ALERT ARE AS FOLLOWS:

[Alert Details]

[Alert Level]

2.1.2.3 Community Forum The community forum will have several templates depending on the screen presented including a home page and post page

Forum Homepage This page includes:

- Titles of posts with links to threads.
- Post search bar

Forum Post This page contains the content of a post, replies to the post, and the ability to reply to a post. The format for this page consists largely of the original post, replies to the original post, and a text box to capture user input to reply to the post. The original poster will require features for editing their post and deleting their post. Users replying to a post require features for their replies to be edited or deleted.

The original post is contained in a box. The box contains the following information:

- Title of post
- Username of poster
- Timestamp of the post
- Content of post

Reply posts are contained in boxes below the original post. Each box contains the following information:

- Username of respondent
- Timestamp of reply
- Content of reply

2.1.2.4 Member Management The member management interface is a web-only interface accessible by certain privileged users. This interface is intended for the creation, viewing, modification, and deletion of accounts.

Create Account This interface contains the inputs for creating a new account. The interface includes the following:

- Text box for Resident Name
- Text box for Email Address
- Text box for Phone Number
- Text box for Resident Address
- Drop-down Menu for Resident Role
- Text box for Comments on this account

Viewing and Modifying Accounts This interface contains the interface for viewing and modifying account information. The interface consists of a tabular list of users in the community containing information that can be filtered by:

- Resident Name
- Email Address
- Phone Number
- Resident Address

- Resident Role

When an account is selected, a new window appears and the user can modify account information. The interface consists of various read-only information as well as modifiable information. Read-only information is as follows:

- Read-only text box for Username
- Read-only text box for Resident Name
- Read-only text box for Email Address
- Read-only text box for Phone Number
- Read-only text box for date and time of account creation

Modifiable information is as follows:

- Text box for Resident Address
- Drop-down Menu for Resident Role
- Text box for Comments on this account
- List of Restrictions where each restriction can be deleted
- Drop-down menu and button for adding a restriction

Finally, there is a push button with the text "Save Changes"

2.1.3 Communications Interfaces

The communication architecture will follow the client-server model. Communication between the client and server should utilize a REST-compliant web services. This application additionally uses Wi-Fi to communicate with mobile devices and the web page associated with the Neighborhood Watch app and the HOA. Lastly, the app uses SMS messaging and email to send flash alerts.

2.2 Product Functions

2.3 User Characteristics

This section outlines the roles and characteristics of users who interact with the system.

2.3.1 Guest

Guests represent someone who has just accessed the program. This user can view general information about the community but is not allowed to contribute to the community's forum or receive notifications from the community group. They will only have access to the home, login, and sign up pages until logging in to an account.

2.3.2 Account Owner

Account Owners represent someone who has an active account. They may be visiting the neighborhood or in the process of buying a home, but they have yet to be approved as a homeowner by an HOA member.

2.3.3 Home Owner

A homeowner is a general user within the community. This user has privileges to report incidents, contribute to and view community forums, register pets, as well as receive notifications from the community group.

2.3.4 Neighborhood Watch

A Neighborhood Watch member is an homeowner that has expanded administrative and emergency privileges in their community group. In the general case, Neighborhood Watch members consist of active or retired police and military personnel. They can use the app but also have limited admin controls. This user has the ability to revoke or restore homeowner privileges, post-emergency notifications, and administrate new Neighborhood Watch members.

2.3.5 Home Owner's Association Member (HOA Member)

A Homeowner's Association (HOA) Member is an administrative user which facilitates the general creation, deletion, and moderation of other community accounts.

2.4 Constraints

An Internet connection is a constraint for this application. It is crucial for anyone who wants to connect to the application to be able to access the database to have an internet connection.

Database size is also a constraint. If the database isn't accurately sized it will have to queue requests to accommodate users.

2.5 Assumptions and Dependencies

Email notifications depend on SMTP.

We assume that people signing up for this application will have a mobile device that they will want to connect to their account. We are also assuming that this device will have adequate resources for operating this application, but that may not be the case and this application may not work at all if it is lacking the necessary resources.

We are assuming that all Neighborhood Watch members are homeowners for the purposes of this application.

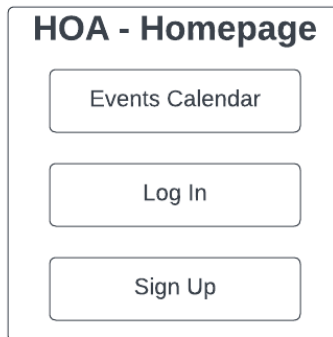
3 Specific Requirements

This section contains the software requirements for the system. External interfaces, user interfaces, hardware interfaces, software interfaces, communications interfaces, functional requirements, performance requirements, design constraints, software system attributes, and other requirements are identified, enumerated, and discussed.

3.1 External Interface Requirements

3.1.1 User Interfaces

- When a user clicks on "Neighborhood Watch" app or when a user search for Neighborhood Watch in the web then "Home Page" should be displayed as shown in figure 2. Here we can view events happening in the Neighborhood Watch community in the "Events Calendar" tab. we also log in to the user account in "Log in" tab. If the user is new to the community, then user can create an account using "Sign up" tab.
- If the user clicks on "Sign up" from figure 2, then it should navigate to figure 3. In here the new homeowner enters all the details and clicks "submit" to create a new account.
- If the user clicks on "Log in" from figure 2, then it should navigate to figure 4. In here the user can give their correct username and password to log in to their account.
- After entering correct credentials by the user in figure 4, then it should navigate to figure 5. In here the user can access all the activities in the neighborhood.



HOA - Homepage

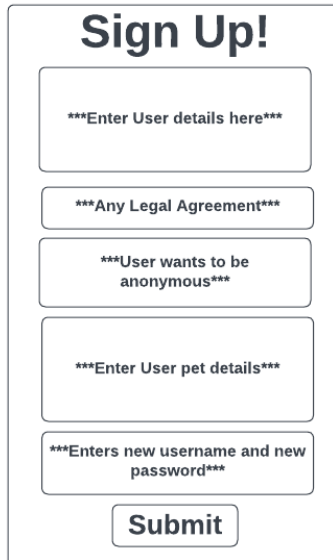
Events Calendar

Log In

Sign Up

The form is titled "HOA - Homepage" and contains three rectangular buttons stacked vertically: "Events Calendar", "Log In", and "Sign Up".

Figure 2: Home Page



Sign Up!

Enter User details here

Any Legal Agreement

User wants to be anonymous

Enter User pet details

Enters new username and new password

Submit

The form is titled "Sign Up!" and contains five rectangular input fields stacked vertically, each with a placeholder text: "***Enter User details here***", "***Any Legal Agreement***", "***User wants to be anonymous***", "***Enter User pet details***", and "***Enters new username and new password***". Below the fields is a rectangular "Submit" button.

Figure 3: Sign Up Page



Neighborhood Watch

User enters username & password here

Log In

Display a solution if user forgets username or password

The form is titled "Neighborhood Watch" and contains three elements: a rectangular input field with placeholder text "***User enters username & password here***", a rounded rectangular "Log In" button, and a rectangular box with placeholder text "***Display a solution if user forgets username or password***".

Figure 4: Log In Page

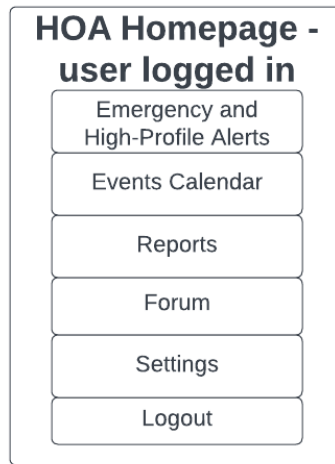


Figure 5: Logged-in user page

- In figure 5, if the user clicks on "Emergency and High-Profile Alerts" tab, then it should navigate to figure 6. In here the user can view all the emergency alerts in the neighborhood.



Figure 6: Emergency and High-Profile Alerts Page

- If the user clicks on "Events Calendar" in figure 5, then it should navigate to figure 7. In here the user can create new events, view other events, edit the event, and delete the event that is create by the user.
- In figure 5, if the user clicks on "Reports" tab, it should navigate to figure 8. In here the user can view all the emergency alerts, can filter options to view the reports, and the user can get access to map showing pin point locations of crimes in the neighborhood.
- If the user clicks on "forum" tab in figure 5, then it should navigate to figure 9. In here the user can communicate with the other users in "Chat Room", provide any information about the crimes in "Reports" tab, and the user also can provide information about the missing pets to the respective owners in "Missing Pets" tab.
- If the user clicks on "Reports" tab in figure 9, then it should navigate to figure 10. In here the user can view the reports filed in the Neighborhood Watch community and the user can provide information on the active reports in the neighborhood.

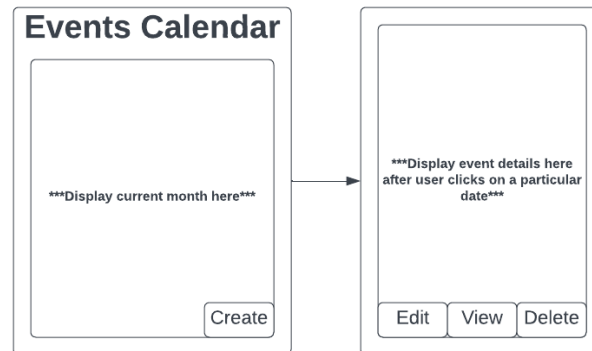


Figure 7: Event Calendar Page

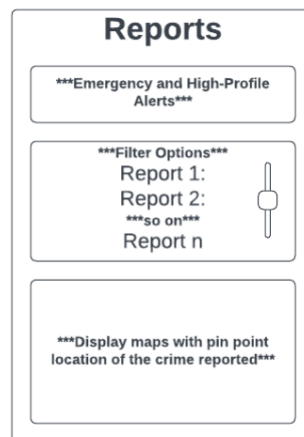


Figure 8: Reports Page

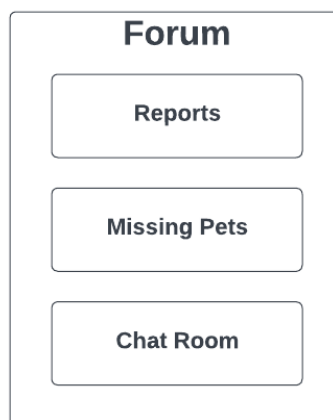


Figure 9: Forum Page



Figure 10: Reports Page in forum tab

- If the user clicks on "Missing Pets" tab in figure 10, then it should navigate to figure 11. In here the user can view all the missing pet reports, and details of the pet and provide information to the respective owner if the user have any.

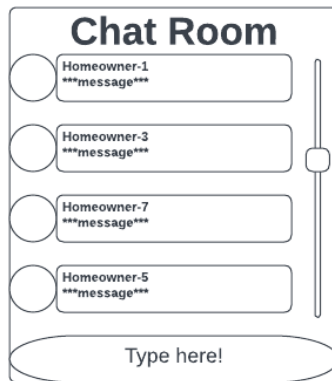


Figure 11: Missing Pets Page

- If the user clicks on "Chat Room" tab in figure 9, then it should navigate to figure 12. In here the user can send a group message to the homeowners in the Neighborhood Watch.
- If the user clicks on "Settings" tab in figure 5, then it should navigate to figure 13. In here the user can set his profile according to his available preferences.
- If user clicks on "Logout" button in figure 5, then it should navigate back to log in page as shown in figure 4.

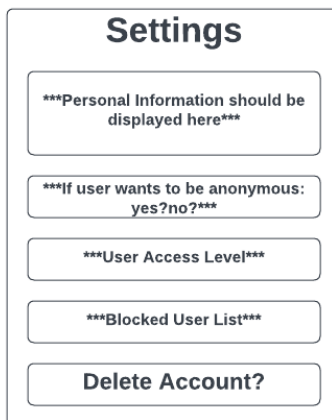
3.1.2 Hardware Interfaces

This "Neighborhood Watch" Project uses some hardware interfaces like GPS, Camera. GPS is used to in the "Reports" tab, where a map is displayed with the pin points representing the locations of the crime reported.



The diagram shows a chat room interface. At the top is a title bar labeled "Chat Room". Below it, there are four message entries, each consisting of a circular profile icon placeholder, a name, and a message body. The entries are: "Homeowner-1" with "***message***", "Homeowner-3" with "***message***", "Homeowner-7" with "***message***", and "Homeowner-5" with "***message***". To the right of these entries is a vertical scrollbar. At the bottom of the chat area is a rounded rectangular input field with the placeholder text "Type here!".

Figure 12: Chat Room Page



The diagram shows a settings interface with a title bar labeled "Settings". Below the title bar are five rectangular buttons or sections, each containing text. From top to bottom, they are: "***Personal Information should be displayed here***", "***If user wants to be anonymous: yes?no?***", "***User Access Level***", "***Blocked User List***", and "Delete Account?".

Figure 13: Settings Page

Camera is used to take a picture and upload a picture in the Chat room, or at the time of reporting a missing dog, or reporting a crime in the neighborhood community.

3.1.3 Software Interfaces

This "Neighborhood Watch" Project uses some software interfaces like Libraries, Operating System, API, and other software systems. Libraries are pre-written code, can be used for specific functionalities for developing the Neighborhood Watch application. Operating System helps in communication between the hardware and the software. Maps API is used to display the map in the "Reports" tab. Software systems such as notification feature is used for emergency alerts in the Neighborhood Watch application.

3.1.4 Communication Interfaces

This "Neighborhood Watch" project must work on HTTPS. The website must work with all the versions of the web browser. This project must use end-to-end encryption during the chat. This application should be able to send/upload a file with maximum file size of 64 MB.

3.2 Functional Requirements

3.2.1 Feature: Account Creation

Stakeholders want users to be able to create accounts to access the various features of the application. Before accounts are finalized a vetting process is required.

