Plano de Negócio + Roadmap Técnico (MVP → Lançamento)\

Projeto: MMO 2D top-down (estilo RPG) com transformação em criaturas, scan para desbloqueio, economia baseada em risco/entrada, casas, guildas e combate em tempo real (server-authoritative).
Data: 2026-02-16

> *Nota importante (mercado e viabilidade): se o jogo estiver usando IP/nomes/arte de terceiros (ex.: "Pokémon"), isso inviabiliza lançamento comercial sem licenciamento. Para ter chance real de sucesso, o caminho é **rebranding + criaturas/skills/ícones 100% originais**, mantendo o conceito (transformação/scan) e não o IP.*

---

1) Resumo executivo

1.1 Tese de produto

- **Identidade única**: jogador é um **avatar persistente** e assume **formas temporárias** dentro de mundos/instâncias.
- **Progressão via Scan**: derrotar → tentar scan → desbloquear forma/uso futuro/mercado. Isso cria colecionismo e progressão sem "level infinito".
- **Economia com risco**: entrada paga (com janela de entrada gratuita diária) + custo de morte controlam spam/bots e dão valor ao tempo.

1.2 O que já existe hoje (base real do projeto)

- Client Unity (2D): UI, skills/cooldowns, scanner FX, transformação, listas, integração WebSocket.
- Server Node.js (WebSocket): mundo/tick, entidades, combate/skills autoritativos, XP/gold, stats (damage/defense/crit/dodge/cooldown/scan/lucky), guilds, houses, wild monsters, interest por área.
- Website PHP: páginas de comunidade/ranking/login/register/news etc.

1.3 Objetivo do plano (foco em desenvolvimento)

- Transformar o projeto em um **produto lançável** com:
  - Funções core consistentes
  - Pipeline de conteúdo escalável
  - Estabilidade/anti-exploit
  - Retenção (social/economia/progressão)
  - Operação (observabilidade, deploy, patching)

---

2) Proposta de valor (mercado)

2.1 Público-alvo (hipótese)

- Jogadores que gostam de: Tibia-like/2D MMO, progressão de builds, coletar formas/skins, economia/mercado, guildas.

2.2 Diferenciais que podem vender

- **Transformação como "classe"**: ao invés de escolher classe fixa, o jogador monta um "arsenal de formas" via scan.
- **Server-authoritative + skillshot**: combate com skill ceiling (posicionamento/cooldowns) com justiça competitiva.
- **Economia ligada a risco**: entrar custa (às vezes), morrer custa (às vezes). Isso cria tensão e valor.

2.3 Riscos de produto (e mitigação)

- **Frustração de morte/entrada** → proteção no early game (ex.: primeiras X mortes sem custo; ou "seguro" diário).
- **Metagame dominado por poucas formas** → contramedidas por terreno, counters e rotação de mapas/eventos.
- **Conteúdo insuficiente** → pipeline de conteúdo e ferramentas internas (ver seção 5).

---

3) Modelo de negócio (sem prometer milagre)

3.1 Monetização recomendada (evitar pay-to-win)

- **Cosméticos**: skins, efeitos, pets, emotes, titles.
- **Convenience limitada** (sem vantagem de combate): slots de loadout, armazenamento extra, fast travel limitado.
- **Season pass** (opcional): focado em cosméticos + QoL.

> *Evite vender poder direto (stats/dano), porque MMO competitivo com P2W perde confiança rápido.*

3.2 Retenção e "live ops"

- Rotação semanal de eventos/mapas.
- Missões diárias simples.
- Bosses e "dungeons" curtas (5–10 min) com recompensa previsível.

---

4) Definição do CORE do game (o que precisa ficar perfeito)

O core (versão lançável) pode ser definido como:

1. **Entrar no mundo** (instância) rápido e claro
2. **Encontrar combate** em 30–60s
3. **Matar** → **loot** → **scan** (tensão e recompensa)
4. **Progredir** (formas, build, economia)
5. **Social** (guilda, trade, houses)

Se qualquer um desses 5 estiver fraco, o jogo "não segura".

---

5) Roadmap técnico por camadas (Unity / Server / Website)

5.1 Pilar A — Qualidade e estabilidade (antes de "mais features")

Server (Node):

- Rate limiting por ação (movimento/skill/autoattack/chat) e validações anti-spam.
- Versionamento de protocolo (campo `protocolVersion` no handshake) para evitar client antigo quebrando.
- Logs estruturados por evento + amostragem (não logar tudo sempre).
- Persistência segura: transações e locks onde há gold/inventário para evitar dupe.

