OVERVIEW/PLAN FOR 21 Sep to 27 Sep 2025 - Track Demons

Immediate Bugs & Refinements:

1. **Critical Bug in `FirstPersonCam.tsx`:** This file contains a copy-paste of the `CameraController.tsx` code. It needs to be replaced with actual first-person camera logic.

2. **Bug in `FollowCam.tsx`:** The camera offset is calculated in *world space*, not the car's *local space*. This means when the car turns, the camera won't swing around behind it properly. It will feel disconnected.

🚗 Team Alpha: Player & Physics (2 People)

This team's mission is to make the **player's experience feel amazing**. The car is the star of the show, so it needs to look and feel great to control.

Priority Tasks:

1. **High-Priority Bug Fixes (1 Dev):**
    - **Fix `FirstPersonCam.tsx`:** Implement logic to position the camera inside the car (e.g., at `[0, 0.3, -0.5]` relative to the car's position).
      - later on add a speedometer when in 1st person view, it doesn't need car to be ready, just find a way to get current speed of car, make a car dashboard(speedometer), rev count, and later implementing a manual gear shift when rev count is high.
    - **Fix `FollowCam.tsx`:** Update the camera logic to use the car's rotation (quaternion) to calculate its offset. This will make it swing around correctly when turning. (I can provide the code snippet for this if needed).
    - **Implement a minimap.**
2. **Car Model Upgrade (1 Dev):**
    - Replace the current `<boxGeometry>` with a 3D model.
    - Find a free, low-poly car model from sites like Sketchfab or Poly Pizza (look for `.glb` or `.gltf` formats).
    - Learn how to import and display the model using `@react-three/drei`'s `<useGLTF>` hook.
    - Needs **wheels** so it can move seamlessly and not like flipping a box
3. **Physics Tuning (Paired Task):**

- Experiment with the `mass`, `linearDamping`, `turnSpeed`, and `speed` values in `Car.tsx`. The goal is to find a balance between arcade fun and responsive handling. Does it drift? Is it snappy? Make it feel good.

🌍 Team Bravo: World & Level Design (3 People)

This team's mission is to build a **fun and interesting environment** for the player to drive in. A great track is key to replayability.

Priority Tasks:

1. **Track Layout Design (Paired Task):**
    - Start on paper or a simple design tool. Plan a race course with interesting turns, a straightaway for speed, and at least one "stunt" feature (like a jump).
    - Maybe the race track has ramp and a gap to make it to the other side of the track
2. **Basic Track Modeling (Paired Task):**
    - Build the road surface. Instead of a single `Ground` plane, create the track using multiple `useBox` or `useTrimesh` physics bodies from `@react-three/cannon`. This will provide a solid, drivable surface with boundaries.
3. **Stunt & Obstacle Implementation (1 Dev):**
    - Model and implement the key stunt features you designed. Start with a simple ramp (`useBox` with rotation).
    - Add some basic obstacles or walls to define the track edges.
4. **Environment Assets (Stretch Goal):**
    - Use helpers from `@react-three/drei` to add simple environmental details like trees, rocks, or basic building shapes to make the world feel less empty.

🎯 Team Charlie: Game Systems & UI (1 Person)

This team's mission is to give the player a **purpose and clear feedback**. They will build the rules and information that turn the sandbox into a game.

Priority Tasks:

1. **Race/Lap System Logic (1 Dev):**
    - Design a system for completing laps. The simplest way is to create a series of invisible trigger volumes (cubes with `isTrigger: true` in `useBox`) that the player must drive through in sequence.
    - Manage the state for the current lap and which checkpoint is next.
2. **In-Game UI (1 Dev):**
    - Expand the UI beyond the camera button.
    - Create and display a **lap counter** (`Lap: 1/3`).
    - Create and display a **lap timer**.
    - Add a "GO!" message at the start and a "FINISH!" message at the end.
3. **AI Research (Paired Task - Low Priority):**
    - This is a research-only task for now. Spend a few hours looking into how AI agents work in Three.js and Cannon.js.
    - **Key questions to answer:** How can we make an AI follow a series of waypoints? What libraries exist for pathfinding?
    - **Do not write code for this yet.** The goal is to prepare for a future sprint. This is a complex task that should only be started after the core game loop is fun and stable.