# TỔNG KẾT PHÁT TRIỂN STEP 2 - PIPELINE V5

## I. NHỮNG GÌ ĐÃ THỰC HIỆN ĐƯỢC

1. Xây dựng file module: v5\_step\_2\_prepare\_timeline\_v2.py

2. Đã phân biệt rõ vòng đầu và các vòng sau thông qua processed\_markers\_by\_robot

3. Cắt đúng đoạn route đến marker cần xử lý, sử dụng marker theo thứ tự đã được quét sẵn từ Step 1

4. Gọi generate\_timeline\_rb\_v4\_3 để sinh timeline đúng cho từng đoạn

5. Đã test thành công cho cả các case đơn giản và phức tạp (ví dụ R4 có 3 marker không theo thứ tự)

6. Đã tạo module sup\_marker\_utils.py để gom hàm tiện ích xử lý marker

7. Tích hợp get\_next\_marker() để xác định marker tiếp theo đơn giản, chính xác

## II. TƯ DUY TỔ CHỨC DỮ LIỆU

- Step 1 cung cấp markers\_by\_robot đúng thứ tự dòng trong route

- Step 2 không cần tìm marker theo dòng nữa, chỉ dùng index để truy xuất marker kế tiếp

- Mọi hành động cắt route, sinh timeline đều phụ thuộc duy nhất vào marker đang xử lý

- Timeline được gộp dần qua từng vòng và giữ nguyên cho đến hết pipeline

## III. KẾ HOẠCH HOÀN THIỆN TIẾP THEO

1. Viết hàm cut\_route\_to\_marker(route, marker) tách riêng logic cắt đoạn

2. Viết thêm các hàm tiện ích marker khác: has\_unprocessed\_marker(), get\_marker\_arrival\_time()

3. Cảnh báo rõ ràng nếu không tìm thấy dòng chứa marker trong route (vị trí sai hoặc lỗi)

4. Ghi log (debug) mỗi đoạn timeline được sinh ra để dễ trace

5. Tối ưu việc gộp timeline nhiều vòng – tránh sort thừa

6. Chuẩn bị viết Step 3 để chọn marker kế tiếp xử lý dựa theo thời điểm đến