**Plano de Trabalho: Automação e Centralização Financeira**

Objetivo Principal: Estabelecer um fluxo automatizado para coletar, processar e centralizar dados financeiros diversos (cartões, boletos, holerites, bancos) no PostgreSQL, e implementar um sistema de lembretes de vencimento.

Abordagem: Implementação modular por fonte de dados, utilizando n8n com preferência por componentes prontos e um microserviço Python para tarefas específicas. Cada passo dentro de um subprojeto será o mais desacoplado possível.

Ferramentas Principais: n8n, PostgreSQL, Microserviço Python (Flask/PyMuPDF), Docker, IA (Gemini) como copiloto.

**Pré-requisitos (Configuração Inicial)**

1. Instalação e Configuração do n8n.
2. Acesso ao Banco de Dados PostgreSQL.
3. Teste de Conexão n8n -> PostgreSQL.
4. Modelagem Inicial do Banco de Dados (core\_transaction e tabelas relacionadas).
5. Configuração Docker para n8n (com Docker CLI) e Microserviço Python (com PyMuPDF), incluindo volumes mapeados e rede Docker.

**Subprojeto 1: Automação de Dados de Cartão de Crédito**

Objetivo: Coletar, processar e inserir transações de um cartão de crédito (geralmente de um arquivo CSV ou similar) no banco de dados PostgreSQL.

* Passo 1.1: Coleta de Dados (Acesso ao Arquivo de Extrato)
  + Status: Concluído! Configurado o Trigger de E-mail (Gmail Trigger) para baixar anexos PDF de faturas e salvar em /shared\_staging, marcando o e-mail como lido.
* Passo 1.2: Carregamento/Parsing do Arquivo (PDF)
  + Status: Concluído! Implementado microserviço Python (Flask/PyMuPDF) rodando em container Docker separado para ler o PDF salvo, extrair dados da fatura e transações, e retornar JSON via HTTP. Nó HTTP Request no n8n configurado para chamar este serviço.
* Passo 1.3: Transformação e Mapeamento dos Dados (Pré-Inserção)
  + Status: Em Andamento. Separar a lista de transações em itens individuais e preparar os dados para inserção no DB.
  + Ação: Adicionar nó "Split Out" (ou "Item Lists") para separar as transações. Adicionar nó "Set" para derivar campos (mês/ano de referência, notas) e definir campos fixos (status, origem, account\_id, user\_id, created\_at, updated\_at).
  + Ajuda da IA: Auxiliar na configuração dos nós e expressões.
* Passo 1.4: Categorização e Marcação
  + Status: Pendente. (Será implementado após a preparação inicial, envolvendo busca no DB para Terceiro e Categoria).
* Passo 1.5: Armazenamento no PostgreSQL
  + Status: Pendente. (Inserir os itens de transação processados no DB).

**Subprojeto 2: Automação de Boletos Recebidos por E-mail**

Objetivo: Monitorar e-mails com boletos, extrair dados e inserir no banco.

* Status: Pendente.

**Subprojeto 3: Automação de Holerite (PDF)**

Objetivo: Processar PDF de holerite, extrair receita e inserir no banco.

* Status: Pendente.

**Subprojeto 4: Automação de Dados de Contas Bancárias (Extrato)**

Objetivo: Processar extratos bancários e inserir no banco.

* Status: Pendente.

**Subprojeto 5: Implementação de Lembretes de Vencimento**

Objetivo: Consultar o banco por contas a vencer e enviar notificações.

* Status: Pendente.

**Subprojeto 6: Integração de Outras Despesas (Manuais/Menos Frequentes)**

Objetivo: Definir processo simples para registrar despesas manuais.

* Status: Pendente.

**Subprojeto 7: Refinamento, Análise e Evolução**

Objetivo: Melhorar fluxos, tratar erros, analisar dados e usar IA para insights.

* Status: Pendente.

**Plano de Trabalho: Automação e Centralização Financeira com n8n e PostgreSQL**

**Objetivo Principal:** Estabelecer um fluxo automatizado para coletar, processar e centralizar dados financeiros diversos (cartões, boletos, holerites, bancos) no PostgreSQL, e implementar um sistema de lembretes de vencimento.

