# Báo cáo Kỹ thuật & Tiến độ Phát triển AI Chatbot Du lịch

Phiên bản: 2.1 (Architecture: Database-First & Auto-Discovery)

Ngày cập nhật: 21/11/2025

Trạng thái: Ổn định (Stable) - Đã hoàn thiện logic tìm kiếm lõi.

## 1. Tổng quan Tiến độ (Status Report)

Hệ thống đã chuyển đổi thành công từ mô hình "NLU-Dẫn dắt" (cũ, dễ sai) sang mô hình **"Dữ liệu-Dẫn dắt" (Database-First)**.

### Đã hoàn thành (Done)

* [x] **Auto-Discovery (Tự động khám phá):** Code tự động quét tất cả collection trong MongoDB (db.listCollections()) để tìm dữ liệu, không cần hardcode tên bảng.
* [x] **Smart Stripping (Cắt từ thông minh):** Tự động loại bỏ từ cảm thán (*ngon, đẹp, mua, bán*) và cắt tên tỉnh (*Hà Nội, Huế*) để tìm ra thực thể gốc (*Phở bò, Bún bò*).
* [x] **Merged Data Handling (Xử lý Gộp):** AI nhận diện đúng nguồn gốc địa danh/món ăn dù dữ liệu bị gộp vào tỉnh khác (VD: Nhận biết *Kỳ Co* thuộc *Bình Định* dù data nằm ở *Gia Lai*).
* [x] **Priority Override:** Cơ chế tìm kiếm dữ liệu cụ thể luôn chạy trước logic Chitchat, khắc phục lỗi câu hỏi ngắn (VD: "Eo Gió") bị hiểu nhầm là chào hỏi.
* [x] **AI Thinking Mode:** Kích hoạt chế độ "Chuyên gia" khi tìm thấy vật thể cụ thể, trả lời sâu sắc thay vì liệt kê.

### Đang phát triển / Cần làm (To-Do)

* [ ] **Caching Layer:** Tích hợp Redis để lưu kết quả tìm kiếm scanItemInDB, giảm tải cho MongoDB với các từ khóa hot.
* [ ] **Feedback Loop:** Thêm cơ chế để người dùng report câu trả lời sai -> Ghi log để dev tinh chỉnh Regex.
* [ ] **Vector Search (Semantic):** Nâng cấp tìm kiếm theo ngữ nghĩa (VD: Hỏi *"Chỗ nào ngắm hoàng hôn đẹp"* -> Tự tìm ra *"Đồi cát Mũi Né"*).

## 2. Luồng hoạt động (System Logic Flow)

Hệ thống xử lý theo mô hình **Thác nước (Waterfall)** với độ ưu tiên giảm dần:

flowchart TD  
 Start([User Input: "Ăn Bún chả cá Quy Nhơn ngon k?"]) --> Step1  
   
 subgraph Step1 [1. SUPER CLEANING & SCAN (Ưu tiên 1)]  
 Direction TB  
 C1[Lọc rác: bỏ "ngon k", "ăn"] --> C2[Keyword: "Bún chả cá Quy Nhơn"]  
 C2 --> C3[Cắt Tỉnh: bỏ "Quy Nhơn"] --> C4[Keyword rút gọn: "Bún chả cá"]  
 C4 --> Scan{Auto-Scan DB}  
 end  
   
 Scan -->|Tìm thấy Item?| CheckFound{Result}  
   
 CheckFound -->|YES (Có trong DB)| Step2  
 CheckFound -->|NO (Không thấy)| Step3  
   
 subgraph Step2 [2. AI THINKING MODE]  
 Thinking[Prompt: "Bạn là chuyên gia, hãy review..."]  
 Thinking --> Response1[Trả lời chi tiết: Nguồn gốc, Vị trí, Hương vị]  
 end  
   
 subgraph Step3 [3. NLU FALLBACK & HYBRID]  
 NLU[Phân tích Intent] --> CheckIntent{Intent?}  
 CheckIntent -->|Chitchat| ReplyChat[Trả lời xã giao]  
 CheckIntent -->|Ask Weather| ReplyWeather[API Thời tiết]  
 CheckIntent -->|Generic Info| ReplyList[Trả lời list địa danh theo Tỉnh]  
 end  
   
 Response1 --> End([Hiển thị User])  
 ReplyChat --> End  
 ReplyWeather --> End  
 ReplyList --> End

## 3. Cách thức hoạt động (Technical Deep Dive)

### A. Auto-Discovery (Tự động quét)

Thay vì quy định cứng *"Tìm trong bảng A, bảng B"*, hệ thống thực hiện:

1. Lấy danh sách toàn bộ Collection hiện có: const allCols = await db.listCollections().
2. Lặp qua từng Collection để tìm kiếm Regex.  
   Lợi ích: Bạn có thể thêm bảng mới (VD: province\_hagiang) mà không cần sửa code, AI tự động tìm ra.

