Academic Collaboration Application

Front End & Back End

FE: Robert Carver, Marleigh Cattaruzza, Austin Carter

BE: Nathaniel Been, Courtney Casperson,Tyler Marler

Version: 0.3

December 9, 2018

Project Description

Overview:

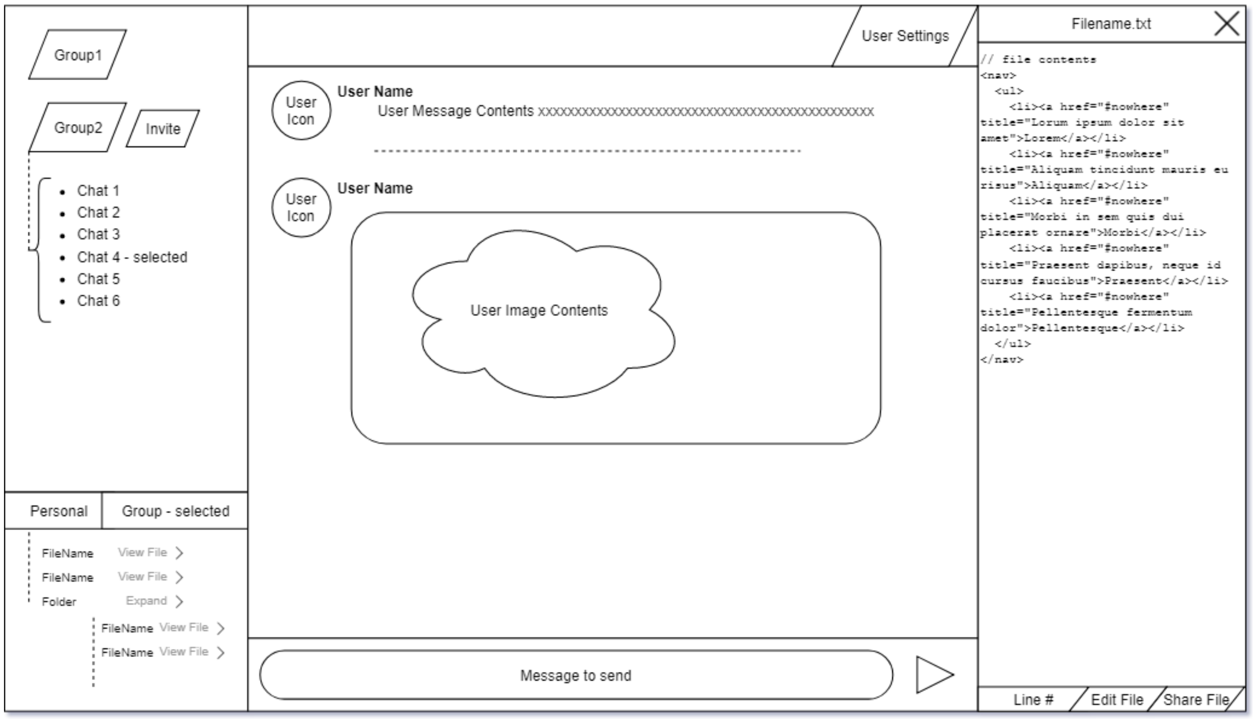
The application is geared toward student collaboration within a team environment with a social media spin. The application will allow teams to communicate and share documents over a cloud platform, with their documents and communications saved for future reference. Teams will be able to edit documents using the same application they communicate with.

Glossary:

- **Login**: the process of entering a username and password and validating this data.
- **User**: a human user of the application.
- **Author**: the creator and owner of a file.
- **Action**: a task that the client-application wants to accomplish.
- **Shared Files**: files the user has access to within the client-application, that they are not the author of.
- **Personal Files**: files the user has access to that they are the author of.
- **Chat-room/Chat**: a real-time display of messages sent by the users within the chat-room.
- **User Information**: email or username are interchangeable fields
- **Group**: a created collection of chat-rooms that can contain multiple users.
- **Client-application**: the desktop application a user will interact with.
- **Web-server**: a website that the client-application can send message requests to.
- **Validate**: the process of the web-server checking sent information from the client-application.
- **Logic**: Code operations
- **Login**: the process of a user entering their information into the client-application and gaining access to their account.
- **Message Object**:
    - Has a field that indicates the type of message.
    - Has a field that contains context specific parameters.
- **Message Request**: An information request made by the client sent to the web-server.
- **Message Response**: An information response made by the web-server and sent to the client.
- **User Token (token)**: An alphanumeric string used to validate user information.
- **Chat Messages**: An object with the attributes: content (string), user ID (int), username (string)
- **JSON**: JavaScript Object Notation. See Reference 3.
- **Deserialize:** Taking data structure from a format and rebuilding it into a object.

User Requirements:

1. Users will interact with an interface as depicted above in Diagram 1.0
2. Users will be able to login to their account and have access to personal and shared files.
3. Users will be able to grant access to files to other users.
4. Users will be able to communicate via team chat-rooms.
5. Users will be able to create new Groups.
6. Users will be able to delete their created Groups.
7. Users will be able to create new chats within their groups.
8. Users will be able to delete chats within their groups.
9. Users will be able to invite other users to groups by entering identifying user information.
10. Users will be able to add other users to chat-rooms by entering identifying user information.
11. Users will be able to update their username and password within the application.
12. Users will be able to logout of the application.
13. Users will be able to edit both shared and personal files.
14. Users will be able to view files as they are edited.



```
// file contents
<nav>
  <ul>
    <li><a href="#nowhere"
title="Lorum ipsum dolor sit
amet">Lorem</a></li>
    <li><a href="#nowhere"
title="Aliquam tincidunt mauris eu
risus">Aliquam</a></li>
    <li><a href="#nowhere"
title="Morbi in sem quis dui
placerat ornare">Morbi</a></li>
    <li><a href="#nowhere"
title="Praesent dapibus, neque id
cursus faucibus">Praesent</a></li>
    <li><a href="#nowhere"
title="Pellentesque fermentum
dolor">Pellentesque</a></li>
  </ul>
</nav>
```

Diagram 1.0: client-application layout

<u>Description of Client-Application Layout (1.0):</u>

Group and chat selection is in the top left corner of the layout. Personal and group file directories are shown in the bottom left corner of the layout. Chat messages are displayed in the center of the layout. The user settings button is in the top center of the layout. The far right of the layout is the file editing pane. The bottom center of the layout is the message entry box.

System Overview:

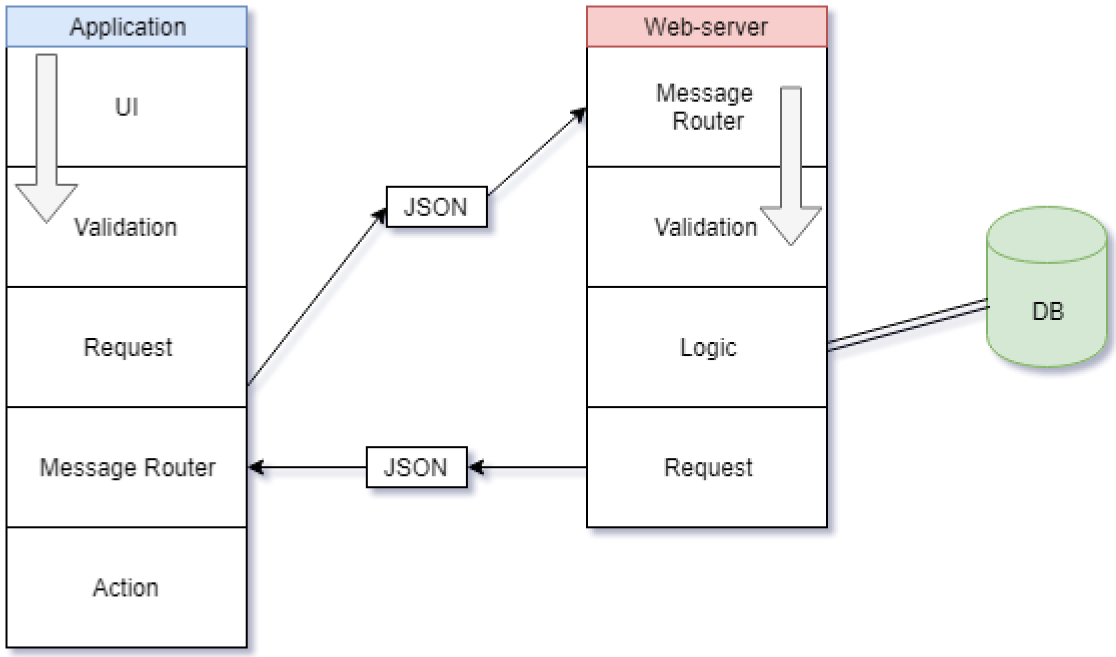


Diagram 1.1: system overview diagram

<u>Description of System Overview Diagram (1.1):</u>
The diagram above (1.1) describes how the system will function at a conceptual level. The client-application will be comprised of 5 distinct operation layers:

1. UI: the UI layer defines an action generated by the user interacting with the interface, the action is then passed into the next layer.
2. Validation: the validation layer defines the operation of validating the action generated by the user interacting with the interface. After validation this action is passed into the next layer.
3. Request: the request layer defines the operation of taking a validated action and packaging it into a message of the format specified below. This message is then sent as JSON to the top layer of the Web-server.
4. Message Router: the message router layer defines the operation of directing the received JSON message object to the correct logic after validation.
5. Action: the action layer describes the general operation of the client-application updating based on the received message.

