

## CIS 4910 – Team AI Usage Log Template (with Prompt History)

Team Name: GameHub

Sprint #: 3

Date: 11/4/2025

### 1. AI Tool(s) Used

List all AI tools used this sprint (e.g., ChatGPT, GitHub Copilot, DALL·E, diagramming tools).

- Tool 1: GitHub Copilot
- Tool 2: Gemini
- Tool 3: ChatGpt

### 2. Purpose of AI Use

Briefly describe what task(s) the AI supported. (Check all that apply)

- ☒ Brainstorming requirements or design alternatives
- ☐ Drafting user stories/backlog items
- ☒ Writing or refining code
- ☒ Debugging / troubleshooting errors
- ☒ Generating or suggesting test cases
- ☐ Drafting documentation (manuals, tutorials, design specs)
- ☐ Reviewing code for quality or security
- ☐ Other:

### 3. Prompt History & Refinement

Document at least two iterations of prompts and outputs. Show how the team improved or refined the request to get better results.

Iteration	Prompt	AI Response (summary) (Attached Below)	Team Action (Accepted / Edited / Rejected)
1	I have a server and im having troubles. the trouble is that i cant find a way to know when user leaves a game vs reloaded or left but came back. I tried implementing this	*1	Accepted / Edited / Rejected

grace period of 15 seconds where when you leave you trigger a 15 second timer and if you dont join back by then the server removes you from room and deletes room if empty but it doesnt work everything but that works. my idea is join room is called everytime you join a game or create one or load into one via reload and disconnect is where you can get removed after a period. I think i over engineered this server honestly i think it should be more simple, if you need to see the prisma say so but i think you can infer like inroom being what i keep track for attendance, im gonna have to paste it in parts cause the server is like 1278 lines of code of the bottom and top of the screen. Is there any way this can be removed on the game play screen only.

[disconnect] user \*2  
4eab901b-bc72-408  
e-be35-c9ee31b574  
57, socket  
TDzkLrDXfg8HFVKE  
AAAh [presence]  
user  
4eab901b-bc72-408  
e-be35-c9ee31b574  
57 disconnected.  
Starting grace timer.  
Registered socket  
li4gc1WJxveuAIURA  
AAI for user  
4eab901b-bc72-408  
e-be35-c9ee31b574  
57 Rejoined socket  
li4gc1WJxveuAIURA  
AAI to room  
a59e0d75-65f4-40c  
3-997f-95ace76678  
22 for user  
4eab901b-bc72-408  
e-be35-c9ee31b574  
57 Rejoined socket  
li4gc1WJxveuAIURA  
AAI to room  
c9bc2bef-70c6-4dcf  
-ab08-287a57da2ac  
d for user  
4eab901b-bc72-408  
e-be35-c9ee31b574  
57 [presence] user  
4eab901b-bc72-408  
e-be35-c9ee31b574  
57 permanently  
removed after  
grace. [presence]  
room  
c9bc2bef-70c6-4dcf  
-ab08-287a57da2ac  
d is now empty.  
Deleting room.

Accepted / Edited /  
Rejected

[presence] user  
fba85a80-79ae-457  
0-bc41-dca5b9e613  
80 permanently  
removed after  
grace. Error in  
disconnect cleanup:  
PrismaClientKnown  
RequestError:  
Invalid  
prisma.room.up  
date() invocation:  
An operation failed  
because it depends  
on one or more  
records that were  
required but not  
found. No record  
was found for an  
update. at  
ei.handleRequestErr  
or  
(C:\Users\benny\D  
ocuments\GitHub\  
GameHub\node\_mo  
dules\@prisma\clie  
nt\runtime\library.j  
s:121:7268) at  
ei.handleAndLogRe  
questError  
(C:\Users\benny\D  
ocuments\GitHub\  
GameHub\node\_mo  
dules\@prisma\clie  
nt\runtime\library.j  
s:121:6593) at  
ei.request  
(C:\Users\benny\D  
ocuments\GitHub\  
GameHub\node\_mo  
dules\@prisma\clie  
nt\runtime\library.j  
s:121:6300) at

```
async a
(C:\Users\benny\Documents\GitHub\GameHub\node_modules\@prisma\client\runtime\library.js:130:9551) at
async
Timeout._onTimeout
t
(file:///C:/Users/benny/Documents/GitHub/GameHub/Backend/presence.js:143:29) { code:
'P2025', meta: {
modelName:
'Room', cause: 'No
record was found
for an update.' },
clientVersion:
'6.16.2' } [presence]
user
4eab901b-bc72-408
e-be35-c9ee31b574
57 permanently
removed after
grace. Error in
disconnect cleanup:
PrismaClientKnown
RequestError:
Invalid
prisma.room.update() invocation:
An operation failed
because it depends
on one or more
records that were
required but not
found. No record
was found for an
update. at
ei.handleRequestErr
```

```
or
(C:\Users\benny\Documents\GitHub\
GameHub\node_modules\@prisma\client\runtime\library.js:121:7268) at
ei.handleAndLogRequestError
(C:\Users\benny\Documents\GitHub\
GameHub\node_modules\@prisma\client\runtime\library.js:121:6593) at
ei.request
(C:\Users\benny\Documents\GitHub\
GameHub\node_modules\@prisma\client\runtime\library.js:121:6300) at
async a
(C:\Users\benny\Documents\GitHub\
GameHub\node_modules\@prisma\client\runtime\library.js:130:9551) at
async
Timeout._onTimeout
(file:///C:/Users/benny/Documents/GitHub/GameHub/Backend/presence.js:143:29) { code:
'P2025', meta: {
  modelName:
'Room', cause: 'No
record was found
for an update.' },
  clientVersion:
```

