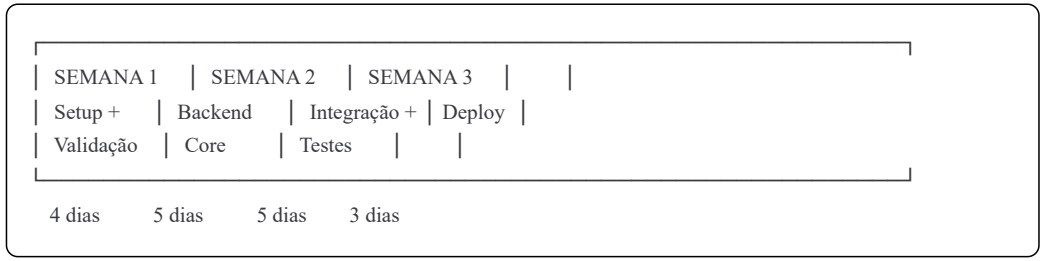
CRONOGRAMA DE DESENVOLVIMENTO

Sistema de Agendamento - Fotografia Imobiliária

Projeto: MVP Sistema de Agendamento Online
Início Previsto: [DATA DE INÍCIO]
Duração: 3-4 semanas (tempo integral) ou 6-8 semanas (part-time)
Equipe: 1 desenvolvedor full-stack

VISÃO GERAL DO PROJETO



Duração Total: 17 dias úteis ≈ **3 semanas**
Total de Horas: 120-140 horas
Custo de Infraestrutura: $0/mês (100% grátis até 1.000 agendamentos/mês)

SEMANA 1: PREPARAÇÃO E SETUP (28-32 horas)

DIA 1: Validação do Protótipo (6-8h)

Manhã (3-4h): Testes com Usuários

- [ ] Recrutar 3-5 testadores (amigos, clientes, família)
- [ ] Preparar roteiro de teste:
  - Tarefa 1: "Agende uma sessão de fotos para amanhã"
  - Tarefa 2: "Agende fotos + vídeo para sábado"
  - Tarefa 3: "Cancele um agendamento"
- [ ] Observar dificuldades e tempo gasto
- [ ] Anotar feedback verbal
- [ ] Registrar pontos de confusão

Tarde (3-4h): Análise e Ajustes

- [ ] Compilar feedback
- [ ] Priorizar ajustes críticos
- [ ] Implementar correções no protótipo
- [ ] Validar novamente (1-2 testadores)

Entregável: Lista de ajustes + Protótipo validado

DIA 2: Definições de Negócio (6-8h)

Manhã (3-4h): Dados Operacionais

- [ ] **Lista de Bairros Atendidos** (whitelist)
  - Pesquisar 20-30 bairros principais de Curitiba
  - Validar com equipe quais realmente atendem
  - Organizar em ordem alfabética
  - Documentar em planilha
- [ ] **Lista de Municípios Bloqueados** (blacklist)
  - Listar municípios da Região Metropolitana
  - Confirmar: Araucária, SJP, Colombo, Pinhais, etc.

Tarde (3-4h): Regras de Negócio

- [ ] **Durações dos Serviços** (confirmar)
  - Fotos: 40min? ✓
  - Vídeo Paisagem: 50min? ✓
  - Vídeo Retrato: 50min? ✓
  - Drone Fotos: 25min? ✓
  - Drone Fotos+Vídeo: 40min? ✓
- [ ] **Regras de Cancelamento** (definir)
  - Prazo gratuito: 24h ou 48h?
  - Taxa se < 24h: 50% ou 100%?
  - Horário mínimo para cancelamento: 2h?
- [ ] **Dados dos Fotógrafos**
  - Nome completo de cada um

- Email profissional
- Telefone/WhatsApp
- Confirmar serviços que cada um oferece

Entregável: Documento de regras de negócio completo

DIA 3: Setup de Contas e APIs (7-9h)

