Dados Gerais do Projeto

Nome do Projeto	
Automação do Processo de Licitação e Gestão de Contratos	
Patrocinador do Projeto	
Caio Sotero Rosa	Justiça Federal / Tribunal Regional Federal 1° Região / Núcleo Administrativo / Cepol / Selit
Equipe do projeto	
Nome	**Função/Papel**
Felipe Galvão Lagares	Analista de Sistemas/Desenvolvedor
Hailton David Lemos	Analista de Sistemas/Desenvolvedor
Raphael Abenom	Analista de Sistemas/Desenvolvedor

Versão do PGP	Data	Responsável	Descrição
1	28/11/2024	Equipe do projeto	Criação do Plano Geral do Projeto

1. Escopo do Projeto

A Justiça Federal da Primeira Região enfrenta desafios significativos na Gestão de Contratos, principalmente devido à dependência de planilhas Excel, que exigem esforços manuais e repetitivos em cada etapa. Essa abordagem resulta em processos lentos, sem mecanismos estruturados de acompanhamento, suscetíveis a erros humanos e marcados por ineficiências que comprometem diretamente a agilidade e a confiabilidade das operações.

Atualmente, o fluxo de trabalho baseia-se em diversas planilhas e documentos para gerenciar e conferir dados de pessoal e informações de empresas contratadas. Esses dados servem como base para decisões críticas relacionadas à liberação ou retenção de verbas de contingenciamento. Embora funcional, esse método é inadequado frente à crescente demanda por precisão, celeridade e transparência.

Para resolver esses problemas, o projeto da Gestão de Contratos busca implementar uma solução tecnológica avançada, utilizando Inteligência Artificial para análise de contratos, automação de cálculos e criação de dashboards interativos que apoiem a tomada de decisões de forma eficiente e confiável.

Diante desse contexto, o projeto de Automação do Processo de Licitação também se propõe a desenvolver uma solução integrada e automatizada, utilizando a linguagem Python e suas bibliotecas, como pyautogui, flaks, pyodbc, base64, bs4, pil, fpdf, pypdf2, transformers, googletrans, pandas, graphviz, fitz, binascii, codecs, hashlib, cryptography, pprint, pdf2image, pyodbc, qrcode, pdfplumber, shutil, pdfminer, re, numpy, além de Node.js e suas bibliotecas, como Express, Chart.js, TensorFlow.js, Puppeteer e D3.js. Essa combinação agrega capacidades avançadas de storylining, inteligência artificial e gráficos dinâmicos, permitindo uma abordagem híbrida e inovadora, que combina a robustez no processamento de dados com a escalabilidade e interatividade nas interfaces.

O projeto de Gestão de Contratos será desenvolvido em cinco etapas principais: planejamento, desenvolvimento, testes e validação, implementação e treinamento com suporte técnico. Cada etapa incluirá atividades específicas, como levantamento de requisitos, desenvolvimento das

interfaces frontend e backend, validação de modelos de IA, configuração do ambiente de produção e capacitação dos usuários.

Com essa abordagem estruturada e o uso combinado das tecnologias mais avançadas, espera-se transformar a Gestão de Contratos em um processo ágil, seguro e escalável, alinhado às necessidades operacionais e estratégicas da Justiça Federal da Primeira Região, promovendo maior eficiência, confiabilidade e aderência às melhores práticas de governança pública.

O projeto de automação do processo de licitação e gestão de contratos apresenta um potencial significativo para transformar a gestão pública. A Justiça Federal da Primeira Região enfrentará um cenário mais eficiente, transparente e ágil. Com a implementação de tecnologias inovadoras, como Inteligência Artificial e automação de processos, espera-se:

- Redução do tempo gasto em atividades repetitivas em 85%.
- Minimização de erros humanos em 90%.
- Maior confiabilidade e rastreabilidade das informações contratuais

2. Apresentação do Projeto

2.1 Objetivo Principal

Automatizar o processo de gestão de contratos, eliminando o uso de macros em planilhas Excel e integrando funcionalidades baseadas em inteligência artificial (IA) para análise contábil, previsões financeiras e operacionais, e gestão eficiente de pessoas e produtos.

Os projetos apresentam uma abordagem inovadora para superar a limitação da falta de integração entre os sistemas, assegurando um fluxo de trabalho mais fluido e eficaz para todos os envolvidos no processo licitatório.

2.1.1 Benefícios Específicos para os Stakeholders

- **Gestores**:
  - Acompanhamento em tempo real do status dos contratos.
  - Dashboards interativos para tomada de decisão baseada em dados.
- **Usuários Finais**:
  - Redução do tempo em tarefas manuais.
  - Exportação de relatórios em formatos versáteis (PDF, Excel).
- **TI**:
  - API documentada para integração com outros sistemas.
  - Logs centralizados para auditoria e análise de erros.

