Roadmap

Here’s a detailed roadmap for your team to approach the ft\_transcendance project, balancing the mandatory part and optional modules while leveraging your strengths and interests:

---

1. Initial Planning (Week 1)

Goal: Understand project requirements and divide responsibilities.

1. Break Down Mandatory Features:

Frontend: Single-page application, pure vanilla JavaScript.

Backend (if chosen): Pure Ruby or overridden with a framework module.

Real-Time Multiplayer Game (Pong):

Same keyboard for local play.

Tournaments with matchmaking.

Reset aliases and handle duplicate usernames.

Docker:

Rootless mode, containerized deployment (docker-compose up --build).

2. Divide Responsibilities:

Teammate A: Frontend/game logic (mandatory Pong game, UI, user interactions).

Teammate B: Backend (optional modules like authentication, databases, matchmaking APIs).

You: Infrastructure, Docker setup, and a key backend/frontend crossover role:

Game-specific logic (matchmaking, state management).

Potential AI logic (to align with your interests).

3. Agree on a Tech Stack:

Frontend: Vanilla JavaScript (or Bootstrap if using the minor module).

Backend: Ruby or overridden with Django (major module).

Database: PostgreSQL if the database module is chosen.

4. Set Milestones:

Week 2: Basic Pong, matchmaking system prototype, container setup.

Week 3: Tournament functionality, optional modules planning.

Week 4+: Additional features, testing, and bonuses.

---

2. Mandatory Part (Weeks 2–4)

Frontend

Key Tasks:

1. Create a responsive single-page application.

2. Implement the Pong game using vanilla JavaScript:

Ball movement, paddle controls, scoring.

Ensure identical paddle speeds for fairness.

3. Build a basic UI for tournaments:

Registration form for aliases.

Display for ongoing games and matchmaking.

Testing:

Test across multiple devices and browsers (Google Chrome compatibility).

---

Backend

Optional Tasks:

1. Build a backend in Ruby for:

Storing player stats or aliases.

Matchmaking logic.

2. Overwrite constraints with Django if using the Framework module.

3. Implement secure user registration and login (hashed passwords).

Matchmaking System:

Pair players for games using a queue system.

Manage tournament brackets and announce matches.

---

Game

Key Tasks:

1. Ensure real-time multiplayer functionality.

2. Test for lag and disconnections if implementing remote players (optional module).

---

Infrastructure

Docker Setup:

Rootless Docker in /goinfre.

Single-command launch with docker-compose.

Security:

HTTPS for all communications.

Protect against SQL injection/XSS.

Store secrets in .env files.

---

3. Bonus Modules (Weeks 5–6)

Choose modules that align with your career goals as a game programmer:

Recommended Modules

1. Remote Players (Major):

Enable multiplayer games between users on separate devices.

Skills gained: WebSockets, real-time synchronization.

2. AI Opponent (Major):

Build an AI player using non-A\* algorithms.

Skills gained: Game AI, decision-making, anticipation mechanics.

3. User and Game Stats Dashboards (Minor):

Create dashboards for match history and player stats.

Skills gained: Data visualization, frontend-backend integration.

4. Game Customization Options (Minor):

Add power-ups, different maps, or other features.

Skills gained: Gameplay programming, UX design.

5. Live Chat (Major):

Create a chat system with invitations and blocking functionality.

Skills gained: Real-time communication, frontend-backend collaboration.

6. Server-Side Pong (Major):

Migrate game logic to the backend with API integration.

Skills gained: Server-side programming, API design.

---

4. Tools and Practices

1. Version Control: Use GitHub for collaboration.

Break tasks into branches (e.g., frontend, backend, docker).

2. Testing: Test early and often.

Use browser developer tools for frontend debugging.

Write unit tests for backend modules.

3. Documentation: Keep track of justifications for module choices and architectural decisions.

4. Learning Resources:

Frontend: JavaScript tutorials, DOM manipulation.

Backend: Ruby or Django documentation.

WebSockets: MDN or Socket.IO guides.

Docker: Official Docker and Compose documentation.

---

5. Stretch Goals (Weeks 7+)

If ahead of schedule, consider adding Five Major Modules for the bonus:

Multiplayer with 3+ players.

OAuth 2.0 (42 login integration).

Infrastructure monitoring (Prometheus/Grafana).

Would you like help prioritizing or scheduling specific modules?