دانشگاه زنجان

دانشکدهٔ مهندسی

پایان‌نامهٔ دورهٔ کارشناسی

گروه مهندسی کامپیوتر

طراحی و پیاده‌سازی سامانهٔ جست‌وجو و ثبت آگهی بیلبورد

نگارش

علیرضا زارعیان

استاد راهنما

دکتر سجاد حق‌زاد کلیدبری

شهریور ۱۴۰۵

تعهدنامهٔ اصالت اثر

اینجانب **علیرضا زارعیان** متعهد می‌شوم که مطالب مندرج در این پایان‌نامه با عنوان «طراحی و پیاده‌سازی سامانهٔ جست‌وجو و ثبت آگهی بیلبورد» حاصل کار پژوهشی اینجانب است و به دستاوردهای پژوهشی دیگران که در این پژوهش از آن‌ها استفاده شده است، مطابق مقررات ارجاع و در فهرست منابع و مآخذ ذکر گردیده است. این پایان‌نامه قبلاً برای احراز هیچ مدرک هم‌سطح یا بالاتر ارائه نشده است. در صورت اثبات تخلف (در هر زمان) مدرک تحصیلی صادرشده توسط دانشگاه از اعتبار ساقط خواهد شد. کلیهٔ حقوق مادی و معنوی این اثر متعلق به دانشگاه زنجان می‌باشد.

نام و نام خانوادگی دانشجو: **علیرضا زارعیان**

امضا:

تاریخ: شهریور ۱۴۰۵

فرم افشای میزان و نحوهٔ استفاده از هوش مصنوعی

در تهیهٔ این پروژه از دستیار هوش مصنوعی استفاده شده است. جزئیات به شرح زیر اعلام می‌شود:

حوزه	میزان و نحوهٔ استفاده
ابزار مورد استفاده	دستیار برنامه‌نویسی مبتنی بر مدل زبانی بزرگ، در محیط خط فرمان و ویرایشگر کد.
نگارش کد	بخش قابل توجهی از کد با کمک دستیار نوشته شده است. نیازمندی‌ها، معماری، تصمیم‌های فنی و پذیرش یا رد هر پیشنهاد بر عهدهٔ نگارنده بوده است. برای جلوگیری از انحراف دستیار از قواعد پروژه، دو سند قید و شرط (`CLAUDE.md` و `AGENTS.md`) در مخزن نوشته شده که ده قاعدهٔ غیرقابل نقض پروژه را در هر جلسه به دستیار تحمیل می‌کند.
نگارش متن پایان‌نامه	ساختاردهی، ویرایش و تنظیم متن با کمک دستیار انجام شده است. مادهٔ خام همهٔ فصل‌ها مستندات فنی خودِ پروژه است که در طول توسعه نوشته شده‌اند. هیچ بخشی از این متن ترجمهٔ ماشینی یک منبع دیگر نیست و هیچ مرجعی بدون مطالعه ارجاع داده نشده است.
اعداد و ادعاهای متن	هیچ عدد یا ادعایی از مدل گرفته نشده است. تمام آمار این سند در زمان نگارش با اجرای مستقیم روی پروژه اندازه‌گیری شده است: شمارش رکوردها با پرس‌وجوی `SQLite` ، شمارش مسیرها و خطوط کد با پیمایش مخزن، و نتیجهٔ آزمون‌ها با اجرای `npm test` . روش هر اندازه‌گیری در فصل ۶ آمده است.

موضع نگارنده

دستیار هوش مصنوعی در این پروژه نقش یک ابزار پرسرعت را داشته است، نه نقش تصمیم‌گیرنده. آنچه ارزش مهندسی این کار را می‌سازد — انتخاب معماری، تشخیص اینکه کدام قابلیت ساخته نشود، و پیدا کردن خطاهایی که ابزار خودش تولید کرده بود — در فصل‌های ۴ تا ۶ مستند شده است. نمونهٔ روشنش در بخش ۳-۶ آمده: یک کلاس خطا که دستیار مکرراً تولید می‌کرد، شناسایی و به یک قاعدهٔ دائمی در `AGENTS.md` تبدیل شد تا دوباره تکرار نشود.

نام و نام خانوادگی دانشجو: **علیرضا زارعیان**

امضا:

تاریخ: شهریور ۱۴۰۵

تقدیم به پدر و مادرم

که پیش از آنکه من به این راه ایمان بیاورم، به من ایمان آوردند.

تشکر و قدردانی

از استاد راهنمای این پایان‌نامه، دکتر سجاد حق‌زاد کلیدبری، برای راهنمایی، وقتی که گذاشتند و نقدهایی که مسیر این کار را اصلاح کرد صمیمانه سپاسگزارم.

از اساتید گروه مهندسی کامپیوتر دانشگاه زنجان که در طول این دوره آموختند، و از داوران محترم که وقت خود را صرف مطالعهٔ این نوشته کرده‌اند، قدردانی می‌کنم.

و از خانواده‌ام، که شرط لازم هر چیزی بودند که در این سال‌ها ساخته شد.

چکیده

بازار رسانه‌های تبلیغاتی محیطی (بیلبورد، پل عابر، استرابورد و نمایشگر شهری) در ایران بازاری پراکنده است. اطلاعات هر رسانه نزد مالک یا پیمانکار همان رسانه است و خریدارِ تبلیغات برای یافتن گزینهٔ مناسب باید جداگانه با چند واسطه تماس بگیرد. نتیجه، نبودِ یک نمای واحد و قابل جست‌وجو از عرضهٔ موجود است.

در این پایان‌نامه یک سامانهٔ تحت وب طراحی و پیاده‌سازی شده که این عرضه را در یک فهرست یکپارچهٔ قابل جست‌وجو گرد می‌آورد. سامانه دو گروه کاربر دارد: خریدار تبلیغات که میان ۳٬۵۳۶ رسانه در ۱۰۱ شهر جست‌وجو و فیلتر می‌کند، جزئیات و موقعیت هر رسانه را روی نقشه می‌بیند و پس از احراز هویت شمارهٔ تماس مالک را دریافت می‌کند؛ و صاحب رسانه که آگهی خود را از راه یک فرم چندمرحله‌ای ثبت می‌کند تا پس از بررسی کارشناس منتشر شود.

سامانه با Next.js 16 و React 19 در یک پایگاه کد یکپارچه پیاده‌سازی شده است؛ لایهٔ داده روی SQLite از طریق Prisma 7، و احراز هویت با توکن امضاشده در کوکی HttpOnly. رابط کاربری کاملاً فارسی و راست‌به‌چپ است. دادهٔ اولیه با یک خزندهٔ Python از سه منبع عمومی گردآوری و پس از حذف تکراری‌ها یکپارچه شده است.

تمرکز اصلی کار بر کیفیت مهندسی سمت سرور بوده است: یک پایپلاینِ ثابت و غیرقابل جابه‌جایی برای هر مسیر برنامه‌نویسی، اعتبارسنجی طرح‌واره‌ای روی تمام ورودی‌ها، محدودسازی نرخ درخواست با شناسهٔ غیرقابل جعل، تضمین یکتایی عملیات غیرخودتوان، و ثبت رد ممیزی برای هر تغییر مدیریتی. صحت سامانه با ۱۱۳ آزمون خودکار در سطح واسط برنامه‌نویسی سنجیده شده که همگی روی یک ساخت تولیدی اجرا و تأیید شده‌اند. اندازه‌گیری کارایی نشان داد اجرای پروژه در حالت ساخت‌شده به‌ازای بازدید نخست از ده مسیر، حدود ۹۷ برابر کمتر از حالت توسعه پردازنده مصرف می‌کند.

کلیدواژه: تبلیغات محیطی، بازارگاه تحت وب، Next.js، رندر سمت سرور، امنیت واسط برنامه‌نویسی، همروندی، SQLite

مسئولیت اجتماعی و اثرگذاری پژوهش

این سامانه اطلاعاتی را که امروز پراکنده و در اختیار تعداد کمی واسطه است، در یک نمای واحد و رایگان برای جست‌وجو قرار می‌دهد. اثر مستقیم آن کاهش نابرابری اطلاعاتی میان خریدار و فروشندهٔ تبلیغات است: کسب‌وکار کوچکی که بودجهٔ محدودی دارد می‌تواند پیش از تماس با هر واسطه، دامنهٔ قیمت و گزینه‌های شهر خود را ببیند.

در طراحی، سه ملاحظهٔ اخلاقی صریح رعایت شده است. نخست، **حفاظت از دادهٔ شخصی**: شمارهٔ تماس مالکان در صفحهٔ عمومی قرار نمی‌گیرد و تنها پس از احراز هویت و ثبت درخواست در دسترس است، تا فهرست تماس‌ها به‌صورت انبوه برداشت نشود. دوم، **صداقت در داده**: رکوردهایی که از منابع عمومی گردآوری شده‌اند به‌عنوان دادهٔ گردآوری‌شده نشانه‌گذاری شده‌اند و به‌جای آنکه ادعای دقتِ تأییدشده داشته باشند، وضعیت کیفیت‌شان در پنل مدیریت قابل پیگیری است. سوم، **پرهیز از ادعای نادرست**: سامانه واسطهٔ مالی نیست و مالک رسانه‌ها نیست، بنابراین عمداً جریان «رزرو آنلاین» ندارد؛ دلیل این تصمیم در بخش ۴-۳ آمده است.

محدودیت‌های صادقانهٔ کار (از جمله اینکه دادهٔ اولیه یک گردآوری نقطه‌ای است و نه یک خوراک زنده) در فصل ۷ فهرست شده‌اند تا خواننده تصویر واقع‌بینانه‌ای از دامنهٔ اعتبار نتایج داشته باشد.

فهرست مطالب

فصل ۱ مقدمه ۱
۱-۱ بیان مسئله ۱
۱-۲ اهمیت و انگیزه ۱
۱-۳ مزیت رقابتی و نوآوری ۱
۱-۴ اهداف پژوهش ۲
۱-۵ دامنه و ساختار
فصل ۲ پیشینه و مرور فناوری‌ها ۴
۲-۱ بازار رسانهٔ محیطی و منابع دادهٔ موجود ۴
۲-۲ سامانه‌های مشابه ۴
۲-۳ انتخاب پشتهٔ فناوری ۴
فصل ۳ تحلیل نیازمندی‌ها ۶
۳-۱ ذی‌نفعان و نقش‌ها ۶
۳-۲ نیازمندی‌های وظیفه‌ای ۶
۳-۳ نیازمندی‌های غیروظیفه‌ای ۷
۳-۴ موارد کاربرد اصلی ۷
۳-۵ آنچه عمداً ساخته نشد ۷
فصل ۴ طراحی سامانه ۹
۴-۱ معماری کلی و دو مسیر داده ۹
۴-۲ موتور پایگاه داده و مدل داده ۱۰
۴-۳ طراحی واسط برنامه‌نویسی ۱۱
۴-۴ طراحی رابط کاربری ۱۲
۴-۵ سازمان‌دهی کد و نقشهٔ وابستگی‌ها ۱۳
فصل ۵ پیاده‌سازی ۱۹
۵-۱ پایپلاینِ ثابت هر درخواست ۱۹
۵-۲ احراز هویت و کنترل دسترسی ۲۰
۵-۳ ثبت آگهی و بارگذاری امن فایل ۲۱
۵-۴ محافظت از دادهٔ تماس مالکان ۲۳
۵-۵ بازیابی گذرواژه با رمز یک‌بارمصرف ۲۴
۵-۶ همروندی و صحت زیر بار هم‌زمان ۲۵
۵-۷ مدل تخمین بازدید روزانه
۵-۸ مدیریت خطا و رصدپذیری ۲۸
فصل ۶ ارزیابی و آزمون ۲۹
۶-۱ روش و نتایج آزمون خودکار ۲۹
۶-۲ یک کلاس خطا و درسی که از آن گرفته شد

۶-۳ اندازه‌گیری کارایی ۳۰
۶-۴ خودارزیابی ۳۱
فصل ۷ نتیجه‌گیری و کار آینده ۳۲
۷-۱ دستاوردها ۳۲
۷-۲ محدودیت‌ها ۳۲
۷-۳ کار آینده ۳۲
۷-۴ جمع‌بندی ۳۳
فهرست مراجع ۳۲
فصل ۸ راهنمای اجرای پروژه ۳۵
۸-۱ پیش‌نیازها و راه‌اندازی ۳۵
فصل ۹ نماهایی از سامانه ۳۶

فهرست شکل‌ها

شکل ۴-۱ معماری کلی سامانه. هر دو مسیر داده از یک لایهٔ دسترسی داده عبور می‌کنند؛ نگهبان لبه پیش از هر چیز روی مسیر قرار دارد. ۹

شکل ۴-۲ نمودار موجودیت-رابطه. سه موجودیت پایین‌سمت (کلید یکتایی، رمز یک‌بارمصرف و رد ممیزی) زیرساخت صحت و امنیت‌اند و رابطهٔ کسب‌وکاری ندارند.

شکل ۴-۳ صفحهٔ جست‌وجو و پالایش. پالایه‌های شهر، نوع، قیمت و ابعاد بالای شبکهٔ نتایج؛ هر کارت قیمت ماهانه و برآورد بازدید روزانه را نشان می‌دهد. ۱۲

شکل ۴-۴ صفحهٔ جزئیات رسانه: گالری تصویر، مشخصات فنی، موقعیت روی نقشه، و پنل «آنالیز ترافیک» که برآورد بازدید روزانه و مبنای محاسبه‌اش را می‌آورد. شمارهٔ تماس در این صفحه نیست؛ جایش دکمهٔ «برای دیدن اطلاعات تماس وارد شوید» است.

شکل ۴-۵ ساختار پوشه‌های پروژه. نکتهٔ کلیدی این چیدمان آن است که lib/db/ تنها راه رسیدن به پایگاه داده است؛ نه صفحه‌ای و نه مسیری، اتصال جداگانه‌ای نمی‌سازد. ۱۴

شکل ۴-۶ سفر یک درخواست از کلیک کاربر تا پایگاه داده و برگشت. صفحهٔ جزئیات رسانه تنها استثناست: چون مؤلفهٔ سمت سرور است، مستقیم lib/db/billboards.ts را صدا می‌زند و دو جعبهٔ میانی را رد می‌کند.

شکل ۴-۷ گراف وابستگی پروژه، استخراج‌شده با پیمایش خودکار مخزن. لایه‌ها از بالا به پایین: نگهبان لبه، صفحه‌ها، مؤلفه‌های رابط، مسیرهای برنامه‌نویسی، احراز هویت، ابزارهای مشترک، لایهٔ داده، طرح‌واره و پیکربندی.

شکل ۴-۸ نمای کلی پنل مدیریت. اعدادِ این صفحه مستقیم از پایگاه داده خوانده می‌شوند و همان‌هایی‌اند که در سراسر این نوشته نقل شده‌اند: ۳٬۵۳۶ رسانه، ۳٬۰۲۰ دارای مختصات (۸۵٪)، و سهم هر منبع. ۱۶

شکل ۴-۹ همان رابط روی نمایشگر تلفن همراه. چیدمان، پالایه‌ها و کارت‌ها برای عرض کم بازآرایی می‌شوند، نه اینکه کوچک شوند.

شکل ۵-۱ پایپلاینِ ثابت هر درخواست مدیریتی. ترتیب این مراحل غیرقابل جابه‌جایی است. ۱۹

قطعه کد ۵-۱ پایپلاین در عمل، از مسیر تصمیم کارشناس. ترتیب پنج مرحله در همهٔ مسیرهای مدیریتی یکسان است.

قطعه کد ۵-۲ بستنِ نشتِ زمانی: در حالت «کاربر وجود ندارد» هم یک درهم‌سازی روی مقدار ثابت انجام می‌شود، تا مدت پاسخ نگوید کدام شماره در سامانه ثبت است.

شکل ۵-۲ جریان ورود. پیام خطا در هر دو حالت «شماره ناموجود» و «گذرواژهٔ اشتباه» یکسان است تا شمارش کاربران ممکن نشود. ۲۱

قطعه کد ۵-۳ تشخیص نوع تصویر از بایت‌های آغازین خودِ فایل. نوع اعلام‌شدهٔ مرورگر اصلاً مبنا قرار نمی‌گیرد.

شکل ۵-۳ ثبت آگهی. کلید یکتایی تضمین می‌کند ارسال دوبارهٔ فرم، آگهی دوم نسازد. ۲۲

شکل ۵-۴ فرم ثبت آگهی از دید صاحب رسانه. هر مرحله پیش از رفتن به بعدی اعتبارسنجی می‌شود و دکمهٔ ارسال هنگام پرواز درخواست غیرفعال می‌ماند.

شکل ۵-۵ دریافت شمارهٔ تماس. سه لایه (نبودِ شماره در صفحه، الزام ورود، و سقف درخواست) برداشت انبوه را پرهزینه می‌کنند.

شکل ۶-۵ بازیابی گذرواژه. لایهٔ پیامک بدون کلید سرویس در حالت خواب می‌ماند؛ کد کامل است ولی پیامی ارسال نمی‌شود. ۲۵
قطعه کد ۴-۵ تصمیم تک‌شلیک. تلاش دوم برای همان گذار، پاسخ «تعارض» می‌گیرد.
شکل ۷-۵ ماشین حالت آگهی. هر گذار تک‌شلیک است؛ تلاش دوم برای همان گذار پاسخ «تعارض» می‌گیرد. ۲۶
شکل ۸-۵ همان ماشین حالت در عمل: نشان‌های «در انتظار تأیید»، «رد شده» و «در انتظار پرداخت» کنار سه تصمیم مجاز کارشناس. توضیح برای فرستنده در حالت «نیاز به اصلاح» و «رد» الزامی است. ۲۶
شکل ۱-۹ سمت کاربر: صفحهٔ نخست با آمار کلی و رسانه‌های شاخص، و پیشخوانی که وضعیت هر آگهی ثبت‌شده را در چرخهٔ تأیید نشان می‌دهد.
شکل ۲-۹ صفحهٔ تحلیل توزیع شهر، نوع و قیمت را از دادهٔ واقعی رسم می‌کند. مرجع کامل ۴۲ نقطهٔ پایانی درون خودِ سامانه سرو می‌شود و نیازی به سند بیرونی ندارد.
شکل ۳-۹ پنل مدیریت: ویرایش رسانه‌ها، و صفحهٔ کیفیت که رکوردهای ناقص (بدون تصویر یا بدون مختصات) را برای پیگیری جدا می‌کند.
شکل ۴-۹ چهار سطح دسترسی مدیران، و رد ممیزی که هر تغییر مدیریتی را با شناسهٔ مدیر، نشانی شبکه و جزئیات تغییر نگه می‌دارد.