3.2.1.1 Create Account Link

Button Click (Sign up): The user shall click the button titled "Sign up" when they would like to create a new account for themselves.

Exceptions: None.

3.2.1.2 New Account Information

Text Field (First Name): The user must enter a first name.

Text Field (Last Name): The user must enter a last name.

Email Address Field (Email Address): The user must enter a unique email address.

Password Field (Password): The user must enter a valid password.

Exceptions:

Empty Text Field (First Name): The system shall display a red asterisk (*) next to field.

Empty Text Field (Last Name): The system shall display a red asterisk (*) next to field.

Empty Email Address Field (Email Address): The system shall display a red asterisk (*) next to field.

In Use Email Address Field (Email Address): The system shall display a red asterisk (*) next to field and red text stating "Email already in use".

Empty Password Field (Password): The system shall display a red asterisk (*) next to field.

Invalid Password Field (Password): The system shall display a red asterisk (*) next to field and a line of text informing the user of the password rules.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.1.3 Submit New Account Information Link

Button Click (Create Account): The user must click the button titled "Create Account" when all information is correctly filled out.

Notice (Verify email): The system must notify the user to check their email for a verification link.

Exceptions: None.

3.2.1.4 Verify New Account

Request Confirmation (Request): The system shall send a verification email to the prospective account's email address.

Request Confirmation (Verification): The user must click the link in the verification email titled "Verify Email".

Request Confirmation (Redirection): The "Verify Email" link will redirect the user to a page to finish the account creation process.

Exceptions:

Request Confirmation (Expiration): If the user does not respond to the email within 30 minutes by clicking the link, the account creation process expires.

Request Confirmation (Redirection): If the user attempts to respond after the allotted time, the user will be redirected to an error page explaining they need to restart the process.

3.2.1.5 New Account Neighborhood information

Map Field (Location): The user must select a pin on the map representing their address.

Exceptions:

Empty Map Field (Location): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.1.6 Submit New Account Neighborhood Information Link

Button Click (Submit): The user must click the button titled "Submit" when the fields are complete.

Notice (Submission Received): The following text is displayed to the user "Thank you, your application is under review."

Exceptions: None.

3.2.1.7 Admin Privileges: New Account Review

Button Click (View List of New Applications): The system shall display to admin users a list of new applicants.

Button Click (Select Application): The system shall display information of the new applicant.

Button Click (Approve Application): The admin user shall click the button titled "Approve applicant".

Button Click (Approve Application): The system shall send an email to the applicant expressing the account has been approved.

Button Click (Deny Application): The admin user shall click the button titled "Deny applicant".

Button Click (Deny Application): The system shall send an email to the applicant expressing the account has been denied.

Exceptions: None.

3.2.2 Feature: Guest Account Creation

Stakeholders want users that do not have an HOA account to have the option to sign up and create one. The minimum information for a basic account is Name, Email, and Password.

3.2.2.1 Create Guest Account Link

Button Click (Sign up as Guest): The user shall click the button titled "Sign up as Guest" when they would like to create a new guest account for themselves.

Exceptions: None.

3.2.2.2 Submit New Account Information

Text Field (First Name): The user must enter a first name.

Text Field (Last Name): The user must enter a last name.

Email Address Field (Email Address): The user must enter a unique email address.

Password Field (Password): The user must enter a valid password.

Button Click (Create Account): The user must click a button titled "Create Account" when all information is correctly filled out.

Notice (Verify email): The system must notify the user to check their email for a verification link.

Exceptions:

Empty Text Field (First Name): The system shall display a red asterisk (*) next to field.

Empty Text Field (Last Name): The system shall display a red asterisk (*) next to field.

Empty Email Address Field (Email Address): The system shall display a red asterisk (*) next to field.

In Use Email Address Field (Email Address): The system shall display a red asterisk (*) next to field and red text stating "Email already in use".

Empty Password Field (Password): The system shall display a red asterisk (*) next to field.

Invalid Password Field (Password): The system shall display a red asterisk (*) next to field and a line of text informing the user of the password rules.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.2.3 Verify New Account

Request Confirmation (Request): The system shall send a verification email to the prospective account's email address.

Request Confirmation (Verification): The user must click a link in the verification email titled "Verify Account".

Request Confirmation (Redirection): The "Verify Account" link will redirect the user to a page to finish the account creation process.

Exceptions:

Request Confirmation (Expiration): If the user does not respond to the email within 30 minutes by clicking the link, the account creation process expires.

Request Confirmation (Redirection): If the user attempts to respond after the allotted time, the user will be redirected to an error page explaining they need to restart the process.

3.2.3 Feature: Account Modification

Stakeholders want users to be able to edit their accounts when personal information changes.

3.2.3.1 Edit Account Link

Button Click (Edit account): The user shall click the button titled "Update account" when they would like to change their account information.

Exceptions: None.

3.2.3.2 Edit Personal Information

Text Field (First Name): The user must replace an existing first name.

Text Field (Last Name): The user must replace an existing last name.

Email Address Field (Email Address): The user must replace an existing unique email address.

Password Field (Password): The user must replace an existing valid password.

Exceptions:

Empty Text Field (First Name): The system shall display a red asterisk (*) next to field.

Empty Text Field (Last Name): The system shall display a red asterisk (*) next to field.

Empty Email Address Field (Email Address): The system shall display a red asterisk (*) next to field.

In Use Email Address Field (Email Address): The system shall display a red asterisk (*) next to field and red text stating "Email already in use".

Empty Password Field (Password): The system shall display a red asterisk (*) next to field.

Invalid Password Field (Password): The system shall display a red asterisk (*) next to field and a line of text informing the user of the password rules.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.3.3 Submit Personal Info Edit

Button Click (Submit): The user must click a button titled "Submit" when all information is correctly filled out.

Notice (Changes received): The system must notify the user that the changes have been made.

Exceptions: None.

3.2.3.4 Edit Neighborhood information

Map Field (Location): The user must replace existing map pin with a new map pin.

Empty Map Field (Location): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.3.5 Submit Neighborhood Info Edit

Button Click (Submit): The user must click the button titled "Submit" when the fields are complete.

Notice (Submission Received): The system must notify the user that the changes have been made.

Exceptions: None.

3.2.4 Feature: Account Deletion

Stakeholders want members of the HOA board to have formal governance (within the bounds of established privacy requirements) over all user accounts. Account holders require the ability to remove their account at will from the system.

3.2.4.1 Delete Account Link

Button Click (Delete Account): The user shall click the a button titled "Delete account".

Exceptions: None.

3.2.4.2 Additional data removal

Check Box (Clear history): The user shall select whether they would like all their personally contributed data removed from the system.

Check Box (Unsubscribe from email list): The user shall select whether they would like their contact information to continue being used for the neighbourhood email list.

Exceptions: None.

3.2.4.3 Verify Delete Account

Notice (Warning): The system shall warn the user that they are proceeding to remove their account and it's associated information before allowing them to finalize their decision.

Request Confirmation (Sign in): The system shall force the user to enter in their sign in credentials again before finalizing remove.

Exceptions: None.

3.2.4.4 Admin Privileges: Account delete

Button Click (Delete Admin): The administrative account holder may remove another administrative account holder. The system shall verify this action by having another administrative account holder approve the action. This may be the account holder of the pending removal account or a third party administrative account holder.

Button Click (Delete Self): The administrative account holder shall be able to delete their own account with the same set of warnings and additional data removal prompts as a standard user.

Admin can remove multiple accounts at once The administrative account holder shall be able to select and delete multiple standard user accounts at once.

Admin cannot persist user data (without user consent) The system shall notify the user that an administrator has removed their account, unless the user selects otherwise all additional data removal items will be enforced.

Exceptions: None.

3.2.5 Feature: Anonymous Settings

Stakeholders desire for current neighbourhood residents with active user accounts in good standing to be able to post alerts, start forum threads and comment on existing thread anonymously.

3.2.5.1 Default anonymous

Global anonymous send setting: The user shall be able to elect that all their posts be made anonymously.

Global anonymous read setting: The user shall be able to elect that all posts, both theirs and others be made anonymous for them. Their post will also be anonymous to others.

Exceptions: None.

3.2.5.2 Post Anonymous Alert

Select Post as Anonymous: The user shall select "Post as Anonymous" when attempting to post an anonymous alert.

Alert vetting: All alerts whether anonymous or named must be approved by an admin before they are published to the alert board (See 3.2.6 Feature: Neighborhood Watch Flash Alert for more).

Exceptions: None.

3.2.5.3 Start Anonymous Forum Thread

Select Post as Anonymous: The user shall select "Post as Anonymous" when attempting to post a new thread on the forum anonymously.

Exceptions: None.

3.2.5.4 Comment on existing thread anonymously

Select Post as Anonymous: The user shall select "Post as Anonymous" when attempting to comment on a new thread on the forum anonymously.

Exceptions: None.

3.2.5.5 Admin Privileges: Anonymous mode

- Anonymity must be maintained at all times between accounts with standard user privileges.

- Users with administrative privileges can request that anonymity be disclosed to them but the anonymous user must give express permission before anonymity is disclosed.

No manual override clause: Anonymity must be maintained at all times between accounts with standard user privileges.

Exceptions: None.

Request Confirmation (Request): The users with administrative privileges must request via email that anonymity be disclosed to them.

Request Confirmation (Confirm): The users with an active request must give express permission before anonymity is disclosed by clicking a link saying they "accept the request"

Exceptions: None.

3.2.6 Feature: Neighborhood Watch Flash Alert

Stakeholders want Neighborhood Watch members with active admin accounts to be able to send flash alerts within the application to warn homeowners of emergent conditions in the neighborhood.

3.2.6.1 Logged In Home Page Flash Alerts Access

Button Click (Flash Alerts): The user shall click the button titled "Flash Alerts" when they would like to announce an alert.

Button Click (Response): The system shall navigate to a page displaying "Reports".

Exceptions: Authentication as a Neighborhood Watch Member with privileges is required to send flash alerts.

3.2.6.2 Populating Alert report

Time field (Time of Incident): The user shall select the time that the incident took place.

Date field (Date of the incident): The user shall select the date that the incident took place.

Map field (Location of the incident): The user shall map pin the location that the incident took place.

Text field (Action Taken): The user shall enter the action taken and the resident action needed.

Exceptions:

Empty Time Field (Time of Incident): The system shall display a red asterisk (*) next to field.

Empty Date Field (Date of the incident): The system shall display a red asterisk (*) next to field.

Empty Map Field (Location of the incident): The system shall display a red asterisk (*) next to field.

Empty Text Field (Action Taken): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.6.3 Submitting Alert report

Button Click (Submit Alert): The user shall click the button titled "Submit Report" to make a report and send an alert to all subscribed users.

Exceptions: None.

3.2.7 Feature: Neighborhood Watch Application

Stakeholders want homeowners to have the ability to apply to become a Neighborhood Watch member. This application, once submitted, is placed in a queue for pending approvals.

3.2.7.1 Logged In Home Page Application Access

Button Click (Watch Application): The user shall click the button titled "Join the watch today!" when they would like to apply to join the Neighborhood Watch.

Button Click (Response): The system shall generate a pop-up form to entries to fill and a deactivated "submit" button.

Exceptions: Authentication as an account holding member is required to apply to be a Neighborhood Watch member.

3.2.7.2 Populating Watch Application

Text field (User Purpose): The user shall enter the reason they would like to become a Neighborhood Watch member.

Text field (Signature): The user shall enter their full name as a statement that they have read the Neighborhood Watch member contract.

Date field (Date): The user shall select the date that they signed.

Text field (Address): The user shall enter their address as their account identifier.

Exceptions:

Empty Text field (User Purpose): The system shall display a red asterisk (*) next to field.

Empty Text field (Signature): The system shall display a red asterisk (*) next to field.

Empty Date field (Date): The system shall display a red asterisk (*) next to field.

Empty Text field (Address): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall maintain deactivated "submit" button.

3.2.7.3 Submitting Watch Application

Button Click (Submit application): The user shall click the button titled "Submit Application" to send a notice to all subscribed Neighborhood Watch member.

Exceptions: None.

3.2.7.4 Admin Privileges: Approving Watch Application

Button Click (Approve): The admin user shall click the button titled "Accept Application" to send a notice to the applicant that they were approved to be a Neighborhood Watch member.

System update (Change Status): The system shall update the applicants account to show that they are a Neighborhood Watch member.

System update (Change privileges): The system shall update the applicants account to allow them their account privileges as a member of the Neighborhood Watch.

Exceptions: None.

3.2.7.5 Admin Privileges: Rejecting Watch Application

Button Click (Reject): The admin user shall click the button titled "Reject Application" to send a notice (once the pop-up has been closed) to the applicant that they were not approved to be a Neighborhood Watch member.

Notice (Rejection details): The system generate a pop item that allows the user to fill in why they rejected the applicant. This information will be sent with the applicants notice.

Exceptions: None.

3.2.8 Feature: User Login

Stakeholders want users to be able to login to access their accounts. HOA members want only authorized users accessing certain information.

3.2.8.1 User Login Link

Button Click (Log in): The user shall click a button titled "Log in".

Exceptions: None.

3.2.8.2 Submit User Login Information

Text Field (Username): The user must enter their email address.

Text Field (Password): The user must enter their password.

Exceptions:

Incorrect Username: The system shall display a red asterisk(*) next to field.

Incorrect Password: The system shall display a red asterisk(*) next to field.

3.2.9 Feature: User Logout

Stakeholders want users to be able to logout to protect their information.

3.2.9.1 User Logout Link

Button Click (Logout): The user shall click the button titled "Logout".

Exceptions: None.

3.2.9.2 Logout user

Action (Logout): The System shall log the user out, sending them back to the HOA Homepage.

Exceptions: None.