'6.16.2' } [presence]  
user  
4eab901b-bc72-408  
e-be35-c9ee31b574  
57 permanently  
removed after  
grace. Error in  
disconnect cleanup:  
PrismaClientKnown  
RequestError:  
Invalid  
prisma.room.up  
date() invocation:  
An operation failed  
because it depends  
on one or more  
records that were  
required but not  
found. No record  
was found for an  
update. at  
ei.handleRequestErr  
or  
(C:\Users\benny\Documents\GitHub\GameHub\node\_modules\@prisma\client\runtime\library.js:121:7268) at  
ei.handleAndLogRequestError  
(C:\Users\benny\Documents\GitHub\GameHub\node\_modules\@prisma\client\runtime\library.js:121:6593) at  
ei.request  
(C:\Users\benny\Documents\GitHub\GameHub\node\_modules\@prisma\client\runtime\library.js

```
s:121:6300) at
async a
(C:\Users\benny\Documents\GitHub\GameHub\node_modules\@prisma\client\runtime\library.js:130:9551) at
async
Timeout._onTimeout
(file:///C:/Users/benny/Documents/GitHub/GameHub/Backend/presence.js:143:29) { code:
'P2025', meta: {
modelName:
'Room', cause: 'No
record was found
for an update.' },
clientVersion:
'6.16.2' }
```

3

\*3

Accepted / **Edited** /  
Rejected

now can you create  
an image to visually  
show how  
everything is split  
up and doing layers  
like this and if you  
cant give me an  
image i can  
download give me a  
prompt sao chatgpt  
can make me an  
image of my  
architecture design  
visually



## 4. Share Links to AI Sessions

Provide links to the AI chat(s) or session(s) used.

- Link 1\*:

I have a server and im having troubles. the trouble is that i cant find a way to know when user leaves a game vs reloaded or left but came back. I tried implementing this grace period of 15 seconds where when you leave you trigger a 15 second timer and if you dont join back by then the server removes you from room and deletes room if empty but it doesnt work everything but that works. my idea is join room is called everytime you join a game or create one or load into one via reload and disconnect is where you can get removed after a period. I think i over engineered this server honestly i think it should be more simple, if you need to see the prisma say so but i think you can infer like inroom being what i keep track for attendance, im gonna have to paste it in parts cause the server is like 1278 lines of code



Absolutely — you **don't need to paste 1200+ lines yet**.

Before we dive into code, let's solve the *core logic problem*, because **you don't need to overengineer this at all**.

Right now you're trying to detect:

1. User closed/reloaded the tab
2. User left and came back quickly
3. User truly left the game

Your 15-second grace timer is the correct idea, but the implementation can easily break if structured the wrong way.

Let's redesign the system cleanly — *simple, reliable, and only a few functions needed*.

### ✅ The Correct Pattern (Used by Multiplayer Games Everywhere)

You only need 3 events:

1. join room
2. heartbeat or reconnect
3. disconnect (socket disconnect)

**\*\*DO NOT use disconnect to instantly remove them.**

Always mark them as "disconnected" and start a timeout.\*\*

### 🔧 Ideal Behavior Flow

When a user joins a room



- Link 2\*:

```
permanently removed after grace.  
Error in disconnect cleanup: PrismaClientKnownRequestError:  
Invalid prisma.room.update() invocation:  
  
An operation failed because it depends on one or more records that  
were required but not found. No record was found for an update.  
  at ei.handleRequestError  
(C:\Users\benny\Documents\GitHub\GameHub\node_modules\@prisma  
a\client\runtime\library.js:121:7268)  
  at ei.handleAndLogRequestError  
(C:\Users\benny\Documents\GitHub\GameHub\node_modules\@prisma  
a\client\runtime\library.js:121:6593)  
  at ei.request  
(C:\Users\benny\Documents\GitHub\GameHub\node_modules\@prisma  
a\client\runtime\library.js:121:6300)  
  at async a  
(C:\Users\benny\Documents\GitHub\GameHub\node_modules\@prisma  
a\client\runtime\library.js:130:9551)  
  at async Timeout_onTimeout  
(file:///C:/Users/benny/Documents/GitHub/GameHub/BackEnd/presenc  
e.js:143:29) {  
  code: 'P2025',  
  meta: { modelName: 'Room', cause: 'No record was found for an  
update.' },  
  clientVersion: '6.16.2'  
}
```



Okay — this log tells us **exactly** what is happening, and the root cause is clear.