Manhã (4-5h): Contas e Credenciais

3.1 Google Cloud Platform (90min)

☐ Criar conta GCP (gmail@empresa.com)
☐ Criar novo projeto: "Agendamento-Foto"
☐ Habilitar APIs:

- ✓ Maps JavaScript API
- ✓ Places API
- ✓ Geocoding API
- ✓ Distance Matrix API

☐ Criar API Key
☐ Configurar restrições (domínio)
☐ Testar chamada básica

3.2 Vercel (30min)

☐ Criar conta Vercel
☐ Conectar GitHub/GitLab
☐ Familiarizar com dashboard

3.3 Banco de Dados (60min)

☐ Criar Supabase (free tier - RECOMENDADO)

- 500MB storage
- Queries ilimitadas
- Dashboard automático
- API REST automática

☐ Obter connection string
☐ Testar conexão local

- [ ] Criar primeiro schema de teste

3.4 Email (30min)

- [ ] Criar conta Resend
- [ ] Verificar domínio (se tiver)
  - OU usar domínio Resend
- [ ] Obter API Key
- [ ] Testar envio de email

Tarde (3-4h): Tadabase e Documentação

3.5 Tadabase/Make.com (90min)

- [ ] Documentar estrutura de dados Tadabase
- [ ] Identificar campos necessários
- [ ] Criar webhook endpoint (simulado)
- [ ] Testar recebimento de payload
- [ ] Configurar autenticação (se houver)

3.6 Organização (90min)

- [ ] Criar arquivo `.env.local` com todas as keys
- [ ] Documentar variáveis de ambiente
- [ ] Criar README inicial
- [ ] Organizar credenciais em gerenciador seguro

Entregável: Todas as contas criadas + credenciais documentadas

---

DIA 4: Repositório Git e Estrutura (6-8h)

Manhã (3-4h): Setup do Projeto

4.1 Repositório (60min)

- [ ] Criar repositório Git (GitHub/GitLab)
- [ ] Clonar localmente
- [ ] Adicionar .gitignore
- [ ] Primeiro commit

4.2 Next.js Boilerplate (120min)

- [ ] Instalar Next.js 14 com TypeScript
- [ ] Configurar Tailwind CSS
- [ ] Instalar Shadcn/ui
- [ ] Configurar Prisma ORM
- [ ] Setup de pastas:

```
/app
  /api
    /address
    /availability
    /bookings
    /webhooks
  /(routes)
    /page.tsx (landing)
    /agendar/page.tsx (booking flow)
/components
/lib
/prisma
/public
```

Tarde (3-4h): Banco de Dados

4.3 Schema Prisma (120min)

- [ ] Criar schema.prisma
- [ ] Definir models:
  - Photographer
  - CoverageArea
  - Booking
  - WebhookLog
- [ ] Criar migrations
- [ ] Rodar seed inicial (fotógrafos + bairros)

4.4 Testes Iniciais (60min)

- [ ] Testar conexão com banco
- [ ] Criar fotógrafo via Prisma
- [ ] Consultar dados
- [ ] Validar relacionamentos

Entregável: Projeto Next.js configurado + banco com dados seed

---

DIA 5: Migração do Protótipo (6-8h)

Manhã (3-4h): Copiar Código do Protótipo

- [ ] Copiar componentes do artifact para `/components`
- [ ] Ajustar imports
- [ ] Converter dados mockados em props
- [ ] Testar renderização

Tarde (3-4h): Integração Básica

- [ ] Criar páginas Next.js
- [ ] Configurar rotas
- [ ] Testar navegação
- [ ] Deploy preview no Vercel

Entregável: Protótipo rodando em Next.js localmente

SEMANA 2: DESENVOLVIMENTO BACKEND (35-42 horas)

DIA 6-7: APIs de Validação (12-16h)

API 1: Validação de Endereço (6-8h)

```typescript
// /app/api/address/validate/route.ts
POST /api/address/validate
Input: { address: string }
Output: {
  valid: boolean,
  neighborhood: string,
  city: string,
  lat: number,
  lng: number,
  inCoverage: boolean
}
```

