**طراحی معماری «سیستم مدیریت خبر و خبرگزاری آنلاین»**

در این طرح، هم روی نیازهای دامنه‌ای تمرکز می‌کنیم و هم روی ویژگی‌های معماری (غیردامنه‌ای) مثل مقیاس‌پذیری، کارایی، امنیت و دسترس‌پذیری. خروجی به‌گونه‌ای است که تیم بتواند آن را به یک MVP تبدیل کند و سپس به‌صورت مرحله‌ای مقیاس دهد.

**1) اولویت‌بندی نیازها**

**دامنه‌ای (Domain) – بر اساس ارزش کاربر/پیچیدگی**

1. ایجاد/انتشار مقاله با دسته‌بندی، برچسب، نویسنده، زمان‌بندی انتشار (Scheduled Publish)
2. مشاهده فهرست و جزئیات خبر، چندرسانه‌ای (متن/عکس/ویدئو)
3. جستجوی پیشرفته بر اساس دسته‌بندی، مکان، تاریخ، کلیدواژه
4. تعامل کاربر: نظر، پسند (Like)/ذخیره‌سازی/اشتراک‌گذاری
5. مدیریت نقش‌ها: خبرنگار، مدیر، کاربر عادی
6. اخبار زنده (Live): پوشش رویداد با به‌روزرسانی لحظه‌ای
7. نظارت/Moderation بر نظرات و محتوا

**غیردامنه‌ای (Architectural Characteristics) – بر اساس ریسک/هزینه**

* **Availability**: هدف 99.9% برای MVP، قابل ارتقاء به 99.95%
* **Performance**: TTFB صفحه خبر ≤ 2 ثانیه برای p95، نتایج جستجو ≤ 1.5 ثانیه p95
* **Scalability**: افقی (stateless services + کش + صف)
* **Security**: OAuth2/OIDC، RBAC، محافظت در برابر XSS/CSRF/SQLi، Rate limiting
* **Searchability**: موتور جستجو اختصاصی (Elasticsearch/OpenSearch)، ایندکس آنی با نهایی‌شدن در پس‌زمینه
* **Observability**: لاگ ساخت‌یافته، metrics، tracing سرتاسری (OTel)

**2) نمای کلی معماری**

معماری پیشنهادی: **Microservices سبک** (یا «ماژولار مونو‌‌لیت» برای MVP و تفکیک تدریجی). اجزای اصلی:

* **API Gateway / BFF**: مسیریابی، احراز هویت، rate limiting، جمع‌بندی پاسخ‌ها برای وب/موبایل
* **Auth Service**: OIDC/OAuth2، مدیریت کاربران/نقش‌ها/نشست‌ها
* **Content Service**: CRUD خبر، Workflow انتشار، نسخه‌بندی، زمان‌بندی انتشار
* **Media Service**: آپلود/تبدیل/بهینه‌سازی تصاویر و ویدئو، ذخیره در Object Storage + تحویل از CDN
* **Search Service**: ایندکس و جستجو (Elastic/OpenSearch)، پیشنهاد کلیدواژه
* **Live Service**: انتشار رویدادهای زنده با WebSocket/SSE، کش کوتاه‌عمر
* **Interaction Service**: لایک/ذخیره‌سازی/اشتراک‌گذاری/بازدید، شمارنده‌ها (با eventual consistency)
* **Comment Service**: ثبت/واکشی نظر، Threading ساده، وضعیت (Pending/Approved/Rejected)
* **Moderation Service**: پالایش خودکار/دستی (Rule-based + مدل ML)، صف‌بندی
* **Admin Service**: داشبورد مدیر، گزارش‌ها، ابزار نظارت و بلاک/بن
* **Notification Service**: پوش/ایمیل/وب‌هوک‌ها (انتشار، پاسخ به نظر، خبر زنده)
* **Reporting/Analytics**: تجمع رخدادها (click/view/like) در انبار داده/Stream برای آمار

**زیرساخت مشترک**

* **Event Bus/Stream**: Kafka/NATS/RabbitMQ (رویدادهای ArticlePublished, CommentSubmitted, …)
* **Databases**: Relational (PostgreSQL/MySQL) برای تراکنش‌ها + Object Storage (S3-compatible) برای رسانه + Search Index
* **Cache/CDN**: Redis برای کش دیتای داغ؛ CDN برای رسانه و صفحات cacheable
* **Secrets & Config**: Vault/Parameter Store
* **CI/CD**: Build، تست، اسکن امنیتی، deployment آبی-سبز/کاناری
* **Observability**: Prometheus/Grafana + ELK + OpenTelemetry