3.2.10 Feature: User Settings

Stakeholders want users to be able to see/edit/delete personal/pet information and change their privacy settings and delete their accounts. HOA Members and Neighborhood Watch members will also have access to User level settings and Revoke user settings.

3.2.10.1 User Settings Link

Button Click (Settings): The user shall click a button titled "Settings".

Exceptions: None.

3.2.10.2 User Setting Page

Display field (Personal Information): Displays the user's personal information.

Button (Edit): Gives users the option to edit their personal information.

Display field (Pet Information): Displays the user's pet information.

Button (Edit): Gives users the option to edit their pet information.

On/Off Setting (Anonymous): Displays the user's anonymity setting check box.

Button (Delete Account): Gives users the option to delete their account.

Display field (User privileges): Displays the users current access level.

Exceptions: None.

3.2.10.3 Admin Privileges: User Settings

Button click (Modify User privileges): Gives admin users the option change a member's access level.

Button click (Remove User): Gives admin users the option change a delete a member's account.

Display field (Blocked Users): Displays the list of user blocked from posting.

Exceptions: None.

3.2.11 Feature: Block Users

Stakeholders want admin users to have the ability to revoke the posting privileges of others.

3.2.11.1 Block User Link

Button Click (Block User): The user shall click the button titled "Block user".

Exceptions: None.

3.2.11.2 Block User Action

Action (Block User): If a user becomes blocked they no longer have the ability to post on forums or create reports.

Exceptions: None.

3.2.12 Feature: Create Forum Post

Stakeholders want users to be able to create a new post in the forum section of the HOA application.

3.2.12.1 Submit Forum Post Link

Button Click (Create Post): The user shall click the button titled "Create post" to create a new post.

Exceptions: None.

3.2.12.2 Forum Post Content

Text Field (Forum Post Content): The user shall enter the content of their forum post.

Text Field (Persistence): All content should persist and not expire until the "Click Create Post" action has taken place.

Exceptions:

Empty Text Field ((Forum Post Content): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall deactivate the "Click Create Post" action.

3.2.13 Feature: View Forum Post

Stakeholders desire that users can view the full content of forum posts.

3.2.13.1 Forum Post Link

Button Click (Forum Post): The user shall click on the forum post title to view a new post.

Exceptions: None.

3.2.14 Feature: Edit Forum Post

Stakeholders desire that users can modify their own forum posts.

3.2.14.1 Edit Forum Post Link

Button Click (Edit Post): The user shall click the "Edit Post" button on the forum post to edit a post that they made.

Exceptions: None

3.2.14.2 Edit Forum Post Content

Text Field (Forum Post Content): The user shall update the content of their forum post.

Text Field (Persistence): All content should persist and not expire until the "Submit Post" action has taken place.

Exceptions:

Empty Text Field ((Forum Post Content): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall deactivate the "Submit Post" action.

3.2.14.3 Submit Edit Post Link

Button Click (Submit Post Edit): The user should click the "Submit" button in the edit window of the forum post to submit the edits they made.

Notice (Submission Received): The system must notify the user that the changes have been made.

Exceptions: None

3.2.15 Feature: Reply To Forum Post

Stakeholders desire that user's can reply to a forum post. Either theirs or that of someone else.

3.2.15.1 Forum Post Reply Link

Button Click (Reply To Forum Post): The user shall click the button titled "Reply" on the forum post to generate a new forum post associated with an existing one.

Exceptions: None.

3.2.15.2 Enter Forum Post Content

Text Field (Forum Post Content): The user shall the content of their forum post reply.

Text Field (Persistence): All content should persist and not expire until the "Click Reply" action has taken place.

Exceptions:

Empty Text Field ((Forum Post Content): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall deactivate the "Click Reply" action.

3.2.16 Feature: Delete Forum Post

Stakeholders want users to be able to delete their own forum post.

3.2.16.1 Forum Post Delete Link

Button Click (Delete Forum Post): The user shall click on the button titled "Delete Post" to remove the forum post from the forum.

Exceptions: None.

3.2.16.2 Forum Post Delete Confirmation

Notice (Delete Post Confirmation): The system shall prompt the user to confirm their previous action before complying.

Exceptions: None.

3.2.17 Feature: Activity Reporting

Stakeholders want users to have the ability to report activities happening in the community. The enumerated are activities a Homeowner may report:

1. General Activity
2. Suspicious Activity
3. Criminal Activity
4. Emergency Activity
5. When a report is made, only a single selection on the type of activity is allowed.

3.2.17.1 Create Activity Report Link

Button Click (Create Report): The user shall click the button titled "Create Report" to create a new report.

Exceptions: None.

3.2.17.2 Activity Report Content

Date Field (Date): The user shall select the date of the activity in their report.

Time Field (Time): The user shall select the time of the activity in their report.

Drop down Field (Type): The user shall select the type of activity in their report (See Feature: Activity Reporting).

Map Field (Location): The user shall pin on map the location of the activity in their report.

Text Field (Description): The user shall enter the description of the activity in their report.

Exceptions:

Unselected Date Field (Date): The system shall display a red asterisk (*) next to field.

Unselected Time Field (Time): The system shall display a red asterisk (*) next to field.

Unselected Drop down Field (Type): The system shall display a red asterisk (*) next to field.

Empty Map Field (Location): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall deactivate the "Submit Report" action.

3.2.17.3 Submit Activity Report Link

Button Click (Submit Report): The user shall click the button titled "Submit Report" to submit the new report.

Exceptions: None.

3.2.17.4 Admin Privileges: Activity Report Review

Notice (Create Report): The system shall notify the user that a report was made.

3.2.17.5 Admin Privileges: Notify Emergency services

Button Click (Notify Services): The user shall click the button titled "Notify Emergency Services" to alert the authorities of the incident.

Exceptions: None.

3.2.18 Feature: Delete Activity Report

Stakeholders want users to be able to delete their own activity report.

3.2.18.1 Activity Report Delete Link

Button Click (Activity Report): The user shall click on the button titled "Delete Activity Report" to remove their own activity report from the list.

Exceptions: None.

3.2.18.2 Activity Report Delete Confirmation

Notice (Delete Report Confirmation): The system shall prompt the user to confirm their previous action before complying.

Exceptions: None.

3.2.18.3 Admin Privileges: Delete Activity Report

Button Click (Activity Report): The user shall click on the button titled "Delete Activity Report" to remove any activity report from the list.

Exceptions: None.

3.2.19 Feature: Edit Activity Report

Stakeholders desire that users can modify their own activity reports.

3.2.19.1 Edit Activity Report Link

Button Click (Edit Report): The user shall click the "Edit Report" button on the activity report to edit a report that they made.

Exceptions: None

3.2.19.2 Edit Activity Report Content

Date Field (Date): The user shall select the date of the activity in their report.

Time Field (Time): The user shall select the time of the activity in their report.

Drop down Field (Type): The user shall select the type of activity in their report (See Feature: Activity Reporting).

Map Field (Location): The user shall pin on map the location of the activity in their report.

Text Field (Description): The user shall enter the description of the activity in their report.

Exceptions:

Unselected Date Field (Date): The system shall display a red asterisk (*) next to field.

Unselected Time Field (Time): The system shall display a red asterisk (*) next to field.

Unselected Drop down Field (Type): The system shall display a red asterisk (*) next to field.

Empty Map Field (Location): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall deactivate the "Submit Edit" action.

3.2.19.3 Submit Edit Report Link

Button Click (Submit Report Edit): The user should click the button titled "Submit Edit" in the edit window of the activity report to submit the edits they made.

Notice (Submission Received): The system must notify the user that the changes have been made.

Exceptions: None

3.2.20 Feature: View Activity Report

Stakeholders want users to be able to view all or a subset of reports (with a filter based on report content).

3.2.20.1 Activity Report Link

Button Click (Activity Report): The user shall click on the activity report map pin to view the report in detail.

Exceptions: None.

3.2.20.2 Filter Report Link

Button Click (Filter Report): The user shall click on the button titled "Search" to filter reports.

Exceptions: None.

3.2.20.3 Filter Activity Report

Date Field (Date Range): The user shall select a date range (start date and end date) of the activity in their report.

Time Field (Time Range): The user shall select a time range (start time and stop time) of the activity in their report.

Drop down Field (Type): The user shall select the type of activity in their report (See Feature: Activity Reporting).

Map Field (Location): The user shall click and drag a circle on map to filter out the report outside of the circle.

Exceptions:

Incomplete Date Field (Date Range): The system shall display a red asterisk (*) next to field.

Incomplete Time Field (Time): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.21 Feature: Create Calendar Event

Stakeholders want users who desire to create community events to be able to use the application calendar to notify the community of the event.

3.2.21.1 Events Calendar Link

Button Click (Events Calendar): The user shall click the button titled "Events Calendar" to navigate to the community events calendar.

Exceptions: None.

3.2.21.2 Create Event Link

Button Click (Create Event): The user shall double-click a date in the calendar create a new community event.

Exceptions:

Button Click (Create Event): If the user performs a single click that denotes a view and not a create action.

3.2.21.3 Event Content

Text Field (Event Name): The user shall enter the title/name of the event.

Text Field (Event Description): The user shall enter the description of the event.

Time Field (Event Start time): The user shall select the start time of the event.

Time Field (Event End time): The user shall select the end time of the event.

Map Field (Location): The user shall click a map pin on the map to denote the location of the event.

Exceptions:

Empty Text Field (Event Name): The system shall display a red asterisk (*) next to field.

Empty Text Field (Event Description): The system shall display a red asterisk (*) next to field.

Unselected Time Field (Event Start time): The system shall display a red asterisk (*) next to field.

Unselected Time Field (Event End time): The system shall display a red asterisk (*) next to field.

Empty Map Field (Location): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall deactivate the "Submit Event" action.

3.2.21.4 Post Event to Calendar Link

Button Click (Submit Event): The user shall click the button titled "Done" and then the event will be posted in the calendar.

Exceptions: None.

3.2.22 Feature: View Calendar Event

Stakeholders want users to be able to view community events happening on a particular date.

3.2.22.1 Events Calendar Link

Button Click (Events Calendar): The user shall click the button titled "Events Calendar" to navigate to the community events calendar.

Exceptions: None.

3.2.22.2 Events Calendar View

Navigate (Events Calendar): The System shall display the current month by default with the ability to navigate forwards and backwards in one month increments.

Exceptions: None.

3.2.22.3 Events Calendar Date Link:

Button Click (Calendar Date): The user shall click on a particular date to display all the events happening on that particular date in the community.

3.2.23 Feature: Edit Calendar Event

Stakeholders want users to be able to make changes to the event that they created.

3.2.23.1 Events Calendar Link

Button Click (Events Calendar): The user shall click the button titled "Events Calendar" to navigate to the community events calendar.

Exceptions: None.

3.2.23.2 Edit Event Link

Button Click (Edit Event): The user shall click the button titled "Edit" on an event in the calendar to edit their own event.

Exceptions: None.

3.2.23.3 Event Content

Text Field (Event Name): The user shall enter the title/name of the event.

Text Field (Event Description): The user shall enter the description of the event.

Time Field (Event Start time): The user shall select the start time of the event.

Time Field (Event End time): The user shall select the end time of the event.

Map Field (Location): The user shall click a map pin on the map to denote the location of the event.

Exceptions:

Empty Text Field (Event Name): The system shall display a red asterisk (*) next to field.

Empty Text Field (Event Description): The system shall display a red asterisk (*) next to field.

Unselected Time Field (Event Start time): The system shall display a red asterisk (*) next to field.

Unselected Time Field (Event End time): The system shall display a red asterisk (*) next to field.

Empty Map Field (Location): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall deactivate the "Submit Event" action.

3.2.23.4 Post Event to Calendar Link

Button Click (Submit Event): The user shall click the button titled "Done" and then the event will be posted in the calendar.

Exceptions: None.

3.2.24 Feature: Delete Calendar Event

Stakeholders want users to be able to delete their own events.

3.2.24.1 Events Calendar Link

Button Click (Events Calendar): The user shall click the button titled "Events Calendar" to navigate to the community events calendar.

Exceptions: None.

3.2.24.2 Delete Event Link

Button Click (Edit Event): The user shall click the button titled "Delete" on an event in the calendar to delete their own the event.

Exceptions: None.

3.2.24.3 Event Delete Confirmation

Notice (Delete Event Confirmation): The system shall prompt the user to confirm their previous action before complying.

Exceptions: None.

3.2.24.4 Admin Privileges: Delete Event

Button Click (Delete Event): The user shall click on the button titled "Delete" to remove any event.

Exceptions: None.

3.2.25 Feature: Register Pet

Stakeholders want users to be able to register their pets on the HOA app.

3.2.25.1 Register Pet Link

Button Click (Register Pet): The user shall click on the button titled "Register Pet" in "Settings" to add a pet to their profile.

Exceptions: None.

3.2.25.2 Pet Information

Text Field (Name): The user must enter a the pets name.

Drop down Field (Species): The user must select a species from a drop down menu.

Text Field (Breed): The user must enter a breed.

Drop down Field (Color): The user must select a color from a drop down menu.

Numerical Field (Weight): The user must enter a number denoting a weight in pounds.

Exceptions:

Empty Text Field (Name): The system shall display a red asterisk (*) next to field.

Unselected Drop down Field (Species): The system shall display a red asterisk (*) next to field.

Empty Text Field (Breed): The system shall display a red asterisk (*) next to field.

Unselected Drop down Field (Color): The system shall display a red asterisk (*) next to field.

Empty Numerical Field (Weight): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.25.3 Submit New Pet Information Link

Button Click (Add Pet): The user must click a button titled "Add pet" when all information is correctly filled out.

Exceptions: None.

3.2.26 Feature: Modify Pet Registration

Stakeholders want users to have the ability to adjust their pet registration in the event they get a new pet or some information needs to be adjusted.