2.1.2 Fluxo dos Processos de Licitação e Gestão de Contratos

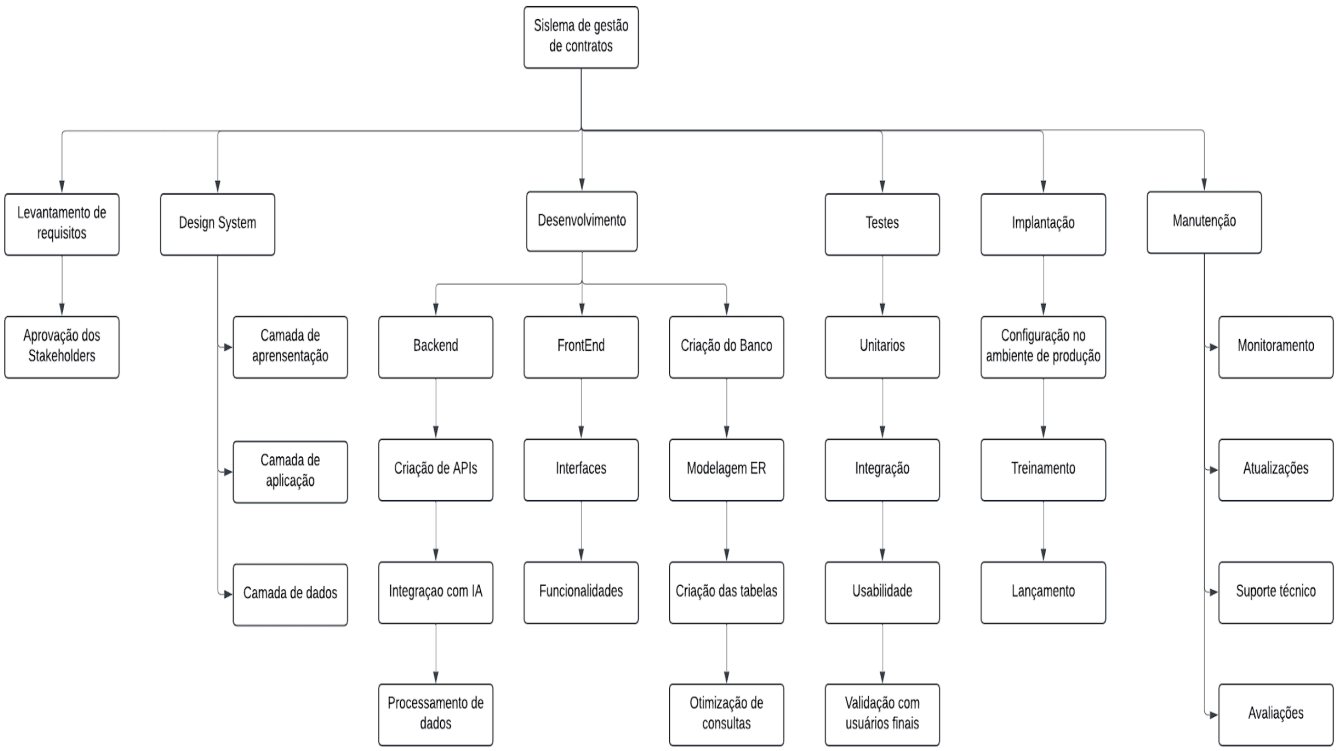

Figura 1 - Processo Licitatório

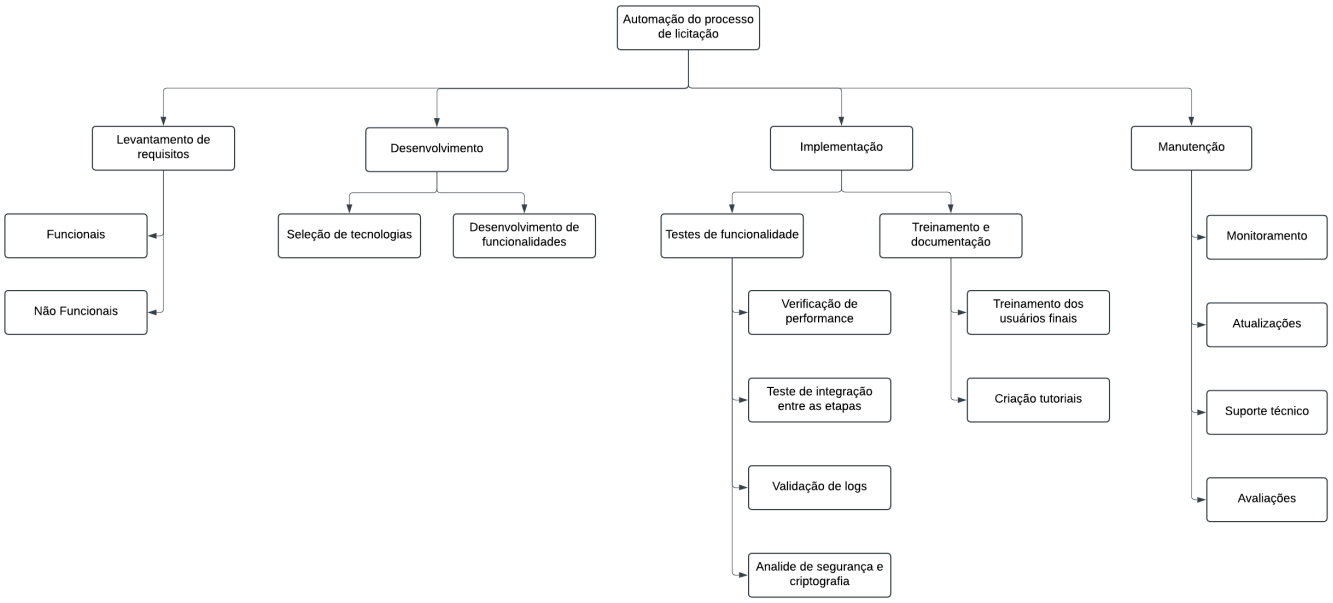

Figura 2 - Gestão de contratos.

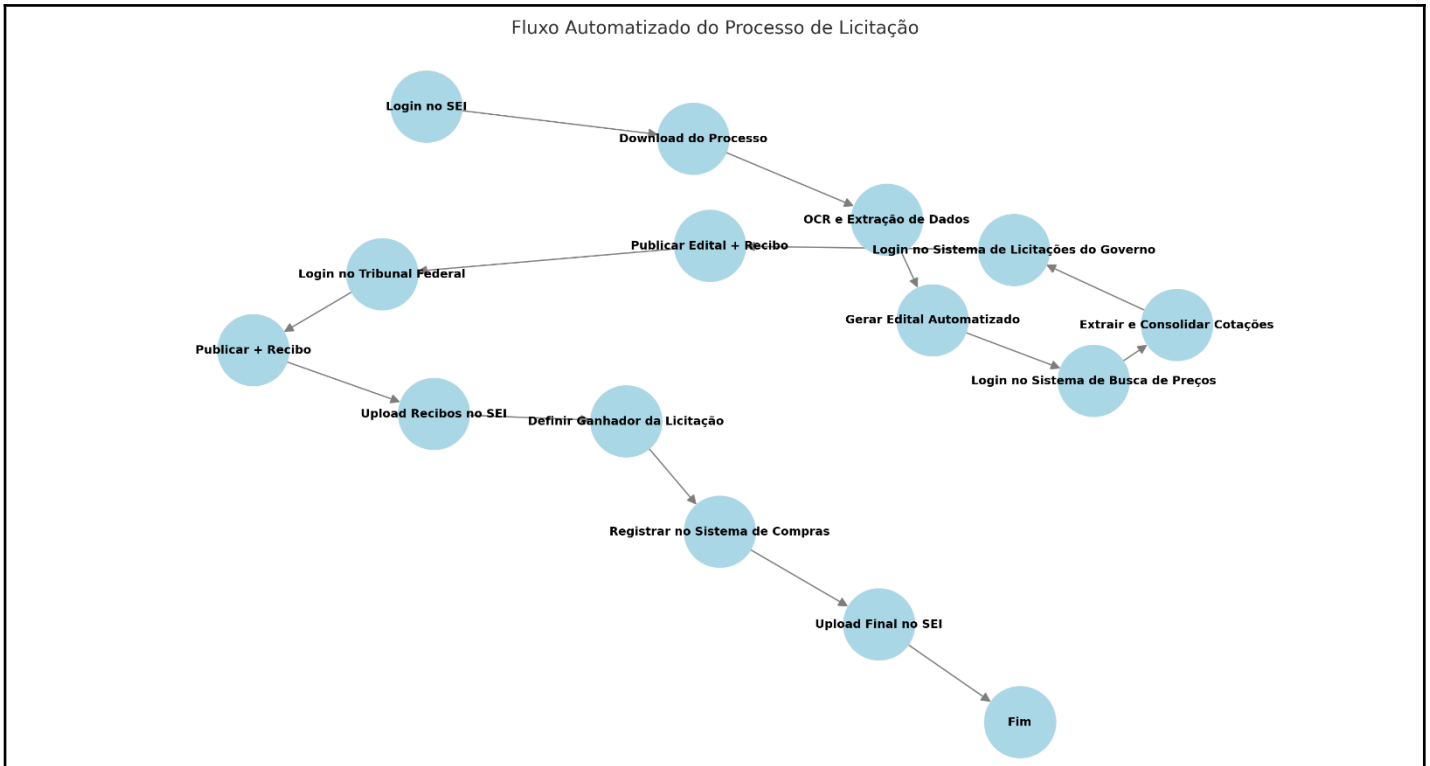

Figura 3 – Fluxo automatizado do Processo de Licitação Automatizado.

Automatizar o processo de licitação da Justiça Federal Primeira Região, reduzindo o tempo gasto com operações manuais e aumentando a precisão no preenchimento e movimentação de dados entre sistemas que atualmente não se comunicam.

2.2 Levantamento de Requisitos Funcionais

2.2.1 - Acesso e Manipulação de Sistemas:

- Automatizar login nos sistemas: SEI, sistema de busca de preços, sistema de licitações do governo federal, sistema do tribunal federal e sistema de compras do governo.
- Realizar o download/upload de documentos nos sistemas.
- Navegar automaticamente pelas interfaces gráficas para realizar tarefas específicas.