The web-server/ "back-end" will be comprised of 4 distinct operation layers:

1. Message Router: the message router layer defines the operation of directing the received JSON message object to the correct logic after validation.
2. Validation: the validation layer defines the operation of validating the received message, after validation the message is directed to the next layer.
3. Logic: the logic layer describes the general operation of code operating on a message object to complete the required task. The code will receive the message and then

interact with the web-server or database to retrieve or accomplish the specified action. After completion the logic will send the action to the next layer.

4. Request: the request layer defines the operation of taking an action and packaging it as the correct message for that action as defined below (System requirements 9-10). The request layer will then send that message as a JSON object to the client-application.

System Requirements:

1. The client-application will send message requests to the web-server.
2. The web-server will only send messages to the client-application when prompted by the client-application.
3. All messages received by the web-server must be validated before being acted on by the web-server.
4. If a message is validated by the web-server then the logic will act on the message.
    a. The logic will access the database as necessary.
5. The web-server must send a message to the client-application once the message request is completed.
6. When a message is received by the client, the client must update the user interface.
7. Messages must abide by the following protocol (requirements 9-10) and process:
    a. A message object will be created as described in the definitions.
    b. Once it is created the message will be converted to JSON.
    c. The JSON message will then be sent as a HTTP request .
        i. If the client is sending the message the target will be the web-server.
        ii. If the web-server is sending the message the target will be the client that issued the request.
8. When the system receives a message:
    a. A message object will be deserialized from the received message request.
    b. The message will be routed to the appropriate logic based on the message type field.
9. Messages sent from the client-application must follow the protocol defined below:
    a. Login message
        i. Username - string
        ii. Password - string
    b. Sign-up
        i. Username - string
        ii. Password - string
        iii. Name - string
    c. Get user info
        i. ID - int
    d. Get all user files
        i. User token - token
    e. Get specific file
        i. User token - token
        ii. File ID - int
    f. Get file update

    i. User token - token
    ii. File ID - int

g. Edit file
    i. User token - token
    ii. File ID - int
    iii. Updated content - string
h. Delete file
    i. User token - token
    ii. File ID - int
i. Create file
    i. User token - token
    ii. File name - string
    iii. Group ID - int
j. Get all user groups
    i. User token - token
k. Get specific group
    i. User token - token
    ii. Group ID - int
l. Edit group
    i. User token - token
    ii. Group ID - int
    iii. New name - string
m. Delete group
    i. User token - token
    ii. Group ID - int
n. Create group
    i. User token - token
    ii. Group name - string
o. Join group
    i. User token - token
    ii. Group ID - int
p. Leave group
    i. User token - token
    ii. Group ID - int
q. Invite to group
    i. User token - token
    ii. Group ID - int
    iii. Username - string
r. Get chat
    i. User token - token
    ii. Group ID - int
    iii. Chat ID - int
s. Create chat
    i. User token - token

        ii. Group ID - int
        iii. Chat name - string
    t. Delete Chat
        i. User token - token
        ii. Group ID - int
        iii. Chat ID - int
    u. Message Chat
        i. User token - token
        ii. Group ID - int
        iii. Chat ID - int
        iv. Content - string
    v. Update Chat
        i. User token - token
        ii. Group ID - int
        iii. Chat ID - int

10. Message responses sent from the web-server must follow the protocol defined below:
    a. Generic confirmation message
        i. Request type - int
    b. Generic error message
        i. Error type - int
        ii. Error description - string
    c. User files
        i. Files - List<int, string>
    d. Specific file
        i. Content - string
        ii. Name - string
    e. User groups
        i. Groups - List<int, string>
    f. Get group
        i. Chats - List<int, string>
    g. Get chat
        i. Conversation - List<Chat messages>
    h. Login response
        i. User token - token
        ii. Username - string
11. A user login action requires the client-application to send a login request message
    a. If the login is valid the web-server will send a login response.
    b. If the login is not valid the web-server will send an error response
    c. Valid is defined: if the name and password match an entry within the database.
12. A user sign up action requires the client-application to send a sign-up request message
    a. If the sign-up is valid the web-server will send a login response.
    b. If the sign-up is not valid the web-server will send an error response.
    c. Valid is defined: if the username is not in the database and the password is 8 characters long.

13. When the client-application requests to get all files from a specific user, a get all user files request must be sent
    a. If the request is valid, the web-server will send a user files response
    b. If the request is not valid, the web-server will send an error response.
    c. Valid is defined: if the token sent in the request is valid and refers to an existing user in the database
14. When a user requests access to a specific file, the client-application will send a get specific file request
    a. If the request is valid, the web-server will send a specific file response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid and the file id refers to an existing file
15. When a user is viewing a file, the client-application will periodically send a get file update request
    a. If the request is valid, the web-server will send a get specific file response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid and the file id refers to an existing file in the database.
16. When a user edits a file, the client-application will send a edit file request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid and the file id refers to an existing file in the database.
17. When a user deletes a file, the client-application will send a delete file request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid and the file id refers to an existing file in the database.
18. When a user creates a file, the client-application will send a create file request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid
19. When the client-application requests to get all the groups a user is a member of, it will send a get all user groups request
    a. If the request is valid, the web-server will send a user groups response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid and refers to an existing user in the database
20. When a user selects a specific group, the client-application will send a get specific group request
    a. If the request is valid, the web-server will send a get group response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid and the group id refers to an existing group in the database.

21. When a user edits the name of a group, the client-application will send a edit group request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid, the group id refers to an existing group in the database, and the user is a member of said group
22. When a user deletes a group, the client-application will send a delete group request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid, the group id refers to an existing group in the database, and the user is a member of said group
23. When a user creates a group, the client-application will send a create group request
    a. If the request is valid, the web-server will send a get group response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid
24. When a user joins a group, the client-application will send a join group request
    a. If the request is valid, the web-server will send a get group response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid and the group ID refers to an existing group in the database
25. When a user leaves a group, the client-application will send a leave group request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid, the group ID refers to an existing group in the database, and the user is currently a member of said group
26. When a user (the sender) invites another user (the receiver) to join a group, the client-application will send a invite to group request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid, the group ID refers to an existing group in the database, the name sent in the request exists in the database, the sender is a member of the group, and the receiver is not a member of the group.
27. When a user creates a new chat in a group, the client-application will send a create chat request
    a. If the request is valid, the web-server will send a get chat response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid and the chat id refers to an existing group in the database.
28. When a user deletes a chat, the client-application will send a delete chat request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid, the group id refers to an existing group in the database, the user is a member of said group, the chat id

refers to an existing chat in the database, and the chat is associated with the group.

29. When a user sends a message to a specific chat, the client-application will send a message chat request
    a. If the request is valid, the web-server will send a confirmation message response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid, the group id refers to an existing group in the database, and the user is a member of said group, the chat id refers to an existing chat in the database, and the chat is associated with the group.
30. When the user is viewing a chat, the client-application will periodically send a update chat request
    a. If the request is valid, the web-server will send a get chat response
    b. If the request is not valid, the web-server will send an error response
    c. Valid is defined: if the token sent in the request is valid, the group id refers to an existing group in the database, and the user is a member of said group, the chat id refers to an existing chat in the database, and the chat is associated with the group.

Non-Functional Requirements:

1. Logging out of the application should take no more than 2 mouse-clicks.
2. Only one user can edit a file at a time.
3. The client-application and web-server should have good security, that is:
    a. Passwords should be encrypted.