**Ferramentas Principais:** n8n, PostgreSQL, Python (para tarefas específicas), IA (Gemini) como copiloto.

**Pré-requisitos (Configuração Inicial):**

1. **Instalação e Configuração do n8n:**
   * Instalar o n8n (via Docker é altamente recomendado para isolamento e facilidade de gerenciamento).
   * Garantir que o n8n esteja acessível e rodando.
2. **Acesso ao Banco de Dados PostgreSQL:**
   * Confirmar as credenciais de acesso ao seu banco PostgreSQL (host, port, database, user, password).
   * Verificar se o n8n consegue se conectar ao banco de dados (testar a conexão com o nó "Postgres").
3. **Modelagem Inicial do Banco de Dados:**
   * Ter um modelo de dados básico no PostgreSQL para receber as transações (tabela transacoes com colunas como data, descricao, valor, tipo (receita/despesa), origem (cartao/banco/boleto/holerite), conta\_id, cartao\_id, categoria\_id, data\_vencimento, pago, data\_pagamento, etc.). Este modelo evoluirá.
   * **A IA pode ajudar:** Peça ao Gemini para revisar ou sugerir um modelo de dados inicial para transações financeiras, considerando seus tipos de despesa e receita.

**Subprojeto 1: Automação de Dados de Cartão de Crédito**

**Objetivo:** Coletar, processar e inserir transações de um cartão de crédito (geralmente de um arquivo CSV ou similar) no banco de dados PostgreSQL.

* **Passo 1.1: Coleta de Dados (Download/Acesso ao Arquivo)**
  + **Ação:** Definir como o arquivo de extrato é obtido (download manual do site do banco é o mais comum para usuários finais).
  + **Implementação no n8n:**
    - **Opção A (Download Manual + Monitoramento):** Use o nó **"Folder Watch"** (se o n8n estiver rodando localmente ou tiver acesso ao sistema de arquivos) ou configure o n8n para monitorar um serviço de nuvem (ex: nó **"Google Drive Trigger"** se você salvar lá). O workflow será disparado quando um novo arquivo for detectado.
    - **Opção B (Download Automatizado - mais complexo):** Pode envolver nós HTTP para interagir com o site (raramente funciona bem para sites de bancos) ou RPA (fora do escopo direto do n8n sem integrações específicas). Foco na Opção A por simplicidade inicial.
  + **Desacoplamento:** Este passo entrega o arquivo ou seu caminho para o próximo passo.
* **Passo 1.2: Carregamento/Parsing do Arquivo**
  + **Ação:** Ler o conteúdo do arquivo de extrato (provavelmente CSV, mas pode ser TXT ou XLS).
  + **Implementação no n8n:** Use o nó **"Read CSV"**. Configure o delimitador (vírgula, ponto e vírgula), se há cabeçalho, etc. Se for XLSX, pode precisar de um nó intermediário ou script Python para converter para CSV.
  + **Desacoplamento:** Este passo transforma o arquivo bruto em uma coleção de itens/objetos dentro do n8n, onde cada item representa uma linha/transação.
* **Passo 1.3: Transformação e Mapeamento dos Dados**
  + **Ação:** Limpar os dados, renomear colunas, formatar datas e valores, extrair informações relevantes (descrição, valor, data da transação), mapear para as colunas do seu banco de dados.
  + **Implementação no n8n:**
    - Nó **"Set"** para renomear campos, adicionar campos fixos (ex: origem = 'cartao', conta\_id = X).
    - Nó **"Convert Data Type"** para garantir que valores sejam números e datas sejam datas.
    - Nó **"Split In Batches"** (se o arquivo for muito grande) para processar em lotes menores.
    - Pode precisar de um nó **"Function"** ou **"Execute Python Function"** para transformações mais complexas ou extração de padrões na descrição da transação (ex: Compra no estabelecimento X - 1/3 -> extrair estabelecimento X e 1/3).
  + **A IA pode ajudar:** Peça ao Gemini para ajudar a criar regex ou código Python para extrair padrões de descrições de transações ou para lógica de transformação complexa.
  + **Desacoplamento:** Este passo gera um formato de dado padronizado, pronto para ser categorizado e inserido no banco.
* **Passo 1.4: Categorização e Marcação (Tagging)**
  + **Ação:** Atribuir uma categoria financeira (alimentação, transporte, lazer) e talvez tags (parcela 1, fixa, variável) a cada transação. Inicialmente, pode ser manual ou baseada em regras simples.
  + **Implementação no n8n:** Use o nó **"Switch"** ou múltiplos nós **"IF"** baseados em condições na descrição da transação (ex: Se descrição contiver "UBER", categoria="Transporte"). Para regras mais complexas ou busca em uma lista de fornecedores, um nó **"Function"** ou **"Execute Python Function"** pode ser necessário.
  + **A IA pode ajudar:** Use o Gemini para gerar listas de palavras-chave comuns para categorias ou para ajudar a refinar as regras no nó Function. No futuro, a IA poderia sugerir categorias para transações novas.
  + **Desacoplamento:** Este passo adiciona informações de categorização aos dados transformados.
* **Passo 1.5: Armazenamento no PostgreSQL**
  + **Ação:** Inserir as transações processadas e categorizadas na tabela transacoes do seu banco PostgreSQL.
  + **Implementação no n8n:** Use o nó **"Postgres"**. Configure a operação como "Insert". Mapeie os campos dos itens do n8n para as colunas da sua tabela transacoes. Considere a operação "Upsert" se você precisar lidar com a possibilidade de re-importar o mesmo extrato.
  + **Desacoplamento:** Este passo finaliza o fluxo para esta fonte de dados, persistindo-a no repositório central.