Client (Unity):

- Reconexão: detectar queda, tentar reconectar e restaurar estado.
- Tratamento de latência: interpolação leve e fallback visual.

- Pooling para FX (projectile/damage popup) para reduzir GC spikes.

Website:

- Painel básico de status (uptime, players online, última versão do server).

5.2 Pilar B — Combate "produto" (loop viciante)

Server:

- Unificar dano: mesma pipeline para skill e autoattack (crit, defense, dodge, resist, buffs).
- Agentes de combate: aggro/target rules e "range" consistente.
- Colisão/hit: padronizar hitboxes (tile-based) e garantir determinismo.

Client:

- Telegrafia: animações/indicadores de área de skillshot.
- Feedback: SFX/VFX e números de dano consistentes.

5.3 Pilar C — Progressão e economia (onde o MMO vive)

Server:

- Loot tables por criatura/bioma (data-driven).
- Crafting simples (2–3 receitas úteis) e sinks de gold (taxas, craft, repair, fast travel).
- Market/Auction minimalista (listar item + taxa + histórico de preço).

Client:

- UI do market (listar/comprar/vender) com filtros mínimos.

Website:

- Página de ranking/economia (top traders, itens em alta).

5.4 Pilar D — Conteúdo escalável (pipeline)

Sem pipeline, você fica preso "codando conteúdo", o que não escala.

O que transformar em data (não código):

- Criaturas: HP/dano/skills/velocidade/loot
- Skills: power/cooldown/area/efeitos
- Mapas: spawn zones, densidade, eventos
- Quests: objetivos e rewards

Ferramentas internas (mínimo viável):

- Editor simples (web admin) para:
  - criar/editar criatura/loot/quest
  - publicar versão de conteúdo (content version)

5.5 Pilar E — Social (retenção real)

Guildas:

- Objetivos de guilda (matar X, controlar spot, craftar Y).
- Benefícios coletivos não-P2W (ex.: storage, banner, cosmético).

Houses:

- Utilidade: storage, craft station, teleport limitado.

---

6) MVP recomendado (vertical slice) — 4 a 6 semanas

O MVP não é "tudo funcionando", é **1 experiência completa**.

MVP (1 instância)

- 1 mapa pequeno com 2 biomas
- 8–12 criaturas
- 4–6 skills
- 1 boss
- Loot básico + 1 crafting
- Scan com progressão (meta clara: desbloquear 3 formas)
- 1 guild feature útil (ex.: missão semanal)

Critério de sucesso do MVP:

- Jogador novo entende em 5 min.
- Jogador fica 30–60 min sem "o que fazer".
- Sem exploits óbvios de gold/dupe.

---

7) Backlog de desenvolvimento (priorizado) — o que fazer agora

P0 (necessário para lançar alpha)

- Observabilidade (logs/metrics), rate limit, validações server
- Pipeline de conteúdo data-driven (ao menos criaturas/loot/skills)
- Tutorial/UX inicial
- Market minimalista ou trade seguro
- Anti-dupe em inventário/gold

P1 (para retenção)

- Quests diárias/semanais
- Eventos rotativos
- Dungeons curtas

P2 (escala e longo prazo)

- Matchmaking/instâncias múltiplas
- Sharding/zonas
- Moderação e ferramentas GM

---

8) Planejamento de entrega (marcos)

- **Alpha (30 dias)**: core loop + estabilidade + conteúdo mínimo
- **Beta (60–90 dias)**: economia + social + pipeline + retenção
- **Launch (120–180 dias)**: live ops, eventos, anti-cheat forte, escalabilidade

---

9) Checklist de "lançável" (o que a maioria esquece)

- Termos/Política/banimento (mínimo)

- Sistema de update do client
- Backup/restore do DB
- Migração de schema
- Monitoramento de memória/CPU/latência

---

10) Próximas ações sugeridas (bem objetivas)

1. Definir **vertical slice** (mapa/biomas/criaturas/skills) e congelar escopo por 30 dias.
2. Transformar criatura/skill/loot em **dados** (JSON/DB) e criar 1 ferramenta mínima para editar.
3. Implementar **market/trade seguro** + sinks de gold.
4. Fortalecer **anti-exploit** (inventário/gold/rate limit) + logs.
5. Rebranding/IP: garantir que tudo é original para comercializar.

Apêndice A — Ajustes recomendados no stack atual

- Server: separar “sistemas” (combat/loot/progression) e padronizar eventos WS.
- Client: pooling de FX, sorting layers consistentes, e reconexão.
- Website: painel admin + estatísticas básicas.