

AbsolutSport Forms

Plataforma de Coleta de Dados — Turismo Esportivo

Especificação do MVP | v0.1

1. Contexto e Objetivo

A agência opera como intermediária no turismo esportivo, vendendo pacotes que incluem ingressos, transfers, aéreo, hospedagem e seguro. O processo atual de coleta de dados dos clientes é feito via SurveyMonkey (B2C) e planilhas (B2B), causando retrabalho, inconsistências e gargalos operacionais.

O MVP resolve o problema central: substituir formulários estáticos por uma plataforma própria, dinâmica e auditável, focando exclusivamente no fluxo B2C.

2. O Que Entra e o Que Fica de Fora

As decisões abaixo foram tomadas com base no princípio de entregar valor real ao cliente no menor tempo possível, sem comprometer a operação core.

| Funcionalidade | Descrição | Versão |
|-------------------------------|---|--------|
| Criação de projetos e eventos | Admin cria projetos (ex: Libertadores 2026) | MVP |
| Formulários dinâmicos | Campos personalizáveis por tipo e formato | MVP |
| Portal do cliente (login) | Área autenticada com acesso aos formulários | MVP |
| Ficha do usuário | Perfil reutilizável entre eventos | MVP |
| Fluxo de acompanhantes | Replicação dinâmica de campos por pessoa | MVP |
| Hospedagem básica (flag) | Campo para sinalizar se pacote inclui | MVP |

| | | |
|--------------------------------|--|-----|
| | hospedagem | |
| Dashboard admin básico | Contagem de status dos formulários | MVP |
| Log de auditoria simplificado | Registro de ações com data/hora e usuário | MVP |
| Janela de edição (data limite) | Admin define prazo para alterações | MVP |
| Auditoria completa (IP/device) | Rastreamento de IP, dispositivo e tempo de preenchimento | V2 |
| Integração com PMS/CRM | Conexão com sistemas externos | V2 |

3. Perfis de Usuário

O sistema possui três níveis de acesso em hierarquia de permissões. Os dois primeiros são usuários internos da agência.

| Nível | Perfil | Escopo de Acesso |
|-------|----------------|---|
| 1 | Master | Acesso total — gestão de usuários, logs completos do sistema e todos os recursos do Admin |
| 2 | Admin | Acesso operacional — propostas, captações, matriz de dados e exportações |
| 3 | Cliente | Acesso restrito — apenas sua própria captação, via link e senha temporária |

3.1 Master

Nível máximo de acesso. O Master é o administrador da plataforma dentro da agência. Apenas um Master pode criar outros Masters ou Admins — essa permissão não é delegável para baixo na hierarquia.

- Cria, edita e desativa usuários Master e Admin
- Acessa o painel de logs completo do sistema — registra absolutamente tudo: logins, ações de Admin, alterações de dados, erros, disparos de e-mail, sincronizações com o Sheets e acessos de clientes
- Visualiza todos os recursos do Admin sem restrição de proposta ou escritório
- Pode operar em nome de qualquer Admin se necessário
- Recebe alertas críticos do sistema: erros de sync com Sheets, falhas de envio de e-mail, tentativas de acesso indevido

O Master não interfere na rotina operacional diária. Seu papel é governança, auditoria e gestão de acesso.

3.2 Admin

Usuário operacional interno da agência. Responsável por toda a gestão de captação de dados da World Cup 2026.

- Acessa a visão completa das propostas importadas do Sales Log
- Verifica os pré-requisitos e dispara a captação: via envio automático por e-mail ou copiando o link para envio manual
- Define o prazo limite para preenchimento de cada captação
- Visualiza a matriz de dados completa: dados da venda (Sales Log) + dados captados dos passageiros
- Edita dados coletados diretamente — correções operacionais ou ajustes solicitados pelo cliente
- Preenche os campos internos da ficha: Ticket Status, Hotel Confirmation Number, Insurance Number
- Acompanha o dashboard de status de captação por proposta
- Exporta fichas operacionais por proposta — separadas por fornecedor: hotel, companhia aérea, seguradora, empresa de transporte e FIFA

O Admin não tem acesso ao painel de logs do sistema nem à gestão de usuários.

3.3 Cliente (Usuário Final)

O titular da compra. Acessa o portal exclusivamente para preencher os dados de captação da sua proposta. Não tem conta prévia na plataforma — o acesso é criado no momento do disparo pelo Admin.

- Recebe e-mail com link único de acesso e senha temporária gerada automaticamente pelo sistema
- No primeiro acesso, é orientado a definir uma nova senha
- Acessa apenas a própria captação — sem visibilidade de outras propostas ou clientes
- Visualiza os detalhes do pacote já preenchidos (jogo, hotel, quartos, datas, ingresso) vindos do Sales Log
- Preenche os dados de cada passageiro nos blocos correspondentes ao seu quarto
- Pode salvar e retornar — o progresso é mantido por passageiro, sem necessidade de concluir tudo de uma vez
- Visualiza o prazo restante e o status de cada ficha: Pendente | Preenchida

4. Funcionalidades do MVP

4.1 Sincronização com Google Sheets — Sales Log

O ponto de partida de todo o processo. O sistema conecta-se ao Google Sheets da Absolut Sport e importa os dados de cada venda. Não há entrada manual de dados de venda na plataforma — o Sales Log é a fonte única de verdade para os dados comerciais.

- A sincronização é disparada manualmente pelo Admin ou agendada periodicamente
- Todas as linhas do Sales Log são importadas, independente de status
- Cada linha vira um registro na tabela sales_orders do banco, com o STATUS original preservado
- Atualizações no Sheets (mudança de status, hotel, datas, número de pax, e-mail) são refletidas no sistema na próxima sync
- O Admin visualiza todas as propostas importadas com seus dados completos antes de disparar qualquer captação

4.2 Geração Automática do Formulário de Captação

Com a venda importada, o sistema já sabe tudo que precisa para montar o formulário: quantos passageiros existem, em quantos quartos se distribuem, qual o tipo de cada quarto, qual hotel, quais datas e qual jogo. Nenhum campo precisa ser configurado manualmente pelo Admin.

Dados coletados por passageiro (fixos para World Cup 2026)

- Nome completo (primeiro nome + sobrenome)
- Nacionalidade
- Gênero
- Tipo de documento + número (CPF, RG, Passaporte ou DNI — tipo varia por nacionalidade)
- País emissor do documento
- Data de validade do documento
- Data de nascimento
- Time que torce (opcional — controle de distribuição de ingressos por setor)
- Telefone (com código de país)
- E-mail

Estrutura do formulário gerado

O formulário é organizado hierarquicamente: por jogo → por hotel → por quarto → por passageiro. O cliente vê blocos nomeados (ex: 'Quarto 1 — KING | Check-in 11/06 | Distrikt Hotel') e dentro de cada bloco as fichas de cada hóspede que precisa preencher.

- Propostas Ticket Only geram apenas fichas pessoais por ingresso — sem bloco de quarto
- Propostas com múltiplos jogos geram seções separadas por jogo, cada uma com seu hotel
- O número de fichas e a estrutura de quartos vêm exclusivamente dos campos NUMBER OF PAX, NUMBER OF ROOMS e ROOM TYPE do Sales Log

4.3 Disparo da Captação — Geração do Link de Acesso

Após a sincronização, o Admin seleciona uma proposta elegível e dispara a captação. O sistema oferece dois modos de disparo, mas em ambos a lógica de criação de acesso é idêntica.

Pré-requisitos obrigatórios para habilitar o disparo

| Condição | Regra | Se não atendido |
|--------------------|---|--|
| STATUS = CONFIRMED | O campo STATUS da proposta no Sales Log deve estar como CONFIRMED | Botão de disparo bloqueado — exibe aviso: 'Proposta não confirmada' |
| EMAIL preenchido | O campo EMAIL da proposta no Sales Log deve conter um endereço válido | Botão de disparo bloqueado — exibe aviso: 'E-mail do cliente ausente no Sales Log' |

Modo 1 — Envio automático por e-mail

O Admin clica em 'Disparar captação'. O sistema executa automaticamente:

- Cria a conta do cliente no banco com o e-mail do Sales Log
- Gera uma senha temporária aleatória (ex: Ab3#kQ9m)
- Envia e-mail transacional via Resend contendo: link único de acesso, senha temporária e prazo de preenchimento
- Registra o disparo no log com data, hora e usuário Admin responsável
- Proposta muda para status 'Aguardando preenchimento' no dashboard

Esse é o modo padrão e recomendado. O cliente já recebe tudo o que precisa em um único e-mail.