### B. Regex "Super Clean"

Bộ lọc từ khóa mạnh tay giúp làm sạch câu hỏi nhiễu:

* **Input:** "Cho hỏi **mua** Tương Bần **ngon** ở **đâu** vậy?"
* **Regex:** Loại bỏ mua, ngon, ở đâu, cho hỏi, vậy.
* **Output:** "Tương Bần" -> Đủ sạch để match chính xác trong DB.

### C. Context Awareness (Nhận thức ngữ cảnh gộp)

Khi tìm thấy item trong một Document gộp (VD: Document Gia Lai chứa data của Bình Định):

* Code gửi kèm danh sách merged\_from: ["Bình Định", "Gia Lai"] vào Prompt.
* AI đọc Prompt -> Suy luận: *"À, món này tên là Quy Nhơn, vậy nó thuộc Bình Định"* -> Trả lời đúng địa lý.

## 4. Bảng Test Case (Kiểm thử & Nghiệm thu)

Dưới đây là danh sách các câu hỏi để kiểm tra độ thông minh của hệ thống qua các cấp độ khó khác nhau.

### Cấp độ 1: Cơ bản (Basic)

*Kiểm tra khả năng tìm kiếm chính xác.*

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Câu hỏi** | **Mong đợi** | **Ghi chú** |
| 1 | "Mô tả Chùa Bút Tháp" | Trả lời chi tiết về Chùa Bút Tháp (Bắc Ninh). | Data Bắc Ninh |
| 2 | "Đặc sản Cao Bằng" | List ra Vịt quay, Phở chua... | Chế độ Generic |
| 3 | "Đi chơi Hà Nội" | List ra Hồ Gươm, Lăng Bác... | Chế độ Generic |
| 4 | "Review Vịt quay Cao Bằng" | Mô tả chi tiết món vịt quay. | Tìm chính xác |

### Cấp độ 2: Xử lý Nhiễu & Cắt từ (Stripping & Noise)

*Kiểm tra logic lọc từ rác và cắt tên tỉnh.*

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Câu hỏi** | **Mong đợi** | **Ghi chú** |
| 5 | "Phở bò Hà Nội ngon không?" | Tìm ra "Phở bò" trong data Hà Nội. | Cắt "Hà Nội", bỏ "ngon không" |
| 6 | "Mua Tương Bần Hưng Yên ở đâu?" | Tìm ra "Tương Bần" trong data Hưng Yên. | Cắt "Hưng Yên", bỏ "mua/ở đâu" |
| 7 | "Ăn Bún chả cá Quy Nhơn" | Tìm ra "Bún chả cá" trong data Gia Lai. | Cắt "Quy Nhơn" |
| 8 | "Chùa Dơi có gì đẹp?" | Tìm ra "Chùa Dơi" trong data Cần Thơ. | Bỏ "có gì đẹp" |

### Cấp độ 3: Data Gộp & Suy luận (Merged & Inference)

*Kiểm tra khả năng định vị nguồn gốc của AI.*

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Câu hỏi** | **Mong đợi** | **Ghi chú** |
| 9 | "Kỳ Co" | Mô tả Kỳ Co là thiên đường biển **Bình Định** (dù data ở Gia Lai). | Câu cực ngắn + Data gộp |
| 10 | "Chùa Tam Chúc" | Mô tả Chùa Tam Chúc ở **Hà Nam** (dù data ở Ninh Bình). | Data gộp |
| 11 | "Bánh khọt Vũng Tàu" | Mô tả đặc sản **Vũng Tàu** (dù data ở TP.HCM). | Data gộp |
| 12 | "Gành Đá Đĩa" | Mô tả thắng cảnh **Phú Yên** (dù data ở Đắk Lắk). | Data gộp |

### Cấp độ 4: Edge Cases (Ca khó)

*Kiểm tra khả năng chịu lỗi.*

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Câu hỏi** | **Mong đợi** | **Ghi chú** |
| 13 | "Pa pỉnh tộp" | Tìm ra món cá nướng (Tây Bắc/Điện Biên/Sơn La). | Tên độc lạ, không có từ khóa "món" |
| 14 | "Eo Gió" | Tìm ra Eo Gió (Bình Định/Gia Lai). | Tên ngắn dễ nhầm chitchat |
| 15 | "Hello bot" | Trả lời xã giao (Chitchat). | Không được tìm DB |
| 16 | "Thời tiết Hà Nội" | Trả lời thông tin thời tiết. | Intent Weather |

## 5. Kết luận

Hệ thống hiện tại đã đạt độ ổn định cao trong việc truy xuất dữ liệu tĩnh. Các lỗi ngớ ngẩn (nhầm địa danh, không tìm thấy món ăn phổ biến) đã được khắc phục 99% nhờ cơ chế **Auto-Discovery** và **Smart Stripping**.