

Synapse – Da reunião ao backlog sem ruídos

Aplicação web para automatizar a documentação de requisitos de projetos.

Autores:

Bruno Sousa Batista

Vanessa Andrade de Sousa

Ian José Soares

Nicollas Ney Alcântara

Curso: Ciência da Computação

Instituição: Universidade Estadual do Ceará

Data: 14/11/2025

Sumário

1. Resumo Executivo
2. Contexto e Diagnóstico do Problema
 3. Problema Central
 4. Objetivos do Projeto
 5. Métricas de Sucesso
 6. Público-alvo
7. Requisitos do Sistema
8. Solução Proposta – Visão de Produto
 9. Arquitetura e Design Técnico
10. Desenvolvimento e Metodologia
 11. Validação, Testes e Feedback
 12. Resultados Atuais
 13. Limitações do MVP
 14. Lições Aprendidas
 15. Roadmap de Evolução
 16. Conclusão
 17. Referências

1. Resumo Executivo

O **Synapse** é uma aplicação web desenvolvida com **Django** que automatiza a documentação de requisitos a partir de gravações de reuniões. A plataforma realiza:

- transcrição automática via **Whisper** (local ou via API),
- interpretação da reunião com **Gemini/OpenAI**,
- geração de backlog, requisitos funcionais, não funcionais e histórias de usuário,
- integração opcional com **Jira**,
- exportação de documentação em **PDF**,
- e um ciclo contínuo de refino, permitindo ajustes a qualquer momento.

Agora conta com:

- sistema completo de autenticação (login/cadastro),
- histórico de análises por usuário,
- opções de integração direta com ferramentas externas,
- Geração incremental de documentação a cada mensagem enviada ao LLM,
- Interface revisada para reduzir atrito no fluxo do usuário.

A solução reduz retrabalho, melhora a consistência das entregas e aumenta a rastreabilidade entre reunião, backlog e documentação final.

2. Contexto e Diagnóstico do Problema

Equipes de desenvolvimento frequentemente enfrentam problemas como:

- perda de informações entre reuniões,
- documentação feita às pressas e com baixa padronização,
- divergências entre o que foi acordado e o que foi registrado,
- retrabalho causado por interpretações diferentes entre áreas,
- dificuldade em conectar reuniões, backlog e documentação formal.

A dependência do registro manual amplia riscos de comunicação e gera atrasos.

3. Problema Central

O registro manual, despadronizado e sujeito a ruídos compromete a documentação de requisitos, gerando:

- inconsistências no backlog,
- desalinhamento entre times,
- dificuldades de rastreabilidade,
- perda de produtividade.

4. Objetivos do Projeto

Objetivo Geral

Automatizar e padronizar a documentação de requisitos a partir de gravações de reuniões.

Objetivos Específicos

- Implementar captura de áudio, upload e transcrição automática (Whisper).
- Gerar requisitos, histórias de usuário e backlog via Gemini/OpenAI.
- Permitir **edição incremental** do resultado conforme feedback do usuário.
- Integrar com **Jira** para criação automática de issues.
- Oferecer geração de documentação em **PDF** a qualquer momento.
- Implementar **autenticação** e histórico por usuário.
- Minimizar retrabalho pós-reunião.

5. Métricas de Sucesso

- Aderência entre documentação gerada e conteúdo da reunião.
- Redução do retrabalho pós-reunião.
- Tempo médio de geração de documentação.
- Taxa de aceitação dos requisitos pelos stakeholders.
- Uso efetivo das integrações (ex.: percentual de issues criadas via Synapse).

6. Público-alvo

- Product Owners
- Desenvolvedores
- Analistas de Requisitos
- Equipes que buscam rastreabilidade e padronização em documentação

7. Requisitos do Sistema

Requisitos Funcionais

- Login e cadastro de usuários.
- Registro de sessões e histórico por usuário.
- Upload ou gravação de áudio.
- Transcrição automática via Whisper.
- Geração automática de requisitos, backlog e histórias de usuário.
- Refinamento incremental a partir da interação com o LLM.
- Integração com **Jira** (criação automática de issues).
- Exportação de documentação em PDF (ReportLab).
- Interface web com feedback em tempo real.

Requisitos Não Funcionais

- Transcrição responsiva para reuniões de até ~5 minutos (Whisper local).
- Interface responsiva e intuitiva (HTML5, CSS3, JS).

- Segurança via autenticação JWT.
- Compatibilidade com múltiplos navegadores.
- Banco de dados simples e portátil (SQLite no MVP).
- Tempo de resposta aceitável para geração de backlog.
- Possibilidade de migração futura para serviços em nuvem.

8. Solução Proposta – Visão de Produto

O Synapse cria um fluxo fluido e automatizado:

1. **Usuário autentica-se.**
2. Envia ou grava o áudio da reunião.
3. O sistema transcreve o conteúdo (Whisper).
4. O LLM (Gemini/OpenAI) interpreta, extrai e organiza requisitos.
5. O usuário pode solicitar ajustes, refinar histórias ou reorganizar o backlog.
6. Se desejar, envia itens diretamente ao **Jira**.
7. A qualquer momento, gera o **PDF completo** da documentação atualizada.

Diferenciais

- Ciclo de refino contínuo.
- Integração com ferramentas de gestão (ex.: Jira).
- Histórico por usuário, mantendo a evolução das reuniões.
- Redução expressiva de ruídos e retrabalho.

