Team

Ellie Frost
Michelle Ma

Project

Glimmr

URL

http://classwork.engr.oregonstate.edu:3003/index.html

Summary

This section provides a high-level overview of the feedback received and design decisions made over the course of this phase of development. The purpose of these changes was to improve clarity, efficiency, and UI/UX.

The first half of development focused on design clarity and consistency. For instance, we renamed the `matches` table to `connections` to better reflect its purpose. There were also necessary changes to the schema and ER diagram to better align with convention. (`connections` being both an entity and an intersection table is a bit unconventional.)

Our use of stored procedures for CRUD operations and database-level validation is also a bit unconventional, so we added block-level comments to make it clear at a glance what we're doing.

Early on, we decided to omit the `blocks` and `reports` tables from this phase of development. In a similar vein, we added an `is_deleted` attribute to `connections` to support unmatching without losing information that may be necessary, for instance, to follow up on reports.

We also dropped support for `UPDATE` and `DELETE` for `likes` and `messages`. Conceptually, these are immutable events, and they’re deleted automatically when the associated connection is deleted. This also reduced complexity.

The second half of development focused on UI/UX improvements. These included supporting back navigation, adding top navigation with `position: sticky` to all pages, expanding the clickable area for links, making notes regarding custom constraints more prominent, and assisting users in inputting valid data (e.g., ID order in `connections`).

There were several feedback items we chose not to act on. For instance, we didn't feel it was necessary to include demographic or statistical data in our project overview. There was also confusion about the `date` attributes, with several reviewers thinking they were redundant.

The next phase of development will likely add `blocks` and `reports` back in as well as modifying `users` and `likes` to support profiles and swiping.

Feedback

The following reviewers provided helpful feedback or suggestions: Adrianna Hoffman, Alexandra Orlova, Alvin Li, Connor Wallace, James Cole, and Jennifer Putsche (database design); August Le, Callum Pickard, Charles Tang, Erik Christiansen, Grant Hopkin, Gilda Duarte, Muhammad Akbar, and Jackson Miller (UI/UX design); and Arianna Joffrion, Grace Kohler, and Siya Sonpatki (documentation).

Overview

Our project is the back end for Glimmr, a pilot program for a dating app serving a test group of 1,000 users for eight weeks (the average expected lifetime of an account). Glimmr is our attempt to address the gender gap on dating apps by putting women’s user experience first. We believe that in doing so, we’ll create a better experience for everyone. The cornerstones of our design philosophy are safety, honesty, and consent. In this phase of development, however, we’re only implementing the necessary functionality for basic user interaction in the form of liking, matching, and messaging.

In terms of scope, we expect the average user to be active for 8 weeks and 3.5 days a week. For simplicity, we can think of this as 4 weeks and 7 days a week. Likes are limited to 10 per day, but there’s no limit on views. If the average user uses all 10 likes, that’s 280 likes total per user. If 1/10 likes results in a match, that’s 1 match per user per day. If each match results in 100 messages over 2 weeks, that’s 50 messages per user per day, or 1,400 messages total per user. In summary, we would expect our test group to generate 280,000 likes and 1,400,000 messages. These are ballpark estimates, but the result is likely to be less than a gigabyte of data.

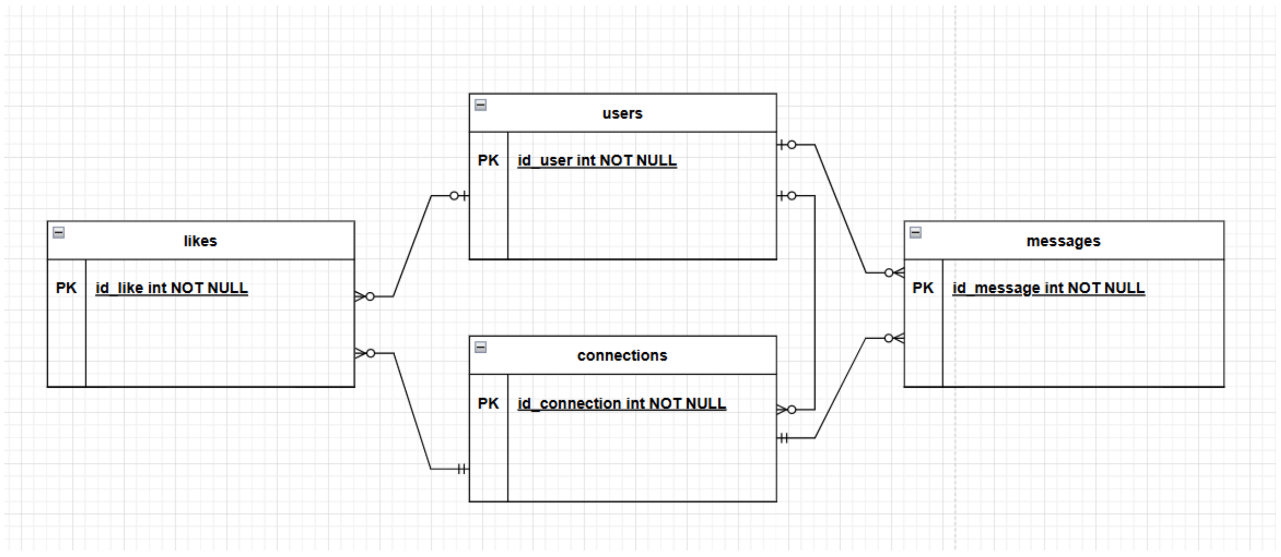
Outline

Users will be able to swipe through a feed of other users' profiles according to their preferences and like profiles that interest them. If two users like each other, they'll appear in each other's matches, where they can exchange messages. The `connections` table represents the many-to-many relationship between `users` and itself. It's a reference point for `likes` and `messages,` each of which it has a one-to-many relationship with. In the current model, if a `user` record is deleted, any associated records will be retained, but the user ID will be nullified. This can lead to situations where a `connection` has only one user ID or a `message` has no user ID. This is a quirk of the current implementation.