**Subprojeto 2: Automação de Boletos Recebidos por E-mail**

**Objetivo:** Monitorar a caixa de e-mail, identificar e processar e-mails com boletos/contas, extrair informações chave (fornecedor, valor, data de vencimento) e inseri-las no banco PostgreSQL (como despesas a pagar).

* **Passo 2.1: Coleta de Dados (Monitorar E-mail)**
  + **Ação:** Conectar ao servidor de e-mail (IMAP/POP3) para buscar novos e-mails.
  + **Implementação no n8n:** Use o nó **"IMAP"** ou **"POP3"** como "Trigger" para iniciar o workflow automaticamente quando um novo e-mail chegar, ou use-o como nó regular em um workflow agendado. Configure as credenciais e filtros (por remetente, assunto).
  + **Desacoplamento:** Este passo entrega o conteúdo do e-mail e seus anexos para o próximo passo.
* **Passo 2.2: Identificação e Extração de Dados do E-mail/Anexo**
  + **Ação:** Analisar o corpo do e-mail e/ou anexos (PDF da fatura) para encontrar o fornecedor, valor e data de vencimento.
  + **Implementação no n8n:**
    - Nó **"HTML Extract"** ou **"Text Extract"** para buscar padrões no corpo do e-mail.
    - **Para PDF:** Requer um nó **"Execute Python Function"**. Você precisará usar bibliotecas Python (ex: PyMuPDF/fitz, pdftotext) para ler o texto do PDF anexado. Dentro do script Python, use regex ou busca de texto para encontrar as informações (procurar por "Vencimento:", "Total a Pagar:", CNPJ do fornecedor, etc.). O script retornará os dados extraídos para o n8n.
  + **A IA pode ajudar:** **Crucial aqui!** Peça ao Gemini para ajudar a escrever o script Python para ler PDF e extrair informações específicas usando regex ou outras técnicas. Forneça exemplos do texto do PDF.
  + **Desacoplamento:** Este passo transforma o conteúdo do e-mail/PDF em dados estruturados (fornecedor, valor, vencimento).
* **Passo 2.3: Transformação e Mapeamento dos Dados**
  + **Ação:** Limpar os dados extraídos, formatar, mapear para as colunas do banco de dados (marcar como despesa a pagar, definir pago = false, preencher data de vencimento).
  + **Implementação no n8n:** Nó **"Set"**, **"Convert Data Type"**. Mapear os dados extraídos no passo 2.2 para as colunas descricao (Fornecedor + detalhes), valor, data\_vencimento, tipo = 'despesa', origem = 'boleto', pago = false, etc.
  + **Desacoplamento:** Gera dados padronizados para inserção.
* **Passo 2.4: Categorização**
  + **Ação:** Atribuir categoria à despesa (Luz, Água, Internet, etc.), baseando-se no fornecedor extraído.
  + **Implementação no n8n:** Nó **"Switch"** ou **"IF"** baseado no nome do fornecedor.
  + **Desacoplamento:** Adiciona a categoria.
* **Passo 2.5: Armazenamento no PostgreSQL**
  + **Ação:** Inserir a conta a pagar processada na tabela transacoes.
  + **Implementação no n8n:** Nó **"Postgres"**, operação "Insert".