ER Diagram:

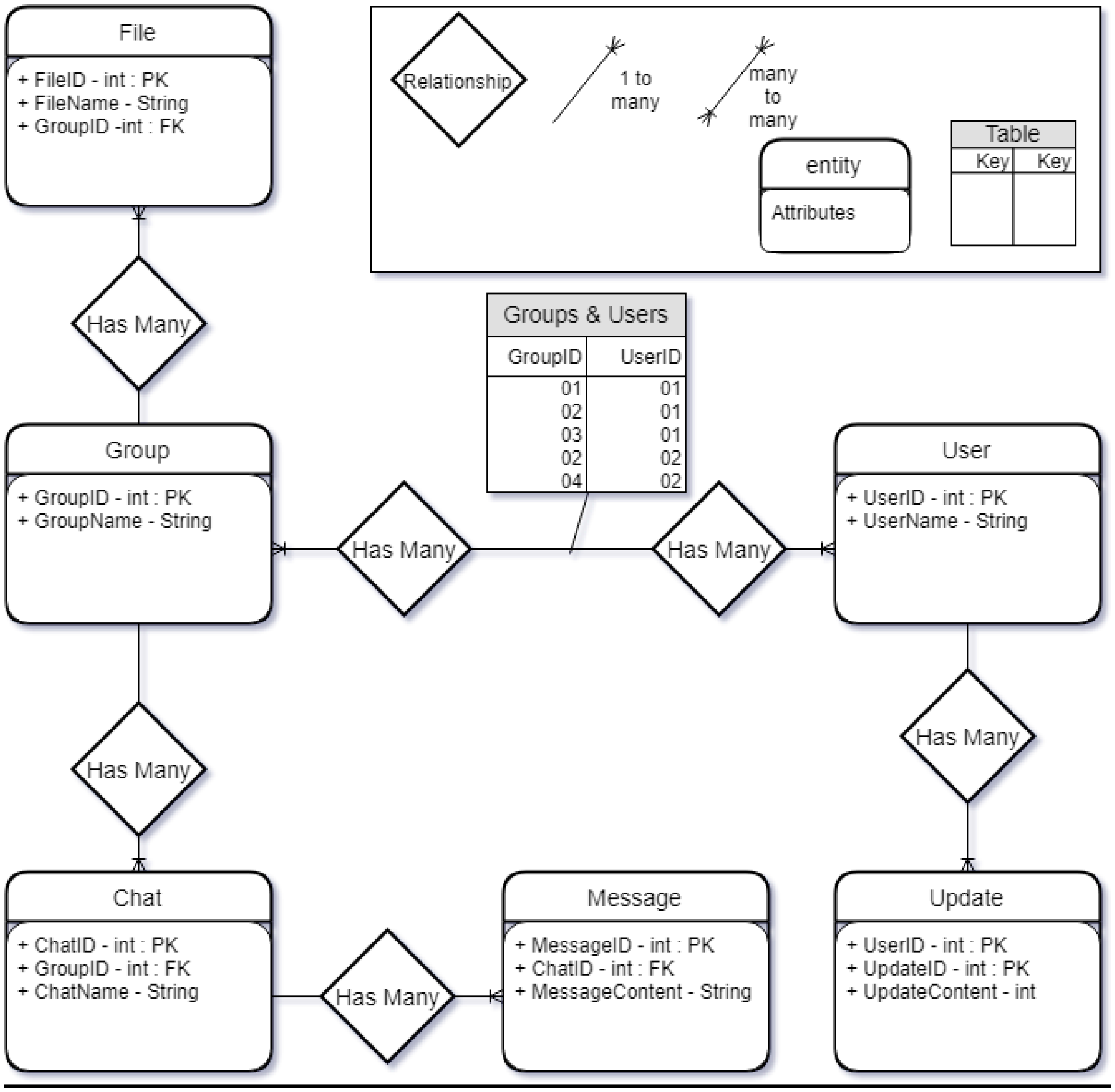


Diagram 1.2: ER diagram for Academic Collaboration Application

<u>Description of ER Diagram (1.2):</u>
Users will be directly associated with two entities, an Updates Entity and a Group entity. The Updates entity is a table that keeps track of pending user updates such as new group invites. The Group entity will contain many Users and a User will be able to belong to multiple groups. The translation table seen above outlines how that association will function.
A Group will be directly associated with two other entities, the File entity and the Chat entity. The File entity relationship refers to document files that are accessible to users of that particular group. The Chat entity relationship refers to multiple chat channels that are accessible to the users of that particular group. The Chat entity is associated with multiple Message entities, these define the specific messages that users will send to the group chat rooms.

Data Flow Diagrams

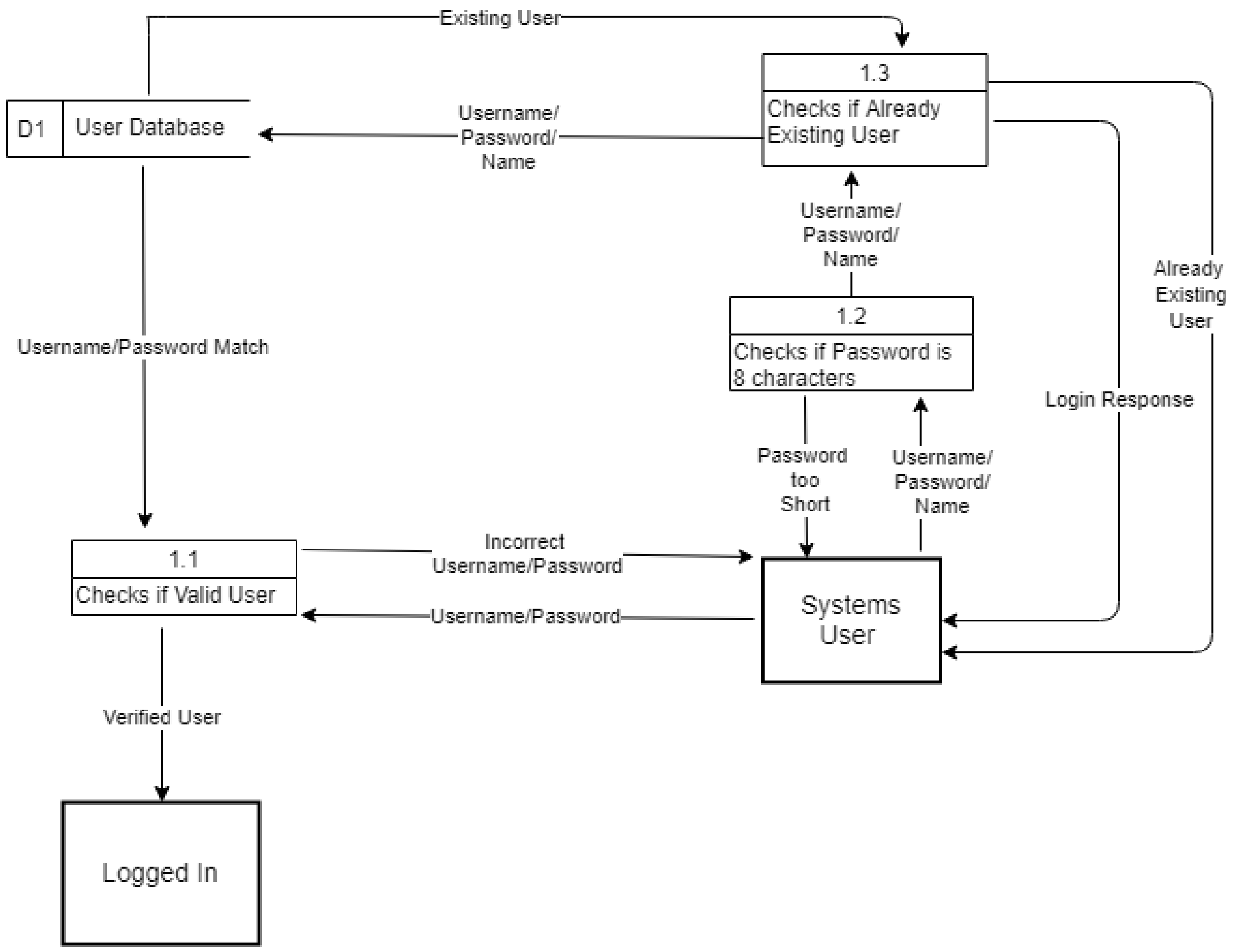


Diagram 2.0: Login/Sign Up Data Flow

<u>Description of DFD 1 (Diagram 2.0):</u>
This DFD matches UC-1 and UC-11. It illustrates the flow of data for the login and signup process.

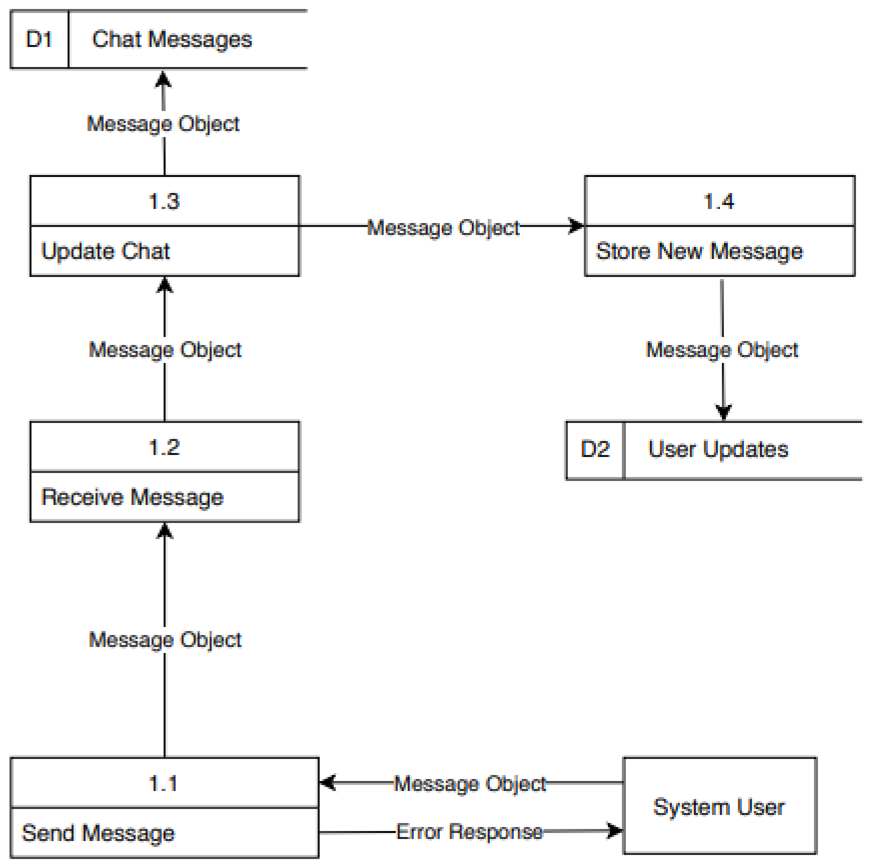


Diagram 3.0: Send Message Data Flow

<u>Description of DFD 2 (Diagram 3.0):</u>
This DFD matches UC-10. It illustrates the flow of data when a user sends a new message to a chat.

