I cannot directly generate a downloadable PDF file, but I have formatted the full project plan below as a clean, structured document.
You can easily save this as a PDF by clicking the **"Share & export"** button at the bottom of this response and selecting **"Export to Docs"**. Once it's in Google Docs, just go to **File > Download > PDF Document (.pdf)**.

Project Plan: Formalizing Competitive Game Meta in Lean 4

Researcher: Arthur Freitas Ramos **Objective:** To bridge formal verification and game theory by building a verified model of complex strategy game mechanics (e.g., Pokémon TCG) to prove "Meta-Safety" invariants.

Phase 1: The Type-Safe Foundation 🏗️

Goal: Build the "Ontology" of the game as a rigid mathematical system.

- **Project Initialization:** Set up a Lean 4 package with lake.
- **Inductive Primitives:** Define the fundamental types for the game world:
  - inductive EnergyType: Modeling Fire, Water, Psychic, etc.
  - structure Card: Containing HP, EnergyType, and a list of Attack effects.
  - structure GameState: A snapshot of the board, hands, decks, and prize pools.
- **The Data Pipeline:** Use Python/Swift to scrape card JSON from the Pokémon TCG API and auto-generate Lean def statements. This ensures the model is grounded in real-world "Standard" card rotations.

Phase 2: Operational Semantics & The Damage Engine 🚀

Goal: Define the rules of movement and interaction (the game "physics").

- **The Transition Function:** Implement applyAction : GameState → Action → Option GameState.
- **Damage Logic:** Formalize the damage calculation pipeline:
  - Base Damage \to Weakness (2\times) \to Resistance (-30) \to Tool/Buff modifiers.
- **Validation:** Use Lean's #eval and #check to simulate key "meta" interactions and verify they yield results identical to the official rulebook.

Phase 3: Formal Verification & Meta-Safety 🛡️

Goal: Move from "it runs" to "it is proven."

- **The Turn-One (T1) Theorem:** Formally state and prove that no valid sequence of actions from a standard starting deck can result in a WinState for the starting player on Turn 1.
  -
- **Invariant Proofs:** Prove that the "Conservation of Cards" holds—the sum of cards in all zones (Deck, Hand, Board, Discard) always equals 60.

Phase 4: Scaling, Tooling & Integration 🛠️

Goal: Transform the proof into a usable software artifact.

- **Verified Solver:** Write an optimization algorithm in Lean 4 that searches the state space for the "Maximum Damage Line." Prove that the solver is **sound** (it only suggests legal moves) and **complete** (it never misses the mathematical maximum).
- **Cross-Platform FFI:** Use Lean 4's C-export or a CLI wrapper to connect your verification engine to a Swift (iOS) or Python front-end, allowing you to "input" a hand and get a "proven" best move.

Phase 5: Publication & Academic Impact 🎓

Goal: Share the findings with the Formal Methods community.

- **Target Venues:** * **ITP (Interactive Theorem Proving):** Focus on the Lean 4 implementation.
    - **FormaliSE (Rio de Janeiro, 2026):** Present the practical applications of formal methods in commercial game design.
- **Core Thesis:** "A Domain-Specific Language (DSL) in Lean 4 for the Automated Verification of Emergent Behavior in Trading Card Games."

I'm always open to corrections if I get something wrong or if you'd like to pivot the focus. The best way to do that is to just tell me what I got wrong or what else I need to know!

Would you like me to draft the specific Lean 4 boilerplate code for the GameState structure to get Phase 1 started today? 📜✨