2.2.2 - Processamento de Dados:

- Ler PDFs para extrair informações relevantes (como requisitos da licitação).
- Preencher formulários automaticamente nos sistemas baseados nos dados extraídos.
- Gerar relatórios de cotação e edital automaticamente.

2.2.3 - Fluxo Automático de Etapas:

- Integrar o processo completo em uma única execução automatizada: desde a entrada no SEI até a publicação da licitação e atualização do pedido de material/serviço.
- Gerenciar notificações e verificações, garantindo que etapas obrigatórias sejam concluídas antes de avançar.

2.2.4 - Gerenciamento de Erros e Logs:

- Implementar captura de erros (ex.: elemento da interface não encontrado).
- Registrar logs detalhados para auditoria.

2.3 Levantamento de Requisitos Não Funcionais

2.3.1 - Performance:

- Garantir que o processo de automação seja rápido o suficiente para não impactar a rotina operacional.

2.3.2 - Segurança:

- Proteger credenciais de login utilizando variáveis de ambiente ou arquivos criptografados.

2.3.3 - Portabilidade:

- Criar um código flexível que possa ser ajustado para alterações futuras nos sistemas.

2.4 Solução Proposta

2.4.1 - Tecnologias

- **Python**: Linguagem principal para automação.
- **PyAutoGUI**: Para automação da interface gráfica (cliques, navegação, e interação com janelas).
- **PyPDF2** ou **PDFPlumber**: Para extração de texto dos PDFs de requisitos.
- **OpenPyXL** ou **Pandas**: Para manipulação de dados tabulares, como tabelas de cotação.
- **Tesseract OCR (via Pytesseract)**: Para extração de texto em partes não editáveis de documentos, caso necessário.

- **Logging**: Para capturar eventos e erros.
- **Schedule**: Para gerenciar fluxos automáticos com agendamento, caso o processo seja recorrente.

2.5 Fluxo Automatizado

2.5.1 - Etapa 1: Início no Sistema SEI

- Logar automaticamente no **Sistema SEI**.
- Navegar até o processo provocador da licitação.
- Fazer o download do arquivo PDF com os requisitos.

2.5.2 - Etapa 2: Preparação do Edital e Cotação

- Ler e processar o PDF para identificar informações relevantes (itens, quantidades, descrições).
- Logar no sistema de busca de preços:
  - Preencher os campos automaticamente para consulta de preços.
  - Gerar e salvar o relatório de cotação.
- Consolidar os dados da cotação e gerar o edital automaticamente.

2.5.3 - Etapa 3: Publicação da Licitação

- Logar no sistema de licitações do governo federal:
  - Publicar o edital.
  - Salvar o recibo de publicação.
- Logar no sistema do Tribunal Federal:
  - Publicar o edital.
  - Salvar o recibo de publicação.

2.5.4 - Etapa 4: Finalização no SEI

- Retornar ao **Sistema SEI**:
  - Fazer upload dos recibos de publicação.
  - Atualizar o status do processo.

2.5.5 - Etapa 5: Pedido de Material/Serviço

- Quando o ganhador da licitação for definido:
  - Logar no sistema de compras do governo.
  - Registrar o pedido de material ou serviço.
- Atualizar novamente o **Sistema SEI** com os arquivos gerados.

2.6 Automação Utilizando PyAutoGUI

2.6.1 - Principais Ações

- **Login Automatizado**:
  - Localizar campos de usuário/senha via reconhecimento de tela (pyautogui.locateOnScreen) e preencher.
  - Aguardar o carregamento das telas com pyautogui.sleep().
- **Navegação e Downloads/Uploads**:
  - Automação para clicar em botões e menus utilizando imagens de referência (print da interface).
  - Abertura e salvamento de arquivos via atalhos (Ctrl+S).
- **Preenchimento de Formulários**:
  - Extração dos dados necessários dos PDFs e preenchimento dos campos utilizando pyautogui.typewrite().
- **Processamento de Arquivos**:
  - Processar PDFs com bibliotecas de manipulação e OCR, consolidar os dados em tabelas usando Pandas.

2.7 Desafios e Considerações

- **Mudanças na Interface dos Sistemas**: A automação baseada em interface gráfica é sensível a alterações nas telas.

- Mitigação: Usar templates robustos de reconhecimento visual e criar scripts parametrizados.

- **Manuseio de Erros**: Garantir que o script pause e notifique o operador caso algum elemento não seja encontrado.
  - Mitigação: Implementar verificações automáticas para confirmar sucesso em cada etapa.
- **Treinamento e Acompanhamento**: Treinar os usuários para validar os resultados iniciais da automação.

2.8 Benefícios Esperados

- Redução do tempo gasto em tarefas repetitivas.
- Minimização de erros humanos em processos manuais.
- Centralização de logs e relatórios para auditoria e análise.
- Melhor rastreabilidade de todo o processo.

2.9 Soluções Adicionais com IA e Ferramentas Avançadas

Para aprimorar ainda mais a automação, podemos incorporar **Inteligência Artificial (IA)**. Essa tecnologia pode ser usada para substituir ou complementar as ações baseadas no pyautogui e adicionar robustez e eficiência à solução.

2.10 Sugestões de Tecnologias Avançadas

2.10.1 - Reconhecimento de Caracteres e Extração de Dados

- **OCR com Tesseract ou AWS Textract**:
  - Detectar e extrair texto de PDFs escaneados ou imagens de documentos.
  - Facilitar a leitura de recibos, relatórios e documentos estruturados.
- **Redes Neurais com Pytesseract ou EasyOCR**:

- Reconhecer caracteres em telas complexas ou formatos não padrão, como captchas ou formulários com baixa qualidade de imagem.

2.10.2 - Conversão de PDF para Excel

- **Tabula ou Camelot**:
  - Extrair tabelas diretamente de PDFs e convertê-las em planilhas do Excel.
- **Pandas com OpenPyXL**:
  - Processar tabelas e gerar arquivos Excel automaticamente, integrando dados de cotações e relatórios.

2.10.3 - Visão Computacional

- **OpenCV**:
  - Identificar elementos visuais na tela para melhorar a navegação e automação baseada em reconhecimento visual.
- **YOLOv5/YOLOv8**:
  - Detectar campos específicos em interfaces gráficas, como botões, tabelas ou formulários.

2.10.4 - Inteligência Artificial para Processamento de Dados

- **Modelos de Machine Learning**:
  - Treinar um modelo simples para prever preenchimentos baseados em dados históricos.
- **GPT ou BERT**:
  - Gerar resumos automáticos de relatórios.
  - Sugerir melhorias para os editais baseados em normas e boas práticas.

2.10.5 - Workflow e Orquestração

- **Apache Airflow ou Prefect**:
  - Gerenciar fluxos de trabalho automatizados, garantindo que cada etapa seja concluída antes da próxima.

2.10.6 - Logs e Relatórios com IA

- **ElasticSearch/Kibana**:

- Monitorar os logs da automação em tempo real e identificar possíveis gargalos.

- **IA para Análise de Logs**:
  - Usar modelos de aprendizado para detectar padrões de erros repetitivos e sugerir melhorias.

2.11 Fluxo do Processo Automatizado

2.11.1 - Etapa: Inicialização no Sistema SEI

- **Login Automático**:
  - O sistema realiza login automaticamente no SEI usando pyautogui ou automação via WebDriver.
- **Extração do Processo**:
  - Baixar o documento do processo no SEI.
  - Usar **OCR com Tesseract** para identificar informações críticas no PDF, como requisitos da licitação.