فهرست جدول‌ها

جدول ۱-۱ مقایسهٔ ساختاری با وب‌گاه‌های تک‌فروشندهٔ موجود ۱
جدول ۲-۱ سهم منابع در پایگاه دادهٔ نهایی، پس از حذف رکوردهای تکراری ۴
جدول ۲-۲ انتخاب‌های فناوری، بدیل بررسی‌شده و دلیل انتخاب ۵
جدول ۳-۱ نقش‌های سامانه، هدف هر نقش و مرز دسترسی آن ۶
جدول ۳-۲ نیازمندی‌های وظیفه‌ای سامانه ۶
جدول ۳-۳ نیازمندی‌های غیروظیفه‌ای و معیار پذیرش هر یک ۷
جدول ۴-۱ توزیع نقاط پایانی بر حسب فعل HTTP ۱۱
جدول ۴-۲ لایه‌های کد، تعداد فایل و نقش هر لایه ۱۳
جدول ۴-۳ پرارجاع‌ترین فایل‌های پروژه — چند فایل به هر یک وابسته‌اند ۱۷
جدول ۵-۱ کلاس‌های معبر و ضرایب هر یک. ستون «سهم دیدن» نسبت جمعیتی است که امکان واقعی خواندن پیام را دارد. ۲۷
جدول ۶-۱ سرفصل‌های پوشش آزمون خودکار ۲۹
جدول ۶-۲ شش خطای هم‌ریشه و اصلاح هر یک ۳۰
جدول ۶-۳ هزینهٔ پردازنده برای نخستین بازدید از ده مسیر ۳۰
جدول ۶-۴ خودارزیابی محورهای پروژه ۳۱
جدول ۷-۱ کارهای آینده به ترتیب اولویت ۳۲

فهرست اختصارها و واژه‌نامه

API	واسط برنامه‌نویسی کاربردی — قراردادی که برنامه‌ها با آن به هم داده می‌دهند
CSRF	جعل درخواست بین‌سایتی
ERD	نمودار موجودیت-رابطه؛ نقشهٔ جدول‌های پایگاه داده
JWT	توکن وب JSON؛ برگهٔ نشست امضاشده
ORM	نگاشت شیء-رابطه‌ای؛ لایه‌ای که جدول‌ها را به اشیای برنامه تبدیل می‌کند
OTP	رمز یک‌بارمصرف
RBAC	کنترل دسترسی مبتنی بر نقش
RSC	مؤلفهٔ سمت سرور React؛ مؤلفه‌ای که فقط روی سرور اجرا می‌شود
RTL	راست‌به‌چپ؛ جهت نوشتار فارسی
SSR	رندر سمت سرور
WAL	ثبت پیشین نوشتن؛ حالتی از SQLite برای خواندن و نوشتن هم‌زمان
Audit log	رد ممیزی — ثبت ماندگار هر کنش حساس
Idempotency	یکتایی عملیات — تکرار یک درخواست، اثرش را تکرار نکند
Middleware / Proxy	لایهٔ میانی — کدی که پیش از رسیدن درخواست به مقصد اجرا می‌شود
Rate limiting	سقف نرخ درخواست — محدودکردن تعداد درخواست در بازهٔ زمانی
Server component	مؤلفهٔ سمت سرور — کدی که فقط روی سرور اجرا می‌شود
State machine	ماشین حالت — مجموعهٔ وضعیت‌ها و گذارهای مجاز میان آن‌ها
Transaction	تراکنش — گروهی از عملیات که یا همه انجام می‌شوند یا هیچ‌کدام

فصل یکم

مقدمه

۱-۱ بیان مسئله

تبلیغات محیطی (بیلبورد، تابلوی پل عابر، استرابورد ایستگاه و نمایشگر شهری) یکی از پرمخاطب‌ترین رسانه‌های تبلیغاتی است، اما برخلاف تبلیغات آنلاین که از یک پیشخوان واحد خریداری می‌شود، خرید آن در ایران هنوز فرایندی تلفنی و واسطه‌محور است.

ریشهٔ مسئله در ساختار مالکیت است: هر سازه در اختیار یک شهرداری، پیمانکار یا شرکت تبلیغاتی است و فهرست موجودی، قیمت و وضعیت آن نزد همان مالک می‌ماند. خریداری که می‌خواهد بداند در شهر خود چه گزینه‌هایی با چه ابعاد و قیمتی وجود دارد، ناچار است با چند واسطه جداگانه تماس بگیرد و پاسخ‌هایی را که در قالب‌های متفاوت داده می‌شود دستی مقایسه کند. پیامد عملی این پراکندگی سه چیز است: زمان تلف‌شده، نبودِ امکان مقایسهٔ منصفانه، و نابرابری اطلاعاتی به سود فروشنده.

> **صورت فشردهٔ مسئله**
>
> عرضهٔ رسانهٔ محیطی در ایران وجود دارد و کم هم نیست؛ آنچه وجود ندارد یک **نمای واحد و قابل جست‌وجو** از این عرضه است. مسئلهٔ این پایان‌نامه ساختن آن نما است.

۲-۱ اهمیت و انگیزه

در بازارهایی که عرضه پراکنده است، اثر یک فهرست متمرکز صرفاً «راحتی» نیست؛ ترکیب بازار را تغییر می‌دهد. کسب‌وکار کوچکی که بودجهٔ محدودی دارد امروز حتی نمی‌داند در شهر خودش گزینه‌ای در توان مالی او هست یا نه، چون هزینهٔ جست‌وجو برایش بالاتر از ارزش احتمالی نتیجه است؛ یک فهرست قابل جست‌وجو این هزینه را تقریباً صفر می‌کند. از منظر مهندسی نرم‌افزار نیز مسئله تمرین کاملی است: دادهٔ واقعی و کثیف، دو گروه کاربر با نیازهای متضاد، محتوای تولیدشده توسط کاربر که باید پیش از انتشار بررسی شود، دادهٔ شخصی‌ای که باید محافظت شود، و رابطی راست‌به‌چپ که بیشتر ابزارهای رایج برای آن بهینه نشده‌اند.

۳-۱ مزیت رقابتی و نوآوری

پیش از آنکه اهداف فهرست شود، باید روشن باشد این سامانه دقیقاً چه چیزی به وضعیت موجود اضافه می‌کند. وب‌گاه‌های امروزی رسانهٔ محیطی در ایران عمدتاً **وب‌گاه یک شرکت‌اند، نه یک بازار:** هر یک موجودی خودش را نشان می‌دهد و (همان‌طور که در گردآوری داده مشاهده شد) برای هر سازه یک عکس، یک اندازه، یک نام خیابان و گاهی یک قیمت منتشر می‌کند و نه بیشتر.

جدول ۱-۱ — مقایسهٔ ساختاری با وب‌گاه‌های تک‌فروشندهٔ موجود

ویژگی	وب‌گاه‌های موجود	این سامانه
دامنهٔ موجودی	فقط سازه‌های همان شرکت	تجمیع سه منبع در یک فهرست واحد: ۳,۵۳۶ رسانه در ۱۰۱ شهر

ویژگی	وب‌گاه‌های موجود	این سامانه
جست‌وجوی یکپارچه	کاربر باید چند وب‌گاه را جداگانه بگردد	یک پالایهٔ واحد روی شهر، نوع، دامنهٔ قیمت، ابعاد و وجود تصویر
نقشه	معمولاً فقط نام خیابان به‌صورت متن	نمایش هم‌زمان فهرست و نقشه؛ ۳٬۰۲۰ رسانه دارای مختصات جغرافیایی (۸۵٫۴٪)
مقایسهٔ کنار هم	وجود ندارد	انتخاب چند رسانه و مقایسهٔ ستونی مشخصات، قیمت و بازدید
تخمین بازدید روزانه	وجود ندارد	مدل شفاف مبتنی بر کلاس معبر — بخش ۷-۵
شفافیت قیمت	پراکنده؛ گاهی «تماس بگیرید»	قیمت در خودِ فهرست، قابل مرتب‌سازی و پالایش
ثبت آگهی خودخدمت	معمولاً تماس تلفنی	فرم چندمرحله‌ای با بارگذاری تصویر و چرخهٔ تأیید کارشناس
حفاظت از دادهٔ تماس	شماره در صفحهٔ عمومی	پشت احراز هویت، با ثبت درخواست و سقف نرخ
فارسی و راست‌به‌چپ	غالباً قالب ترجمه‌شده با ایرادهای چیدمانی	راست‌به‌چپ بومی در تمام چیدمان، جدول‌ها، نمادها و انیمیشن‌ها

نقطهٔ تمایز اصلی — تخمین بازدید روزانه

هیچ‌یک از منابع بررسی‌شده عددی برای «چند نفر در روز این سازه را می‌بینند» منتشر نمی‌کند، در حالی که این دقیقاً عددی است که خریدار تبلیغات برای سنجش ارزش خرید به آن نیاز دارد. در این پروژه مدلی ساخته شده که این عدد را از داده‌ای که *موجود است* (کلاس معبر، اندازهٔ سازه، نوع رسانه و شهر) برآورد می‌کند.

مدل از شکلی پیروی می‌کند که صنعت تبلیغات محیطی جهانی به کار می‌برد (Geopath در آمریکا و Route در بریتانیا): شمارش عبور، تبدیل آن به نفر، و سپس تنزیل بر اساس اینکه چه سهمی از این جمعیت واقعاً امکان دیدن این سازهٔ مشخص را دارد. خروجی آن **قطعی و بازتولیدپذیر** است (یک رسانه در دو اجرای متفاوت دقیقاً یک عدد می‌گیرد) و منابع هر ضریب در کد ثبت شده است. شرح کامل روش در بخش ۷-۵ آمده است.

۱-۴ اهداف پژوهش

هدف کلی، طراحی و پیاده‌سازی یک سامانهٔ تحت وب کارا و امن برای جست‌وجو و ثبت آگهی رسانه‌های تبلیغاتی محیطی است. اهداف جزئی:

1. گردآوری و یکپارچه‌سازی دادهٔ رسانه‌ها از منابع عمومی موجود در یک پایگاه دادهٔ واحد؛
2. جست‌وجو، پالایش و مقایسه بر پایهٔ شهر، نوع، ابعاد، قیمت و موقعیت جغرافیایی؛
3. چرخهٔ خودخدمتِ ثبت آگهی، همراه با بررسی و تأیید کارشناس پیش از انتشار؛
4. محافظت از دادهٔ تماس مالکان در برابر برداشت انبوه، بدون دشوار کردن دسترسی کاربر واقعی؛
5. کیفیت مهندسی قابل دفاع در برابر خطا، سوءاستفاده و بار هم‌زمان؛
6. سنجش صحت با آزمون خودکار و سنجش کارایی با اندازه‌گیری واقعی، نه تخمین.

۵-۱ دامنه و ساختار

مرز این پروژه آگاهانه تعیین شده است. سامانه یک **فهرست و بازارگاه اطلاعاتی** است: کاربر را تا لحظهٔ تماس با مالک می‌رساند و توافق از آنجا بیرون از سامانه انجام می‌شود. سامانه مالک هیچ رسانه‌ای نیست و تقویم اشغال هیچ سازه‌ای را در اختیار ندارد، بنابراین جریان «رزرو آنلاین» عمداً ساخته نشده است (بخش ۵-۳). همچنین برای بار یک نمونهٔ اجرایی و ترافیک خواندن‌محور طراحی شده، نه برای مقیاس چندسروری (بخش ۲-۴ و فصل ۷).

فصل ۲ بازار هدف، سامانه‌های مشابه و دلیل انتخاب هر فناوری را بررسی می‌کند؛ **فصل ۳** ذی‌نفعان، نیازمندی‌ها و موارد کاربرد را استخراج می‌کند؛ **فصل ۴** طراحی را در چهار سطح معماری، مدل داده، واسط برنامه‌نویسی و رابط کاربری ارائه می‌دهد؛ **فصل ۵** به پیاده‌سازی و تصمیم‌های مهندسی می‌پردازد؛ **فصل ۶** روش و نتایج ارزیابی را گزارش می‌کند و **فصل ۷** جمع‌بندی می‌کند.

پیشینه و مرور فناوری‌ها

۲-۱ بازار رسانهٔ محیطی و منابع دادهٔ موجود

در ایران چند وب‌گاه تخصصی وجود دارد که موجودی رسانه‌های محیطی را منتشر می‌کنند. این وب‌گاه‌ها معمولاً متعلق به یک شرکت تبلیغاتی‌اند و طبیعتاً فهرست خودشان را نشان می‌دهند، نه فهرست بازار را. افزون بر آن، آگهی‌های پراکنده‌ای هم در بسترهای عمومی نیازمندی درج می‌شود که ساختار یکسانی ندارند.

برای این پروژه دادهٔ اولیه از سه منبع عمومی گردآوری شد. جدول زیر سهم هر منبع را در پایگاه دادهٔ نهایی نشان می‌دهد.

جدول ۲-۱ — سهم منابع در پایگاه دادهٔ نهایی، پس از حذف رکوردهای تکراری

منبع	تعداد رکورد	سهم
`billboardiha`	۲٬۷۸۱	۷۸٪
`aradholding`	۶۱۵	۱۷٫۴٪
`irbillboard`	۱۱۵	۳٫۳٪
ثبت‌شده توسط کاربران سامانه	۸	۰٫۲٪
بدون منبع مشخص	۱۷	۰٫۵٪
مجموع	**۳٬۵۳۶**	**۱۰۰٪**

نکتهٔ مهم آنکه این رکوردها یک **گردآوری نقطه‌ای** است، نه خوراک زنده. هر رکورد با برچسب منبع خود نگهداری می‌شود تا هم قابل ردیابی باشد و هم بعداً بتوان همان منبع را به‌روزرسانی کرد. رکوردهایی که در بیش از یک منبع تکرار شده بودند با مقایسهٔ نام، شهر و موقعیت جغرافیایی شناسایی و ادغام شدند.

۲-۲ سامانه‌های مشابه

بازارگاه‌های تبلیغات محیطی معمولاً یکی از دو الگو را دنبال می‌کنند. الگوی نخست، **بازارگاه معاملاتی** است که موجودی و تقویم اشغال سازه‌ها را در اختیار دارد و خرید را تا پرداخت آنلاین پیش می‌برد. الگوی دوم، **فهرست اطلاعاتی** است که عرضه را قابل جست‌وجو می‌کند و طرفین را به هم می‌رساند بی‌آنکه خودش طرف معامله باشد.

الگوی نخست تنها زمانی کار می‌کند که سامانه به تقویم واقعی اشغال دسترسی داشته باشد، یعنی یا مالک باشد یا با مالکان یکپارچگی داده‌ای داشته باشد. در نبود این دسترسی، «رزرو آنلاین» چیزی جز یک وعدهٔ بی‌پشتوانه نیست. به همین دلیل الگوی دوم انتخاب شده است؛ این یک محدودیت فنی نیست، یک تصمیم طراحی است (بخش ۵-۳).

۲-۳ انتخاب پشتهٔ فناوری

هر انتخاب فناوری در این پروژه در برابر دست‌کم یک بدیل جدی سنجیده شده است. جدول زیر خلاصهٔ این سنجش است.

جدول ۲-۲ — انتخاب‌های فناوری، بدیل بررسی‌شده و دلیل انتخاب

لایه	انتخاب	بدیل	دلیل
چارچوب	Next.js 16 (App Router)	Django + DRF با یک رابط جدا	رابط کاربری، مسیرهای برنامه‌نویسی و رندر سمت سرور در یک پایگاه کد؛ یک بار استقرار به‌جای دو سرویس. برای یک توسعه‌دهندهٔ تنها با مهلت محدود، حذف مرز شبکه میان دو سرویس مهم‌ترین صرفه‌جویی است.
زبان	TypeScript در حالت strict	JavaScript	خطاهای نوع در زمان ساخت گرفته می‌شوند، نه در جلسهٔ دفاع. هزینه‌اش کندتر نوشتن است و در پروژه‌ای با یک نویسنده، ارزشش را دارد.
پایگاه داده	SQLite	PostgreSQL	بار خواندن‌محور، یک نمونهٔ اجرایی، بدون نیاز به سرویس جداگانه. مهاجرت بعدی هزینهٔ بازنویسی ندارد — بخش ۲-۴.
لایهٔ داده	Prisma 7	SQL دستی	طرح‌وارهٔ نوع‌دار، مهاجرت نسخه‌بندی‌شده، و مصونیت ساختاری از تزریق SQL.
اعتبارسنجی	Zod	بررسی دستی `if`	یک طرح‌واره هم نوع زمان ساخت می‌دهد هم بررسی زمان اجرا؛ یک منبع حقیقت به‌جای دو.
احراز هویت	توکن امضاشده در کوکی HttpOnly	سرویس احراز هویت میزبانی‌شده	پرهیز از وابستگی به سرویس پولی یا تحریم‌شده. کد آن حدود دویست خط است و کاملاً قابل بازرسی.
نقشه	Leaflet + کاشی نشان	نقشهٔ گوگل	کتابخانهٔ آزاد و ارائه‌دهندهٔ کاشی داخلی؛ بدون کلید پولی و بدون مسدودیت منطقه‌ای.

قید حاکم بر همهٔ انتخاب‌ها

در سراسر پروژه هیچ سرویس پولی یا مسدودشده برای ایران به کار نرفته است. هرجا بهترین‌روشِ رایج به چنین سرویسی نیاز داشته، یا معادل درون‌کدی آن ساخته شده، یا از آن صرف‌نظر و در مستندات ثبت شده که چه چیزی از دست رفته است. این قید کیفیت مهندسی را پایین نیاورده، ولی شکل بعضی راه‌حل‌ها را تعیین کرده است.

فصل سوم

تحلیل نیازمندی‌ها

۳-۱ ذی‌نفعان و نقش‌ها

سامانه چهار نقش دارد که دسترسی هر یک جداگانه تعریف شده است:

جدول ۳-۱ — نقش‌های سامانه، هدف هر نقش و مرز دسترسی آن

نقش	هدف	مرز دسترسی
مهمان	ارزیابی اینکه آیا سامانه به دردش می‌خورد	جست‌وجو، پالایش، دیدن جزئیات و نقشه. شمارهٔ تماس مالک را نمی‌بیند.
خریدار تبلیغات (کاربر واردشده)	یافتن رسانهٔ مناسب و تماس با مالک	همهٔ دسترسی‌های مهمان، به‌علاوهٔ دریافت شمارهٔ تماس، ثبت نظر و مقایسه.
صاحب رسانه	معرفی رسانهٔ خود به خریداران	ثبت آگهی با تصویر، پیگیری وضعیت آن در پیشخوان، و اصلاح آگهی در صورت درخواست کارشناس.
کارشناس / مدیر	حفظ کیفیت و صحت فهرست	چهار سطح پلکانی: `viewer` فقط می‌بیند، `editor` ویرایش می‌کند، `admin` تأیید و رد می‌کند، `super_admin` مدیران را مدیریت می‌کند.

۳-۲ نیازمندی‌های وظیفه‌ای

جدول ۳-۲ — نیازمندی‌های وظیفه‌ای سامانه

شناسه	نیازمندی	شرح
`FR-1`	جست‌وجو و پالایش	پالایش بر پایهٔ شهر، نوع رسانه، دامنهٔ قیمت، ابعاد و وجود تصویر؛ به‌همراه مرتب‌سازی و صفحه‌بندی.
`FR-2`	نمایش روی نقشه	نمایش هم‌زمان فهرست و نقشه، با موقعیت جغرافیایی هر رسانه.
`FR-3`	صفحهٔ جزئیات	گالری تصویر، مشخصات فنی، برآورد بازدید روزانه، موقعیت و نظرات کاربران.
`FR-4`	مقایسه	انتخاب چند رسانه و مقایسهٔ کنار هم مشخصات آن‌ها.
`FR-5`	ثبت‌نام و ورود	ثبت‌نام با شمارهٔ موبایل و گذرواژه؛ بازیابی گذرواژه با رمز یک‌بارمصرف پیامکی.
`FR-6`	دریافت شمارهٔ تماس	نمایش شمارهٔ مالک تنها به کاربر واردشده، همراه با ثبت درخواست.
`FR-7`	ثبت آگهی	فرم چندمرحله‌ای با بارگذاری تصویر و دو پلن رایگان و ویژه.