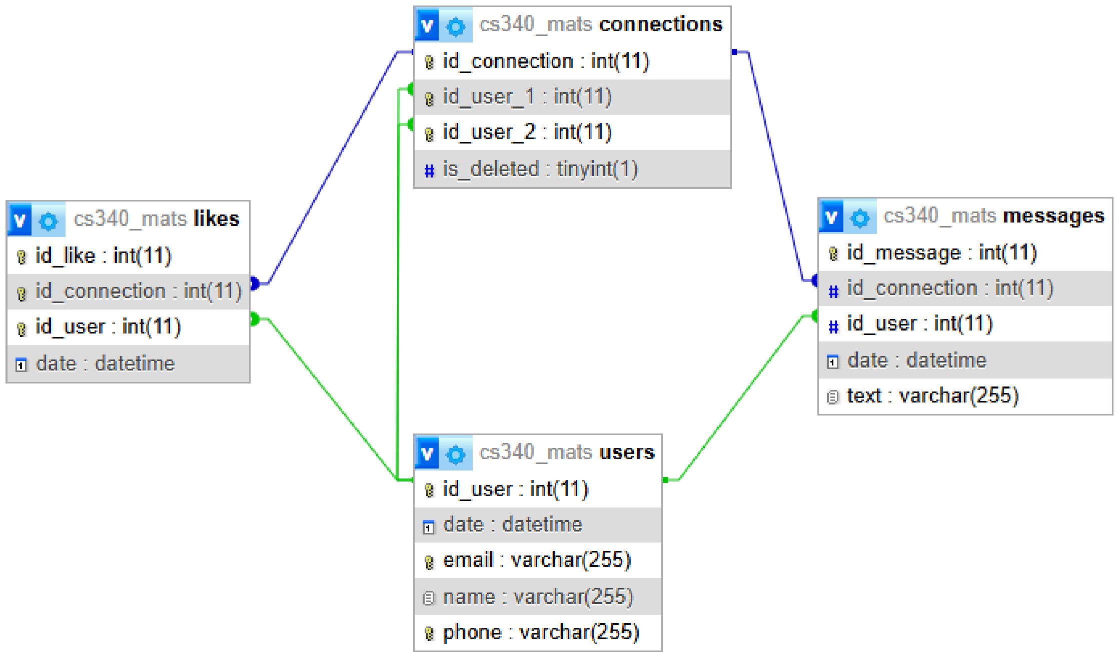
- connections - `connections` represents any interaction between two users
    - attributes
        - id_connection - int, not null, pk
        - id_user_1 - int, null, fk
        - id_user_2 - int, null, fk
        - is_deleted - bool, not null
    - relationships
        - 0-2 likes - one-to-many, optional
        - 0+ messages - one-to-many, optional
        - 0-2 users - many-to-many, optional
    - constraints
        - there can be at most one connection for any two users
        - the smaller user id must come first
- likes - `likes` represents that a user has liked the other user in a connection
    - attributes
        - id_like - int, not null, pk
        - id_connection - int, not null, fk
        - id_user - int, null, fk
        - date - datetime, not null
    - relationships
        - 1 connection - many-to-one, required

        - 1 user - many-to-one, optional
    - constraints
        - there can be at most one like per user and connection
        - there can be at most two likes per connection
        - the user must be part of the connection
        - the connection must not be deleted
- messages - `messages` represents a message a user has sent to the other user in a connection
    - attributes
        - id_message - int, not null, pk
        - id_connection - int, not null, fk
        - id_user - int, null, fk
        - date - datetime, not null
        - text - varchar, not null
    - relationships
        - 1 connection - many-to-one, required
        - 1 user - many-to-one, optional
    - constraints
        - the connection must have two likes
        - the user must be part of the connection
        - the connection must not be deleted
- users - `users` represents users and their interactions with the app and other users (passwordless authentication is used)
    - attributes
        - id_user - int, not null, pk
        - date - datetime, not null
        - email - varchar, not null, unique
        - name - varchar, not null
        - phone - varchar, not null, unique
    - relationships
        - 0+ connections - one-to-many, optional
        - 0+ likes - one-to-many, optional
        - 0+ messages - one-to-many, optional
    - constraints
        - n/a

ER Diagram


users
PK id_user int NOT NULL
likes
PK id_like int NOT NULL
connections
PK id_connection int NOT NULL
messages
PK id_message int NOT NULL

Schema


cs340_mats connections
id_connection : int(11)
id_user_1 : int(11)
id_user_2 : int(11)
is_deleted : tinyint(1)
cs340_mats likes
id_like : int(11)
id_connection : int(11)
id_user : int(11)
date : datetime
cs340_mats messages
id_message : int(11)
id_connection : int(11)
id_user : int(11)
date : datetime
text : varchar(255)
cs340_mats users
id_user : int(11)
date : datetime
email : varchar(255)
name : varchar(255)
phone : varchar(255)

Sample Data

connections			
id_connection	id_user_1	id_user_2	is_deleted
1	1	2	TRUE
2	1	NULL	FALSE
3	NULL	NULL	FALSE

likes			
date	id_connection	id_like	id_user
2525-01-03 13:30:00	2	1	1
2525-01-04 18:00:00	2	2	NULL
2525-01-12 13:30:00	1	3	1
2525-01-13 15:00:00	1	4	2

messages				
date	id_connection	id_message	id_user	text
2525-01-05 13:30:00	1	3	1	[REDACTED]
2525-01-07 13:30:00	2	2	1	[REDACTED]
2525-01-14 18:00:00	2	1	NULL	[REDACTED]

users				
date	email	id_user	name	phone
2525-01-02 13:30:00	alex@example.com	1	Alex	555-111-1111
2525-01-11 15:00:00	taylor@example.com	2	Taylor	555-222-2222
2525-01-16 16:30:00	riley@example.com	3	Riley	555-333-3333

UI Pages

Figure 1.1. CREATE users (users/create.html)