2.11.2 - Etapa: Preparação do Edital

- **Extração e Organização de Dados**:
  - Converter o PDF em tabelas com **Tabula** ou **Camelot**.
  - Processar as tabelas com **Pandas** e preencher automaticamente o modelo de edital.
- **Gerar Edital Automatizado**:
  - Gerar o documento do edital com **Python-docx** ou **ReportLab**.

2.11.3 - Etapa: Cotação de Preços

- **Login no Sistema de Busca de Preços**:
  - Automatizar o login e preenchimento dos campos de busca.
- **Extração de Dados de Cotações**:
  - Capturar resultados com **OCR** ou manipular as tabelas retornadas.
  - Gerar o relatório consolidado das cotações com **Pandas**.

2.11.4 - Etapa: Publicação da Licitação

- **Publicação nos Sistemas**:
  - Automatizar o upload do edital nos sistemas de licitações do governo e tribunal federal.
  - Capturar recibos de publicação usando **Visão Computacional** para identificar botões ou ícones de download.
- **Upload no SEI**:
  - Subir os recibos de publicação no SEI automaticamente.

2.11.5 - Etapa: Finalização e Pedido

- **Registro do Pedido no Sistema de Compras**:
  - Automatizar o registro do pedido, preenchendo formulários automaticamente.
- **Atualização Final no SEI**:
  - Subir os arquivos finais no SEI para documentar o encerramento do processo.

2.11.6 - Fluxograma dos Processos

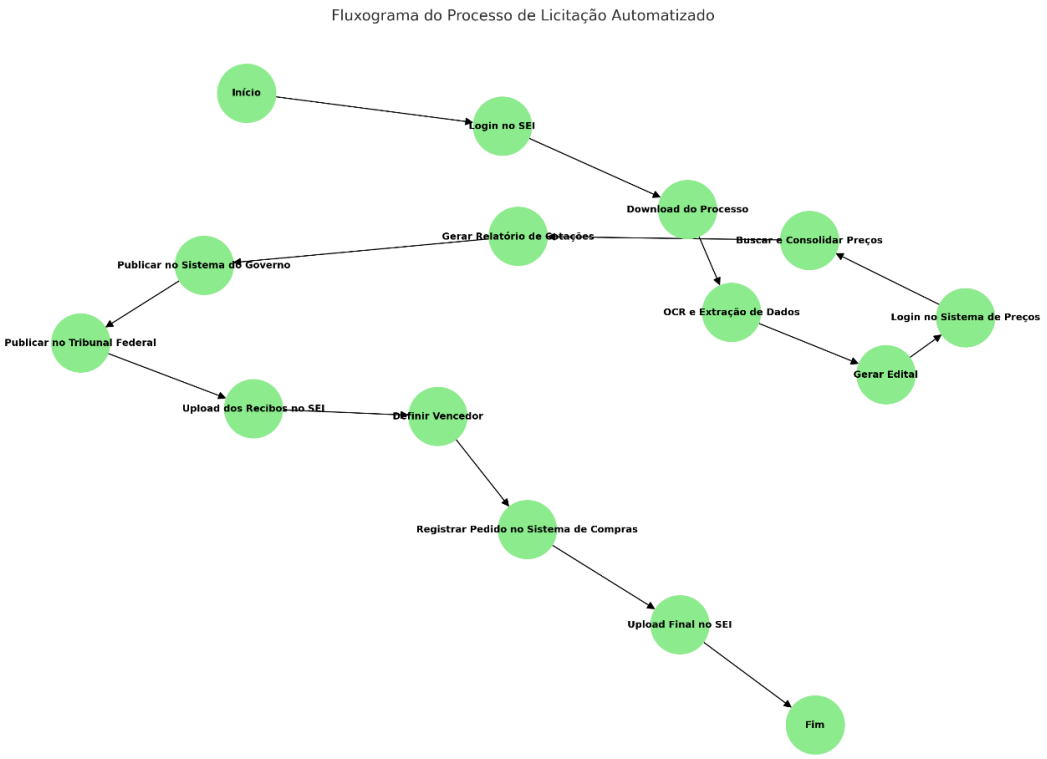

Figura 4 – Fluxograma do Processo de Licitação Automatizado.

2.12 Benefícios com IA

- **Velocidade e Precisão**:
  - Automação robusta com ferramentas baseadas em IA.
- **Redução de Erros**:
  - OCR e visão computacional evitam erros manuais ao ler documentos.
- **Escalabilidade**:
  - Fluxo modular e adaptável para novos requisitos.
- **Relatórios Inteligentes**:
  - IA pode sugerir melhorias nos documentos gerados.

2.13 Descrição do Processo para Automação Usando Node.js e Python

A integração de **Node.js** e **Python** será estruturada para que o **Node.js** atue como servidor principal, responsável por receber requisições HTTPs, enquanto o **Python** realizar os processamentos complexos, como OCR, uso de LLMs e outras automações, rodando em background e sendo acionado por comandos específicos.

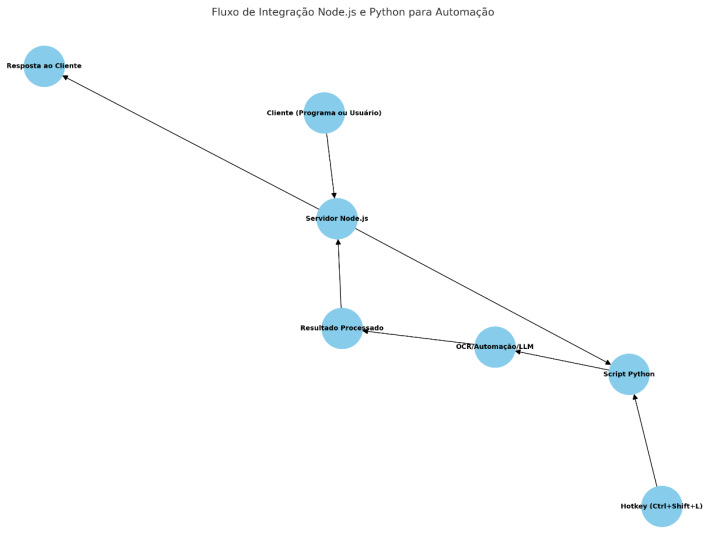

Figura 5 – Fluxo de Integração.

2.14 Arquitetura do Sistema

Componentes Principais

- **Servidor Node.js**:
  - Serve como o ponto de entrada para requisições HTTPS.
  - Gerência autenticação e autorização (TLS/SSL para segurança).
  - Encaminha requisições para o programa Python via chamadas locais ou API.
  - Responde ao cliente com o resultado do processamento.
- **Script Python**:
  - Realiza as tarefas de processamento pesado:
    - OCR com Tesseract ou AWS Textract.
    - Automação com PyAutoGUI e outras bibliotecas.
    - Uso de LLMs para análises e respostas.
  - Funciona como um serviço que escuta comandos enviados pelo Node.js.
  - É ativado diretamente por um **hotkey** configurado no sistema.
- **HTTPS**:
  - Comunicação segura entre cliente e servidor (Node.js).
  - Permite integração com sistemas externos e ferramentas de monitoramento.
- **Hotkey para Acionamento**:
  - Uma combinação de teclas (ex.: Ctrl+Shift+L) será configurada para iniciar o script Python via um programa de monitoramento rodando no cliente.