شناسه	نیازمندی	شرح
FR-8	چرخهٔ تأیید	ماشین حالت آگهی از ثبت تا انتشار، با امکان درخواست اصلاح.
FR-9	نظر و امتیاز	ثبت یک نظر برای هر کاربر روی هر رسانه، با امکان پاسخ مدیر.
FR-10	پنل مدیریت	مدیریت رسانه‌ها، صف تأیید، وضعیت کیفیت داده، کاربران و رد ممیزی.

۳-۳ نیازمندی‌های غیروظیفه‌ای

جدول ۳-۳ — نیازمندی‌های غیروظیفه‌ای و معیار پذیرش هر یک

شناسه	نیازمندی	معیار پذیرش
NFR-1	امنیت ورودی	هیچ ورودی کاربر بدون اعتبارسنجی طرح‌واره‌ای به لایهٔ داده نرسد.
NFR-2	کنترل دسترسی	مرز دسترسی روی خودِ مسیر برنامه‌نویسی اعمال شود، نه فقط در رابط کاربری.
NFR-3	مقاومت در برابر سوءاستفاده	سقف نرخ درخواست روی ورود، ثبت‌نام، بازیابی گذرواژه و مسیرهای پرهزینه.
NFR-4	صحت زیر بار هم‌زمان	ارسال دوبارهٔ یک درخواست نباید اثرش را دو بار اعمال کند.
NFR-5	مدیریت خطا	کاربر هرگز صفحهٔ سفید یا رد پشتهٔ خام نبیند؛ هر خطای پیش‌بینی‌نشده شناسهٔ پیگیری داشته باشد.
NFR-6	زبان، جهت و موبایل	تمام متن دیدنیِ کاربر فارسی و چیدمان راست‌به‌چپ باشد و همهٔ صفحه‌ها روی نمایشگر تلفن همراه قابل استفاده باشند.
NFR-7	کارایی	سامانه روی یک رایانهٔ شخصی معمولی و بدون شتاب‌دهندهٔ بیرونی روان اجرا شود.
NFR-8	ردیابی‌پذیری	هر تغییر مدیریتی در یک رد ممیزی ماندگار ثبت شود.

۴-۳ موارد کاربرد اصلی

پنج مورد کاربرد، ستون فقرات سامانه‌اند. هر یک در فصل ۵ با نمودار توالی کامل ارائه شده است:

1. **جست‌وجوی رسانه** — بازدیدکننده شهر و نوع را انتخاب می‌کند و به فهرست پالایش‌شده می‌رسد.
2. **ورود به سامانه** — کاربر با شماره و گذرواژه وارد می‌شود و نشست امضاشده دریافت می‌کند (شکل ۲-۵).
3. **دریافت شمارهٔ تماس** — کاربر واردشده شمارهٔ مالک را می‌گیرد و درخواستش ثبت می‌شود (شکل ۴-۵).
4. **ثبت آگهی توسط صاحب رسانه** — فرم چندمرحله‌ای، بارگذاری تصویر، و ورود به صف بررسی (شکل ۳-۵).
5. **بازیابی گذرواژه** — کاربر با رمز یک‌بارمصرف پیامکی گذرواژه‌اش را بازنشانی می‌کند (شکل ۵-۵).

۵-۳ آنچه عمداً ساخته نشد

بخشی از تحلیل نیازمندی‌ها، تشخیص نیازمندی‌هایی است که *نباید* برآورده شوند. سه قابلیت که در نگاه اول بدیهی به نظر می‌رسند از دامنهٔ کار خارج شده‌اند و دلیل هر یک ثبت شده است.

سه قابلیت که عمداً ساخته نشدند

۱. رزرو آنلاین. رزرو تنها وقتی معنا دارد که سامانه بداند سازه در بازهٔ خواسته‌شده آزاد است؛ این دانش نزد مالک است. فرم رزرو بدون آن یعنی تعهدی بی‌پشتوانه. سامانه کاربر را تا لحظهٔ تماس می‌رساند و همان‌جا متوقف می‌شود.

۲. درگاه پرداخت. اتصال به درگاه بانکی نیازمند نماد اعتماد الکترونیکی و شخصیت حقوقی است. به‌جای شبیه‌سازی یک درگاه جعلی که هیچ چیز واقعی را نمی‌آزماید، پلن ویژه با تأیید دستی کارشناس پیاده شده است: منطق کسب‌وکار واقعی است و فقط لحظهٔ تراکنش بیرون از سامانه رخ می‌دهد.

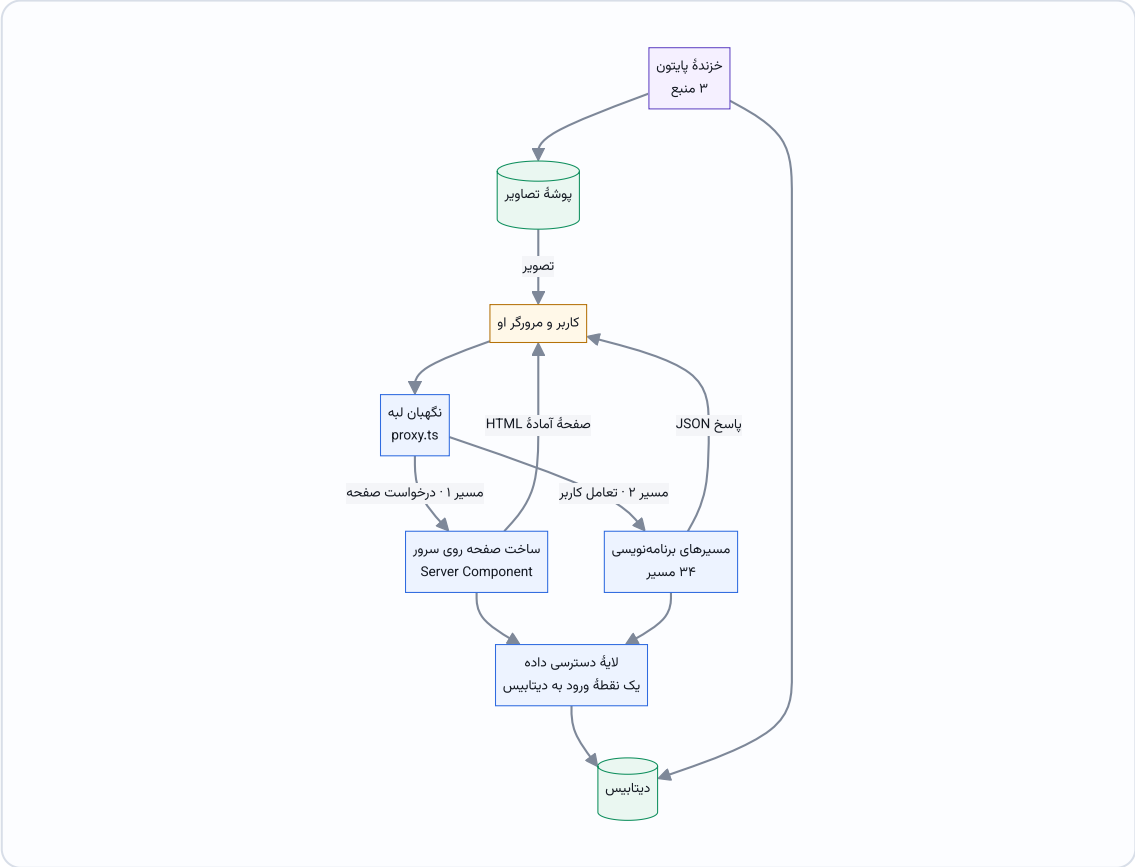
۳. سرویس‌های میزبانی‌شده. ابزارهای رصد خطا، تحلیل رفتار کاربر و احراز هویت به‌عنوان سرویس، یا پولی‌اند یا برای ایران در دسترس نیستند. معادل درون‌کدی آنچه لازم بود (ثبت ساخت‌یافتهٔ رخدادها و شناسهٔ خطا) ساخته و بقیه کنار گذاشته شده است.

فصل چهارم

طراحی سامانه

۴-۱ معماری کلی و دو مسیر داده

کل سامانه (رابط کاربری، مسیرهای برنامه‌نویسی و رندر سمت سرور) در یک پایگاه کد قرار دارد و به‌صورت یک واحد مستقر می‌شود. این یک چارچوب headless نیست؛ تفاوتی که پیامد معماری مهمی دارد و در ادامه توضیح داده می‌شود.



شکل ۴-۱ — معماری کلی سامانه. هر دو مسیر داده از یک لایهٔ دسترسی داده عبور می‌کنند؛ نگهبان لبه پیش از هر چیز روی مسیر قرار دارد.

۴-۱-۱ دو مسیر، یک لایهٔ داده

مسیر یکم: صفحهٔ جزئیات رسانه یک مؤلفهٔ سمت سرور است. هنگام ساخته‌شدن صفحه، سرور خودش تابع لایهٔ داده را صدا می‌زند و HTML آماده را می‌فرستد؛ فقط یک پرش، از سرور به پایگاه داده. **مسیر دوم:** هر تعامل کاربر پس از بارگذاری صفحه (پالایش فهرست، صفحهٔ بعد، ثبت آگهی، هر عملیات پنل مدیریت) با `fetch` به یکی از مسیرهای `/api/` می‌رود و پاسخ JSON می‌گیرد.

نکتهٔ کلیدی آنکه هر دو مسیر **یک لایهٔ داده** را صدا می‌زنند: یک منبع حقیقت، بدون منطق تکراری، و بدون حالتِ «صفحه کهنه است ولی مسیر برنامه‌نویسی تازه».

چرا هر صفحه‌ای مسیر برنامه‌نویسی خودش را صدا نمی‌زند؟

این نخستین پرسشی است که یک سامانهٔ Next.js برمی‌انگیزد، و پاسخ با یک تشبیه روشن می‌شود. یک واسط برنامه‌نویسی headless مانند **آشپزخانهٔ ابری** است: فقط ارسال بیرون، بدون سالن پذیرایی. هر بشقاب از دریچهٔ تحویل بیرون می‌رود، چون راه دیگری وجود ندارد — این خاصیتِ ساختمان است، نه نشانهٔ کیفیت.

این سامانه یک **رستوران با سرویس بیرون‌بر** است. مهمانی که سر میز نشسته (صفحه‌ای که در حال ساخته‌شدن است) غذایش را مستقیم از آشپزخانه می‌گیرد؛ بدون بسته‌بندی و بدون پیک — این مسیر یکم است. سفارش بیرون‌بر (مرورگری که پس از بارگذاری داده تازه می‌خواهد) بسته‌بندی می‌شود و پیک می‌بردش — این مسیر دوم است. آشپزخانه برای هر دو یکی است.

فرستادن بشقابِ مهمانِ سر میز از دریچهٔ تحویل به بیرون و برگرداندنش به داخل، فقط به‌خاطر یکدستی، کندتر و بی‌فایده است — و دقیقاً همان کاری است که عبور دادن رندر هر صفحه از `fetch("/api/…")` به آن می‌انجامد. مسیر یکم نه یک میان‌بر است و نه یک نقص؛ الگوی توصیه‌شدهٔ خودِ چارچوب و گزینهٔ سریع‌تر است.

تفاوت عملی چشمگیر است: مسیر یکم یک پرش شبکه دارد (سرور به پایگاه داده) و حدود ۵ تا ۲۰ میلی‌ثانیه هزینه می‌برد، در حالی که صفحه‌ای که مسیر برنامه‌نویسی خودش را صدا بزند دست‌کم دو پرش دارد، باید درخواست بسازد و JSON را تولید و دوباره تجزیه کند، و حدود ۴۰ تا ۱۵۰ میلی‌ثانیه می‌گیرد. افزون بر آن، تولید ایستا و انباره‌سازی در مسیر یکم کار می‌کند و در حالت دوم می‌شکند، چون به نشانی مطلق و سرورِ در حال اجرا نیاز دارد.

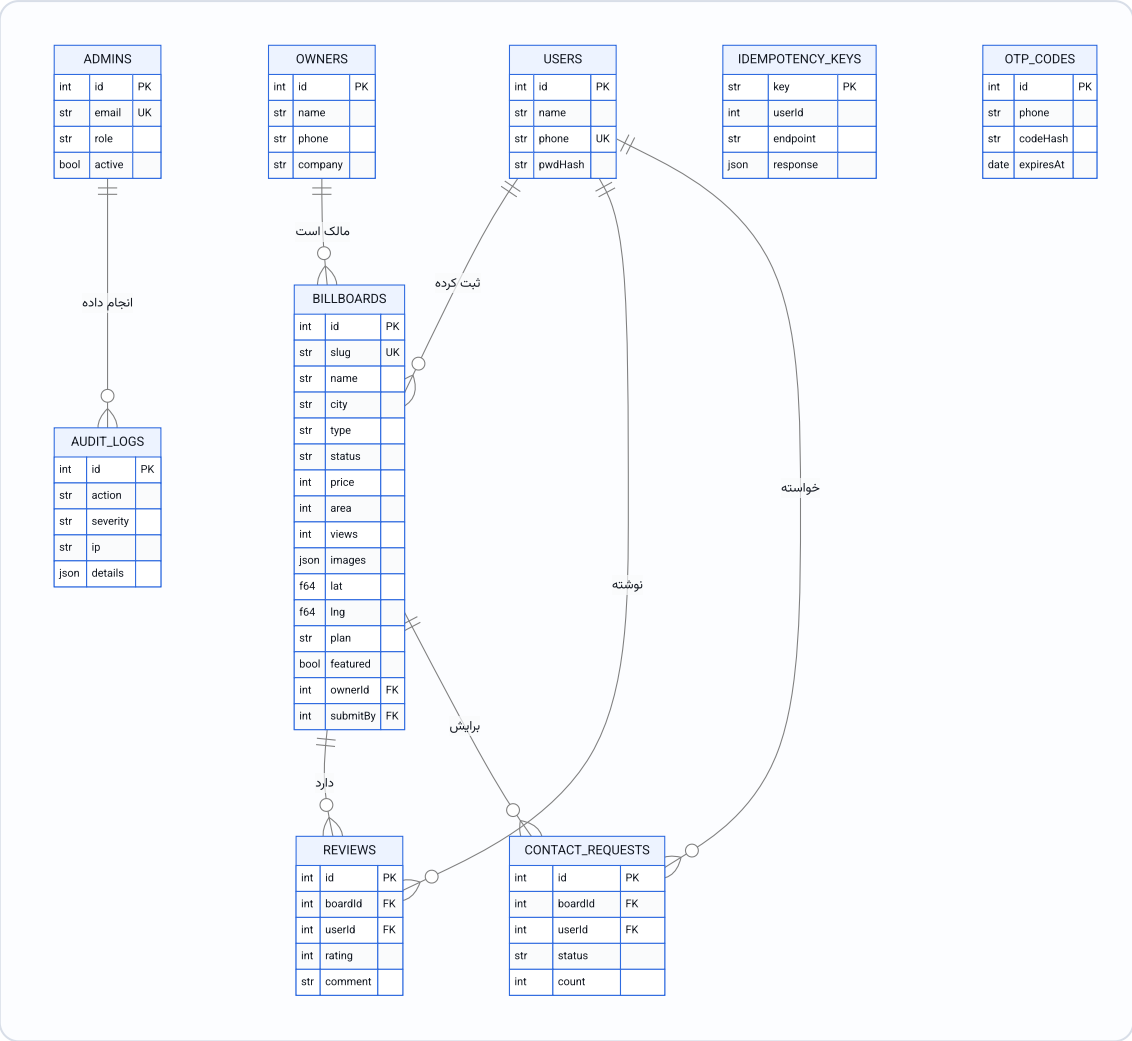
۲-۴ موتور پایگاه داده و مدل داده

موتور انتخاب‌شده SQLite است؛ نقطه‌ای که معمولاً نخستین فشار داور روی آن وارد می‌شود، پس استدلالش صریح می‌آید. بار این سامانه خواندن‌محور است (کاربران جست‌وجو و مطالعه می‌کنند و نوشتن رویدادی کم‌بسامد است) و روی یک نمونهٔ اجرایی مستقر می‌شود. در این شرایط SQLite در حالت WAL خواندن هم‌زمان را بدون قفل‌شدن پشتیبانی می‌کند و یک سرویس جداگانه، یک استخر اتصال و یک نقطهٔ خرابی دیگر را از معماری حذف می‌کند.

این انتخاب بن‌بست هم نیست: چون تمام دسترسی به داده از یک لایهٔ نگاشت شیء-رابطه‌ای عبور می‌کند و هیچ پرس‌وجوی خام SQL در کد پراکنده نیست، مهاجرت به PostgreSQL تغییر مقصد در پیکربندی و اجرای دوبارهٔ مهاجرت‌هاست، نه بازنویسی. آنچه امروز از دست می‌رود نوشتن هم‌زمان از چند نمونهٔ اجرایی است که این سامانه به آن نیاز ندارد.

۱۰	۲۵	۱۳	۳,۵۳۶
موجودیت	نمایه و قید یکتایی	مهاجرت نسخه‌بندی‌شده	رکورد رسانه

ستون مرکزی مدل، موجودیت `billboards` است. هر رسانه به یک مالک (`owners`) تعلق دارد و در صورتی که از راه فرم ثبت آگهی وارد شده باشد، به کاربر ثبت‌کننده هم پیوند می‌خورد. نظرها، درخواست‌های تماس و پاسخ مدیر همگی به همین موجودیت وصل می‌شوند. سه موجودیت باقی‌مانده (کلید یکتایی عملیات، رمزهای یک‌بارمصرف و رد ممیزی) به کسب‌وکار مربوط نیستند و زیرساخت صحت و امنیت را می‌سازند.



شکل ۲-۴ — نمودار موجودیت-رابطه. سه موجودیت پایین‌سمت (کلید یکتایی، رمز یک‌بارمصرف و رد ممیزی) زیرساخت صحت و امنیت‌اند و رابطهٔ کسب‌وکاری ندارند.

۴-۲-۱ نمایه‌گذاری بر پایهٔ پرس‌وجوهای واقعی

نمایه‌ها بر اساس شکل پرس‌وجوهایی که رابط کاربری واقعاً می‌سازد تعریف شده‌اند، نه به‌صورت حدسی روی تک‌تک ستون‌ها. سه نمایهٔ ترکیبی `(city,status)`، `(city,type)` و `(type,price)` دقیقاً پالایه‌های صفحهٔ جست‌وجو را پوشش می‌دهند. افزون بر آن، دو کلید مرتب‌سازی پرکاربرد در جدول ذخیره شده‌اند تا مرتب‌سازی به‌جای محاسبه در حافظه، در خودِ پایگاه داده انجام شود.