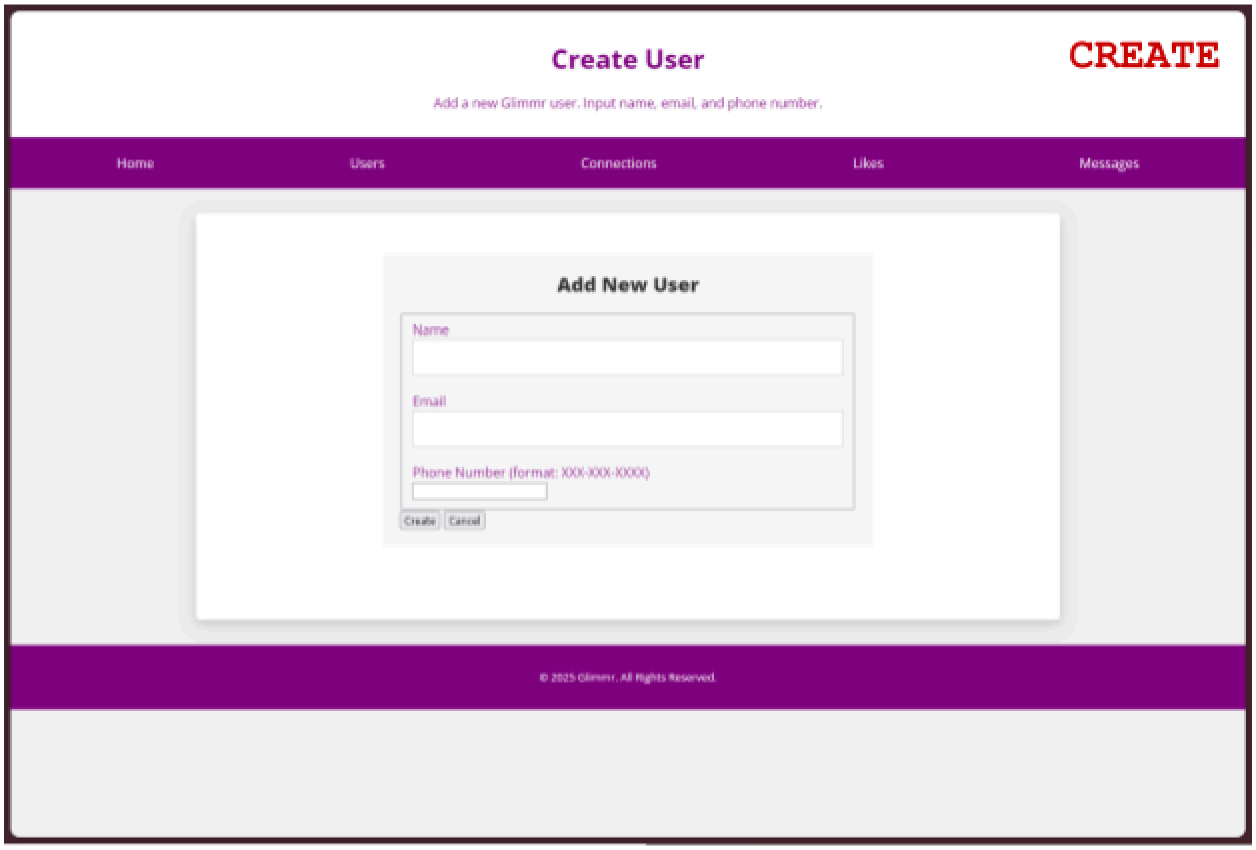
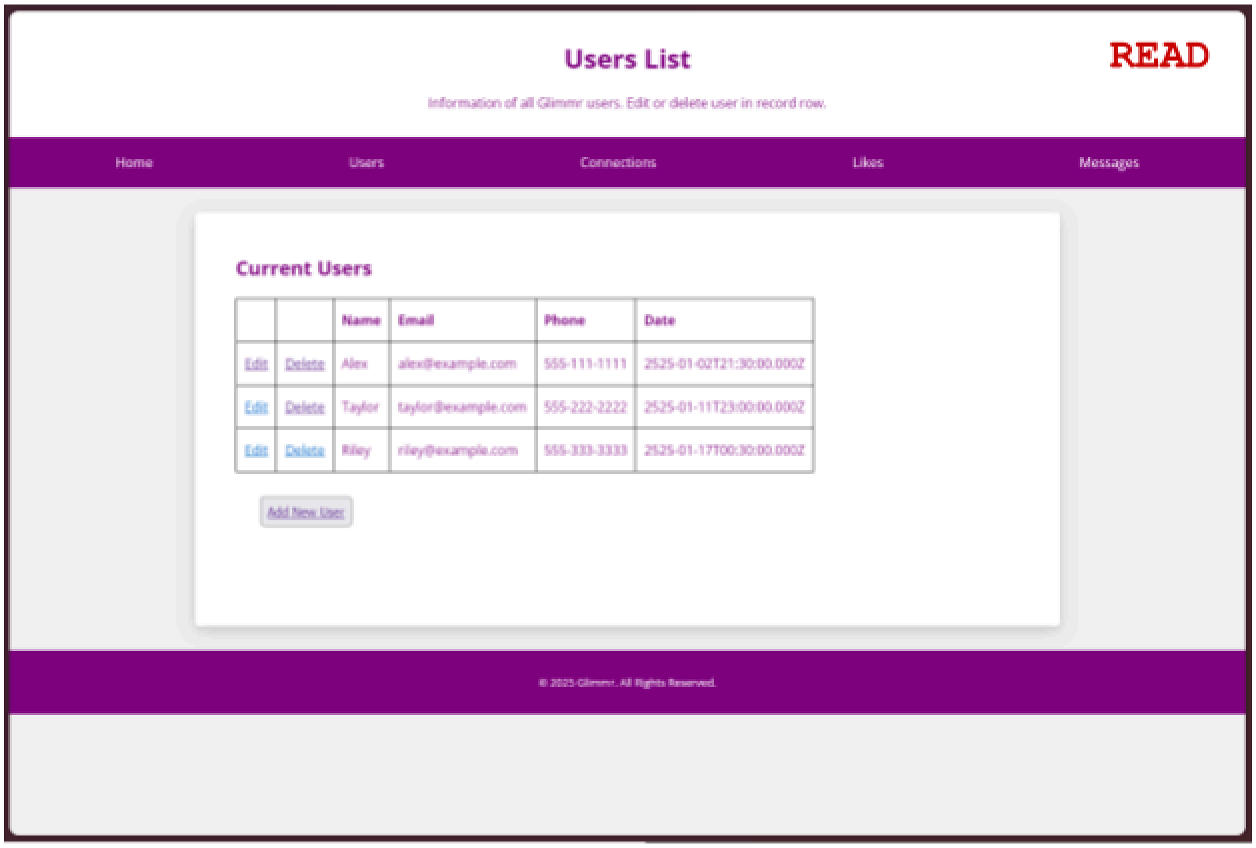

CREATE
Create User
Add a new Glimmr user. Input name, email, and phone number.
Home
Users
Connections
Likes
Messages
Add New User
Name
Email
Phone Number (format: XXX-XXX-XXXX)
Create
Cancel
© 2025 Glimmr. All Rights Reserved.

Figure 1.2. READ users (users/read.html)



Users List

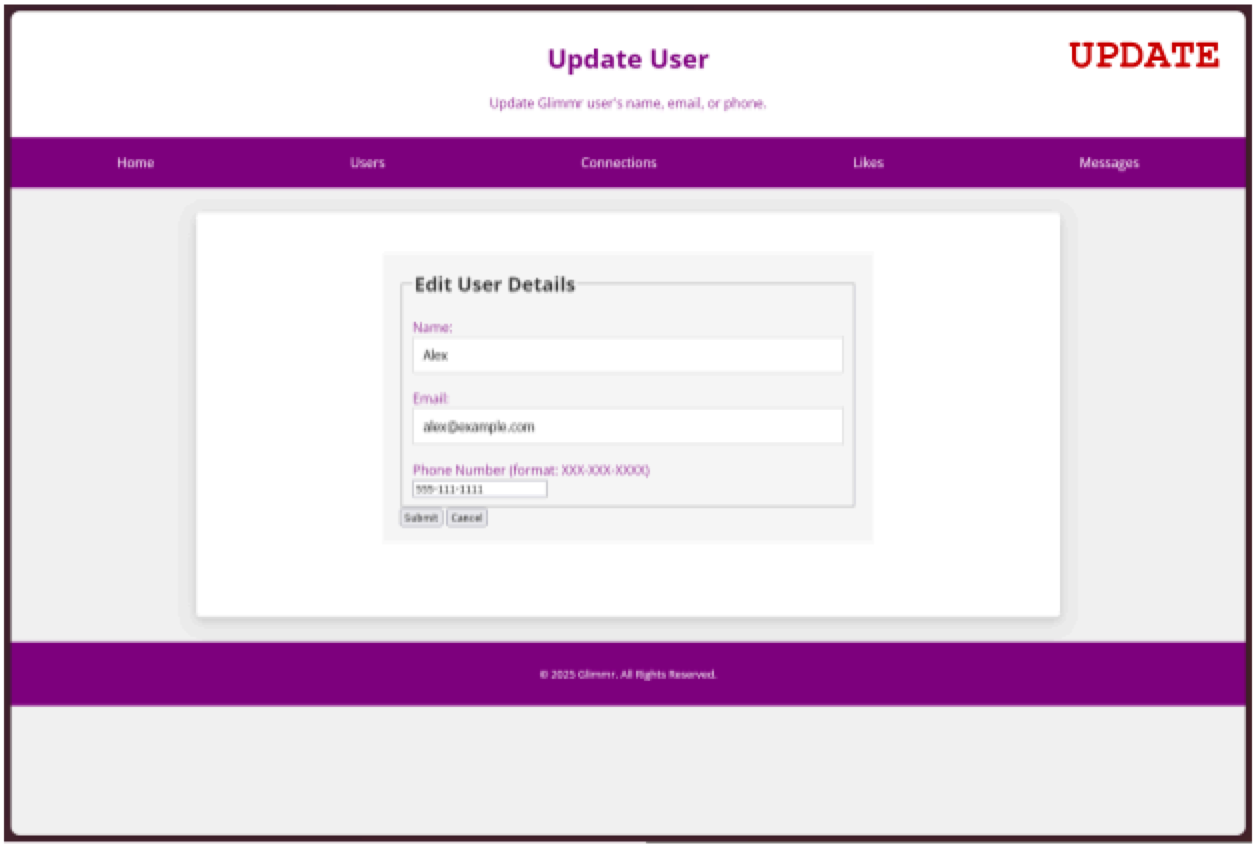
Information of all Glimmr users. Edit or delete user in record row.