Use Cases:

Login

Name: User Login **ID:**UC-1
Created: Dec 6, 2018
Actor: System User
Description: The user enters their account information into the login form, and if successful is logged in. The system validates the users information and logs them in if the information is correct.
Trigger: The user has an account with the service, and is not currently logged in.
Type: External
Preconditions:

1. The user has a valid account
2. The user is not logged in
3. The user is on the login form

Normal Course:

1. The user will enter their username and password into the form defined below (interface diagram 1.0 in the appendix):
    a. The form contains the following fields: Username (required, text box), Password (required, text box), Submit (Button), Signup Link
    b. The submit button will be clickable once the user has entered both their username and password.
    c. If the user selects the signup link go to UC-11
2. The user clicks the submit button. The following can occur:
    a. The users username and password match an entry in the user database table, go to P1.
    b. The users username and password do not match an entry in the user database table, go to E2.

Post Conditions:
P1. Successful Login

1. The user is taken to the home page. (will define)
2. The home page will display all of the users associated groups and chats.

Exceptions:
E1. Unsuccessful Login

1. An error message is displayed to the user with the contents “Incorrect Username or Password”.
2. The username and password fields are cleared.

Grant File Access

Name: File Access **ID:**UC-2
Created: Dec 6, 2018
Actor: System User
Description: The user grants access to a a file to another user.
Trigger: The user is viewing a file and wants the share the file
Type: External
Preconditions:

1. The user is logged in
2. The user is viewing a file

Normal Course:

1. The user selects the share button (update interface diagram).
2. The file share form will be displayed, the form is defined below:
    a. The form will contain the following fields: Target User (text field, required), Submit (button).
    b. Once a target user has been entered, the submit button will become clickable.
3. The user clicks the submit button. The following can occur:
    a. If the target user exists, and does not already have access to the file, go to P1.
    b. If the target user does not exists, go to E1.
    c. If the target user does exist but already has access to the file, go to E2.

Post Conditions:

P1. Successful Share

1. The user is returned to the file view and a success dialog is shown.

Exceptions:

E1. Unsuccessful Share - target user DNE

1. A message is shown indicating the target user does not exist.
2. The user remains on the share form.
3. The target user field is cleared.

E2. Unsuccessful Share - target user has file access

1. A message is shown indicating the target user already has access to the file.
2. The user remains on the share form.
3. The target user field is cleared.

Edit File

Name: Edit File **ID:**UC-3
Created: Dec 8, 2018
Actor: System User
Description: The user to edit an already existing file
Trigger: The user is viewing a file and wants to edit the file
Type: External
Preconditions:

1. The user is logged in
2. The user is viewing a file

Normal Course:

1. The user selects the edit file button (update interface diagram).
2. The file edit form will be displayed, the form is defined below:
    a. The form will contain the following fields: Edit Title (text field, required), Edit Body (text field, required), and Submit (button).
    b. Once the user makes an edit to the title or body of the file, the submit button will become clickable.
3. The user clicks the submit button. The following can occur:
    a. If the user has permission to edit the file, go to P1.
    b. If the user does not have permission to edit the file, go to E1.

Post Conditions:
P1. Successful Edit

1. The user is returned to the file view and a success dialog is shown.

Exceptions:
E1. Unsuccessful Edit - the user does not have permission to edit the file

1. A message is shown indicating the user does not have permission to edit the file.
2. The changes made to the file are undone
3. The user is taken back to the original file

View File

Name: View File **ID:**UC-4
Created: Dec 8, 2018
Actor: System User
Description: The user to view an already existing file
Trigger: The user is wanting to view a file
Type: External
Preconditions:

1. The user is logged in
2. The user has the permission to view the file

Normal Course:

1. The user selects the view file button (update interface diagram).
    a. If the user has permission to view the file, go to P1.
    b. If the user does not have permission to view the file, go to E1.

Post Conditions:
P1. Successful View

1. The user is taken to the file view.

Exceptions:
E1. Unsuccessful View - the user does not have permission to view the file

1. A message is shown indicating the user does not have permission to view the file.
2. The user is taken back to the home page

Create File

Name: Create File **ID:**UC-5
Created: Dec 8, 2018
Actor: System User
Description: The user to create a file
Trigger: The user is wanting to create a file
Type: External
Preconditions:

1. The user is logged in
2. The user is on the Create File form

Normal Course:

1. The file edit form will be displayed, the form is defined below:
    a. The form will contain the following fields: Edit Title (text field, required), Edit Body (text field, required), Share with (drop down, not required), and Create (button).
    b. Once the user fills in the required fields, the create button will become clickable.
2. The user clicks the Create button. The following can occur:
    a. If the user is successfully created, go to P1.
    b. If the file has a duplicate name, go to E1.

Post Conditions:
P1. Successful Create

1. The user is taken to the file view.

Exceptions:
E1. Unsuccessful Create

1. A message is shown indicating the file was not created, due to “Title Already in Use”.
2. The user is taken back to create file form.

Delete File

Name: Delete File **ID:**UC-6
Created: Dec 8, 2018
Actor: System User
Description: The user to delete a file
Trigger: The user is wanting to delete a file
Type: External
Preconditions:

1. The user is logged in
2. The user has permission to view the file
3. The user has permission to delete the file, is the owner of the file

Normal Course:

1. The user selects the view file button (update interface diagram).
    a. If the user has permission to view the file, go to P1.
    b. If the user does not have permission to view the file, go to E1.
2. The user selects delete file from the menu
    a. If the user has the permission to delete the file, go to P2.
    b. If the user does not have permission to delete the file, go to E2.

Post Conditions:

P1. Successful View

1. The user is taken to the file view.

P2. Successful Deletion

1. The user is shown a message indicating the file was deleted.
2. The user is taken back to the home page.

Exceptions:

E1. Unsuccessful View - the user does not have permission to view the file

1. A message is shown indicating the user does not have permission to view the file.
2. The user is taken back to the home page

E2. Unsuccessful Deletion - the user does not have permission to delete the file

1. A message is shown indicating the user does not have permission to delete the file.
2. The user is taken back to the home page

Invite to Group

Name: Invite to Group **ID:**UC-7
Created: Dec 6, 2018
Actor: System User
Description: The user enters the username of the other user that they want to invite to a group. The system sends an invitation to the user that they want to invite.
Trigger: The user clicks the “invite” button
Type: External
Preconditions:

1. The user has a valid account
2. The user is logged in

Normal Course:

1. The user will enter the email of the user that they wish to invite to the group, and the ID of the group into the form as seen in interface diagram 1.4 in the appendix.
    a. The form contains the following fields: Username (required, text box), Send (Button)
    b. The send button will be clickable once the user has entered the user they want to invite to the group is entered.
2. The user presses the Send button
    a. If the username is in the database and the user is not already a member of the group, go to P1
    b. If the username is in the database and the user is a member of the group, go to E1
    c. If the username is not in the database, go to E2

Post Conditions:
P1. Valid username:

1. The system sends an invitation to the user associated with the given username
2. The system displays a message saying “Invitation sent”
3. All fields are cleared

Exceptions:
E1. User already member

1. An error message is displayed to the user with the contents “User already a member of group”.
2. The username field is cleared.

E1. Username does not exist

1. An error message is displayed to the user with the contents “User does not exist”.
2. The username field is cleared.

Invite to Chat

Name: Invite to Chat **ID:**UC-8
Created: Dec 6, 2018
Actor: System User
Description: The user enters the username of the other user that they want to invite to a chat. The system sends an invitation to the user that they want to invite.
Trigger: The user clicks the “invite” button
Type: External
Preconditions:

1. The user has a valid account
2. The user is logged in

Normal Course:

1. The user will enter the username of the user that they wish to invite to the chat, and the ID of the chat (interface diagram 1.4, except the title says “invite to Chat”).
    a. The form contains the following fields: Username (required, text box), Send (Button)
    b. The send button will be clickable once the user they want to invite to the chat is entered.
2. The user presses the Send button
    a. If the username is in the database and the user is not already a member of the group, go to P1
    b. If the username is in the database and the user is a member of the group, go to E1
    c. If the username is not in the database, go to E2

Post Conditions:
P1. Valid username:

1. The system sends an invitation to the user associated with the given username
2. The system displays a message saying “Invitation sent”
3. All fields are cleared

Exceptions:
E1. User already member

1. An error message is displayed to the user with the contents “User already a member of chat”.
2. The username field is cleared.

