**Especificação Técnica e Guia de Implementação: Dashboards de Análise de Lançamentos**

**1. Visão Geral do Projeto**

**Objetivo:** Desenvolver um conjunto de dashboards analíticos para monitorizar e analisar a performance de lançamentos de cursos online. O sistema deve fornecer uma visão clara sobre a aquisição de leads, canais de marketing, performance de campanhas e o perfil dos alunos.

**Público-alvo deste documento:** Desenvolvedores (ex: na plataforma Lovable AI) encarregados de manter, evoluir ou recriar esta solução.

**2. Arquitetura da Solução Adotada**

Para garantir a **consistência dos dados**, **performance** e **manutenção simplificada**, foi implementada uma arquitetura onde a lógica de negócio complexa é centralizada no backend (Banco de Dados Supabase) e consumida pelo frontend (Next.js).

**Princípio Fundamental: O Backend como "Fonte Única da Verdade"**

* **Lógica no Backend:** Todas as regras de negócio, como a classificação de tráfego (Pago, Orgânico, Não Traqueado) e a agregação de dados (cálculos diários, totais por campanha), são implementadas diretamente no Supabase através de **Funções de Banco de Dados (RPC)**.
* **Frontend Responsável pela Exibição:** O frontend (a aplicação Next.js) é responsável apenas por:
  1. Chamar as funções RPC apropriadas, passando os filtros necessários (ex: ID do lançamento).
  2. Receber os dados já calculados e agregados.
  3. Exibir esses dados em KPIs, tabelas e gráficos.

Esta abordagem evita a duplicação de lógica, previne inconsistências entre diferentes dashboards e otimiza a performance, pois os cálculos pesados são executados pelo servidor de banco de dados, que é otimizado para isso.

**3. Especificações do Backend (Funções no Supabase)**

As seguintes funções SQL (RPC) foram criadas e devem ser a base para qualquer nova implementação.

**3.1. Classificação e Detalhamento de Canais**

* **Função:** get\_leads\_para\_detalhamento(launch\_id\_param, p\_limit, p\_offset)
* **Objetivo:** Retornar uma lista de todos os leads de um lançamento, com uma coluna adicional tipo\_trafego que já classifica cada lead.
* **Lógica de Classificação tipo\_trafego:**
  + **'organic'**: Onde LOWER(TRIM(utm\_source)) é exatamente 'organic'.
  + **'untracked'**: Onde utm\_source é nulo, vazio, ou contém os placeholders '{:utm\_source}', '{{site\_source\_name}}', etc.
  + **'paid'**: Todos os outros casos.
* **Parâmetros:**
  + launch\_id\_param: O ID do lançamento a ser analisado.
  + p\_limit, p\_offset: Para suportar a paginação e carregar um grande volume de leads.
* **Uso:** Utilizada pela página **"Detalhamento dos Canais"**.

**3.2. Resumo Diário e KPIs Gerais**

* **Função:** get\_resumo\_diario\_dashboard(launch\_id\_param)
* **Objetivo:** Calcular e retornar, numa única chamada, todos os dados necessários para a página "Resumo Diário".
* **Lógica Interna:**
  1. Utiliza a mesma lógica de classificação de tráfego descrita no ponto 3.1.
  2. Calcula os KPIs gerais (Total de Inscrições, Total de Check-ins, e totais por tipo de tráfego).
  3. Agrega todos os leads por dia, calculando as inscrições, check-ins e a quebra por tipo de tráfego para cada dia.
  4. Retorna um único objeto JSON contendo uma chave kpis com os totais e uma chave dailyData com o array de dados diários.
* **Uso:** Utilizada pela página **"Resumo Diário do Lançamento"**.