Current Users

		Name	Email	Phone	Date
Edit	Delete	Alex	alex@example.com	555-111-1111	2525-01-02T21:30:00.000Z
Edit	Delete	Taylor	taylor@example.com	555-222-2222	2525-01-11T23:00:00.000Z
Edit	Delete	Riley	riley@example.com	555-333-3333	2525-01-17T00:30:00.000Z

© 2025 Glimmr. All Rights Reserved.

Figure 1.3. UPDATE users (users/update.html)



Update User

UPDATE

Update Glimmr user's name, email, or phone.

Edit User Details

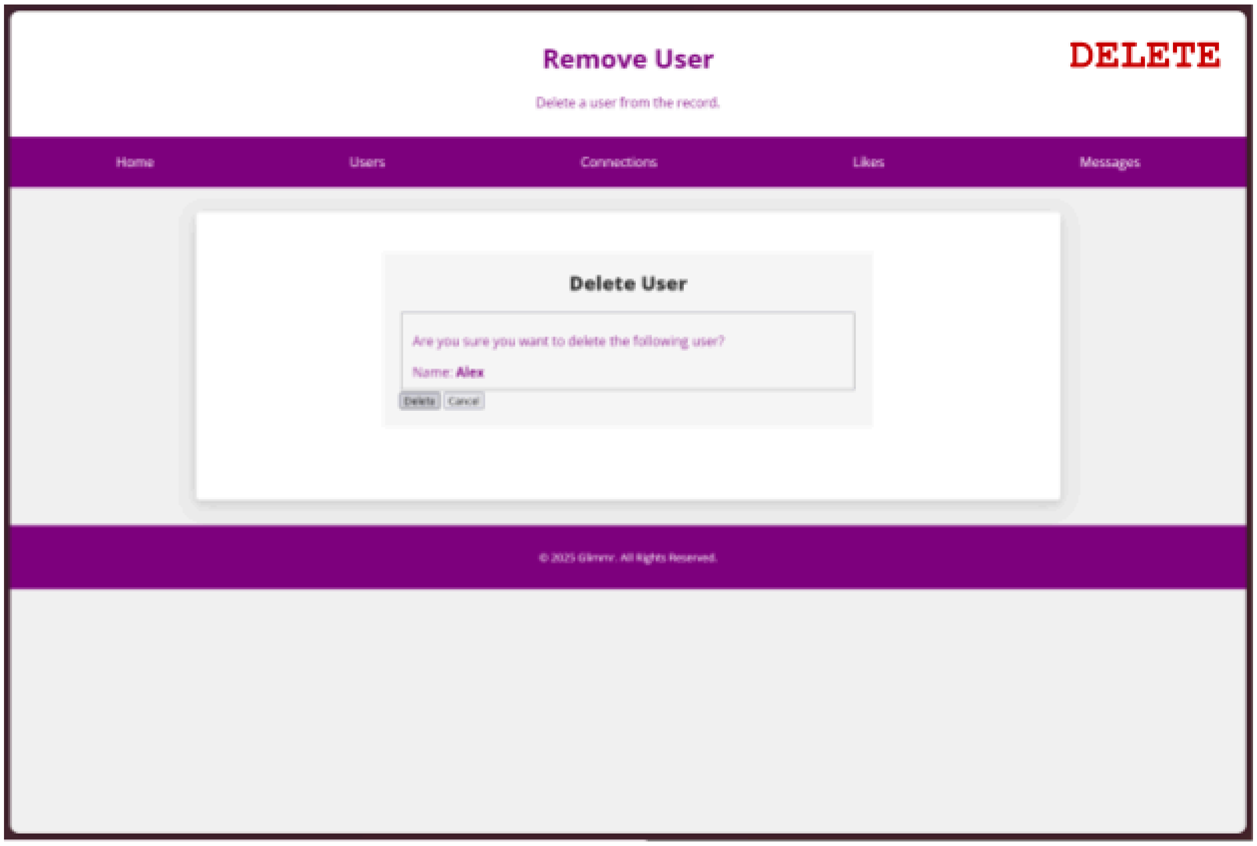
Name:
Alex

Email:
alex@example.com

Phone Number (format: XXX-XXX-XXXX)
555-111-1111

© 2025 Glimmr. All Rights Reserved.

Figure 1.4. DELETE users (users/delete.html)



DELETE

Remove User

Delete a user from the record.