2.15 Fluxo do Processo

2.15.1 - Inicialização

- O servidor **Node.js** é iniciado e começa a escutar requisições HTTPS.

- O script Python também é iniciado como um serviço de segundo plano.

2.15.2 - Recebimento da Requisição

- Um evento externo (programa ou comando via cliente) faz uma requisição HTTPS ao servidor Node.js.
- Exemplo de requisição: POST /process com payload contendo informações do processo a ser automatizado.
- O servidor valida a requisição e a encaminha para o serviço Python.

2.15.3 - Processamento em Python

- O serviço Python recebe a solicitação, realiza os seguintes passos:
  - Executar OCR nos documentos enviados (se necessário).
  - Usa visão computacional ou automação para interagir com sistemas baseados em interface gráfica (PyAutoGUI).
  - Realiza processamento adicional com LLMs para gerar respostas ou resumos.
  - Retorna os resultados ao servidor Node.js.

2.15.4 - Resposta ao Cliente

- O Node.js recebe o resultado do script Python.
- Formata a resposta em um JSON estruturado.
- Envia o resultado ao cliente via HTTPS.

2.15.5 - Hotkey para Processamento Manual

- O usuário pode acionar manualmente o script Python usando uma combinação de teclas.
- O script Python executa o processo completo de automação baseado no estado atual.

2.16 Tecnologias Utilizadas

2.16.1 - Servidor Node.js

- **Express.js**:
  - Framework para construir e gerenciar rotas HTTP.
- **HTTPS/TLS**:

  - Para garantir segurança nas conexões.

- **Body-Parser**:
  - Para processar payloads JSON enviados nas requisições.
- **Nodemon**:
  - Para monitorar e reiniciar o servidor durante o desenvolvimento.

Processamento Python

- **PyAutoGUI**:
  - Automação de interações em sistemas não integrados.
- **PyPDF2/Pytesseract**:
  - Para manipulação de PDFs e OCR.
- **LangChain ou OpenAI API**:
  - Para análise de textos com LLMs.
- **Keyboard Library**:
  - Para configurar atalhos de teclado para acionar processos.

2.17 Comunicação Node.js e Python Visão Computacional Determinística

A **visão computacional determinística** refere-se ao uso de técnicas pré-definidas e algoritmos baseados em regras explícitas para processar e interpretar imagens digitais. Essas técnicas não aprendem com os dados, mas dependem de métodos matemáticos, estatísticos ou geométricos para extrair informações de imagens.

2.17.1 - Características da Visão Computacional Determinística

- **Regras Fixas:**
  - As operações são programadas para seguir passos explícitos, como filtragem de imagens, detecção de bordas ou análise de padrões.
  - Exemplo: Aplicar um filtro de Sobel para detectar bordas ou calcular a média de cores em um bloco de pixels.

- **Determinismo:**
  - Dada uma entrada, o programa sempre produzirá a mesma saída, desde que as condições não mudem.
  - Não há aprendizado ou ajuste dinâmico baseado em dados.
- **Eficiência Computacional:**
  - Métodos determinísticos geralmente são mais rápidos e simples do que técnicas de aprendizado de máquina, especialmente para tarefas bem definidas.

2.17.2 - Exemplos Comuns de Técnicas Determinísticas:

- **Cálculo de estatísticas de cor:**
  - Usar a média ou desvio padrão de cores para identificar similaridades entre regiões de uma imagem.
  - Exemplo: Determinar se um bloco de pixels tem características semelhantes ao padrão fornecido.
- **Transformação de Imagem:**
  - Aplicar transformações como binarização, thresholding, suavização ou morfologia matemática.
- **Template Matching:**
  - Buscar uma subimagem (template) dentro de uma imagem maior, comparando blocos com base em correlação ou diferenças.

2.17.3 - Cálculo de Média de Cores

O cálculo da média de cores é uma abordagem simples e eficiente para comparar padrões em uma imagem:

- Dividir a imagem maior em blocos do mesmo tamanho que o padrão.
- Calcular a média (ou outro estatístico) das intensidades de cor (RGB ou tons de cinza) em cada bloco.
- Comparar a média de cada bloco com a média do padrão para determinar semelhança.

Isso foi implementado no programa anexado para detectar padrões semelhantes ao modelo na imagem maior.

2.17.4 - Agrupamento por heurística

Agrupamento por heurística refere-se à aplicação de regras fixas para organizar ou agrupar dados sem o uso de algoritmos adaptativos ou baseados em aprendizado.

2.17.5 - Como Funciona no Programa

- **Coordenadas das Detecções:**
  - Após identificar regiões da imagem que correspondem ao padrão, o programa gera um conjunto de coordenadas correspondentes ao centro dos blocos detectados.

- **Regras para Agrupamento:**
  - Uma métrica, como a **distância euclidiana**, é usada para verificar se dois pontos estão "próximos" o suficiente para serem considerados parte do mesmo grupo.
  - Parâmetros fixos, como group_threshold, definem o limite de proximidade para agrupar pontos.
- **Passos no Agrupamento:**
  - Iterar sobre todas as coordenadas.
  - Para cada coordenada, verificar se há outras dentro de uma distância limite.
  - Calcular a posição central do grupo (geralmente a média das coordenadas de cada grupo).

2.17.6 - Diferença para Algoritmos de ML (Ex.: k-means, DBSCAN)

- **Heurística:**
  - Baseia-se em regras explícitas e fixas para agrupar coordenadas.
  - Não há adaptação automática às características dos dados.
- **k-means:**
  - Agrupamento iterativo que minimiza a distância entre pontos e centros (clusters).
  - Os centros dos clusters são ajustados dinamicamente.
- **DBSCAN:**
  - Agrupamento baseado em densidade. Agrupa pontos que estão próximos entre si e identifica os outliers automaticamente.
  - É mais flexível para dados que não têm uma distribuição uniforme.

2.18 Visão Computacional Determinística

- **Vantagens:**
  - Simples de implementar.
  - Computacionalmente eficiente para tarefas pequenas ou específicas.
  - Não requer treinamento nem muitos dados de exemplo.
- **Limitações:**
  - Menos flexível para casos complexos ou imagens com grande variabilidade.
  - Sensível a ruídos ou pequenas alterações na imagem.
  - Regras fixas podem não generalizar bem para novos casos.

2.19 Agrupamento por Heurística

- **Vantagens:**
  - Fácil de ajustar (ex.: definir um limiar de distância).
  - Rápido para conjuntos de dados pequenos ou moderados.
- **Limitações:**
  - Depende fortemente do limiar escolhido (pode precisar de experimentação manual).
  - Não lida bem com outliers ou distribuições irregulares de dados.
  - Não possui uma noção adaptativa para dados com grande variabilidade.

O programa que você utiliza aplica **técnicas determinísticas de visão computacional** para buscar padrões e realiza o agrupamento por heurística usando regras de proximidade. Embora essas técnicas sejam rápidas e eficazes para tarefas bem definidas, não têm a flexibilidade ou a adaptabilidade de soluções baseadas em aprendizado de máquina.

2.20 Banco de dados

Um **banco de dados** é uma coleção organizada de informações que podem ser facilmente acessadas, gerenciadas e atualizadas. Ele é essencial para sistemas modernos que necessitam de alta confiabilidade, desempenho e escalabilidade. Com o uso de um Sistema de Gerenciamento de Banco de Dados (SGBD), como MySQL, PostgreSQL ou Oracle, é possível armazenar, consultar e manipular dados de forma eficiente.