Modo 2 — Geração de link para compartilhamento manual

Quando o Admin prefere enviar o link por outro canal (WhatsApp, telefone, e-mail próprio):

- Admin clica em 'Gerar link'
- O sistema cria a conta do cliente, gera a senha temporária e retorna o link único + a senha para o Admin copiar
- O Admin compartilha o link e a senha com o cliente pelo canal de sua preferência
- O sistema registra a geração do link no log — independente de o cliente acessar ou não

Atenção: o link é único por proposta. O mesmo e-mail pode ter múltiplos links ativos simultaneamente — cada um para uma proposta diferente.

1 proposta = 1 link = 1 captação independente

Um mesmo e-mail pode aparecer múltiplas vezes no Sales Log — seja porque o mesmo cliente comprou pacotes diferentes, seja porque uma agência parceira (como uma operadora ou B2B) usa um único e-mail de contato para gerir diversas propostas de passageiros distintos.

Independente do motivo, a regra é invariável: cada proposta gera seu próprio link e sua própria captação. O vínculo nunca é pelo e-mail — é sempre pela proposta (PROPOSAL ID).

| Cenário | Como o sistema trata |
|---------|----------------------|
|---------|----------------------|

| | |
|---|--|
| Cliente individual com 2 compras (ex: M29 e M49 — mesmo e-mail) | 2 propostas distintas geram 2 links independentes. O cliente faz login e vê as duas captações ativas no seu painel — cada uma com seus dados, quartos e prazos separados |
| Agência parceira com 1 e-mail de contato e 8 propostas (ex: Travepack) | 8 propostas geram 8 links independentes. O responsável da agência acessa com o mesmo login e vê todas as captações listadas — preenche cada uma de forma isolada |
| Agência com dezenas de propostas (ex: Ferrara Viajes — 40+ no mesmo e-mail) | Cada proposta tem seu link, sua ficha e seu prazo. O sistema não mistura dados entre propostas. O painel do cliente lista todas ordenadas por status e prazo |
| Proposta cancelada com mesmo e-mail de outra ativa | A captação da proposta CANCELED nunca é disparada — os pré-requisitos bloqueiam. Apenas as propostas CONFIRMED geram link. As demais ficam invisíveis para o cliente |

Consequência técnica direta: a chave de identidade do sistema nunca é o e-mail isolado. A conta do cliente é identificada pelo par e-mail + proposta. Ao fazer login com seu e-mail, o cliente vê um painel com todas as captações ativas vinculadas àquele e-mail — nunca os dados cruzados de uma proposta na outra.

Senha temporária — regras

- Gerada automaticamente pelo sistema em ambos os modos de disparo
- Expiração: o cliente é obrigado a definir uma nova senha no primeiro login
- Se o cliente já tem conta ativa (e-mail já cadastrado por outra proposta), não recebe nova senha — o acesso à nova captação é adicionado ao mesmo login existente
- Complexidade mínima: 8 caracteres, letras e números
- Não é armazenada em texto puro — apenas o hash bcrypt é salvo no banco
- Se o cliente perder a senha, o Admin pode regenerar via painel — novo e-mail disparado ou nova senha temporária para copiar

4.4 Portal do Cliente

O cliente acessa o portal com seu e-mail e senha. O painel inicial lista todas as captações ativas vinculadas ao seu e-mail, organizadas por status e prazo. Se tiver apenas uma, cai diretamente nela.

- Cada captação é isolada — dados de uma proposta nunca aparecem em outra
- Dentro de cada captação, visualiza os detalhes do pacote já preenchidos (jogo, hotel, quartos, datas, ingresso) vindos do Sales Log
- Preenche os dados de cada passageiro nos blocos correspondentes ao seu quarto
- Pode salvar e retornar — o progresso é mantido por passageiro, sem necessidade de concluir tudo de uma vez
- Visualiza o status de cada ficha: Pendente | Preenchida
- Visualiza o prazo restante de cada captação

O portal é responsivo e funciona em mobile e desktop.

4.5 Matriz de Dados do Admin — Visão Completa

O Admin tem acesso à visão consolidada de cada proposta: os dados da venda (importados do Sales Log) lado a lado com os dados captados dos passageiros. Essa é a principal ferramenta operacional da plataforma.

- Visualiza por proposta: todos os passageiros, seus quartos, hotéis, datas e ingressos
- Edita qualquer dado coletado diretamente — para correções reportadas pelo cliente ou erros de digitação
- Preenche os campos operacionais internos por produto: Ticket Status/Link (ingresso), Hotel Confirmation Number (hospedagem), Flight Locator/Cia Aérea (voo), Insurance Number/Seguradora (seguro), Transfer Reference (transporte)
- Filtra e busca por proposta, cliente, status, jogo ou hotel
- Exporta fichas operacionais por proposta — separadas por fornecedor: hotel, companhia aérea, seguradora, empresa de transporte e FIFA

4.6 Dashboard de Acompanhamento

O dashboard é a central de controle operacional da captação. Admin e Master visualizam em tempo real o estado de cada proposta — quem recebeu, quem preencheu, quanto falta, e onde estão os gargalos. Todas as métricas são calculadas dinamicamente a partir do banco, sem cache fixo.

Visão global — cards de resumo no topo

| | | | |
|--|--|---|--|
| Total disparadas Link enviado ou gerado para o cliente | Não iniciadas Cliente ainda não acessou o portal | Em andamento Pelo menos 1 ficha preenchida, mas não todas | Concluídas 100% das fichas preenchidas |
|--|--|---|--|

Percentual de preenchimento — cálculo por proposta

Cada proposta exibe seu percentual de preenchimento individualmente. O cálculo é feito sobre fichas de passageiro — não sobre propostas inteiras — para refletir a realidade operacional com precisão.

| Métrica | Como é calculada |
|--------------------------------|---|
| % de preenchimento da proposta | $\text{fichas_preenchidas} \div \text{total_fichas_da_proposta} \times 100$ |
| % global do evento | $\text{total_fichas_preenchidas} \div \text{total_fichas_disparadas} \times 100$ — visão consolidada de toda a operação |
| Fichas pendentes | $\text{total_fichas} - \text{fichas_preenchidas}$ — quantas faltam para concluir a proposta |
| Passageiros sem nenhum dado | fichas onde nenhum campo foi salvo ainda — distinção importante de 'em andamento' |
| Prazo em risco | propostas em andamento com menos de 48h para o deadline — sinalizadas em amarelo |
| Prazo expirado incompleto | propostas onde o prazo venceu com percentual < 100% — sinalizadas em vermelho |

Tabela de propostas — visão operacional linha a linha

Abaixo dos cards globais, uma tabela lista cada proposta ativa com todas as informações de acompanhamento visíveis de uma vez. Cada linha é clicável e abre a matriz de dados completa da proposta.

| PROPOSAL | Cliente | Jogo | Hotel | Fichas | % | Status / Prazo |
|----------|-------------|----------|----------------|--------|------|----------------|
| 20250602 | BTG Pactual | M7 — NYC | Distrikt Hotel | 2 / 2 | 100% | Concluída |
| 20260040 | Personal | M7 — NYC | Riu Plaza | 8 / 15 | 53% | Em andamento |
| 20250445 | Grupo Salta | Final | Marriott | 0 / 12 | 0% | Nao iniciada |

Exemplo ilustrativo com propostas reais do Sales Log. Os percentuais e status são calculados em tempo real.

Filtros e busca

- Por status: Não iniciada | Em andamento | Concluída | Prazo em risco | Expirada
- Por jogo: M7, Final, M49 etc — filtra todas as propostas que incluem aquele jogo
- Por hotel: lista propostas vinculadas a um hotel específico
- Por seller: quais propostas cada vendedor tem e qual o percentual de captação delas
- Por faixa de percentual: ex. 'mostrar apenas propostas com menos de 50% preenchido'
- Busca livre por PROPOSAL ID, nome do cliente ou e-mail

Gráfico de progresso global

Uma barra de progresso horizontal consolidada no topo do dashboard exhibe o avanço geral da operação: X fichas preenchidas de Y total disparadas, com percentual. Abaixo, um breakdown por status em formato de barras horizontais empilhadas — concluídas (verde), em andamento (azul), não iniciadas (cinza), expiradas (vermelho). Simples, sem gráficos complexos — o que importa é a leitura imediata.