E1. Username does not exist

1. An error message is displayed to the user with the contents “User does not exist”.
2. The username field is cleared.

Select Chat

Name: Select Chat **ID:**UC-9
Created: Dec 6, 2018
Actor: System User
Description: The user clicks on the link to the chat they wish to select. The system opens that chat.
Trigger: The user clicks the link to the chat they wish to join.
Type: External
Preconditions:

1. The user has a valid account
2. The user is logged in

Normal Course:

1. The user will click on the link to the chat that they wish to join.
    a. The system take them to the chat that they clicked on.

Post Conditions:
P1. User can comment and add to the chat:.
Exceptions:
E1. The user is not a member of the chat they wish to select

1. An error message is displayed to the user with the contents "You are not a member of this chat".
2. The user is not linked to the chat.

Select Group

Name: Select Group **ID:**UC-10
Created: Dec 6, 2018
Actor: System User
Description: The user clicks on the link to the group they wish to select. The system opens that group.
Trigger: The user clicks the link to the group they wish to join.
Type: External
Preconditions:

1. The user has a valid account
2. The user is logged in

Normal Course:

1. The user will click on the link to the group that they wish to join.
    a. The system take them to the group that they clicked on.

Post Conditions:
P1. User can comment and add to the group:.
Exceptions:
E1. User is not a member of the group they wish to select

1. An error message is displayed to the group with the contents “You are not a member of this group”.
2. The user is not linked to the group.

Send Message

Name: Send Message **ID:**UC-11
Created: Dec 9, 2018
Actor: System User
Description: The user types message into empty text box to send out a message to the chat.
Trigger: The user clicks send message.
Type: External
Preconditions:

1. The user has a valid account
2. The user is logged in
3. The user has typed out a message into the text box
4. The user is successfully apart of the chat group they are attempting to send a message in.

Normal Course:

1. The user will click into the empty text box and begin populating the text box with their message.
2. The user will click the send message button when they are done typing and wish to send the message.
    a. The system will successfully send message contents to chat including username of sender.

Post Conditions:
P1. The message will be sent to the chat room in which the user is currently in.
Exceptions:
E1. Loss of connection before attempting to send message.

Creating User Account

Name: Creating a User Account **ID:**UC-12
Created: Dec 8, 2018
Actor: System User
Description: The user is creating an account by entering in their name, username and password.
Trigger: The user clicks on the sign up button.
Type: External
Preconditions:

1. The user does not have a account.

Normal Course:

1. The user will input their name, username, and password into the form defined below:
    a. The form contains the following fields: Name (required, text box), Username (required, text box), Password (required, text box), Submit (Button), Login Link
    b. The submit button will be clickable once the user has entered their name, username and password.
2. The user clicks the submit button the following can occur:
    a. If the users credentials do not match a record in the database go to P1.
    b. If the user's password is less than 8 characters go to E1.
    c. If the user credentials match a record in the database go to E2.

Post Conditions:
P1. Successful Account creation

1. The user is taken to the login form.

Exceptions:
E1. Unsuccessful Account Creation - the user's password is less than 8 characters

1. A message is shown indicating the user's password is too short.
2. The name, username and password text boxes are cleared.

E2. Unsuccessful Account Creation - the user's credentials match a record in the database.

1. A message is shown indicating the user's account already exists.
2. The name, username and password text boxes are cleared.

Create Chat

Name: Create Chat **ID:** UC-13

Created: Dec 9, 2018

Actor: System User

Description: The user is creating a new chat within a group.

Trigger: The user clicks the “add chat” button within the group page.

Type: External

Preconditions:

1. The user is signed in and on a group page.

Normal Course:

1. The user will input the chat name into the form defined below (interface diagram 1.2 in the appendix):
    a. The form will contain the following fields: Name (required, text box), Submit (button).
2. The user clicks the submit button, the following can occur:
    a. The chat name is distinct, go to P1.
    b. The chat name is not distinct, go to E1.

Post Conditions:

P1. Successful chat creation

1. The user is taken to the newly created chat.

Exceptions:

E1. Unsuccessful chat creation

1. The user remains on the chat creation form.
2. A message displays to the user that the chat name is not distinct.

Create Group

Name: Create Group **ID**: UC-14
Created: Dec 9, 2018
Actor: System User
Description: The user is creating a new group.
Trigger: The user clicks the “create group” button from their homepage.
Type: External
Preconditions:

1. The user is signed in and on their homepage.

Normal Course:

1. The user will input the group name and group icon into the form defined below (interface form 1.1 in the appendix):
    a. The form will contain the following fields: Name (required, text box), Icon (optional, media submission), Submit (button).
2. The user clicks the submit button, the following can occur:
    a. The group name is distinct, the icon is of the format .png, go to P1.
    b. The group name is not distinct, go to E1.
    c. The icon is not of the format .png, go to E2.

Post Conditions:
P1. Successful group creation

1. The user is taken to the default chat of the new group.

Exceptions:
E1. Unsuccessful group creation, indistinct name

1. The user remains on the group creation form.
2. The user is notified via a message that the group name is not distinct to their account.

E1. Unsuccessful group creation, icon error

1. The user remains on the group creation form.
2. The user is notified via a message that the submitted icon must be of the format .png.

Delete Chat

Name: Delete Chat **ID:** UC-15
Created: Dec 9, 2018
Actor: System User
Description: The user deletes a chat from a group, or edits the chat.
Trigger: The user clicks the “settings” button within the selected chat page.
Type: External
Preconditions:

1. The user is signed in and on the specific chat page they wish to delete.

Normal Course:

1. The user will complete the form defined below:
    a. The form will contain the following fields: Edit (Button), Delete (Button).
2. The user clicks the delete button, the following can occur:
    a. The group is deleted, go to P1.
3. The user clicks the edit button, the following can occur:
    a. The edit chat form is displayed, go to E1

Post Conditions:
P1: Successful chat deletion

1. The user is taken to the default chat of the group they are currently in.
2. The selected chat is deleted.

Exceptions:
E1: Edit Chat Selected

1. Go to UC-21

Delete Group

Name: Delete Group **ID:** UC-16

Created: Dec 9, 2018

Actor: System User

Description: The user deletes one of their owned groups, or edits the group.

Trigger: The user clicks the “settings” button within the selected group page.

Type: External

Preconditions:

1. The user is signed in and is the owner of the specific group they wish to delete.

Normal Course:

1. The user will complete the form defined below:
    a. The form will contain the following fields: Edit (Button), Delete (Button).
2. The user clicks the delete button, the following can occur:
    a. The group is deleted, go to P1.
3. The user clicks the edit button, the following can occur:
    a. A new menu is displayed, go to E1.

Post Conditions:

P1: Successful group deletion

1. The user is taken to their home group page.
2. The selected group is deleted.

Exceptions:

E1: Edit Group Selected

1. Go to UC-20

Update Username and Password

Name: User Login **ID:**UC-17
Created: Dec 6, 2018
Actor: System User
Description: The user changes their password and/or username
Trigger: The user is currently logged in and click on “Change Username/Password”
Type: External
Preconditions:

1. The user has a valid account
2. The user is logged in
3. The user has selected “Change Username/Password” on the “User Settings” menu

Normal Course:

1. The user will enter their current password and their new password and/or username into the form as seen in interface diagrams 1.3 in the appendix
    a. The form contains the following fields: Current Password (required, text box), New Username (optional, textbox), New Password (optional, textbox), Confirm New Password (optional, text box), Submit (Button)
    b. The submit button will be clickable under the following conditions:
        i. If the current password and new username fields are filled out, the button will be clickable
        ii. If the current password, new password, and confirm new password fields are filled out and the new password matches the confirm new password, the button will be clickable
        iii. If all fields are filled out and the new password field matches the confirm new password field, the button will be clickable
        iv. In all other scenarios, the button will not be clickable
2. The user clicks the submit button. The following can occur:
    a. The current password does not match the user’s password. Go to E1
    b. The current password matches the new password or the current username matches the new username. Go to E2
    c. The new username is already taken. Go to E3
    d. The current password matches the user’s actual password and the new username is unique (if changed) and the new password is different from the current password (if changed). Go to P1

Post Conditions:
P1. Successful Login

1. A confirmation message indicates that the username and password have been changed
2. All fields on the form are cleared

Exceptions:
E1. Incorrect Password

1. An error message is displayed to the user with the contents “Password Incorrect”.

2. The current password, new password, and confirm new password fields are cleared

E2. Username and Password Unchanged

1. An error message is displayed to the user with the contents “Username and password match current username and password.”
2. All fields on the form are cleared.