No contexto de sistemas corporativos e institucionais, os bancos de dados desempenham um papel crucial, permitindo:

- **Organização de Dados**: Estruturar informações em tabelas relacionadas.
- **Confiabilidade**: Garantir a consistência dos dados por meio de transações seguras.
- **Escalabilidade**: Suportar grandes volumes de dados com eficiência.
- **Facilidade de Acesso**: Proporcionar interfaces de consulta e relatórios dinâmicos.

2.21 Diagramas

- **Diagrama de Entidade-Relacionamento (ER)**
  - Esse diagrama mostra as tabelas e seus relacionamentos (chaves primárias e estrangeiras).
- **Diagrama de Classe**
  - Representa as tabelas como classes com atributos, tipos de dados e relacionamentos (associações, multiplicidade).

- **Diagrama de Fluxo de Dados (DFD)**
  - Mostra o fluxo de dados entre os processos e tabelas, representando a entrada, o processamento e a saída dos dados.
- **Diagrama Relacional**
  - Este diagrama vai mostrar como as tabelas se inter-relacionam com chaves primárias e estrangeiras.
- **Diagrama de Uso**
  - Representa as interações entre os atores (usuários, sistemas externos) e as tabelas em um sistema.
- **Diagrama de Normalização**
  - Representa o processo de normalização das tabelas, destacando as dependências e as formas normais (1FN, 2FN, 3FN).
- **Diagrama de Sequência**
  - Mostra como as tabelas interagem com base em uma sequência de operações, indicando o fluxo de trabalho ou interações entre elas.

2.22 Uso de API

- **Opção 1: API Local**
  - O Python pode atuar como um servidor de API separado (usando Flask ou FastAPI).
  - O Node.js envia requisições HTTP para o serviço Python.
- **Opção 2: Chamadas Diretas**
  - O Node.js utiliza bibliotecas como child_process para executar o script Python diretamente.
  - O Node.js espera o resultado do script como retorno da execução.

2.23 Benefícios da Arquitetura

- **Flexibilidade:**
  - O Node.js permite integração com sistemas externos via HTTPS.
  - O Python lida com o processamento pesado, especializado em automação e IA.
- **Segurança**:
  - Comunicação criptografada com HTTPS.
  - Separação de responsabilidades entre Node.js (gateway) e Python (processamento).
- **Escalabilidade**:
  - O servidor pode ser ampliado para suportar múltiplas requisições simultâneas.
  - O Python pode ser otimizado para lidar com tarefas paralelas usando threading ou multiprocessing.

Com essa arquitetura, o processo é centralizado, seguro e robusto, aproveitando o melhor de ambas as tecnologias.

24. Antes e Depois da Implementação

Aspecto	Situação Atual	Após Implementação
Ferramentas	Planilhas manuais e macros em Excel	Sistema automatizado com IA
Precisão nos dados	Suscetível a erros humanos	Alta precisão e consistência
Tempo de execução	Demorado e repetitivo	Otimizado e dinâmico

3. Estrutura Analítica do Projeto (EAP)

3.1. Visão Geral do Projeto

A Estrutura Analítica do Projeto (EAP) para a **Automação do Processo de Gestão de Contratos** foi criada para organizar e descrever, de forma clara e hierárquica, todos os elementos necessários para alcançar os objetivos propostos. Este projeto visa substituir o uso de planilhas manuais por um sistema automatizado e inteligente, oferecendo uma solução moderna para os desafios enfrentados pela Justiça Federal da Primeira Região.

A EAP está organizada em fases principais, que são subdivididas em componentes e pacotes de trabalho específicos.

3.1.1 - Automação do Processo de Gestão de Contratos

3.1.1.1 - **Planejamento do Projeto**

3.1.1.1.1 - Levantamento de Requisitos

3.1.1.1.2. Análise de Viabilidade Técnica

3.1.1.1.3. Planejamento de Recursos e Cronograma

3.1.1.1.4. Aprovação e Alinhamento com Stakeholders

3.1.2 - Desenvolvimento do Sistema

3.1.2.1. Design da Arquitetura do Sistema

3.1.2.2. Camada de Apresentação (Frontend)

3.1.2.3. Camada de Aplicação (Backend)

3.1.2.4. Camada de Dados (Banco de Dados)

3.1.2.5. Implementação do Backend

3.1.2.5.1. Desenvolvimento de APIs

3.1.2.5.2. Integração com Modelos de IA

3.1.2.5.3. Scripts para Processamento de Dados

3.1.2.6. Implementação do Frontend

3.1.2.6.1. Criação de Interfaces Responsivas

3.1.2.6.2. Integração de Funcionalidades Dinâmicas

3.1.2.7. Configuração do Banco de Dados

3.1.2.7.1. Modelagem do Diagrama Entidade-Relacionamento

3.1.2.7.2. Criação das Tabelas e Scripts

3.1.2.7.3. Otimização de Consultas

3.1.3. Testes e Validação

3.1.3.1. Testes de Unidade

3.1.3.2. Testes de Integração

3.1.3.3. Testes de Usabilidade

3.1.3.4. Validação com Usuários Finais

3.1.4. Implantação

3.1.4.1. Configuração de Ambiente de Produção

3.1.4.2. Migração de Dados

3.1.4.3. Treinamento de Usuários

3.1.4.4. Lançamento do Sistema

3.1.5. Monitoramento e Manutenção

3.1.5.1. Monitoramento Contínuo

3.1.5.2. Atualizações e Melhorias

3.1.5.3. Suporte Técnico

3.1.5.4. Avaliação de Desempenho do Sistema

3.2 Explicação dos Elementos Principais

- Planejamento do Projeto: Esta fase inicial é crucial para definir os requisitos, alocar recursos e obter aprovação. Serve como base para a execução de todo o projeto.
- Desenvolvimento do Sistema: Envolve o design e a implementação de todas as camadas do sistema, desde a interface com o usuário até o backend e o banco de dados. É o núcleo técnico do projeto.
- Testes e Validação: Garantem a qualidade do sistema desenvolvido, com foco em identificar e corrigir problemas antes da implantação.

- Implantação: Trata da entrega do sistema aos usuários finais, incluindo a configuração do ambiente, treinamento e migração de dados.
- Monitoramento e Manutenção: Após o lançamento, é essencial monitorar o desempenho do sistema e realizar ajustes conforme necessário para garantir sua eficiência e usabilidade a longo prazo.

4. Premissas do projeto

O projeto parte de algumas premissas importantes para seu sucesso. Presume-se que as planilhas fornecidas pelos fornecedores terão um formato padronizado, garantindo a eficiência da importação de dados. A infraestrutura tecnológica disponível será suficiente para suportar a operação do sistema. Adicionalmente, os gestores de contratos estarão disponíveis para participar de treinamentos e se adaptar às mudanças propostas. Por fim, a base de dados histórica é considerada confiável, permitindo a calibragem dos modelos de inteligência artificial.

Disponibilidade de Planilhas Padronizadas:

- As planilhas enviadas pelos fornecedores seguirão um formato mínimo padronizado, facilitando o processamento automatizado.

Engajamento dos Stakeholders:

- Os principais stakeholders (gestores e operadores do processo de contratos) estarão disponíveis para fornecer informações detalhadas durante o levantamento de requisitos e validação do sistema.

Infraestrutura Adequada:

- A Justiça Federal da Primeira Região disponibilizará os recursos de infraestrutura necessários, como servidores, licenças de software e acesso às bases de dados existentes.

Aceitação de Mudanças nos Processos:

- Os usuários finais estarão abertos a adotar novos processos, ferramentas e fluxos de trabalho automatizados, com o devido treinamento.

