Resumo

Este projeto consiste em um bot com personalidade inspirada no Rick Sanchez para interação via Telegram, usando um sistema de inteligência baseado em LLM. Ele inclui uma arquitetura inicial de banco de dados SQLite para armazenamento de dados, com planos de evolução para sistemas de memória e buscas semânticas mais avançados.

Estrutura de Pastas

```
Copy
├── agents/
│   └── llm_agent.py           # Agente LLM com personalidade do Rick
├── database/
│   └── db_init.py             # Inicialização do banco de dados SQLite
├── handlers/
│   └── telegram_llm_handler.py# Handler para integrar com Telegram
├── main.py                    # Arquivo principal do bot
├── requirements.txt           # Dependências do projeto
└── .env                       # Variáveis de ambiente
```

O que já foi desenvolvido

- **Agente LLM básico** com personalidade do Rick Sanchez.
- **Integração com Telegram** funcionando.
- **Estrutura inicial do banco de dados SQLite** para armazenamento de informações.

Próximos Passos

1. **Sistema de Memória Dual**: Implementar memória de curto e longo prazo.
2. **Integração com ChromaDB**: Para vetorização e busca semântica de mensagens ou conteúdos relacionados.
3. **Banco de Dados SQLite**:

    - Gestão de mensagens: criar métodos para gravar e recuperar conversas, configurando tags e contexto.
    - Armazenamento permanente dos dados.
4. **Sistema de Tags e Categorização**: Atribuir rótulos às conversas/mensagens para melhorar a navegação e o contexto.
5. **Sistema de Contexto e Memória**: Aprofundar a capacidade do bot de entender o que já foi dito e manter o "raciocínio" ao longo das interações.
6. **Redis para Cache (futuro)**: Possível uso para acelerar operações de leitura e escrita.

Sinta-se à vontade para adaptar o texto conforme o estilo que preferir e incluir instruções extras (como forma de contribuição, setup do ambiente etc.). Isso aqui é só um pontapé inicial.

Chatbot para Engenharia Civil - Documentação Consolidada

Visão Geral

Este projeto consiste em um **chatbot** no Telegram desenvolvido para auxiliar **engenheiros civis** em tarefas diárias, integrando funcionalidades de gestão de obras, controle financeiro e organização pessoal. O bot tem personalidade inspirada no **Rick Sanchez (C-137)** e utiliza inteligência artificial para prover uma experiência interativa e eficiente.

Objetivos

- **Automatizar tarefas rotineiras** de engenheiros civis
- **Facilitar o registro e acompanhamento de obras**
- **Proporcionar controle financeiro eficiente**
- **Centralizar informações e documentações** importantes
- **Oferecer uma interface amigável** e acessível via Telegram

---

Funcionalidades Principais

1. **Diário de Obra (RDO)**
    - Registro diário de atividades
    - Documentação do clima
    - Controle de efetivo
    - Registro fotográfico
    - Geração de relatórios
2. **Controle Financeiro**
    - Registro de receitas e despesas
    - Categorização de gastos
    - Acompanhamento de orçamentos
    - Relatórios financeiros
    - Alertas de custos
3. **Cronogramas**

    - Acompanhamento de prazos
    - Registro de marcos importantes
    - Alertas de deadlines
    - Visualização de progresso
4. **Gestão de Tarefas**

    - Lista de pendências
    - Priorização de atividades
    - Lembretes automáticos
    - Organização por projetos

Arquitetura Técnica

Componentes Principais

1. **Bot Telegram**

    - Interface principal com usuário
    - Processamento de comandos
    - Gestão de interações
2. **Agente LLM (C-137)**

    - Personalidade baseada em Rick Sanchez
    - Processamento de linguagem natural
    - Assistência contextual
    - Capacidade de manter contexto em curto prazo (memória de até 2 horas)
    - Futuras integrações para memória de longo prazo
3. **Sistema de Memória**

    - **Memória de curto e longo prazo** em desenvolvimento
    - **ChromaDB** para vetorização e busca semântica
    - **SQLite** para armazenamento permanente
    - Possibilidade de uso de **Redis** para cache (futuro)

Estrutura do Projeto (Código)

Tecnologias Utilizadas

- **Python** como linguagem principal
- **API do Telegram** para interação com usuários
- **OpenAI API** para processamento de linguagem natural
- **SQLite** para armazenamento permanente de dados
- **ChromaDB** para vetorização e busca semântica
- **Google Sheets** para relatórios (possível integração)
- **Notion** para documentação (possível integração)

Aspectos Técnicos

Base de Dados

1. **SQLite**
    - Armazenamento de mensagens e dados estruturados
    - Registros permanentes
2. **ChromaDB**
    - Vetorização de mensagens
    - Busca semântica e contextualização

Integrações

- **Telegram Bot API**
- **OpenAI API**
- **Google Sheets API** (futuro)
- **Notion API** (futuro)

Desenvolvimento

O que já foi desenvolvido

- **Agente LLM básico** com personalidade do Rick Sanchez
- **Integração com Telegram** funcionando
- **Estrutura inicial do banco de dados SQLite**
- **Processos de desenvolvimento e testes de integração**

Fase Atual

- Validação das funcionalidades básicas
- Definição de requisitos para implantação de memória e vetorização

Próximos Passos

1. **Implementar métodos para gestão de mensagens no SQLite**
2. **Integrar ChromaDB para busca semântica**
3. **Desenvolver sistema de tags e categorização**
4. **Implementar sistema de contexto e memória** (dual: curto e longo prazo)
5. **Testes de usabilidade** (coletar feedback)
6. **Implantação em produção**

Implantação

- **Desenvolvimento inicial local**
- Migração futura para **Google Cloud**
- **Atualizações incrementais** conforme feedback e testes
- Monitoramento de desempenho e ajustes contínuos

Considerações Futuras

- **Escalabilidade** do sistema
- Adição de **novas funcionalidades** (ex.: dashboards avançados)
- **Otimização de performance**
- **Melhorias baseadas em feedback** de usuários
- **Integração com Redis** para cache e aumento de velocidade

---

Segurança e Privacidade

- Proteção de **dados sensíveis**
- Backups regulares do banco de dados
- Controle de acesso a funções administrativas
- Conformidade com **normas de privacidade** e boas práticas

Estrutura real

```
📁 projeto/
├── 📁 agents/
│   └── llm_agent.py            # Agente de IA com personalidade Rick Sanchez
│
├── 📁 config/
│   └── turing-lyceum-*.json    # Arquivo de credenciais do Google Sheets
│
├── 📁 database/
│   └── db_init.py              # Inicialização e estrutura do banco de dados
│
├── 📁 handlers/
│   ├── handlers.py             # Handlers gerais do bot
│   ├── financeiro_handler.py  # Handler para comandos financeiros
│   ├── handler_rdo.py          # Handler para Registro Diário de Obra
│   └── telegram_llm_handler.py # Handler principal para interação com LLM
│
├── 📁 testes/
│   ├── teste_database.py       # Testes do banco de dados SQLite
│   └── test_integrations.py    # Testes de integração com APIs externas
│
├── .env                        # Variáveis de ambiente e tokens
├── .gitignore                 # Arquivos ignorados pelo git
├── main.py                    # Arquivo principal do bot
├── mainn.py                   # (Possível duplicata a ser removida)
└── requirements.txt           # Dependências do projeto
```