Gather
Design
Version 1.0

Ryan Hill, Calvin Keats, Noah Williams

Title Page 1

Table of Contents 2

Change History 3

Introduction 4

Project Description 5

Requirements 6

- Functional 6
- Nonfunctional 7

UML Diagrams 8

- Use Case Diagrams 8
- Class Diagram 11
- Activity Diagrams 13
- Sequence Diagrams 19

Change History

Version 0.1 -- Initial Documentation
Version 1.0 -- Initial Presented Version

Introduction

Motivation/Purpose

Gather is being designed to break the isolating mould of current social networking by actually enabling users to socialize with new people. In doing so we hope that, with appropriate usage, Gather will be able to help people acclimate to a new place or even branch-out socially by providing them with a tool to find and interact with like-minded people

Scope

Gather will utilize several different features of the android system. To start, we will use the Google Maps API in order to show events based on location data. Event data, such as the description, location, time, etc. will be held in a database. Optionally, users will be able to create user profiles which will also be held on our database.

Goals

Our current goals for Gather are the following:

- Create a sleek, usable application that allows users to intuitively interact within our system
- Provide users with a graphical representation of events using the Google Maps API
- Provide users who opt-in to have a profile in Gather with extended functionality and customisability
    - Eventually provide Facebook/Google+ integration

Key definitions

- **Gather**: Our application for which this documentation is written
- **MTBF**: Mean time between failures
- **User**:
    - *When in Use-Case*: A user of our application who has created a personal profile
    - *General*: A user of our application
- **AnonUser**: A user of our application who has not created a personal profile

Project Description

For Gather to be successful, we will design and implement several different key features. These features include the essential functionality:

- Creating Events
- Managing Events
- Viewing Events

Additional functionality that does not need to be included in an initial release of Gather would be as follows:

- Ability to create and manage a user profile
- Ability to integrate profile with existing social networks
- Integration into a website so mobile-viewing is not mandatory

In addition to the front-end functionality listed above, there are several features on the backend that must be considered:

- Database to store event and user information
- Web server to handle website functionality

Functional Requirements

- Allow user to create events
- Allow user to modify event details
    - Title
    - Category
    - Location
    - Time
    - Description
    - Target number of people
    - Age requirements
    - Picture
- Allow user to browse events via menu
    - General browse
    - Category browse
    - Time browse
- Allow user to browse events via map
- Allow user to search for events via search bar
- Allow user to see event details
- Allow users to interact with events
    - Comment
    - Claim to be attending
    - Opt-in to be notified of event comments
- Notify users when an event they claim to be attending has changes in its details
- Notify event creator when a user claims that they will attend the creator's event
- Notify event creator when a user comments on their event
- Allow user to see events they have created or claim to be attending

Nonfunctional Requirements

- Service Availability/Reliability
    - MTBF
- Security
    - Secure transfer of login credentials
- Usability
    - Clean Interface
    - Non Resource-Intensive
- Scalability
    - Keeping up with a growing user base

Use Case Diagrams

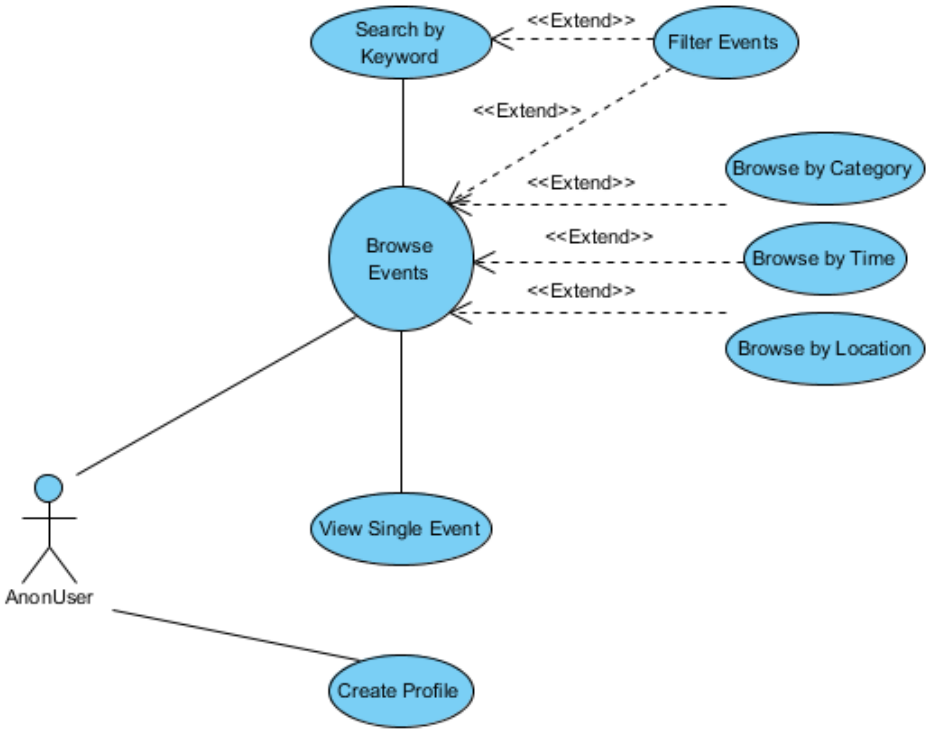
User Use Case Diagram

The user use-case diagram has one actor, the “User”. The User can browse events and extend browsing functionality by filtering the events they see by category, time, or location. Alternatively, they can use the search functionality to search for keywords in the event’s description and title.

While browsing events, the user can view a single event to get more details. From the single event view page, the user can toggle notification for the event, claim that they will attend that event, or post comments on that event’s page. The user can also manage any events they have created.

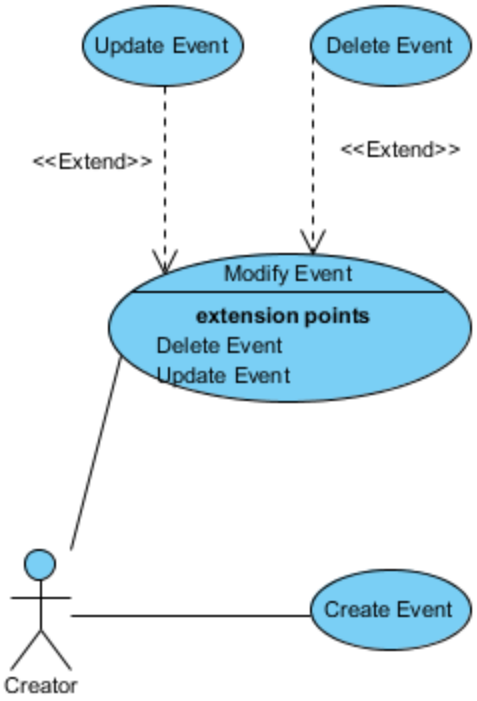
The user can log in to a profile (granted that they have created one) and their profile information as well.