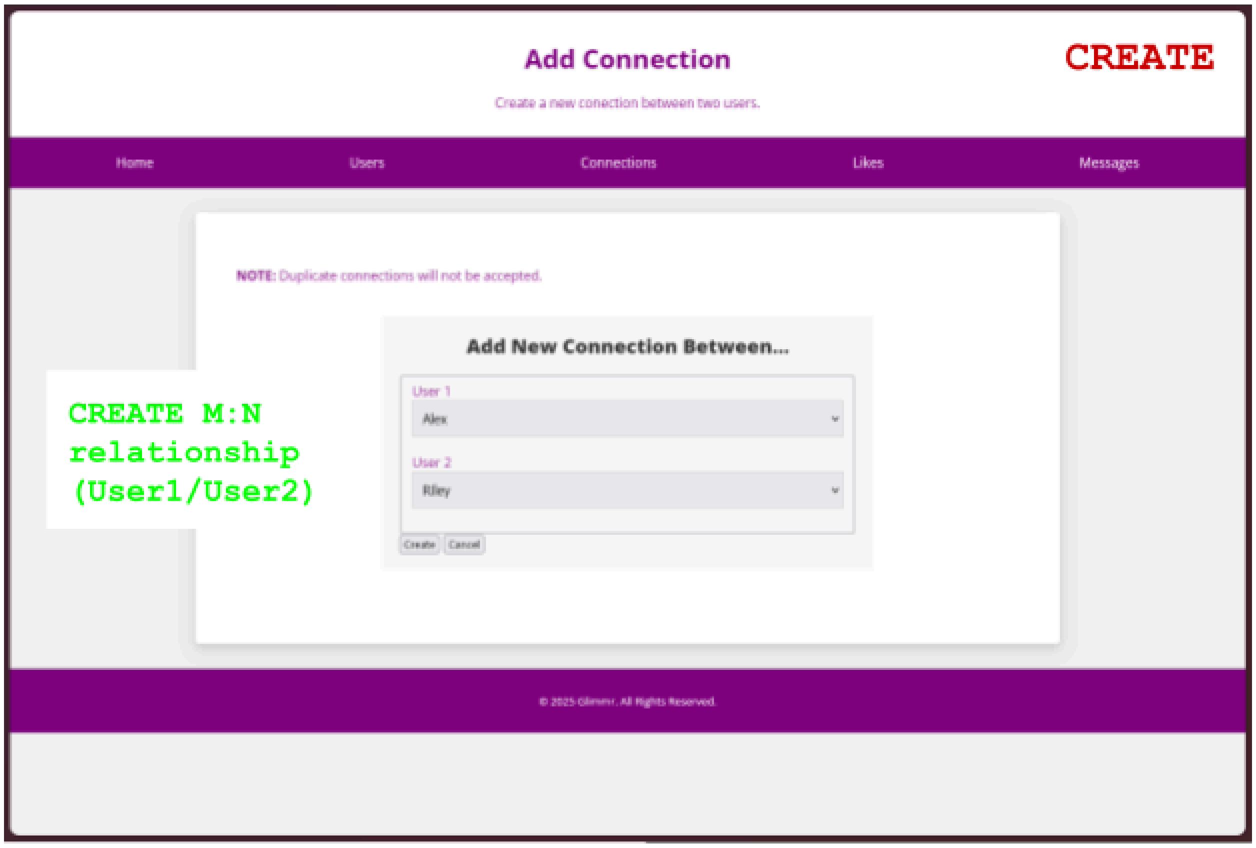
Delete User

Are you sure you want to delete the following user?

Name: **Alex**

© 2025 Glimmr. All Rights Reserved.

Figure 2.1. CREATE connections (M:N)(connections/create.html)



Add Connection

Create a new conection between two users.

NOTE: Duplicate connections will not be accepted.

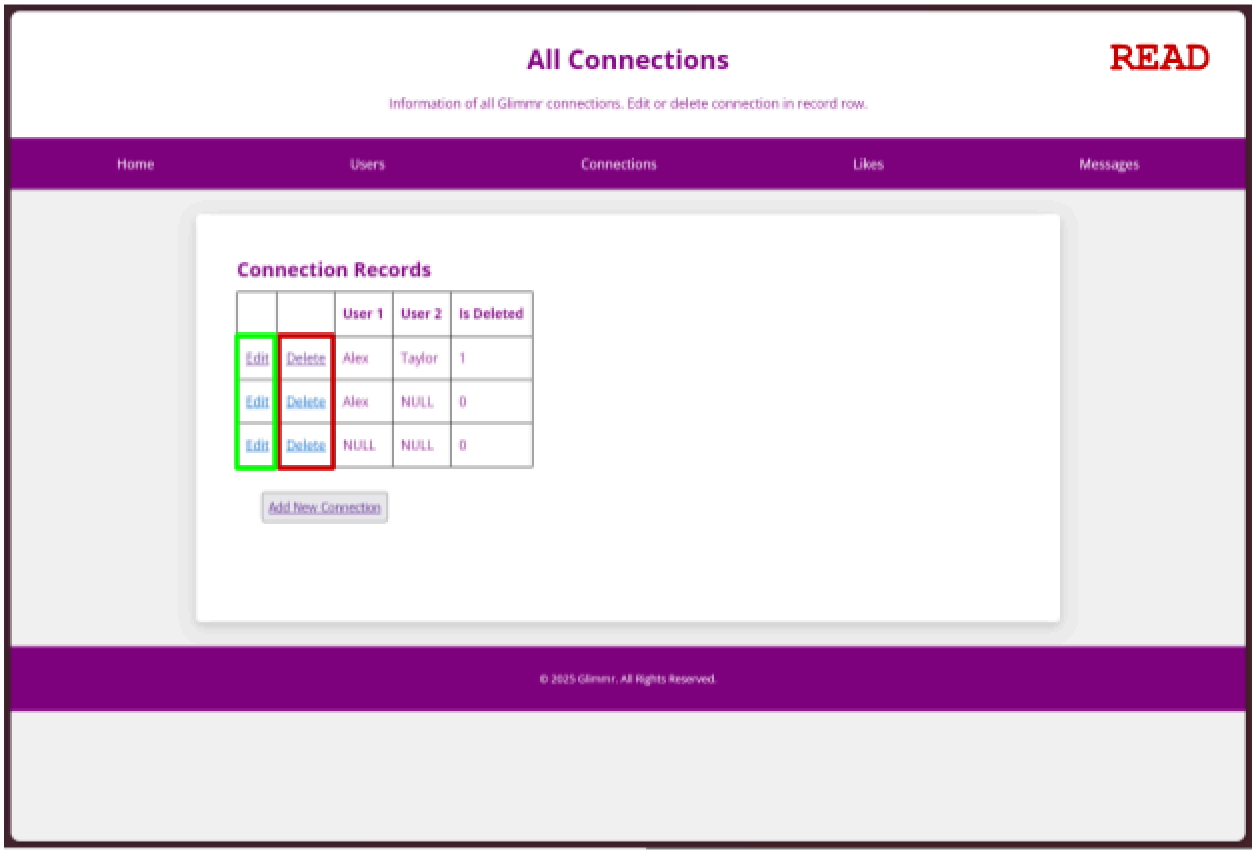
Add New Connection Between...

User 1: Alex

User 2: Riley

© 2025 Glimmr. All Rights Reserved.

Figure 2.2. READ connections (M:N) (connections/read.html)



READ

All Connections

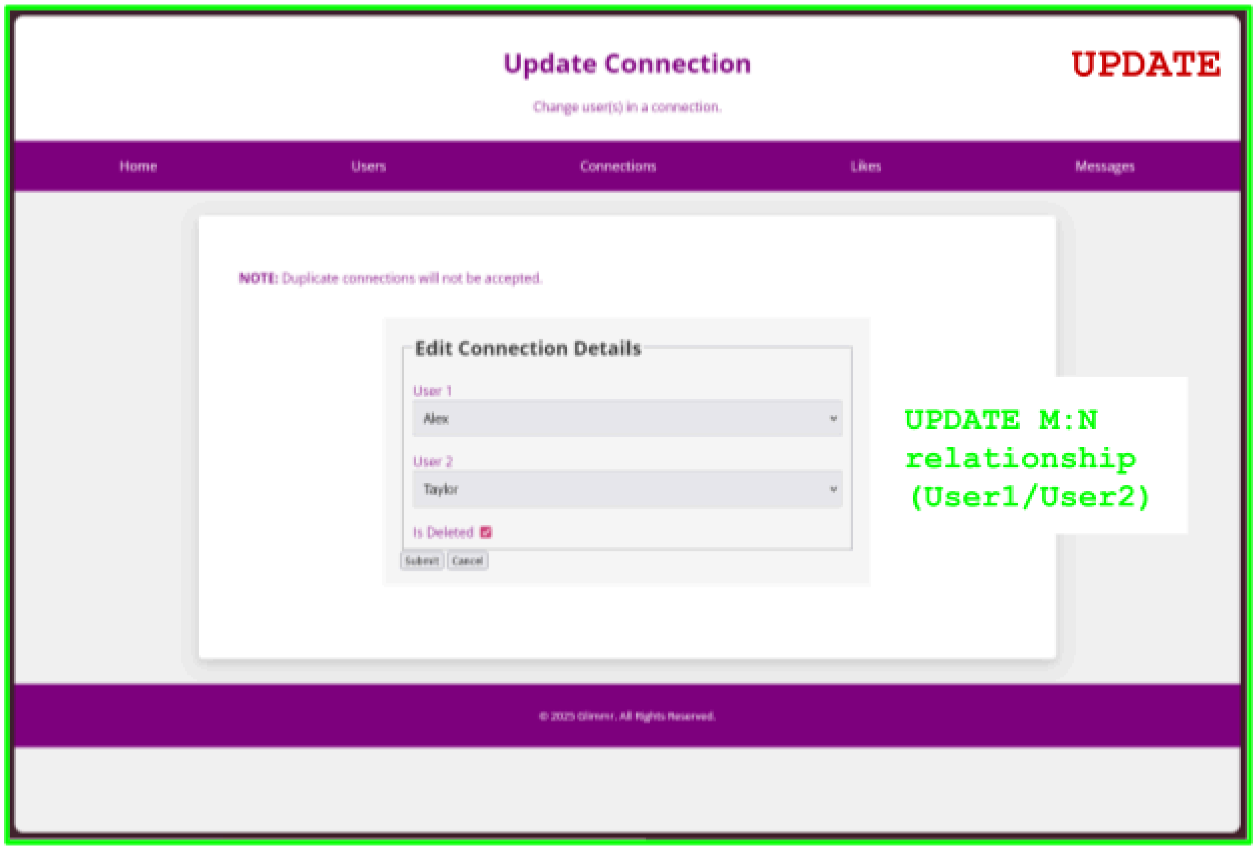
Information of all Glimmr connections. Edit or delete connection in record row.