3.2.26.1 Edit Pet Link

Button Click (Edit Pet): The user shall click on the button titled "Edit" in "Pet Information" to modify a pets information in their profile.

Exceptions: None.

3.2.26.2 Pet Information

Text Field (Name): The user must enter a the pets name.

Drop down Field (Species): The user must select a species from a drop down menu.

Text Field (Breed): The user must enter a breed.

Drop down Field (Color): The user must select a color from a drop down menu.

Numerical Field (Weight): The user must enter a number denoting a weight in pounds.

Exceptions:

Empty Text Field (Name): The system shall display a red asterisk (*) next to field.

Unselected Drop down Field (Species): The system shall display a red asterisk (*) next to field.

Empty Text Field (Breed): The system shall display a red asterisk (*) next to field.

Unselected Drop down Field (Color): The system shall display a red asterisk (*) next to field.

Empty Numerical Field (Weight): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.26.3 Submit Modified Pet Information Link

Button Click (Submit): The user must click a button titled "Submit" when all information is correctly filled out.

Exceptions: None.

3.2.27 Feature: Delete Pet Registration

Stakeholders want users to have the ability to remove a pet registration in the event that they lose a pet or no longer own pets.

3.2.27.1 Delete Pet Link

Button Click (Remove): The user shall click the button titled "Remove" in "Pet Information" to remove pet from their profile.

Exceptions: None.

3.2.27.2 Pet Delete Confirmation

Notice (Delete Pet Confirmation): The system shall prompt the user to confirm their previous action before complying.

Exceptions: None.

3.2.28 Feature: Create Missing Pet Report

Pet Owners want the ability to report their pet missing.

3.2.28.1 Missing Pets Page Stakeholders want users to be able access the missing pets page which contains a list of missing pets in chronological order. Each element of this list includes the following information from the missing pet's profile:

- Pet's name
- Pet's species
- Pet's breed
- Pet's color
- Pet's size
- Owner's name
- Owner's Address
- Owner's contact information

3.2.28.2 Missing Pet Link

Button Click (Missing): The user shall click the button titled "Missing" in "Pet Information" to notify app users of a missing pet.

Exceptions: None.

3.2.28.3 Missing Pet Information

Date Field (Date Last Seen): The user shall select the date that the missing pet was last seen.

Time Field (Time Last Seen): The user shall select the time that the missing pet was last seen.

Map Field (Location Last Seen): The user shall select the location that the missing pet was last seen.

Text Field (Additional information): The user shall enter any additional information that will help the community find the missing pet.

Exceptions:

Unselected Date Field (Date Last Seen): The system shall display a red asterisk (*) next to field.

Unselected Time Field (Time Last Seen): The system shall display a red asterisk (*) next to field.

Empty Map Field (Location Last Seen): The system shall display a red asterisk (*) next to field.

Empty Text Field (Additional information): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.28.4 Pet Missing Confirmation

Notice (Missing Pet Confirmation): The system shall prompt the user to confirm their previous action before complying.

Confirmation (Missing Pet Confirmation): The user shall clicking "Okay" to upload the missing pet report to the alerts page.

Exceptions: None.

3.2.29 Feature: Edit Missing Pet Report

Stakeholders want users to have the ability to adjust their missing pet report with new information or corrected information.

3.2.29.1 Missing Pet Link

Button Click (Edit): The user shall click the button titled "Edit" in "Pet Information" to modify the missing pet information.

Exceptions: None.

3.2.29.2 Missing Pet Information

Date Field (Date Last Seen): The user shall select the date that the missing pet was last seen.

Time Field (Time Last Seen): The user shall select the time that the missing pet was last seen.

Map Field (Location Last Seen): The user shall select the location that the missing pet was last seen.

Text Field (Additional information): The user shall enter any additional information that will help the community find the missing pet.

Exceptions:

Unselected Date Field (Date Last Seen): The system shall display a red asterisk (*) next to field.

Unselected Time Field (Time Last Seen): The system shall display a red asterisk (*) next to field.

Empty Map Field (Location Last Seen): The system shall display a red asterisk (*) next to field.

Empty Text Field (Additional information): The system shall display a red asterisk (*) next to field.

Any Field Exception: The system shall display red text stating "Some field improperly filled".

3.2.29.3 Pet Missing Confirmation

Notice (Missing Pet Confirmation): The system shall prompt the user to confirm their previous action before complying.

Exceptions: None.

3.2.30 Feature: Close Missing Pet Report

Stakeholders want users to have the ability to remove their missing pet report once their pet is found.

3.2.30.1 Close Missing Pet Report Link

Button Click (Remove): The user shall click the button titled "Remove" in "Pet Information" to remove pet from missing pets list.

Exceptions: None.

3.2.30.2 Missing Pet Report Remove Confirmation

Notice (Delete Pet Confirmation): The system shall prompt the user to confirm their previous action before complying.

Exceptions: None.

3.3 Non-Functional Requirements

3.3.1 Usability

3.3.1.1 Notification Features Users should be able to interact with notification features as follows:

- The notification feature should direct users to activity reports, emergency and high-profile alerts, chat room, missing pet.
- GPS feature is used to show locations of the crimes reported on map.

3.3.1.2 Emergency and High-Profile Alerts page In case of house on fire, the surrounding users must be able to raise alerts with 99% success rate. The system should be capable to receive the requests from multiple users and should act accordingly.

3.3.2 Robustness

3.3.2.1 Re-send Verification Email The user shall have the option to re-send the verification email required during the account creation process.

3.3.2.2 Recoverable Storage In the event of mass data loss, 98% of user data must be recoverable from another source.

3.3.3 Reliability

- The system shall issue notifications within 1 minute at least 98% of the time.
- The system shall return correctly filtered activity reports within 5 seconds.

3.3.3.1 Acceptable Failing Tests

Prior to deployment, the system should fail no less than:

- 5% of integration tests.
- 2% of unit tests.

3.3.4 Performance

3.3.4.1 Page Loading Speed

95% of webpages shall load in less than 5 seconds during regular operation where users are expected to have a download speed of at least 20 Mbps or faster.

3.3.4.2 Login Page

- When multiple users enter their credentials and attempts log in simultaneously, the system should be capable to authenticate all those attempts and should acknowledge back to the users.

3.3.4.3 Events Calendar Page

- When multiple users attempts to create events on same/different date simultaneously, the system should be capable to create events for multiple users.
- When multiple users attempts to view the same/different event simultaneously, the system should be capable to display all events requested from multiple users.
- When multiple users attempts to modify their events on the same/different date simultaneously, the system should be capable to modify the requests for multiple users.
- When multiple users attempts to delete their events on same/different date simultaneously, the system should be capable to delete events for multiple users.

3.3.4.4 Reports Page

- The system should be capable to display map with pinpoint locations to multiple users simultaneously.
- The system should be capable to display a single report to multiple users simultaneously.

3.3.4.5 Forums Page

- In Reports tab, when multiple users post comments on same/different report simultaneously, the system should be capable to display all the comments from multiple users.
- In Missing pets tab, when multiple users send a message simultaneously to the pet's owner, the system be capable to send those multiple messages to the owner.
- In Chat Room tab, when multiple users post multiple messages simultaneously, the system should be capable to receive and display all the messages posted.

3.3.4.6 Logout

- When multiple users attempts to logout of their accounts simultaneously, the system should be capable to process multiple logout requests and redirect their page to log in page.

3.3.5 Maintainability

3.3.5.1 System Maintenance System maintenance, if required, should be completed between the hours of 12:01 AM - 5:00 AM EST Mondays.

3.3.5.2 Future Feature Development The system should be designed in such a way that, future developers should be able to add new features.

3.3.5.3 Code Headers All modules of software must contain the following information at the top of files:

- File name
- Description of Contents
- Date created
- Developer in charge of module

3.3.6 Availability

The system should be connected to the following:

- Internet
- GPS

3.3.6.1 Concurrent Users The system shall accommodate at least 50 concurrent users for each neighborhood group at a time.

3.3.6.2 System Uptime The system shall be available 98% of the time between the hours of 5:00 AM EST and Midnight EST Tuesday through Sunday.

The system shall be available 90% of the time between the hours of 5:00 AM EST and Midnight EST on Mondays.

3.3.7 Security

3.3.7.1 Secure Login Account login information must be transmitted using a secure encrypted protocol such as SSL.

3.3.7.2 Email addresses

- Must include username.
- Must include an "@" sign.
- Must include domain name.
- Must include top-level domain (such as .com, .org, .net, et cetera).

3.3.7.3 Passwords

- Must be at least 12 characters long.
- Must include at least one number.
- Must include at least one special character.

3.3.7.4 Encrypted Chat Room Messages The messages in the chat room should be encrypted.

3.3.7.5 Inactivity Logout The system shall automatically log a user out of their account after 30 minutes of inactivity.

3.3.7.6 Inactivity Warning The system shall warn the user of inactivity 5 minutes prior to automatically logging a user out of their account

3.3.7.7 Editing User Information Users logged into their accounts shall be the only ones allowed to modify their personal information.

3.3.7.8 Privilege Modification The elevation or downgrading of privileges shall only be done by users with administrative privileges to do so.

3.3.8 Portability

The system should be designed in such a way that this application can be accessible through mobile phones and desktops.

3.3.8.1 iOS Support The system shall be supported on iOS devices.

3.3.8.2 Android Support The system shall be supported on Android devices.

3.3.8.3 Desktop Web Browser Support The system shall support the latest versions of the web browsers:

- Chrome
- Firefox
- Edge
- Safari

3.3.8.4 Mobile Web Browser Support The system may support the latest versions of mobile view for the following web browsers:

- Chrome
- Safari

3.3.8.5 Text Format

- All standard ASCII characters should be permitted in text fields.

3.3.8.6 Time Format

- All times must be displayed in 24-hour format as follows "HH:MM".

3.3.8.7 Date Format

- All dates must be displayed in US-format as follows "MM/DD/YYYY".

3.3.8.8 Location Format

- All locations must include street number in the format "[NUMBER] [STREET NAME]".
- All locations must include city name in the format "[CITY NAME]".
- All locations must include state name in the format "LL".
- All locations must include zip code in the format "NNNNN".

3.4 Design Constraints

There are no identified software design constraints imposed by hardware.

A - Lofi Diagrams

This section contains Lo-fi diagrams created during requirements elicitation.

A.1 - Account Creation

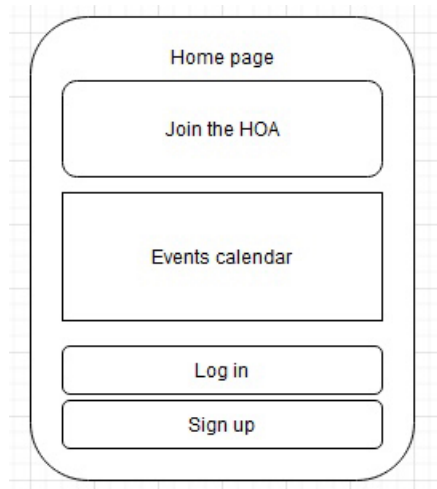


Figure 14: Home Page Lo-Fi Diagram: This describes the view at the landing page of the Neighborhood Watch web app

A.2 - Logged-in User Page Lo-Fi Diagram

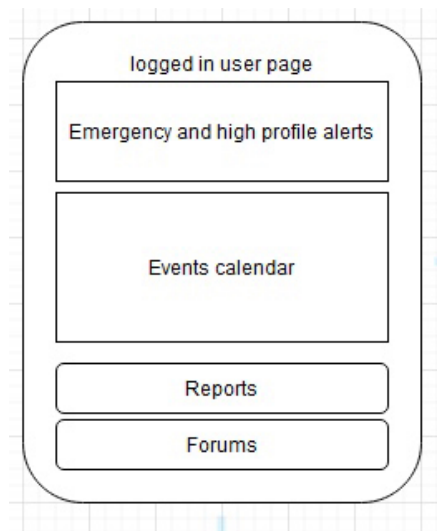


Figure 15: Logged-in user page Lo-Fi Diagram: This describes the view of the logged-in user page where of the Neighborhood Watch web app. See figure 3 for the reports view.

A.3 - Reports Page Lo-Fi Diagram

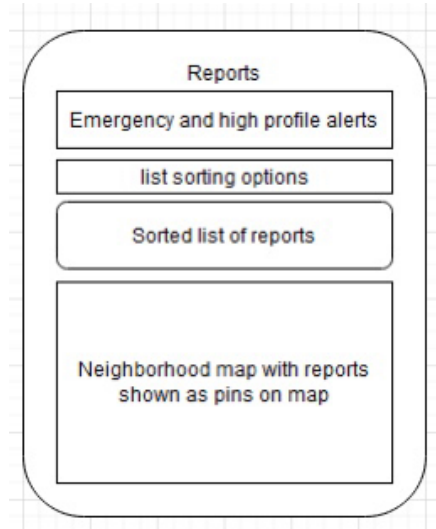


Figure 16: Reports page Lo-Fi Diagram: This describes the view of the reports with a map that shows the location of the Neighborhood Watch web app.

A.4 - Sign-Up Page Lo-Fi Diagram

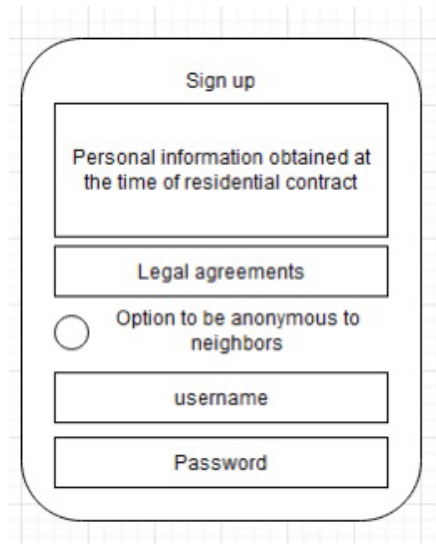


Figure 17: Sign-up page Lo-Fi Diagram: This describes the view for users that need to sign up for the Neighborhood Watch app for the first time.

