

Relatório de Projeto Final: I2A2 Agentes Inteligentes

1. Nome do Grupo

ProcessON

2. Integrantes do Grupo

- Eric Bueno | ericflorianogmail.com
- Leonardo Santos | leonardo@teajudo.com.br
- Letícia Machado | leticiiamachado1510@gmail.com
- Marco Andrey | andreyalvim1996@gmail.com

3. Descrição do Tema Escolhido

O projeto final do grupo **ProcessON** consiste numa solução de automação inteligente para apuração fiscal e tributária, com foco principal no cálculo da Contribuição Social sobre o Lucro Líquido (CSLL) e outros impostos correlatos (ICMS, ISS, PIS, COFINS, IRPJ).

A solução foi desenvolvida como um agente autônomo utilizando o framework Agno SDK 2.0, projetado para realizar um fluxo completo de análise de dados fiscais:

1. **Coleta e Leitura:** O sistema conecta-se diretamente a um banco de dados PostgreSQL para ler dados brutos das tabelas `notas` e `itens`.
2. **Apuração e Cálculo:** Um motor de cálculo (`apuracao.py`) processa esses dados, normaliza colunas, converte valores e realiza a apuração estimada dos tributos com base em alíquotas e regime de enquadramento (ex: Lucro Presumido) fornecidos pelo usuário.
3. **Geração de Resultados:** O processo gera arquivos de saída estruturados, incluindo um resumo em JSON (`resumo_apuracao.json`) e diversos pivôs de faturamento em formato CSV (ex: por mês, por UF, por CFOP).
4. **Análise e Visualização:** Um agente de IA (`csvx.py`), equipado com o modelo `gpt-5-mini`, analisa os arquivos CSV gerados. Este agente utiliza IA generativa e uma base de conhecimento sobre a reforma tributária para interpretar os dados, gerar relatórios diagnósticos, identificar riscos fiscais e destacar oportunidades de otimização.

5. **Interação:** A solução é operada através de uma interface web interativa construída com Gradio ([app_gradio.py](#)), onde o utilizador pode ajustar os parâmetros de cálculo, executar a análise e visualizar os resultados, incluindo um gráfico de faturamento mensal gerado com Plotly.

4. Público Alvo

A solução ProcessON foi desenhada para atender a uma ampla gama de profissionais e organizações do setor fiscal e contábil. O público-alvo principal inclui:

- Escritórios de contabilidade;
- Departamentos fiscais de empresas médias e grandes;
- Consultorias e auditorias tributárias;
- Legal/TaxTechs e desenvolvedores de software fiscal;
- CFOs (Chief Financial Officers), controllers e analistas tributários.

5. Justificativa do Tema Escolhido

A escolha do tema justifica-se por um problema de negócio latente e significativo no mercado brasileiro.

O Problema: Atualmente, a apuração de tributos complexos como a CSLL ainda é realizada de forma manual por muitos dos mais de 70 mil escritórios contábeis no Brasil. Este processo manual é intrinsecamente:

- **Repetitivo e Lento:** Consome um tempo valioso de analistas qualificados.
- **Suscetível a Erros:** A complexidade da legislação e o volume de dados aumentam a probabilidade de erros humanos.
- **Arriscado:** Erros na apuração geram riscos de não conformidade tributária e a necessidade de retrabalho.

A Oportunidade: O mercado para esta solução é vasto, englobando mais de 89 mil empresas contábeis ativas (CFC) e mais de 249 mil empresas enquadradas no regime de Lucro Real, que lidam diretamente com essa complexidade.

Valor Agregado: A solução ProcessON agrega valor direto ao público-alvo ao:

- **Reducir drasticamente o tempo de apuração**, com estimativas de redução de até 80%.
- **Eliminar erros manuais** nos cálculos e na consolidação de dados.
- **Aumentar a segurança jurídica** e a conformidade fiscal.
- **Fornecer rastreabilidade completa** através de relatórios e logs.
- **Elevar a produtividade da equipe fiscal**, liberando analistas de tarefas repetitivas para se concentrarem em análises estratégicas.

6. Detalhamento do que foi Desenvolvido

A solução é composta por três componentes principais de software que operam de forma integrada: o motor de apuração (`apuracao.py`), o agente de IA (`csvx.py`) e a interface do utilizador (`app_gradio.py`).

6.1. Componente 1: Motor de Cálculo (`apuracao.py`)

Este módulo é o núcleo de processamento de dados da solução. A sua operação é detalhada abaixo:

- **Conexão de Dados:** Utiliza as bibliotecas `pandas` e `SQLAlchemy` para criar uma conexão (engine) com um banco de dados PostgreSQL.
- **Leitura de Dados:**
 - Lê a tabela `notas` (dados das notas fiscais) integralmente para a memória.
 - Lê a tabela `itens` (itens das notas fiscais) de forma eficiente, utilizando "chunks" (pedaços) de tamanho configurável (ex: 20.000 linhas) para processar grandes volumes de dados sem esgotar a memória.
- **Normalização e Mapeamento (Pré-processamento):**
 - **Colunas:** Aplica uma função `normalize_colname` para limpar e padronizar os nomes das colunas (ex: "Valor Total" -> "valor_total").
 - **Deteção Inteligente:** Utiliza a função `find_best_column` para localizar colunas essenciais mesmo que tenham nomes diferentes (ex: encontrar "chave_acesso", "valor_total", "data_emissao", "cfop", "ncm").
 - **Valores:** Converte valores monetários em formato brasileiro (ex: "R\$ 1.234,56") para o tipo numérico `float` usando a função `parse_valor_brasileiro`.
- **Lógica de Cálculo Tributário:**
 - Itera sobre cada "chunk" de itens e combina-os com os dados das respetivas notas fiscais (via `merge` na coluna `chave_acesso`).
 - Calcula os impostos **ICMS**, **ISS**, **PIS (cumulativo)** e **COFINS (cumulativo)** aplicando as alíquotas padrão (ex: `ALIQ_ICMS_DEFAULT`) sobre o valor total de cada item.
 - Calcula **IRPJ** e **CSLL** com base no **ENQUADRAMENTO** tributário:
 - **Lucro Presumido:** Aplica a alíquota de presunção (ex: `PRESUNCAO = 0.08`) sobre o faturamento bruto total para obter a base de cálculo, e então aplica as alíquotas de IRPJ e CSLL (`ALIQUOTA_IRPJ`, `ALIQUOTA_CSLL`).
 - **Lucro Real:** Aplica uma margem estimada (`ESTIMATED_MARGIN`) para simular o lucro contabilístico antes de aplicar as alíquotas.
- **Geração de Saídas (Outputs):**
 - Salva todos os resultados no diretório `output`.

- **Resumo Principal:** Gera `resumo_apuracao.json` e `resumo_apuracao.csv`, contendo os totais de faturamento bruto e o valor estimado para cada um dos seis tributos calculados.
- **Pivôs de Análise (CSVs):** Gera arquivos CSV totalizados por dimensão, essenciais para a análise do agente:
 - `faturamento_por_mes.csv`
 - `faturamento_por_uf_emitente.csv`
 - `faturamento_por_cfop.csv`
 - `faturamento_por_ncm.csv`

6.2. Componente 2: Agente de IA (`csvx.py`)

Este módulo define o agente inteligente que interpreta os resultados do motor de cálculo.

- **Framework e Modelo:** Utiliza o `Agno SDK` para definir um `Agent` movido pelo modelo `OpenAIChat(id="gpt-5-mini")`.
- **Base de Conhecimento (Knowledge):**
 - O agente é conectado a uma base de conhecimento vetorial (`LanceDb`) e um banco de conteúdos (`SqliteDb`).
 - No início, o script `csvx.py` alimenta essa base com arquivos Markdown (`*.md`) de um diretório local chamado "tributaria", aplicando a metatag "Reforma tributária". Isso permite ao agente consultar informações contextuais sobre a legislação.
- **Ferramentas (Tools):**
 - O agente é equipado com a ferramenta `CsvTools`, que é configurada para ler os arquivos CSV do diretório `output` (onde `apuracao.py` salvou os resultados).
- **Instruções e Personalidade (Prompt Engineering):**
 - O agente é instruído a atuar como um "consultor tributário especializado em análise fiscal" e na "nova Reforma tributária Brasileira".
 - As suas diretrizes são claras: analisar os dados, gerar um resumo estratégico, focar em tributos de maior impacto, oportunidades de economia e riscos fiscais.
 - Crucialmente, ele é instruído a usar "linguagem clara, objetiva e voltada para executivos, sem listar comandos técnicos" e a formatar a saída em Markdown.

6.3. Componente 3: Interface do Utilizador (`app_gradio.py`)

Este módulo une os dois componentes anteriores numa aplicação web funcional.

- **Interface Gráfica:** Utiliza `gradio` (importado como `gr`) para criar uma interface `gr.Blocks`. A interface é dividida em duas abas: "Parâmetros da Apuração" e "Resultados".

- **Entradas de Utilizador (Parâmetros):**
 - A primeira aba contém um conjunto de `gr.Slider` para ajuste fino das alíquotas (ICMS, ISS, PIS, COFINS, IRPJ, CSLL) e da percentagem de presunção de lucro.
 - Um `gr.Dropdown` permite ao utilizador selecionar o `enquadramento` tributário (`lucro_presumido`, `lucro_real`, `simples_nacional`).
- **Fluxo de Execução (Operação):**
 - O utilizador clica no botão "Executar Apuração e Análise".
 - Isso dispara a função `executar_apuracao_e_agent`, que exibe uma barra de progresso (`gr.Progress`).
 - A função atualiza dinamicamente as variáveis de alíquota globais no módulo `apuracao` (ex: `ap.ALIQ_ICMS_DEFAULT = aliq_icms`).
 - Conecta-se ao PostgreSQL (usando configurações de `DB_CONFIG`) e executa `ap.process_from_postgres(...)`.
 - A interface aguarda a criação dos arquivos de saída (ex: `resumo_apuracao.json`) usando uma função `esperar_arquivo`.
 - **Geração do Gráfico:** Lê o `faturamento_por_mes.csv` com `pandas` e usa `plotly.express(px.bar)` para gerar um gráfico de barras do faturamento mensal.
 - **Execução do Agente:** Importa o agente (`csvx.agent`) e o executa (`agent.run(...)`) com um prompt específico, solicitando a análise dos arquivos gerados.
- **Exibição de Resultados:**
 - Os resultados da função são enviados para os componentes da aba "Resultados":
 - O conteúdo do `resumo_apuracao.json` é exibido num bloco `gr.Code`.
 - O gráfico Plotly é renderizado num componente `gr.Plot`.
 - A resposta em texto do agente de IA é exibida numa `gr.Textbox` (Análise do Agente).

7. Link para o Repositório do Github

Link: <https://github.com/ericfloriano/processon-final-project>