**3) مدل داده (طرح پایگاه داده – سطح منطقی)**

**جداول هسته‌ای (Relational)**

* users(id, email, password\_hash, name, role[admin|reporter|user], status, created\_at)
* profiles(user\_id FK, avatar\_url, bio, location)
* articles(id, slug, title, summary, body, status[draft|scheduled|published|archived], author\_id FK, location, published\_at, created\_at, updated\_at, version)
* article\_assets(id, article\_id FK, type[text|image|video], url, metadata(jsonb))
* categories(id, name, slug, parent\_id NULLABLE) (درختی)
* article\_categories(article\_id FK, category\_id FK)
* tags(id, name, slug)
* article\_tags(article\_id FK, tag\_id FK)
* comments(id, article\_id FK, user\_id FK, parent\_id NULLABLE, body, status[pending|approved|rejected], created\_at, moderated\_by NULLABLE)
* interactions(id, article\_id FK, user\_id FK, type[like|bookmark|share], created\_at)
* live\_events(id, article\_id FK NULLABLE, title, payload(jsonb), created\_at) (برای فید زنده)
* audit\_logs(id, actor\_id, action, entity\_type, entity\_id, payload(jsonb), created\_at)

**ایندکس‌های کلیدی**

* articles(published\_at DESC, status)، articles(slug)، article\_categories(category\_id, article\_id)
* Full-text داخلی فقط برای fallback؛ جستجوی اصلی در **Elastic**.

**ایندکس جستجو (Search)**

* **Article Index**: id, title, summary, body, categories, tags, author, location, published\_at, popularity\_signals (views, likes)
* **Comment Index (اختیاری)**: برای جستجو در نظرات

**4) جریان‌های داده مهم**

**انتشار خبر (Publish)**

1. خبرنگار POST /articles → پیش‌نویس ایجاد می‌شود
2. افزودن رسانه‌ها → Media Service → Object Storage + ایجاد thumbnail
3. تعیین دسته/برچسب و زمان انتشار
4. تغییر وضعیت به scheduled|published
5. Event ArticlePublished → مصرف توسط Search (ایندکس)، Notification (خبر مشترکین)، Cache Invalidation (Gateway/Edge)

**نمایش خبر**

* Client → **BFF/Gateway**
* Hit کش صفحه/Article fragment در CDN/Redis؛ در صورت Miss → Content Service → DB
* شمارنده بازدید/تعاملات **غیرهمزمان** به Interaction Service ارسال می‌شود (event)

**نظرات و نظارت**

* کاربر POST /comments (status=pending) → Event به Moderation
* Rule-based (لیست سیاه عبارات) + مدل ML → خودکار approved/rejected
* موارد خاکستری به صف مدیر می‌روند؛ پس از تأیید، Cache/PubSub برای به‌روزرسانی UI

**اخبار زنده**

* خبرنگار یا ربات پوشش رویداد POST /live → Live Service رویداد را به کانال مربوطه publish می‌کند
* مصرف در کلاینت از طریق **WebSocket/SSE**؛ snapshot های لحظه‌ای در Redis برای Late-joiners
* در پایان رویداد، snapshot نهایی به‌عنوان مقاله خلاصه منتشر می‌شود

**5) API نمونه (خلاصه)**

**Auth**

* POST /auth/login، POST /auth/refresh، GET /auth/me
* OIDC/OAuth2 با JWT دسترسی + Refresh Token

**Articles**

* POST /articles (Reporter)
* PUT /articles/{id}، POST /articles/{id}/publish، POST /articles/{id}/schedule
* GET /articles?category=&tag=&q=&from=&to=&location=&page=
* GET /articles/{slug}

**Search**

* GET /search?q=&category=&location=&from=&to=&sort= (پروکسی به Search Service)

**Comments**

* POST /articles/{id}/comments
* GET /articles/{id}/comments?status=approved
* POST /comments/{id}/moderate (Admin/Moderator)

**Live**

* POST /live/channels/{key}/events (Reporter/Admin)
* GET /live/channels/{key}/stream (SSE) یا WS /live

**Interactions**

* POST /articles/{id}/like|bookmark|share
* GET /users/me/bookmarks

**6) جستجو و رتبه‌بندی**

* **ایندکس لحظه‌ای** پس از انتشار (رویدادمحور). برای کاهش تأخیر، ابتدا سند با حداقل فیلدها ایندکس می‌شود؛ enrichment (NER، استخراج مکان/نهاد) در پس‌زمینه.
* **Scoring**: ترکیب TF-IDF/BM25 با سیگنال‌های محبوبیت (views/likes) با decay زمانی.
* **فیلترها**: دسته، برچسب، مکان، بازه زمانی؛ **Highlighting** برای قطعه‌های نتیجه.
* **Auto-suggest/Did-you-mean** با لغت‌نامه و جستجوی فازی.