AnonUser Use Case Diagram



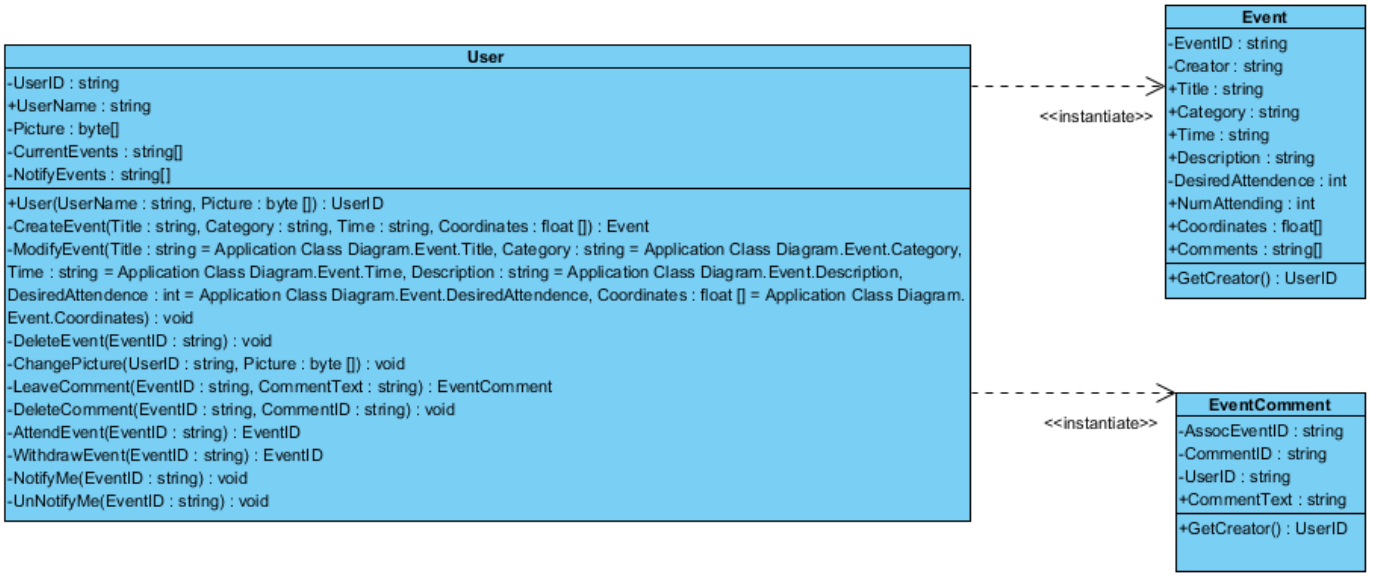
The AnonUser (short for Anonymous User) use case has one actor, that of the AnonUser. The AnonUser has all of the same functionality as the “User” diagram above, with the exception of the ability to login and manage a profile. The AnonUser also has one action that the “User” lacks, that being the ability to create a profile.

Creator Use Case Diagram



The “Creator” diagram has one actor, the “Creator”. The Creator is a User or an Anonymous user that chooses to create a new event. This is the first action of the “Creator” actor. In addition, the creator can modify any event they have created. The modifications that a creator can make to their events are to update the event information or delete the event completely.

Class Diagram



The class diagram to Gather is rather simple. The main class for our application, the User, is the most complicated. The User has a UserID, which will be a global unique identifier (GUID). In addition, the user will have a collection of EventIDs which reference the events they have claimed they will attend. Another collection of EventIDs will represent which events the User wants to receive notifications from. If a user chooses to create a profile, the Picture and UserName attributes become available, which respectively are a byte array representing an image and a unique (via database query) string. The User class also has several operations. The User has a simple User() initialization operation, as well as functions to Create, Modify, and Delete events. “Create Event” only requires that the user specify a title, category, time, and place. “Modify Event” has only optional parameters, and the parameters that are not entered will simply default to whatever they were already set to for the event. A user who has an account has the ability to use the ChangePicture() operation as well, which changes the profile picture they use to represent themselves. Registered users can leave a comment on an event using LeaveComment(), and can also delete comments that they have made using DeleteComment(). Additional functionality for all users includes the ability to claim they are attending an event using AttendEvent(), and the ability to withdraw that attendance using WithdrawEvent(). NotifyMe() and UnNotifyMe() allow a user to indicate whether or not they wish to receive notifications for an event, or halt notifications from that event, respectively.

The Event class can be instantiated by a User through the CreateEvent() method. An Event has a GUID formatted to be a string as an EventID and a string representing the UserID of the creator of said event with is the Creator attribute. In addition to these attributes, the Event has several attributes used to hold information about itself, those being the Title, Category, Time, Description, DesiredAttendence, and NumAttening all of which are self-explanatory. The Coordinates attribute is an array of float values representing the

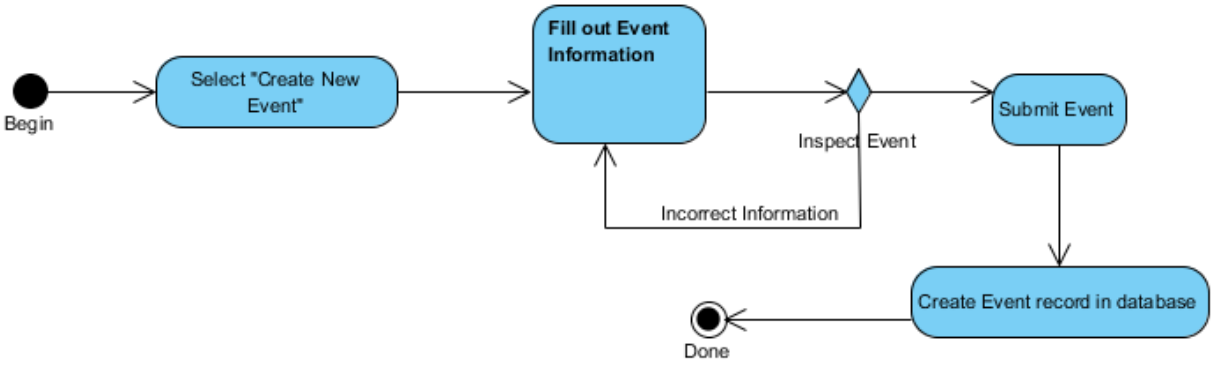
longitudinal and latitudinal coordinates of the event's location. The Event class also has an array of strings which would be CommentIDs representing the list of comments for that event.