A.5 - Settings Page/Tab Lo-Fi Diagram

The diagram shows a light blue rectangular box representing the 'Settings' page. Inside the box, the title 'Settings' is centered at the top. Below it, the section 'Personal Information:' is followed by 'Name: Din Djarin' and 'Address: 123 Mandalore St'. A radio button group for 'Anonymous' is shown with the 'X' selected in the first box, followed by 'Yes' and 'No' with empty boxes. Below this, the text 'Access Level: HOA Member' is displayed. Further down, 'Blocked User List: Moff Gideon' is listed. At the bottom of the settings area, four options are listed: 'Delete Account', 'Change User Access Level', and 'Remove User'.

Settings

Personal Information:

Name: Din Djarin

Address: 123 Mandalore St

Anonymous ☒ Yes ☐ No

Access Level: HOA Member

Blocked User List: Moff Gideon

Delete Account

Change User Access Level

Remove User

Figure 18: Settings Page/tab Lo-fi Diagram: This describes the view for users that have access to administrative options. If they do not have administrative options, "Change user level" and "Remove User" will not appear.

Appendix B - Use Cases

This section contains use cases created for requirements elicitation.

B.1 - Account Creation

B.1.1 - Use Case Scenarios

This section contains use cases specific to the account creation process.

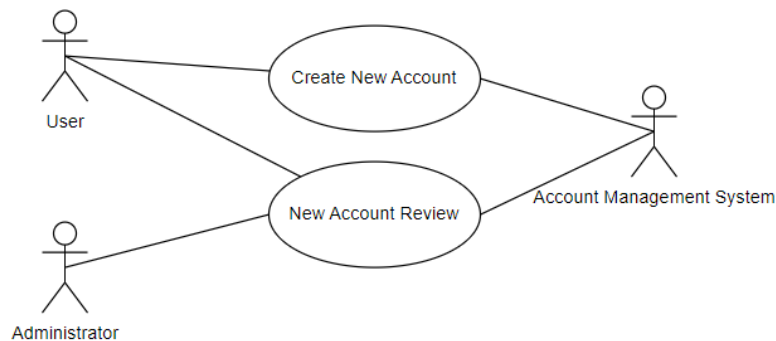


Figure 19: Account Creation Use Case Diagram

B.1.1.1 - Create New Account

User Task: Create New Account

Actors: User, System

Description: A new user creates and verifies a new account on the app.

Precondition: User is presented with the account creation form.

Postcondition: User has created and verified their new account.

Scenario 1: User creates a new account

1. New user clicks the "Create Account" button.
2. System displays the account creation form.
3. User enters their first name, last name, unique email address, and valid password
4. User clicks the "Submit" button.
5. System validates the input and acknowledges a successful submission.
6. System sends a verification email to the user.
7. User navigates to their verification email, and clicks the verification link labeled "Verify Account".
8. System verifies the user's account and activates it.
9. Once verified, user is navigated to the "Neighborhood Application Page".
10. System prompts user to provide relevant address information including:
 - Address line 1, Address line 2(optional), City, State, and Zip code
 - Any additional documentation providing proof of residency
11. User enters residency documentation and clicks the button labeled "Submit"

12. System validates the input and sends user application to the appropriate neighborhood group for review.
13. System notifies user that their information has been received and is under review and that user will receive an email once their application has been verified.

Scenario 2: User attempts to submit incomplete or invalid account information

1. User fails to fill in the required fields such as name, email, or password
2. System detects the errors, and disallows submitting the information.
3. System displays an error message and highlights the sections that require changes in order for submission to be considered valid.
4. User corrects the errors and proceeds with the account creation process as described in scenario 1.

Scenario 3: User enters a weak or common password

1. During the initial account creation phase, user enters a weak or commonly used password.
2. System detects the password error and disallows submission.
3. System displays an error and informs the user that the password provided does not meet requirements.
4. System provides password strength requirements and recommendations.
5. User enters a stronger password in accordance to requirements and submits form.
6. User proceeds with account creation process as described in scenario 1.

Scenario 4: User attempts to submit incomplete or invalid address information

1. While on the "Neighborhood Application Page", user enters invalid address information.
2. Once form is completed, user presses the button labeled "Submit".
3. System checks the provided address against its database of registered addresses.
4. System recognizes address is already registered, nonexistent, or invalid, and displays an error message informing user that the address information is invalid.
5. User corrects the address information and presses the button labeled "Submit".
6. System authenticates new address information and confirms validity of address provided.
7. User now proceeds with account creation process from step 12 of scenario 1.

Alternate Flow 1: User requires a new verification email to be sent

1. While on the account creation page, User clicks the button labeled "Resend Verification Email"
2. System sends a new verification email to the user's email address.
3. System invalidates the previous verification email
4. User receives the new verification email, clicks it, and proceeds with account creation.

Alternate Flow 2: User attempts to use an expired email verification link

1. User opens an expired verification email they were sent by System
2. User clicks the "Verify Account" link in the expired verification email.
3. System recognizes that the link is expired and redirects user to an error page.
4. Error page explains to user that the verification link has expired.

5. User clicks a link on the error page labeled "Send new verification email"
6. System sends a new verification email to the email address connected to the expired verification link.
7. User navigates to the new verification email they were sent and clicks the "Verify Account" link.
8. User has now verified account and proceeds from step 9 of Scenario 1.

B.1.1.2 - New Account Review

User Task: Review New Accounts

Actors: Admin, User, System

Description: Admin reviews new account applications and either approves or denies them.

Precondition: New account applications are in the review queue.

Postcondition: Applications are either approved or denied, and applicants are notified via email.

Scenario 1: Admin reviews and approves new user application

1. Admin views the list of new applications in the system
2. Admin selects a user's new application to review, and the applicant's information is displayed.
3. Admin reviews the user's information and approves the application by clicking the "Approve".
4. System processes the decision, updates the account management database (if approved).
5. and sends an email to the user with the decision.
6. The new user now has access to the neighborhood's app community to view and respond to events, participate in forums, and receive community alerts.

Scenario 2: Admin rejects new user application

1. Admin views the list of new applications in the system.
2. Admin selects a user's new application to review, and the applicant's information is displayed.
3. Admin reviews the user's information and determines that the application cannot be approved.
4. Admin clicks the "Reject" button and enters a reason for the rejection.
5. The user receives an email explaining that they are not granted access to the neighborhood's app community and are provided instructions on how to reapply in the future if desired.

Scenario 3: Admin requests additional information from user

1. Admin views the list of new applications in the system.
2. Admin selects a user's new application to review, and the applicant's information is displayed.
3. Admin reviews the user's information and determines that additional information is needed to make a decision.
4. Admin clicks the button labeled "Request Additional Information" and enters the specific information needed.
5. System sends an email to the user requesting the additional information.
6. User submits the additional information via the app or email.
7. Admin reviews the additional information and approves or denies the application as in Scenario 1 or 2.

B.1.2 - Misuse Case Scenarios

This section contains misuse cases specific to the account creation process

B.1.2.1 - Unauthorized Account Verification

User Task: Unauthorized Verification

Actors: Attacker, System

Description: An attacker intercepts the verification email and attempts to gain unauthorized access to the account.

Precondition: Attacker intercepts the verification email.

Postcondition: System detects unauthorized access attempts and implements countermeasures.

Scenario: Attacker attempts unauthorized account verification

1. Attacker intercepts the verification email intended for the user.
2. Attacker clicks the "Verify Account" link in the intercepted verification email.
3. The link redirects the attacker to the account verification page.
4. System monitors for suspicious activity, such as multiple verification attempts from different IP addresses or an attempt from an unexpected location.
5. If the system detects unauthorized access attempt(s), it implements countermeasures like temporarily locking the account, requesting additional verification from the user (e.g., two-factor authentication), or notifying the user and system administrators about the suspicious activity.
6. If the system does not detect any suspicious activity, the attacker successfully verifies the account, potentially gaining unauthorized access.

B.1.2.2 - System Overload via Mass Account Registration

User Task: Overload System by Creating Multiple Accounts

Actors: Attacker, System, System Administrators

Description: An attacker attempts to overload the account review server by submitting a large number of new accounts simultaneously

Precondition: Attacker has access to the account creation form.

Postcondition: System detects the abusive behavior and implements countermeasures.

Scenario: System becomes under threat due to attack by mass account registration

1. Attacker uses a script or bot to automate the process of filling out the account creation form with different sets of valid information
2. Attacker submits a large number of account creation forms in a short period of time, with the intention of overloading the system and degrading its performance.
3. System detects the abusive behavior (e.g., by identifying unusual patterns or rate of submissions) and implements countermeasures, such as blocking the attacker's IP address, requiring additional verification like CAPTCHA, or rate-limiting the account creation process.
4. System administrators are alerted of the potential attack, allowing them to monitor the situation and take further action if necessary.

B.2 - Account Administration

This section details the use cases associated the administration of all user accounts.

B.2.1 - Scenarios

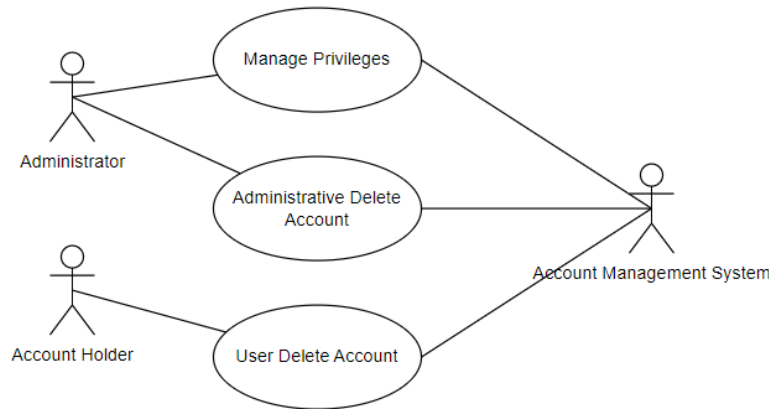


Figure 20: Account Administration Use Case Diagram

This section contains scenarios for account administration

B.2.1.1 Manage Privileges

User Task: Account Privilege Management

Actors: Account Administrator, Account Management System

Description: The Account Administrator wants to change the the privileges allowed for an account through the Account Management System.

Precondition: Account Administrator is authenticated with the account management system. There is at least one user account registered with the Account Management System.

Post-condition: User account privileges are modified.

Scenario 1: Account Administrator upgrades user account privileges

1. Account Administrator navigates to account modification window.
2. Account Administrator selects an account from a list of users provided by the Account Management System.
3. The Account Management System provides to the Account Administrator a list of privilege policies for the selected account.
4. The Account Administrator selects from a list of policies to apply to the selected account.
5. The Account Administrator clicks a button to apply changes.
6. The Account Management System updates the changes for the selected account to be applied as needed.

Scenario 2: Account Administrator downgrades user account.

1. Account Administrator navigates to account modification window.

2. Account Administrator selects an account from a list of users provided by the Account Management System.
3. The Account Management System provides to the Account Administrator a list of privilege policies for the selected account.
4. The Account Administrator selects from a list of policies which are currently applied to the selected user account.
5. The Account Administrator clicks a button to remove policies from the selected user account.
6. The Account Management System updates the changes for the selected account and removes the policies for that account.

B.2.1.2 Delete Account by Administrator

User Task: Administrative Account Deletion

Actors: Account Administrator, Account Management System

Description: The Account Administrator wants to delete an account from the system.

Precondition: Account Administrator is authenticated with the account management system. There is at least one user account registered with the Account Management System.

Post-condition: User account is deleted.

Scenario 1: Account Administrator deletes account.

1. Account Administrator navigates to account deletion window.
2. Account Administrator selects an account from a list of users provided by the Account Management System.
3. The Account Management System provides to the Account Administrator relevant account information, status, and applied privilege policies.
4. The Account Administrator clicks a button to delete the selected account.
5. The Account Management System updates the system to disable the account. A basic record of the account should be kept by the system for auditing purposes.

Alternative Flow: Account is not allowed to be deleted.

1. If the account is protected from deletion, display a warning to the Account Administrator informing them the action is not allowed.
2. The Account Management System is retained.

B.2.1.3 User Account Deletion

User Task: User Account Modification

Actors: User, Account Management System

Description: The User wants to delete their account.

Precondition: User is authenticated with the account management system.

Post-condition: User account is disabled.

Scenario 1: User deletes account.

1. User navigates to account settings.
2. The system displays to the User their current account information as well an option to delete their account.

3. The User clicks the button to delete their account.
4. The system prompts the User with an "are you sure?" screen with an option to either go back to the previous screen or continue with account deletion.
5. The User clicks the option to continue with account deletion.
6. The system sends the account deletion request to the Account Management System.
7. The Account Management System archives the account and places it in a disabled state.
8. The system logs the User out of their account.

Alternative Flow: Go back to previous screen and do not delete account.