**7) دسترس‌پذیری، کارایی و کش**

* **CDN** برای رسانه و صفحات عمومی (HTML/JSON cacheable با ETag/Cache-Control)
* **Edge cache** برای صفحه خبر تا 30–120 ثانیه (Stale-While-Revalidate)
* **Redis** برای:
  + Fragment cache (header، box خبرهای مرتبط، شمارنده‌ها)
  + Live snapshot ها
  + Rate limiting (token bucket)
* **DB**: Replica برای خواندن (Read scaling)، Failover اتوماتیک
* **Backpressure**: صف‌ها برای درخواست‌های پرهزینه (ایندکس، نوتیفیکیشن، تبدیل ویدئو)
* **SLO**: p95 پاسخ API ≤ 300ms برای endpoints داده‌ای (بدون رسانه)

**8) امنیت**

* **OAuth2/OIDC + RBAC** (نقش: admin/reporter/user)، **Scopes** برای API
* **WAF** + **Rate limiting** در Gateway
* **Input Sanitization** (escape HTML در نظرات)، **CSRF** برای فرم‌ها، **CSP** برای فرانت‌اند
* **Storage**: رمزنگاری at-rest (KMS) و in-transit (TLS everywhere)
* **Least privilege** برای سرویس‌ها (Service-to-Service via mTLS/Service Mesh)
* **Auditing**: ثبت همه اقدامات حساس (انتشار/حذف/تغییر نقش)
* **Moderation** چندلایه برای UGC

**9) استقرار و توپولوژی**

* **Kubernetes**: هر سرویس یک Deployment + HPA بر اساس CPU/RPS/Latency
* **Stateful**: دیتابیس مدیریتی (Postgres/MySQL) با Operator، Elastic cluster با 3+ نود، Redis Sentinel/Cluster
* **CI/CD**: تست واحد/تراکمی، SAST/DAST، کاناری برای سرویس‌های حساس (Gateway, Content)
* **Observability**:
  + Metrics (RPS, latency, error rate) با آلارم‌های SLO
  + Logs ساختاریافته در ELK/ClickHouse
  + Tracing (B3/W3C) بین Gateway ←→ Services ←→ DB/Search

**10) طرح فرانت‌اند (خلاصه)**

* **BFF**: یک لایه تطبیقی برای وب و موبایل
* **SSR/ISR** (Next.js) برای SEO و TTFB خوب؛ صفحات خبر و فهرست‌ها با بازتولید زمان‌بندی‌شده
* **WebSocket/SSE** برای Live Ticker
* **AMP/Instant View** اختیاری برای کانال‌های توزیع

**11) طرح Moderation**

* Rule Engine: فهرست واژگان ممنوع، لینک‌های مشکوک، اسپم
* ML Service (بعد از MVP): طبقه‌بندی توهین/نفرت/هرزنامه (قابل تنظیم آستانه)
* Human-in-the-loop: صف بازبینی برای موارد بینابینی
* **Action**: حذف/پنهان، سایه‌بن (Shadowban)، محدودیت نرخ ارسال نظر

**12) Trade-offs کلیدی**

* **Microservices vs Modular Monolith**:
  + *MVP*: مونو‌لیت ماژولار (Content/Comment/Interaction ماژول‌های داخلی) → ساده‌تر، تح交 سریع‌تر
  + *اسکیل*: شکستن ماژول‌های پرترافیک (Search, Live, Media) به سرویس‌های مستقل
* **Elastic vs Full-Text DB**:
  + Elastic سریع‌تر و غنی‌تر؛ هزینه عملیات و نگهداری بالاتر. DB FTS برای fallback/small scale
* **Consistency**:
  + Strong در تراکنش‌های حیاتی (انتشار، ویرایش)
  + Eventual برای شمارنده‌ها/محبوبیت/Live (برای کارایی و مقیاس)
* **Live Transport (SSE vs WS)**:
  + SSE ساده و مناسب یک‌طرفه؛ WS برای تعاملی دوسویه (چت رویداد)

**13) برنامه تحویل (Roadmap)**

**فاز 1 – MVP (4–6 هفته)**

* Auth + RBAC ساده
* Content CRUD + انتشار + Media (تصویر)
* لیست/جزئیات خبر (SSR) + کش
* Comments با moderation دستی
* Search اولیه (Elastic تک شاخه)
* Observability پایه

**فاز 2 – Scale & Features**

* Live Service (SSE) + Snapshot
* Interaction Service + شمارنده‌ها، پیشنهاد محتوا
* Media ویدئو (Transcoding queue) + CDN پیشرفته
* Moderation ML، Notification، Analytics اولیه

**فاز 3 – سخت‌گیری غیرعملکردی**

* HA کامل برای DB/Search/Redis
* Service Mesh + mTLS، WAF پیشرفته
* Multi-region (اختیاری) و DR