The EventComment class has only four attributes, those being the AssocEventID which is the EventID of the Event that the EventComment is linked to, the CommentID which is the indentifier of the comment itself, the UserID which is the GUID of the user who created the comment, and the CommentText which is the actual string content of the comment.

Both the EventComment and the Event classes have a single operation, GetCreator(), which returns the UserID of the User who created the EventComment or the Event.

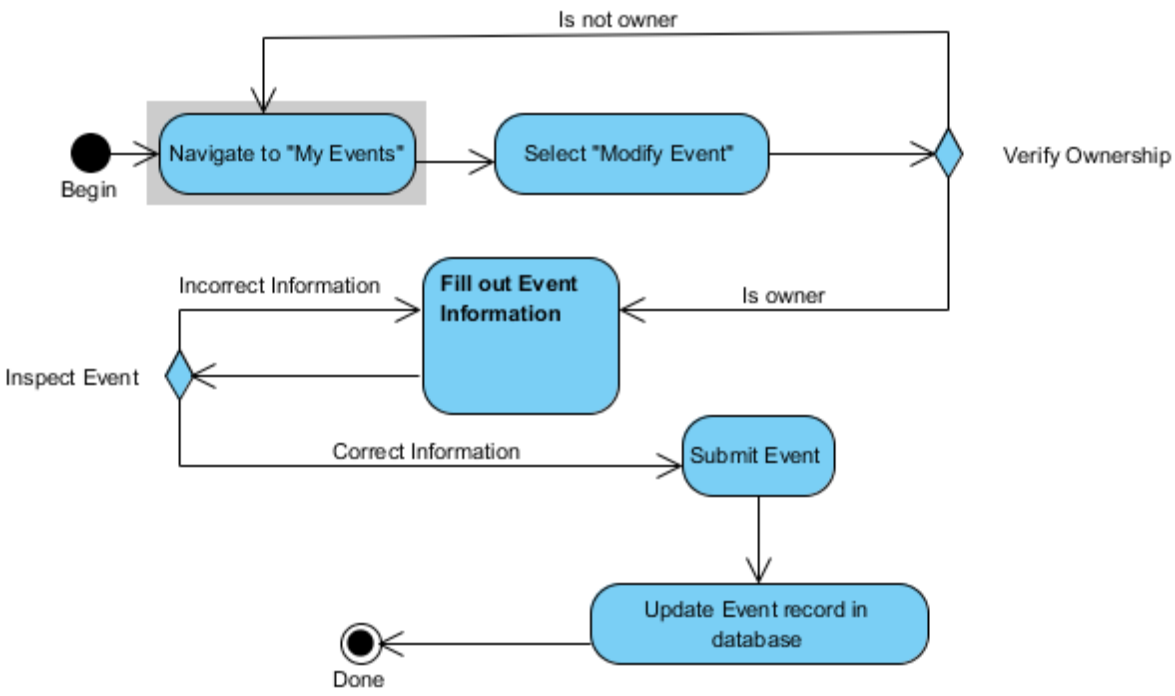
Activity Diagrams

Create Event



The Create Event activity diagram begins when the user selects “Create New Event” option from our application’s main page. They then fill out a form of all the information required to make the event, and then they inspect that form to verify that it is correct. If the user deems the information incorrect, they are returned to the form to correct their information. If the user deems the information correct they submit the event, then the event is created within our database and the action is complete.

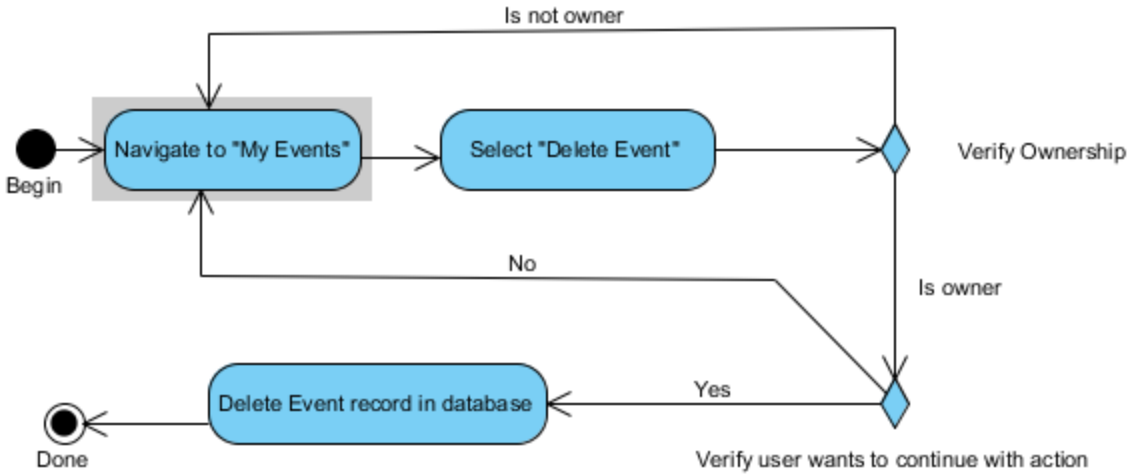
Edit Event



The “Edit Event” activity begins when the user navigates to the “My Events” tab of our application. They will then select the “Modify Event” option to an event they have created, which checks whether or not they are the one who created the event. Ideally, this will always return that they are the creator because they should not have the option to choose “Modify Event” on an event they did not create.

Once ownership is verified, the user fills is taken to the form of information for the event and can change it as they see fit. The user then they inspect that form to verify that it is correct. If the user deems the information incorrect, they are returned to the form to correct their information. If the user deems the information correct they submit the event, then the event is updated in our database and the action is complete.

