FinegocIA

Equipe FinegocIA

https://github.com/LucasFigAze/T-picos-Avan-ados-SI3-LucasRafaelVitorLuisArthurFelipe.git

Rafael Mourato(rmdv)
Lucas Figueiredo(lfa5)
Arthur Santos(asms)
Luís Felipe(lfgsm)
Felipe Santos(fss9)

Recife, 10/12/2025

Prof. Vinicius Cardoso Garcia (vcg@cin.ufpe.br)

Sumário

1. Introdução

Este relatório consolida o desenvolvimento do FinegocIA, um agente inteligente de negociação projetado para realizar negociações financeiras em nome de empresas com devedores, entendendo regras de negociação e oferecendo comunicação limpa e viável.

Contextualização do problema de negócio:
O Brasil tem um número recorde de empresas e pessoas inadimplentes, com cerca de 8,1 milhões de CNPJs negativados e mais de 78,8 milhões de pessoas endividadas em 2025, segundo dados da Serasa e InfoMoney.
Hoje as empresas de mercado tem equipes focadas unicamente no processo de cobrança e negociação de dívidas, o que é custoso, exige tempo e muitas vezes não tem uma disponibilidade alta pelo fato da equipe depender das pessoas e suas avaliações funcionando no horário comercial. Pensando nisso, o FinegocIA veio como uma solução para automatizar esse processo, 24h por dia, todo dia, respeitando as regras com uma comunicação efetiva e buscando o benefício mútuo.

Objetivos da solução:
Geral: O sistema visa otimizar o processo de negociação utilizando IA Generativa (GenAI), permitindo conversas empáticas, personalizadas e automatizadas, que conduzem o cliente a acordos viáveis e sustentáveis, reduzindo custos operacionais e aumentando as taxas de recuperação.

Específicos:
- Automatizar o processo de negociação de dívida.
- Conseguir seguir regras de negócio definidas em contrato.
- Fornecer uma comunicação que seja de acordo com o estilo e formato da empresa.

2. Metodologia

O projeto seguiu a metodologia AIDesign, estruturada em quatro fases: Imersão, Ideação, Produção e Validação, alinhado também com a metodologia do sinfonia, para produção dos documentos e guiar as decisões de negócio da solução.
Rodamos sprints semanais, fazendo o desenvolvimento do que tinhamos definido como nosso backlog, testando a funcionalidade no discord e ajustando a medida que os desafios apareciam.

Gestão do Projeto: A equipe utilizou metodologias ágeis, com divisão de tarefas baseada em issues no GitHub e reuniões semanais para alinhamento, além de comunicação via mensagens para alinhamento constante sobre evoluções na plataforma. Devido a boa comunicação que conseguimos desenvolver pelas reuniões e mensagens constantes, não foi necessário recorrer a uma ferramenta para gerenciar as sprints.

Ferramentas: Github (controle de versão), VSCODE(desenvolver o código), Discord (comunicação), Figma (prototipagem), Python/Streamlit (desenvolvimento), whatsapp e google meet (para comunicação interna).

3. Documentação da Execução do Projeto

Todos os canvas propostos pela metodologia sinfonia foram desenvolvidos e lançados no github, para evitar redundância manteremos apenas a execução por etapa do AI Design, mas os documentos são possíveis acessar no github mencionado no início do documento.

3.1 Imersão

- Persona 1: O cidadão/ empresa que precisa fazer uma negociação de dívida com uma empresa, banco ou instituição financeira e que se submete a tratar com pessoas ou mensagens automáticas.
- Persona 2: Instituição que disponibiliza o crédito e precisa investir em uma equipe para negociação, além de grandes regras para como tratar essas negociações.
- Fontes de Dados:
    - ERP: Fonte principal que será usada quando estivermos com o cliente real. Contém dados cadastrais, históricos financeiros, contratos ativos e informações essenciais para análise de risco de crédito.
    - Discord: Canal de comunicação incial utilizado para registrar as conversas entre o bot e o cliente do parceiro e realizar as negociações.
    - Banco da solução: Base central da solução, armazenando histórico de análises, scores, logs de execução, parâmetros de modelos, métricas, usuários, permissões, auditoria e registros operacionais do sistema.