۴-۳ طراحی واسط برنامه‌نویسی

سامانه ۳۴ فایل مسیر دارد که روی هم ۴۲ نقطهٔ پایانی HTTP عرضه می‌کنند. طراحی از قرارداد REST پیروی می‌کند: منبع در نشانی، عمل در فعل.

جدول ۴-۱ — توزیع نقاط پایانی بر حسب فعل HTTP

فعل	تعداد	کاربرد
`GET`	۱۷	خواندن: فهرست رسانه‌ها، جزئیات، آمار، نظرها، داده‌های پنل مدیریت

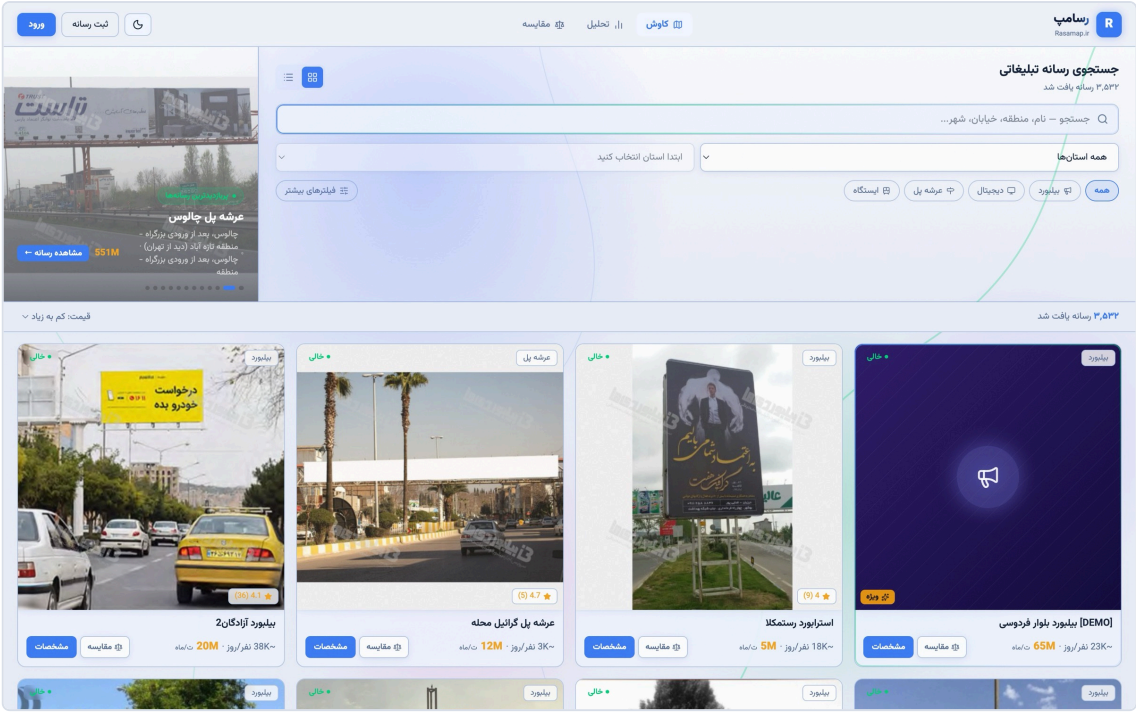
فعل	تعداد	کاربرد
`POST`	۱۵	ایجاد و کنش: ورود، ثبت‌نام، ثبت آگهی، درخواست تماس، ارسال رمز یک‌بارمصرف
`PATCH`	۵	تغییر جزئی: تأیید یا رد آگهی، تغییر نقش مدیر
`DELETE`	۳	حذف: رسانه، نظر، پاسخ
`PUT`	۲	جایگزینی کامل رکورد در پنل مدیریت
مجموع	**۴۲**	در ۳۴ فایل مسیر

یک تصمیم طراحی در این جدول پنهان است: دریافت شمارهٔ تماس با `POST` انجام می‌شود، نه `GET`. از نظر معنایی هم درست است (این عمل یک رکورد درخواست ایجاد می‌کند) و از نظر عملی، خواندن انبوه با یک حلقهٔ ساده روی نشانی‌ها را ناممکن می‌کند.

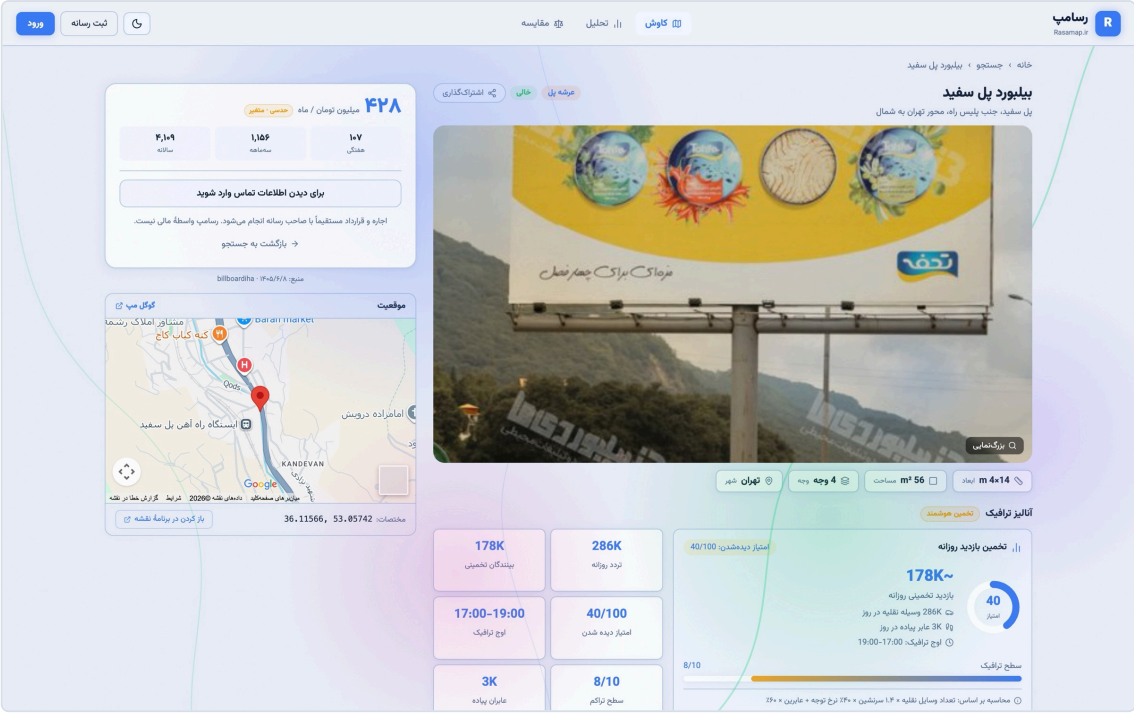
۴-۴ طراحی رابط کاربری

رابط کاربری کاملاً فارسی و راست‌به‌چپ است. این تصمیم فراتر از افزودن یک صفت جهت است: چیدمان، جهت نمادها، ترتیب ستون‌های جدول و حتی جهت انیمیشن‌ها بر همان اساس تعریف شده‌اند.

سبک‌دهی به‌صورت شیء درون‌خطی روی هر مؤلفه انجام می‌شود. این انتخاب خطوط کد را طولانی‌تر می‌کند، ولی سامانهٔ طراحی را از وابستگی به پیکربندی پالایش یک ابزار بیرونی جدا نگه می‌دارد و تضمین می‌کند سبکی که در کد دیده می‌شود همان است که در مرورگر اجرا می‌شود.



شکل ۳-۴ — صفحهٔ جست‌وجو و پالایش. پالایه‌های شهر، نوع، قیمت و ابعاد بالای شبکهٔ نتایج؛ هر کارت قیمت ماهانه و برآورد بازدید روزانه را نشان می‌دهد.



شکل ۴-۴ — صفحهٔ جزئیات رسانه: گالری تصویر، مشخصات فنی، موقعیت روی نقشه، و پنل «آنالیز ترافیک» که برآورد بازدید روزانه و مبنای محاسبه‌اش را می‌آورد. شمارهٔ تماس در این صفحه نیست؛ جایش دکمهٔ «برای دیدن اطلاعات تماس وارد شوید» است.

سه اصل در طراحی حالت‌های رابط رعایت شده است: هر فهرست حالت «خالی» طراحی‌شده دارد و صفحهٔ سفید نشان نمی‌دهد؛ هر دکمهٔ ارسال هنگام پرواز درخواست غیرفعال می‌شود تا ارسال دوباره رخ ندهد؛ و هر خطا پیام فارسی روشن دارد، نه کد وضعیت خام.

۴-۵ سازمان‌دهی کد و نقشهٔ وابستگی‌ها

پروژه ۱۳۲ فایل TypeScript دارد که با ۴۰۱ پیوند `import` به هم وصل‌اند. این گراف با پیمایش خودکار مخزن استخراج شده است، پس آنچه در ادامه می‌آید توصیف معماریِ آرزوشده نیست؛ اندازه‌گیری معماریِ موجود است.

جدول ۴-۲ — لایه‌های کد، تعداد فایل و نقش هر لایه

لایه	فایل	نقش
نگهبان لبه	۱	`proxy.ts` — پیش از هر درخواست اجرا می‌شود
صفحه‌ها	۳۰	آنچه کاربر در مرورگر می‌بیند
مؤلفه‌های رابط	۳۲	قطعه‌های قابل استفادهٔ مجدد رابط کاربری
مسیرهای برنامه‌نویسی	۳۴	نقطه‌های ورود HTTP
احراز هویت	۶	نشست، سقف نرخ، شناسهٔ کاربر، رد ممیزی
ابزارهای مشترک	۱۷	ثبت رخداد، خطا، اعتبارسنجی محیط، قالب‌بندی
لایهٔ داده	۲	تنها راه رسیدن به پایگاه داده
طرح‌واره و دادهٔ اولیه	۶	مهاجرت، پرکردن، حذف تکراری، ژئوکدینگ

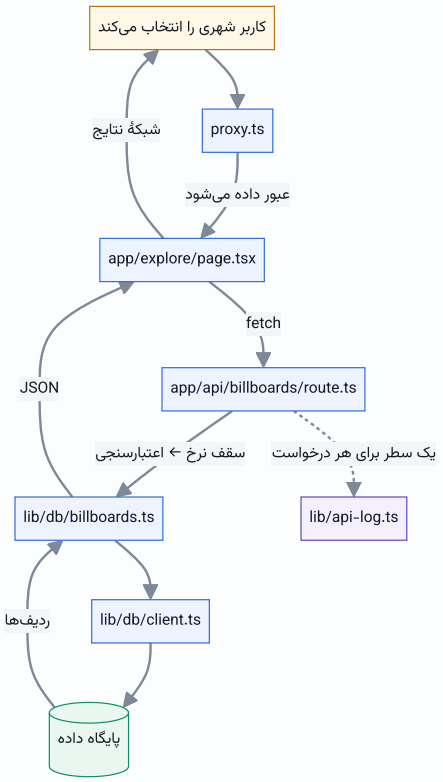
لایه	فایل	نقش
پیکربندی	۴	تنظیمات ساخت و اجرا

```
app/                          صفحه‌ها و مسیرهای برنامه‌نویسی — چارچوب از روی نام پوشه نشانی می‌سازد
├── page.tsx                  صفحهٔ نخست ‹/›
├── explore/page.tsx          جست‌وجو ‹explore/›
├── billboard/[slug]/page.tsx جزئیات — تنها مؤلفهٔ سمت سرور
├── admin/page.tsx            پنل مدیریت
└── api/                      ۳۴ فایل route.ts — هر کدام یک نقطهٔ ورود HTTP
    ├── billboards/route.ts   فهرست و پالایش
    ├── auth/                 ورود، ثبت‌نام، رمز یک‌بارمصرف
    └── admin/                مسیرهای مدیریتی
components/                   ۳۲ قطعهٔ رابط کاربری قابل استفادهٔ مجدد
lib/                          منطق مشترک — هیچ صفحه‌ای مستقیم به پایگاه داده دست نمی‌زند
├── db/client.ts              تنها جای ساخت اتصال
├── db/billboards.ts          لایهٔ دسترسی داده
├── auth/                     نشست، سقف نرخ، شناسهٔ کاربر، رد ممیزی
├── api-log.ts · api-error.ts ثبت رخداد و پاسخ خطای یکسان
└── types.ts                  انواع داده، بدون خودِ داده
prisma/schema.prisma          ۱۰ موجودیت، ۲۵ نمایه، ۱۳ مهاجرت
proxy.ts                      نگهبان لبه — پیش از هر درخواستی اجرا می‌شود
```

شکل ۵-۴ — ساختار پوشه‌های پروژه. نکتهٔ کلیدی این چیدمان آن است که `/lib/db` تنها راه رسیدن به پایگاه داده است؛ نه صفحه‌ای و نه مسیری، اتصال جداگانه‌ای نمی‌سازد.

۴-۵-۱ سفر یک درخواست، با نام فایل‌های واقعی

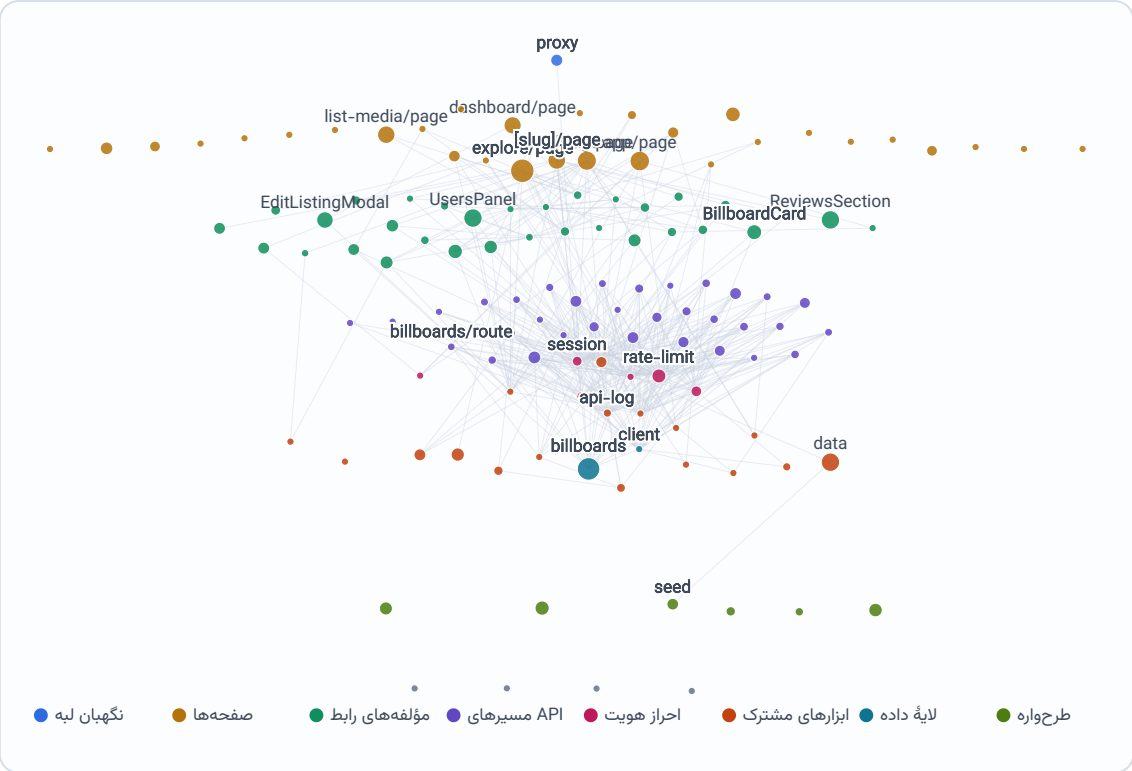
نمودار زیر همان مسیری است که وقتی کاربر در صفحهٔ جست‌وجو شهری را انتخاب می‌کند طی می‌شود. هر جعبه یک فایل واقعی در مخزن است.



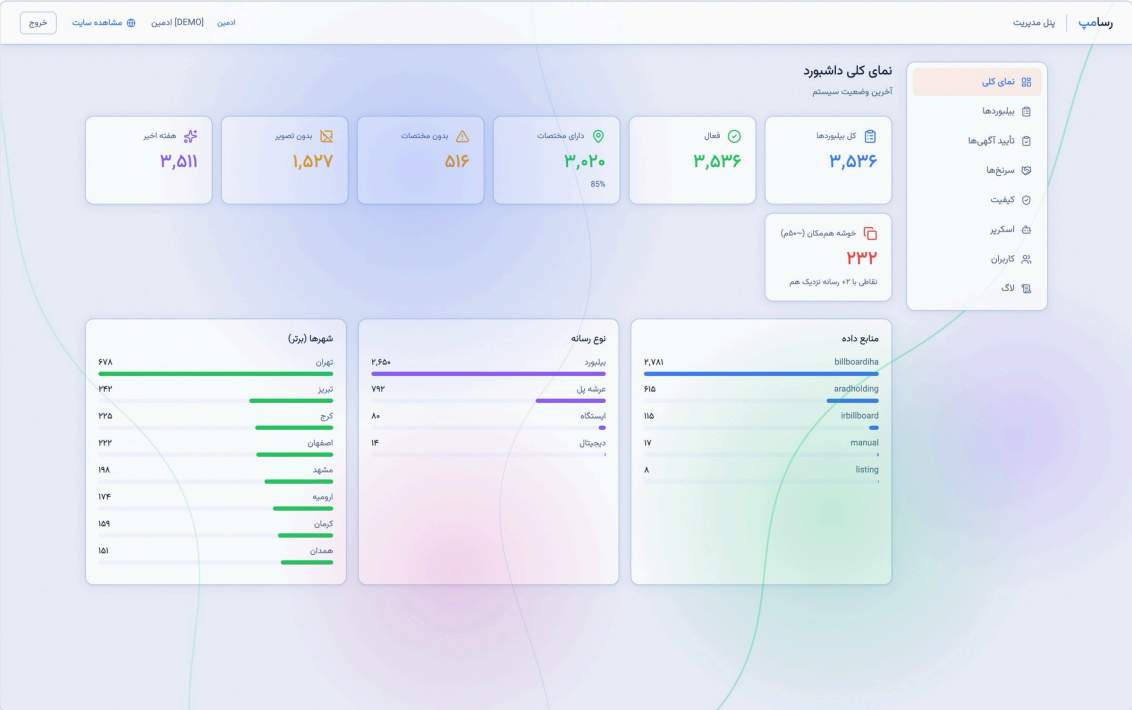
شکل ۶-۴ — سفر یک درخواست از کلیک کاربر تا پایگاه داده و برگشت. صفحهٔ جزئیات رسانه تنها استثناست: چون مؤلفهٔ سمت سرور است، مستقیم `lib/db/billboards.ts` را صدا می‌زند و دو جعبهٔ میانی را رد می‌کند.

۴-۵-۲ گراف کامل وابستگی

شکل زیر هر ۱۳۲ فایل و هر ۴۰۱ پیوند `import` را با هم نشان می‌دهد. رنگ‌ها همان لایه‌های جدول بالایند و اندازهٔ هر دایره به تعداد خطوط آن فایل است. چیدمان نیرومحور است: فایل‌هایی که همه به آن‌ها وابسته‌اند خودبه‌خود به مرکز کشیده می‌شوند.