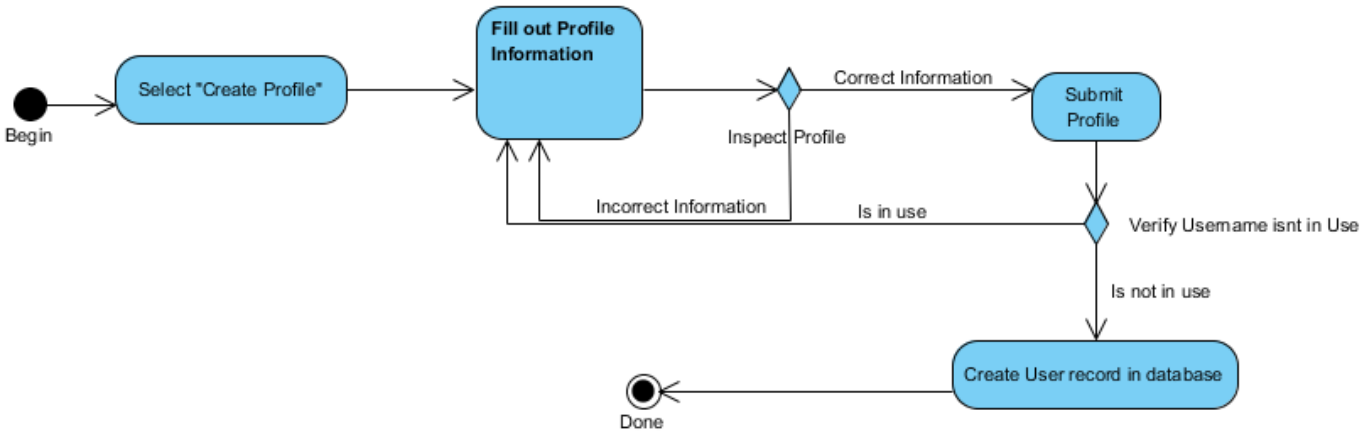
Delete Event



The “Delete Event” activity begins when the user navigates to “My Events”. They will then select the “Delete Event” option to an event they have created, which checks whether or not they are the one who created the event. Ideally, this will always return that they are the creator because they should not have the option to choose “Delete Event” on an event they did not create.

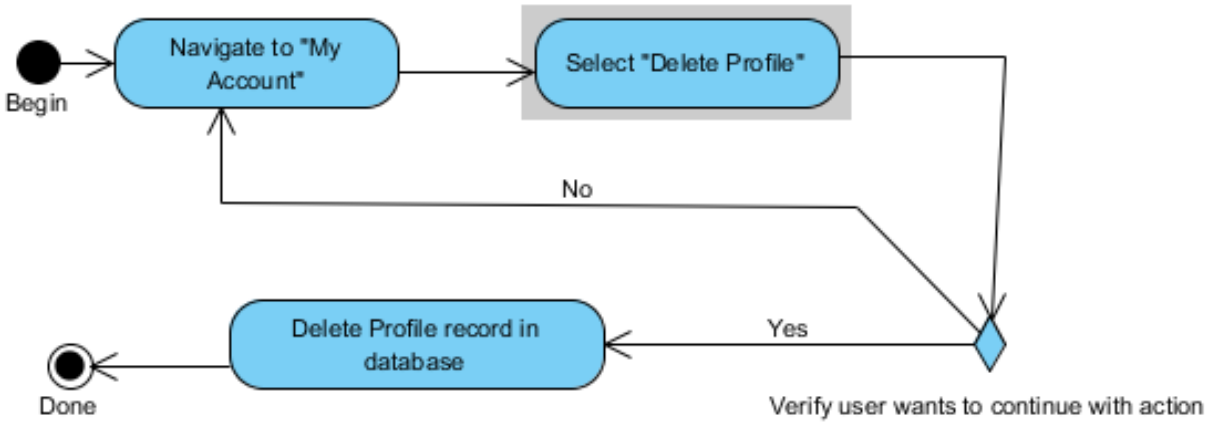
Once ownership is verified, the user is prompted to verify their action. If the user verifies their intent to delete the event, the event is deleted in our database and the action is complete.

Make Profile



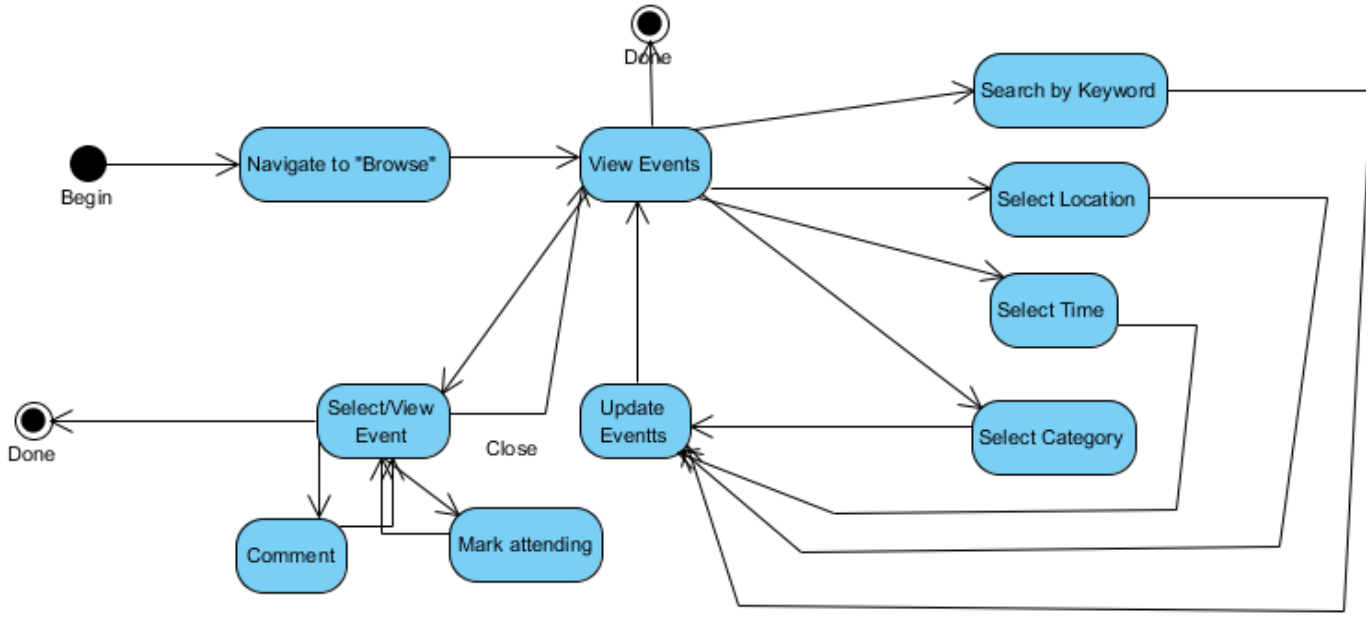
The “Make Profile” activity begins when an Anonymous User selects the “Create Profile” option. They must fill out a form of data relevant to profiile creating, and then inspect that form. If the user deems that information incorrect, they are taken back to the form. If the user deems the information correct, they submit the profile. This then verifies that the username they input to the form is not currently in use by another user. If the username is currently being used, the user is taken back to the form and asked to choose a different username. If the username is not in use, the User record is created in the database and the action is complete.