1. The User clicks the option to go back to the previous screen.
2. No changes are made to the User's account.

B.3 - Account Actions

This section details the use cases associated with general user account activities

B.3.1 - Scenarios

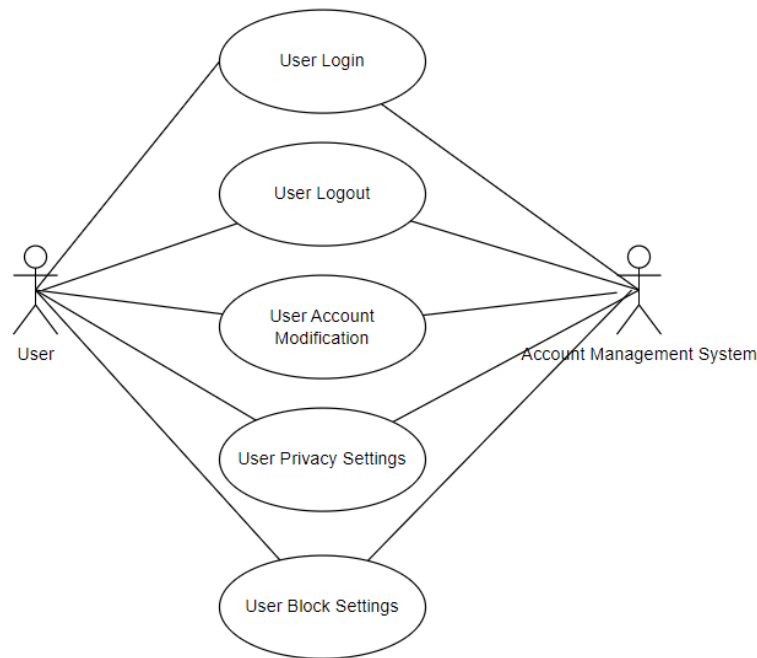


Figure 21: Account Actions Use Cases

B.3.1.1 Account Modes

1. Normal user account
2. Anonymous Mode
3. Guest Account

B.3.1.2 User Login/Logout

B.3.1.3 User Account Modification

User Task: User Account Modification

Actors: User, Account Management System

Description: The User wants to modify their account information.

Precondition: User is authenticated with the account management system.

Post-condition: User account information is modified.

Scenario 1: User modifies account information.

1. User navigates to account settings.
2. The system displays to the User their current account information as well as buttons which allow the user to edit fields.
3. The User clicks a button to edit a piece of information.
4. The system provides an input field for the user along with a button for submitting changes.
5. The User fills the input field and clicks a button to submit changes.
6. The system sends the change request to the Account Management System, which updates the information for the account which submitted the request.
7. The system displays the changed information to the User.

Alternative Flow: Invalid input

1. The system rejects an invalid input the User submitted.
2. No changes are made to the User's account.

B.3.1.4 User Privacy Settings

User Task: User Privacy Settings

Actors: User, Account Management System

Description: The User wants to modify their account privacy settings.

Precondition: User is authenticated with the account management system.

Post-condition: User account privacy settings are modified.

Scenario 1: User modifies account privacy settings.

1. User navigates to account privacy settings.
2. The system displays to the User their current account privacy settings as well as buttons which allow the user to edit their privacy settings.
3. The User clicks on input fields to edit their settings.
4. The User clicks a button to submit changes for their privacy settings.
5. The system sends a change request to the Account Management System with the User's updated privacy settings.
6. The system displays the changes made to the User.

B.3.1.5 User Block Settings

User Task: User Block Settings

Actors: User, Account Management System

Description: The User wants to block accounts.

Precondition: User is authenticated with the account management system.

Post-condition: Any posts by specified users are blocked.

Scenario 1: User adds another user to their account block settings.

1. User navigates to the block users settings.
2. The system displays to the User currently a selectable list blocked users with a remove button next to it as well as a input text box with an add button next it.
3. The User inputs the username of the account they wish to block.
4. The User clicks the button to add the account to the list of blocked users.
5. The system sends a change request to the Account Management System.
6. The Account Management System updates the changes for the account's block users list.
7. The system displays the changes to the User.

Scenario 2: User removes a user from their account block settings.

1. User navigates to the block users settings.
2. The system displays to the User currently a selectable list blocked users with a remove button next to it as well as a input text box with an add button next it.
3. The User selects account(s) in the list they would like to remove from the blocked users list.
4. The User clicks the remove button.
5. The system sends a change request to the Account Management System.
6. The Account Management System removes the specified accounts from the User's block user list.
7. The system displays the changes to the User.

B.4 - Neighborhood Watch

This section details the use cases associated with the Neighborhood Watch group

B.4.1 - Scenarios**B.4.1.1 Create Flash Alert**

User Task: Create a Flash Alert.

Actors: Neighborhood Watch Member, Notification Services, Users

Description: Neighborhood Watch Member submits a Flash Alert to all account within the neighborhood.

Precondition: Neighborhood Watch Member has permissions to submit a Flash Alert.

Post-condition: A new Flash Alert is created and sent to all users in the neighborhood group.

Scenario 1: Neighborhood Watch Member submits a Flash Alert.

1. The Neighborhood Watch Member navigates to the Flash Alert page.

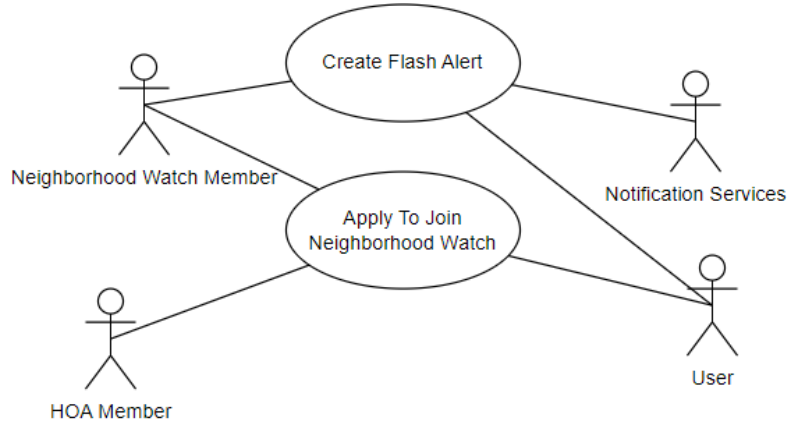


Figure 22: Neighborhood Watch Use Case Diagram

2. The Neighborhood Watch Member fills a form with required and optional information regarding the Flash Alert.
3. The Neighborhood Watch Member clicks a button to send the Flash Alert.
4. The system checks the privileges of the Neighborhood Watch Member.
5. If the Neighborhood Watch Member has sufficient privileges, the Flash Alert is sent to Notification Services for the neighborhood.
6. Notification Services broadcasts the Flash Alert to Users in the neighborhood.
7. Users within the neighborhood group and with filter settings which approve of notifying the User for the level of Flash Alert receive the Flash Alert.

Alternative Flow: The Neighborhood Watch Member does not have permissions to send the Flash Alert.

1. The system displays a warning message to the Neighborhood Watch Member that they do not have permissions to send the Flash Alert.

Alternative Flow: User has Flash Alert filter settings.

1. The User does not receive a Flash Alert.

B.4.1.2 Apply to Join Neighborhood Watch

User Task: Creating a Neighborhood Watch Application

Actors: User, a Neighborhood Watch or HOA Member

Description: A Homeowner fills out and submits a form to become a Neighborhood Watch member. This form needs to be approved by a current Neighborhood Watch or HOA member prior to the application's acceptance.

Precondition: A User has the permissions of a Homeowner.

Post-condition: The same User has the permissions of a Neighborhood Watch member.

Scenario 1: Application Successful

1. A Homeowner navigates to the account settings page.
2. The Homeowner clicks a button labeled "join the Neighborhood watch".
3. The Homeowner fills out the application form that appears then clicks a button labeled "Submit".

4. An active HOA member or Neighborhood Watch member receives a app notification: "Homeowner name wishes to join the watch!".
5. The HOA/Watch member can navigate to the Homeowner's account settings.
6. The Homeowner's application is displayed at the bottom of their account page (only visible to HOA/Watch members) along with two buttons labeled "accept" and "deny".
7. The HOA/Watch member clicks "accept" and the Homeowner becomes a Neighborhood Watch member.

Scenario 2: Application Denied

1. A Homeowner navigates to the account settings page.
2. The Homeowner clicks a button labeled "join the Neighborhood Watch".
3. The Homeowner fills out the application form that appears then clicks a button labeled "Submit".
4. An active HOA member or Neighborhood Watch member receives a app notification: "Homeowner name wishes to join the watch!".
5. The HOA/Watch member can navigate to the Homeowner's account settings.
6. The Homeowner's application is displayed at the bottom of their account page (only visible to HOA/Watch members) along with two buttons labeled "accept" and "deny".
7. The HOA/Watch member clicks "Deny" and the Homeowner receives a notification stating their application has been denied.

Alternative Flow 1: Application Failed to Send/Expired

1. A Homeowner navigates to the account settings page.
2. The Homeowner clicks a button labeled "join the Neighborhood Watch".
3. The Homeowner fills out the application form that appears then clicks a button labeled "Submit".
4. The application does not reach a HOA/Watch member or the application is ignored for whatever reason.
5. After 48 hours of no action, the Homeowner receives a notification: "We are sorry, your application to become a member of the Neighborhood Watch was lost or expired. We suggest applying again. If problems persist, please speak to an HOA representative."

B.5 - User Forum

This section details the use cases associated with the user forum

B.5.1 - Scenarios

B.5.1.1 Create Forum Post

User Task: Create a new forum post.

Actors: User, System

Description: User creates a new post in the forum section of the application.

Precondition: User has a verified account and is at the forum section of the application.

Post-condition: A new forum post is created with the user's content.

Scenario: User successfully creates a new forum post

1. User navigates to the forum section of the application.

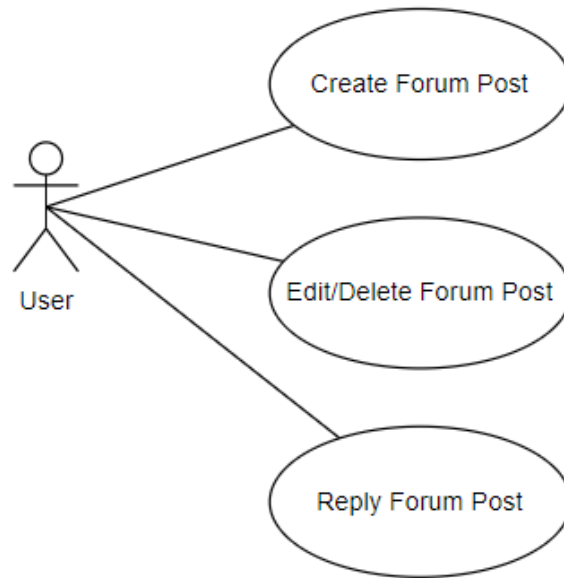


Figure 23: User Forum Use Case Diagram

2. User clicks the "Create Post" button.
3. A form appears where the user can enter a title and the content of their post.
4. User adds a post title and enters content in post body and presses the "Submit" button.
5. The new forum post is created and displayed on the forum page.

Alternate Flow: User attempts to submit a post that is missing body or title content

1. User follows the Create Forum Post process and is at the form to add a new post.
2. User adds no content to the post title, body, or both and presses "Submit" button.
3. The post is not created, User is not navigated from the form, and an error informing them they are missing post information is displayed.
4. User then enters relevant information and continues normal flow of form creation.

B.5.1.2 Edit/Delete Forum Post

User Task: Modify or remove a forum post.

Actors: User, System

Description: User modifies or removes their own forum post.

Precondition: User has a verified account and has created a forum post. The forum post user wants to edit is owned/created by the user.

Post-condition: The forum post has been successfully edited or deleted.

Scenario 1: User edits their forum post

1. User navigates to the forum post they want to edit.
2. User clicks the "Edit" button.
3. The post content is displayed in an editable form.

4. User makes the desired changes to the post content and clicks the "Submit" button.
5. The forum post is updated with the new content.

Scenario 2: User deletes their forum post

1. User navigates to the forum post they want to delete.
2. User clicks the "Delete" button.
3. A confirmation box appears asking if they wish to delete the post.
4. The user presses "Confirm", and the post is removed.

Alternate Flow: Cancel update/delete

1. The user decides they don't want to update or delete their forum post and clicks the "Cancel" button.
2. The user is redirected to the forum page where the post remains unchanged.

B.5.1.3 Reply to Forum Post

User Task: Reply to a forum post.

Actors: User, System

Description: User replies to an existing forum post.

Precondition: User has a verified account and is at the forum section of the application.

Post-condition: A new reply is added to the forum post.

Scenario 1: User replies to a forum post

1. User navigates to the forum post they want to reply to.
2. User clicks the "Reply" button.
3. A form appears where the user can enter the content of their reply.
4. User enters the reply content and presses the "Submit" button.
5. The new reply is added to the forum post and displayed on the forum page.

Scenario 2: User replies to a comment on a forum post

- 1.

Alternate Flow: User cancels reply

- 1.

B.6 - Event Calendar

This section details the use cases associated with the Event Calendar

B.6.1 - Scenarios

B.6.1.1 Create Event

User Task: Create a new event on the app's calendar.

Actors: User, System

Description: User creates a new event on the app's calendar.

Precondition: User has a verified account and is logged into the app.

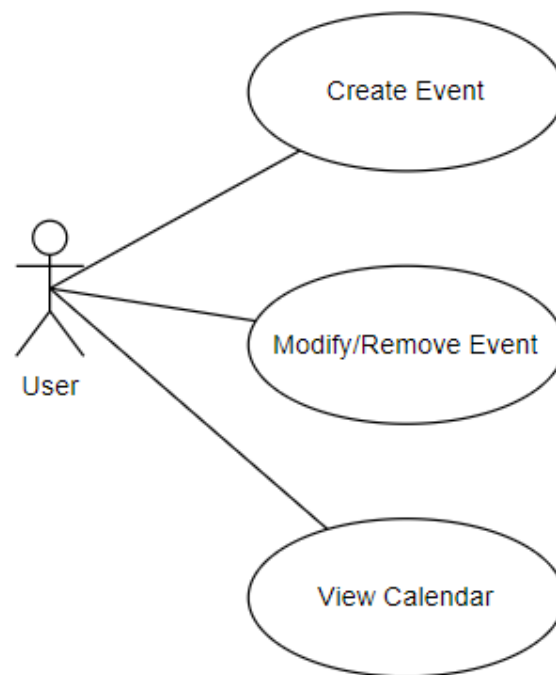


Figure 24: Event Calendar Use Case Diagram

Postcondition: The event is created and added to the app's calendar.