**14) معیارهای پذیرش (نمونه)**

* جستجو بر اساس «فناوری + تهران + ۱۴۰۳/۰۶/۰۱ تا ۱۴۰۳/۰۶/۳۰» ≤ ۱.۵ ثانیه (p95)
* بارگذاری صفحه خبر عمومی ≤ ۲ ثانیه (p95) در 95٪ درخواست‌ها با 500 RPS و 90% cache hit
* انتشار خبر: از کلیک «Publish» تا قابل‌جستجو شدن ≤ 3 ثانیه (p95)
* Moderation: 99% بلاک واژگان ممنوع در < 1 ثانیه

**15) ریسک‌ها و راهکارها**

* **Hotspot روی یک خبر پربازدید** → Edge cache با SWR + fragment cache
* **Elastic cluster pressure** → شاردینگ بر اساس زمان/دسته + ILM (warm/cold)
* **حجم رسانه** → Object Storage ارزان + خط‌لوله فشرده‌سازی/thumbnail
* **اسپم نظرات** → CAPTCHA تطبیقی + rate limit per IP/user + shadowban

**16) دیاگرام‌های متنی کوتاه**

**جریان انتشار**  
Reporter → Gateway → ContentSvc(DB TX) → EventBus → SearchIndexer(Elastic) → Cache/Edge Invalidate → Users

**اخبار زنده**  
Reporter → LiveSvc → PubSub → WS/SSE → Clients (Redis snapshot for late joiners)

اگر دوست دارید، می‌تونم این طراحی را به یک **نمودار معماری قابل چاپ** یا یک **Schema SQL اولیه** و همین‌طور **نمونه فایل‌های OpenAPI** تبدیل کنم تا تیم سریع‌تر شروع کند. همچنین می‌تونم نسخه «مونو‌لیت ماژولار MVP» را با پوشه‌بندی و لایه‌بندی کد پیشنهاد بدهم.

**ADR 001 – سبک معماری (Modular Monolith → Evolutionary Microservices)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** نیاز به تحویل سریع MVP، پیچیدگی پایین در شروع، اما قابلیت جداسازی سرویس‌های پرترافیک (Search/Live/Media) در آینده.  
**تصمیم:** شروع با مونو‌لیت ماژولار (ماژول‌های Content, Comment, Interaction, Admin, Auth به‌صورت ماژول‌های داخلی) و تعریف مرزهای صریح دامنه؛ برنامه جداسازی تدریجی سرویس‌های پرترافیک پس از رسیدن به آستانه‌های بار.  
**گزینه‌های بررسی‌شده:** (الف) میکروسرویس کامل از ابتدا؛ (ب) مونو‌لیت سنتی؛ (ج) مونو‌لیت ماژولار.  
**پیامدها:** سادگی توسعه/دیباگ، هزینه عملیاتی کمتر در شروع، مسیر تکامل شفاف به سرویس‌ها. ریسک «ماژول‌های درهم‌تنیده» با اجرای Boundary و تست‌های قرارداد کاهش می‌یابد.

**ADR 002 – پایگاه‌داده اصلی (Relational + JSONB)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** تراکنش‌های قوی برای انتشار/ویرایش خبر، روابط غنی (دسته، برچسب، نویسنده)، نیاز به جستجوی ثانویه.  
**تصمیم:** استفاده از PostgreSQL به‌عنوان منبع حقیقت (OLTP)، با استفاده از JSONB برای فراداده‌ی رسانه و انعطاف.  
**گزینه‌ها:** PostgreSQL/MySQL/NoSQL سندی.  
**پیامدها:** ACID برای عملیات حیاتی، اکوسیستم بالغ، امکان مقیاس‌پذیری خوانش با Replica. نیاز به مدیریت ایندکس‌ها و مهاجرت‌ها.

**ADR 003 – موتور جستجو (OpenSearch/Elasticsearch)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** جستجوی متنی، فیلتر بر اساس زمان/دسته/مکان، Highlight، Suggest، رتبه‌بندی با سیگنال‌های محبوبیت.  
**تصمیم:** استقرار کلاستر OpenSearch (یا Elasticsearch) برای ایندکس اخبار؛ ایندکس near real-time پس از انتشار از طریق Event Bus.  
**گزینه‌ها:** FTS در PostgreSQL، Algolia، Elastic/OpenSearch.  
**پیامدها:** توانمندی‌های جستجوی غنی و مقیاس‌پذیر؛ هزینه عملیاتی بالاتر از FTS؛ نیاز به ILM برای مدیریت عمر داده‌ها.

**ADR 004 – گذرگاه رویداد/پیام (Event Bus)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** همگام‌سازی ایندکس، شمارنده تعاملات، اعلان‌ها، Moderation.  
**تصمیم:** Kafka به‌عنوان ستون فقرات استریم؛ NATS/RabbitMQ قابل‌قبول برای محیط‌های کوچک‌تر.  
**گزینه‌ها:** Kafka، RabbitMQ، NATS، Pub/Sub ابری.  
**پیامدها:** تفکیک سرویس‌ها، تحمل backpressure؛ نیاز به مانیتورینگ و طرح پارتیشن‌بندی.