Delete Profile



The “Delete Profile” action occurs when a User with a profile chooses navigates to the “My Account” page and selects the “Delete Profile” option. This simply verifies that the user does want to delete the profile. If the user chooses not to confirm this action, they are returned to the “My Account” page; otherwise the Profile record is deleted in the database and the action is complete.

Browse Events

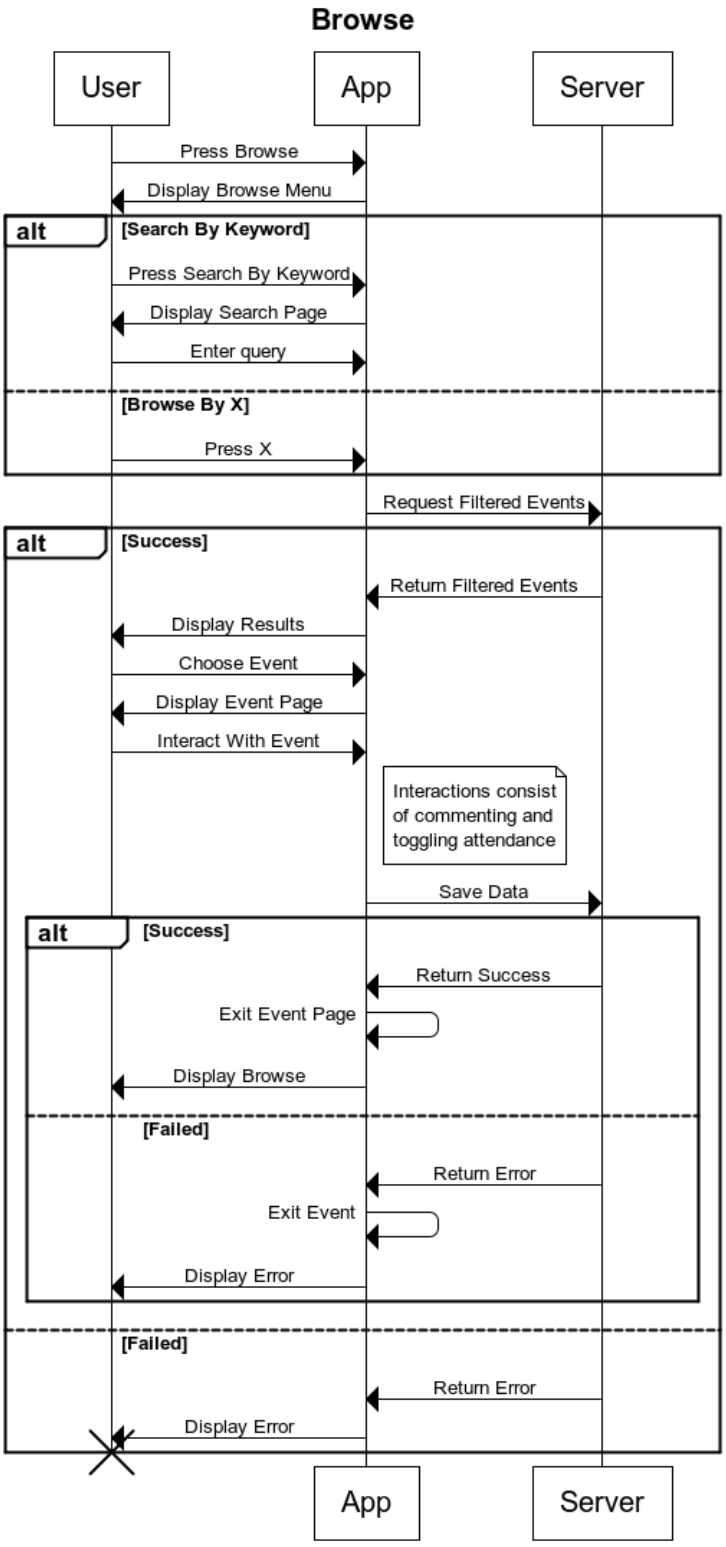


The “Browse Events” activity begins when the user navigates to the “Browse” tab of our application. From this view, they are shown a list of all events within a certain proximity of themselves. They can either choose to exit the “Browse” section from here thus terminating the activity, select a single event to view more details, or update their results. To update their results, the user can search events by keyword, select a new location, set a timeframe, or select a category. All of these options can be used at once or in any combination. Once the user has opted to filter their results by any of these methods, the events listed are updated via a query to our database and the new listing will be shown.

If a user selects a single event to view its details, they can comment on the event or mark themselves as attending. Once they have done either of those things they are returned to the single event viewing page. From this page they can either close the page thus returning to the previous list of events, or leave the “Browse” section thus terminating the activity.