Scenario: Successful event creation

1. User navigates to the event calendar section of the app.
2. User clicks on the "create event" button.
3. A form appears where the user can enter the details of the event, such as the event name, date, time, location, and description.
4. The user submits the form and the event is created and added to the app's calendar with the user as the owner.

Alternate Flow: Invalid input

1. In step 3 of the main flow, if the user enters invalid input or leaves a required field blank, an error message is displayed and the user is prompted to correct the input and resubmit the form.
2. The user corrects the input and resubmits the form.
3. The system validates the input and creates the event as in the main flow.

B.6.1.2 Modify/Remove Event

User Task: Modify or remove an existing event on the app's calendar.

Actors: User, System

Description: User modifies or removes an existing event on the app's calendar.

Precondition: User has a verified account and is logged into the app. User is the owner/creator of the event.

Postcondition: The event is modified or removed from the app's calendar.

Scenario 1: User edits an existing event

1. User navigates to the event calendar section of the app.
2. User selects an event they own and want to edit.
3. User clicks on the "edit" button.
4. A form appears where the user then edits the details of the event.
5. The user submits the form and the edited event is displayed on the app's calendar.

Scenario 2: User removes an event from the calendar

1. User navigates to the event calendar section of the app.
2. User selects an event they own and want to remove.
3. User selects the "edit" button, and the form to edit the event appears.
4. User scrolls to the bottom of the page and clicks the button labeled "Delete"
5. User is navigated back to the event calendar where the item is no longer present.

Alternate Flow: User attempts to edit remove an event they did not create

1. User navigates to the event calendar and selects an event they did not create.
2. The page for the event opens and user looks for the "edit" button.
3. The "edit" button does not exist for users who did not create the event and the user is unable to edit or delete the event.

B.6.1.3 View Calendar

User Task: View the app's calendar in different views.

Actors: User, System

Description: User views the app's calendar in different views.

Precondition: User has a verified account and is logged into the app.

Postcondition: The calendar is displayed in the selected view.

Scenario 1: User views monthly calendar

1. User navigates to the event calendar section of the app.
2. User selects the "monthly" view option.
3. The app displays the calendar in a monthly view, showing all the events scheduled for the selected month.

Alternate Flow: user selects "weekly" view option

1. While at the event calendar section of the app, the user selects the "weekly" view
2. The app displays the calendar for the selected week, showing all scheduled events for that week.

Scenario 2: User views daily calendar events

1. User navigates to the event calendar section of the app.
2. User selects a day from the calendar.
3. The app displays the daily view, showing all the events scheduled for the selected day.

B.7 - Activity Reports

This section details the use cases associated with neighborhood activity reporting.

B.7.1 - Scenarios

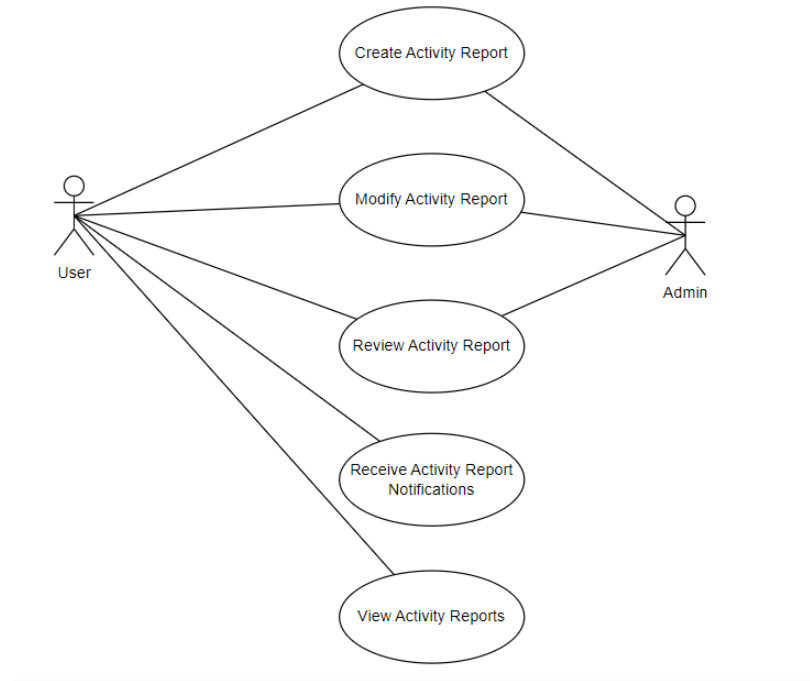


Figure 25: Activity Reports Use Case Diagram

B.7.1.1 - Create Activity Report

User Task: Create Activity Report

Actors: User, Admin, System

Description: User creates an Activity Report, filling in the necessary details.

Precondition: User is logged in and on the home page.

Postcondition: Activity Report is created, and notifications are sent to relevant users.

Scenario 1: User creates new Activity Report without notifying emergency services.

1. User clicks the "Create Activity Report" button on the home page.
2. System displays the Activity Report form with fields for Date, Time, Type, Location, and Description.
3. User selects the type of activity from a drop-down list.
4. User fills in the required fields.
5. If User chooses to post the report anonymously, they check a checkbox labeled "Post anonymously".
6. User leaves "notify emergency services" box unchecked.
7. User submits the form.

8. System saves the Activity Report and sends notifications to users with Activity Report enabled.
9. System sends a request for review to admins and, if it's a 'suspicious activity' report, to Neighborhood Watch users.

Scenario 2: User chooses to notify emergency services from Activity Report

1. User follows Scenario 1 up to step 6.
2. User checks the box labeled "notify emergency services".
3. System checks if the user has reached their limit for emergency notifications within a time-frame.
 - (a) If the user has exceeded their limit, System displays an error message and disallows notifying emergency services.
4. If the limit is not reached, a confirmation box pops up asking the user to confirm notifying emergency services.
5. User confirms and submits the form.
6. System saves the Activity Report and sends notifications to users with Activity Reporting notifications enabled.
7. System sends a request for review to admins and, if it's a 'suspicious activity' report, to Neighborhood Watch users.
8. System waits for admin review before notifying emergency services and notifies user of admin response.

Alternate Flow: User does not fill in a required field

1. If the user does not fill in a required field, the system displays an error message and asks the user to complete the missing information.

Alternate Flow: User chooses to notify emergency services

B.7.1.2 - Modify Activity Report (Edit or Delete)

User Task: Delete Activity Report (Edit or Delete)

Actors: User, Admin, System

Description: User or Admin edits or deletes an existing Activity Report.

Precondition: User or Admin is viewing the Activity Report they want to modify.

Postcondition: Activity Report is either modified or deleted.

Scenario 1: Edit Activity Report

1. User or Admin clicks the "Edit" button on the Activity Report page.
2. System displays the Activity Report form pre-filled with the current information.
3. User or Admin modifies the fields as necessary.
4. User or Admin submits the updated form.
5. System saves the updated report and displays a success message.

Scenario 2: Delete Activity Report

1. User or Admin clicks the "Delete" button on the Activity Report page.
2. System prompts for confirmation to delete the report.
3. User or Admin confirms the deletion.

4. System deletes it and displays a success message.

B.7.1.3 - Review Activity Report

User Task: Review Activity Report

Actors: Admin, System, User

Description: Admin reviews a submitted Activity Report.

Precondition: Admin has received a request to review an Activity Report.

Postcondition: Activity Report is reviewed and updated or removed if necessary.

Scenario 1: Admin edits Activity Report

1. Admin views the Activity Report details.
2. Admin decides if any changes are needed and, if so, edits the Activity Report.
3. Admin marks the Activity Report as reviewed.

Scenario 2: Activity Report contains information that breaks community rules

1. Admin views the Activity Report details.
2. Admin decides the Activity Report violates community rules.
3. Admin deletes the Activity Report or contacts the user for further clarification.

Scenario 3: Admin reviews Activity Report with emergency notification

1. Admin receives a request to review an Activity Report with emergency service notification.
2. Admin views the Activity Report details.
3. Admin decides if the situation warrants notifying emergency services.
4. If the admin approves the notification, the system sends a notification to the appropriate emergency services with the relevant Activity Report details.
5. System provides a status update to the user that submitted the report after admin completes review.

B.7.1.4 - Receive Activity Report Notifications

User Task: Receive Activity Report notifications

Actors: User, System

Description: User receives or does not receive notifications about new Activity Reports.

Precondition: User has Activity Report notifications enabled.

Postcondition: User is notified about new Activity Reports.

Scenario: User with Activity Report notifications enabled receives notification

1. System sends a push notification to the user's phone when a new Activity Report is posted.
2. System sends a notification to the user's desktop app.
3. If the user has email notifications enabled, the system sends an email notification as well.

Scenario: User with Activity Report notifications disabled

1. If a user has Activity Report notifications disabled, they do not receive any notifications about new Activity Reports.

Scenario: User updates their Activity Report notification preferences

1. User navigates to the settings or preferences section of the app.
2. User enables or disables Activity Report notifications (push, email, desktop app) based on their preferences.
3. System saves the updated preferences for the user.

B.7.1.5 - View Activity Reports

User Task: View Activity Reports

Actors: User, System

Description: User views a feed of neighborhood Activity Reports.

Precondition: User is logged in and navigates to the "Activity Reports" page.

Postcondition: User can view Activity Reports in chronological order and filter them as needed.

Scenario: User views all Activity Reports

1. User navigates to the "Activity Reports" page in the app.
2. System displays a feed of neighborhood Activity Reports sorted in chronological order.
3. User (optionally) selects the filter criteria from the available options.
4. System applies the filter to the Activity Reports list and displays only the matching items.

Alternate Flow: No matching Activity Reports found for the applied filter(s)

1. User selects filter criteria from the available options.
2. System applies the filter to the Activity Reports list, but no matching items are found.
3. System displays a message to inform the user that no matching Activity Reports were found and suggests to adjust the filter criteria or view all Activity Reports.

B.7.2 - Misuse Cases

B.7.2.1 - Spamming Activity Reports

Task: Create multiple fake Activity Reports

Actors: Malicious User

Description: A malicious user creates numerous fake Activity Reports to flood the system, causing confusion and decreasing the usefulness of the platform.

Precondition: Malicious user has access to the Activity Report creation interface.

Postcondition: System is flooded with fake Activity Reports, affecting the user experience and the platform's credibility.

Scenario: Malicious user creates multiple fake Activity Reports

1. Malicious user accesses the Activity Report creation interface.
2. Malicious user creates multiple fake Activity Reports with false or misleading information.
3. System processes and stores the fake Activity Reports, leading to a flood of fake information.

B.7.2.2 - False Emergency Notifications

Task: Create false Activity Reports with emergency notifications

Actors: Malicious User

Description: A malicious user submits false Activity Reports with emergency notifications enabled, causing unnecessary involvement of emergency services.

Precondition: Malicious user has access to the Activity Report creation interface.

Postcondition: Emergency services are involved unnecessarily, wasting resources and potentially delaying response to real emergencies.

Scenario: Malicious user submits a false Activity Report with emergency notification

1. Malicious user accesses the Activity Report creation interface.
2. Malicious user creates a false Activity Report and enables the emergency notification option.
3. System processes and stores the false Activity Report, and notifies the admin for review.
4. If admin approves the notification, the system sends a notification to emergency services, resulting in unnecessary involvement.

B.7.2.3 - Personal Attacks and Harassment

Task: Create Activity Reports targeting specific individuals or groups

Actors: Malicious User

Description: A malicious user creates Activity Reports with malicious content targeting specific individuals, neighborhoods, or groups, leading to harassment, discrimination, or other negative consequences.

Precondition: Malicious user has access to the Activity Report creation interface.

Postcondition: Targeted individuals or groups experience harassment or negative consequences.

Scenario: Malicious user creates an Activity Report targeting an individual or a group

1. Malicious user accesses the Activity Report creation interface.
2. Malicious user creates an Activity Report with harmful content targeting a specific individual or group.
3. System processes and stores the harmful Activity Report, potentially causing harm to the targeted individual or group.

B.7.2.4 - Misinformation and Rumors

Task: Create Activity Reports containing false information or unverified rumors

Actors: Malicious User

Description: A malicious user creates Activity Reports containing false information or unverified rumors, leading to confusion, panic, or an unnecessary reaction from other users or authorities.

Precondition: Malicious user has access to the Activity Report creation interface.

Postcondition: False information or rumors spread through the system, causing confusion or panic.

Scenario: Malicious user creates an Activity Report with false information or unverified rumors

1. Malicious user accesses the Activity Report creation interface.
2. Malicious user creates an Activity Report containing false information or unverified rumors.

3. System processes and stores the Activity Report with false information, potentially causing confusion, panic, or unnecessary reactions from other users or authorities.

B.7.2.5 - Unauthorized Modification or Deletion of Activity Reports

Task: Modify or delete Activity Reports without proper authorization

Actors: Malicious User, Hacker

Description: A malicious user or hacker gains unauthorized access to the Activity Report editing or deletion interface and modifies or deletes Activity Reports, causing loss of data or misinformation.

Precondition: Malicious user or hacker has gained unauthorized access to the Activity Report editing or deletion interface.

Postcondition: Activity Reports are modified or deleted without proper authorization, resulting in loss of data or misinformation.

Scenario: Malicious user or hacker modifies or deletes an Activity Report without proper authorization

1. Malicious user or hacker gains unauthorized access to the Activity Report editing or deletion interface.
2. Malicious user or hacker modifies or deletes an Activity Report without proper authorization.
3. System processes the unauthorized modification or deletion, resulting in loss of data or misinformation.

B.8 - Pet Registration

This section details the use cases associated with registering user pets.

B.8.1 - Scenarios

B.8.1.1 Register Pet

User Task: Register a pet to user their account

Actors: User, System

Description: User registers their pet to their profile.