**Subprojeto 3: Automação de Holerite (PDF)**

**Objetivo:** Processar o arquivo PDF do holerite, extrair informações de receita (salário líquido, benefícios) e inseri-las no banco PostgreSQL.

* **Passo 3.1: Coleta de Dados (Acesso ao Arquivo PDF)**
  + **Ação:** Obter o arquivo PDF do holerite (download manual ou recebimento por e-mail).
  + **Implementação no n8n:** Nó **"Folder Watch"** ou monitoramento de serviço de nuvem (como no Passo 1.1). Ou nó **"IMAP"** se for recebido por e-mail (como no Passo 2.1).
  + **Desacoplamento:** Entrega o arquivo PDF.
* **Passo 3.2: Extração de Dados do PDF**
  + **Ação:** Ler o texto do PDF do holerite e extrair o valor do salário líquido e outros proventos/descontos relevantes.
  + **Implementação no n8n:** **Nó "Execute Python Function"** é o mais provável. Use bibliotecas Python para ler PDF e extrair dados. Holerites têm layouts variados, então este script Python precisará ser robusto ou adaptado. Busque por rótulos como "SALÁRIO LÍQUIDO", "TOTAL DE PROVENTOS", "TOTAL DE DESCONTOS".
  + **A IA pode ajudar:** **Crucial aqui!** Forneça um exemplo (anonimizado) do texto do seu holerite para o Gemini e peça ajuda para escrever o script Python que extraia os valores corretos de salário líquido, etc.
  + **Desacoplamento:** Transforma o PDF em dados estruturados (salário líquido, data de referência).
* **Passo 3.3: Transformação e Mapeamento dos Dados**
  + **Ação:** Limpar e formatar os dados extraídos, mapear para as colunas do banco de dados (marcar como receita).
  + **Implementação no n8n:** Nó **"Set"**, **"Convert Data Type"**. Mapear para valor (salário líquido), data (data de recebimento ou data de referência do holerite), tipo = 'receita', origem = 'holerite', etc.
  + **Desacoplamento:** Gera dados padronizados para inserção.
* **Passo 3.4: Armazenamento no PostgreSQL**
  + **Ação:** Inserir a receita na tabela transacoes.
  + **Implementação no n8n:** Nó **"Postgres"**, operação "Insert".

**Subprojeto 4: Automação de Dados de Contas Bancárias (Extrato)**

**Objetivo:** Processar dados de movimentação de contas correntes/poupança e inseri-los no banco PostgreSQL. (Este pode ser o mais desafiador devido à variedade de formatos e acesso).

