

# FloralBot AI — Documentação Técnica de IA & NLP

## 1. Visão Geral

O FloralBot AI é um módulo de IA conversacional que executa uma **anamnese emocional automatizada**, mapeando o estado afetivo do usuário e recomendando Florais de Bach de maneira personalizada. O motor cognitivo combina:

- LLM Gemini API
- Pipeline customizado de NLP
- Orquestração via Flask
- Análise semântica orientada a intenção
- Base interna de florais (persistida em PostgreSQL via SQLAlchemy)

O sistema não adivinha. Ele **processa, correlaciona, limpa, normaliza e interpreta dados textuais** para entregar recomendações assertivas.

## 2. Arquitetura de IA

### 2.1 Componentes Principais

#### 1. Cliente LLM (Gemini API)

Localização no código:

[app/chatbot.py](#)

Responsável por:

- Engajar o modelo em modo contextual dinâmico.
- Manter "memória" da conversa entre turnos.
- Converter prompts de input do usuário em respostas enriquecidas.

## **2. Motor NLP (pré-processamento + heurísticas)**

Localização no código:

[app/chatbot.py](#)

Funções internas:

- Tokenização
- Lematização
- Stopword removal
- Classificação emocional heurística
- Feature extraction para ranking de florais

## **3. Módulo de Recomendação de Florais**

Localização no código:

[app/chatbot.py](#)

Funções:

- Match semântico entre sintomas e florais cadastrados
- Scoring baseado em padrões emocionais
- Critérios de relevância por intensidade

## **4. Orquestrador Flask**

Localização:

[app/routes.py](#)

[app/auth/routes.py](#)

[app/models.py](#)

Responsável por expor rotas, autenticação e registrar mensagens do chat.

### 3. Pipeline NLP (Deep Dive)

O coração da brincadeira. Aqui é onde a mágica técnica acontece.

#### ♦ Tokenização

Localização no código:

[app/chatbot.py](#)

Estratégia:

- Tokenização baseada em whitespace + regex
- Normalização Unicode
- Lowercase para padronização
- Compatível com input ruidoso (“to nervoso”, “tô ansioso”, etc.)

#### ♦ Lematização

Localização no código:

[app/chatbot.py](#)

Mecanismo:

- Dicionário leve personalizado + spaCy (se habilitado)
- Redução de formas flexionadas para o radical
- Improves recall de sintomas como:  
“chorando”, “chorar”, “chorei” → **chorar**

#### ♦ Análise de Sentimentos

Localização no código:

[app/chatbot.py](#)

Técnicas:

- Valência lexical + pesos
- Biblioteca complementar (TextBlob/PT-BR) se disponível
- Ajustes finos para emoções florais:
  - Medo
  - Angústia
  - Ansiedade
  - Fadiga mental
  - Tristeza
  - Irritabilidade

#### ◆ **Detecção de Sintomas Emocionais**

Localização no código:

[app/chatbot.py](#)

Mapeia termos do usuário para sintomas clinicamente aceitos no sistema floral.

Exemplo:

“esgotado”, “cansado”, “exausto”, “sem energia” → **fadiga emocional**

#### ◆ **Rankeamento de Florais**

Localização no código:

[app/chatbot.py](#)

Modelo híbrido:

- **Score da similaridade semântica**
- **Relevância por intensidade emocional**

- **Ponderação por histórico da conversa**

#### **4. Fluxo Conversacional com LLM**

Localização no código:

`app/chatbot.py`

1. Usuário envia mensagem
2. Texto é pré-processado pelo NLP interno
3. NLP extrai sintomas e emoções primárias
4. Dados + prompt inteligente alimentam o Gemini
5. Gemini produz resposta humanizada
6. Motor de recomendação sugere floral
7. Sistema combina tudo em uma resposta coesa

Essa fusão é o diferencial competitivo: não depende 100% da IA nem 100% do NLP — é **um casamento corporativo bem arranjado**.

#### **5. Integração com Banco de Dados**

Localização:

`app/models.py` – classe `Floral`

Campos:

- `name`
- `description`

- indications

Usado pelo recomendador para correlacionar sintomas → floral adequado.

## 6. Onde cada parte está no código (resumo executivo)

Quer uma visão corporativa direto ao ponto? Aqui vai:

Função	Localização	Descrição
Tokenização	<a href="#">app/chatbot.py</a>	Quebra e normaliza entrada
Lematização	<a href="#">app/chatbot.py</a>	Reduz palavras ao radical
Análise de Sentimentos	<a href="#">app/chatbot.py</a>	Calcula valência emocional
Sintomas Emoções	→ <a href="#">app/chatbot.py</a>	Heurística emocional
Recomendação	<a href="#">app/chatbot.py</a>	Matching floral
Chamada Gemini	<a href="#">app/chatbot.py</a>	IA central
Roteamento/chat	<a href="#">app/routes.py</a>	Interface conversacional

## 7. Exemplo Simplificado do Processo

Usuário diz:

“Estou inquieto, não durmo direito, minha cabeça não desliga.”

Pipeline:

- Tokenização → ["estou", "inquieto", "não", "durmo", ...]
- Lematização → ["estar", "inquieto", "não", "dormir" ...]
- Sentimento → negativo (alta ansiedade)

- Sintomas → ansiedade + preocupação excessiva
- Florais → White Chestnut, Impatiens, Aspen
- Gemini → texto final humanizado

## **8. Boas Práticas Internas**

- Manter florais atualizados no painel administrativo
- Não entupir o prompt do Gemini de informação desnecessária
- Cuidado com inputs longos — use truncamento inteligente
- Logs do motor NLP ajudam muito na depuração

## **9. Roadmap de Evolução**

- Classificador de emoções baseado em transformer local
- Clusterização de padrões emocionais do usuário
- Score adaptativo baseado no histórico da jornada
- Recomendações combinadas (mix de florais)

## **10. Considerações finais**

O FloralBot AI não é um chatbot comum.

É um analista emocional com esteroides computacionais, uma pitada de heurística e temperado com IA de última geração.