شکل ۷-۴ — گراف وابستگی پروژه، استخراج‌شده با پیمایش خودکار مخزن. لایه‌ها از بالا به پایین: نگهبان لبه، صفحه‌ها، مؤلفه‌های رابط، مسیرهای برنامه‌نویسی، احراز هویت، ابزارهای مشترک، لایهٔ داده، طرح‌واره و پیکربندی.



نمای کلی پنل مدیریت

شکل ۸-۴ — نمای کلی پنل مدیریت. اعدادِ این صفحه مستقیم از پایگاه داده خوانده می‌شوند و همان‌هایی‌اند که در سراسر این نوشته نقل شده‌اند: ۳,۵۳۶ رسانه، ۳,۰۲۰ دارای مختصات (۸۵٪)، و سهم هر منبع.

۴-۵-۳ معماری در خودِ گراف دیده می‌شود

اگر فایل‌ها را بر حسب اینکه چند فایل دیگر به آن‌ها وابسته‌اند مرتب کنیم، نتیجه به‌تنهایی ادعای «پایپلاینِ ثابت» را تأیید می‌کند: هشت فایل نخست دقیقاً همان مرحله‌های آن پایپلاین‌اند.

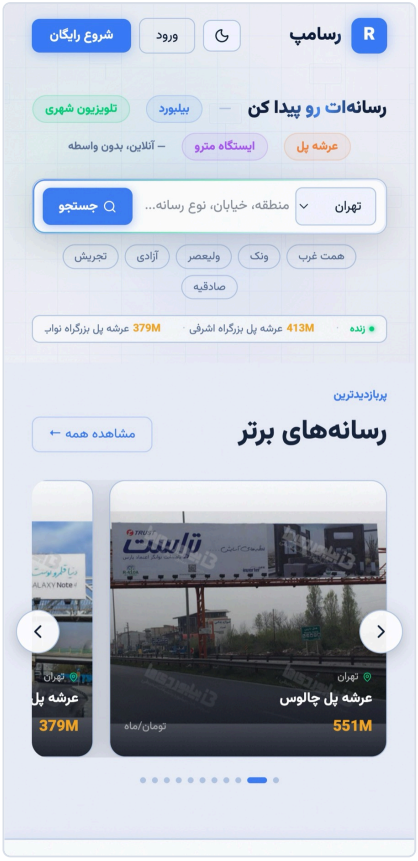
جدول ۳-۴ — پرارجاع‌ترین فایل‌های پروژه — چند فایل به هر یک وابسته‌اند

وابسته	فایل	نقش در پایپلاین
۳۴	`lib/api-log.ts`	ثبت هر درخواست
۳۳	`lib/auth/session.ts`	بررسی نشست
۳۲	`lib/auth/client-ip.ts`	شناسهٔ غیرقابل جعل کاربر
۳۲	`lib/auth/rate-limit.ts`	سقف نرخ درخواست
۳۱	`lib/api-rate-limit.ts`	پاسخ مشترک «تعداد زیاد»
۳۱	`lib/db/client.ts`	تنها اتصال به پایگاه داده
۲۶	`lib/types.ts`	انواع دادهٔ دامنه، بدون داده
۱۵	`lib/db/billboards.ts`	لایهٔ دسترسی دادهٔ رسانه‌ها

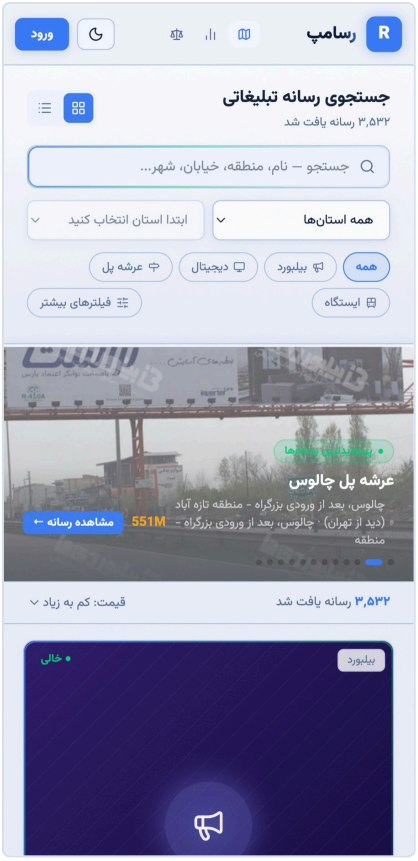
چرا این عددها مهم‌اند

اینکه ۳۱ فایل به `lib/db/client.ts` وابسته‌اند و هیچ *فایلی* اتصال جداگانه‌ای نمی‌سازد، یعنی ادعای «یک نقطهٔ ورود به پایگاه داده» قابل بررسی است و نه شعاری. به همین ترتیب، حضور `session.ts` و `rate-limit.ts` در صدر فهرست یعنی این بررسی‌ها در مسیرها فراموش نشده‌اند. اگر روزی مسیری بدون این وابستگی‌ها اضافه شود، از همین فهرست پیدا می‌شود.

یک نقشهٔ تعاملی از همین گراف (که در آن می‌توان روی هر فایل زد و دید چه چیزی را صدا می‌زند و چه کسی صدایش می‌زند) در مخزن پروژه به نشانی `docs/codemap.html` قرار دارد و از روی همان استخراج خودکار ساخته می‌شود.



صفحهٔ نخست



جست‌وجو

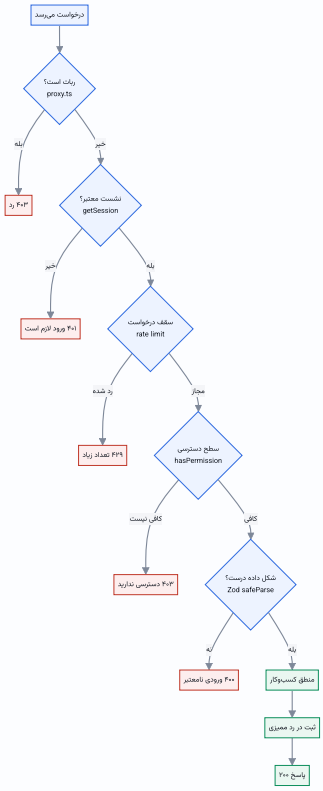
شکل ۹-۴ — همان رابط روی نمایشگر تلفن همراه. چیدمان، پالایه‌ها و کارت‌ها برای عرض کم بازآرایی می‌شوند، نه اینکه کوچک شوند.

فصل پنجم

پیاده‌سازی

۵-۱ پایپلاینِ ثابت هر درخواست

مهم‌ترین تصمیم پیاده‌سازی این پروژه، ثابت‌کردن ترتیب بررسی‌ها در هر مسیر برنامه‌نویسی است. هر مسیر مدیریتی دقیقاً همین ترتیب را دارد و جابه‌جایی آن مجاز نیست.



شکل ۵-۱ — پایپلاینِ ثابت هر درخواست مدیریتی. ترتیب این مراحل غیرقابل جابه‌جایی است.

```
async function POSTHandler(req, { params }) {
  const session = await getSession();                        // ۱ — نشست
  if (!session || session.role === "user")
    return NextResponse.json({ error: "احراز هویت لازم" }, { status: 401 });

  const ip = getClientIp(req);                               // ۲ — سقف نرخ
  const rl = adminApiRateLimit(ip);
  if (!rl.allowed) return rateLimited(rl, { … });

  if (!hasPermission(session.role, "admin"))             // ۳ — سطح دسترسی
    return NextResponse.json({ error: "فقط ادمین …" }, { status: 403 });

  const parsed = BodySchema.safeParse(body);                // ۴ — اعتبارسنجی
  if (!parsed.success)
    return NextResponse.json({ error: … }, { status: 400 });

  …                                                          // ۵ — منطق کسب‌وکار
}
```

قطعه کد ۱-۵ — پایپلاین در عمل، از مسیر تصمیم کارشناس. ترتیب پنج مرحله در همهٔ مسیرهای مدیریتی یکسان است.

دلیل ثابت‌بودن ترتیب، امنیتی است و نه سلیقه‌ای:

- اگر **سقف درخواست** پیش از **بررسی نشست** بیاید، مهاجم بدون داشتن حساب می‌تواند سهمیهٔ درخواست را تمام کند و کاربر واقعی را از سرویس محروم سازد.
- اگر **اعتبارسنجی** پیش از **سقف درخواست** بیاید، مهاجم می‌تواند بارِ سنگین بفرستد و سرور را مجبور به تجزیهٔ رایگان آن کند.

این ترتیب در سند قواعد پروژه به‌عنوان یک قاعدهٔ غیرقابل نقض ثبت شده تا در توسعهٔ بعدی به‌اشتباه جابه‌جا نشود.

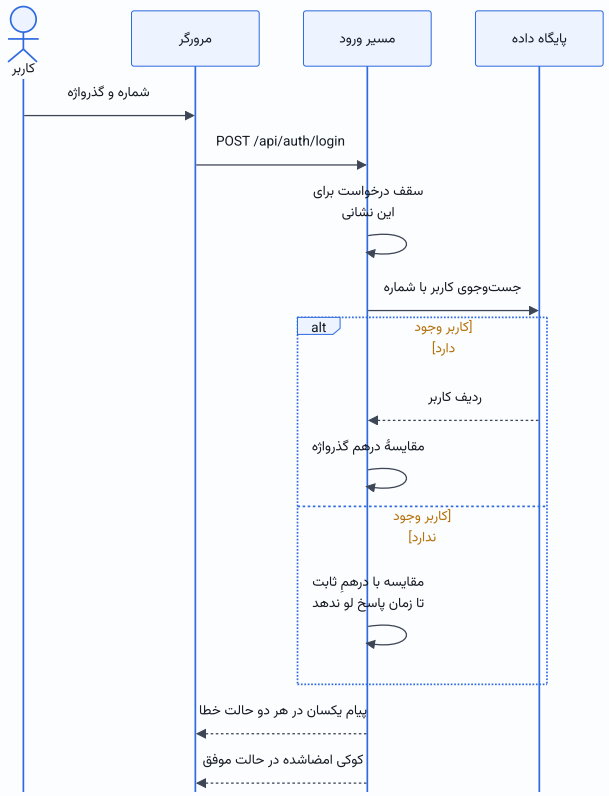
۵-۲ احراز هویت و کنترل دسترسی

نشست کاربر یک توکن امضاشده با الگوریتم HS256 است که در کوکی HttpOnly با صفت SameSite=Strict قرار می‌گیرد؛ چنین کوکی‌ای برای کد JavaScript صفحه قابل خواندن نیست، پس اسکریپتِ تزریق‌شده هم نمی‌تواند نشست را بدزدد. گذرواژه‌ها با bcrypt و ضریب هزینهٔ ۱۲ درهم می‌شوند.

نکتهٔ ظریف‌تر، رفتار سامانه در برابر شمارهٔ ناموجود است: پاسخِ بی‌درنگ در این حالت، با اختلاف زمان به مهاجم می‌گوید کدام شماره ثبت است. قطعه کد زیر این نشت را می‌بندد و رفتارش آزمون خودکار دارد.

```
export async function validateCredentials(email, password) {
  const user = await findUserByEmail(email);
  if (!user) {
    await bcrypt.compare(password, TIMING_PAD_HASH);   // همان هزینهٔ زمانی
    return null;
  }
  const ok = await verifyPassword(password, user.passwordHash);
  return ok ? user : null;
}
```

قطعه کد ۲-۵ — بستنِ نشتِ زمانی: در حالت «کاربر وجود ندارد» هم یک درهم‌سازی روی مقدار ثابت انجام می‌شود، تا مدت پاسخ نگوید کدام شماره در سامانه ثبت است.



شکل ۲-۵ — جریان ورود. پیام خطا در هر دو حالت «شماره ناموجود» و «گذرواژهٔ اشتباه» یکسان است تا شمارش کاربران ممکن نشود.

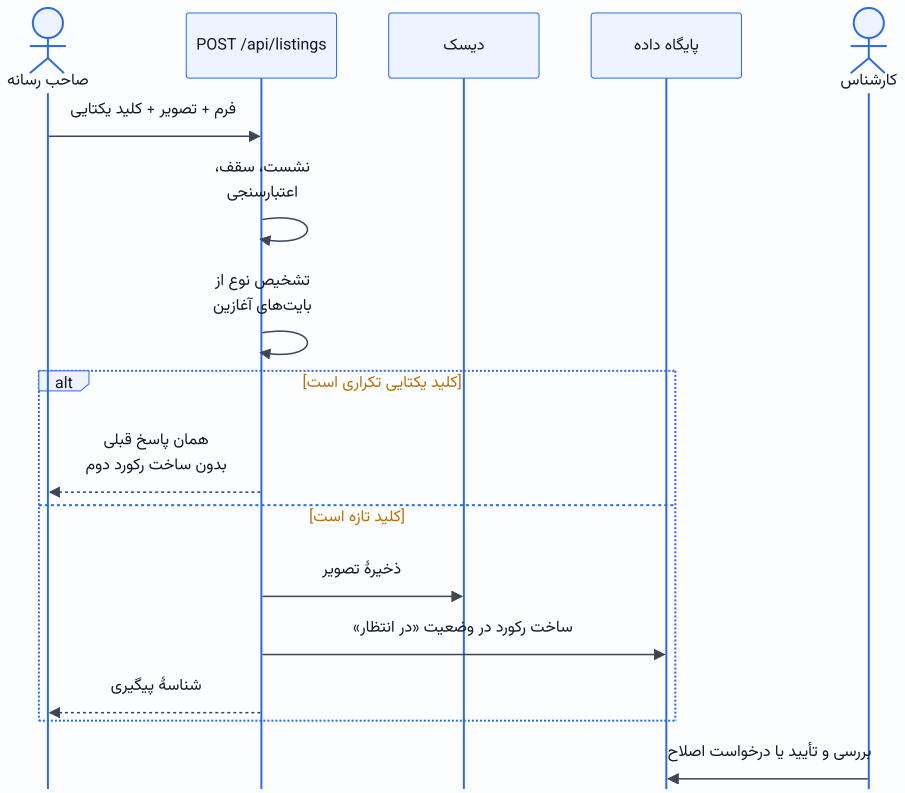
دسترسی مدیران چهار سطح پلکانی دارد و در هر مسیر مدیریتی، سطح لازم صریحاً بررسی می‌شود. نکتهٔ کلیدی آنکه این بررسی روی **نقطهٔ پایانی** انجام می‌شود، نه در رابط کاربری: پنهان‌کردن یک دکمه امنیت نیست، چون درخواست را می‌توان مستقیم ساخت.

۳-۵ ثبت آگهی و بارگذاری امن فایل

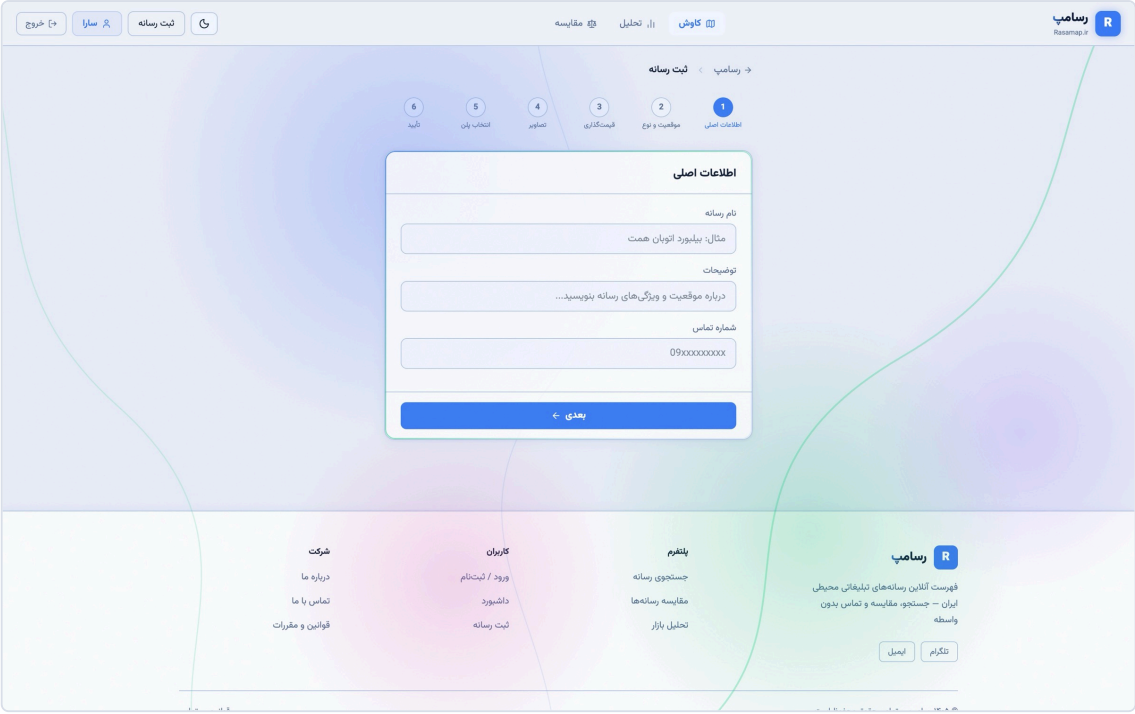
صاحب رسانه فرمی چندمرحله‌ای پر می‌کند و تصویر بارگذاری می‌کند. پذیرش فایل از عموم خطرناک‌ترین نقطهٔ ورودی هر سامانه است، پس نوع فایل از **بایت‌های آغازین خودِ فایل** تشخیص داده می‌شود، نه از نوع اعلام‌شدهٔ مرورگر. فایلی که ادعا می‌کند تصویر است ولی امضای بایتی تصویر ندارد، رد می‌شود.

```
function sniff(buf: Buffer): ImageKind | null {
  if (buf.length >= 3 &&
      buf[0] === 0xff && buf[1] === 0xd8 && buf[2] === 0xff) return "jpeg";
  if (buf.length >= 8 &&
      buf[0] === 0x89 && buf[1] === 0x50 && buf[2] === 0x4e && buf[3] === 0x47 &&
      buf[4] === 0x0d && buf[5] === 0x0a && buf[6] === 0x1a && buf[7] === 0x0a)
    return "png";
  if (buf.length >= 12 &&
      buf.toString("ascii", 0, 4) === "RIFF" &&
      buf.toString("ascii", 8, 12) === "WEBP") return "webp";
  return null;                    // ادعای مرورگر خوانده نمی‌شود
}
```

قطعه کد ۳-۵ — تشخیص نوع تصویر از بایت‌های آغازین خودِ فایل. نوع اعلام‌شدهٔ مرورگر اصلاً مبنا قرار نمی‌گیرد.



شکل ۳-۵ — ثبت آگهی. کلید یکتایی تضمین می‌کند ارسال دوبارهٔ فرم، آگهی دوم نسازد.