* **Passo 4.1: Coleta de Dados (Acesso ao Extrato)**
  + **Ação:** Obter o extrato da conta (CSV, OFX, PDF). OFX é um formato comum e mais fácil de processar.
  + **Implementação no n8n:**
    - **Download Manual + Monitoramento:** Nó **"Folder Watch"** ou monitoramento de nuvem (como no Passo 1.1).
    - **OFX:** Se o banco permitir download em OFX, é o ideal.
    - **API (Raro/Complexo):** Alguns bancos oferecem APIs (Open Banking, mas acesso para usuários finais é limitado no Brasil).
  + **Desacoplamento:** Entrega o arquivo de extrato.
* **Passo 4.2: Carregamento/Parsing do Extrato**
  + **Ação:** Ler o arquivo de extrato.
  + **Implementação no n8n:**
    - Nó **"Read CSV"** se for CSV.
    - **Para OFX:** Requer um nó **"Execute Python Function"** com uma biblioteca Python para parsing de OFX (ex: ofxparse). O script lerá o OFX e retornará as transações em um formato estruturado.
    - **Para PDF:** Requer **"Execute Python Function"** com bibliotecas de leitura de PDF e lógica de extração de dados (muito variável e difícil para extratos complexos).
  + **A IA pode ajudar:** Peça ajuda ao Gemini para encontrar bibliotecas Python para parsing de OFX ou para estratégias de extração de texto de PDFs de extrato (reconhecendo que PDFs de extrato são difíceis).
  + **Desacoplamento:** Transforma o arquivo bruto em transações estruturadas dentro do n8n.
* **Passo 4.3: Transformação e Mapeamento dos Dados**
  + **Ação:** Limpar dados, mapear campos (data, descrição, valor), identificar tipo (débito/crédito).
  + **Implementação no n8n:** Nó **"Set"**, **"Convert Data Type"**, **"IF"** (para determinar se é receita ou despesa baseado no sinal do valor ou coluna específica do extrato). Mapear para data, descricao, valor, tipo ('receita' ou 'despesa'), origem = 'banco', conta\_id = Y.
  + **Desacoplamento:** Gera dados padronizados para inserção.
* **Passo 4.4: Categorização**
  + **Ação:** Atribuir categorias às transações (igual ao passo 1.4, mas para transações bancárias).
  + **Implementação no n8n:** Nó **"Switch"**, **"IF"**, **"Function"** (baseado na descrição).
  + **Desacoplamento:** Adiciona a categoria.
* **Passo 4.5: Armazenamento no PostgreSQL**
  + **Ação:** Inserir as transações na tabela transacoes.
  + **Implementação no n8n:** Nó **"Postgres"**, operação "Insert". Considere "Upsert" se houver risco de duplicidade ao re-importar extratos.

**Subprojeto 5: Implementação de Lembretes de Vencimento**

**Objetivo:** Consultar o banco de dados por contas a pagar próximas ao vencimento e enviar notificações.

* **Passo 5.1: Consulta de Vencimentos**
  + **Ação:** Rodar um workflow diariamente que busca no banco por transações (tipo='despesa', pago=false) com data\_vencimento nos próximos X dias.
  + **Implementação no n8n:**
    - Nó **"Schedule"** (Trigger) para rodar o workflow diariamente.
    - Nó **"Postgres"** com operação "Select". Escreva uma query SQL para selecionar as contas a pagar relevantes.
  + **A IA pode ajudar:** Peça ao Gemini para ajudar a escrever a query SQL correta para selecionar contas a vencer em um intervalo de datas.
* **Passo 5.2: Formatação da Mensagem de Lembrete**
  + **Ação:** Criar o conteúdo da mensagem de alerta com os detalhes da conta (fornecedor, valor, vencimento).
  + **Implementação no n8n:** Nó **"Set"** ou **"Function"** para formatar o texto da mensagem usando os dados retornados pela consulta SQL.
* **Passo 5.3: Envio da Notificação**
  + **Ação:** Enviar o lembrete através do canal desejado.
  + **Implementação no n8n:** Use nós de notificação prontos: **"Send Email"**, **"Telegram"**, **"Slack"**, **"Google Calendar"** (para adicionar um evento no seu calendário).

**Subprojeto 6: Integração de Outras Despesas (Manuais/Menos Frequentes)**

**Objetivo:** Definir um processo para registrar despesas que não são facilmente automatizadas (ex: manutenção de veículo em dinheiro, pequenas compras em feiras).