4.7 Janela de Edição

O Admin define, ao disparar a captação, a data limite até a qual o cliente pode editar suas respostas. Após essa data, os campos ficam visíveis porém bloqueados para o cliente.

- O Admin pode estender o prazo manualmente a qualquer momento
- O Admin pode editar os dados coletados a qualquer momento, independente do prazo
- Após o prazo, o sistema sinaliza automaticamente as fichas como 'Prazo encerrado'

4.9 Exportação de Dados

O sistema permite baixar os dados em múltiplos recortes, cobrindo todas as necessidades operacionais: da ficha completa para cada fornecedor (hotel, companhia aérea, seguradora, empresa de transporte) até extrações parciais para análise interna. Todos os downloads são em formato CSV — compatível com Excel e Google Sheets sem conversão. Cada exportação é registrada no log de auditoria com data, hora e Admin responsável.

| Tipo de exportação | O que contém | Caso de uso principal |
|--------------------|--------------|-----------------------|
|--------------------|--------------|-----------------------|

| | | |
|---|--|---|
| Matriz completa — proposta | Todos os campos: dados do Sales Log (azul) + dados coletados dos passageiros (verde) + campos internos preenchidos pelo Admin (amarelo). Uma linha por passageiro. | Envio ao hotel, companhia aérea, seguradora, empresa de transporte e FIFA — ficha operacional completa por proposta |
| Somente respostas do formulário | Apenas os dados captados diretamente dos passageiros: nome, documento, nacionalidade, gênero, data de nascimento, telefone, e-mail e time. Sem dados de venda. | Análise interna, validação de documentos, controle de qualidade das respostas |
| Somente dados do Sales Log | Dados importados do Sheets: PROPOSAL, CLIENT, GAME, HOTEL, ROOM TYPE, CHECK IN/OUT, TICKET CAT, SELLER etc. Sem dados dos passageiros. | Relatório comercial, verificação da sync, auditoria de propostas importadas |
| Status de captação — todas as propostas | Uma linha por proposta: PROPOSAL ID, cliente, jogo, total de fichas, fichas preenchidas, percentual, status e prazo. | Relatório de desempenho da operação — monitoramento gerencial |
| Propostas pendentes / em risco | Subconjunto filtrado: apenas propostas com percentual < 100% e prazo próximo ou expirado. Inclui e-mail do cliente para follow-up. | Lista de ação imediata — equipe de vendas sabe quem acionar |
| Exportação global — evento completo | Todas as propostas ativas do evento, todos os passageiros, todos os campos. Uma linha por passageiro em ordem de proposta → quarto → passageiro. | Visão consolidada de toda a operação World Cup 2026 em um único arquivo |

Regras e comportamento das exportações

- Formato: CSV com encoding UTF-8 BOM — garante abertura correta no Excel sem distorção de caracteres especiais (acentos, ã, ç)
- Nome do arquivo gerado automaticamente: PROPOSAL_ID_TIPO_DATA.csv — ex: 20250602_matriz_completa_2026-06-01.csv
- Exportações parciais (por proposta) ficam disponíveis na tela da proposta. A exportação global fica no menu do dashboard
- Campos ainda não preenchidos pelo cliente aparecem como célula vazia — nunca como 'null' ou 'undefined'
- O Admin pode exportar qualquer proposta a qualquer momento, independente do status de preenchimento
- Propostas com 0% preenchido exportam apenas os dados do Sales Log — as colunas dos passageiros ficam em branco

4.8 Log de Auditoria — Sistema Completo

O sistema registra absolutamente todas as ações, de todos os atores, com timestamp e identificação. O painel de logs completo é acessível exclusivamente pelo Master. O Admin visualiza apenas os logs das captações que ele mesmo operou.

| Categoria | O que é registrado | Visível para |
|-----------|--------------------|--------------|
|-----------|--------------------|--------------|

| | | |
|------------------------|--|----------------|
| Autenticação | Login, logout, troca de senha, tentativas com falha, acesso com senha temporária | Master |
| Gestão de usuários | Criação, edição e desativação de Masters e Admins — com identificação de quem executou | Master |
| Sincronização Sheets | Início, fim, número de registros importados, erros de sync e linhas ignoradas | Master |
| Disparo de captação | Modo (e-mail ou link manual), proposta, Admin responsável, data e hora | Master + Admin |
| E-mail transacional | Envio, entrega, bounce, falha — por destinatário e proposta | Master |
| Acesso do cliente | Primeiro acesso, trocas de senha, acessos subsequentes | Master + Admin |
| Preenchimento de ficha | Cada save de passageiro: data, hora, usuário (cliente ou admin) e campos alterados | Master + Admin |
| Edição pelo Admin | Qual campo foi alterado, valor anterior e novo, data, hora e Admin responsável | Master + Admin |
| Exportação | Data, hora, Admin que exportou e proposta exportada | Master + Admin |
| Erros do sistema | Falhas de API, erros de validação, exceções não tratadas | Master |

Cada entrada de log contém: timestamp UTC, ID do usuário, role (Master/Admin/Cliente), IP de origem, tipo de ação, entidade afetada (proposta, ficha, usuário) e payload da mudança. Logs são imutáveis — nenhum Admin ou Master pode deletar ou editar um registro de log.

5. Stack Técnica

5.1 Visão Geral

Stack definida para máxima velocidade de entrega e manutenibilidade. Node.js no backend por consistência de linguagem com o frontend, ecossistema maduro para APIs REST e integração nativa com as bibliotecas de sync com Google Sheets.

| Camada | Decisão |
|---------------------|---|
| Runtime / API | Node.js 18+ + Express.js 5 |
| Banco de dados | PostgreSQL 15+ com multi-tenancy |
| Autenticação | Passport JWT — stateless, sem sessão em servidor |
| Hash de senhas | bcrypt — padrão de mercado |
| E-mail transacional | Resend — API moderna, retry nativo, boa deliverability |
| Validação | Zod — schemas dinâmicos, compartilhado entre front e back |

| | |
|-----------------|--|
| Segurança HTTP | Helmet + express-rate-limit |
| Observabilidade | Winston (logs) + Sentry (erros em produção) |
| Storage | Supabase Storage — arquivos de documentos dos clientes |
| Frontend | React 19 + Vite 6 + TypeScript |
| ORM | Prisma — migrations, type safety, schema como fonte de verdade |
| Containerização | Docker + Docker Compose — back, front e postgres em containers |
| Deploy | Railway — CI/CD automático via GitHub, suporte nativo a Docker |

5.2 Backend — Dependências

Pacotes Node.js com versões fixadas para o MVP:

| Pacote | Versão | Finalidade |
|---------------------------------|----------|---|
| <code>express</code> | 5.1.0 | Framework HTTP |
| <code>@prisma/client</code> | 6.9.0 | ORM type-safe — queries, migrations e schema declarativo |
| <code>prisma</code> | 6.9.0 | CLI Prisma — dev dependency (migrate, generate, studio) |
| <code>passport-jwt</code> | 4.0.1 | Autenticação stateless JWT |
| <code>bcrypt</code> | 5.1.1 | Hash seguro de senhas |
| <code>zod</code> | 3.24.2 | Validação de schemas — regras de captação por passageiro e validação compartilhada entre front e back |
| <code>resend</code> | 4.5.1 | E-mails transacionais com retry nativo |
| <code>helmet</code> | 8.0.0 | Headers de segurança HTTP (XSS, CSRF, etc.) |
| <code>express-rate-limit</code> | 7.5.0 | Rate limiting — protege endpoints públicos de formulários |
| <code>winston</code> | 3.17.0 | Logging estruturado |
| <code>@sentry/node</code> | 9.12.0 | Monitoramento de erros em produção |
| <code>dayjs</code> | 1.11.13 | Manipulação de datas — bundle leve, substitui date-fns |
| <code>googleapis</code> | 130.0.0 | SDK oficial do Google — autenticação OAuth2/Service Account e acesso à Sheets API v4 |
| <code>node-cron</code> | 3.0.3 | Scheduler de tarefas recorrentes — executa a sync com o Sheets a cada 1 minuto |
| <code>crypto</code> | built-in | Módulo nativo Node.js — gera hash SHA-256 de cada linha para detectar mudanças sem reescrever o banco |

