Aqui está o plano de desenvolvimento para o Checkout em 3 Passos com Mercado Pago.

✅ Checklist de Desenvolvimento: Checkout e Pagamentos

1. Backend: Banco de Dados e Configuração

- Revisar/Criar Tabelas no Banco:
    - Criar uma nova migração SQL para garantir que as tabelas orders, order_items, e inventory_movements existam e estejam corretas.
    - A tabela orders deve suportar status como awaiting_payment, paid, canceled.
    - A tabela inventory_movements deve suportar tipos como reserve, confirm, cancel_reserve.
- Configurar Variáveis de Ambiente:
    - Adicionar as chaves do Mercado Pago (MP_ACCESS_TOKEN) e do Supabase (SERVICE_ROLE_KEY) ao seu arquivo de ambiente.
- Instalar SDK do Mercado Pago:
    - Executar pnpm install mercadopago para adicionar a biblioteca oficial ao projeto.

2. Backend: Rotas de API

- API 1: Criar Pedido (`POST /api/checkout/criar-pedido`)
    - Criar o arquivo da rota: src/app/api/checkout/criar-pedido/route.ts.
    - Definir um schema Zod para validar os dados do carrinho e do cliente recebidos.
    - Implementar a lógica para:
        - Criar um registro na tabela orders com status awaiting_payment.
        - Criar os registros correspondentes em order_items para cada produto.
        - Reservar o estoque: Inserir registros em inventory_movements com o tipo reserve.
        - Retornar o order_id gerado.
- API 2: Criar Preferência de Pagamento (`POST /api/payments/mp/preferencia`)
    - Criar o arquivo da rota: src/app/api/payments/mp/preferencia/route.ts.
    - Definir um schema Zod para validar o order_id recebido.
    - Implementar a lógica para:
        - Buscar os dados do pedido no banco usando o order_id.

        - Configurar e chamar o SDK do Mercado Pago para criar uma "preferência de pagamento".
        - Retornar os dados necessários para o frontend (como init_point para redirecionamento ou qr_code_base64 para Pix).
- API 3: Webhook de Notificação (`POST /api/payments/webhook`)
    - Criar o arquivo da rota: src/app/api/payments/webhook/route.ts.
    - Implementar Segurança e Confiabilidade:
        - Validar a assinatura do webhook para garantir que a chamada veio do Mercado Pago.
        - Implementar lógica de idempotência usando o x-idempotency-key ou o ID do evento para evitar processamento duplicado.
    - Implementar Lógica de Negócio:
        - Validar o corpo do webhook com um schema Zod.
        - Se o pagamento for `approved`:
            - Atualizar orders.status para paid.
            - Confirmar a baixa de estoque (mudar inventory_movements de reserve para confirm).
        - Se o pagamento for `rejected` ou `expired`:
            - Atualizar orders.status para canceled.
            - Reverter a reserva de estoque (criar um novo movimento cancel_reserve ou deletar o reserve).

3. Frontend: Página de Checkout (/checkout)

- Criar Estrutura da Página:
    - Criar o arquivo src/app/checkout/page.tsx.
    - Implementar um gerenciador de estado (pode ser useState ou Zustand) para controlar os 3 passos do formulário.
- Implementar os 3 Passos:
    - Passo 1: Endereço: Criar um formulário com React Hook Form e Zod para os dados de entrega.
    - Passo 2: Frete: Exibir opções de frete (pode ser um valor fixo por enquanto).
    - Passo 3: Pagamento:
        - Ao chegar neste passo, chamar a API .../criar-pedido para registrar o pedido no sistema.
        - Com o order_id retornado, chamar a API .../mp/preferencia.
        - Renderizar o botão de pagamento ou o QR Code do Pix com base na resposta do Mercado Pago.
- Página de Sucesso/Confirmação:
    - Criar uma página de confirmação de pedido (ex: /meus-pedidos/[id]) para onde o usuário será redirecionado após o pagamento.

Desenvolva a partir do checklist "Checklist de Desenvolvimento: Checkout e Pagamentos". Sempre analise para não desenvolvermos logicas duplicatas o redundantes