Precondition: User has a verified account and is at the pet registration page.

Postcondition: User has registered their pet on their account.

Scenario: User registers their pet on the app

1. User navigates to their personal information settings page
2. User clicks "Register Pet".
3. A form pops up to add their pet information including: animal, breed, age, color, name.
4. User can also attach files including vaccination records and an image of the pet.
5. User clicks "Submit Pet Information" button and the pet is added to their user information page.

Alternative Flow: Missing Information

1. user does not include all necessary pet information so the pet is not added, the form fields that were not filled out gain a red border with a message saying "required field".

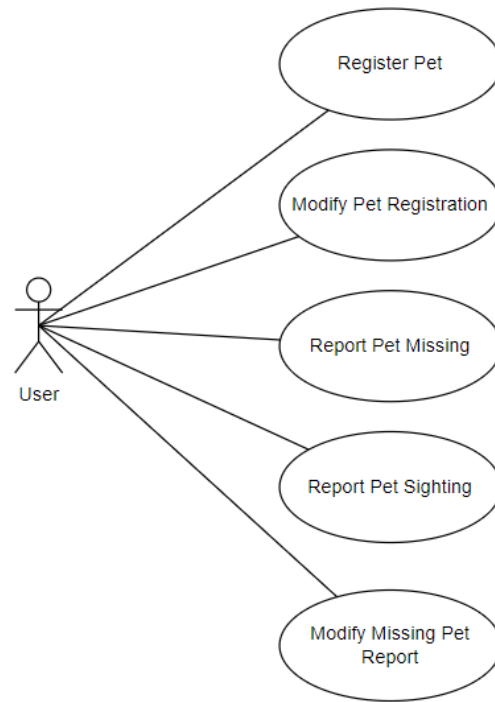


Figure 26: Pet Registration Use Case Diagram

B.8.1.2 Modifying Pet Registration**User Task:** Modify pet registration**Actors:** User, System**Description:** User modifies the information of their registered pet on the application.**Precondition:** User has a verified account and has a pet registered.**Postcondition:** The pet information is successfully updated.**Scenario 1:** User updates registered pet's information

1. User navigates to the pet section of the personal information settings page.
2. A form with all the pet information appears where the user can edit the pet's information.
3. User adds new information or edits existing information.
4. Once the user has updated the pet information, the user clicks the "update" button and the information is saved to the user's account.

Scenario 2: User removes pet from account

1. After reaching the "Edit" page, the user clicks a button that says "Remove Pet".
2. A confirmation box appears asking if they wish to remove the pet from their account with two buttons saying "Confirm" and "Cancel" respectively.
3. The user presses confirm, they are redirected to their personal information page with the pet successfully removed from the account.

Alternative Flow: User cancels update

1. The user decides they don't want to update or delete their pet information and presses the cancel button in the update/delete page.
2. The user is redirected to their personal information page where no information has been changed.

B.8.1.3 Report Pet Missing

User Task: Report Missing Pet

Actors: User, System

Description: User reports their pet as missing via the application

Precondition: User has a verified account, has a registered pet, and is at the pets section of their personal account page.

Postcondition: A new forum post is created detailing the missing pet's information

Scenario: User successfully creates missing pet report

1. User clicks the "Report Missing" button while at the pet section of their personal account page.
2. A form pops up with fields to capture the pet's information, including animal, breed, age, color, name, any distinguishing characteristics, last seen location, and a field for any additional information the user wants to add. The form also includes an option to upload a recent picture of the pet.
3. User can optionally opt to submit as an Activity Report to alert neighbors of missing pet.
4. After filling out the information, the user clicks the "submit" button and is prompted to confirm their submission.
5. If the user confirms, a forum post with a status of 'Open' is created in the "pets" forum section, detailing the missing pet's information.
6. Users who have opted-in to receive notifications about lost pets are notified via email and/or an in-app notification about the new post.

Alternate Flow: User submits Activity Report with missing pet post

1. User follows process to create a missing pet report and opts to file an Activity Report with the post.
2. User submits the post and a missing pet forum post is created.
3. In addition to notifications sent to opted-in users, an alert is sent to neighbors on the app, alerting them of the missing pet.

B.8.1.4 Report Pet Sighting

User Task: Report a sighting of a missing pet

Actors: User, Pet Owner, System

Description: A user reports they have sighted the missing pet via the application.

Precondition: There is a missing pet post, user has a verified account.

Postcondition: A reply to the missing pet post is made and the pet owner is notified about the sighting.

Scenario: User replies to a missing pet forum post

1. User navigates to the post for the missing pet where there is a button that says "reply".
2. User clicks "reply" button and the forum reply screen appears.
3. User enters information about the sighting and presses "send" button.

4. The pet owner is notified via email, in-app notification, or both, about the sighting and the details provided by the user.

Alternate Flow: User cancels sighting report

1. User is unsure if they saw the pet, decides to cancel their reply by pressing the button labeled "cancel".
2. User's comment is discarded and they are navigated back to the missing pet post.

B.8.1.5 Modify Missing Pet Report

User Task: Modify Missing Pet Report

Actors: User, System

Description: A user who created a missing pet report modifies their post, possibly closing it, updating, adding, or removing post content.

Precondition: User has a verified account and has made a Missing Pet post.

Postcondition: The forum post is updated with new information, and relevant users are notified (if applicable).

Scenario 1: User updates missing pet information on their post.

1. User navigates to their missing pet post and clicks the "Update Post" button.
2. The user is redirected to the pet information form where it contains the pet's identifying information, last seen location, a recent photo of the pet, and any other relevant details provided in the original report.
3. The user updates the post with new information or changes to existing information and presses the "submit" button.
4. The post is now updated with the new information, and relevant users are notified about the update.

Scenario 2: User closes the post since pet has been found or is no longer missing

1. User navigates to the Missing Pet Post and clicks "Close Post" button.
2. A form pops up prompting the user to include an update message to the post (e.g. "Pet has been found!").
3. The user enters a message (if desired) and clicks the "submit" button.
4. The post is updated with the new information (e.g. "Pet has been found!") and moved to an archived forum section, or deleted. Relevant users are notified about the closure of the post.

Appendix C - Class Diagrams

The following is a depiction of all class types as well as associated relationships providing a high-level overview of the classes. Dotted lines with solid arrows indicate an implementation of an interface. Solid lines with solid arrows indicate the class extends (inherits) from the class in which the arrow is pointing.

C.1 - Homeowner Association Account

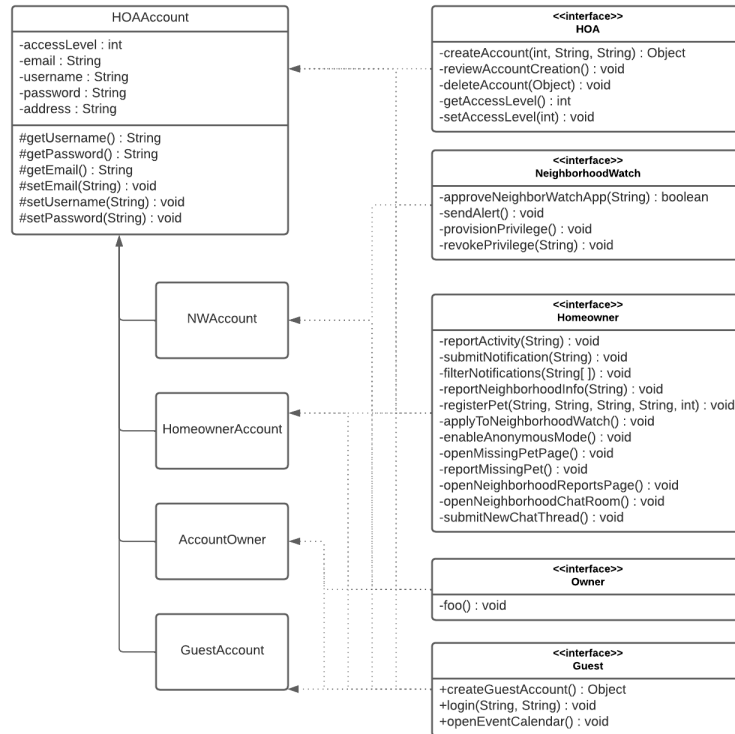


Figure 27: Overall Class Diagram

C.2 - Neighborhood Watch Account

The following is a depiction of the Neighborhood Watch class type in UML format. The NW account holder is a semi-privileged user account having all but admin privileges (such as the HOA account holder). The NW class implements the NeighborhoodWatch, Homeowner, Owner, and Guest interfaces.

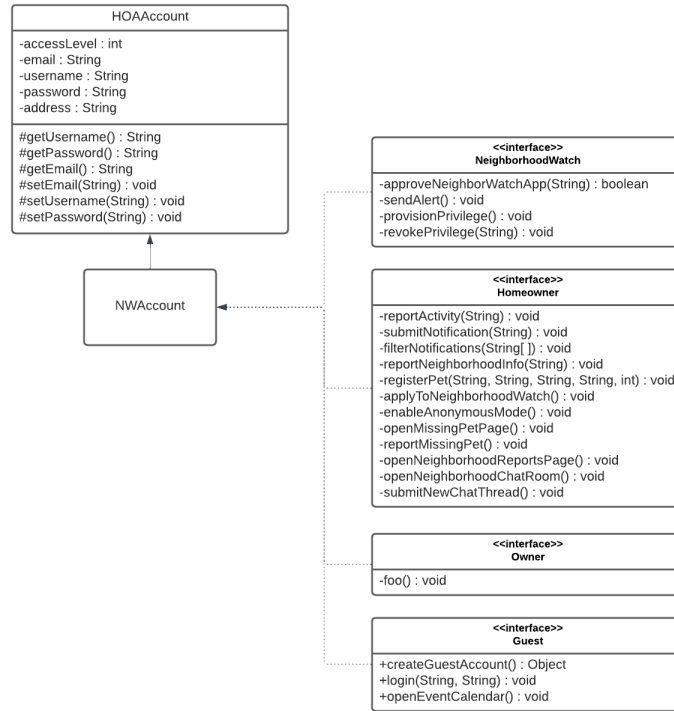


Figure 28: Neighborhood Watch Account

C.3 - Homeowner Account

The following is a depiction of the Homeowner class type in UML format. The Homeowner account holder is a general user account having access to the entire user domain of the application. The Homeowner class implements the Homeowner, Owner, and Guest interfaces.

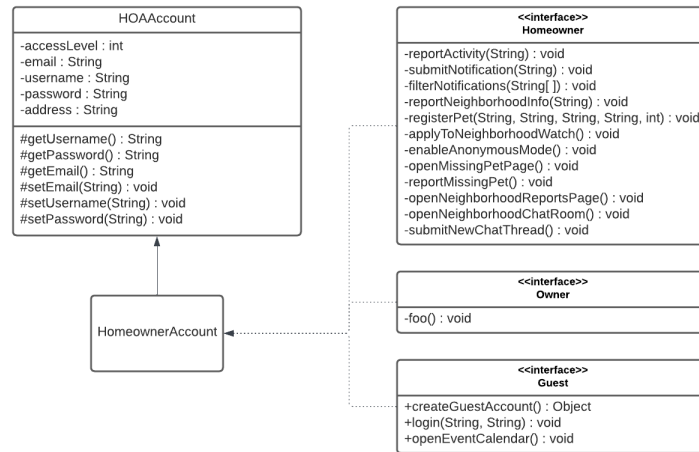


Figure 29: Homeowner Account

C.4 - Owner Account

The following is a depiction of the Account Owner class type in UML format. The Account Owner account holder is a general user account having limited access to the entire user domain of the application. The Account Owner class implements the Owner, and Guest interfaces.

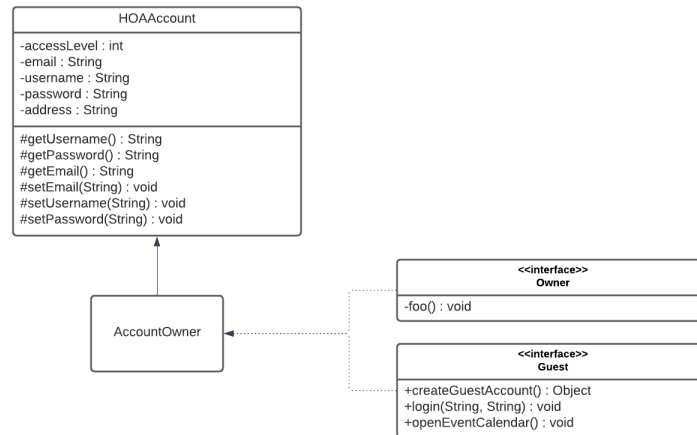


Figure 30: Account Owner Account

C.5 - Guest Account

The following is a depiction of the Guest class type in UML format. The Guest account holder is a limited user account having access to only a few limited functionalities allowing the creation and login of an account, and the ability to view the neighborhood events calendar. The Guest class implements the Guest interface.

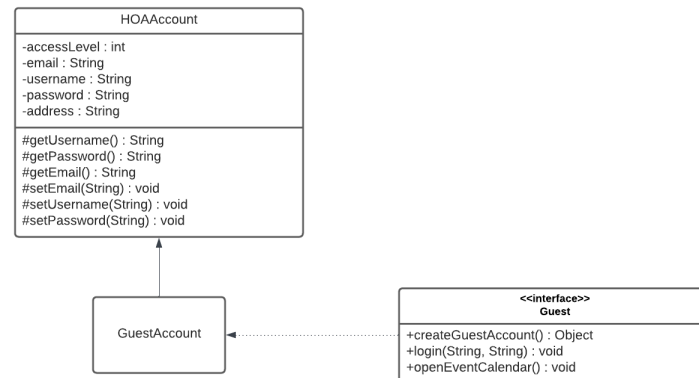


Figure 31: Guest Account