Estrutura das Coleções do Firebase

Este documento descreve a estrutura das coleções no Firebase Firestore, essencial para entender a organização dos dados da aplicação.

Coleções Principais:

1. users

- Propósito: Armazena perfis de usuários do sistema.
- Documentos: Cada documento representa um usuário. O documentId é o uid do Firebase Authentication.
- Campos Típicos:
    - name: Nome completo do usuário.
    - email: Endereço de e-mail do usuário.
    - role: Função/cargo do usuário (ex: 'admin', 'manager', 'employee', 'finance').
    - storeId: (Opcional) ID da loja à qual o usuário está associado.
    - createdAt: Timestamp de criação do usuário.

2. stores

- Propósito: Armazena informações sobre cada loja física.
- Documentos: Cada documento representa uma loja. O documentId é gerado automaticamente pelo Firestore.
- Campos Típicos:
    - name: Nome da loja.
    - address: Endereço completo da loja.
    - phone: Telefone de contato da loja.
    - managerId: (Opcional) ID do usuário que é gerente da loja.
    - isActive: Booleano indicando se a loja está ativa.

3. product_attributes

- Propósito: Armazena atributos globais e listas de valores para produtos, usados em toda a aplicação. Esta coleção contém documentos específicos para cada tipo de atributo.
- Documentos:

- categories: Contém uma lista de categorias de produtos.
  - list: Array de strings (ex: ['Camisetas', 'Calças', 'Vestidos']).
- sizeGrades: Contém grades de tamanho.
  - Padrão: Objeto com arrays (ex: { P: ['PP', 'P', 'M', 'G', 'GG'], Infantil: ['RN', 'P', 'M'] }).
- colors: Contém uma lista de cores.
  - list: Array de strings (ex: ['Preto', 'Branco', 'Vermelho']).
- suppliers: Contém uma lista de fornecedores.
  - list: Array de strings (ex: ['Fornecedor A', 'Fornecedor B']).

4. products

- Propósito: Catálogo de produtos da loja, incluindo variações e estoque.
- Coleção Aninhada: /stores/{storeId}/products/{productId}
- Documentos: Cada documento representa um produto.
- Campos Típicos:
  - code: Código SKU do produto (único).
  - name: Nome do produto.
  - description: Descrição do produto.
  - category: Categoria do produto (string).
  - supplier: Fornecedor do produto (string).
  - costPrice: Preço de custo do produto.
  - salePrice: Preço de venda sugerido do produto.
  - active: Booleano indicando se o produto está ativo.
  - hasVariations: Booleano indicando se o produto tem variações de cor/tamanho.
  - variations: (Se hasVariations for true) Array de objetos, cada um representando uma variação:
    - color: Cor da variação.
    - size: Tamanho da variação.
    - quantity: Quantidade em estoque para esta variação.

        - isActive: Se a variação está ativa.

5. sales

- Propósito: Registra todas as transações de vendas concluídas.
- Coleção Aninhada: /stores/{storeId}/sales/{saleId}
- Documentos: Cada documento representa uma venda.
- Campos Típicos:
    - saleDate: Timestamp da venda.
    - totalAmount: Valor total da venda.
    - paymentMethod: Método de pagamento (ex: 'Dinheiro', 'Cartão de Crédito', 'Pix', 'Débito').
    - productsSold: Array de objetos, cada um representando um item vendido:
        - productId: ID do produto.
        - productCode: Código do produto.
        - productName: Nome do produto.
        - variation: Objeto com color e size (se aplicável).
        - quantity: Quantidade vendida.
        - price: Preço unitário no momento da venda.
    - cashierId: ID do usuário que realizou a venda.
    - cashRegisterSessionId: ID da sessão de caixa à qual a venda pertence.

6. cash_register_sessions

- Propósito: Gerencia o estado e o histórico das sessões do caixa (abertura, fechamento, suprimentos, sangrias).
- Coleção Aninhada: /stores/{storeId}/cash_register_sessions/{sessionId}
- Documentos: Cada documento representa uma sessão de caixa diária (ou por turno).
- Campos Típicos:
    - openingTime: Timestamp de abertura.
    - closingTime: (Opcional) Timestamp de fechamento.

    - openingBalance: Saldo inicial do caixa.
    - closingBalance: (Após fechamento) Saldo final do caixa.
    - status: 'open' ou 'closed'.
    - cashierId: ID do usuário que abriu a sessão.
    - dailySalesSummary: Objeto resumindo as vendas da sessão (total, dinheiro, crédito, débito, pix).
    - supplies: Array de objetos de suprimentos (ex: [{ amount: 50, description: 'Troco inicial' }]).
    - outflows: Array de objetos de sangrias/saídas (ex: [{ amount: 20, description: 'Compra de material de limpeza' }]).

