PHÂN CHIA CÔNG VIỆC (GIỮ REALTIME, BỎ CHAT)
+ TÍNH NĂNG 360° & AI TÌM PHÒNG

Phiên bản: v1.1 • Ngày: 31/01/2026 • Kiến trúc: Razor Pages + Web API • Phạm vi:

1. Mục tiêu & nguyên tắc chia việc

Mục tiêu: chia theo module rõ ràng (bounded context), giảm chồng chéo, công việc tương đối cân bằng giữa 4 dev.
Nguyên tắc: mỗi route/API có "owner" rõ ràng; các module giao tiếp qua event contract; realtime dùng cho Notification/Status update, không làm chat người-người.

3. Kiến trúc Razor Pages + Web API (NestFlow)

Giải pháp triển khai theo mô hình: Razor Pages (UI) + Web API (Controllers) dùng chung service layer (Application).

Cấu trúc thư mục khuyến nghị

• Controllers/: Web API (JSON) — chỉ nhận/validate request, gọi Application services.

• Pages/: Razor Pages — UI theo role (Admin/Landlord/Renter). Có thể tách Areas để dễ quản lý.

• Application/: Business logic, use-cases, domain services, integrations (Payment, AI, Media).

• Database/: EF Core DbContext, migrations, seed data.

• Models/: Entities + DTO/ViewModel (tách rõ Entity vs DTO để tránh lộ nội bộ).

• wwwroot/: static assets (JS/CSS) — viewer 360°, map scripts, AI chat UI scripts.

Realtime: dùng SignalR (WebSocket + fallback) cho Notification/Status updates; KHÔNG làm chat người-người.

4. Phân chia theo Dev

Dev 1 (Tùng) — Core Platform + Auth/RBAC + Admin (UI + API)

Owner folders: Controllers/Auth*, Middleware/RBAC*, Pages/Account/*, Pages/Admin/*.

Chuẩn hoá response/error + logging/audit tối thiểu để các module khác dùng chung.

- Auth: đăng ký/đăng nhập/refresh/forgot-reset; email verify/reset.
- RBAC + ownership guard middleware dùng chung toàn hệ thống.
- Quản lý thông tin user (profile, trạng thái, role).
- Admin console: quản lý user; moderation listing/review/report; giám sát & xử lý nghiệp vụ admin (ví dụ duyệt withdraw).
- Chuẩn hoá conventions: error format, logging/audit tối thiểu, pagination.

Dev 2 (Kiên) — Inventory/Search/Map + Media + 360° (UI + API)

Owner folders: Controllers/Properties*, Listings*, Search*, Media*; Pages/Properties/*, Pages/Listings/*, Pages/Search/*; wwwroot/js/360-viewer*.

360° (phương án A): upload 1 ảnh equirectangular → lưu metadata → hiển thị viewer panorama trên trang chi tiết listing.

- Property & Listing: CRUD (landlord), listing lifecycle (active/inactive/expired), view count.
- Search/filter nâng cao (SQL tối ưu hoặc Elasticsearch) + API trả dữ liệu cho AI search.
- Tích hợp Map API: geocoding/toạ độ, bounds query, marker data.
- Media service: upload/delete ảnh (S3/Cloudinary), album ảnh, thumbnail/compress.
- 360°: upload 1 ảnh equirectangular + lưu metadata + hiển thị viewer panorama trên trang chi tiết listing.

Dev 3 (Bắc) — AI Chat Tìm Phòng + Realtime (SignalR) + Notifications + Booking/Engagement

Owner folders: Controllers/Bookings*, Notifications*, Favorites*, Reviews*, Reports*, AiSearch*; SignalR Hubs/NotificationHub; Pages/Bookings/*, Pages/Notifications/*, Pages/AiSearch/*, Pages/Favorites/*.

AI chat tìm phòng (A): nhận câu hỏi tự nhiên → trích xuất điều kiện → gọi search của Dev2 → trả 5–10 listing + giải thích; hỏi thêm khi thiếu dữ liệu.

