Guia de Ajustes do Front — InfinityVerse (Pré■Integração)

Objetivo: detalhar campos, validações e lógicas que o front deve implementar AGORA para ficar 100% compatível com o banco posteriormente. ■■ Este documento foca em UX/validações e não inclui a integração com o Supabase.

1) Visão Geral das Páginas

Página	Função
index.html	Portal principal e navegação global
cadastrar_aluno.html	Formulário de cadastro de aluno (entrada na base)
cadastrar_projetos_empresas.html	Formulário de projetos criados por empresas
empresas.html	Página institucional e CTA para cadastro de projetos
login.html	Tela de autenticação (UI e validações)
admin.html	Painel administrativo (projetos/discipulos/indicadores)
discipulo.html	Dashboard do aluno (níveis, meus projetos e entregas)

2) Cadastro de Aluno — `cadastrar_aluno.html`

Campos e Regras (sem integração):

- nome (text, obrigatório): apenas letras e espaços; mínimo 3 caracteres.
- cpf (text, obrigatório): 11 dígitos (permitir máscara 000.000.000-00).
- data_nascimento (date, obrigatório): idade mínima de 10 anos.
- email (text, recomendado): formato válido; usado para login no futuro.
- foto (file, opcional): preview local já funciona; upload será na integração.

Validações obrigatórias:

- Nome: usar regex para bloquear números e caracteres especiais; trim() e normalizar espaços.
- CPF: validar dígitos verificadores e/ou garantir 11 dígitos; impedir caracteres não numéricos no value final.
- Data de nascimento: calcular idade a partir de hoje; bloquear < 10 anos; exibir mensagem clara.
- E-mail: validar com regex simples e toLowerCase(); limitar tamanho (ex.: ≤ 120 chars).
- Estados de erro/sucesso: borda/vermelho e mensagem inline abaixo do campo; evitar apenas alert().
- Botão 'Cadastrar': desabilitado enquanto houver erros visíveis ou campos vazios obrigatórios.

Mensagens UX sugeridas:

- CPF inválido. Use 11 dígitos (ex.: 000.000.000-00).
- Idade mínima: 10 anos.
- Informe um e■mail válido (ex.: nome@dominio.com).
- Preencha todos os campos obrigatórios.

Observações de implementação:

• Manter o `id` dos inputs exatamente iguais aos usados no JS (nome, cpf, dataNascimento, email, foto). • Separar a função de validação por campo (ex.: validateCPF, validateBirthdate) e validar a cada `input`/`blur`.

3) Cadastro de Projetos (Empresas) — `cadastrar_projetos_empresas.html`

Campos e Regras:

- empresa_nome (text, obrigatório): não aceitar só números; mínimo 3 caracteres.
- categoria (select, obrigatório): valores válidos: programacao, design, marketing, fotografia.
- descricao (textarea, obrigatório): mínimo 20 caracteres; ideal ≤ 2000.
- arquivo (file, opcional): aceitar apenas .pdf, .doc, .docx, .jpg, .jpeg, .png (validar no front).

Validações e UX:

- Categoria: impedir submit com valor vazio/placeholder.
- Descrição: contador de caracteres e bloqueio se < 20.
- Arquivo: validar extensão pelo nome e por `type` do File; exibir nome do arquivo e tamanho (ex.: ≤ 10 MB).
- Feedback visual: destacar campos inválidos; mensagens específicas sob cada campo.
- Botão 'Cadastrar': desabilitado até passar nas validações.

4) Login — `login.html`

Campos e Regras:

- email (text, obrigatório): formato válido, lowercased.
- senha (password, obrigatório): mínimo 6 caracteres (ideal: 8); exibir/ocultar senha.

Validações e UX:

- Bloquear submit com campos vazios.
- Exibir mensagem geral amigável para credenciais inválidas (sem detalhes técnicos).
- Tratar foco: ao erro, focar o primeiro campo inválido.

- Acessibilidade: labels associadas e `aria-live` para mensagens de erro.

5) Painel Admin — `admin.html`

Formulário de Projetos (sem integração):

- Campos obrigatórios: nomeProjeto, categoriaProjeto, descricaoProjeto; arquivo opcional.
- Validações: nome ≥ 3 chars; categoria válida; descrição ≥ 20 chars; arquivo com extensão válida se presente.
- UX: confirmação antes de deletar; rótulos/placeholder claros; manter consistência com tela de cadastro de empresas.

Módulo Discípulos (lista, detalhes, edição):

- Busca por nome com debounce (ex.: 300ms).
- Lista clicável com foto/nome; ao selecionar, mostrar detalhes e gráfico.
- Formulário de edição: validações iguais às do cadastro (nome/curso/telefone/email).
- Telefone: máscara e regex; Email: regex e lowercase.
- Confirmar antes de deletar um discípulo.

6) Dashboard do Discípulo — `discipulo.html`

Fluxos e Regras:

- Separar projetos por nível (iniciante/intermediario/avancado) e manter estado ativo na sidebar.
- Contadores: em andamento, concluídos, disponíveis (atualizar ao participar/entregar).
- Participar: mover projeto selecionado para 'Meus Projetos' com status 'andamento'.
- Modal de entrega: o campo link é obrigatório; validar URLs mínimas (github.com, instagram.com) quando o tipo correspondente for escolhido.
- Ao confirmar a entrega: marcar status como 'concluido' e registrar data local (para exibir no front).

Mensagens UX sugeridas:

- Selecione um tipo de entrega (GitHub, Instagram ou Arquivo).
- Informe um link válido para a entrega.
- Projeto enviado com sucesso!

7) Navegação, Rotas e Assets

- Links entre páginas: garantir que os botões do menu apontem para os arquivos corretos (ex.: caminho para cadastrar_aluno.html/discipulo.html).
- Padronizar caminhos de imagens e CSS/JS (ex.: usar `/assets/images/` e `/assets/js/`).

- Evitar múltiplos imports duplicados (ex.: FontAwesome em `index.html`).

8) Regras e Padrões Gerais de Validação

- Todos os formulários devem prevenir submit se houver campos inválidos (usar `form.checkValidity()` + validações custom).
- Exibir mensagens abaixo do campo com `aria-live='polite'` para acessibilidade.
- Limitar tamanho de strings (ex.: nome ≤ 120, descrições ≤ 2000).
- Sanitizar entrada: trim(), normalização de espaços, remoção de tags HTML.
- Desabilitar botão de submit durante processamento para evitar cliques duplos.

9) Checklists por Tela (Pré■Integração)

• Cadastro de Aluno

- Validação de nome/CPF/data/email implementadas.
- Mensagens de erro inline e acessíveis.
- Botão 'Cadastrar' bloqueado até tudo válido.

• Cadastro de Projetos (Empresas)

- Categoria obrigatória; descrição ≥ 20.
- Validação de extensão de arquivo e tamanho.
- Feedback visual de erro/sucesso por campo.

• Login

- Validação de e■mail e senha (mínimos).
- Feedback amigável para credenciais inválidas.
- Acessibilidade e foco no primeiro erro.

• Admin

- Form de projetos com validações e confirmação ao deletar.
- Edição de discípulo com validações (nome/curso/telefone/email).
- Busca com debounce.

• Discípulo

- Fluxos de participar/entregar atualizando contadores.
- Validação do link por tipo de entrega.
- Mensagens claras ao concluir uma entrega.

Observação Final

Depois que todas as validações e UX acima estiverem no front, a integração com o Supabase será plug■and■play: os mesmos campos/ids são reutilizados para `insert/select/update`, minimizando retrabalho.

Versão: 1.0 — Gerado automaticamente.