Sequence Diagrams

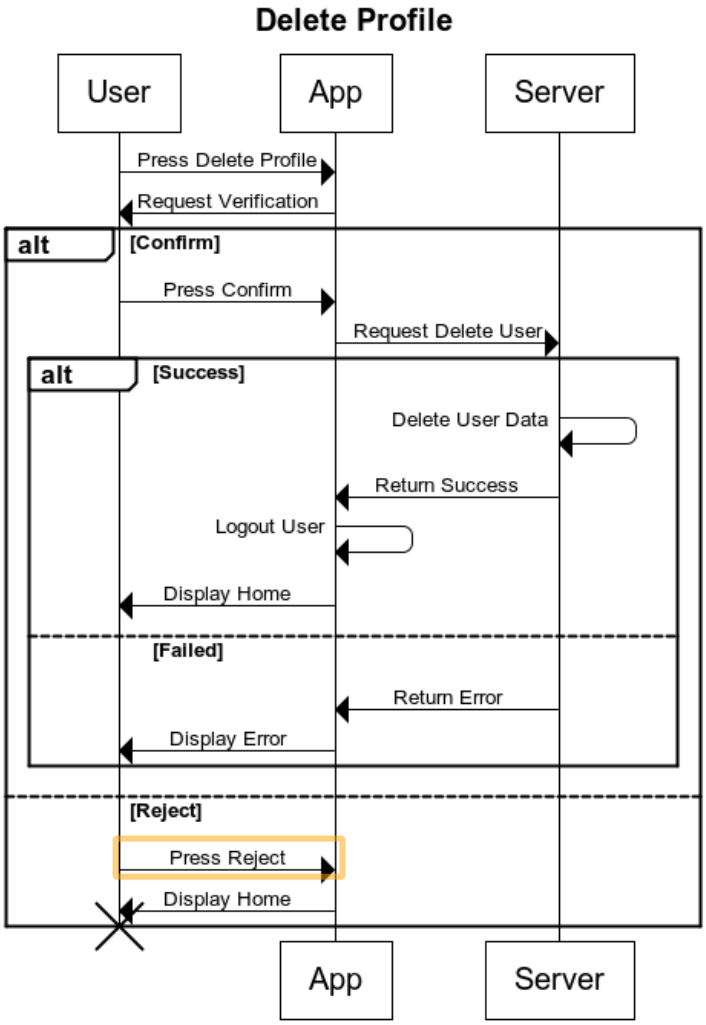
User Sequence Diagrams



When browsing events, a user would press the browse button, then choose a method of browsing.
If the user wishes to search a keyword, they will press that button, and when presented with a text field, enter the keyword they wish to search. The app will then form a SQL query and retrieve the relevant information from the database. Upon retrieval the app will then display it to the user.

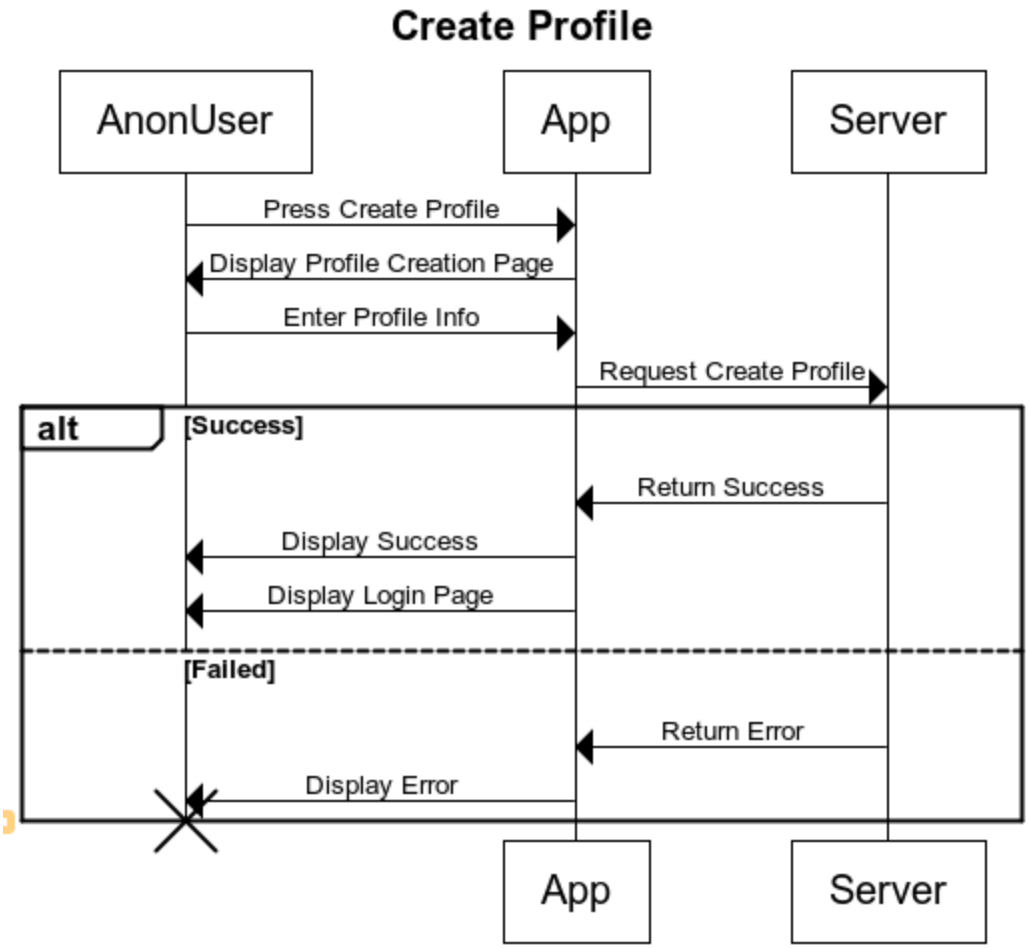
If the user wishes to browse by Category, Time, or Location, they will press the relevant button, then the app will form the proper SQL query and retrieve the data from the database. upon retrieval, the app will display the filtered list to the user.

If an error occurs with the database, the user will be notified.



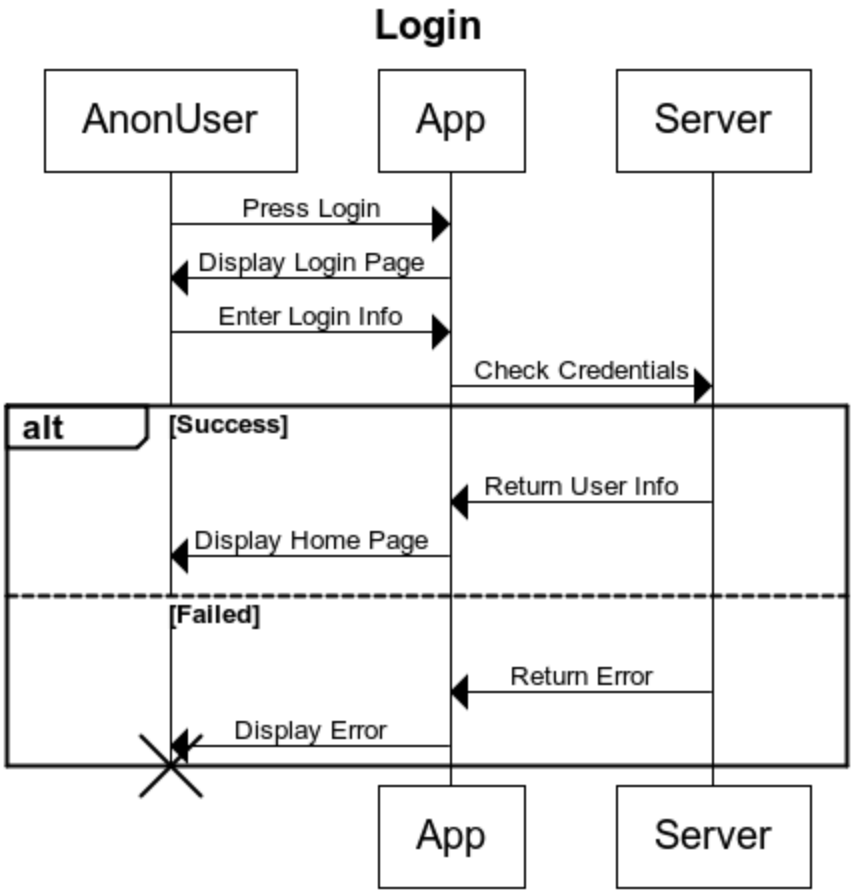
If the user wishes to delete their profile, they will navigate to the settings menu, and press “Delete Profile.” The app will then return a confirmation prompt and upon confirmation will send the proper delete statement(s) to the server. After the account has been deleted the user will be logged out and returned to the home screen.