5. Restrições

- **Limitação de Tempo:**
  O projeto deve ser concluído dentro do prazo estipulado no cronograma (com marcos já definidos, como prototipação e validação).
- **Capacidade de Adaptação dos Usuários:**
  A curva de aprendizado dos usuários finais para as novas ferramentas pode ser uma barreira, especialmente se o treinamento for insuficiente ou a aceitação das mudanças for menor que o esperado.
- **Gestão de Segurança e Conformidade:**
  A implementação deve atender aos padrões de segurança e conformidade legal para proteger dados sensíveis, o que pode aumentar a complexidade do projeto.
- **Restrições de Testes:**
  O ambiente de testes pode não refletir integralmente a complexidade do ambiente de produção, dificultando a validação completa do sistema.

6. Cronograma Geral do Projeto

Principais marcos / eventos do projeto	Responsável	Sprint	Data de Início	Data de Término
Desenvolvimento das Atividades				
Brainstorming dos Sistema de Licitação e Gestão de Contratos;	Equipe	Sprint 1	08/11/2024	08/11/2024
Levantamento de requisitos dos Sistemas de Licitação e Gestão de Contratos junto aos stakeholders. Início da prototipação dos sistemas;	Equipe	Sprint 2	09/11/2024	24/11/2024
Definição das ferramentas que serão utilizadas no processo de desenvolvimento e dos ambientes de desenvolvimento, homologação e produção. Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 3	25/11/2024	27/11/2024
Instalação das ferramentas nos ambientes de desenvolvimento, homologação e produção para iniciar o desenvolvimento e configuração. Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 4	28/11/2024	29/11/2024
Apresentação de funcionalidades desenvolvidas para os Stakeholders;	Equipe	Sprint 5	02/12/2024	02/12/2024
Alterações e correções no servidor de aplicação. Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 6	03/12/2024	10/12/2024
Apresentação de funcionalidades desenvolvidas para os todos os Stakeholders do TRF1 e Professores Orientadores da UFG;	Equipe	Sprint 7	11/12/2024	11/12/2024
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 8	12/12/2024	31/12/2024
Primeira entrega dos artefatos desenvolvidos;	Equipe	Sprint 9	06/01/2025	06/01/2025

Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 10	07/01/2025	14/01/2025
Segunda entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 11	15/01/2025	15/01/2025
Terceira entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 12	15/01/2025	31/01/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 13	01/02/2025	14/02/2025
Quarta entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 14	15/02/2025	28/02/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 15	01/03/2025	15/03/2025
Apresentação de funcionalidades desenvolvidas para os Stakeholders;	Equipe	Sprint 16	16/03/2025	17/03/2025
Quinta entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 17	18/03/2025	31/03/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 18	01/04/2025	15/04/2025
Sexta entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 19	16/04/2025	30/04/2025
Apresentação de funcionalidades desenvolvidas para os Stakeholders;	Equipe	Sprint 20	01/04/2025	02/04/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 21	03/04/2025	25/04/2025
Sétima entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 22	26/04/2025	20/05/2025

Apresentação de funcionalidades desenvolvidas para os Stakeholders;	Equipe	Sprint 23	21/05/2025	23/05/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 24	24/05/2025	25/06/2025
Apresentação de funcionalidades desenvolvidas para os Stakeholders;	Equipe	Sprint 25	26/06/2025	28/06/2025
Oitava entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 26	29/06/2025	30/07/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 27	01/08/2025	05/08/2025
Apresentação de funcionalidades desenvolvidas para os Stakeholders;	Equipe	Sprint 28	06/08/2025	10/08/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 29	11/08/2025	25/08/2025
Nona entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 30	26/08/2025	18/09/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 31	19/09/2025	04/10/2025
Apresentação de funcionalidades desenvolvidas para os Stakeholders.	Equipe	Sprint 32	05/10/2025	08/10/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 33	09/10/2025	31/10/2025
Décima entrega dos artefatos desenvolvidos, com correções sugeridas pelos stakeholders;	Equipe	Sprint 34	01/11/2025	20/11/2025
Ajustes e correções / Seguimento no desenvolvimento da prototipação dos sistemas;	Equipe	Sprint 35	21/10/2025	28/11/2025

Apresentação de funcionalidades desenvolvidas para os Stakeholders. Passagem do Projeto para a próxima equipe que assumirá a Residência.;	Equipe	Sprint 36	29/11/2025	30/11/2025

Justificativa para Flexibilidade na Descrição das Entregas

Devido à complexidade do projeto e à natureza dinâmica do desenvolvimento dos sistemas, foi adotada uma abordagem flexível para a descrição das entregas. Essa decisão baseia-se em:

1. **Complexidade Técnica**:
   - Projetos desse porte frequentemente enfrentam desafios e requisitos adicionais que surgem apenas durante a fase de desenvolvimento, testes e validação.
2. **Evolução dos Requisitos**:
   - Durante as entrevistas iniciais e o levantamento de requisitos, é natural que algumas demandas não sejam totalmente captadas. Com isso, o processo precisa se adaptar a novas necessidades à medida que surgem.
3. **Foco na Qualidade e Eficiência**:
   - Ao invés de detalhar as entregas em um nível micro desde o início, priorizamos a entrega pontual e com alta qualidade, ajustando o escopo com base nas prioridades identificadas durante o ciclo de desenvolvimento.

Estratégia de Comunicação das Entregas

Para garantir a transparência e o alinhamento com os stakeholders:

1. **Relatórios Mensais**:
   - As entregas realizadas serão descritas detalhadamente nos relatórios mensais, permitindo um acompanhamento contínuo e atualizado das atividades desenvolvidas.
   - Esses relatórios incluirão informações sobre:
     - Funcionalidades implementadas.
     - Melhorias realizadas.
     - Problemas resolvidos.

        - Etapas planejadas para o próximo período.
2. **Revisões Periódicas**:
    - As reuniões regulares com stakeholders servirão para revisar o progresso, validar as entregas e ajustar os detalhes das próximas fases, garantindo que o projeto permaneça alinhado às expectativas.
3. **Cronograma Base**:
    - Apesar de não descrevermos as entregas em detalhes neste momento, seguimos um cronograma bem definido com marcos claros e objetivos de cada sprint, garantindo pontualidade na execução.

Compromisso com Pontualidade

Embora os detalhes das entregas possam ser ajustados conforme a evolução do projeto, há um forte compromisso com:

- **Cumprimento dos prazos estipulados no cronograma**.
- **Entrega incremental e contínua** de resultados funcionais, garantindo que os benefícios do sistema sejam percebidos ao longo do desenvolvimento.

Essa abordagem permite atender às necessidades reais do projeto, com flexibilidade e foco na entrega de valor aos stakeholders, minimizando riscos e maximizando a eficiência.

7. Histórias de Usuário

Como (papel) eu quero (algo) para (me beneficiar)
Como supervisor responsável pela liberação de verbas de contingenciamento, **quero** que os cálculos sejam realizados automaticamente para cada pessoa e empresa contratada, **a fim de** reduzir erros e agilizar as decisões financeiras.
Como supervisor responsável **quero** acompanhar o andamento do processo de contingenciamento.
Como Supervisor responsável pela liberação de contingenciamento eu **quero** que dashboards sejam exibidos **para** facilitar a visualização das informações, que, em planilhas, não são muito perceptíveis.
Como supervisor, **quero** gerar relatórios personalizados com métricas financeiras e operacionais **para** justificar decisões em reuniões estratégicas.
Como gestor administrativo, **quero** acessar um painel que exibe o status de todos os contratos ativos, **para** garantir o cumprimento das obrigações contratuais.
Como analista de TI, **quero** que o sistema tenha APIs documentadas e fáceis de usar, **para** integrar o novo sistema às plataformas existentes da Justiça Federal.
Como analista de TI, **quero** configurar perfis de acesso baseados nas funções dos usuários, **para** garantir a segurança e a privacidade dos dados.