Connection Records

		User 1	User 2	Is Deleted
Edit	Delete	Alex	Taylor	1
Edit	Delete	Alex	NULL	0
Edit	Delete	NULL	NULL	0

© 2025 Glimmr. All Rights Reserved.

Figure 2.3. UPDATE connections (M:N) (connections/update.html)



Update Connection

Change user(s) in a connection.

NOTE: Duplicate connections will not be accepted.

Edit Connection Details

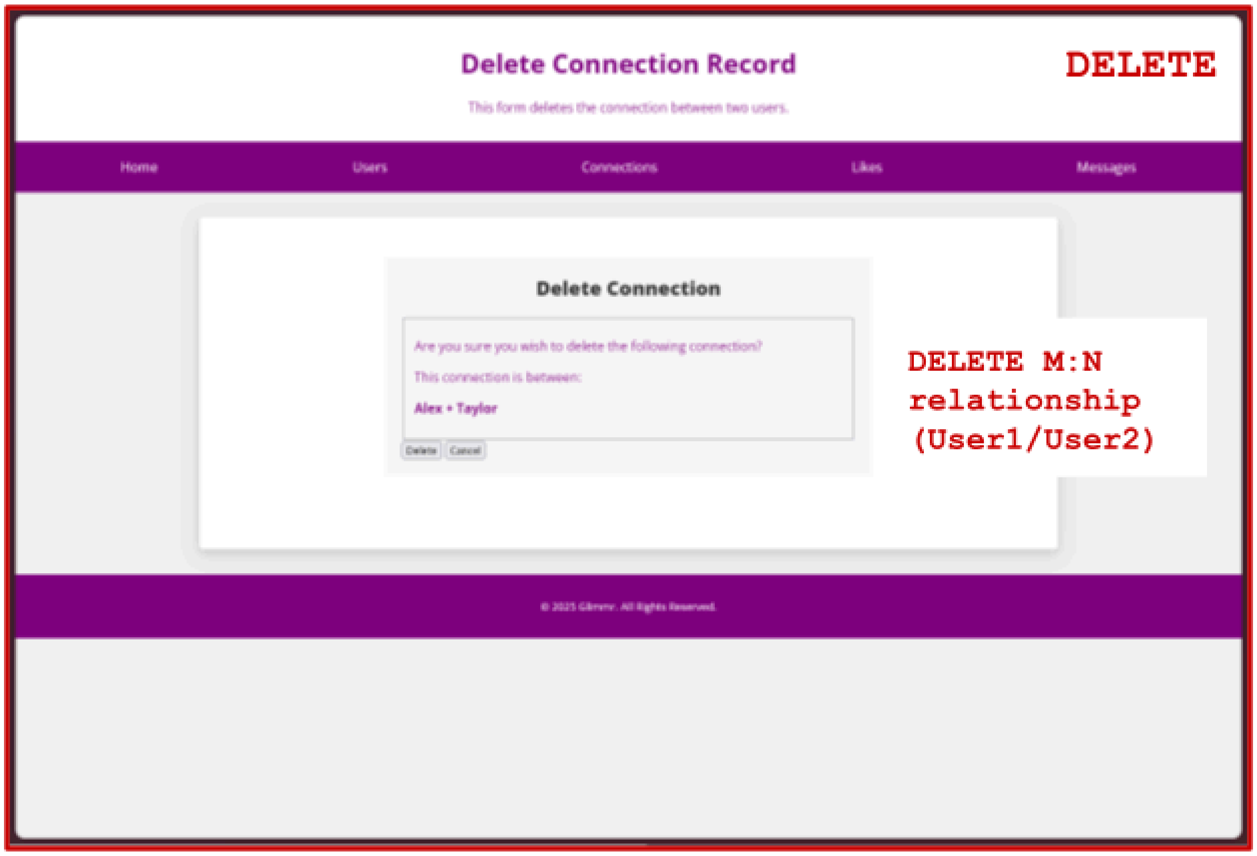
User 1: Alex

User 2: Taylor

Is Deleted ✓

© 2025 Glimmr. All Rights Reserved.

Figure 2.4. DELETE connections (M:N) (connections/delete.html)



Delete Connection Record

This form deletes the connection between two users.