- AI chat tìm phòng (renter): nhận câu hỏi tự nhiên → trích xuất điều kiện → gọi search của Dev2 → trả 5–10 listing + giải thích; hỏi thêm khi thiếu dữ liệu.
- Lưu conversation history + feedback (helpful/not) + logging cho truy vết.
- SignalR Hub (NotificationHub): auth JWT; kênh theo userId/role; push realtime cho Notification Center (không chat người-người).
- Notification Center: lưu DB + API list/mark-read; realtime push khi booking/invoice/payment/withdraw/subscription đổi trạng thái.
- Booking workflow: create/cancel; landlord confirm/reject; complete; state machine + validation.

Dev 4 (Duy) — Rentals + Invoices + Payments + Wallet/Withdraw + Plans (UI + API)

Owner folders: Controllers/Payments*, Rentals*, Invoices*, Wallet*, Withdraws*, Plans*; Pages/Landlord/Rentals/*, Pages/Landlord/Invoices/*, Pages/Landlord/Wallet/*, Pages/Plans/*.

Webhooks/payment phải idempotent; mọi thay đổi số dư đi qua WalletTransactions (sổ cái).

- Deposit/payment webhook → tạo rental; quản lý trạng thái payment; idempotency cho webhook.
- Rental lifecycle: gia hạn/kết thúc; occupants; rent schedules (due date, mark-paid offline).
- Invoice engine: điện/nước/phí dịch vụ; issue/paid/overdue; xuất báo cáo doanh thu (Excel/PDF nếu có).
- Tích hợp cổng thanh toán (theo hệ thống team chọn) + xử lý callback/webhook.
- Wallet ledger + withdraw flow; plans/subscription + quota listing.

5. Ranh giới API (route ownership)

Quy ước: Controllers chỉ nhận request/DTO; logic đặt ở Application services. Razor Pages gọi API nội bộ hoặc trực tiếp service (khuyến nghị gọi service để reuse).

Mỗi nhóm chịu trách nhiệm chính cho route (Controllers) và trang (Pages) sau (người khác chỉ gọi, không sửa logic bên trong nếu không bàn trước):

Dev	Route/API ownership
Dev 1	/auth/*, /users/* (core), /admin/*
Dev 2	/properties/*, /listings/*, /search/*, /map/*, /media/*, /360/*
Dev 3	/ai-search/*, /conversations/*, /bookings/*, /notifications/*, /favorites/*, /reviews/*, /reports/*, SignalR Hub
Dev 4	/payments/*, /rentals/*, /occupants/*, /rent-schedules/*, /invoices/*, /wallet/*, /withdraws/*, /plans/*, /subscription/*

6. Event contract tối thiểu (để nối module & realtime)

Các module phát event (emit) khi trạng thái đổi; Dev 3 nhận event để tạo notification + push realtime.

- booking.created | booking.confirmed | booking.rejected | booking.cancelled | booking.completed
- listing.status_changed (optional)
- payment.paid | payment.failed (deposit/invoice/subscription)
- invoice.issued | invoice.paid | invoice.overdue
- withdraw.requested | withdraw.approved | withdraw.rejected | withdraw.completed
- subscription.purchased | subscription.quota_changed

6. Ghi chú triển khai 360°

- Input: 1 ảnh equirectangular (tỉ lệ 2:1, ví dụ 6000x3000).
- Backend: validate format/kích thước; nén + tạo thumbnail; lưu metadata (type=360, url, panoId).
- Frontend: viewer panorama (pannellum hoặc three.js) trên listing detail; fallback ảnh thường nếu thiếu 360.
- Quy ước: mỗi listing có thể có 0..n pano 360 (giai đoạn 1 có thể giới hạn 1 pano).

7. Ghi chú triển khai AI chat tìm phòng

- Đầu ra: 5–10 listing phù hợp + lý do ngắn (giá, khu vực, tiện ích, khoảng cách…).
- Luôn dựa trên search thật (Dev2) để tránh bịa; AI chỉ đóng vai trò trích xuất điều kiện + giải thích + hỏi thêm.
- Có cơ chế 'clarifying questions' khi thiếu ràng buộc (ngân sách, khu vực, loại phòng).
- Lưu conversation history để người dùng xem lại; có feedback để cải thiện.
- Giới hạn an toàn: không tiết lộ dữ liệu nhạy cảm; không trả thông tin cá nhân ngoài dữ liệu listing/public.

Kết thúc tài liệu.