Teste Técnico – Game Developer

Tema: Megabonk-like Mitológico 3D (estilo Vampire Survivors)

Prazo: 7 dias corridos

■ Objetivo

Criar um protótipo jogável 3D inspirado em Vampire Survivors, com tema mitológico. O jogador deve enfrentar hordas crescentes de inimigos em uma arena tridimensional, coletar XP, evoluir e escolher itens com efeitos distintos. O foco é em gameplay, performance e arquitetura de código, não em gráficos ou arte.

■■ Requisitos de Gameplay

1. Movimento do jogador: controle via WASD ou analógico, com aceleração/desaceleração simples. 2. Ataque automático: pelo menos uma arma base que ataque sozinha em intervalos regulares. 3. Inimigos em hordas: spawner com curva de dificuldade (taxa, vida, dano, velocidade). 4. XP e Level-up: inimigos dropam gemas de XP; ao subir de nível, exibir painel de escolha com 3 opções aleatórias (armas/itens/upgrades). 5. Itens/Armas: mínimo de 5 efeitos distintos implementados. 6. Condições de jogo: vitória após X minutos/ondas; derrota ao zerar HP. 7. Ambiente 3D com movimentação livre, mesmo que usando apenas meshes primitivas (cubos, esferas, planos).

■ Itens e Efeitos (mínimo 5)

Exemplos possíveis (você pode criar variações): - Raio de Zeus – projétil em cadeia que salta entre inimigos. - Asas de Hermes – aumenta velocidade e raio de coleta de XP. - Égide de Atena – escudo orbital que causa dano por toque. - Tridente de Poseidon – lança onda perfurante com knockback. - Toque de Midas – chance de drop extra de ouro. - Elmo de Hades – invisibilidade breve que reduz aggro.

■ Arquitetura de Código

- Use ScriptableObjects ou JSON para definir armas/itens (parâmetros: dano, cooldown, alcance, evolução). - Componentes modulares e reutilizáveis (ex.: Health, Damageable, Mover, WeaponController, ItemEffect). - Evolução e upgrades devem ser escaláveis (níveis 1→5, cumulativos ou com mudanças qualitativas). - Exponha parâmetros importantes no Inspector para facilitar balanceamento.

■■ Interface e Feedback

- HUD: barra de HP, nível, XP, tempo de partida e slots de itens/armas. - Painel de escolha (level-up): 3 opções com nome e descrição. - Som e partículas opcionais (foco é o sistema funcional).

■ Performance e Técnica

- Uso de Object Pooling para projéteis e inimigos. - Sem alocações por frame (evite Garbage Collection excessivo). - Manter performance de 60 FPS com 200+ inimigos ativos em cena simples.

■ Entregáveis

1. Projeto Unity 3D completo (2022.3 LTS+ recomendado). 2. Build jogável (Windows ou WebGL). 3. README.md com: - Versão da Unity usada; - Controles e instruções; - Descrição dos itens e suas evoluções; - Notas técnicas e explicações de arquitetura.

■ Critérios de Avaliação

- Critério 1 – Qualidade de código: 35% - Critério 2 – Gameplay e feedback: 25% - Critério 3 – Sistema de itens/armas: 20% - Critério 4 – Performance e estabilidade: 10% - Critério 5 – Entrega e documentação: 10%

■ Observações

- Gráficos e arte NÃO são avaliados. - Pode usar qualquer asset gratuito, ou apenas primitivas 3D. - Scripts prontos pesados (frameworks) não recomendados. - Uso de IA é permitido se houver compreensão e explicação no README.

■ Bônus (Opcional)

- Implementar modo Multiplayer com Photon Fusion 2 (2 jogadores cooperativos ou competitivos). - Sincronizar movimento, XP e ataques básicos. - Lobby simples com join/host local ou via room ID. - Sincronizar spawn de inimigos pelo host.