AnonUser Sequence Diagrams



If an anonymous user wishes to create a profile, they will press the “Create Profile” button. The app will take the user to the profile creation page where they may enter the requesting information. Upon completion the app will send the insert (SQL) statements to the database. After a successful insertion the user will be notified and taken to the login screen.

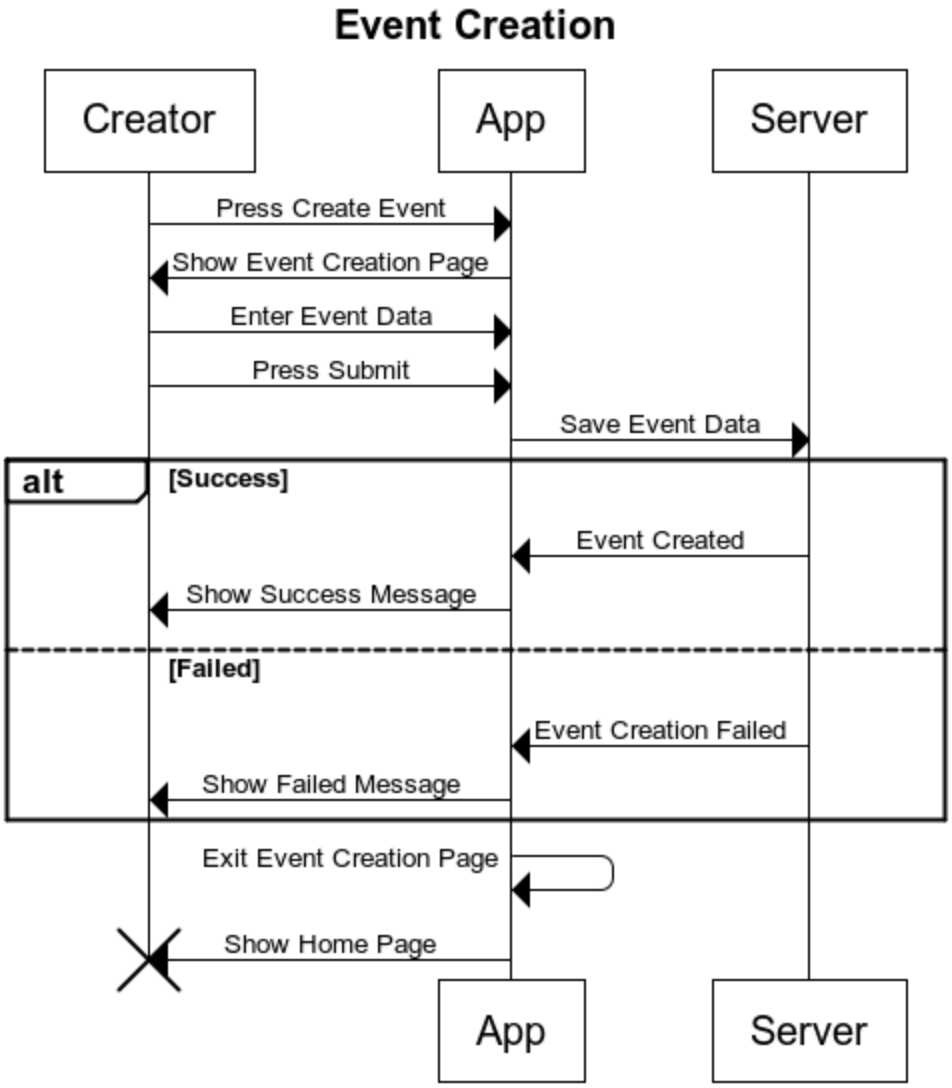
If an error occurs the user will be notified.



If an anonymous user wishes to log in with a pre-existing account, they will press the “Login” button and be taken to the login page. After entering username and password the app will send the credentials to the server for comparison. If a match is found the server will return the appropriate profile information, the user will then be logged in, and taken to the home page.