5.3 Frontend — Dependências

React 19 + Vite 6 como base. Destaques: react-hook-form com Zod para validação dinâmica dos campos de captação, @tanstack/react-query para sincronização de estado com a API e framer-motion para feedback visual no preenchimento das fichas.

| Pacote | Versão | Finalidade |
|-----------------------|---------|--|
| react + react-dom | 19.1.0 | Core framework |
| react-router-dom | 7.5.1 | Navegação SPA |
| vite | 6.3.1 | Build tool ultra-rápido |
| zustand | 5.0.3 | State management local — leve e sem boilerplate |
| @tanstack/react-query | 5.74.4 | Server state, cache e sincronização com API |
| axios | 1.8.4 | Cliente HTTP |
| react-hook-form | 7.55.0 | Gerenciamento de formulários dinâmicos — essencial para o form builder |
| @hookform/resolvers | 4.1.3 | Integração react-hook-form + Zod |
| zod | 3.24.2 | Validação compartilhada (mesmo schema do backend) |
| tailwindcss | 4.1.3 | Design system utilitário |
| framer-motion | 12.7.3 | Animações de transição e feedback de UI |
| lucide-react | 0.487.0 | Ícones modernos e tree-shakeable |
| react-dropzone | 14.3.8 | Upload de documentos pelos clientes |
| dayjs | 1.11.13 | Manipulação de datas — consistente com o backend |

5.4 Decisões Arquiteturais — Pontos de Atenção

Zod + react-hook-form + @hookform/resolvers — Schema de captação

Os campos coletados de cada passageiro são fixos para a World Cup 2026 (nome, documento, nacionalidade, etc.), mas o schema de validação ainda precisa ser dinâmico por um motivo importante: o tipo de documento varia por nacionalidade (CPF para brasileiros, Passaporte para estrangeiros, DNI para argentinos), e algumas regras de validação podem variar por campo ou por proposta.

A estratégia: o schema Zod é gerado no backend com base nas regras da proposta e devolvido para o frontend. react-hook-form consome esse schema via @hookform/resolvers. O mesmo schema revalida no POST. Uma única fonte de verdade — sem duplicação entre cliente e servidor.

- Backend gera o Zod schema a partir das regras da proposta (documento obrigatório, tipo de doc por nacionalidade, etc.)
- Frontend recebe e aplica o schema via @hookform/resolvers — validação em tempo real no preenchimento
- Backend revalida no POST — zero inconsistência possível

Deve ser planejado desde a modelagem do banco. Refatorar isso depois custa caro.

Modelagem do Banco — Estratégias para Campos Dinâmicos

Formulários com campos arbitrários têm três abordagens clássicas de modelagem, cada uma com trade-offs reais:

| Abordagem | Como funciona | Vantagem | Problema |
|------------------------------|--|--|---|
| Schema rígido + JSONB | Campos fixos na tabela + dados extras em coluna JSONB | Simples de implementar, queries diretas nos campos fixos | Validação e indexação dos campos dinâmicos é manual |
| EAV Puro | Tabela entity-attribute-value: cada campo vira uma linha | Totalmente flexível, sem limite de campos | Queries complexas, performance ruim em leitura, difícil de manter |
| Híbrido (recomendado) | Definição dos campos em tabela própria + respostas em JSONB tipado | Flexível, performático, validação centralizada via Zod | Exige modelagem cuidadosa no início |

A abordagem híbrida funciona assim: uma tabela `form_field_definitions` armazena os metadados de cada campo (tipo, label, obrigatório, ordem, quem preenche). As respostas dos clientes ficam em `form_responses` com uma coluna JSONB `data` validada contra o schema gerado dinamicamente. Isso permite queries eficientes nos metadados, flexibilidade total nas respostas e indexação JSONB quando necessário.

A modelagem completa do banco será detalhada na próxima seção do documento.

5.5 Integração com Google Sheets — Arquitetura Técnica

Esta seção detalha como o AbsolutSport Forms conecta, lê e sincroniza os dados do Sales Log para o PostgreSQL. A implementação é baseada em código já validado em produção e adaptada às necessidades do projeto.

Estrutura de arquivos do módulo de sync

| Arquivo | Responsabilidade |
|---|---|
| <code>src/config/sheetsConfig.js</code> | Autenticação com a API do Google — detecta dev vs prod e escolhe a estratégia correta |
| <code>src/services/sheets.service.js</code> | Lê o Sales Log do Sheets e transforma linhas brutas em objetos estruturados |

| | |
|---|---|
| <code>src/services/salesOrder.service.js</code> | Upsert das linhas no PostgreSQL via Prisma — detecta mudanças por hash |
| <code>src/jobs/sheetsSync.job.js</code> | Cron job que orquestra o ciclo completo a cada 1 minuto e registra o resultado no banco |
| <code>src/routes/sync.routes.js</code> | Endpoint admin para disparar sync manual via painel — sem precisar aguardar o cron |

sheetsConfig.js — Autenticação dev vs produção

O maior ponto de atenção na integração com a Sheets API é a estratégia de autenticação. Em desenvolvimento, usa-se o arquivo `credentials.json` localmente (Service Account baixado do Google Cloud Console). Em produção no Railway, esse arquivo não existe — a solução é serializar o JSON das credenciais como variável de ambiente e parsear em runtime.

| Ambiente | Estratégia | Variável necessária |
|--------------------|--|---|
| Desenvolvimento | <code>keyFile: 'credentials.json'</code> — arquivo local ignorado pelo <code>.gitignore</code> | Nenhuma — arquivo no disco |
| Produção (Railway) | <code>credentials: JSON.parse(process.env.GOOGLE_SERVICE_ACCOUNT_JSON)</code> | <code>GOOGLE_SERVICE_ACCOUNT_JSON</code> — JSON completo da Service Account como string |
| Ambos | <code>scope: spreadsheets.readonly</code> — princípio do menor privilégio, a API nunca escreve na planilha | <code>SHEETS_SPREADSHEET_ID</code> — ID da planilha extraído da URL do Sheets |

sheets.service.js — Leitura e transformação do Sales Log

A leitura usa a Sheets API v4 com os mesmos parâmetros já validados: `UNFORMATTED_VALUE` para preservar tipos numéricos (NUMBER OF ROOMS, NUMBER OF PAX) sem conversão automática, e `FORMATTED_STRING` para datas (CHECK IN, CHECK OUT) mantendo o formato original da planilha. A primeira linha da sheet é tratada como cabeçalho e mapeada para as colunas — isso permite que a Absolut Sport reorganize colunas na planilha sem quebrar a integração.

| Parâmetro da API | Valor | Motivo |
|-----------------------------------|--------------------------------|--|
| <code>range</code> | <code>World Cup</code> | Nome da aba da planilha — toda a sheet é lida de uma vez |
| <code>valueRenderOption</code> | <code>UNFORMATTED_VALUE</code> | Preserva inteiros (2 quartos = 2, não '2.0') e evita formatação monetária |
| <code>dateTimeRenderOption</code> | <code>FORMATTED_STRING</code> | Datas chegam como '6/11/26' — o serviço normaliza para ISO antes de salvar |
| <code>majorDimension</code> | <code>ROWS (padrão)</code> | Cada elemento do array é uma linha — linha 0 é o cabeçalho |

salesOrder.service.js — Upsert com detecção de mudança por hash

A cada ciclo de sync, todas as linhas são lidas. Para evitar reescrever o banco inteiro a cada minuto — o que geraria I/O desnecessário e timestamps de `updated_at` incorretos — cada linha passa por uma comparação de hash SHA-256 antes do upsert.