3.2 Ideação

- Design de Prompts: Criamos o documento de design de prompts para o agente FinegocIA. Utilizamos técnicas de Chain-of-Thought para que o agente "pense" na estratégia de negociação antes de responder ao usuário.
- Exemplo de Prompt:"""
  Você é o Fin.negocia, um agente de negociação oficial da empresa {nome_empresa}.
  SUA PERSONA (System Prompt):
  {prompt_base_persona}
  --- DADOS DO CLIENTE —
  Cliente: {nome_cliente}
  Valor Total da Dívida: R$ {valor_divida}
  Faturas em Aberto:
  {detalhes_faturas}
  --- REGRAS DE NEGOCIAÇÃO (Siga estritamente) —
  1. Desconto MÁXIMO à vista: {max_desconto}%
  2. Parcelamento MÁXIMO: {max_parcelas}x
  3. Juros de parcelamento: {juros}% ao mês.
  4. Se o cliente pedir algo fora dessas regras, recuse educadamente e faça uma contraproposta dentro das regras.
  5. Seja objetivo, empático e profissional.
  --- REGRA DE OURO (COMPLIANCE) —
  NUNCA finalize o acordo imediatamente após o cliente dizer "aceito".
  Se o cliente aceitar uma proposta, você DEVE OBRIGATORIAMENTE:
  1. Resumir explicitamente os termos (Valor total, Número de parcelas, Valor da parcela).
  2. Perguntar: "Você confirma estes termos para a geração do boleto?"
  Somente após essa confirmação explícita o processo será encerrado.
  --- TABELA DE CÁLCULOS PRÉ-APROVADA (USE ESTES VALORES) —
  Abaixo estão os valores EXATOS para parcelamento. NÃO faça cálculos matemáticos, apenas consulte esta lista se o cliente perguntar sobre parcelas:
  {tabela_calculada}
  --- HISTÓRICO RECENTE DA CONVERSA —
  {historico}
  MENSAGEM ATUAL DO CLIENTE:
  {input}
  SUA RESPOSTA:
  """
- Prototipagem: Como nossa solução é primariamente focada na integração de outras soluções não foi necessário um protótipo.

3.3 Produção

- Arquitetura do Sistema (C4 Model): Está presente no github.
- Contexto: Usuário interage com o Sistema FinegocIA, que usa a API da OpenAI/Gemini.
- Container: Aplicação Web (Streamlit) + Backend (Python) + Banco Vetorial (para contexto, se aplicável).
- Tecnologias: Python, Django, LangChain, Streamlit, API OpenAI, Docker, SQL.
- Integração: O fluxo conecta o input do usuário a um orquestrador que injeta o system prompt da persona escolhida antes de chamar a LLM para então dirigir a um canal de comunicação e realizar uma conversa com o usuário final.

3.4 Validação

- Diversificação Funcional: O sistema foi testado não apenas para negociação de preços, mas também para resolução de conflitos.
- Os testes foram rodados pela nossa equipe de forma automatizada e manual, porém seria necessário um cliente real para testes mais robustos
- Feedback e Iteração: Testes iniciais mostraram que a IA cedia muito fácil. Ajustamos a "temperatura" e o prompt de resistência para tornar a negociação mais desafiadora.

4. Discussões Técnicas e Estratégicas

- Decisões Arquiteturais:
    - Optamos por uma arquitetura stateless no backend para facilitar a escalabilidade, mantendo o histórico da conversa na sessão do cliente (browser/Streamlit session state).
    - As linguagens utilizadas, como o Django, foram consideradas não só a facilidade na construção da solução como familiaridade da nossa equipe.
    - Não desenvolvemos um frontend por não sentir necessidade disso inicialmente, o django oferece um layout amigável para o admin, e as conversas são feitas em uma plataforma terceira, logo não foi necessário o desenvolvimento
    - Usamos o Gemini pela facilidade de integração e pela disponibilidade de ser feito sem custo.
    - Colocamos um protocolo também para identificar se a pessoa ainda estava digitando para evitar que a IA respondesse a mensagesns como “oi”.