Tarefas:

- [ ] Integrar Google Geocoding API
- [ ] Extrair componentes do endereço
- [ ] Validar contra whitelist de bairros
- [ ] Validar contra blacklist de municípios
- [ ] Calcular distância se não estiver na whitelist
- [ ] Aplicar margem de 3km
- [ ] Testes unitários
- [ ] Testes de integração

API 2: Buscar Endereços (Autocomplete) (6-8h)

```
typescript

// /app/api/address/search/route.ts
GET /api/address/search?q=Rua XV
Output: { suggestions: Address[] }
```

Tarefas:

- [ ] Integrar Google Places Autocomplete
- [ ] Filtrar apenas Curitiba
- [ ] Formatar resultados
- [ ] Cache de resultados (5min)
- [ ] Rate limiting
- [ ] Testes

Entregável: APIs de endereço funcionando

DIA 8-9: API de Disponibilidade (12-16h)

API: Consultar Horários Disponíveis (12-16h)

```
typescript
```

```
// /app/api/availability/route.ts
GET /api/availability?date=2026-01-25&services=photo,video_landscape
Output: {
  slots: [
    { time: "09:00", endTime: "10:30", available: 2 }
  ]
}
```

Tarefas:

- [ ] **Buscar agendamentos do dia** (2h)
  - Query Prisma com filtros
  - Incluir fotógrafo
  - Incluir serviços
- [ ] **Calcular fotógrafos disponíveis por serviço** (3h)
  - Verificar quem oferece o serviço
  - Filtrar quem está livre no horário
  - Considerar duração do serviço
- [ ] **Calcular distância e tempo** (4h)
  - Buscar último agendamento do fotógrafo
  - Usar Distance Matrix API
  - Filtrar se > 30min de deslocamento
  - Cache de distâncias calculadas
- [ ] **Gerar slots disponíveis** (2h)
  - Intervalos de 30min
  - Considerar duração total dos serviços
  - Bloquear slots insuficientes
- [ ] **Testes** (2h)
  - Cenário: nenhum agendamento
  - Cenário: alguns horários ocupados
  - Cenário: fotógrafo longe
  - Cenário: todos ocupados

Entregável: API de disponibilidade precisa

DIA 10: API de Agendamento (8-10h)

API: Criar Agendamento (8-10h)

```typescript
// /app/api/bookings/route.ts
POST /api/bookings
Input: {
  address: { full, lat, lng, neighborhood },
  services: ["photo", "video_landscape"],
  date: "2026-01-25",
  time: "09:00",
  client: { name, email, phone, notes }
}
Output: {
  protocol: "AG20260125001",
  booking_id: "uuid"
}
```

Tarefas:

- [ ] **Validações** (2h)
  - Todos os campos obrigatórios
  - Email válido
  - Telefone válido
  - Data futura (D+1)
  - Horário disponível
- [ ] **Lock temporário** (2h)
  - Criar registro com status "locked"
  - Expirar após 10min
  - Limpar locks expirados (cron job)
- [ ] **Alocar fotógrafo** (2h)
  - Filtrar por serviço
  - Filtrar por disponibilidade
  - Calcular distância
  - Escolher o mais próximo
- [ ] **Criar agendamento** (1h)

- Gerar protocolo único
- Gerar token de cancelamento
- Salvar no banco
- Atualizar status para "confirmed"

☐ **Integração Tadabase** (1h)

- Enviar webhook
- Retry em caso de falha
- Marcar como synced

☐ **Testes** (1-2h)

- Agendamento bem sucedido
- Horário ocupado (conflito)
- Fotógrafo indisponível
- Campos inválidos

Entregável: API de agendamento completa

SEMANA 3: INTEGRAÇÃO, TESTES E DEPLOY (30-36 horas)

DIA 11-12: Integração Frontend ↔ Backend (12-16h)

Tarefas:

☐ **Passo 1: Endereço** (3h)

- Substituir mock por API real
- Autocomplete real
- Validação de cobertura
- Loading states
- Error handling