Como usuário final, **quero** exportar relatórios em diferentes formatos (PDF, Excel) **para** apresentações e compartilhamento com outros departamentos.

8. Riscos do Projeto

Riscos Identificados	Ações de Controle
Os modelos de inteligência artificial podem não atender inicialmente aos requisitos de precisão para previsões financeiras, operacionais e de riscos.	Realizar treinamentos e ajustes contínuos nos modelos de IA com dados confiáveis e representativos.
Dados sensíveis podem estar sujeitos a ataques cibernéticos ou uso indevido, especialmente em endpoints de APIs.	Implementar autenticação robusta, criptografia de dados e monitoramento contínuo de segurança.
O sistema pode enfrentar dificuldades em lidar com grandes volumes de dados ou muitos usuários simultâneos.	Projetar uma arquitetura escalável, realizar testes de carga e otimizar consultas ao banco de dados.
Os stakeholders podem não participar ativamente do levantamento de requisitos, validação ou treinamentos.	Promover reuniões regulares, comunicar benefícios do sistema e estabelecer compromissos claros com as partes interessadas.
Usuários finais podem resistir à adoção do novo sistema, preferindo continuar com processos manuais.	Oferecer treinamentos detalhados, suporte inicial e envolver os usuários no processo de desenvolvimento.
A infraestrutura tecnológica atual pode não ser adequada para suportar o sistema em produção.	Realizar auditorias na infraestrutura e, se necessário, alocar orçamento para melhorias.

O prazo para a entrega do projeto pode ser insuficiente devido à complexidade técnica e validações necessárias.	Priorizar entregas incrementais (MVP) e alinhar expectativas com stakeholders.
A integração com sistemas existentes pode ser mais difícil e demorada do que o esperado.	Realizar um levantamento técnico detalhado dos sistemas legados e testar APIs em fases iniciais.
O escopo pode expandir durante o desenvolvimento, comprometendo o prazo e os recursos.	Adotar uma gestão de mudanças rigorosa e priorizar requisitos essenciais.
Resistência dos usuários finais	Oferecer treinamentos abrangentes e suporte inicial.
Falhas de integração com sistemas existentes	Realizar testes antecipados e levantamentos detalhados.
Volume elevado de dados comprometendo desempenho	Implementar arquitetura escalável e otimizar consultas.

9. Técnicas e Algoritmos de IA

9.1 - Análise Preditiva

- Objetivo: Prever eventos como férias, gastos futuros e riscos operacionais.
- Algoritmos:
    - Regressão Linear e Regressão Logística: Para prever valores contínuos (como gastos) e probabilidades de eventos (como a chance de atrasos).
    - Árvores de Decisão e Florestas Aleatórias: Para modelos mais robustos e interpretáveis em previsões financeiras.
    - Redes Neurais (e.g., Redes Neurais Recorrentes - RNNs): Para analisar séries temporais e padrões complexos nos dados.

9.2 - Classificação de Dados

- Objetivo: Classificar contratos quanto ao nível de risco ou eficiência.
- Algoritmos:
    - Support Vector Machines (SVM): Para identificar categorias em conjuntos de dados bem definidos.
    - K-Nearest Neighbors (KNN): Para segmentação baseada em similaridades entre contratos.
    - XGBoost e LightGBM: Para problemas complexos de classificação com eficiência computacional.

9.3 - Processamento de Linguagem Natural (NLP)

- Objetivo: Analisar textos contratuais e extrair insights automatizados.
- Técnicas:
    - Extração de Entidades Nomeadas (NER): Identificar partes importantes em contratos, com cláusulas específicas ou datas-chave.

- Modelos Pré-Treinados (BERT, GPT): Para compreender e resumir textos contratuais ou responder perguntas relacionadas.
- Análise de Sentimentos: Avaliar o tom de comunicações relacionadas aos contratos.

9.4 - Detecção de Anomalias

- Objetivo: Identificar dados inconsistentes ou atípicos, como valores fora do padrão em contratos.
- Algoritmos:
  - Autoencoders: Detectar anomalias em dados financeiros complexos.
  - Isolation Forests: Para identificar contratos ou transações com padrões incomuns.
  - Clusterização (e.g., K-Means): Para segmentar dados e detectar pontos fora do cluster principal.

9.5 - Recomendações e Otimização

- Objetivo: Sugerir ações para otimizar a gestão de contratos.
- Algoritmos:
  - Sistemas de Recomendação Baseados em Conteúdo: Para sugerir melhores práticas ou fornecedores com base em contratos passados.
  - Algoritmos Genéticos: Para otimizar alocações ou decisões contratuais.
  - Programação Linear: Para resolver problemas de alocação de recursos com restrições.

9.6 - Visualização e Dashboards

- Objetivo: Gerar gráficos e visualizações interativas baseadas em IA.
- Ferramentas e Técnicas:
  - Redução de Dimensionalidade (PCA, T-SNE): Para criar visualizações simplificadas e intuitivas.

- Bibliotecas de Visualização (Plotly, Dash, D3.js): Para gráficos dinâmicos e interativos.

9.7 - Aprendizado por Reforço

- Objetivo: Melhorar processos de tomada de decisão em tempo real.
- Técnicas:
  - Q-Learning: Para simular cenários e identificar ações ótimas.
  - Deep Reinforcement Learning (e.g., DQN): Para gerenciar fluxos de trabalho complexos com múltiplas variáveis.

9.8 - Aplicação no Projeto

- Análise de Contratos: Usar NLP para automatizar a leitura e interpretação de documentos contratuais.
- Previsões Financeiras: Aplicar algoritmos de regressão e redes neurais.
- Riscos Operacionais: Combinar detecção de anomalias e classificação para identificar áreas problemáticas.
- Dashboards Inteligentes: Incorporar insights gerados por IA para visualizações interativas.

10. Partes Interessadas [*Stakeholders*]

1. Caio Sotero Rosa (Diretor do Nucad)
2. Neisson Abadio Silva (Supervisor da Sevit)
3. Frankmar dos Reis (Nuasg)

"Juntos, podemos transformar a Justiça Federal da Primeira Região em um exemplo de eficiência e modernidade na Gestão de Contratos e Licitações."

gov.br Documento assinado digitalmente
FELIPE GALVAO LAGARES
Data: 23/01/2025 15:00:25-0300
Verifique em https://validar.iti.gov.br

Felipe Galvão Lagares

gov.br Documento assinado digitalmente
HAILTON DAVID LEMOS
Data: 23/01/2025 14:55:22-0300
Verifique em https://validar.iti.gov.br

Hailton David Lemos

gov.br Documento assinado digitalmente
RAPHAEL ABENOM DOS SANTOS SILVA
Data: 23/01/2025 15:39:22-0300
Verifique em https://validar.iti.gov.br

Raphael Abenom

gov.br Documento assinado digitalmente
RONALDO MARTINS DA COSTA
Data: 12/02/2025 15:42:19-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Ronaldo Martins da Costa
Coordenador Acadêmico

Caio Sotero Rosa
Diretor do Nucad