# 1: Game Concept and Design

**Define Core Mechanics:**

* **Resource Management:** Players accumulate mana/energy over time, which can be used to deploy various types of troops.
* **Troop Deployment:** Players can choose when to deploy troops along a path to counter advancing enemies. Troops always originate from the player’s base and advance until destroyed or until they reach the enemy’s base.
* **Enemy Waves:** Enemies come in waves with varying strength and numbers, challenging the player's defense strategy.
* **Enemy collision:** Player and enemy troops will stop to fight each other when in contact until they get destroyed.

**Game Elements:**

* Different types of troops, each with unique costs and abilities.
* An enemy AI that adjusts its strategy based on player actions.
* Levels of increasing difficulty with different maps.

# 2: Study Similar Games

**Research:**

* Look into games like "Zombie Trailer Park" or even mobile hits like "Clash Royale" which employ similar mechanics of deploying troops based on resource management.
* Analyze how these games balance resource accumulation, troop cost, and enemy strength to keep the game challenging and engaging.

# 3: Tools and Technologies

**p5.js for Game Development:**

* Use p5.js for creating the game environment, handling animations, and managing user interactions.
* Consider integrating other libraries if needed for more complex features like pathfinding or more advanced physics.

**Learning Resources:**

* p5.js tutorials specifically aimed at game development.
* Online courses or tutorials on JavaScript game programming.

# 4: Prototype Basic Features

**Start Small:**

* Create a simple prototype that includes a basic game loop where:
  + Players can accumulate a fixed amount of mana over time.
  + Deploy simple troop units by clicking or tapping on specific screen areas.
  + Basic enemy units move towards a goal, and troops can intercept them.

**Iterate:**

* Test and refine the mechanics with simple shapes and interactions.
* Gradually introduce complexities like different types of troops and enemies.

# 5: Develop the Game

**Incremental Development:**

* Develop the game in stages, starting with core gameplay and then adding features like different troop types, level progression, and enemy AI.
* Regularly test game balance to make it neither too easy nor too hard.

**Interface and Graphics:**

* Design a user-friendly interface that allows players to easily select and deploy troops.
* Enhance visual appeal as gameplay mechanics become solid.**6: Testing and Feedback**

**Playtest:**

* Involve others in playtesting your game to get feedback on its mechanics, difficulty, and enjoyment.
* Use this feedback to make adjustments.

# 7: Finalization and Launch

**Polish:**

* Refine graphics and interfaces based on feedback.
* Ensure the game is bug-free and runs smoothly across targeted platforms.

**Documentation:**

* Prepare comprehensive documentation on how the game works, including guides on how to play and troubleshoot common issues.

# 8: Presentation

**Showcase:**

* Prepare to present your game to peers or in a project showcase.
* Highlight the key features and what you learned during the development process.

# Minimum Viable Product (MVP) Requirements

1. **Core Game Loop:**
   * **Mana Accumulation:** Implement a basic system where players accumulate mana over time.
   * **Troop Deployment:** Allow players to deploy troops using accumulated mana. At this stage, even one or two types of troops would be sufficient.
   * **Enemy Waves:** Design a simple mechanism for spawning enemy waves at regular intervals.
2. **Basic Graphics and User Interface:**
   * **Simple Graphics:** Use basic shapes or simple sprites to represent troops and enemies. p5.js can handle these graphics effectively.
   * **User Interface:** Develop a simple UI that displays available mana, controls for deploying troops, and perhaps a basic level progression or wave counter.
3. **Game Mechanics:**
   * **Troop and Enemy Interaction:** Implement basic interactions where troops and enemies meet and fight until one is destroyed.
   * **Win/Lose Conditions:** Define clear conditions for winning (e.g., surviving a certain number of waves) and losing (e.g., enemies breach the base).
4. **One Level Design:**
   * Start with a single level to test the core gameplay mechanics. The level design should include a path from an enemy entry point to the player’s base.

**Additional Requirements to Get Started**

1. **Development Environment Setup:**
   * Ensure you have a suitable code editor (like Visual Studio Code) and a working setup of p5.js (either through a CDN link or a local library copy).
2. **Project Structure:**
   * Organize your project files with separate directories for scripts, assets (like images or sounds), and other resources.
3. **Version Control:**
   * Set up a Git repository to manage your project versions and back up your work. GitHub is a great platform for this, and it also allows you to host your game using GitHub Pages.
4. **Basic Game Physics:**
   * For interactions between troops and enemies, implement straightforward physics—mainly collision detection and response, which p5.js can handle through basic distance checks.
5. **Debugging Tools:**
   * Prepare to use debugging tools available in your browser or through your code editor to troubleshoot and refine game mechanics.

**Steps to Start Development**

1. **Initial Setup:**
   * Create your project structure and initialize your p5.js sketch with a basic canvas.
   * Set up the main game loop (**setup** and **draw** functions in p5.js).
2. **Implement Core Features:**
   * Start coding the mana accumulation system.
   * Add functionality to deploy troops and spawn enemy waves.
   * Code the basic interactions between troops and enemies.
3. **Create Basic UI:**
   * Add text or buttons to allow the player to see and use their mana to deploy troops.
4. **Testing:**
   * Begin with simple playtests to ensure the basic mechanics are functioning as expected. Adjust mechanics based on these tests.
5. **Iterative Development:**
   * Continuously expand and refine the game features based on testing feedback and your project goals.