☐ **Passo 2: Serviços** (2h)

- Manter lógica local (já funciona)
- Adicionar analytics

☐ **Passo 3: Calendário** (3h)

- Buscar disponibilidade real via API
- Desabilitar datas sem slots

- Loading skeleton

☐ **Passo 4: Horários** (3h)

- Carregar slots da API
- Mostrar duração calculada
- Atualizar em tempo real

☐ **Passo 5: Dados** (1h)

- Validações client-side
- Máscaras de input (telefone)

☐ **Passo 6: Confirmação** (2h)

- Enviar para API
- Handle de sucesso/erro
- Mostrar protocolo

Entregável: Fluxo completo integrado

DIA 13: Sistema de Emails (7-9h)

Templates de Email (4h)

☐ Design HTML responsivo
☐ Template: Confirmação de agendamento

- Logo da empresa
- Detalhes completos
- Link de cancelamento
- Instruções de preparação

☐ Template: Cancelamento
☐ Template: Notificação para fotógrafo

Integração Resend (3h)

☐ Configurar Resend SDK
☐ Função de envio
☐ Queue de emails (se falhar)
☐ Logs de envio

Testes (1-2h)

- [ ] Envio bem sucedido
- [ ] Teste de formatação (mobile)
- [ ] Links funcionando
- [ ] Fallback se falhar

Entregável: Emails automáticos funcionando

DIA 14: Webhook Tadabase → Sistema (7-9h)

API Webhook (4h)

```typescript
// /app/api/webhooks/tadabase/route.ts
POST /api/webhooks/tadabase
Input: Payload do Tadabase
Output: { success: boolean }
```

Tarefas:

- [ ] Validar assinatura HMAC
- [ ] Parsear payload
- [ ] Buscar fotógrafo por email
- [ ] Verificar disponibilidade
- [ ] Criar ou rejeitar agendamento
- [ ] Retornar resposta adequada

Tratamento de Conflitos (2h)

- [ ] Detectar horário ocupado
- [ ] Retornar erro 409
- [ ] Notificar admin
- [ ] Log completo

Testes (1-2h)

- [ ] Webhook válido
- [ ] Assinatura inválida
- [ ] Conflito de horário
- [ ] Dados incompletos

Entregável: Sincronização bidirecional funcionando

DIA 15: Cancelamento (6-8h)

API de Cancelamento (3h)

```typescript
// /app/api/bookings/[token]/cancel/route.ts
DELETE /api/bookings/:token/cancel
Output: { cancelled: boolean, fee: number }
```

Tarefas:

- [ ] Validar token único
- [ ] Verificar prazo (24h)
- [ ] Calcular taxa (se aplicável)
- [ ] Atualizar status no banco
- [ ] Liberar horário
- [ ] Notificar fotógrafo
- [ ] Atualizar Tadabase

Página de Cancelamento (2h)

- [ ] UI com confirmação
- [ ] Mostrar taxa aplicável
- [ ] Contador regressivo
- [ ] Sucesso/erro

Testes (1-2h)

- [ ] Cancelamento dentro do prazo
- [ ] Cancelamento fora do prazo
- [ ] Token inválido

Entregável: Cancelamento funcional

SEMANA 4: TESTES E DEPLOY (30-40 horas)

DIA 16-17: Testes e Deploy Final (12-15h)

Testes End-to-End (6-8h)

- [ ] Fluxo completo: endereço → confirmação
- [ ] Múltiplos serviços selecionados
- [ ] Diferentes datas e horários
- [ ] Cancelamento
- [ ] Webhook do Tadabase
- [ ] Conflitos de horário
- [ ] Testes mobile (iOS + Android)

Deploy em Produção (4-5h)

- [ ] Configurar variáveis de ambiente
- [ ] Deploy no Vercel
- [ ] Migrar banco Supabase
- [ ] Configurar domínio
- [ ] Testar em produção
- [ ] 5-10 agendamentos de teste

Monitoramento (2h)