- Para cada linha lida do Sheets, o serviço calcula SHA-256 do JSON da linha
- Compara com o `raw_hash` salvo no banco para aquela proposta + número de linha
- Se o hash for idêntico, a linha é ignorada (`rows_skipped++`)

- Se diferente ou novo registro, executa `prisma.salesOrder.upsert()` e atualiza `raw_hash` e `last_synced_at`
- O campo `raw_data` (JSONB) armazena a linha completa — garante que colunas não mapeadas explicitamente não sejam perdidas

Resultado prático: uma planilha com 2000 linhas onde apenas 3 mudaram em 1 minuto gera 3 writes no banco — não 2000. O timestamp `updated_at` reflete alterações reais, não ciclos de sync.

sheetsSync.job.js — Cron job e ciclo de execução

O cron job é inicializado junto com o servidor Express e roda de forma assíncrona — uma falha no job nunca derruba o servidor. Cada execução é registrada na tabela `sync_logs` com status, contagens e duração.

| Etapas do ciclo | O que acontece |
|---------------------|--|
| 1. Trigger | node-cron dispara a função <code>runSync()</code> a cada 1 minuto (expressão: <code>* / 1 * * * *</code>) |
| 2. Log de início | Cria registro em <code>sync_logs</code> com status <code>RUNNING</code> e <code>started_at = now()</code> |
| 3. Autenticação | <code>sheetsConfig.getAuthSheets()</code> retorna cliente autenticado — reutiliza token em memória enquanto válido |
| 4. Leitura | <code>sheets.service.fetchSalesLog()</code> chama a API e retorna array de objetos normalizados |
| 5. Upsert | <code>salesOrder.service.upsertSalesOrders()</code> processa cada linha com comparação de hash |
| 6. Log de conclusão | Atualiza <code>sync_logs</code> : status <code>SUCCESS</code> , <code>finished_at</code> , <code>rows_read</code> , <code>rows_upserted</code> , <code>rows_skipped</code> |
| 7. Erro (catch) | Se qualquer etapa falhar: status <code>ERROR</code> , mensagem no campo <code>error</code> , Winston registra — servidor continua rodando |

Prisma — Modelos de banco para o módulo de sync

Dois modelos no `schema.prisma` suportam o módulo inteiro:

| Tabela | Campo-chave | Descrição |
|---------------------------|--|--|
| <code>sales_orders</code> | <code>proposal + line_number (@@unique)</code> | Uma linha por registro do Sales Log. <code>raw_hash</code> detecta mudanças. <code>raw_data</code> (JSONB) preserva colunas não mapeadas. <code>last_synced_at</code> registra quando foi processada pela última vez |
| <code>sync_logs</code> | <code>id (CUID)</code> | Um registro por execução do cron. Campos: <code>started_at</code> , <code>finished_at</code> , <code>rows_read</code> , <code>rows_upserted</code> , <code>rows_skipped</code> , status (<code>RUNNING</code> / <code>SUCCESS</code> / <code>ERROR</code>), <code>error</code> (mensagem em caso de falha) |

Variáveis de ambiente necessárias

| Variável | Ambiente | Valor / Fonte |
|--|---------------|--|
| <code>GOOGLE_SERVICE_ACCOUNT_JSON</code> | Produção | JSON completo da Service Account — gerado no Google Cloud Console, colado como string no Railway |
| <code>SHEETS_SPREADSHEET_ID</code> | Dev + Prod | ID da planilha — extraído da URL: <code>docs.google.com/spreadsheets/d/{ID}/edit</code> |
| <code>credentials.json</code> | Dev (arquivo) | Arquivo baixado do Google Cloud Console — nunca commitado, listado no <code>.gitignore</code> |

Ponto crítico de segurança: a Service Account deve ter acesso somente leitura à planilha (viewer). O escopo da autenticação é `spreadsheets.readonly` — mesmo que a credencial vaze, ninguém consegue escrever na planilha. O `SHEETS_SPREADSHEET_ID` não é segredo, mas deve estar em variável de ambiente para facilitar a troca de planilha sem redeploy.

6. Design System

6.1 Princípios

O design system é a fonte única de verdade visual do AbsolutSport Forms. Todas as decisões de cor, tipografia, espaçamento e componentes foram derivadas da identidade da Absolut Sport — invertendo a lógica de fundo escuro para branco predominante, adequado a uma plataforma operacional.

| Princípio | Aplicação |
|------------------------|---|
| Branco predominante | Fundo #FFFFFF e superfícies #F8F9FA — clareza para formulários e dashboards |
| Azul como ação | #155F97 para todos os CTAs, links e estados de foco — herança direta da Absolut Sport |
| Verde como confirmação | #16B85F para badges de sucesso, status concluído e ações positivas |
| Escuro só no nav | #041628 restrito ao header global — mesmo tom do site da agência |
| Tipografia Barlow | Família da Absolut Sport — Barlow Condensed para títulos, Barlow para body |

6.2 Paleta de Cores

Três escalas principais + cores semânticas. Tokens nomeados para uso direto no `tailwind.config.js`.

Primary — Azul (herda Absolut Sport CTA)

| Token | Hex | Uso |
|--------------------------|---------|--|
| <code>primary.50</code> | #EBF3FB | Backgrounds de cards info, hover states suaves |
| <code>primary.100</code> | #C8DFF5 | Borders de elementos informativos, chips |
| <code>primary.200</code> | #91BFE9 | Ícones secundários, separadores coloridos |
| <code>primary.5</code> | #155F97 | CTA principal, links, bordas de foco — BASE |

| | | |
|-------------|---------|--|
| 00 | | |
| primary.700 | #0D3B60 | Hover de botões primários, texto de links |
| primary.900 | #041628 | Background do nav global — herda Absolut Sport |

Accent — Verde (herda Absolut Sport Badge)

| Token | Hex | Uso |
|------------|---------|---|
| accent.50 | #E8F9EF | Background de status concluído, alertas de sucesso |
| accent.500 | #16B85F | Badge concluído, botão confirmar, ícone de sucesso — BASE |
| accent.700 | #0D7A3E | Texto de status concluído, hover de ações positivas |

Neutral — Escala de cinzas

| Token | Hex | Uso |
|-------------|---------|---|
| neutral.0 | #FFFFFF | Background de página e cards — tom predominante |
| neutral.500 | #F8F9FA | Surface alternativa, rows de tabela par |
| neutral.100 | #F1F3F5 | Surface alt, hover de itens de lista |
| neutral.200 | #E9ECEF | Borders leves, separadores |
| neutral.300 | #DEE2E6 | Borders padrão de inputs |
| neutral.400 | #ADB5BD | Placeholder de inputs, ícones inativos |
| neutral.500 | #6C757D | Texto auxiliar, labels secundários |
| neutral.700 | #343A40 | Texto secundário, labels de campo |
| neutral.900 | #0D1117 | Texto primário — títulos e conteúdo principal |

Semânticas

| Nome | Hex | Uso |
|---------|---------|---|
| success | #16B85F | Formulário concluído, validação aprovada, ação positiva |
| warning | #F59E0B | Prazo próximo, dados incompletos, atenção necessária |
| error | #EF4444 | Validação falhou, prazo expirado, erro de sistema |
| info | #155F97 | Informações contextuais, dicas, estados neutros |

6.3 Tipografia

| Família | Token | Uso |
|------------------|--------------|--|
| Barlow Condensed | font-display | Títulos de página, labels de status, headings H1–H2, nav items |
| Barlow | font-body | Body text, inputs, parágrafos, labels de campo, texto geral |
| JetBrains Mono | font-mono | Códigos, IDs de formulário, tokens, valores de configuração |

Escala tipográfica

| Token | Size | Line Height | Uso |
|-----------|------|-------------|---|
| text-xs | 12px | 16px | Labels, captions, timestamps de auditoria |
| text-sm | 14px | 20px | Body small, helper text, conteúdo de tabelas |
| text-base | 16px | 24px | Body padrão, inputs, parágrafos de formulário |
| text-lg | 18px | 28px | Body grande, texto de intro de eventos |
| text-xl | 20px | 28px | Card titles, subtítulos de seção |
| text-2xl | 24px | 32px | Títulos de página, modal headers |
| text-3xl | 30px | 36px | Page headers — Barlow Condensed |
| text-4xl | 36px | 40px | Display hero, nome do evento — Barlow Condensed |