**ADR 005 – تحویل رسانه و ذخیره‌سازی (Object Storage + CDN)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** تصاویر و ویدئو با حجم بالا، نیاز به بهینه‌سازی و کش لبه.  
**تصمیم:** ذخیره‌سازی رسانه در S3-compatible Object Storage؛ تحویل از طریق CDN؛ پردازش تصویر/ویدئو در Media Service (تبدیل/thumbnail).  
**گزینه‌ها:** فایل‌سیستم محلی، SAN/NAS، Object Storage ابری.  
**پیامدها:** مقیاس‌پذیری افقی، هزینه بهینه؛ نیاز به Pipeline پردازش و امضای URLها (Signed URLs).

**ADR 006 – راهبرد کش و تحویل محتوا (CDN + Redis + SWR)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** نیاز به TTFB < 2s، بار بالا در خبرهای ترند.  
**تصمیم:** کش لبه در CDN برای صفحات عمومی با Stale-While-Revalidate؛ Redis برای fragment cache (باکس‌های «پربازدید»/«مرتبط») و شمارنده‌ها؛ سیاست Invalidation رویدادمحور هنگام انتشار/ویرایش.  
**گزینه‌ها:** فقط کش اپلیکیشن، فقط CDN، ترکیبی.  
**پیامدها:** نرخ اصابت کش بالا و کاهش فشار به DB؛ پیچیدگی Invalidation.

**ADR 007 – مدل هم‌زمانی و سازگاری (Strong + Eventual)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** انتشار و ویرایش باید اتمیک باشد؛ شمارنده‌ها/محبوبیت و Live می‌تواند نهایتاً سازگار باشد.  
**تصمیم:** تراکنش‌های قوی برای «Content» و «Comments (پس از تأیید)»؛ Eventual Consistency برای Views/Likes/Bookmarks و فید Live.  
**گزینه‌ها:** Strong everywhere، Eventual everywhere، مدل ترکیبی.  
**پیامدها:** کارایی بهتر در مسیرهای خواندن پرترافیک، پذیرش تاخیر جزئی در همگام‌سازی سیگنال‌های محبوبیت.

**ADR 008 – احراز هویت و مجوز (OIDC/OAuth2 + RBAC)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** نقش‌های خبرنگار/مدیر/کاربر، نیاز به یکپارچگی با سرویس‌های خارجی.  
**تصمیم:** استفاده از OIDC/OAuth2 با توکن‌های JWT دسترسی و Refresh؛ RBAC با نقش‌های پایه و Scopeهای API.  
**گزینه‌ها:** سشن‌های سروری سنتی، JWT بدون OIDC، OIDC کامل.  
**پیامدها:** مقیاس‌پذیری stateless، سازگاری با SSO؛ نیاز به مدیریت چرخه عمر توکن و چرخش کلیدها (JWKS).

**ADR 009 – بستر فرانت‌اند و رندر (SSR/ISR)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** SEO برای خبرها، زمان بارگیری سریع، صفحات به‌روزرسانی‌شونده.  
**تصمیم:** استفاده از Next.js با SSR برای صفحات پویا و ISR/SSG برای فهرست‌ها/صفحات قابل کش؛ WebSocket/SSE برای Live.  
**گزینه‌ها:** SPA صرف، SSR سفارشی، Next.js/Remix.  
**پیامدها:** SEO مناسب، TTFB پایین، پیچیدگی DevOps اندک بیشتر.

**ADR 010 – حمل زنده (SSE در MVP، WebSocket برای تعاملی)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** پوشش زنده یک‌طرفه (Ticker)، سادگی استقرار.  
**تصمیم:** MVP با **SSE** برای پخش یک‌طرفه؛ ارتقاء به **WebSocket** برای تعاملات دوطرفه (چت رویداد/واکنش‌ها) در فاز بعد.  
**گزینه‌ها:** SSE، WebSocket، Long Polling.  
**پیامدها:** سادگی و سازگاری بالا در MVP؛ انعطاف ارتقاء برای نیازهای آتی.

**ADR 011 – Moderation نظرات (Rule-first + Human-in-the-loop)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** ریسک محتوای نامناسب، الزامات حقوقی.  
**تصمیم:** Rule Engine (لیست سیاه/الگوها/لینک‌ها) در مسیر همزمان؛ صف بازبینی برای موارد خاکستری؛ گزینه ML طبقه‌بندی بعد از MVP.  
**گزینه‌ها:** تنها انسان، تنها ML، ترکیبی.  
**پیامدها:** نرخ خطای پایین، SLA مناسب؛ هزینه عملیاتی صف انسانی.