- Desafios:
    - O principal desafio foi a alucinação da IA em relação a valores numéricos. A solução foi instruir a IA no prompt a ser vaga com números a menos que explicitamente definidos no cenário.
    - Outro desafio foi fazer a integração com as plataformas, tivemos que aprender a fazer do zero e construir algo para nossa solução.
    - Além disso, tivemos algumas dificuldades nas definições de como seriam passadas as regras de negócio, pensando em uma solução incial passamos como algo definido, porém o intuíto é evoluir para o contrato do cliente virar um input para a IA.

5. Considerações Éticas

- Vieses: A IA pode oferecer condições diferentes de negociação dependendo de atributos correlacionados com raça, gênero, localidade, idade, ou mesmo uso de linguagem, o que reforça discriminação.
- Privacidade: Exposição de dados financeiros sensíveis, uso além do consentido, ou re-identificação a partir de dados anonimizados.
- Robustez: Manipulação do modelo, via promp injection, exploração de lógica de negociação, ou abuso para extrair dados de outros clientes / manipular termos.
- Mitigação: Implementamos "Safety Guidelines" no system prompt para garantir que a IA mantenha o respeito e não se submeta a táticas de manipulação, focando em negociação baseada em acordos. Além disso, seguiremos a lgpd anonimizando dados sensíveis.

6. Lições Aprendidas e Reflexões Finais

Reflexão da Equipe:

- A metodologia AIDesign nos forçou a não pular direto para o código, o que economizou tempo ao definir bem as personas. O uso de IA Generativa provou ser poderoso, mas requer uma engenharia de prompt robusta para ser consistente.

Relatos Individuais:

- Lucas Figueiredo:
  Focar na documentação da arquitetura utilizando o modelo C4 foi essencial para visualizar as dependências do projeto. Além disso, a criação do documento de design de prompts para o agente FinegocIA me permitiu entender a importância de estruturar bem as instruções para obter

respostas coerentes e seguras, alinhando com meu interesse em threat modeling e segurança de aplicações.

- Rafael:
  A documentação pelo método sinfonia é muito rica e precisa da devida atenção, não correr e fazer de qualquer forma, mas sim dedicar o tempo necessário para garantir que seja bem feito irá salvar tempo no momento de discutir pontos estratégicos e garantir que faça sentido o que foi desenvolvido.
- Luís:
  A principal lição foi lidar com a complexidade arquitetural, especialmente ao equilibrar fluxos assíncronos do Discord com as demandas síncronas do serviço principal, o que gerou desafios de coordenação e definição de escopo. No fim, ficou claro que priorizar o essencial para o MVP e aceitar limitações é tão importante quanto a qualidade técnica — e que saber o que não fazer também faz parte de entregar um produto viável
- Felipe:
  Acredito que o maior desafio do projeto foi a definição estratégica e arquitetural deste projeto, acima de tudo, um exercício de priorização. A necessidade de delimitar o escopo para um MVP me obrigou a tomar decisões difíceis sobre o que era essencial e o que era apenas desejável, aceitando que nem tudo que foi planejado caberia no prazo e que o feito é melhor do que o perfeito. Esse processo também me fez perceber (novamente) a diferença real entre construir um sistema e desenvolver um produto, que envolve muitas outras coisas. O maior aprendizado que fica é que a estratégia de entrega é tão importante quanto a qualidade técnica, e que saber o que ou como não fazer é necessário para a viabilidade do projeto.

- Arthur:
  O meu principal aprendizado foi sobre as dificuldades de uso da Inteligência Artificial em casos extremos e a importância dos testes. Testando o código, vi que respostas inusitadas podem fazer o Gemini alucinar. Por isso, a revisão e melhoria contínua do prompt se faz tão necessária. Além disso, nosso código foi revisado pela IA, usando a metodologia de “LLM as a judge”, que eu nunca havia utilizado antes, e que facilita a testagem em massa, outro aprendizado obtido no desenvolvimento do projeto.

7. Referências

- Vídeos no youtube sobre como trabalhar a API do discord e Gemini.
- Gemini para dúvidas pontuais.
- Documentação do Sinfonia.