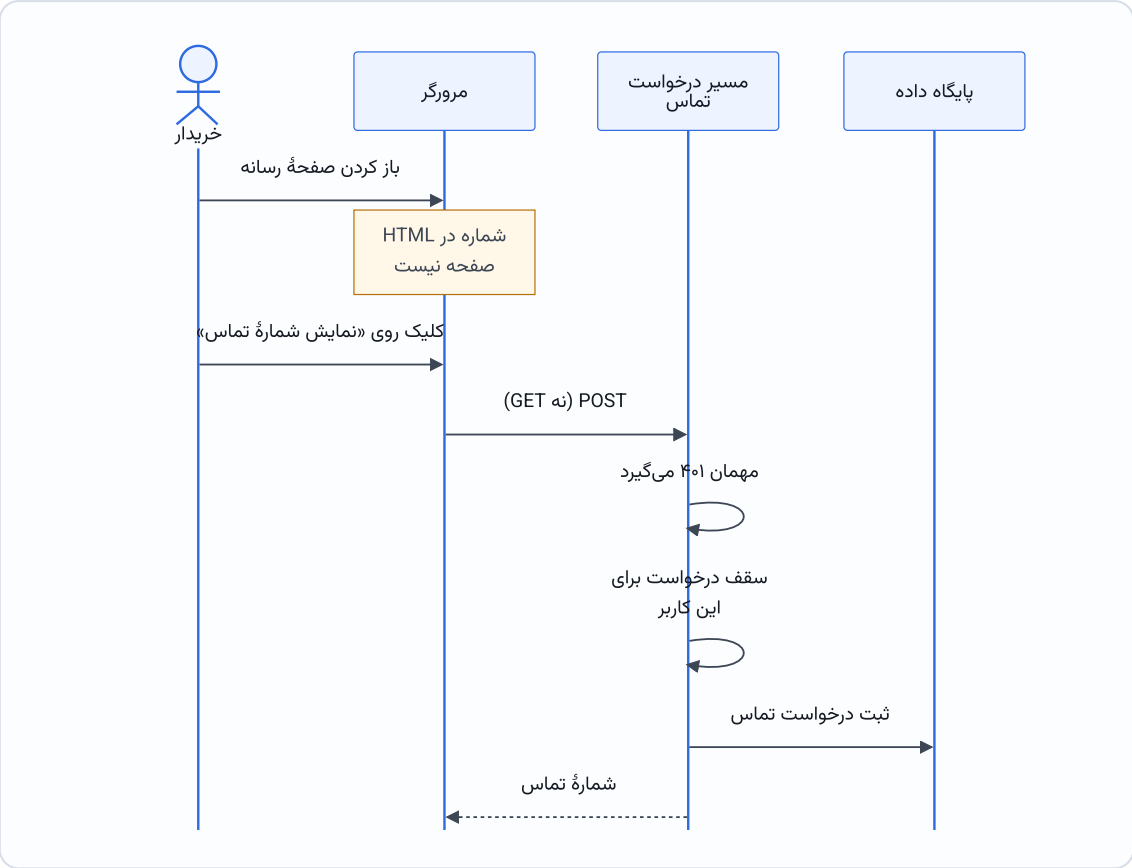


فرم چندمرحله‌ای ثبت آگهی

شکل ۴-۵ — فرم ثبت آگهی از دید صاحب رسانه. هر مرحله پیش از رفتن به بعدی اعتبارسنجی می‌شود و دکمهٔ ارسال هنگام پرواز درخواست غیرفعال می‌ماند.

۴-۵ محافظت از دادهٔ تماس مالکان

ارزشمندترین دادهٔ این سامانه فهرست شماره‌های تماس مالکان است و ساده‌ترین راه نابودی ارزش آن، قراردادنش در HTML صفحهٔ عمومی است. در این پیاده‌سازی شماره **هرگز** در صفحه قرار نمی‌گیرد (حتی برای کاربر واردشده) و تنها در پاسخ یک درخواست جداگانه ارسال می‌شود.

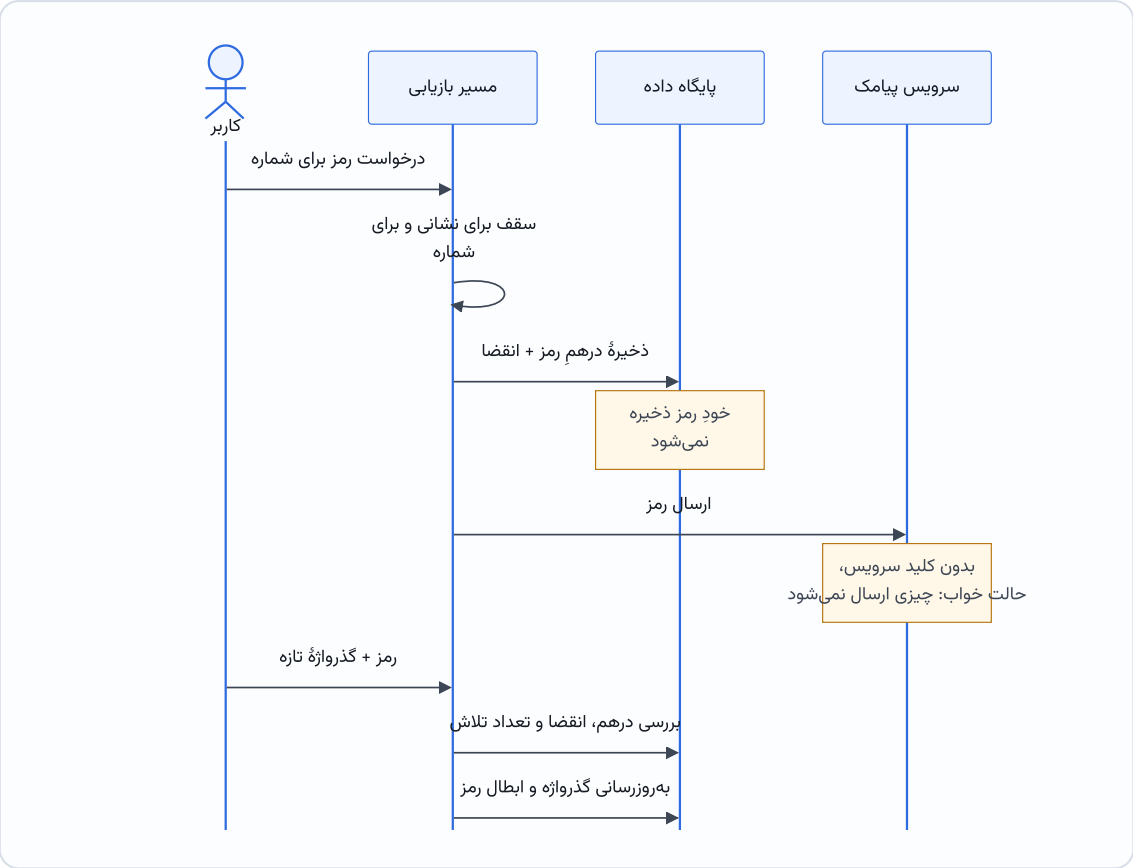


شکل ۵-۵ — دریافت شمارهٔ تماس. سه لایه (نبودِ شماره در صفحه، الزام ورود، و سقف درخواست) برداشت انبوه را پرهزینه می‌کنند.

> **ادعای صادقانه**
>
> این سازوکار برداشت انبوه را *ناممکن* نمی‌کند؛ آن را *پرهزینه* می‌کند. مهاجمی که حاضر باشد حساب بسازد، وارد شود و سقف درخواست را رعایت کند، در نهایت می‌تواند داده را جمع کند. هدف طراحی، بالا بردن هزینه تا جایی است که برداشت انبوه صرف نکند — و ادعای بیش از این، ادعای نادرستی است.

۵-۵ بازیابی گذرواژه با رمز یک‌بارمصرف

بازیابی گذرواژه در سه گام انجام می‌شود: درخواست رمز، تأیید رمز، و تعیین گذرواژهٔ تازه. خودِ رمز هیچ‌جا ذخیره نمی‌شود؛ تنها درهم آن به‌همراه زمان انقضا نگهداری می‌گردد.



شکل ۶-۵ — بازیابی گذرواژه. لایهٔ پیامک بدون کلید سرویس در حالت خواب می‌ماند؛ کد کامل است ولی پیامی ارسال نمی‌شود.

۶-۵ همروندی و صحت زیر بار هم‌زمان

سه نوع خطای همروندی در این سامانه به‌صورت ساختاری بسته شده‌اند.

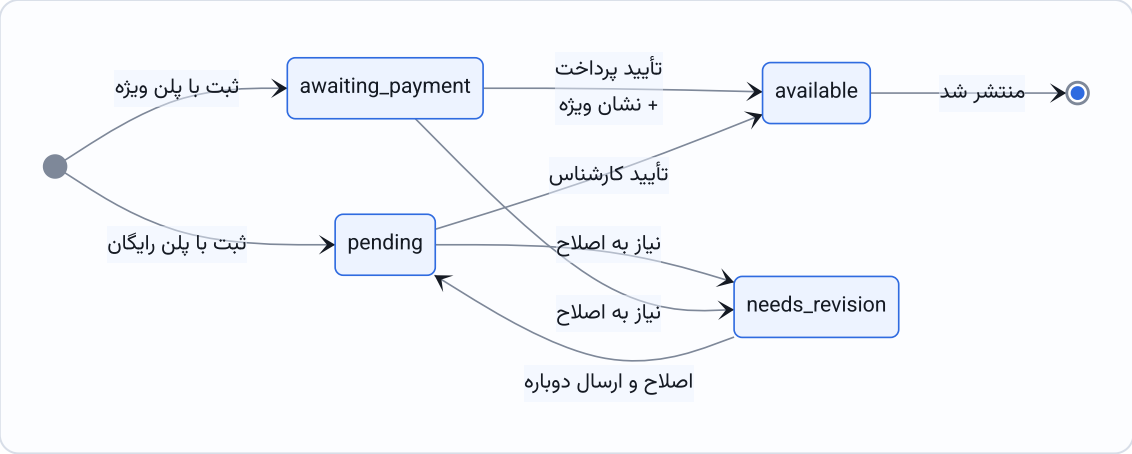
ارسال دوبارهٔ یک عملیات. کاربری که دکمهٔ ثبت را دو بار می‌زند یا اتصالش قطع می‌شود و دوباره می‌فرستد، نباید دو آگهی بسازد. هر ثبت آگهی یک کلید یکتایی همراه دارد؛ اگر همان کلید دوباره برسد، پاسخ ذخیره‌شده بازگردانده می‌شود و رکورد دومی ساخته نمی‌شود.

تصمیم دوباره روی یک آگهی. اگر دو کارشناس هم‌زمان یک آگهی را تأیید کنند، پلن ویژه نباید دو بار اعطا شود. تصمیم تأیید تک‌شلیک است: تلاش دوم پاسخ «تعارض» می‌گیرد.

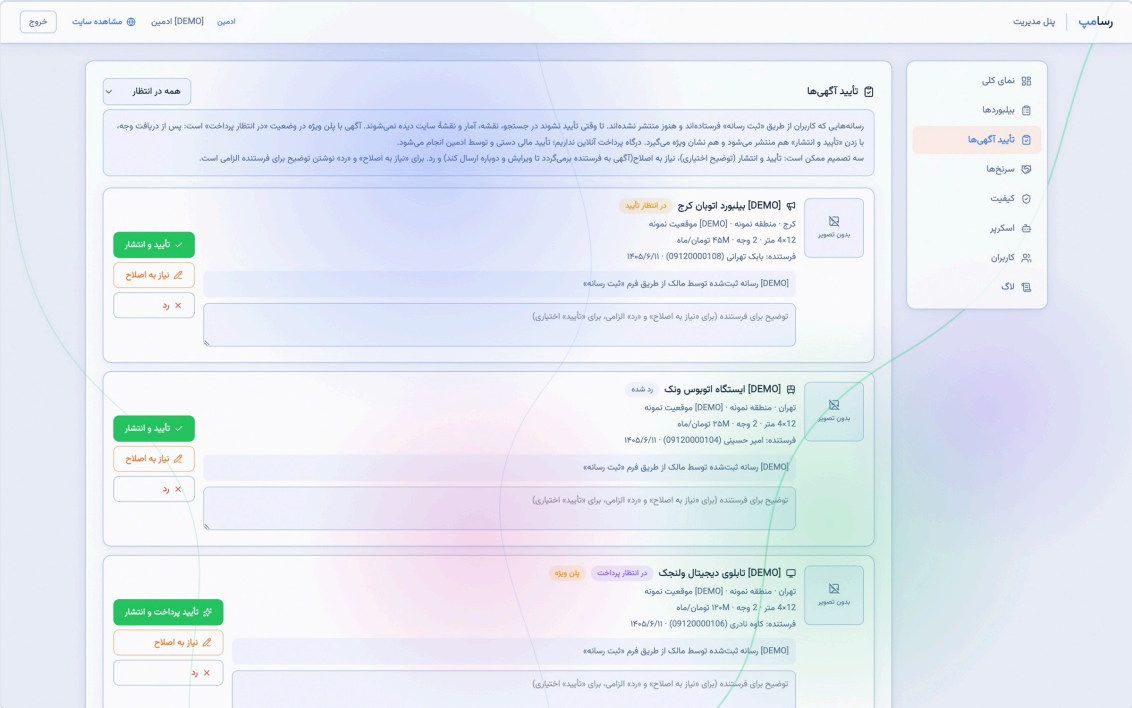
خواندن، تغییر، نوشتن. ثبت یک نظر، میانگین امتیاز رسانه را هم تغییر می‌دهد. اگر این دو جدا انجام شوند، دو نظر هم‌زمان می‌توانند میانگین را خراب کنند. هر دو در یک تراکنش انجام می‌شوند: یا هر دو یا هیچ‌کدام.

```
// فقط ردیف‌هایی که هنوز تصمیمی نگرفته‌اند پذیرفته می‌شوند، پس کلیک دوم
// نمی‌تواند آگهیِ منتشرشده را دوباره تأیید یا پلن ویژه را دوباره اعطا کند.
if (!UNPUBLISHED_STATUSES.includes(existing.status)) {
  return NextResponse.json(
    { error: "این آگهی قبلاً بررسی شده است" },
    { status: 409 },
  );
}
```

قطعه کد ۴-۵ — تصمیم تک‌شلیک. تلاش دوم برای همان گذار، پاسخ «تعارض» می‌گیرد.



شکل ۷-۵ — ماشین حالت آگهی. هر گذار تک‌شلیک است؛ تلاش دوم برای همان گذار پاسخ «تعارض» می‌گیرد.



صف بررسی آگهی‌ها در پنل مدیریت

شکل ۸-۵ — همان ماشین حالت در عمل: نشان‌های «در انتظار تأیید»، «رد شده» و «در انتظار پرداخت» کنار سه تصمیم مجاز کارشناس. توضیح برای فرستنده در حالت «نیاز به اصلاح» و «رد» الزامی است.

۵-۷ مدل تخمین بازدید روزانه

منابع دادهٔ موجود برای هر سازه یک عکس، یک اندازه و یک نام خیابان منتشر می‌کنند و نه بیشتر. اما عددی که خریدار واقعاً برای تصمیم‌گیری لازم دارد (چند نفر در روز این سازه را می‌بینند) در هیچ منبعی وجود ندارد. مدل زیر آن را از داده‌ای که در دسترس است برآورد می‌کند و از شکل مرسوم صنعت پیروی می‌کند: شمردن عبور روزانه، تبدیل آن به نفر، و تنزیل بر اساس اینکه چه سهمی از این جمعیت امکان *دیدن* این سازهٔ مشخص را دارد.

```
vehicles   =  roadClass_base  ×  cityFactor  ×  spread
opportunity  =  vehicles × occupancy  +  pedestrians
visibility  =  see_road × see_type × (1 + sizeBonus)
impressions  =  opportunity × visibility
```

۵-۷-۱ کلاس معبر، نه جمعیت شهر

پایهٔ مدل، **کلاس معبر** است و نه جمعیت شهر. این تمایز، اصلاح یک خطای واقعی در نسخهٔ نخست مدل بود: آن نسخه هر سازهٔ تهران را با درصدی از جمعیت نه‌میلیونی شهر آغاز می‌کرد و در نتیجه یک تابلو در یک کوچهٔ فرعی ادعای ۴۵۰٬۰۰۰ بیننده در روز داشت. یک بزرگراه در هر شهری ترافیک بزرگراه دارد؛ جمعیت شهر فقط آن را مقیاس می‌دهد، آن هم زیرخطی.

جدول ۵-۱ — کلاس‌های معبر و ضرایب هر یک. ستون «سهم دیدن» نسبت جمعیتی است که امکان واقعی خواندن پیام را دارد.

کلاس معبر	عبور روزانه	ضریب پیاده	سهم دیدن
بزرگراه و آزادراه	۱۲۰٬۰۰۰	۰٫۰۱	۳۴٪
بلوار و خیابان اصلی	۵۲٬۰۰۰	۰٫۰۹	۴۲٪
تقاطع و میدان	۳۸٬۰۰۰	۰٫۲۲	۵۵٪
خیابان معمولی	۱۷٬۰۰۰	۰٫۱۸	۴۶٪
ایستگاه حمل‌ونقل	۶٬۰۰۰	۲٫۴۰	۶۱٪
معبر محلی	۷٬۰۰۰	۰٫۱۴	۴۰٪

کلاس هر سازه با تطبیق واژگان فارسی نشانی آن تعیین می‌شود و تطبیق از بلندترین کلیدواژه آغاز می‌شود تا نام یک مسیر مشخص بر واژهٔ عام درون آن اولویت بگیرد. هر نشانی ناشناخته به «خیابان معمولی» نگاشت می‌شود — که انتخاب صادقانه‌ای است، چون بیشتر نشانی‌های موجود در داده همین‌اند.

۵-۷-۲ سه ملاحظهٔ روش‌شناختی

مقیاس شهر زیرخطی است. ضریب شهر برابر است با نسبت جمعیت به یک میلیون، به توان ۰٫۳۵. تهران بیست‌وسه برابر زنجان جمعیت دارد ولی خیابان‌هایش بیست‌وسه برابر خودرو حمل نمی‌کنند، حدود سه برابر حمل می‌کنند. توان کسری روش استاندارد بیان همین نکته است و پراکندگی را در کل فهرست از حدود ۷۰ برابر به حدود ۴ برابر می‌رساند.

اندازه بازده نزولی دارد. دو برابر کردن سطح یک سازه، تعداد خوانندگانش را دو برابر نمی‌کند؛ بنابراین مساحت با جذر وارد فرمول می‌شود و اثرش به حداکثر ۳۰٪ افزایش محدود است. نسبت نهایی دیده‌شدن نیز میان ۱۸٪ و ۸۲٪ مقید می‌ماند.

خروجی قطعی است. مدل هیچ عدد تصادفی‌ای ندارد. تنوع لازم (تا دو سازه در یک خیابان عدد یکسان نگیرند) از درهم‌سازی هویت خودِ رکورد به دست می‌آید که در اجراهای مختلف پایدار است. نسخهٔ نخست از یک ضریب تصادفی استفاده می‌کرد و نتیجه‌اش این بود که هیچ عددی در گزارش بازتولیدپذیر نبود.

