

































































ESTRUTURA DE PASTAS - GPT MESTRE AUTÔNOMO

Estrutura Completa do Projeto






gpt-mestre-autonomo/

	└─  README.md	# Documentação principal
	└─  RESUMO_IMPLEMENTACAO.md	# Resumo do que foi implementado
	└─  ESTRUTURA_PASTAS.md	# Este arquivo
	└─  requirements.txt	# Dependências Python
	└─  .env	# Configurações (suas chaves de API)
	└─  .env.example	# Exemplo de configuração
	└─  config.py	# Configurações centralizadas
	└─  app.py	# Interface Streamlit principal
	└─  run.py	# Script de execução
	└─  test_basic.py	# Testes básicos do sistema
	└─  Makefile	# Comandos facilitados
	└─  agents/	# 🤖 Agentes do Sistema
	└─  __init__.py	# Inicialização do módulo
	└─  base_agent.py	# Classe base para todos os agentes
	└─  carlos.py	# Agente Carlos (Interface Principal)
	└─  reflexor.py	# [Fase 2] Agente Reflexor (Auditor)
	└─  oraculo.py	# [Fase 3] Agente Oráculo (Decisor)
	└─  deep_agent.py	# [Fase 3] Agente de Análise Profunda
	└─  auto_master.py	# [Fase 3] Agente de Automação
	└─  utils/	# 🛠 Utilitários do Sistema
	└─  __init__.py	# Inicialização do módulo
	└─  logger.py	# Sistema de logging avançado
	└─  helpers.py	# [Futuro] Funções auxiliares
	└─  validators.py	# [Futuro] Validadores
	└─  memory/	# 🧠 Sistema de Memória
	└─  __init__.py	# Inicialização do módulo
	└─  vector_store.py	# [Fase 2] ChromaDB e memória vetorial
	└─  memory_manager.py	# [Fase 2] Gerenciador de memória
	└─  chroma_db/	# [Fase 2] Base de dados vetorial
	└─  backups/	# [Futuro] Backups da memória
	└─  integrations/	# 🔗 Integrações Externas
	└─  __init__.py	# Inicialização do módulo
	└─  telegram_bot.py	# [Fase 4] Bot do Telegram
	└─  notion_client.py	# [Fase 4] Cliente Notion
	└─  google_sheets.py	# [Fase 4] Google Sheets
	└─  shopee_api.py	# [Fase 5] API Shopee
	└─  magalu_api.py	# [Fase 5] API Magazine Luiza
	└─  logs/	# 📄 Sistema de Logs
	└─  gpt_mestre.log	# Log geral do sistema
	└─  errors.log	# Log específico de erros
	└─  agents.log	# Log específico dos agentes
	└─  archived/	# [Futuro] Logs arquivados
	└─  backend/	# ⚙ Backend API (Futuro)
	└─  __init__.py	# Inicialização do módulo




└─  main.py	# [Fase 3] FastAPI principal
└─  routes.py	# [Fase 3] Rotas da API
└─  scheduler.py	# [Fase 3] Agendador de tarefas
└─  webhooks.py	# [Fase 4] Webhooks externos
└─  frontend/	# 🎨 Frontend Alternativo (Futuro)
└─  index.html	# [Fase 5] Interface React/Next.js
└─  components/	# [Fase 5] Componentes React
└─  styles/	# [Fase 5] Estilos CSS
└─  tests/	# 🧪 Testes do Sistema
└─  __init__.py	# Inicialização do módulo
└─  test_agents.py	# [Futuro] Testes dos agentes
└─  test_memory.py	# [Futuro] Testes da memória
└─  test_integrations.py	# [Futuro] Testes das integrações
└─  docs/	# 📖 Documentação Avançada
└─  architecture.md	# [Futuro] Documentação da arquitetura
└─  api_reference.md	# [Futuro] Referência da API
└─  deployment.md	# [Futuro] Guia de deploy
└─  backups/	# 💾 Backups do Sistema
└─  YYYYMMDD_HHMMSS/	# Backups por data/hora
└─  restore.py	# [Futuro] Script de restauração

Status de Implementação por Pasta





✅ IMPLEMENTADO (Fase 1)

-  **Raiz** - Todos os arquivos principais
-  **agents/** - `base_agent.py` e `carlos.py`
-  **utils/** - `logger.py` completo
-  **logs/** - Sistema funcionando
-  **integrations/** - `telegram_bot.py` (preview)

🔄 EM DESENVOLVIMENTO (Fase 2-3)

-  **memory/** - ChromaDB será implementado
-  **backend/** - FastAPI e scheduler
-  **agents/** - Reflexor, Oráculo, etc.

🕒 PLANEJADO (Fase 4-5)

-  **integrations/** - APIs completas
-  **frontend/** - Interface React
-  **tests/** - Suite completa de testes
-  **docs/** - Documentação avançada

Arquivos que você precisa criar AGORA:

1. Criar estrutura de pastas:

```
bash

mkdir -p gpt-mestre-autonomo
cd gpt-mestre-autonomo
mkdir -p agents utils memory integrations logs backend tests docs backups
```

2. Criar arquivos `__init__.py`:

```
bash

touch agents/__init__.py
touch utils/__init__.py
touch memory/__init__.py
touch integrations/__init__.py
touch backend/__init__.py
touch tests/__init__.py
```

3. Salvar os arquivos dos artifacts:

- 📄 **Raiz:** `requirements.txt`, `config.py`, `app.py`, `run.py`, `.env`, etc.
- 📁 **agents/:** `base_agent.py`, `carlos.py`
- 📁 **utils/:** `logger.py`
- 📁 **integrations/:** `telegram_bot.py`

💡 Comandos para criar estrutura rapidamente:

Linux/Mac:

```
bash

# Criar projeto
mkdir gpt-mestre-autonomo && cd gpt-mestre-autonomo

# Criar pastas
mkdir -p {agents,utils,memory,integrations,logs,backend,tests,docs,backups}

# Criar __init__.py
touch {agents,utils,memory,integrations,backend,tests}/__init__.py

# Verificar estrutura
tree . # ou ls -la
```

Windows:

cmd

```
# Criar projeto
```

```
mkdir gpt-mestre-autonomo
```

```
cd gpt-mestre-autonomo
```

```
# Criar pastas
```

```
mkdir agents utils memory integrations logs backend tests docs backups
```

```
# Criar __init__.py
```

```
type nul > agents\__init__.py
```

```
type nul > utils\__init__.py
```

```
type nul > memory\__init__.py
```

```
type nul > integrations\__init__.py
```

```
type nul > backend\__init__.py
```

```
type nul > tests\__init__.py
```

Próximos Passos:

1. **Criar a estrutura** de pastas acima
2. **Salvar todos os arquivos** dos artifacts nas pastas corretas
3. **Executar:** `python run.py`
4. **Acessar:** `http://localhost:8501`

A estrutura está pronta para crescer até um sistema completo! 🚀