**3.3. Análise de Campanhas e Criativos**

* **Função:** get\_kpis\_campanha(launch\_id\_param)
  + **Objetivo:** Calcular os KPIs específicos desta página (Totais Gerais e Totais de Tráfego Pago).
* **Função:** get\_filtros\_campanha(launch\_id\_param)
  + **Objetivo:** Retornar uma lista de todos os utm\_campaign únicos e, para cada um, uma lista de utm\_term associados, para popular os filtros da página.
* **Função:** get\_performance\_campanha(launch\_id\_param, campaign\_filter, term\_filter)
  + **Objetivo:** Retornar os dados agregados de performance (inscritos e check-ins) agrupados por utm\_campaign e utm\_term, com base nos filtros selecionados pelo utilizador.
* **Uso:** Todas utilizadas pela página **"Análise de Campanhas e Criativos"**.

**4. Especificações do Frontend (Páginas e Componentes)**

**4.1. Princípios Gerais**

* **Seleção de Lançamento:** Todas as páginas devem ter um seletor de lançamento que, ao ser alterado, dispara uma nova busca de dados.
* **Estado de Carregamento (isLoading):** É crucial mostrar um feedback visual (ex: um spinner) enquanto os dados estão a ser buscados, para informar o utilizador.
* **Proteção Contra "Race Conditions":** Para páginas que buscam um grande volume de dados em lotes (paginação), é **mandatório** implementar um mecanismo para cancelar ou ignorar buscas antigas. A abordagem recomendada é usar a função de limpeza do useEffect para invalidar os resultados de chamadas que já não são as mais recentes.

**4.2. Páginas Desenvolvidas**

* **Detalhamento dos Canais:**
  + **Funcionalidade:** Exibe KPIs gerais e KPIs específicos para um tipo de tráfego (Pago, Orgânico, Não Traqueado). Mostra uma tabela detalhada com os leads do tráfego selecionado.
  + **Fonte de Dados:** Chama a função get\_leads\_para\_detalhamento em loop (com paginação) e faz os cálculos de KPI no lado do cliente (useMemo) após receber todos os dados.
* **Resumo Diário do Lançamento:**
  + **Funcionalidade:** Exibe KPIs gerais e por tipo de tráfego. Mostra uma tabela com a evolução diária de leads, check-ins e a quebra por tráfego. Apresenta dois gráficos com a evolução diária.
  + **Fonte de Dados:** Faz uma única chamada à função get\_resumo\_diario\_dashboard e renderiza os dados pré-calculados que ela retorna.
* **Análise de Campanhas e Criativos:**
  + **Funcionalidade:** Exibe KPIs gerais e de tráfego pago. Apresenta filtros em cascata para utm\_campaign e utm\_term. Uma tabela mostra os resultados da pesquisa, agrupando os termos sob a sua campanha.
  + **Fonte de Dados:** Chama get\_kpis\_campanha e get\_filtros\_campanha no carregamento inicial. Ao clicar em "Pesquisar", chama get\_performance\_campanha com os filtros selecionados.

**5. Funcionalidade Adicional: Gerador de Diplomas**

* **Objetivo:** Permitir a geração de diplomas personalizados para os alunos, individualmente ou em lote.
* **Tecnologia:** HTML, CSS e JavaScript puro, sem dependência de um framework de frontend.
* **Implementação:**
  1. **HTML Canvas:** Um elemento <canvas> é usado como uma "tela de desenho" invisível.
  2. **Imagem de Fundo:** A imagem do diploma é carregada e desenhada no canvas.
  3. **Fonte Personalizada:** Uma fonte manuscrita (ex: 'Great Vibes' do Google Fonts) é carregada na página.
  4. **Desenho do Texto:** A função ctx.fillText() do canvas é usada para escrever o nome do aluno sobre a imagem, com a fonte, tamanho e cor personalizados.
  5. **Importação de CSV:** A biblioteca PapaParse é usada para ler um ficheiro de texto carregado pelo utilizador, extraindo a lista de nomes.
  6. **Download:** A função canvas.toDataURL('image/png') converte o resultado final numa imagem PNG, que é então descarregada pelo navegador.

Esta abordagem é autónoma e pode ser facilmente integrada em qualquer sistema web.