### ✓ Root Problem

You are running **disconnect cleanup multiple times** for the **same user**, and by the time the cleanup runs, the room has already been deleted.

This causes Prisma to throw:

pgsql

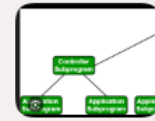
Copy code

P2025: No record was found for an update.

Because your cleanup does something like:



- Link 3\*:



now can you create an image to visually show how everything is split up and doing layers like this and if you cant give me an image i can download give me a prompt sao chatgpt can make me an image of my architecture design visually

can you do it so it represents my modular architecture and have it also more representative of the flow of things rather than just my folders

Perfect, Benito — let's shift from just listing folders to actually **mapping the modular architecture and lifecycle flow** of GAMEHUB. Here's a Markdown block that captures the way components interact, showing how data and events move through your system:

Markdown

Copy

## # 🎮 GAMEHUB Modular Architecture & Flow

GAMEHUB is designed around modular components that interact through predictable

---

### ## 🔄 High-Level Flow

```

'''
[ User ]
  |
  v
[ FrontEnd UI ] → [ Socket Layer ] → [ BackEnd Game Server ]
  |               |                   |
  |               v                   v
  |               [ AI Module ]       [ Game State Logic ]
  |               |                   |
  |               v                   v
  |               [ Auth/Profile ]     [ StreamChat ]
  |               |                   |
  |               v                   v
  |               [ Prisma ORM ]       [ Database ]
  |               |                   |
  |               v                   v
  |               [ Database ]       [ Database ]
'''

```

---

### ## 🌱 Modular Components

\*(If a tool doesn't allow link sharing, attach screenshots as evidence.)\*

## 5. Outputs Received & Team Revisions

Summarize final outputs adopted into the project and explain how they were refined by the team.

AI debugged many terminal results and runtime issues, and accepted most attempts to diagnose the issue. The AI generated 12 different visual representations of our architecture and we accepted 1. The AI

Example:

AI generated 8 user stories. We kept 5, rephrased 2, and rejected 1. Final stories were added to the backlog.

## 6. Ethical & Professional Considerations

- Did the AI output raise accuracy, bias, or copyright concerns? No
- How did the team ensure compliance with the course AI policy? Accurately recorded use of AI

## 7. Contribution to Project

Briefly describe how this AI has advanced your project this sprint.

During the sprint, AI helped refine our codebase by improving mobile compatibility, allowing faster debugging and smoother deployment across devices. We enhanced navigation from GameHub to Drop 4 and restructured game joining and menu selection. The AI generated a suggestion for server changes to improve and they were rejected as they conflicted with our game features.

## 9. Sign-Off

We confirm this log accurately reflects the team's AI usage.

Team Members (type names):

- Benito Reyes
- Mekhi Green
- Ramsey Burls
- Brandon Brown

☒ Submission Instructions: Upload this log with each sprint package in Canvas. Ensure prompt history and share links are included.

## **Ethical Implications**

**The team recognizes that building an online multiplayer platform involving user accounts, chat functionality, and real-time gameplay introduces ethical considerations regarding data privacy, responsible communication, and intellectual property. Our system features a secure user login, randomized game room codes, and a real-time chat system for players and spectators. The following sections outline the ethical risks, mitigation measures, and ongoing monitoring plans.**

### **1. Ethical Risks**

#### **a. Copyright**

- i. Established companies originally own similar games included.  
Replicating its rules, name, or visual features without permission could  
infringe on intellectual property rights.**

#### **b. Data Security and Privacy**

- i. Enabling user accounts introduces risks like unauthorized access, password leaks, or improper data storage.
    - ii. Although our design doesn't store unnecessary personal data, handling credentials can risk confidentiality if not correctly secured.
  - c. Chat Misuse and User Conduct
    - i. The live chat feature could be exploited for harassment, hate speech, or NSFW language, creating a toxic environment for players and spectators.
- 2. Mitigation Strategies
  - a. Copyright Compliance
    - i. Our implementation of games avoids copyrighted assets, using only the generic gameplay mechanics under fair-use principles.
    - ii. The final configuration will feature original designs, artwork, and naming conventions to ensure the project's independence from commercial versions.
  - b. Data Protection and Security
    - i. All user passwords are encrypted before transmission and storage.
    - ii. No sensitive personal data (such as addresses or payment details) is collected or stored.
    - iii. Access tokens and game room identifiers are randomly generated, preventing unauthorized access to sessions.
  - c. Safe Communication Channels
    - i. A basic moderation system will be implemented to minimize inappropriate messages.
- 3. Monitoring and Continuous Improvement
  - a. Routine Security Testing
    - i. We will conduct regular vulnerability checks and audits of authentication endpoints and encryption methods to prevent data leaks.
  - b. Feedback Mechanisms
    - i. Encourage player reporting and feedback on technical or ethical concerns to promote transparency and responsiveness.
  - c. Version Control documentation
    - i. Maintain detailed commit logs for updates, ensuring traceability of security-related changes.
  - d. Ethical Review Iterations
    - i. With each major update, we will re-evaluate emerging risks, especially as more games are added to adapt to policies and protections.