GameNite

Social & Privacy Feature - CS4530 Spring 2026 - Group 611

Varsha Basker - Evan Yang - Heng Xu - Sunny Shen

Demo: https://spring26-project-spring26-project-group-k1ml.onrender.com/
Repo: https://github.com/neu-cs4530/spring26-project-spring26-project-group-611

Problem

GameNite originally treated all players and games as fully public, which made users feel anonymous and limited their ability to build meaningful connections. There was little control over who you could interact with or play with, which created a less personalized experience.

Our Solution

Friends & DMs

Send requests, accept/decline, message friends privately

Private Games

Hosts invite friends, control visibility, manage players

Profile Stats & Personal Info

Win/loss streaks, game history & Hometown, age, favorites with visibility controls - Public / Friends / Private

Tech Stack

Frontend	React + TypeScript
Backend	REST API
Auth	Session-based access control
CI/CD	GitHub + Render
Testing	ViTest unit + PlayWright E2E tests
Database	MongoDB

What's Next

- **Host ending the game** for all players
- **User blocking** and moderation tools
- **Group chats** for game lobbies
- **Leaderboards** with friend-aware rankings
- Refactor access control into reusable middleware

3 User Stories

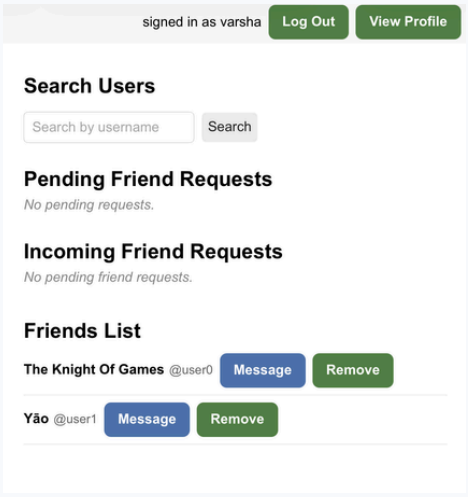
30+ Tasks Done

4 Sprints

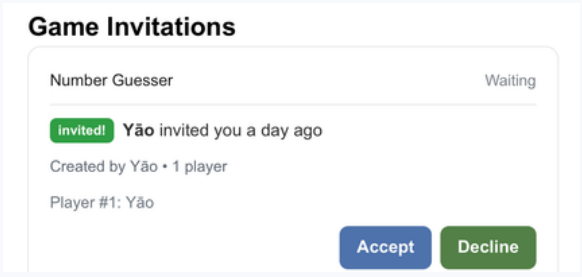
Feature Screenshots

Home - Public & Private Games

Friends - Search & Requests



Game Invites - Accept / Decline



Profile Stats

Friends System

Search for players, send/accept requests, manage your network. Friends unlock DMs, profile stats, and private game access.

Private Games

Hosts set public or private visibility at creation. Invite-only games let hosts manage players and kick participants.

Profile & Stats

Track games played, win/loss streaks, favorite game. Each stat configurable: Public, Friends-only, or Private.

Direct Messaging

Friends message each other privately outside game rooms, alongside the existing public forum.

Design Decisions

- **Friend-gate invites**

Only friends can be invited, enforced server-side

- **Role-based access**

Host vs player permissions inside game rooms

- **Visibility middleware**

canViewStats() centralizes all access control

- **Prioritized sprints**

Stories worked in parallel with friends system given highest priority