7. stock_movements

- Propósito: Registra todas as entradas e saídas de estoque que não são vendas.
- Coleção Aninhada: /stores/{storeId}/stock_movements/{movementId}
- Documentos: Cada documento representa um movimento de estoque.
- Campos Típicos:
    - timestamp: Timestamp do movimento.
    - productId: ID do produto afetado.
    - productCode: Código do produto.
    - variation: Objeto com color e size (se aplicável).
    - quantity: Quantidade movimentada (positiva para entrada, negativa para saída).
    - type: Tipo de movimento (ex: 'entrada_compra', 'ajuste_estoque', 'devolucao_fornecedor', 'perda').
    - reason: Descrição do motivo do movimento.
    - userId: ID do usuário que registrou o movimento.

8. purchase_orders

- Propósito: Gerencia pedidos de compra feitos a fornecedores.
- Coleção Aninhada: /stores/{storeId}/purchase_orders/{orderId}
- Documentos: Cada documento representa um pedido de compra.

- Campos Típicos:
    - orderDate: Data do pedido.
    - supplier: Nome do fornecedor.
    - status: Status do pedido (ex: 'pending', 'received', 'canceled').
    - expectedDeliveryDate: Data de entrega esperada.
    - items: Array de objetos, cada um representando um item pedido:
        - productId: ID do produto.
        - productCode: Código do produto.
        - productName: Nome do produto.
        - variation: Objeto com color e size (se aplicável).
        - quantity: Quantidade pedida.
        - costPrice: Preço de custo unitário.
    - totalCost: Custo total do pedido.
    - receivedDate: (Opcional) Data de recebimento dos itens.

9. bankAccounts

- Propósito: Armazena informações de contas bancárias da empresa ou loja.
- Documentos: Cada documento representa uma conta bancária.
- Campos Típicos:
    - bankName: Nome do banco.
    - accountNumber: Número da conta.
    - agency: Agência.
    - accountType: Tipo de conta (ex: 'Corrente', 'Poupança').
    - balance: Saldo atual da conta.
    - storeId: (Opcional) ID da loja à qual a conta pertence.

10. transactions

- Propósito: Registra transações financeiras mais amplas (pagamentos de contas, transferências, etc.).
- Documentos: Cada documento representa uma transação financeira.

- Campos Típicos:
    - timestamp: Data e hora da transação.
    - type: Tipo de transação (ex: 'revenue', 'expense', 'transfer').
    - amount: Valor da transação.
    - description: Descrição da transação.
    - sourceAccount: (Opcional) ID da conta de origem.
    - destinationAccount: (Opcional) ID da conta de destino.
    - relatedEntity: (Opcional) Entidade relacionada (ex: fornecedor, cliente).
    - storeId: (Opcional) ID da loja à qual a transação está associada.

Relacionamentos entre Coleções:

- users e stores: Um usuário pode estar associado a uma store através do storeId no documento do usuário.
- products e stores: Produtos são aninhados sob stores, indicando que cada loja tem seu próprio catálogo de produtos.
- sales e stores: Vendas são aninhadas sob stores, registrando transações específicas da loja.
- sales e products: Vendas referenciam products e suas variations pelos seus códigos e IDs.
- sales e cash_register_sessions: Cada venda está ligada a uma sessão de caixa.
- cash_register_sessions e stores: Sessões de caixa são aninhadas sob stores.
- stock_movements e stores: Movimentações de estoque são aninhadas sob stores.
- stock_movements e products: Movimentações de estoque referenciam products e suas variations.
- purchase_orders e stores: Pedidos de compra são aninhados sob stores.
- purchase_orders e products: Pedidos de compra referenciam products e suas variations.

- product_attributes: Esta coleção é de nível raiz e serve como um repositório central de valores pré-definidos que são usados em products e outras partes da aplicação para garantir consistência.

Considerações sobre as Regras de Segurança:

As regras de segurança do Firestore (rules.txt) são projetadas para controlar o acesso a essas coleções com base na função (role) do usuário e no storeId (onde aplicável), garantindo que os usuários acessem apenas os dados autorizados e relevantes para sua loja. Por exemplo, apenas admins podem gerenciar product_attributes, enquanto funcionários e gerentes podem gerenciar products dentro de suas respectivas lojas e sales apenas para suas lojas.