6.4 Espaçamento & Border Radius

| Token Tailwind | Valor | Uso típico |
|----------------|-------|------------------------------------|
| p-1 / gap-1 | 4px | Espaço mínimo entre ícone e texto |
| p-2 / gap-2 | 8px | Padding interno de badges e chips |
| p-3 / gap-3 | 12px | Padding de inputs e botões small |
| p-4 / gap-4 | 16px | Padding padrão de cards e modals |
| p-5 / gap-5 | 20px | Padding generoso de painéis |
| p-6 / gap-6 | 24px | Seções de formulário |
| p-8 / gap-8 | 32px | Separação entre blocos de conteúdo |

| | | |
|------|------|----------------------------------|
| p-10 | 40px | Padding de page sections |
| p-12 | 48px | Espaço entre seções maiores |
| p-16 | 64px | Hero sections, top-level spacing |

Border Radius

| Token | Valor | Componente |
|--------------|--------|---|
| rounded-sm | 4px | Tags, badges pequenos, chips de tipo de campo |
| rounded-md | 8px | Inputs, botões, selects, cards secundários |
| rounded-lg | 12px | Cards principais, painéis, blocos de formulário |
| rounded-xl | 16px | Modals, sidebars, drawers |
| rounded-full | 9999px | Status pills, avatares, badges de contagem |

6.5 Sombras & Elevação

| Token | Valor CSS | Uso |
|----------------|---------------------------------|--|
| shadow-sm | 0 1px 3px rgba(0,0,0,0.06) | Inputs, elementos inline, separadores suaves |
| shadow-md | 0 4px 12px rgba(0,0,0,0.08) | Cards de evento, cards de stats do dashboard |
| shadow-lg | 0 8px 24px rgba(0,0,0,0.10) | Modals, dropdowns, menus de contexto |
| shadow-xl | 0 16px 40px rgba(0,0,0,0.12) | Toasts, popovers de alto nível |
| shadow-primary | 0 4px 14px rgba(21,95,151,0.25) | Botões primários com foco, CTA destacado |
| shadow-accent | 0 4px 14px rgba(22,184,95,0.25) | Estados de sucesso, botões de confirmação |

6.6 Componentes — Especificação

Buttons

| Variante | Background | Texto | Uso |
|-------------|------------|---------|--|
| Primary | #155F97 | | Ação principal — Salvar, Publicar, Enviar formulário |
| Success | #16B85F | #FFFFFF | Confirmações positivas — Concluir, Aprovar |
| Outline | #FFFFFF | #155F97 | Ação secundária — Cancelar, Voltar |
| Destructive | #FEF2F2 | #EF4444 | Ações irreversíveis — Excluir campo, Remover cliente |
| Disabled | #E9ECEF | #ADB5BD | Estado inativo — qualquer ação indisponível |

Status Badges — Formulários

| Label | Background | Cor do texto | Quando usar |
|----------------|------------|--------------|--|
| Concluído | #E8F9EF | #0D7A3E | Formulário 100% preenchido dentro do prazo |
| Em andamento | #EBF3FB | #114D7C | Formulário acessado, preenchimento iniciado |
| Não iniciado | #F1F3F5 | #343A40 | Convite enviado, formulário ainda não acessado |
| Prazo expirado | #FEF2F2 | #DC2626 | Prazo de edição ultrapassado sem conclusão |
| Ativo | #E8F9EF | #0D7A3E | Projeto/evento com formulários abertos |
| Rascunho | #FFF8E1 | #92400E | Formulário criado mas não publicado |
| Encerrado | #F1F3F5 | #343A40 | Projeto encerrado, edição bloqueada |

Form Inputs — Estados

| Estado | Border | Background | Descrição |
|----------|---------|------------|--|
| Default | #DEE2E6 | #FFFFFF | Input em repouso — borda neutra, sem ação do usuário |
| Focus | #155F97 | #FFFFFF | Input ativo — borda azul primária + ring rgba(21,95,151,0.15) |
| Success | #16B85F | #F8FFF9 | Validação aprovada — borda verde + fundo verde muito suave |
| Error | #EF4444 | #FFF8F8 | Validação falhou — borda vermelha + fundo vermelho muito suave |
| Disabled | #E9ECEF | #F8F9FA | Campo bloqueado — opacidade reduzida, cursor not-allowed |
| ReadOnly | #E9ECEF | #F1F3F5 | Dado visível mas não editável após prazo |

6.7 Tailwind Config — Tokens

Configuração completa para colar em tailwind.config.js — todos os tokens do design system mapeados:

```
// tailwind.config.js
import { fontFamily } from 'tailwindcss/defaultTheme';

export default {
  content: ['./index.html', './src/**/*.js,ts,jsx,tsx'],
  theme: {
    extend: {
      colors: {
        primary: { 50: '#EBF3FB', 100: '#C8DFF5', 200: '#91BFE9',
                    500: '#155F97', 700: '#0D3B60', 900: '#041628' },
        accent: { 50: '#E8F9EF', 500: '#16B85F', 700: '#0D7A3E' },
```

```

    neutral: { 0:'#FFFFFF', 50:'#F8F9FA', 100:'#F1F3F5',
              200:'#E9ECEF', 300:'#DEE2E6', 400:'#ADB5BD',
              500:'#6C757D', 700:'#343A40', 900:'#0D1117' },
  },
  fontFamily: {
    display: ["'Barlow Condensed'", ...fontFamily.sans],
    body:    ["'Barlow'",          ...fontFamily.sans],
    mono:    ["'JetBrains Mono'",  ...fontFamily.mono],
  },
  boxShadow: {
    'sm': '0 1px 3px rgba(0,0,0,0.06)',
    'md': '0 4px 12px rgba(0,0,0,0.08)',
    'lg': '0 8px 24px rgba(0,0,0,0.10)',
    'xl': '0 16px 40px rgba(0,0,0,0.12)',
    'primary': '0 4px 14px rgba(21,95,151,0.25)',
    'accent': '0 4px 14px rgba(22,184,95,0.25)',
  },
  borderRadius: {
    sm:'4px', md:'8px', lg:'12px', xl:'16px', full:'9999px'
  },
},
},
},
plugins: [],
};

```

7. Infraestrutura — Docker, Prisma e Railway

6.1 Visão Geral

O projeto roda em containers Docker desde o desenvolvimento local. O mesmo docker-compose que sobe o ambiente local é a base para o deploy em produção no Railway, garantindo paridade total entre os ambientes.

6.2 Docker — Estrutura de Containers

O projeto utiliza três containers orquestrados via Docker Compose:

| Container | Imagem base | Responsabilidade |
|-----------|--|---|
| api | node:18-alpine | Backend Express — porta 3000 |
| web | node:18-alpine (build) + nginx:alpine (serve) | Frontend React — porta 80 |
| db | postgres:15-alpine | Banco de dados local (dev apenas — Railway injeta o Postgres em prod) |

docker-compose.yml (desenvolvimento local)

O arquivo abaixo sobe os três serviços localmente com hot-reload no backend (nodemon) e HMR no frontend (Vite):

```
services:
```

Arquivo completo: `docker-compose.yml` (ver repositório)

6.3 Prisma ORM

O Prisma substitui o driver pg direto. O `schema.prisma` é a única fonte de verdade do banco — nenhuma migration é escrita à mão.

- `prisma migrate dev` — aplica migrations em desenvolvimento, gera o Prisma Client atualizado
- `prisma migrate deploy` — aplica migrations em produção (Railway executa automaticamente no deploy)
- `prisma studio` — UI visual para explorar o banco durante o desenvolvimento
- `prisma generate` — regenera o client typado após alterações no schema

Localização do schema: `prisma/schema.prisma` na raiz do backend.

6.4 Railway — Deploy em Produção

O Railway foi escolhido como plataforma de deploy do MVP por oferecer o menor atrito operacional sem abrir mão de funcionalidades essenciais:

| Recurso Railway | Benefício para o projeto |
|------------------------------------|---|
| Deploy via GitHub | Push na branch main dispara o build e deploy automaticamente |
| Suporte nativo a Docker | Lê o Dockerfile do repositório — zero configuração adicional |
| PostgreSQL managed | Banco provisionado pelo Railway com backup automático |
| Variáveis de ambiente | <code>DATABASE_URL</code> injetada automaticamente no container via Railway |
| <code>prisma migrate deploy</code> | Configurado no start command — migrations rodam antes do servidor subir |
| Domínios automáticos | URL pública gerada imediatamente, sem configuração de DNS |
| Escalabilidade | Planos escaláveis conforme o projeto cresce — sem replatforming |

Variáveis de ambiente obrigatórias no Railway

- `DATABASE_URL` — injetada automaticamente pelo Railway ao provisionar o PostgreSQL
- `JWT_SECRET` — segredo para assinatura dos tokens
- `RESEND_API_KEY` — chave da API de e-mail
- `NODE_ENV=production`

- VITE_API_URL — URL do backend (para o container do frontend)

Start command do backend (Railway)

```
npx prisma migrate deploy && node dist/index.js
```

Isso garante que toda migration pendente é aplicada antes do servidor aceitar tráfego.