مرز اعتبار مدل

این یک **برآورد** است، نه اندازه‌گیری، و سامانه هم همین را به کاربر می‌گوید. اعتبارش از دو چیز می‌آید: پیروی از شکلی که صنعت به کار می‌برد، و ثبت منبع هر ضریب در کد — جمعیت شهرها از سرشماری، ضریب ۱٫۵۵ نفر بر خودرو از ارقام منتشرشدهٔ شرکت کنترل ترافیک تهران برای سفرهای شهری، و لنگرهای قیمت از میانهٔ قیمت‌هایی که خودِ منابع منتشر کرده‌اند. آنچه این مدل ادعا نمی‌کند، جایگزینی یک سرشماری میدانی است.

۵-۸ مدیریت خطا و رصدپذیری

قاعدهٔ حاکم این است: کاربر هرگز نباید صفحهٔ سفید یا رد پشتهٔ خام ببیند، و توسعه‌دهنده هرگز نباید نتواند بفهمد چه چیزی خراب شده است. این دو خواسته با هم در تعارض‌اند و پاسخ، جدا کردن دو مخاطب است.

هنگام بروز خطای پیش‌بینی‌نشده، یک شناسهٔ کوتاه تولید می‌شود. کاربر پیام آرام فارسی به‌همراه همان شناسه می‌بیند؛ رد پشته، پرس‌وجو و جزئیات درونی فقط در ثبت رخداد سرور نوشته می‌شوند و کنار همان شناسه می‌نشینند. کاربر شناسه را گزارش می‌کند و توسعه‌دهنده دقیقاً همان رخداد را پیدا می‌کند، بی‌آنکه چیزی از درون سامانه بیرون درز کند.

هر رخداد به‌صورت یک شیء JSON در یک خط ثبت می‌شود. این قالب برای چشم انسان بهینه نیست، ولی برای پردازش ماشینی آماده است: اگر بعدها یک انباشتگر رخداد اضافه شود، هیچ تغییری در کد لازم نخواهد بود.

افزون بر این، هر تغییر مدیریتی (تأیید، رد، حذف، تغییر نقش) در یک رد ممیزی ماندگار با نشانی شبکه، شناسهٔ مدیر و جزئیات تغییر ثبت می‌شود.

ارزیابی و آزمون

۶-۱ روش و نتایج آزمون خودکار

ارزیابی بر سه پایه انجام شده است: آزمون خودکار در سطح واسط برنامه‌نویسی، اندازه‌گیری کارایی با ابزار سیستم‌عامل، و یک خودارزیابی صریح که نقاط ضعف را هم فهرست می‌کند. نکتهٔ روش‌شناختی مهم آنکه آزمون‌ها روی یک **ساخت تولیدی** اجرا می‌شوند و نه سرور حالت توسعه؛ دلیلش در بخش ۶-۴ می‌آید. هر اجرا پایگاه دادهٔ جداگانهٔ خود را از نو می‌سازد تا نتیجه به وضعیت پیشین وابسته نباشد.

آزمون	موفق	ناموفق	زمان اجرا
۱۱۳	۱۱۳	۰	۱۷٫۶ ث

آزمون‌ها بدون وابستگی بیرونی نوشته شده‌اند و مستقیماً با درخواست HTTP به یک سرور واقعی کار می‌کنند.

جدول ۶-۱ — سرفصل‌های پوشش آزمون خودکار

سرفصل	آنچه سنجیده می‌شود
اعتبارسنجی ورودی	رد ورودی بدشکل، بیش‌ازحد بزرگ، یا خارج از فهرست مجاز برای مرتب‌سازی و پالایش
کنترل دسترسی	دسترسی سطح‌به‌سطح مدیران و دسترسی در سطح شیء — کاربر به دادهٔ کاربر دیگر نرسد
سقف نرخ درخواست	فعال‌شدن سقف و بازگشت پاسخ درست پس از پایان بازه
عدم شمارش کاربران	یکسان بودن پاسخ برای شمارهٔ ناموجود و گذرواژهٔ اشتباه — هم در متن پاسخ و هم در **زمان پاسخ**
بارگذاری فایل	رد فایلی که ادعای تصویر دارد ولی امضای بایتی تصویر ندارد
ماشین حالت تأیید	مجاز بودن فقط گذارهای تعریف‌شده و تک‌شلیک بودن هر تصمیم
بازیابی گذرواژه	چرخهٔ کامل رمز یک‌بارمصرف، انقضا و محدودیت تعداد تلاش
درستی مرتب‌سازی	صحت ترتیب خروجی، با محافظ در برابر «موفقیت پوچ» روی فهرست خالی
رفتار وابسته به اتصال	نشانه‌گذاری درست کوکی امن پشت پروکسی رمزنگاری‌شده و عدم نشانه‌گذاری روی اتصال ساده

آزمون‌های محافظ

چند آزمون، رفتار کاربر را نمی‌سنجند بلکه **قواعد پروژه** را می‌سنجند: اینکه هر پاسخ «تعداد زیاد» از یک تابع مشترک عبور کند و هیچ انیمیشن بی‌پایانی بدون توقف در زبانهٔ پنهان نماند. این‌ها آزمون‌هایی‌اند که جلوی بازگشت خطاهای گذشته را می‌گیرند.

۶-۲ یک کلاس خطا و درسی که از آن گرفته شد

در جریان توسعه، شش خطای جداگانه شناسایی شد که همگی یک ریشه داشتند: **استنتاج ویژگی اتصال از محیطی که کد در آن ساخته شده است.** این خطاها روی رایانهٔ توسعه‌دهنده کاملاً نامرئی بودند و فقط وقتی ظاهر می‌شدند که سامانه از دستگاه دیگری روی شبکهٔ محلی باز شود — دقیقاً همان کاری که یک داور با تلفن همراهش انجام می‌دهد.

جدول ۶-۲ — شش خطای هم‌ریشه و اصلاح هر یک

خطا	اصلاح
صفت امن کوکی از متغیر محیط ساخت خوانده می‌شد	خوانده‌شدن از خودِ پروتکل درخواست
بررسی مبدأ با نام میزبانی که سرور به آن مقید است مقایسه می‌شد	مقایسه با سرآیند میزبانِ درخواست
نسخه‌برداری در حافظه بدون بستر امن کار نمی‌کرد	تابع مشترک با مسیر جایگزین
نشانی مطلق سایت به نشانی محلی بازمی‌گشت	یک منبع واحد برای نشانی سایت
بارگذاری تنبل روی کارت‌های چرخان	حذف از عناصری که کاربر نمی‌تواند به آن‌ها برسد
انیمیشن بی‌پایان در زبانهٔ پنهان ادامه می‌یافت	افزودن به فهرست توقف هنگام پنهان‌شدن صفحه

درس این تجربه فراتر از شش اصلاح است. پس از شناسایی الگو، یک **قاعدهٔ دائمی** در سند قواعد پروژه ثبت شد به‌همراه یک پرسش آزمون: «آیا این تغییر وقتی سایت از دستگاه دیگری روی شبکهٔ محلی باز شود هم کار می‌کند؟» دو مورد از این رفتارها اکنون آزمون خودکار دارند.

۶-۳ اندازه‌گیری کارایی

یک اندازه‌گیری در این پروژه چنان تفاوت بزرگی نشان داد که به یک قاعدهٔ عملیاتی تبدیل شد. پرسش ساده بود: هزینهٔ پردازندهٔ نخستین بازدید از ده مسیر سامانه، در حالت توسعه در برابر حالت ساخته‌شده، چقدر است؟

جدول ۶-۳ — هزینهٔ پردازنده برای نخستین بازدید از ده مسیر

حالت اجرا	پردازندهٔ مصرفی	نسبت
حالت توسعه	۹٫۷ ثانیه	۹۷ برابر
حالت ساخته‌شده	۰٫۱ ثانیه	مبنا

تفسیر

حالت توسعه هر صفحه را در همان لحظه‌ای که کاربر رویش کلیک می‌کند از نو ترجمه می‌کند و هزاران فایل را هم زیر نظر دارد. حالت ساخته‌شده این ترجمه را یک بار و پیشاپیش انجام می‌دهد. عدد ۹۷ برابر یک جزئیات اجرایی نیست: روی یک رایانهٔ همراهِ بدون فن، دقیقاً همین تفاوتِ «داغ و پرسروصدا» با «خنک» است. برای ارائهٔ زنده، سامانه باید **پیش از** شروع جلسه ساخته و اجرا شود، نه در حین آن.

همین قاعده یک بار به‌صورت عینی خود را نشان داد. مجموعهٔ آزمون در ابتدا روی سرور حالت توسعه اجرا می‌شد؛ یکی از آزمون‌ها فهرست را ۱۲۰ بار پیاپی می‌خواند، سرور برای هر درخواست از نو ترجمه می‌کرد، یک فراخوانی از مهلت سرآیند گذشت و سرور گیر کرد. نتیجه، شکست حدود بیست آزمون پایانی بود که همگی کدِ سالم را متهم می‌کردند و اجرا بیش از بیست دقیقه طول می‌کشید و هرگز به پایان نمی‌رسید. پس از انتقال آزمون‌ها به ساخت تولیدی، همان مجموعه در ۱۷٫۶ ثانیه کامل و بدون خطا اجرا شد.

۶-۴ خودارزیابی

جدول زیر ارزیابی صریح نگارنده از محورهای مختلف کار است. ستون «نقطهٔ ضعف» عمداً پر شده است؛ سندی که فقط نقاط قوت را فهرست کند، سند ارزیابی نیست.

جدول ۶-۴ — خودارزیابی محورهای پروژه

محور	شواهد	نقطهٔ ضعف
معماری	یک لایهٔ داده، دو مسیر ورود چارچوب در جای درست، پایپلاینِ ثابت روی هر مسیر	بستهٔ ارسالی به مرورگر یک بار کل مجموعهٔ داده را حمل می‌کرد؛ خطا با اندازه‌گیری پیدا و ساختاری اصلاح شد
پایگاه داده	طرح‌وارهٔ هنجارشده، نمایه‌های ترکیبی منطبق بر پرس‌وجوهای واقعی، مهاجرت نسخه‌بندی‌شده	موتور تک‌فایلی است؛ آرایه‌ها به‌صورت JSON ذخیره می‌شوند که یک سازش مدل‌سازی است
امنیت	مرز روی نقطهٔ پایانی، چهار سطح دسترسی، کوکی امضاشده، سقف نرخ، رد ممیزی، پیام خطای عمومی	بدون دیوارهٔ آتش کاربردی یا آزمون نفوذ بیرونی
همروندی	کلید یکتایی روی ثبت آگهی، تصمیم تک‌شلیک تأیید، تراکنش روی بازمحاسبهٔ امتیاز	سقف نرخ در حافظهٔ همان نمونه است؛ برای چند نمونه به انباره‌ای مشترک نیاز دارد
آزمون	۱۱۳ آزمون در سطح واسط برنامه‌نویسی، روی ساخت تولیدی و پایگاه دادهٔ ایزوله	آزمون مؤلفه‌ای و آزمون سرتاسری مرورگر وجود ندارد
مدیریت خطا	صفحه‌های خطای فارسی، شناسهٔ پیگیری، حالت‌های خالی و درحال‌بارگذاری طراحی‌شده	چند جریان هنوز نخستین نشانهٔ خطا را به‌صورت کد وضعیت خام نشان می‌دهند
داده	۳٬۵۳۶ رکورد از سه منبع، حذف تکراری، ۸۵٪ دارای مختصات	گردآوری نقطه‌ای است، نه خوراک زنده؛ خزنده جزو سامانهٔ در حال اجرا نیست

نتیجه‌گیری و کار آینده

۷-۱ دستاوردها

هدف این پایان‌نامه ساختن یک نمای واحد و قابل جست‌وجو از عرضهٔ رسانهٔ تبلیغاتی محیطی بود. سامانه‌ای ساخته شده که ۳٬۵۳۶ رسانه از سه منبع را در یک فهرست یکپارچه گرد آورده، ۸۵٪ آن‌ها را روی نقشه نشانده، و برای همه‌شان عددی برای بازدید روزانه برآورد کرده که در هیچ منبع اصلی وجود نداشت.

- **یکپارچه‌سازی داده** — گردآوری از سه منبع، پاک‌سازی، حذف تکراری با تطبیق نام و موقعیت، و ژئوکد کردن ۳٬۰۲۰ رکورد.
- **مدل تخمین بازدید** — قطعی، بازتولیدپذیر، با منبع ثبت‌شده برای هر ضریب؛ نقطهٔ تمایز اصلی سامانه.
- **کیفیت مهندسی سمت سرور** — پایپلاینِ ثابت روی ۴۲ نقطهٔ پایانی، اعتبارسنجی طرح‌واره‌ای روی تمام ورودی‌ها، سه سازوکار مستقل برای صحت زیر بار هم‌زمان، و رد ممیزی ماندگار برای هر تغییر مدیریتی.
- **صحت سنجیده‌شده** — ۱۱۳ آزمون خودکار روی ساخت تولیدی، همگی موفق.

۷-۲ محدودیت‌ها

فهرست زیر صادقانه است و برای پیش‌گیری از برداشت اغراق‌آمیز از نتایج آورده می‌شود:

1. **داده زنده نیست.** مجموعهٔ فعلی یک گردآوری نقطه‌ای است. خزنده در مخزن وجود دارد ولی به‌صورت زمان‌بندی‌شده اجرا نمی‌شود، پس تغییر قیمت یا حذف یک سازه در منبع، خودکار بازتاب نمی‌یابد.
2. **تخمین بازدید، اندازه‌گیری نیست.** مدل از شکل صنعتی پیروی می‌کند ولی با شمارش میدانی اعتبارسنجی نشده است.
3. **یک نمونهٔ اجرایی.** موتور پایگاه داده و سقف نرخ درخواست هر دو تک‌نمونه‌ای‌اند. مسیر مهاجرت روشن است ولی پیموده نشده.
4. **پوشش آزمون در سطح واسط برنامه‌نویسی است.** رفتار مرورگر و مؤلفه‌های رابط کاربری دستی آزموده شده‌اند، نه خودکار.
5. **پرداخت بیرون از سامانه است.** پلن ویژه با تأیید دستی کارشناس فعال می‌شود.
6. **سنجش کارایی زیر بار واقعی انجام نشده است.** اندازه‌گیری‌ها روی یک دستگاه و با بار مصنوعی بوده‌اند.

۷-۳ کار آینده

مسیر ادامهٔ کار، به ترتیب اولویت و با در نظر گرفتن اینکه هیچ‌کدام نیازمند بازطراحی معماری نیستند:

جدول ۷-۱ — کارهای آینده به ترتیب اولویت

#	کار	شرح
۱	اجرای زمان‌بندی‌شدهٔ خزنده	تبدیل گردآوری نقطه‌ای به خوراک دوره‌ای، با تشخیص تغییر و آرشیو نسخه‌های پیشین

#	کار	شرح
۲	اعتبارسنجی میدانی مدل بازدید	مقایسهٔ خروجی مدل با شمارش واقعی در چند نقطهٔ نمونه و کالیبره‌کردن ضرایب
۳	آزمون سرتاسری مرورگر	پوشش خودکار جریان‌های کاربری کلیدی در کنار آزمون‌های موجود
۴	مهاجرت به موتور کلاینت-سروری	تغییر مقصد اتصال و اجرای دوبارهٔ مهاجرت‌ها؛ بدون بازنویسی کد
۵	سقف نرخ اشتراکی	انتقال شمارنده‌ها به یک انبارهٔ مشترک برای استقرار چندنمونه‌ای
۶	اتصال به درگاه پرداخت	پس از احراز شرایط حقوقی؛ منطق کسب‌وکار از پیش آماده است
۷	انباشتگر رخداد و هشدار	قالب ثبت رخداد از ابتدا ماشین‌خوان است و تغییری در کد لازم ندارد

۷-۴ جمع‌بندی

آنچه این پروژه را از یک تمرین دانشگاهی متمایز می‌کند، نه فهرست قابلیت‌هایش که **استدلال پشت مرزهایش** است. تصمیم به نساختن رزرو آنلاین، انتخاب یک موتور پایگاه دادهٔ ساده به‌همراه مستندکردن مسیر خروج از آن، پذیرش صریح اینکه محافظت از دادهٔ تماس مطلق نیست، و اصلاح مدلی که پیش‌تر عددی غیرواقعی تولید می‌کرد، همگی تصمیم‌هایی‌اند که ثبت و توجیه شده‌اند. سامانه‌ای که می‌داند چه کاری را *انجام نمی‌دهد* و چرا، از سامانه‌ای که همه چیز را ناقص ادعا می‌کند قابل اتکاتر است.

فهرست مراجع

1. مستندات رسمی Next.js، نسخهٔ ۱۶ — راهنمای App Router، مؤلفه‌های سمت سرور و لایهٔ میانی. نسخهٔ همراه بستهٔ نصب‌شده در مخزن پروژه.
2. مستندات رسمی React، نسخهٔ ۱۹ — مؤلفه‌های سمت سرور و مرز کلاینت.
3. مستندات Prisma ORM، نسخهٔ ۷ — طرح‌واره، مهاجرت و وفق‌دهندهٔ راه‌انداز.
4. مستندات SQLite — حالت ثبت پیشین نوشتن و رفتار همروندی.
5. OWASP Top Ten — فهرست ده ریسک برتر امنیت برنامه‌های تحت وب؛ مبنای بازبینی امنیتی این پروژه.
6. مستندات Zod — اعتبارسنجی طرح‌واره‌محور در TypeScript.
7. مستندات Leaflet — کتابخانهٔ نقشهٔ تعاملی متن‌باز.
8. سرویس نقشه و ژئوکدینگ نشان — مستندات واسط برنامه‌نویسی.
9. Geopath — استاندارد سنجش مخاطب رسانهٔ محیطی در ایالات متحده؛ مبنای شکل مدل تخمین بازدید.
10. Route Research Ltd. — استاندارد سنجش مخاطب رسانهٔ محیطی در بریتانیا.
11. مرکز آمار ایران — نتایج سرشماری عمومی نفوس و مسکن؛ مبنای جمعیت شهرها در مدل.
12. شرکت کنترل ترافیک تهران — ارقام منتشرشدهٔ سرنشین به‌ازای خودرو در سفرهای شهری.
13. منابع دادهٔ رسانهٔ محیطی به‌کاررفته در گردآوری: billboardiha.com، aradholding.com، irbillboard.com.
14. مستندات درون‌مخزنی پروژه: سند معماری، مرجع واسط برنامه‌نویسی، و سند تصمیم‌های مهندسی — در پوشهٔ `docs/` مخزن.