- [ ] Configurar Vercel Analytics
- [ ] Configurar Supabase Logs
- [ ] Configurar alertas de erro
- [ ] Documentação de troubleshooting

Entregável: MVP EM PRODUÇÃO! 🚀

RESUMO DE HORAS POR ATIVIDADE (ATUALIZADO)

Atividade	Horas
Validação e Setup	28-32h
Backend APIs	35-42h

Atividade	Horas
Integração e Testes	30-36h
Deploy e Monitoramento	12-15h
TOTAL	**120-140h**

DISTRIBUIÇÃO POR PERFIL (ATUALIZADO)

Se você tem 1 desenvolvedor full-time (8h/dia):

- **Duração:** 15-18 dias úteis ≈ **3 semanas**

Se você tem 1 desenvolvedor part-time (4h/dia):

- **Duração:** 30-35 dias úteis ≈ **6-7 semanas**

Se você tem equipe (2 devs):

- **Duração:** 8-9 dias úteis ≈ **1.5-2 semanas**

MARCO DE ENTREGAS (MILESTONES) - ATUALIZADO

M1 - Fim da Semana 1 ✅

Entregável: Setup completo + Protótipo validado

- [ ] Todas as contas criadas (Vercel, Supabase, Google Maps, Resend)
- [ ] Banco Supabase com schema e seed
- [ ] Projeto Next.js rodando localmente
- [ ] Protótipo testado com 3-5 usuários reais

M2 - Fim da Semana 2 ✅

Entregável: Backend funcional

- [ ] APIs de validação funcionando (Google Maps integrado)
- [ ] API de disponibilidade calculando corretamente

- [ ] API de agendamento completa com alocação
- [ ] Webhook Tadabase recebendo e processando
- [ ] Testes unitários das APIs passando

M3 - Fim da Semana 3 🚀

Entregável: MVP EM PRODUÇÃO

- [ ] Frontend 100% integrado ao backend
- [ ] Emails automáticos funcionando
- [ ] Sistema de cancelamento operacional
- [ ] Testes mobile (iOS + Android) aprovados
- [ ] Deploy no Vercel com domínio
- [ ] Sistema acessível e funcional
- [ ] Primeiros 5-10 agendamentos reais de teste

Testes End-to-End (6-8h)

- [ ] Fluxo completo: endereço → confirmação
- [ ] Múltiplos serviços selecionados
- [ ] Diferentes datas e horários
- [ ] Cancelamento
- [ ] Webhook do Tadabase
- [ ] Conflitos de horário

Testes de Edge Cases (3-4h)

- [ ] Endereço fora de cobertura
- [ ] Todos fotógrafos ocupados
- [ ] Domingo/feriado
- [ ] Horário muito próximo do fim do expediente
- [ ] Email inválido
- [ ] Telefone inválido

Testes de Carga (3-4h)

- [ ] 10 usuários simultâneos
- [ ] 50 agendamentos em sequência
- [ ] Tempo de resposta < 2s
- [ ] Sem vazamento de memória

Entregável: Lista de bugs + correções

DIA 18: Testes Mobile e UX (6-8h)

Dispositivos (4-5h)

- [ ] iPhone (Safari)
  - Fluxo completo
  - Autocomplete
  - Calendário touch
  - Botões clicáveis
- [ ] Android (Chrome)
  - Mesmos testes
- [ ] Tablet iPad
- [ ] Tablet Android

Acessibilidade (2-3h)

- [ ] Navegação por teclado
- [ ] Screen reader (básico)
- [ ] Contraste de cores (WCAG AA)
- [ ] Tamanho mínimo de fonte (16px)

Entregável: Sistema aprovado em mobile

DIA 19: Correções e Refinamentos (6-8h)

Prioridade Alta (4-5h)

- [ ] Corrigir bugs críticos
- [ ] Melhorar mensagens de erro
- [ ] Ajustar textos confusos
- [ ] Otimizar loading states

Prioridade Média (2-3h)