9. Arquitetura e Design Técnico

Stack Tecnológico

- **Backend:** Python, FastAPI
- **Frontend:** HTML5, CSS3, JavaScript, TailWindCSS
- **IA:**
 - Whisper local para transcrição
 - Gemini/OpenAI para processamento semântico
- **Autenticação:** JWT
- **Banco de Dados:** SQLite (MVP)
- **Geração de PDF:** ReportLab
- **Integração Externa:** API Jira

Fluxo de Dados

1. Usuário autentica-se.
2. Envia áudio → processado pelo Whisper.
3. Transcrição enviada ao LLM → requisitos/backlog.
4. Usuário interage com o modelo → refinamentos incrementais.
5. Integração opcional com Jira → criação de issues.
6. Geração de PDF consolidado a qualquer momento.

10. Desenvolvimento e Metodologia

Metodologia ágil (Scrum) com sprints semanais.

Backlog monitorado no Trello.

Papéis

- **Ian José** – RAG & Back-end Development
- **Vanessa Andrade** – Front-end & Presentation Design
- **Bruno Sousa** – Back-end and Front-end Developer & Gemini Integration
- **Nicollas Ney** – Technical Documentation, Content Architecture, test & QA

11. Validação, Testes e Feedback

Testes com Django Test Framework (endpoints, geração, exportação).

Testes de fluxo completo: upload → transcrição → geração → PDF.

Validação com usuários reais para ajustar:

- responsividade,
- clareza das mensagens do sistema,
- estabilidade da transcrição,
- integração com Jira.

12. Resultados Atuais

O Synapse hoje oferece:

- Login e cadastro funcional.
- Upload/gravação de áudio.
- Transcrição automática.
- Geração de requisitos, backlog e histórias de usuário.
- Refinamento incremental pelo LLM.
- Integração com Jira para criação de issues.
- Exportação de PDF com documentação atualizada.
- Interface organizada e mais fluida.

13. Limitações do MVP

- Whisper local depende do hardware disponível → o tempo de processamento varia.
- O Modelo LLM depende da estabilidade da API.
- Geração de PDF ainda limitada em estilos avançados.
- Falta um sistema de versionamento de documentos (ainda não implementado).
- Integração com Trello ainda básica.

14. Lições Aprendidas

- Usuários valorizam mais **clareza no fluxo** do que automação agressiva.
- A integração com Jira exige cuidados de autenticação e rate limits.
- PDF precisa ser gerado sob demanda, não automaticamente, para evitar travamentos.
- O refinamento incremental melhora muito a qualidade final da documentação.

15. Roadmap de Evolução

Curto prazo

- Melhorar interface de edição dos requisitos.
- Criar histórico de versões da documentação.
- Melhorar a pré-visualização antes do PDF.

Médio prazo

- Migração para banco de dados PostgreSQL.
- Transcrição via API (para ambientes de baixa capacidade).
- Integração com plataformas adicionais (Azure DevOps, Notion).

Longo prazo

- Análises avançadas de reuniões com NLP (detecção de conflitos, riscos).
- Dashboard com métricas automáticas de qualidade de requisitos.
- Geração automática de diagramas UML.

16. Conclusão

O Synapse aborda de forma prática um desafio histórico no desenvolvimento de software: **transformar conversas em documentação estruturada**, reduzindo ruído, aumentando a rastreabilidade e acelerando entregas.

Com as novas funcionalidades — login, integração com Jira, PDF sob demanda e ciclo incremental — o sistema deixa de ser apenas um gerador automático e aproxima-se de uma plataforma completa para gestão de requisitos assistida por IA.

17. Referências

Frameworks e Tecnologias Utilizadas

- Tiangolo, Sebastián. *FastAPI Documentation*. Disponível em: <https://fastapi.tiangolo.com>
- Python Software Foundation. *Python 3 Documentation*. Disponível em: <https://docs.python.org/3>
- SQLite Consortium. *SQLite Documentation*. Disponível em: <https://www.sqlite.org/docs.html>
- ReportLab. *ReportLab User Guide*. Disponível em: <https://www.reportlab.com/documentation/>
- React Team. *React Documentation*. Disponível em: <https://react.dev>
- Tailwind Labs. *TailwindCSS Documentation*. Disponível em: <https://tailwindcss.com/docs>

APIs e Ferramentas de IA

- OpenAI. *Whisper Speech-to-Text Model*. Disponível em: <https://openai.com/research/whisper>
- OpenAI. *Chat Completions API*. Disponível em: <https://platform.openai.com/docs>

- Google DeepMind. *Gemini API Documentation*. Disponível em: <https://ai.google.dev>
- Atlassian. *Jira REST API Reference*. Disponível em: <https://developer.atlassian.com/cloud/jira/platform/rest>

Conceitos, Metodologias e Boas Práticas

- Sommerville, Ian. *Software Engineering*, 10ª edição. Pearson.
- Pressman, Roger. *Engenharia de Software: Uma Abordagem Profissional*, 8ª edição. McGraw-Hill.
- Schwaber, Ken; Sutherland, Jeff. *The Scrum Guide*. Scrum.org.
- Wiegers, Karl; Beatty, Joy. *Software Requirements*, 3rd Edition. Microsoft Press.

Artigos e Materiais Complementares

- IEEE. *IEEE 830-1998 — Recommended Practice for Software Requirements Specifications*.
- Cohn, Mike. *User Stories Applied: For Agile Software Development*. Addison-Wesley.
- Nielsen Norman Group. *UX Documentation and Requirements Best Practices*. Disponível em: <https://www.nngroup.com>