Delete Connection

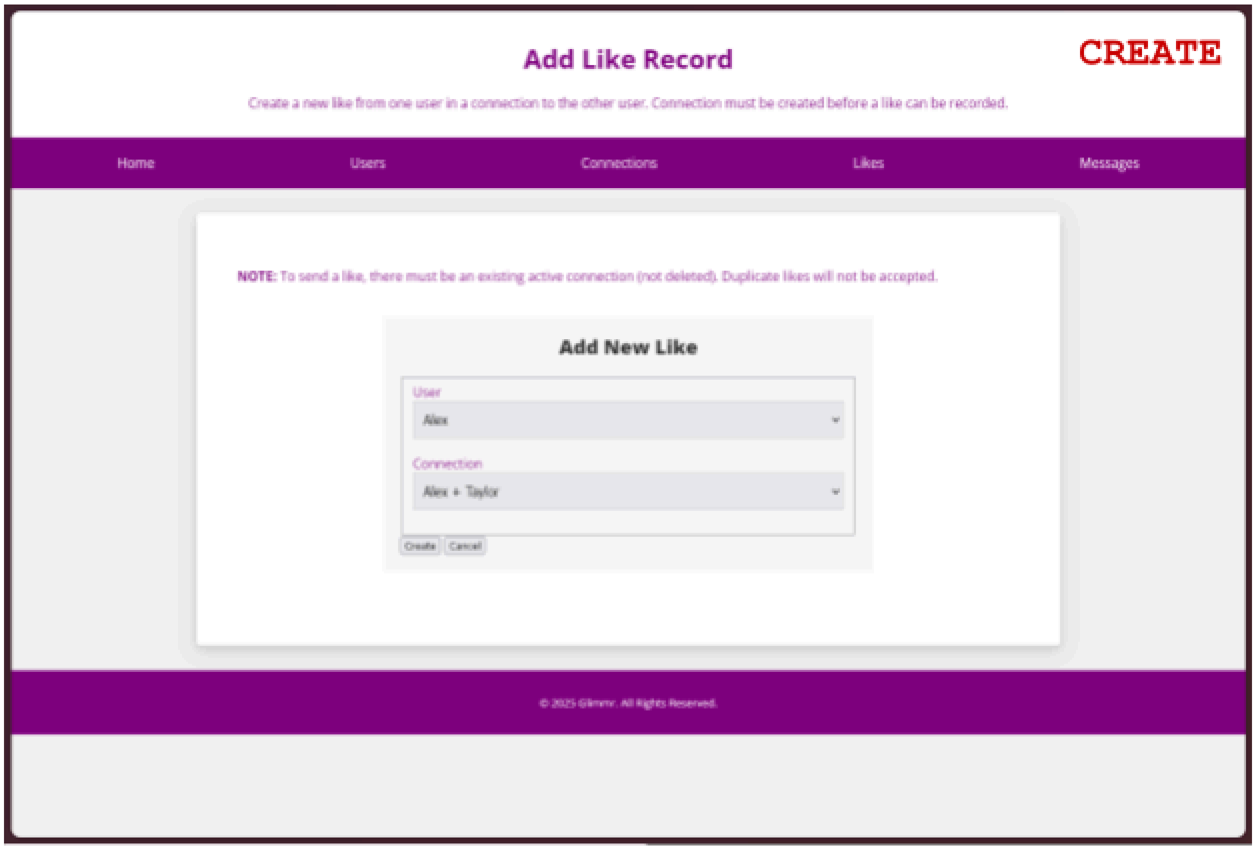
Are you sure you wish to delete the following connection?

This connection is between:

Alex • Taylor

© 2025 Glmmr. All Rights Reserved.

Figure 3.1. CREATE likes (likes/create.html)



CREATE

Add Like Record

Create a new like from one user in a connection to the other user. Connection must be created before a like can be recorded.

NOTE: To send a like, there must be an existing active connection (not deleted). Duplicate likes will not be accepted.

Add New Like

User

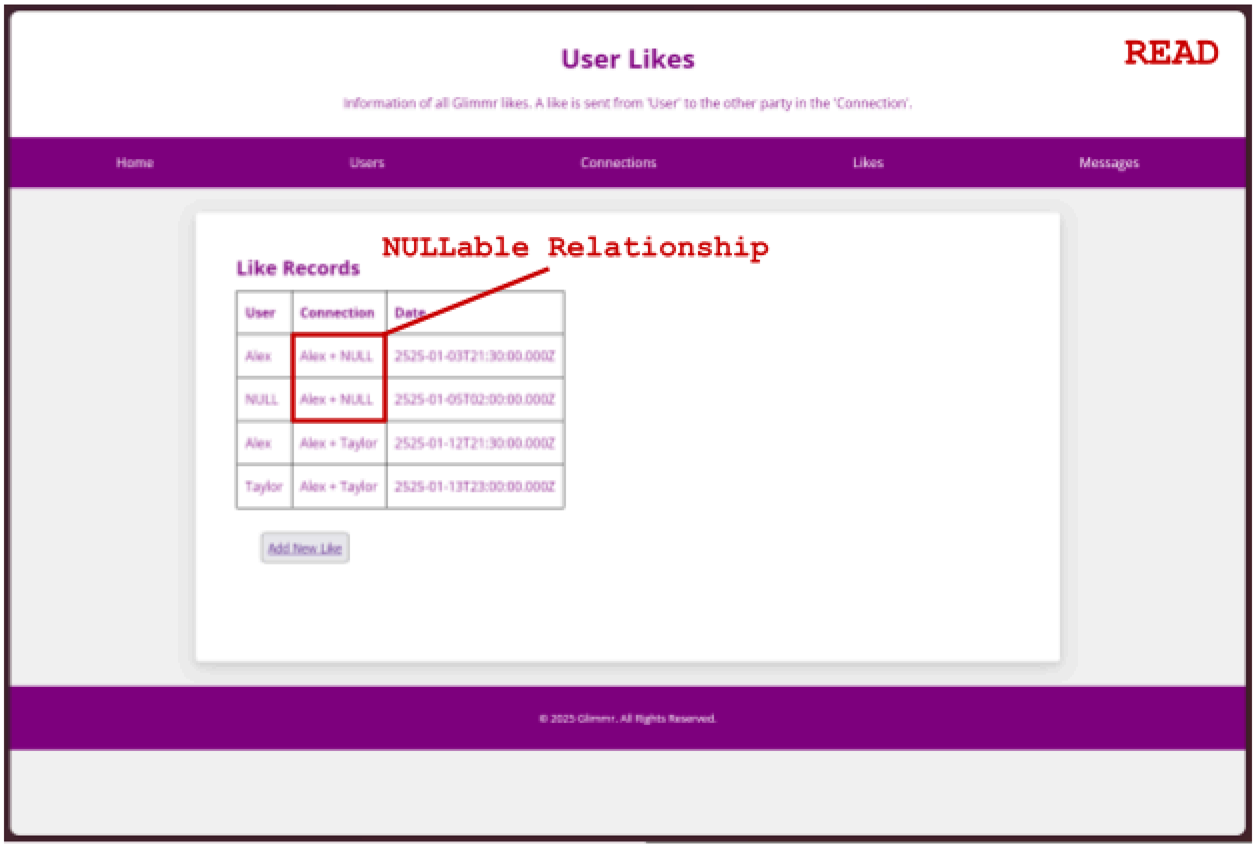
Alex

Connection

Alex + Taylor

© 2025 Glimmr. All Rights Reserved.

Figure 3.2. READ likes (likes/read.html)



