

ENTREGAS DESTA RESPOSTA

1. **Fluxo recomendado completo da solução** (arquitetura + etapas)
 2. **Requisitos funcionais (RF)**
 3. **Regras de Negócio (RN)**
 4. **Estrutura recomendada do projeto no Replit**
 5. **Descrição técnica do backend (Node.js)** ideal para Replit
-



1. FLUXO RECOMENDADO – ARQUITETURA DA SOLUÇÃO



Visão geral

O sistema deve:

1. Receber um arquivo-template (DOCX, XLSX, CSV ou PDF).
2. Identificar o tipo do arquivo.
3. Ler placeholders dentro desse arquivo.
4. Gerar análise da transcrição via GPT (JSON estruturado).
5. Inserir o conteúdo gerado nos placeholders.

6. Gerar um arquivo final com o mesmo layout, pronto para download.
-



Fluxo Detalhado

1) Upload do Template

- O usuário envia um arquivo modelo.
- O backend salva temporariamente e detecta o tipo (docx, xlsx, csv, pdf).

2) Upload da Transcrição

- O usuário envia texto
- O backend envia a transcrição ao GPT para extrair:
 - regras de negócio
 - matriz de risco
 - encaminhamentos
 - pontos de ação
 - outras estruturas necessárias

GPT retorna um **JSON** padronizado.

3) Processamento por Tipo de Template

→

Se DOCX

Usar docxtemplater:

1. Carrega o template

2. Lê os placeholders
3. Preenche com o JSON
4. Gera um novo DOCX
5. (Opcional) Converte para PDF

→

Se XLSX

Usar exceljs:

1. Percorre células
2. Substitui placeholders
3. Recalcula fórmulas (automático)
4. Gera novo XLSX

→

Se CSV

1. Lê como texto
2. Substitui placeholders direto
3. Salva novo CSV

→

Se PDF

(Apenas PDF com campos editáveis – AcroForm)

Usar pdf-lib:

1. Abre o PDF
2. Localiza campos (form.getTextField())
3. Preenche valores

4. Regera PDF

4) Retorno ao Usuário

- O backend envia o arquivo final para download.
 - Pode gerar:
 - .docx
 - .xlsx
 - .csv
 - .pdf
-



2. REQUISITOS FUNCIONAIS (RF)

A seguir, requisitos funcionais claros para documentar a aplicação:



RF 001 — Upload de modelos

O sistema deve permitir que o usuário envie um arquivo-template nos formatos:

- DOCX
- XLSX
- CSV
- PDF (preenchível)



RF 002 — Upload da transcrição

O sistema deve permitir que o usuário envie:

- texto puro



RF 003 — Análise da transcrição via GPT

O sistema deve enviar a transcrição para a IA e receber como saída um **JSON estruturado**, contendo:

```
{
  "REGRAS_NEGOCIO": "...",
  "MATRIZ_RISCO": "...",
  "ENCAMINHAMENTOS": "...",
  "PONTOS_ACAO": "..."
}
```



RF 004 — Detecção automática do tipo de arquivo

O backend deve identificar automaticamente o tipo do template usando:

- extensão
- MIME type



RF 005 — Identificação de placeholders

O sistema deve localizar placeholders no template com o padrão:

{{NOME_DO_CAMPO}}



RF 006 — Preenchimento do template

O sistema deve substituir cada placeholder encontrado pelo valor correspondente no JSON vindo do GPT.



RF 007 — Geração do arquivo final

O sistema deve gerar um arquivo final preservando layout e estrutura do template.



RF 008 — Disponibilização do arquivo

O sistema deve permitir que o usuário baixe o arquivo final.



RF 009 — Tratamento de erros

O sistema deve informar erros em casos como:

- placeholders não encontrados
- JSON incompleto

- template corrompido
-

3. REGRAS DE NEGÓCIO (RN)

RN 01 — O template deve conter placeholders

Se o template não possuir placeholders, o sistema deve:

- gerar um relatório vazio apenas com seções existentes.
-

RN 02 — Placeholder sem valor deve ser removido

Caso um campo vindo do GPT esteja vazio:

{{CAMPO}} → ""

RN 03 — Layout original deve ser preservado

O sistema **não pode** alterar:

- fontes
- cores
- logos
- margens
- tabelas
- alinhamentos

Somente substituir texto.

RN 04 — PDF só pode ser preenchido se tiver AcroForm

Se o PDF **não tiver campos editáveis**, exibir erro:

"Este PDF não permite preenchimento automático. Use DOCX ou XLSX."

RN 05 — JSON deve corresponder aos placeholders

Se o template possui:

{{REGRAS_NEGOCIO}}

mas o JSON não tem REGRAS_NEGOCIO DEVE: preencher o documento com “Campo não encontrado”.

RN 06 — Tipos de conteúdo

- Em DOCX, conteúdo pode conter HTML básico.
 - Em XLSX, células não devem receber HTML, apenas texto.
 - Em CSV, somente texto plano.
-