* **Passo 6.1: Definição do Processo Manual Otimizado**
  + **Ação:** Criar um formulário simples (ex: Google Forms, ou no seu próprio app customizado) ou uma planilha específica onde essas poucas transações sejam registradas.
  + **Implementação no n8n (Opcional):** Se usar Google Forms ou Sheets, pode-se criar um workflow que monitore as novas entradas e as insira no PostgreSQL automaticamente (usando os nós **"Google Forms Trigger"** ou **"Google Sheets Trigger"** e o nó **"Postgres"**).

**Subprojeto 7: Refinamento, Análise e Evolução**

**Objetivo:** Melhorar os fluxos existentes, lidar com erros, adicionar funcionalidades de análise e usar a IA para insights.

* **Passo 7.1: Tratamento de Erros nos Workflows**
  + **Ação:** Adicionar tratamento de erro nos workflows do n8n (ex: o que acontece se o formato do CSV mudar? Se o script Python falhar?).
  + **Implementação no n8n:** Usar a funcionalidade de "Error Handling" nos nós e "Error Workflow". Configurar alertas (e-mail, notificação) quando um workflow falhar.
  + **A IA pode ajudar:** Peça ao Gemini para sugerir estratégias de tratamento de erro para partes específicas dos seus workflows.
* **Passo 7.2: Refinamento das Regras de Categorização/Rateio**
  + **Ação:** Ajustar e adicionar regras de categorização e, se necessário, implementar a lógica de rateio entre despesas da família no n8n ou via queries no banco.
  + **A IA pode ajudar:** Use a IA para analisar suas transações passadas (uma vez no banco) e sugerir novas regras de categorização ou identificar inconsistências.
* **Passo 7.3: Dashboards e Análise (Usando o Banco Centralizado)**
  + **Ação:** Utilizar a base de dados PostgreSQL para criar visualizações dos seus gastos, receitas, saldos, etc.
  + **Ferramentas:** Você pode usar o frontend do seu app customizado (focando na parte de visualização), ferramentas de BI (Tableau Public, Power BI, Metabase - muitos têm conectores para PostgreSQL), ou até mesmo consultas SQL diretas.
  + **A IA pode ajudar:** Peça ao Gemini para ajudar a escrever queries SQL complexas para relatórios específicos ("Quanto gastei em cada categoria nos últimos 3 meses?", "Qual a evolução do meu saldo na poupança?").
* **Passo 7.4: Utilização de IA para Insights Financeiros**
  + **Ação:** Uma vez com os dados centralizados, explorar como a IA pode fornecer insights preditivos ou analíticos (conforme discutido anteriormente).
  + **Implementação:** Pode envolver exportar dados para uma ferramenta de análise com IA, ou desenvolver (no seu app ou uma ferramenta separada) uma interface que consulta o banco e envia os dados para o Gemini para análise.

**Considerações Importantes:**

* **Comece por um mini-projeto:** Não tente fazer tudo de uma vez. O Subprojeto 1 (Cartão de Crédito) costuma ser um bom ponto de partida.
* **Itere:** Cada subprojeto pode ser implementado em etapas. Primeiro, apenas leia o arquivo e mostre os dados no n8n. Depois, mapeie. Depois, insira no banco. Depois, adicione a categorização.
* **Documentação:** Mantenha um registro simples de como cada workflow do n8n funciona e quais dados ele processa.
* **Segurança:** Ao lidar com dados financeiros, garanta que seu n8n esteja seguro e que as credenciais do banco de dados e e-mail estejam protegidas.
* **Mudanças:** Bancos e serviços podem mudar o formato dos seus extratos/e-mails sem aviso. Se um workflow parar de funcionar, a abordagem desacoplada facilita identificar e corrigir o passo quebrado (ex: se o formato do CSV mudar, você só precisa ajustar o nó "Read CSV" ou a lógica de transformação, não todo o fluxo).

Este plano detalhado te dá um roteiro para começar a construir sua automação financeira usando n8n e seu banco de dados PostgreSQL, passo a passo e de forma modular. Lembre-se de usar o Gemini ativamente em cada etapa para superar desafios de codificação, configuração ou análise.