Dark Core Hiring Case: “Cluster Crusher”

**Dark Core Game Developer Hiring Case: “Cluster Crusher”  
  
Overview**Build a web-based puzzle game in Pixi.js v7 + TypeScript. Players clear matching gems under a move limit.  
  
**Duration:** 3 business days  
**Delivery:** Git repo link + README

### **1. Core Requirements**

**Board & Gems**

* 5×5 grid, initially filled with 3 gem colors

**Preview Queue**

* Holds 3 upcoming gems
* Player drags a preview gem onto a board cell to swap

**Cluster Detection & Clearing**

* When three or more same-color gems touch orthogonally, they clear
* If a swap does **not** create any cluster, the move is canceled and the gem immediately snaps back to the preview queue
* No cascades: newly spawned gems do not trigger further clears

**Refill & Animation**

* After a clear, refill empty cells at once with random gems, using a simple scale/pop animation

**Moves, Scoring & End Conditions**

* 25 moves per game
* Score for a cluster of *N* gems = (N – 2)² × 10 (e.g. 5 gems → 90 points)
* Win if score ≥ 500; otherwise, lose after 25 moves
* Show result (“You Win!” or “Game Over”) in a modal with a restart button — no screen transition

**UI & Asset Loading**

* Always display remaining moves and current score in the UI
* Game must run entirely on a single screen — no scene/screen transitions
* Load assets via PIXI.Loader (or fetch); show a “Loading…” indicator until complete
* Provided assets include PNGs for button, modal, info box, grid cells, and four gem colors. You are expected to use these directly — do not redesign or replace them.

**Code Quality & Libraries**

* Use **Pixi.js v7+** & **TypeScript**
* State management with a custom **Zustand** store
* No heavy helper libraries
* Organize code into clear modules (Board, Preview, GameLogic) with no unused code
* README only needs basic install & run instructions

### **2. Optional Features**

* **@pixi/react Integration**: Use React wrappers for Pixi rendering
* **Preview Dimming**: During any animation, reduce the preview queue’s opacity to 50%
* **4-Color Mode**: Begin with 4 colors; reshuffle board if no clusters exist
* **Invalid-Placement Return**: Add a small “snap-back” animation when a swap is canceled
* **Gravity Refills**: Let gems above fall down before spawning new ones at the top
* **Responsive & Touch**: Support window resizing and touch input
* **Match Highlight**: After two canceled moves, highlight one available cluster (max 3 hints)
* **Local Leaderboard**: Save and display top scores

### **3. 3-Day Timeline**

**Day 1**

* Project setup & install dependencies (Pixi.js, TypeScript, Zustand)
* Asset loading flow & “Loading…”
* Implement Board & Preview UI
* Show move count & score

**Day 2**

* Drag-and-drop swaps
* Cluster detection, gem clearing & refill spawn
* Scale/pop animation for refilled gems

**Day 3**

* Win & lose flow, “You Win!” / “Game Over” modals, restart button
* Finalize README (install/run steps)
* **If time remains**, pick one or more optional features from section 2

### **4. Contact**

Chris Veyne — chris@darkcore.net