🗂️ CRONOGRAMA – 1soMath

# 1 ETAPA CONCEITUAL E PLANEJAMENTO

* Definição do conceito central
* Objetivo educacional: foco matemático e emocional
* Inspirações visuais e mecânicas: Gris, Monument Valley
* Como fazer:
* Brainstorm com referências visuais
* Criação de um documento com tópicos curtos sobre o que o jogo não será (limitar escopo)

# 2 DOCUMENTAÇÃO COMPLETA DO JOGO

* Organizar todos os documentos já finalizados
* Validar coerência: deixar pronto para desenvolvimento técnico
* Como fazer:
* Fazer uma última leitura geral
* Salvar versões finais numeradas (ex.: Pre\_projeto\_v1.0)

# 3 DESENVOLVIMENTO VISUAL E ARTÍSTICO (DETALHADO)

## a) Conceito Visual

* Elementos:
* Moodboard com referências de luz, sombra, cores
* Paleta de cores final (Runas, ambientes, Pit)

Como fazer:

Usar ferramentas como Figma ou Miro para o moodboard

Definir hexadecimal ou RGB exatos das cores das Runas

## b) Design de Runas

* Elementos:
* 12 Runas em vetor (ou PNG de alta qualidade)
* Animação de:
* Ativação (brilho/pulsar)
* Desgaste (escurecer, rachar, vibrar)
* Recarga (crescimento de energia visual)

Como fazer:

Criar esboços no Photoshop/Illustrator

Animar com After Effects ou Spine (para exportação leve)

## c) Cenários e Ambientação

* Elementos:
* Telas da Torre (fundo, plataformas, elementos flutuantes)
* Ambientes dos puzzles (cada um com tema: água, luz, sombra, etc.)
* Efeitos: névoa do Vazio, luz ambiente dinâmica

Como fazer:

Mapear cada tela como camadas separadas (background, middle, foreground)

Usar parallax para profundidade (Unity, Godot)

## d) Animações

* Elementos:
* Pit (andar, interagir, refletir, se cansar)
* Runas flutuando em torno de Pit
* Puzzles em movimento (plataformas, luzes, reflexos)

Como fazer:

Animações quadro-a-quadro ou esqueletais usando Spine ou Unity Animator

Exportar com transparência (WebP, PNG sequence)

# 4 PROGRAMAÇÃO E DESENVOLVIMENTO DE MECÂNICAS (DETALHADO)

## a) Sistema de Runas

* Tarefas:
* Script para desgaste progressivo (% de uso)
* Timer de recarga (contagem real de 8h)
* Verificação de recarga via anúncio (SDK de anúncios)

Como fazer:

Criar uma classe Runa com atributos: cor, energia, status

Usar PlayerPrefs ou banco local para armazenar tempo restante

Integrar Google AdMob ou Unity Ads

## b) Sistema de Puzzles

* Tarefas:
* Lógica para identificar solução correta
* Liberação da próxima área após sucesso
* Detecção do uso de Runas (qual foi usada)

Como fazer:

Scripts por tipo de puzzle (ex.: sequência, equilíbrio)

Criar prefabs para elementos comuns (botões, alavancas, luzes)

Usar coroutines para efeitos e transições após resolução

## c) Narrativa e Roteiro Interativo

* Tarefas:
* Inserir caixas de texto/diálogo
* Sistema de "inscrições" que aparecem no cenário

Como fazer:

Criar um sistema simples de UI Text ou Ink (para narrativa não linear)

Trigger por proximidade de Pit para ativar textos

## d) HUD e Interface

* Tarefas:
* Mostrador de Runas ativas
* Barra de desgaste + temporizador
* Botão de troca de Runa

Como fazer:

Usar Canvas do Unity

Atualizar valores com binding a variáveis da classe Runa

Efeito de brilho quando quase carregada (>90%)

## e) Economia e Anúncios

* Tarefas:
* SDK de anúncio (recompensa por recarga)
* Limitação de anúncios por tempo

Como fazer:

Instalar AdMob/Unity Ads SDK

Vincular evento de recarga ao anúncio (via callback)

# 5 TESTES INTERNOS E BALANCEAMENTO

## a) Mecânicas

* Tarefas: Testar desgaste das Runas, puzzles e HUD
* Como fazer:
* Rodar build em simuladores mobile
* Verificar fluxo: puzzle -> desgaste -> troca -> recarga

## b) UX/UI

* Tarefas: Avaliar clareza visual, resposta dos controles, legibilidade
* Como fazer:
* Prototipar em Unity Preview
* Gravar sessões para revisão detalhada

## c) Balanceamento

* Tarefas: Ajustar tempo dos puzzles, desgaste e quantidade de puzzles por Runa
* Como fazer:
* Inserir variáveis públicas para ajuste dinâmico
* Criar logs de tempo por sessão de jogo

# 6 ALPHA PLAYTESTING (FECHADO)

* Tarefas: Testes com equipe limitada
* Como fazer:
* Criar APKs e distribuir via Google Play Internal Testing
* Usar Google Forms para coletar feedback

# 7 AJUSTES PÓS-ALPHA

* Tarefas: Corrigir bugs e melhorar fluidez
* Como fazer:
* Revisar código usando debug logs
* Testar em diferentes aparelhos físicos

# 8 BETA PLAYTESTING (AMPLIADO)

* Tarefas: Testes com mais jogadores
* Como fazer:
* Ampliar via Google Play Beta
* Inserir sistema de coleta automática de erros (Unity Analytics)

# 9 AJUSTES FINAIS E POLIMENTO

* Tarefas: Revisão visual e performance
* Como fazer:
* Otimizar imagens/sprites (reduzir tamanhos, compressão)
* Ajustar memória e consumo de CPU/GPU

# 10 PUBLICAÇÃO E ENTREGA

* Tarefas: Submissão nas lojas
* Como fazer:
* Criar conta de desenvolvedor Google e Apple
* Preencher fichas de metadata (título, descrição, imagens)