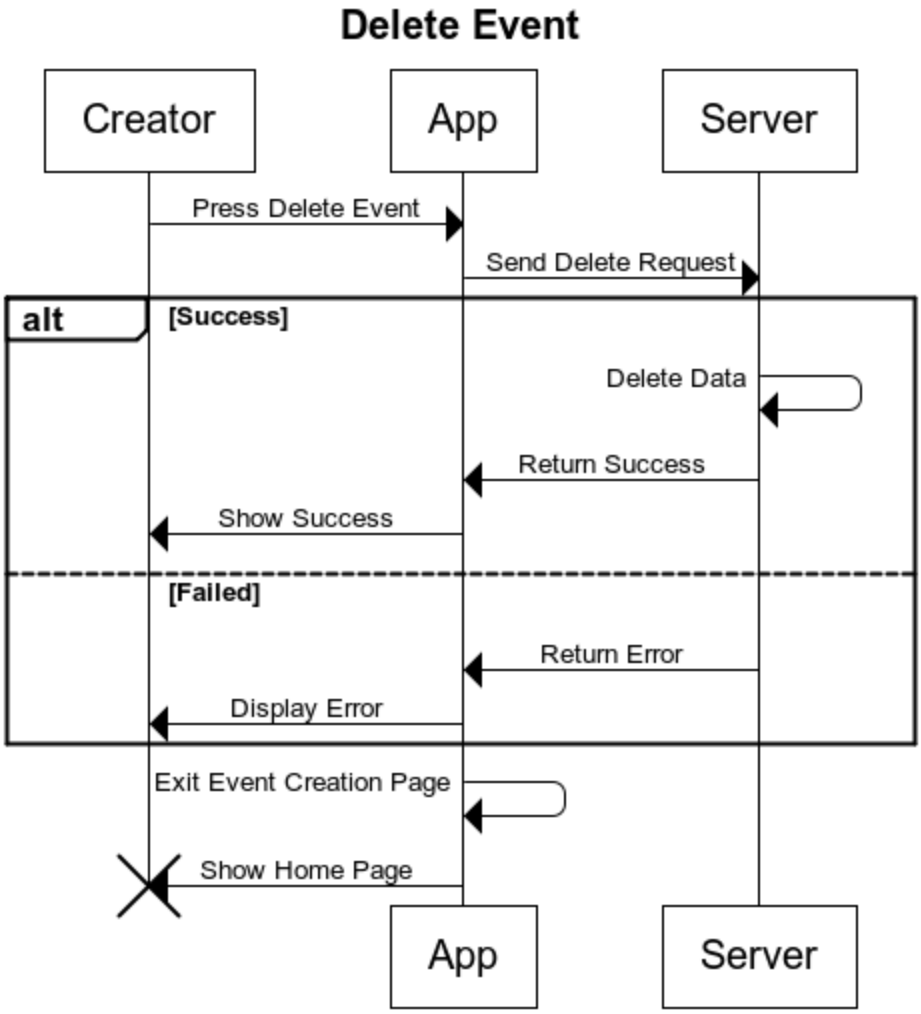
If an error occurs the user will be notified.

Creator Sequence Diagrams



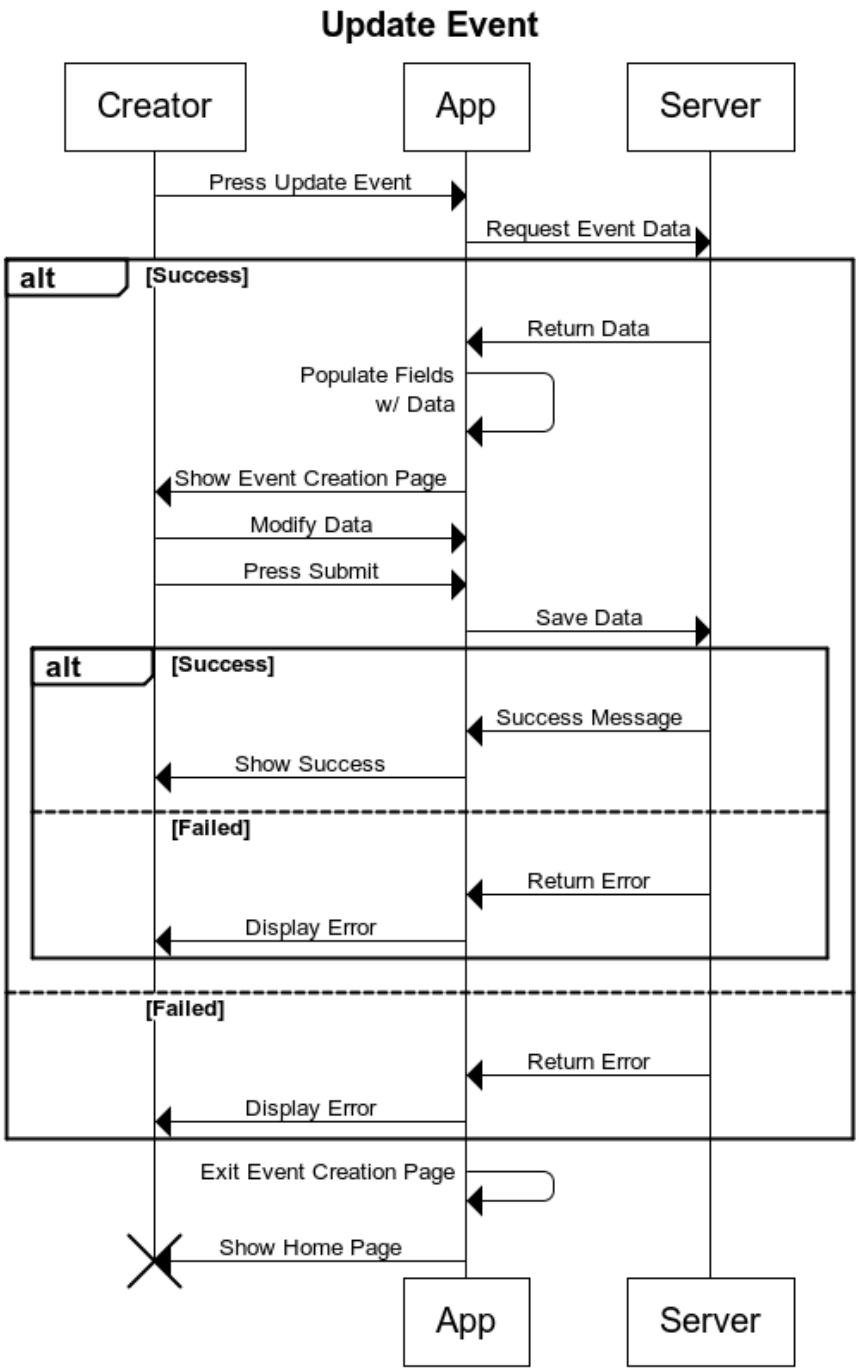
If a user wishes to create an event, they will press “Create Event” whereupon they will be taken to the event creation page. Here they will enter the requested event data and press submit. The app will then send the data to the server. If the event is successfully created the user will be notified and taken to the home page.

If an error occurs the user will be notified.



If a user wishes to delete an event they have created they will navigate to the event, and press “Delete Event” after the user confirms this, the delete request will be sent to the server. If successful the User will be notified and taken to the home page.

If an error occurs the user will be notified.



If a user wishes to modify an event they have created they will navigate to the event page and press “Update Event” The User will then be taken to the event creation page (which will pre-populate all known fields. After the user has modified the data they wish to and submitted, the app will update the database. If successful the user will be notified and returned to the home screen.

If an error occurs the user will be notified.