**ADR 012 – درگاه API و امنیت مرزی (API Gateway)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** Rate limiting، احراز هویت متمرکز، مسیریابی به سرویس‌ها، observability.  
**تصمیم:** استقرار API Gateway با قابلیت WAF، Rate limiting، mTLS داخلی، OIDC integration، و تجمیع پاسخ‌ها برای BFF.  
**گزینه‌ها:** بدون Gateway، NGINX ساده، Gateway کامل (Kong/Envoy/APIM).  
**پیامدها:** کنترل متمرکز امنیت و ترافیک؛ پیچیدگی پیکربندی.

**ADR 013 – مشاهده‌پذیری (Logs/Metrics/Tracing با OTel)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** رفع اشکال در تولید، پایش SLOها، ردیابی درخواست‌های توزیع‌شده.  
**تصمیم:** OpenTelemetry برای Trace/Metric/Log؛ Prometheus + Grafana برای Metrics؛ ELK/Opensearch برای لاگ؛ Trace backend (Tempo/Jaeger).  
**گزینه‌ها:** Agentهای اختصاصی، Stack ابری مدیریت‌شده.  
**پیامدها:** استاندارد صنعتی، قابلیت vendor-neutral؛ نیاز به بودجه ذخیره‌سازی.

**ADR 014 – استقرار و مقیاس (Kubernetes + HPA)**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** نیاز به مقیاس افقی، به‌روزرسانی بدون‌وقفه، جداسازی منابع.  
**تصمیم:** استقرار روی Kubernetes با HPA بر اساس CPU/RPS/Latency؛ الگوی کاناری/آبی-سبز برای سرویس‌های حساس.  
**گزینه‌ها:** VMها، Serverless، K8s.  
**پیامدها:** انعطاف بالا، انزوا، اما پیچیدگی عملیاتی بیشتر.

**ADR 015 – راهبرد ایندکس و رتبه‌بندی جستجو**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** نیاز به تازه‌بودن نتایج و مرتبط‌بودن.  
**تصمیم:** ایندکس بلادرنگ (RT) حداقلی پس از انتشار؛ enrichment (NER، استخراج مکان/نهاد) غیرهمزمان؛ رتبه‌بندی BM25 + سیگنال‌های محبوبیت با decay زمانی.  
**گزینه‌ها:** ایندکس دوره‌ای Batch، فقط متن، شخصی‌سازی کامل.  
**پیامدها:** تازگی نتایج با هزینه پردازش پس‌زمینه؛ امکان ارتقاء به شخصی‌سازی در آینده.

**ADR 016 – محدودسازی نرخ و حفاظت از سوء‌استفاده**

**وضعیت:** پذیرفته‌شده — 2025-08-19  
**زمینه:** خطر بات‌ها و اسپم به‌خصوص در نظرات و جستجو.  
**تصمیم:** Rate limiting توزیع‌شده مبتنی بر Redis (Token Bucket/Leaky Bucket) در Gateway؛ CAPTCHA تطبیقی برای مسیرهای حساس؛ Shadowban برای حساب‌های خاطی.  
**گزینه‌ها:** بدون محدودسازی، محدودسازی فقط در اپ، WAF ابری.  
**پیامدها:** کاهش سوء‌استفاده با حداقل اثر بر کاربران واقعی؛ نیاز به تنظیم آستانه‌ها.

**ADR 017 – پایداری داده و DR**

**وضعیت:** پیشنهادی — 2025-08-19  
**زمینه:** هدف دسترس‌پذیری 99.9% در MVP، نیاز به بازیابی از فاجعه.  
**تصمیم:** پشتیبان‌گیری ساعتی/روزانه با آزمایش بازیابی؛ Replica خواندنی؛ برنامه ارتقا به چند-منطقه (Active/Passive) در فاز بعد.  
**گزینه‌ها:** تک‌منطقه بدون DR، چند-منطقه Active/Active.  
**پیامدها:** هزینه پایین‌تر در MVP؛ RTO/RPO متوسط، ارتقاپذیر در آینده.

**نکات نگهداری ADR**

* هر تغییر معماری باید با ADR جدید یا به‌روزرسانی وضعیت ثبت شود.
* شماره‌گذاری افزایشی است؛ «پیشنهادی» → «پذیرفته‌شده/رد شده».
* پوشه پیشنهادی: docs/adr/ADR-00X-<slug>.md.

**C4 – مدل معماری سیستم مدیریت خبر**

این سند شامل چهار سطح C4 (Context, Container, Component, Code/Optional) به‌همراه اسکریپت **Structurizr DSL** برای تولید نمودارهاست. می‌توانید همین DSL را در Structurizr/PlantUML تبدیل کنید.

