Mock API Padel

Données

Types

data.ts

```
// TYPES
export type Player = {
  id: number;
  name: string;
  ranking: number;
};

export type Team = {
  id: number;
  name: string;
  playerIds: number[];
};

export type Tournament = {
  id: number;
  name: string;
  location: string;
  date: string; // ISO
};

export type Match = {
  id: number;
  tournamentId: number;
  teamA: number;
  teamB: number;
  score: string | null;
  winnerTeamId: number | null;
};
```

Joueurs

```
export const players: Player[] = [
  { id: 1, name: "Thomas Dupont", ranking: 15 },
  { id: 2, name: "Lucas Martin", ranking: 22 },
  { id: 3, name: "Enzo Garcia", ranking: 8 },
  { id: 4, name: "Hugo Bernard", ranking: 12 },
  { id: 5, name: "Nathan Petit", ranking: 30 },
  { id: 6, name: "Leo Robert", ranking: 18 },
  { id: 7, name: "Maxime Roux", ranking: 5 },
  { id: 8, name: "Antoine Morel", ranking: 9 },
```

```
  { id: 9, name: "Julien Fournier", ranking: 25 },
  { id: 10, name: "Paul Girard", ranking: 17 }
];
```

Equipes

```
export const teams: Team[] = [
  { id: 1, name: "Team Smash", playerIds: [1, 2] },
  { id: 2, name: "Les Lobbers", playerIds: [3, 4] },
  { id: 3, name: "Padel Masters", playerIds: [5, 6] },
  { id: 4, name: "Top Spin", playerIds: [7, 8] },
  { id: 5, name: "Les Défenseurs", playerIds: [9, 10] }
];
```

Tournois

```
export const tournaments: Tournament[] = [
  { id: 1, name: "Open de Paris", location: "Paris", date: "2026-05-10" },
  { id: 2, name: "Padel Cup Lyon", location: "Lyon", date: "2026-06-15" },
  { id: 3, name: "Marseille Padel Tour", location: "Marseille", date: "2026-07-20"
}
];
```

Matchs

```
export const matches: Match[] = [
  // Tournoi 1
  {
    id: 1,
    tournamentId: 1,
    teamA: 1,
    teamB: 2,
    score: "6-4 3-6 10-7",
    winnerTeamId: 1
  },
  {
    id: 2,
    tournamentId: 1,
    teamA: 3,
    teamB: 4,
    score: "6-2 6-3",
    winnerTeamId: 4
  },
  {
    id: 3,
    tournamentId: 1,
    teamA: 1,
```

```
    teamB: 4,
    score: "7-6 6-4",
    winnerTeamId: 1
  },

  // Tournoi 2
  {
    id: 4,
    tournamentId: 2,
    teamA: 2,
    teamB: 3,
    score: "6-2 6-2",
    winnerTeamId: 2
  },
  {
    id: 5,
    tournamentId: 2,
    teamA: 4,
    teamB: 5,
    score: "6-7 6-3 10-6",
    winnerTeamId: 4
  },
  {
    id: 6,
    tournamentId: 2,
    teamA: 2,
    teamB: 4,
    score: null,
    winnerTeamId: null
  },

  // Tournoi 3
  {
    id: 7,
    tournamentId: 3,
    teamA: 1,
    teamB: 5,
    score: null,
    winnerTeamId: null
  },
  {
    id: 8,
    tournamentId: 3,
    teamA: 3,
    teamB: 4,
    score: null,
    winnerTeamId: null
  }
];
```

Helpers

utils/helpers.ts

```
export const getTeamById = (id: number) =>
  teams.find(t => t.id === id);

export const getPlayerById = (id: number) =>
  players.find(p => p.id === id);

export const getPlayersOfTeam = (teamId: number) => {
  const team = getTeamById(teamId);
  if (!team) return [];

  return team.playerIds.map(getPlayerById);
};
```

Routes

Joueurs

GET /api/players

```
import { players } from "@/data";

export async function GET() {
  return Response.json(players);
}
```

GET /api/players/[id]

```
import { players } from "@/data";

export async function GET(
  req: Request,
  { params }: { params: { id: string } }
) {
  const id = parseInt(params.id);
  const player = players.find(p => p.id === id);
  if (!player) {
    return new Response("Player not found", { status: 404 });
  }
  return Response.json(player);
}
```

Équipes

GET /api/teams

```
import { teams } from "@/data";

export async function GET() {
  return Response.json(teams);
}
```

GET /api/teams/[id]

```
import { teams, players } from "@/data";

export async function GET(
  req: Request,
  { params }: { params: { id: string } }
) {
  const id = parseInt(params.id);
  const team = teams.find(t => t.id === id);
  if (!team) {
    return new Response("Team not found", { status: 404 });
  }
  const teamPlayers = players.filter(p =>
    team.playerIds.includes(p.id)
  );
  return Response.json({
    ...team,
    players: teamPlayers
  });
}
```

Tournois

GET /api/tournaments

```
import { tournaments } from "@/data";

export async function GET() {
  return Response.json(tournaments);
}
```

GET /api/tournaments/[id]

```
import { tournaments, matches } from "@/data";

export async function GET(
  req: Request,
  { params }: { params: { id: string } }
) {
```

```
  const id = parseInt(params.id);
  const tournament = tournaments.find(t => t.id === id);
  if (!tournament) {
    return new Response("Tournament not found", { status: 404 });
  }
  const tournamentMatches = matches.filter(
    m => m.tournamentId === id
  );
  return Response.json({
    ...tournament,
    matches: tournamentMatches
  });
}
```

Matchs

GET /api/matches

Paramètres possibles :

tournamentId teamId

```
import { matches } from "@/data";

export async function GET(req: Request) {
  const { searchParams } = new URL(req.url);
  const tournamentId = searchParams.get("tournamentId");
  const teamId = searchParams.get("teamId");
  let result = matches;
  if (tournamentId) {
    result = result.filter(
      m => m.tournamentId === parseInt(tournamentId)
    );
  }
  if (teamId) {
    const id = parseInt(teamId);
    result = result.filter(
      m => m.teamA === id || m.teamB === id
    );
  }
  return Response.json(result);
}
```

Classement

GET /api/rankings

```
import { matches, teams } from "@/data";
```

```
export async function GET() {
  const scores: Record<number, number> = {};
  teams.forEach(team => {
    scores[team.id] = 0;
  });
  matches.forEach(match => {
    if (match.winnerTeamId) {
      scores[match.winnerTeamId] += 3;
    }
  });
  const ranking = teams
    .map(team => ({
      team,
      points: scores[team.id]
    }))
    .sort((a, b) => b.points - a.points);
  return Response.json(ranking);
}
```