E3. New Username Taken

1. An error message is displayed to the user with the contents “Username already taken”
2. All fields on the form are cleared.

Logout

Name: User Login **ID:**UC-18
Created: Dec 6, 2018
Actor: System User
Description: The user logs out of the system
Trigger: The user is logged in and wishes to log out
Type: External
Preconditions:

1. The user is logged in

Normal Course:

1. The user selects the “Log out” button under the “User Settings” menu. Go to P1

Post Conditions:
P1. Successful Log out

1. The application takes the user to the log in form as seen in UC - 1

Respond to invitation

Name: User Login **ID:**UC-19
Created: Dec 6, 2018
Actor: System User
Description: The user either accepts or declines an invitation to a group or chat, which was sent by another user in UC-7 or UC-8
Trigger: The user is currently logged in and another user has sent an invitation to the user.
Type: External
Preconditions:

1. The user is logged in.
2. Another user has sent an invitation to the user
    a. If the invitation was received when the user was logged out, this occurs immediately after a successful login
    b. If the invitation was received while the user is logged in, this occurs immediately

Normal Course:

1. A pop-up window appears over the application indicating to a user that they have been invited to a group. The window contains a Decline button and an Accept button (interface diagram 1.5 in the appendix).
    a. If the user selects “Decline”, go to P1
    b. If the user selects “Accept”, go to P2

Post Conditions:

P1. Invitation Declined

1. The pop-up disappears

P2. Invitation Accepted

1. The pop-up disappears
2. The list of chats and groups the user is a member of, as seen on the left hand side of the home screen, is refreshed (and will thus contain the newly added group or chat).

Edit Group

Name: Delete Group **ID:** UC-20
Created: Dec 9, 2018
Actor: System User
Description: The user changes the name and icon of a group
Trigger: The user clicks the “edit” button in the settings of a group
Type: External
Preconditions:

1. The user is signed in and is the owner of the specific group they wish to edit
2. The user has selected the “settings” button on a group and then have selected the “edit button” on that menu, as seen in UC-16

Normal Course:

1. The user is taken to the edit group form, defined below (interface diagram 1.3, except the title is “Edit Group”)
    a. The form will contain the following fields:
        i. Name (Required, text, autofill to current group name), Icon (optional, media submission, autofill to current icon), Submit (Button)
    b. The user clicks the submit button:
        i. The group is edited successfully, go to P1.
        ii. The new name is not distinct to the user, go to E1
        iii. The new icon is not a .png, go to E2

Post Conditions:
P1: Group Edit Success

1. The user is taken to the default chat of the group
2. The group name has been updated

Exceptions:
E1: Name is not distinct

1. The user remains on the group edit form
2. The user is notified via a message that the name must be distinct to their groups.

E2: Image is not .png

1. The user remains on the group edit form
2. The user is notified via a message that the icon must be of the format .png.

Edit Chat

Name: Delete Group **ID:** UC-21
Created: Dec 9, 2018
Actor: System User
Description: The user changes the name of a chat
Trigger: The user clicks the “edit” button of the chat settings
Type: External
Normal Course:

1. The user is taken to the edit chat form, defined below (interface 1.2 expect the title is “Update Chat”):
    a. The form will contain the following fields:
        i. Name (Required, text, autofill to current group name), Submit (Button)
    b. The user clicks the submit button:
        i. The group is edited successfully, go to P1.
        ii. The new name is not distinct to the user, go to E1

Post Conditions:
P1: Chat Edit Success

1. The user is taken to the chat
2. The chat name has been updated

Exceptions:
E1: Name is not distinct

1. The user remains on the chat edit form
2. The user is notified via a message that the name must be distinct to their chats.

Deliverables:

1. A complete requirements document including system architecture, system requirements, syntax and semantics of messages, etc. and appendices as appropriate.
2. Design documents detailing all system models required by the course.
3. Fully-functioning implementation of system components.
4. Source code of the components.
5. A short video demonstrating the main functions of the system.
6. Test documents as required by the course.

Platforms:

The application will be targeted at a windows desktop environment

Similar Software:

The application will function in a way similar to Discord. Discord is a group oriented desktop chat app that organizes its' groups intro "servers". See reference 2 for more information. The application will have editing functionality similar to Google Docs, albeit less advanced. In Google Docs, users are able to share files with each-other and edit them in real-time all at once. See reference 1 for more information.

References:

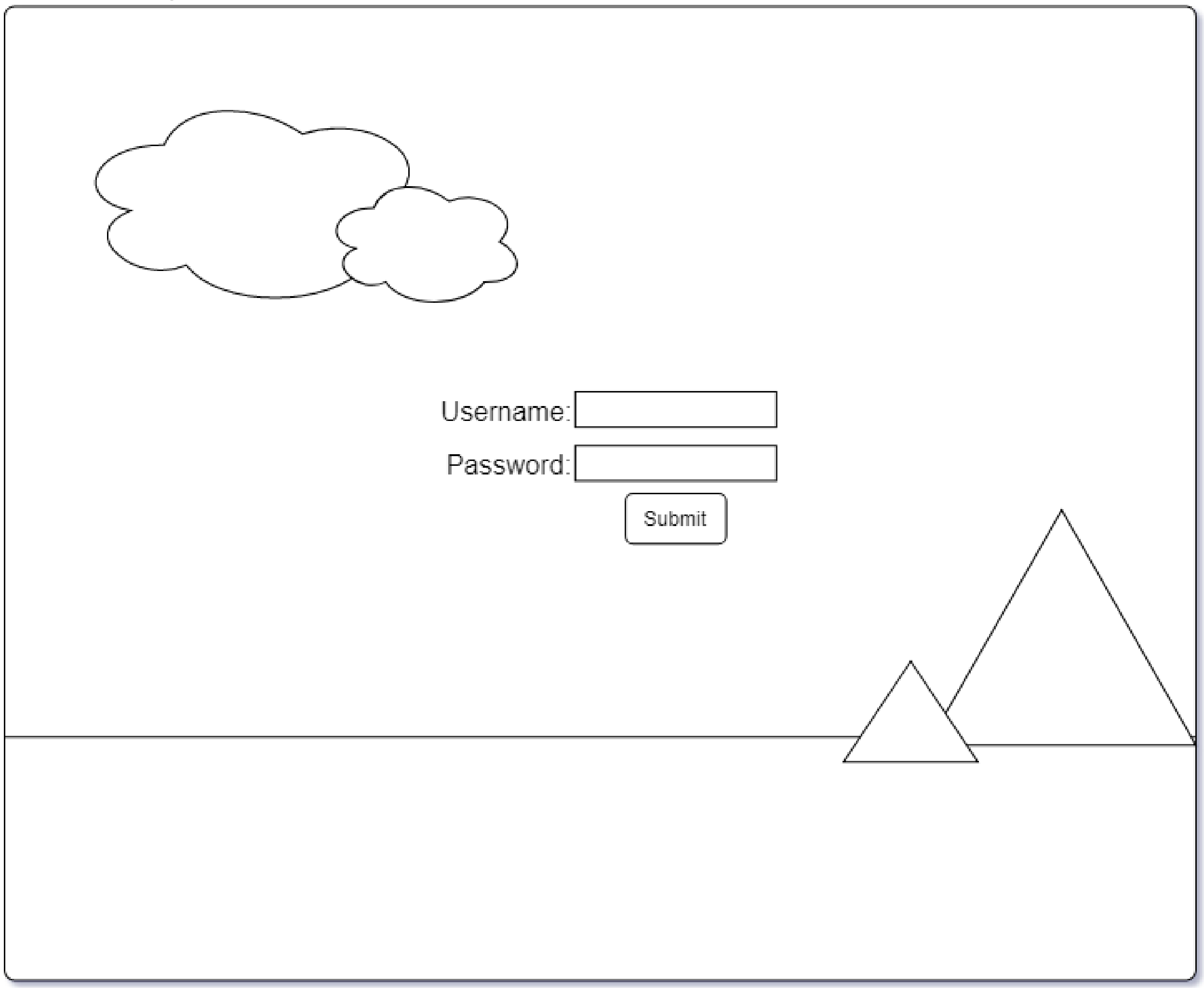
Alphabet. "Google Docs." [Web application software]. Retrieved from https://docs.google.com.

Discord. "Discord." [Desktop application software]. Retrieved from https://discordapp.com/download.