**Level 1 – System Context (C1)**

**هدف:** نمایش بازیگران اصلی و سامانه‌های پیرامونی.

* **Persons**
  + کاربر عادی (Reader): مشاهده/جستجو/نظر/ذخیره
  + خبرنگار (Reporter): ایجاد/ویرایش/انتشار خبر، مدیریت رسانه
  + مدیر سیستم (Admin/Moderator): نظارت محتوا و کاربران
* **External Systems**
  + CDN/شبکه توزیع محتوا
  + سرویس هویت خارجی (IdP) اختیاری برای SSO
  + شبکه‌های اجتماعی (برای اشتراک‌گذاری)
  + سرویس ایمیل/Push (اعلان‌ها)

**روابط کلیدی**

* همه کاربران از طریق **وب/موبایل** به سیستم (  
  API Gateway/BFF  
  ) متصل می‌شوند.
* سیستم برای جستجو از **OpenSearch** استفاده می‌کند.
* رسانه‌ها از طریق **Object Storage + CDN** تحویل داده می‌شوند.

**Level 2 – Container (C2)**

**کانتینرها/زیربخش‌ها**

* **Web App (Next.js SSR/ISR)** – ارائه صفحات و تعامل کاربر
* **Mobile App** (اختیاری) – مصرف API های BFF
* **API Gateway/BFF** – احراز هویت، Rate Limit، آگریگیشن پاسخ
* **Auth Service** – OIDC/OAuth2، RBAC
* **Content Service** – CRUD مقاله، Workflow انتشار، زمان‌بندی
* **Media Service** – آپلود/تبدیل، امضای URL، مدیریت دارایی‌ها
* **Search Service** – ایندکس و جستجو (پروکسی به OpenSearch)
* **Live Service** – کانال‌های SSE/WS برای رویدادهای زنده
* **Comment Service** – مدیریت نظرات و threading
* **Moderation Service** – Rule Engine + صف بازبینی
* **Interaction Service** – Like/Bookmark/Share و شمارنده‌ها
* **Notification Service** – ایمیل/Push/Webhook
* **Admin Service** – پنل مدیریت و گزارش‌ها
* **Event Bus** – Kafka (Publish/Subscribe)
* **Databases** – PostgreSQL (OLTP)، Redis (Cache/Rate limit)، OpenSearch (Index)، Object Storage (S3)

**جریان‌های اصلی**

* انتشار خبر: Content → Event Bus → Search Indexer → Cache/CDN Invalidate → Notification
* نمایش خبر: Web/Mobile → BFF → (Redis/DB) → Media از طریق CDN
* نظر: Client → BFF → Comment (pending) → Moderation → Publish
* خبر زنده: Reporter → Live → Clients (SSE/WS) + Redis Snapshot

**Level 3 – Component (C3)**

**A) Content Service – Components**

* **ArticleController / GraphQL Resolver** – API
* **PublishingWorkflow** – تغییر وضعیت Draft/Scheduled/Published
* **Scheduler** – زمان‌بندی انتشار (cron/queue)
* **AssetLinker** – ارتباط مقاله با رسانه‌ها
* **CategoryTagManager** – دسته/برچسب
* **SearchIndexerProducer** – ارسال رویداد ArticlePublished
* **CacheInvalidator** – بی‌اعتبارسازی Edge/Redis
* **Repository Layer** (ORM) – دسترسی به PostgreSQL

**Data Stores:**

* **PostgreSQL**: tables articles, article\_assets, article\_tags, ...

**Interactions:**

* دریافت درخواست از BFF؛ تراکنش ذخیره؛ رویداد به Kafka؛ کش Invalid.

**B) Search Service – Components**

* **QueryParser** – پارس پارامترها (q, filters)
* **RelevanceScorer** – BM25 + سیگنال‌های محبوبیت با decay
* **Suggestion/Spellcheck** – پیشنهاد/تصحیح
* **IndexerConsumer** – مصرف ArticlePublished/Updated
* **OpenSearchAdapter** – ارتباط با کلاستر

**Data Stores:**

* **OpenSearch**: شاخص articles

**C) Comment & Moderation – Components**

* **CommentAPI** – ایجاد/واکشی
* **ModerationRuleEngine** – واژگان ممنوع/الگوها
* **MLClassifier (optional)** – برچسب‌زنی توهین/اسپم
* **ReviewQueue** – صف انسانی
* **StateTransitioner** – pending→approved/rejected
* **CommentRepository** – PostgreSQL (comments)

**D) Live Service – Components**

* **ChannelManager** – ایجاد/مدیریت کانال
* **EventPublisher** – انتشار رویداد به کانال‌ها
* **ClientGateway (SSE/WS)** – ارتباط پایدار با کلاینت‌ها
* **SnapshotStore** – Redis برای last state

**Level 4 – Code (Optional) برای PublishingWorkflow**

یک نمای نمونه از کلاس‌ها/ماژول‌ها برای جریان انتشار مقاله در Content Service.