7. Fluxos Principais

6.1 Fluxo do Admin — Criação e Envio

1. Admin loga na plataforma
2. Cria um novo Projeto (nome, data, status)
3. Dentro do projeto, cria um Formulário
4. Adiciona campos (tipo, obrigatoriedade, visibilidade, quem preenche)
5. Define prazo limite para edições
6. Associa clientes ao formulário (via e-mail ou importação)
7. Envia os convites — cada cliente recebe um link único
8. Acompanha o dashboard em tempo real

6.2 Fluxo do Cliente — Preenchimento

1. Cliente recebe e-mail com link de acesso
 2. Cria senha (primeiro acesso) ou loga com credenciais existentes
 3. Visualiza o evento e o formulário pendente
 4. Seus dados da ficha base são pré-preenchidos automaticamente
 5. Revisa e completa os dados restantes
 6. Informa quantidade de acompanhantes e preenche os dados de cada um
 7. Salva e finaliza o formulário
 8. Pode retornar e editar até o prazo limite
-

7. Fora do Escopo do MVP

Os itens abaixo foram intencionalmente excluídos do MVP para garantir velocidade de entrega. Serão revisitados na V2 com base no feedback real de uso:

- Modelo B2B — fluxo para agências parceiras com planilhas/portal próprio
- Auditoria avançada — IP, dispositivo, tempo de preenchimento, histórico campo a campo

- Integração com PMS, CRM ou ERPs externos
- Notificações automáticas por WhatsApp
- Assinatura digital de documentos
- Multi-idioma (i18n)

8. Fluxo de Captação de Dados — Google Sheets → Formulário Dinâmico

8.1 Visão Geral do Processo

A captação de dados dos clientes é o processo central que viabiliza a operação do evento. Todo o ciclo começa em uma planilha Google Sheets (o Sales Log), avança pelo sistema para gerar formulários dinâmicos personalizados por venda, e termina com os dados dos passageiros correlacionados aos hotéis e ingressos para envio aos fornecedores.

Hoje esse processo é manual — a equipe monta planilhas individuais por cliente (como o modelo CVC) e as preenche durante a captação. O AbsolutSport Forms automatiza e estrutura exatamente isso.

| Etapa | Descrição |
|------------------------|--|
| 1. Importação | Sistema lê o Sales Log do Google Sheets e sincroniza os dados de cada venda para o banco |
| 2. Disparo | Admin seleciona uma venda confirmada e dispara o formulário de captação para o cliente |
| 3. Formulário dinâmico | Sistema gera automaticamente os campos corretos com base na realidade da venda |
| 4. Preenchimento | Cliente preenche os dados pessoais de cada passageiro no portal |
| 5. Correlação | Sistema associa cada passageiro ao seu quarto, tipo, hotel, jogo e ingresso |
| 6. Exportação | Admin extrai fichas operacionais por fornecedor: hotel, companhia aérea, seguradora, transporte e FIFA |

8.2 Fonte Primária — Google Sheets (Sales Log)

O Sales Log é a planilha de controle de vendas da Absolut Sport. Cada linha representa uma venda ou linha de item de uma venda. O sistema importa e mantém sincronizado esse conteúdo na tabela `sales_orders` do banco de dados.

Campos consumidos do Sales Log

| Coluna no Sheets | Tipo | Uso no sistema |
|------------------|------|----------------|
|------------------|------|----------------|

| | | |
|-------------------|-------|---|
| PROPOSAL | ID | Identificador único da venda — chave primária da ordem |
| # | Int | Número da linha dentro da proposta (uma proposta pode ter múltiplas linhas) |
| STATUS | Enum | Filtra vendas elegíveis para captação: CONFIRMED, PROCESSING |
| CLIENT | Text | Nome do cliente comprador — titular da venda |
| EMAIL | Email | E-mail para disparo do convite de captação |
| CELL_PHONE | Phone | Telefone de contato do titular |
| GAME_DETAILS | Text | Jogo que o pacote contempla (ex: M7, Final) |
| CITY | Text | Cidade-sede do jogo (New York, Miami, etc.) |
| TICKET_CATEGORY | Text | Categoria do ingresso (Cat 1, Cat 3, FIFA Pavilion...) |
| NUMBER_OF_TICKETS | Int | Quantidade de ingressos — define quantos passageiros têm direito a jogo |
| HOTEL | Text | Nome do hotel contratado |
| NUMBER_OF_ROOMS | Int | Quantidade de quartos reservados — eixo principal da geração do formulário |
| NUMBER_OF_PAX | Int | Total de passageiros que se hospedarão — distribuídos pelos quartos |
| ROOM_TYPE | Enum | Tipo de quarto: SGL, DBL Twin, DBL Casal, QQ, KING, TPL, QDP |
| CHECK_IN | Date | Data de entrada no hotel |
| CHECK_OUT | Date | Data de saída do hotel |
| PRODUCT | Text | Tipo do pacote: Ticket Only, Hotel + Ticket, Package, etc. |
| COMPANY | Text | Escritório responsável pela venda (BRA, GER, US...) |
| SELLER | Text | Vendedor responsável — notificado em eventos de captação |

8.3 Lógica de Geração do Formulário Dinâmico

O coração do processo: a partir dos dados da venda já importados, o sistema sabe exatamente quantos passageiros existem, como estão distribuídos nos quartos e qual o tipo de cada quarto. Com essa informação, gera automaticamente os campos de coleta necessários — sem que o Admin precise configurar nada manualmente.

Capacidade por tipo de quarto

| ROOM TYPE | Pax máx. | Ocupação esperada | Observação |
|-----------|----------|---------------------|--|
| SGL | 1 | 1 pessoa / quarto | Quarto individual — 1 ficha por quarto |
| DBL Twin | 2 | 2 pessoas / quarto | Dois beliches ou camas separadas |
| DBL Casal | 2 | 2 pessoas / quarto | Casal — cama de casal |
| QQ | 2 | 2 pessoas / quarto | Queen/Queen — mais comum na base |
| KING | 1–2 | 1 ou 2 pax / quarto | Pode ser uso individual ou duplo |
| TPL | 3 | 3 pessoas / quarto | Triple — três fichas por quarto |
| QDP | 4 | 4 pessoas / quarto | Quadruple — quatro fichas por quarto |

Algoritmo de geração — passo a passo

Dado: uma linha (ou conjunto de linhas) do Sales Log com ROOMS, PAX e ROOM TYPE definidos, o sistema executa:

1. Calcula $\text{pax_por_quarto} = \text{NUMBER OF PAX} \div \text{NUMBER OF ROOMS}$
2. Para cada quarto (1..N), cria um QuartoSlot com: número do quarto, tipo, hotel, check-in, check-out
3. Para cada QuartoSlot, abre pax_por_quarto formulários de passageiro
4. Cada formulário de passageiro é vinculado ao seu QuartoSlot e ao ingresso correspondente
5. O primeiro passageiro de cada quarto recebe status de 'titular do quarto' (room_holder)

O cliente preenchedor vê a interface organizada por quarto: cada bloco representa um quarto com seu tipo e datas, e dentro dele os formulários de cada hóspede. Não é uma lista plana — é uma hierarquia visual que reflete a realidade da reserva.

8.4 Exemplos Reais da Base de Dados

Os três exemplos abaixo são propostas reais do Sales Log e ilustram os níveis de complexidade que o formulário deve suportar:

Exemplo 1 — Simples: Proposta 20250602 (BTG Pactual)

| Campo | Valor |
|-----------------|--|
| PRODUCT | Hotel + Ticket |
| GAME | M7 — New York |
| HOTEL | Distrikt Hotel New York City (Tapestry/Hilton) |
| ROOM TYPE | KING |
| NUMBER OF ROOMS | 2 |
| NUMBER OF PAX | 2 |
| CHECK IN/OUT | 11/06/2026 → 14/06/2026 |

Resultado gerado: 2 quartos KING × 1 pax cada = 2 fichas individuais. Cada passageiro em quarto separado. O formulário abre 2 blocos, cada bloco com 1 ficha.