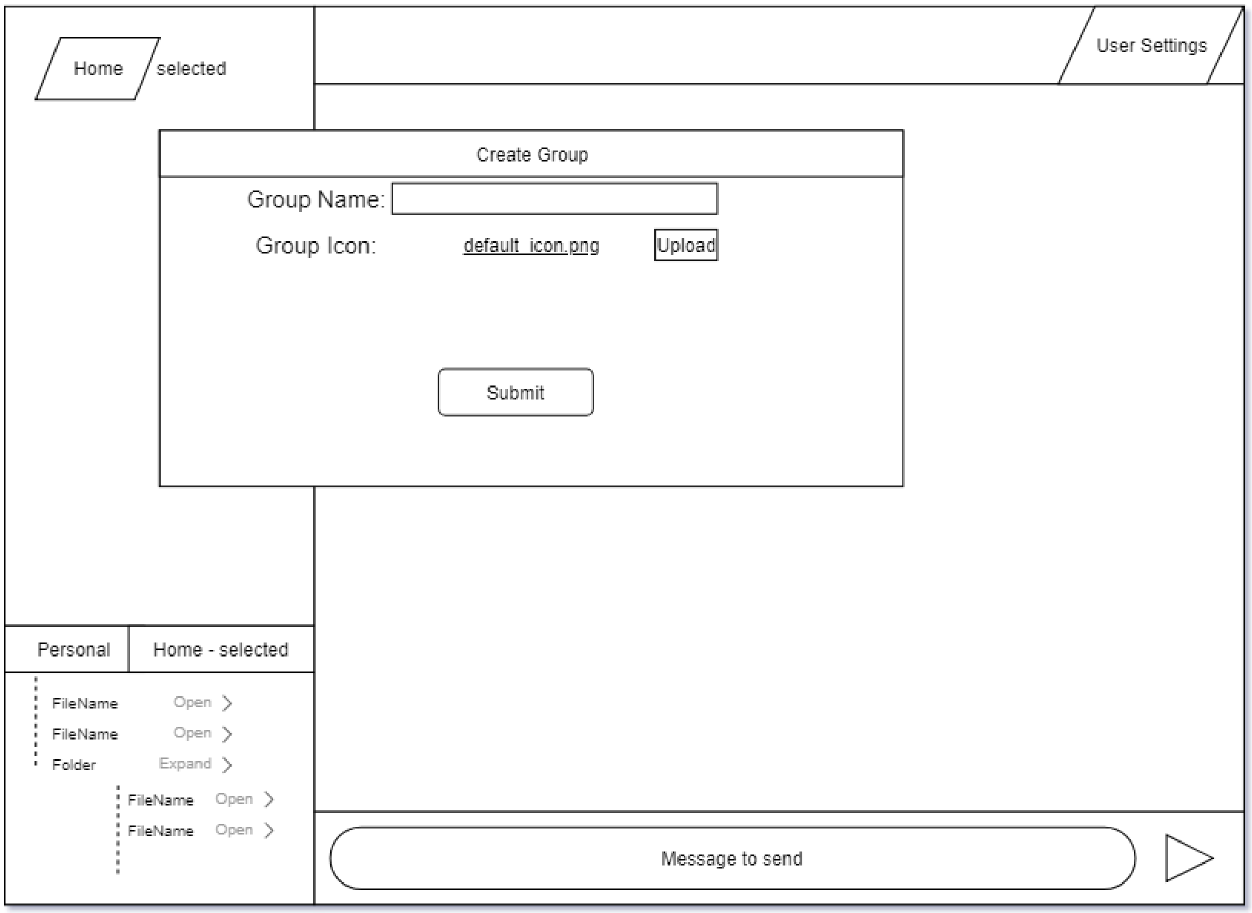
ECMA International. "Introducing JSON." Retrieved from https://www.json.org/.

Appendix:

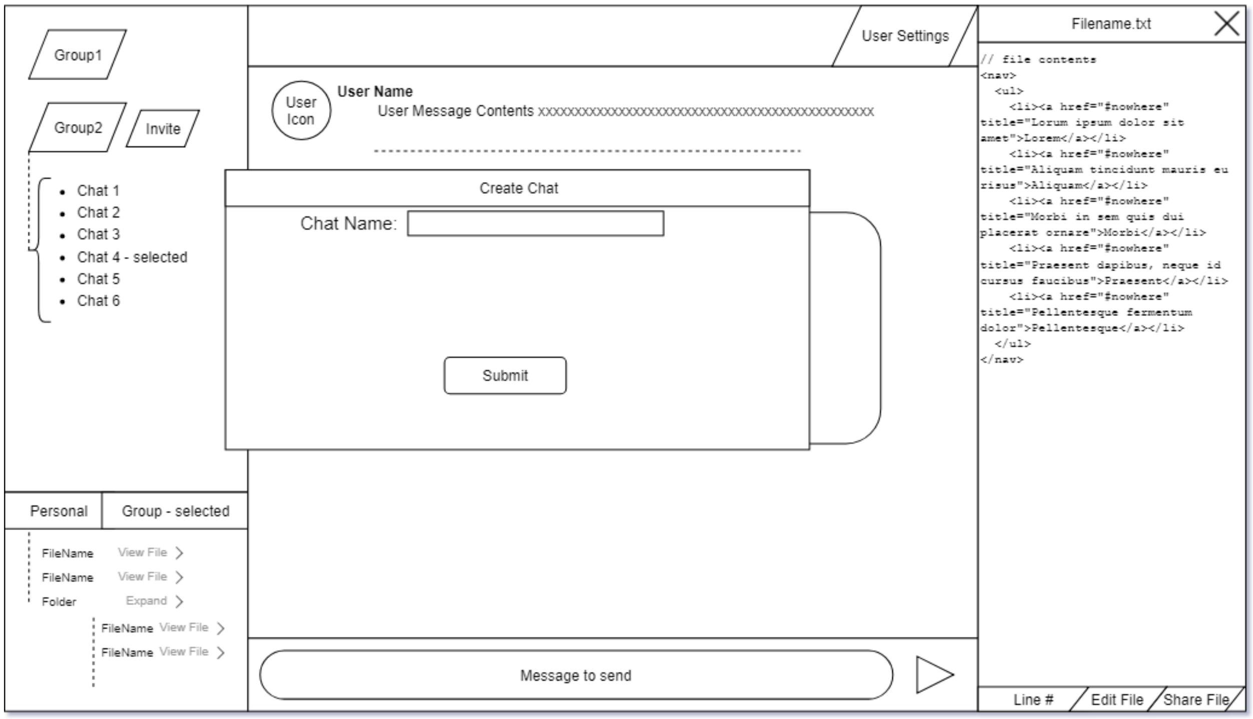
Interface Diagram 1.0:



Interface Diagram 1.1:

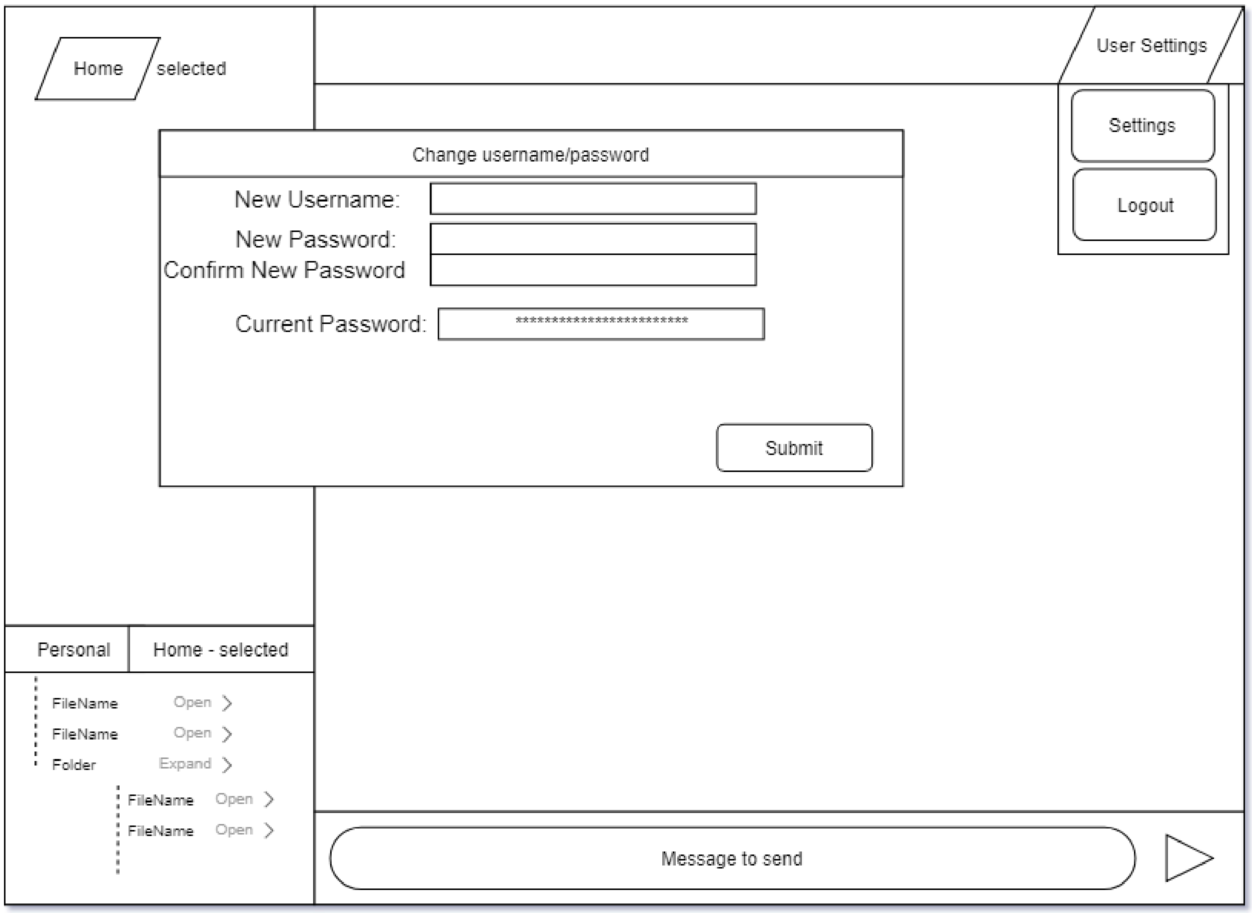


Interface Diagram 1.2:

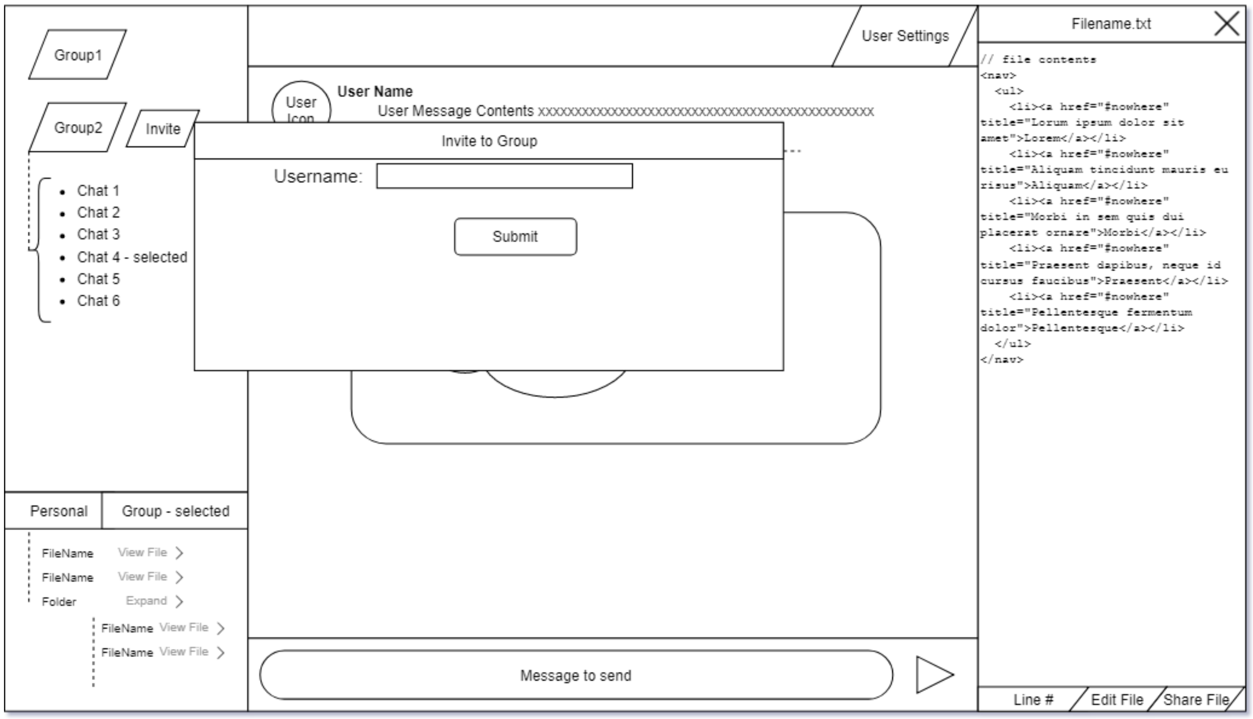


```
// file contents
<nav>
  <ul>
    <li><a href="#nowhere"
title="Lorum ipsum dolor sit
amet">Lorem</a></li>
    <li><a href="#nowhere"
title="Aliquam tincidunt mauris eu
risus">Aliquam</a></li>
    <li><a href="#nowhere"
title="Morbi in sem quis dui
placerat ornare">Morbi</a></li>
    <li><a href="#nowhere"
title="Praesent dapibus, neque id
cursus faucibus">Praesent</a></li>
    <li><a href="#nowhere"
title="Pellentesque fermentum
dolor">Pellentesque</a></li>
  </ul>
</nav>
```

Interface Diagram 1.3:

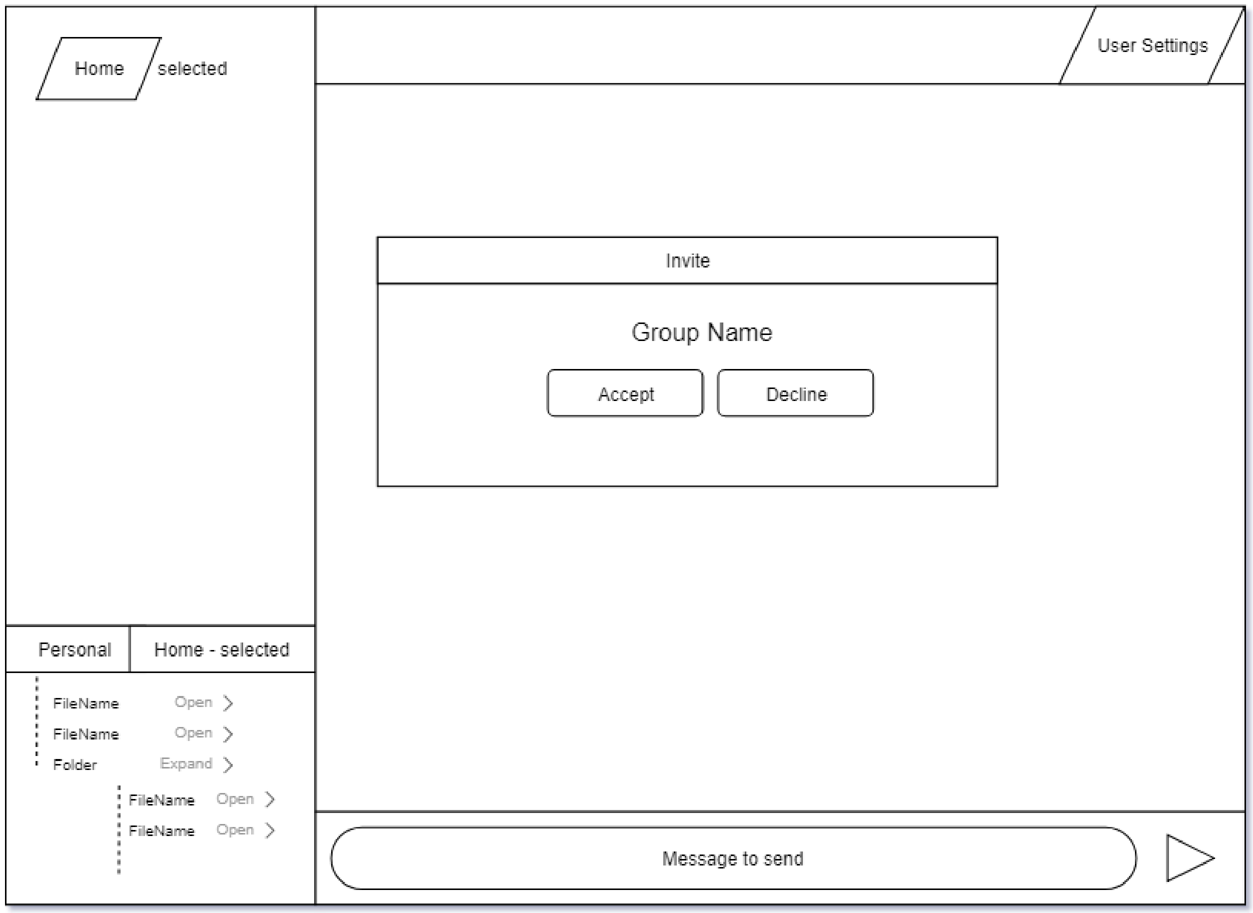


Interface Diagram 1.4:



```
// file contents
<nav>
  <ul>
    <li><a href="#nowhere"
title="Lorum ipsum dolor sit
amet">Lorem</a></li>
    <li><a href="#nowhere"
title="Aliquam tincidunt mauris eu
risus">Aliquam</a></li>
    <li><a href="#nowhere"
title="Morbi in sem quis dui
placerat ornare">Morbi</a></li>
    <li><a href="#nowhere"
title="Praesent dapibus, neque id
cursus faucibus">Praesent</a></li>
    <li><a href="#nowhere"
title="Pellentesque fermentum
dolor">Pellentesque</a></li>
  </ul>
</nav>
```

Interface Diagram 1.5:


Home
selected
User Settings
Invite
Group Name
Accept
Decline
Personal
Home - selected
FileName Open
FileName Open
Folder Expand
FileName Open
FileName Open
Message to send