* PublishingWorkflow
  + متد publish(articleId)
  + اعتبارسنجی وضعیت، بارگذاری وابستگی‌ها (assets/categories)
  + تراکنش: به‌روزرسانی وضعیت، published\_at
  + ارسال رویداد به Kafka: ArticlePublished {id, title, slug, categories, tags, published\_at}
  + بی‌اعتبارسازی کش: cache.invalidate(articleId, slug)
* ArticleRepository (ORM)
* EventBus (Kafka Producer)
* CacheClient (Redis)

**Structurizr DSL (قابل اجرا)**

کد زیر را می‌توانید در [Structurizr DSL](https://structurizr.com/dsl) وارد و نمودارها را تولید کنید.

workspace "News Management System" "C4 model for online news agency" {

!identifiers hierarchical

model {

user = person "Reader" "مشاهده/جستجو/نظر/ذخیره خبر" "external"

reporter = person "Reporter" "ایجاد و انتشار خبر" "external"

admin = person "Admin/Moderator" "نظارت بر محتوا و کاربران" "external"

system "News Platform" {

webapp = container "Web App (Next.js)" "SSR/ISR UI" "TypeScript/React"

mobile = container "Mobile App" "اختیاری" "Kotlin/Swift"

bff = container "API Gateway/BFF" "Auth, Rate limit, Aggregation" "Node/Go"

auth = container "Auth Service" "OIDC/OAuth2, RBAC" "Keycloak/Custom"

content = container "Content Service" "CRUD, Publishing, Scheduling" "Java/Kotlin/Go"

media = container "Media Service" "Upload/Transcode/Signed URLs" "Go/Node"

search = container "Search Service" "Index & Query" "Java/Go"

live = container "Live Service" "SSE/WS channels" "Node/Go"

comment = container "Comment Service" "Comments & Threads" "Node/Go"

mod = container "Moderation Service" "Rules/ML + review queue" "Python/Go"

interact = container "Interaction Service" "Likes/Bookmarks/Share counters" "Go/Node"

notify = container "Notification Service" "Email/Push/Webhook" "Node/Go"

adminsvc = container "Admin Service" "Backoffice & reports" "React/Node"

db = container "PostgreSQL" "OLTP relational DB" "PostgreSQL"

cache = container "Redis" "Caching/Rate limiting/Snapshots" "Redis"

index = container "OpenSearch" "Full-text search index" "OpenSearch"

obj = container "Object Storage (S3)" "Media files" "S3-compatible"

bus = container "Event Bus (Kafka)" "Async events" "Kafka"

cdn = container "CDN" "Edge caching & delivery" "CDN"

}

idp = softwareSystem "External IdP" "SSO provider" "external"

mail = softwareSystem "Email/Push Provider" "Notifications" "external"

social = softwareSystem "Social Networks" "Share links" "external"

user -> webapp "استفاده"

reporter -> webapp "استفاده"

admin -> adminsvc "استفاده"

webapp -> bff "HTTP(S)"

mobile -> bff "HTTP(S)"

bff -> auth "OIDC introspect"

bff -> content "REST/GraphQL"

bff -> search "Search API"

bff -> live "SSE/WS"

bff -> comment "Comments API"

bff -> interact "Interactions API"

bff -> media "Signed URLs"

content -> db "CRUD"

comment -> db "CRUD"

interact -> db "Counters (eventual)"

adminsvc -> content "Ops/Reports"

media -> obj "Store media"

webapp -> cdn "Fetch media/pages"

search -> index "Index & Query"

content -> bus "publish ArticlePublished"

comment -> bus "publish CommentSubmitted"

mod -> bus "consume moderation events"

search -> bus "consume indexing events"

notify -> mail "send emails/push"

auth -> idp "federation (optional)"

webapp -> social "share links"

bff -> cache "read/write cache"

content -> cache "invalidate fragments"

live -> cache "snapshot state"

}

views {

systemContext newsCtx "C1 - System Context" {

include \*

autoLayout lr

}

container newsC2 "C2 - Containers" {

include \*

autoLayout lr

}

component contentC3 "C3 - Content Service Components" {

container content

include content->db

include content->bus

autoLayout lr

}

styles {

element "external" { background #eeeeee }

element "container" { shape RoundedBox }

element "person" { shape Person }

relationship { routing Orthogonal }

}

}

}

**نمودار توالی (کمکی – انتشار خبر)**

خارج از C4 اما برای شفافیت جریان حیاتی

Reporter → WebApp → BFF → ContentService → DB

ContentService → EventBus: ArticlePublished

EventBus → SearchService: consume & index

ContentService → Cache/CDN: Invalidate

BFF → WebApp: 200 OK with Article URL

**یادداشت‌های پیاده‌سازی**

* برای C3 سایر سرویس‌ها (Search/Comment/Live) می‌توانیم DSL مستقل اضافه کنیم.
* در صورت نیاز، نسخه «ماژولار مونو‌لیت» با Componentهای لایه‌ای (Controller, Service, Repository) نیز ارائه می‌شود.