User Likes

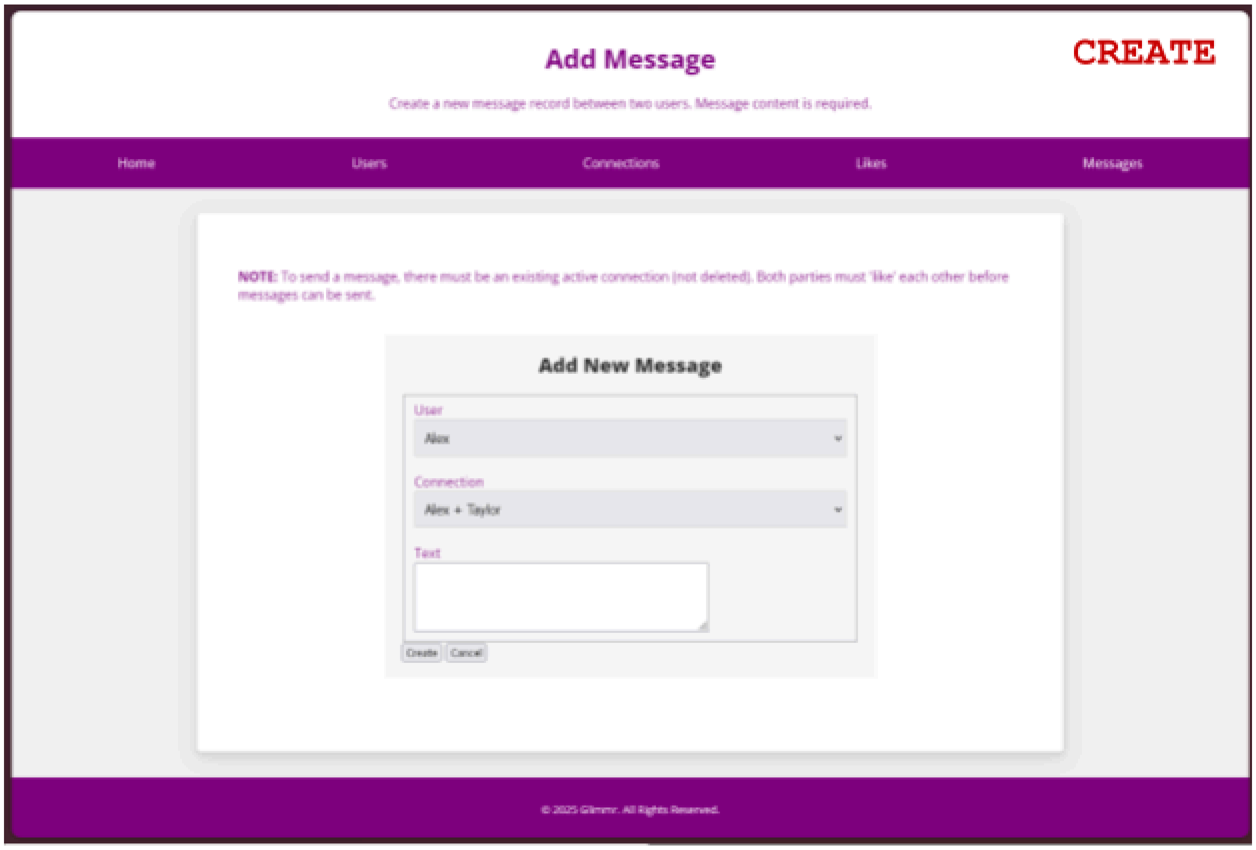
Information of all Glimmr likes. A like is sent from 'User' to the other party in the 'Connection'.

Like Records

User	Connection	Date
Alex	Alex + NULL	2525-01-03T21:30:00.000Z
NULL	Alex + NULL	2525-01-05T02:00:00.000Z
Alex	Alex + Taylor	2525-01-12T21:30:00.000Z
Taylor	Alex + Taylor	2525-01-13T23:00:00.000Z

© 2025 Glimmr. All Rights Reserved.

Figure 4.1. CREATE messages (messages/create.html)


CREATE

Add Message

Create a new message record between two users. Message content is required.

NOTE: To send a message, there must be an existing active connection (not deleted). Both parties must 'like' each other before messages can be sent.

Add New Message

User

Alex

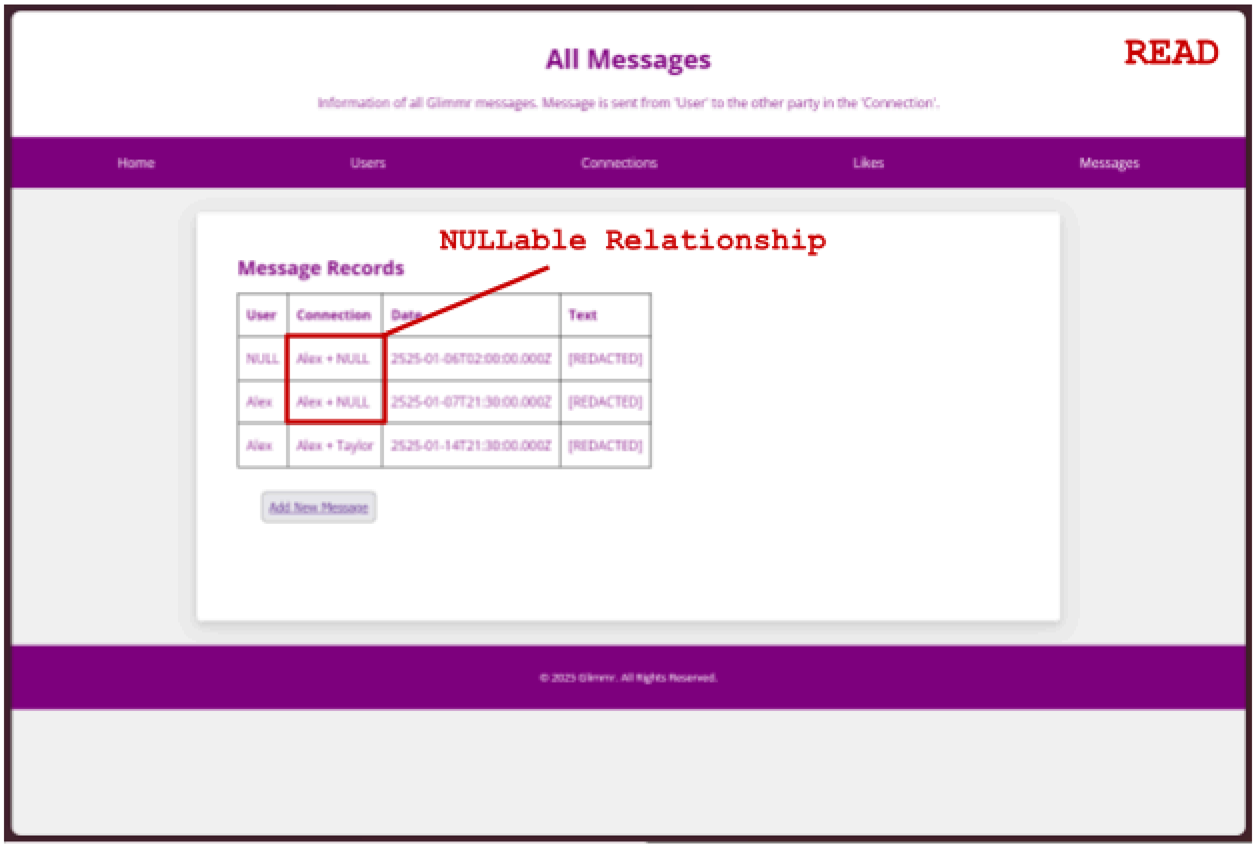
Connection

Alex + Taylor

Text

© 2025 Glimmr. All Rights Reserved.

Figure 4.2. READ messages (messages/read.html)



All Messages

Information of all Glimmr messages. Message is sent from 'User' to the other party in the 'Connection'.

Message Records

User	Connection	Date	Text
NULL	Alex + NULL	2525-01-06T02:00:00.000Z	[REDACTED]
Alex	Alex + NULL	2525-01-07T21:30:00.000Z	[REDACTED]
Alex	Alex + Taylor	2525-01-14T21:30:00.000Z	[REDACTED]

© 2025 Glimmr. All Rights Reserved.