Exemplo 2 — Médio: Proposta 20260040 (Personal Brasil)

| Linha | Rooms | Pax | Room Type | Fichas geradas |
|--------------|-----------|-----------|-----------|----------------------------------|
| Linha 1 | 5 | 5 | SGL | 5 quartos × 1 ficha = 5 fichas |
| Linha 2 | 5 | 10 | DBL Twin | 5 quartos × 2 fichas = 10 fichas |
| TOTAL | 10 | 15 | — | 15 fichas de passageiro |

O hotel Hilton NY Times Square recebe uma reserva mista: 5 individuais + 5 duplos. O formulário gera 10 blocos de quarto organizados por tipo, com 15 fichas no total para o cliente preencher.

Exemplo 3 — Complexo: Proposta 20250445 (multi-jogo, multi-tipo)

| Jogo | Hotel | Rooms | Pax | Tipo | Fichas |
|------|------------------|-------|-----|------|--------|
| M101 | Hilton Arlington | 14 | 28 | QQ | 28 |

| | | | | | |
|-------|---------------------|-----|-----|------|-------------|
| M101 | Hilton Arlington | 3 | 6 | KING | 6 |
| M101 | Hilton Arlington | 2 | 6 | TPL | 6 |
| M101 | Hilton Arlington | 1 | 4 | QDP | 4 |
| Final | Canopy Jersey City | 17 | 34 | QQ | 34 |
| Final | Canopy Jersey City | 3 | 6 | KING | 6 |
| Final | Canopy Jersey City | 2 | 6 | TPL | 6 |
| Final | Canopy Jersey City | 1 | 4 | QDP | 4 |
| M102 | Hyatt Place Atlanta | 20 | 40 | QQ | 40 |
| ... | ... +8 linhas | ... | ... | ... | ... |
| TOTAL | — | — | — | — | 224+ fichas |

Este caso representa um grande pacote de agência parceira com múltiplos jogos, múltiplos hotéis e múltiplos tipos de quarto em uma mesma venda. O formulário organiza tudo em seções por jogo → por hotel → por quarto → por passageiro.

8.5 Dados Coletados por Passageiro

Baseado no modelo operacional atual (planilha CVC), cada passageiro exige os seguintes dados para que a empresa possa operacionalizar a hospedagem e emitir o ingresso:

| Campo | Obrigatório | Finalidade operacional |
|-----------------------------|-------------|--|
| Nome completo (Name + Last) | Sim | Reserva no hotel, emissão do ingresso FIFA |
| Nacionalidade | Sim | Requisito do hotel e da FIFA para registro |
| Gênero | Sim | Controle de distribuição de quartos |
| CPF / RG / Passaporte / DNI | Sim | Identificação legal — tipo varia por nacionalidade |
| País emissor do documento | Sim | Validação do documento de identidade |
| Validade do documento | Sim | Hotel e FIFA exigem documento válido |
| Data de nascimento | Sim | Controle etário — menores em pacotes específicos |
| Time que torce | Não | Controle interno — distribuição de ingressos por setor |
| Telefone | Sim | Contato em emergências no destino |
| E-mail | Sim | Comunicações da agência com o passageiro |

8.6 Modelo de Correlação — O que o sistema entrega ao final

A saída do processo de captação é uma ficha operacional consolidada que correlaciona cada passageiro com todos os elementos da sua compra. Esse é o dado que a agência envia a cada fornecedor: ao hotel para confirmar a hospedagem, à companhia aérea para emissão do bilhete, à seguradora para emissão da apólice, à empresa de transporte para confirmação do transfer e à FIFA para validação do ingresso.

Inspirado diretamente no modelo planilha CVC, cada registro da saída conterá:

| Dado | Origem |
|-------------------------------|---|
| PROPOSAL | Sales Log — importado do Sheets |
| SELLER | Sales Log — importado do Sheets |
| CLIENT | Sales Log — importado do Sheets |
| TICKET CAT / NUM TICKET | Sales Log — importado do Sheets |
| HOTEL | Sales Log — importado do Sheets |
| ROOM NUMBER | Gerado pelo sistema — número sequencial do quarto no pacote |
| PAX NUMBER | Gerado pelo sistema — número sequencial do passageiro |
| ROOM TYPE | Sales Log — importado do Sheets |
| CHECK IN / CHECK OUT | Sales Log — importado do Sheets |
| NAME / LAST NAME | Coletado do passageiro via formulário |
| NATIONALITY | Coletado do passageiro via formulário |
| GENDER | Coletado do passageiro via formulário |
| CPF / PASSAPORT / RG / DNI | Coletado do passageiro via formulário |
| ISSUING COUNTRY | Coletado do passageiro via formulário |
| EXPIRY DATE | Coletado do passageiro via formulário |
| BIRTH DATE | Coletado do passageiro via formulário |
| TEAM | Coletado do passageiro via formulário |
| PHONE / EMAIL | Coletado do passageiro via formulário |
| STADIUM TRANSFER | Sales Log — campo do pacote |
| TICKET STATUS / LINK | Preenchido pelo operador interno após emissão pela FIFA |
| HOTEL CONFIRMATION | Preenchido pelo operador interno após confirmação do hotel |
| FLIGHT LOCATOR | Preenchido pelo operador interno — localizador da reserva aérea |
| AIRLINE / FLIGHT NUMBER | Preenchido pelo operador interno — companhia e número do voo |
| INSURANCE NUMBER | Preenchido pelo operador interno — número da apólice de seguro |
| INSURANCE PROVIDER | Preenchido pelo operador interno — nome da seguradora |
| TRANSFER REFERENCE | Preenchido pelo operador interno — confirmação do serviço de transporte |
| TRANSFER TYPE | Sales Log — campo do pacote (aeroporto, stadium, city tour etc.) |

[AZUL] Azul = dado do Sales Log
(já temos)

[VERDE] Verde = captado do
passageiro (precisamos coletar)

[AMARELO] Amarelo = preenchido
pelo operador pós-captação

8.7 Casos Especiais e Regras de Negócio

| Situação | Regra |
|------------------------------------|--|
| Ticket Only (sem hotel) | Formulário não gera blocos de quarto. Coleta apenas dados pessoais dos passageiros para emissão de ingresso |
| Proposta com múltiplas linhas | Cada linha do Sales Log com hotel gera seu bloco independente. Um cliente pode ter M7 + Final com hotéis diferentes — o formulário organiza por jogo |
| Pax ímpar em quartos duplos | Ex: 5 pax em 3 QQ — o sistema cria 2 quartos com 2 pax e 1 quarto com 1 pax. O quarto 'incompleto' é sinalizado |
| KING com 1 ou 2 pax | KING é ambíguo — pode ser uso individual ou duplo. O sistema respeita o valor de NUMBER OF PAX para calcular a distribuição real |
| Pax sem hospedagem (Transfer only) | Passageiros em linhas sem hotel recebem fichas de dados pessoais + transfer e/ou ingresso, sem vínculo de quarto. Os demais produtos (voo, seguro) são correlacionados normalmente |
| Repasse parcial de dados | O cliente pode salvar e retornar. O sistema mantém o progresso por passageiro e por quarto — não precisa concluir tudo de uma vez |
| Prazo expirado | Após o deadline definido pelo Admin, todas as fichas ficam somente leitura. O Admin pode reabrir individualmente |

10. Fluxo Operacional — Visão Geral

O diagrama abaixo sintetiza o ciclo completo do AbsolutSport Forms: desde a importação automática dos dados do Sales Log até a entrega das fichas operacionais a cada fornecedor — hotel, companhia aérea, seguradora, empresa de transporte e FIFA. Cada cor representa um ator ou camada do sistema, conforme a legenda.

AbsolutSport Forms

Fluxo operacional completo — do Sales Log à entrega a cada fornecedor

Log de Auditoria

— todas as ações de todos os atores são registradas