- [ ] Melhorias de UX
- [ ] Animações suaves

- [ ] Feedback visual

Entregável: Sistema polido

DIA 20: Deploy em Produção (4-6h)

Preparação (2h)

- [ ] Revisar variáveis de ambiente
- [ ] Configurar domínio (se tiver)
- [ ] Configurar SSL
- [ ] Testar em ambiente de staging

Deploy (1h)

- [ ] Push para branch `main`
- [ ] Vercel faz deploy automático
- [ ] Verificar build bem sucedido
- [ ] Migrar banco de dados

Validação (1-2h)

- [ ] Testar em produção
- [ ] Verificar emails chegando
- [ ] Verificar webhook Tadabase
- [ ] Fazer 3-5 agendamentos teste

Monitoramento (1h)

- [ ] Configurar Vercel Analytics
- [ ] Configurar alertas de erro
- [ ] Configurar backup diário

Entregável: MVP EM PRODUÇÃO! 🚀

RESUMO DE HORAS POR ATIVIDADE

Atividade	Horas
Validação e Setup	35-40h

Atividade	Horas
Backend APIs	40-50h
Integração Frontend	35-45h
Testes e Deploy	30-40h
TOTAL	**145-170h**

DISTRIBUIÇÃO POR PERFIL

Se você tem 1 desenvolvedor full-time (8h/dia):

- **Duração:** 18-22 dias úteis (~4 semanas)

Se você tem 1 desenvolvedor part-time (4h/dia):

- **Duração:** 36-42 dias úteis (~6-8 semanas)

Se você tem equipe (2 devs):

- **Duração:** 9-11 dias úteis (~2 semanas)

MARCO DE ENTREGAS (MILESTONES)

M1 - Fim da Semana 1 ✅

Entregável: Setup completo + Protótipo validado

- [ ] Todas as contas criadas
- [ ] Banco de dados com seed
- [ ] Projeto Next.js rodando
- [ ] Protótipo testado com usuários

M2 - Fim da Semana 2 ✅

Entregável: Backend funcional

- [ ] APIs de validação funcionando

- [ ] API de disponibilidade precisa
- [ ] API de agendamento completa
- [ ] Testes unitários passando

M3 - Fim da Semana 3 ✅

Entregável: Sistema integrado

- [ ] Frontend conectado ao backend
- [ ] Emails automáticos
- [ ] Webhook Tadabase funcionando
- [ ] Cancelamento operacional

M4 - Fim da Semana 4 🚀

Entregável: MVP EM PRODUÇÃO

- [ ] Todos os testes passando
- [ ] Deploy no Vercel
- [ ] Sistema acessível publicamente
- [ ] Primeiros clientes reais agendando

CHECKLIST DIÁRIO DO DESENVOLVEDOR

Início do Dia:

- [ ] Review do que foi feito ontem
- [ ] Definir 3 tarefas prioritárias de hoje
- [ ] Verificar se há blockers

Fim do Dia:

- [ ] Commit do código (mensagem clara)
- [ ] Atualizar status das tarefas
- [ ] Documentar decisões importantes
- [ ] Anotar blockers para amanhã

RISCOS E CONTINGÊNCIAS

Risco	Ação
Google Maps API demora muito	Implementar cache agressivo + usar mock temporário
Vercel deployment falha	Ter plano B: Railway, Render
Tadabase API instável	Implementar retry + queue
Desenvolvedor fica doente	Buffer de 2-3 dias no cronograma
Requisitos mudam no meio	Congelar escopo até MVP lançar

PRÓXIMAS AÇÕES

Imediato (esta semana):

- [ ] Aprovar este cronograma
- [ ] Definir data de início
- [ ] Decidir: quem desenvolve? (você, contrato, eu)
- [ ] Reunir dados de negócio (bairros, fotógrafos)

Semana que vem:

- [ ] Criar todas as contas
- [ ] Setup inicial do projeto
- [ ] Começar desenvolvimento

Cronograma preparado para execução
Ajustar datas conforme início real do projeto