راهنمای اجرای پروژه

کد منبع

تمام کد، مستندات فنی و تاریخچهٔ توسعهٔ این پروژه در مخزن زیر در دسترس است:

github.com/alireza-zareian/Rasamap-Repository

۸-۱ پیش‌نیازها و راه‌اندازی

تنها پیش‌نیاز، محیط اجرای Node.js نسخهٔ ۲۰ یا بالاتر است. هیچ سرویس بیرونی، پایگاه دادهٔ جداگانه یا کانتینری لازم نیست.

```
npm install
cp .env.example .env
# سپس مقادیر را پر کنید
npm run db:migrate && npm run db:seed
npm run demo
# ساخت + اجرا روی درگاه ۳۰۰۰
```

پس از اجرا، سامانه از هر دستگاه دیگری روی همان شبکهٔ محلی با نشانی `http://<نشانی‌محلی>:3000` در دسترس است؛ تمام رفتارهای وابسته به اتصال برای همین حالت آزموده شده‌اند (بخش ۳-۶). دستور `npm test` نیز ۱۱۳ آزمون را روی یک ساخت تولیدی و پایگاه دادهٔ ایزوله اجرا می‌کند.

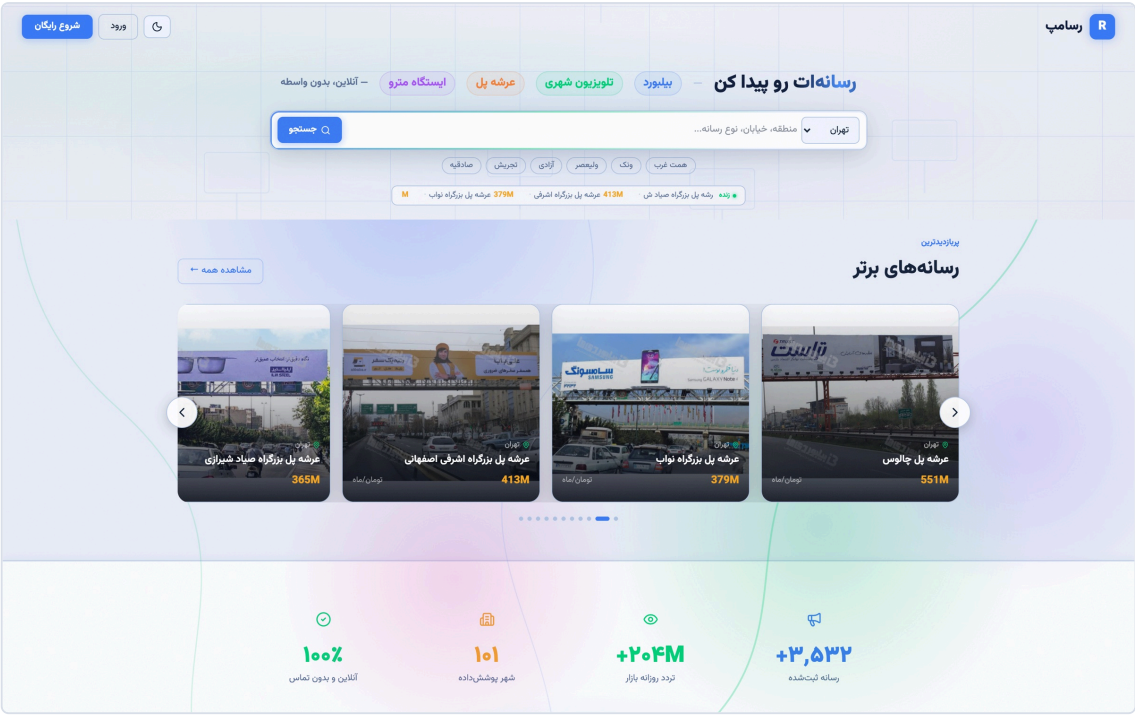
قاعدهٔ عملیاتی — برای دیدن یا ارائهٔ سامانه `npm run demo`

دستور `npm run dev` فقط هنگام *نوشتن کد* کاربرد دارد. برای نخستین بازدید از ده مسیر، حالت توسعه **۹٫۷ ثانیه** پردازنده مصرف می‌کند و حالت ساخته‌شده **۰٫۱ ثانیه**، یعنی حدود **۹۷ برابر** کمتر. توصیه می‌شود سامانه **پیش از** آغاز جلسه ساخته و اجرا شود، نه در حین آن. شرح در بخش ۴-۶.

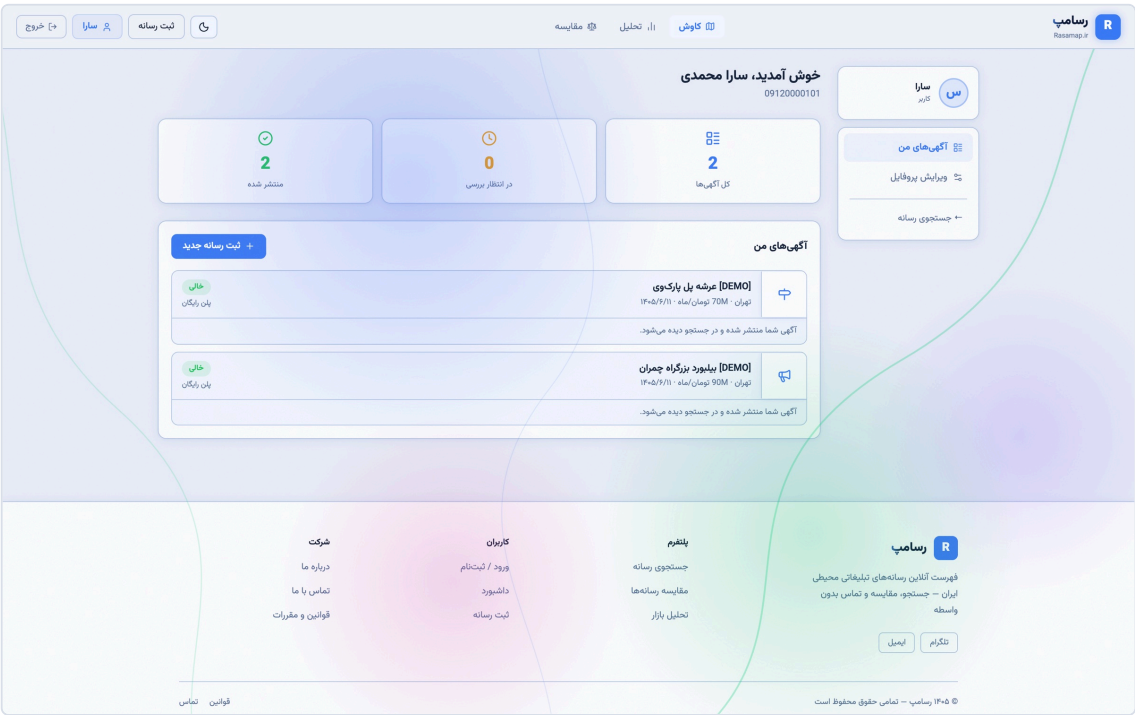
پیوست ب

نماهایی از سامانه

تصاویر زیر از نسخهٔ در حال اجرای سامانه گرفته شده‌اند و دادهٔ درونشان واقعی است. صفحه‌هایی که نیاز به ورود دارند با حساب‌های نمونه باز شده‌اند. صفحهٔ جست‌وجو، صفحهٔ جزئیات، فرم ثبت آگهی و صف بررسی در فصل‌های ۴ و ۵ آمده‌اند.

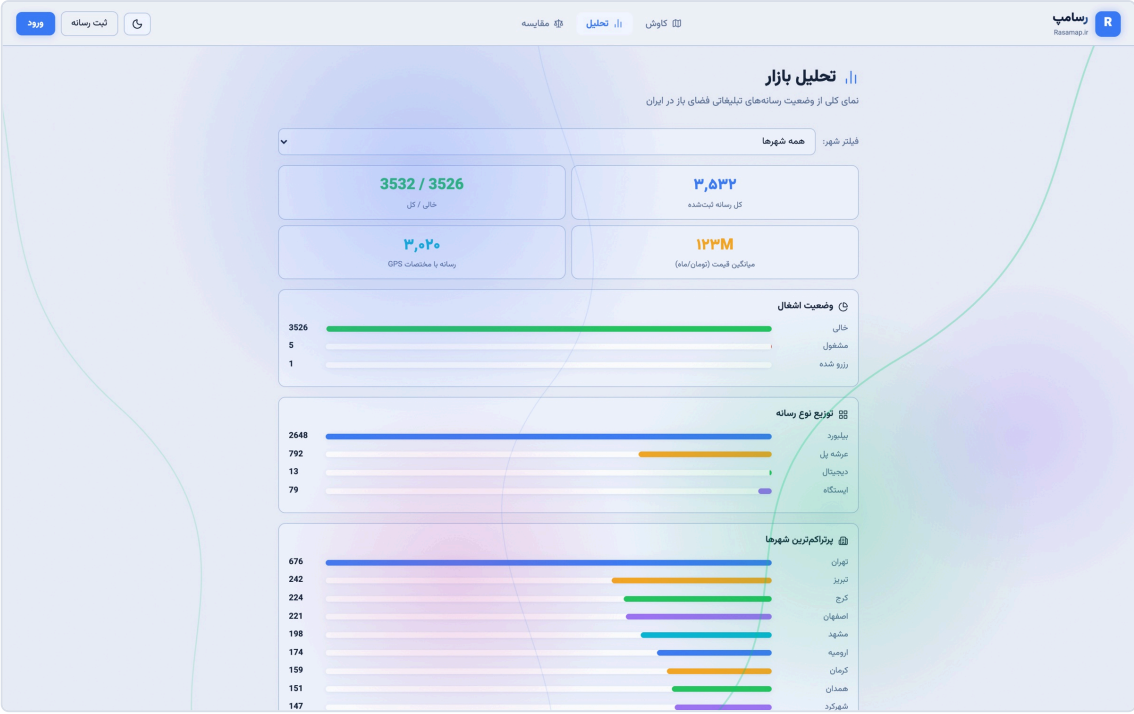


صفحهٔ نخست



پیشخوان کاربر

شکل ۱-۹ — سمت کاربر: صفحهٔ نخست با آمار کلی و رسانه‌های شاخص، و پیشخوانی که وضعیت هر آگهی ثبت‌شده را در چرخهٔ تأیید نشان می‌دهد.



صفحهٔ تحلیل

مرجع واسط برنامه‌نویسی

< `api.md` در انتهای همین فایل ادغام شده است.

مرجع نقاط پایانی

Rasamap — HTTP API reference

All routes are Next.js Route Handlers under `app/api/`. Inputs are validated with Zod `.safeParse()`. User-facing error messages are in Persian. This document is maintained by hand — update it when a route changes.

This file is also rendered in-app at `/api-docs` (self-hosted, no external CDN). Demo accounts for trying the endpoints: `RUNBOOK.md`.

Auth levels

Level	Meaning
public	no session needed
user	session cookie (`rasamap_session`) valid `role: "user"`
admin	session with role `viewer` / `editor` / `admin` / `super_admin` (enforced by `proxy.ts` and the route)
+editor	`super_admin` role `editor`, `admin` or
+admin	`super_admin` role `admin` or

Conventions

Session is a signed JWT (jose, HS256) in an HttpOnly `SameSite=Strict` cookie.
Admin route order is fixed: **session check → rate limit → Zod → business logic**.
Rate limits are per-IP, in-memory sliding window (`lib/auth/rate-limit.ts`).
Auth failures use generic messages (no user enumeration).

Idempotency-Key (optional header on `POST /api/listings`)

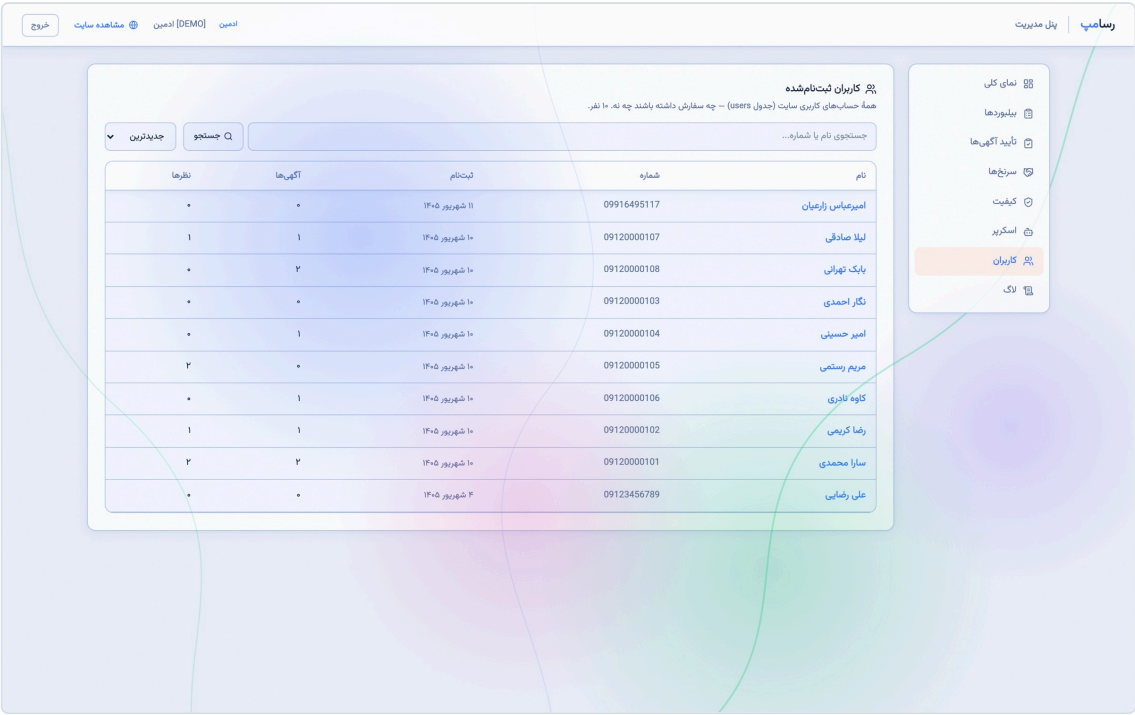
مرجع واسط برنامه‌نویسی

شکل ۲-۹ — صفحهٔ تحلیل توزیع شهر، نوع و قیمت را از دادهٔ واقعی رسم می‌کند. مرجع کامل ۴۲ نقطهٔ پایانی درون خودِ سامانه سرو می‌شود و نیازی به سند بیرونی ندارد.

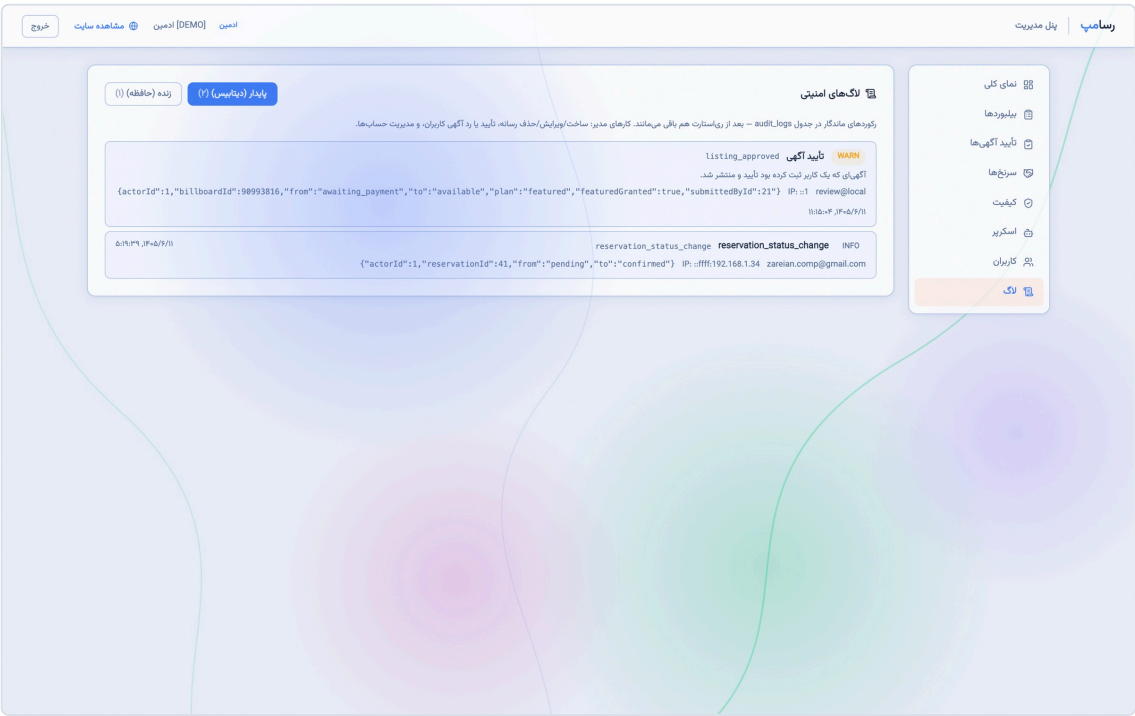
فهرست و ویرایش رسانه‌ها

وضعیت کیفیت داده

شکل ۳-۹ — پنل مدیریت: ویرایش رسانه‌ها، و صفحهٔ کیفیت که رکوردهای ناقص (بدون تصویر یا بدون مختصات) را برای پیگیری جدا می‌کند.



مدیریت مدیران و سطوح دسترسی



رد ممیزی

شکل ۴-۹ — چهار سطح دسترسی مدیران، و رد ممیزی که هر تغییر مدیریتی را با شناسهٔ مدیر، نشانی شبکه و جزئیات تغییر نگه می‌دارد.

Abstract

The market for out-of-home advertising media in Iran (billboards, footbridge panels, transit shelters and urban displays) is fragmented. Each structure's details sit with its own owner or contractor, so a buyer looking for a suitable placement must contact several intermediaries separately. The result is that no single searchable view of the available supply exists.

This thesis presents the design and implementation of a web system that gathers that supply into one searchable catalogue. It serves two user groups: advertising buyers, who search and filter across 3,536 media units in 101 cities, view each unit's details and map position, and (after authenticating) obtain the owner's contact number; and media owners, who submit their own listings through a multi-step form for expert review before publication.

The system is implemented as a single codebase on Next.js 16 and React 19, with a data layer over SQLite through Prisma 7 and authentication by a signed token in an HttpOnly cookie. The interface is entirely Persian and right-to-left. Seed data was collected from three public sources by a Python scraper and consolidated after duplicate removal.

The work's distinguishing contribution is a daily-audience estimation model. None of the surveyed sources publishes an exposure figure, although this is precisely the number a buyer needs. The model follows the shape used by the international out-of-home industry — count passing traffic, convert it to people, then discount by the share who could actually see that particular face — driven by road class rather than city population, and is fully deterministic and reproducible.

Engineering effort concentrated on server-side quality: a fixed, non-reorderable request pipeline on every route, schema validation on all input, rate limiting keyed to a non-spoofable client identity, idempotency guarantees on non-idempotent operations, and a durable audit trail for every administrative change. Correctness is verified by 113 automated API-level tests, all passing against a production build. A performance measurement showed that serving the project from a production build costs about 97 times less CPU than development mode for a first visit to ten routes.

Keywords: out-of-home advertising, web marketplace, Next.js, server-side rendering, API security, concurrency, SQLite

University of Zanjan

Faculty of Engineering

Thesis Submitted in Partial Fulfilment of the Requirements
for the Degree of Bachelor of Science (B.Sc.) in Computer Engineering

Design and Implementation of a Billboard Search and Listing System

By

Alireza Zareian

Supervisor

Dr. Sajjad Haghzad Klidbary

September 2026