برای پیاده‌سازی یک سایت فروشگاهی در مقیاس بزرگ مانند آمازون با معماری میکروسرویس، در ادامه نسخه‌ای کامل‌تر و حرفه‌ای‌تر از لیست میکروسرویس‌ها، وظایف هر سرویس، دیتابیس پیشنهادی، زبان‌های پیشنهادی (با در نظر گرفتن ترجیح شما به .NET Core و انعطاف برای زبان‌های بهتر در موارد خاص)، و تکنولوژی‌های مرتبط برای بهینه‌سازی پرفرمنس ارائه می‌کنم. هدف این است که معماری‌ای ارائه شود که نه‌تنها مقیاس‌پذیر و قابل نگهداری باشد، بلکه از نظر حرفه‌ای با پیچیدگی‌های یک پلتفرم مثل آمازون هم‌راستا باشد.

**لیست میکروسرویس‌ها و جزئیات**

**1. میکروسرویس کاربران (User Service)**

* **وظایف:**
  + ثبت‌نام، ورود و خروج، احراز هویت چندمرحله‌ای (MFA) با پروتکل‌های JWT و OAuth2.
  + مدیریت پروفایل پیشرفته (آدرس‌ها، علاقه‌مندی‌ها، تنظیمات اعلان‌ها، تاریخچه جستجو).
  + مدیریت نقش‌ها و دسترسی‌ها (RBAC و ABAC برای کنترل دقیق‌تر).
  + رمزنگاری داده‌های حساس (مانند رمز عبور و اطلاعات پرداخت).
  + لاگ فعالیت‌ها و تحلیل رفتار کاربر برای امنیت (مثل تشخیص ورود مشکوک).
* **دیتابیس پیشنهادی:**
  + **PostgreSQL**: برای تراکنش‌های ACID، پشتیبانی از JSONB برای داده‌های پویا، و قابلیت‌های رمزنگاری داخلی.
  + **Redis**: برای کش کردن توکن‌ها و جلسات (Session Management).
* **زبان/فریمورک:**
  + **.NET Core**: با ASP.NET Identity برای مدیریت کاربران و امنیت.
  + **جایگزین پیشنهادی:** Node.js + TypeScript با NestJS برای توسعه سریع و مقیاس‌پذیری بالا.
* **تکنولوژی‌های کمکی:**
  + **Keycloak**: برای مدیریت هویت و SSO (Single Sign-On).
  + **HashiCorp Vault**: برای ذخیره‌سازی امن کلیدها و رمزنگاری.

**2. میکروسرویس محصولات (Product Service)**

* **وظایف:**
  + مدیریت کاتالوگ محصولات (ویژگی‌های متغیر، دسته‌بندی‌ها، SKUها).
  + مدیریت تصاویر و محتوای چندرسانه‌ای (مانند ویدئوهای محصول).
  + مدیریت نظرات، امتیازات، و پرسش و پاسخ کاربران.
  + همگام‌سازی با سیستم جستجو و توصیه‌گر.
  + اعمال تخفیف‌ها و پیشنهادات ویژه با زمان‌بندی.
* **دیتابیس پیشنهادی:**
  + **MongoDB**: برای انعطاف‌پذیری در ذخیره‌سازی محصولات با ویژگی‌های متنوع.
  + **Amazon S3**: برای ذخیره‌سازی تصاویر و ویدئوها.
* **زبان/فریمورک:**
  + **.NET Core**: با MongoDB.Driver برای مدیریت کاتالوگ.
  + **جایگزین پیشنهادی:** Python + FastAPI برای پردازش داده‌های نیمه‌ساختاریافته و ادغام با ML.
* **تکنولوژی‌های کمکی:**
  + **Cloudflare CDN**: برای تحویل سریع محتوای چندرسانه‌ای.
  + **Apache Kafka**: برای انتشار تغییرات محصول به سرویس‌های دیگر.

**3. میکروسرویس سفارشات (Order Service)**

* **وظایف:**
  + ثبت و اعتبارسنجی سفارشات (چک کردن موجودی و پرداخت).
  + مدیریت چرخه حیات سفارش (لغو، بازگشت، تغییر آدرس).
  + پیاده‌سازی الگوی Saga (Choreography یا Orchestration) برای تراکنش‌های توزیع‌شده.
  + همگام‌سازی با انبار، پرداخت، و تحویل.
  + گزارش‌گیری لحظه‌ای از وضعیت سفارشات.
* **دیتابیس پیشنهادی:**
  + **PostgreSQL**: برای یکپارچگی داده‌ها و پشتیبانی از تراکنش‌های پیچیده.
* **زبان/فریمورک:**
  + **.NET Core**: با Entity Framework Core برای ORM قوی.
  + **جایگزین پیشنهادی:** Java + Spring Boot با Axon Framework برای مدیریت Saga.
* **تکنولوژی‌های کمکی:**
  + **Apache Kafka**: برای ارسال رویدادهای سفارش به سرویس‌ها.
  + **Dapr**: برای ساده‌سازی مدیریت تراکنش‌های توزیع‌شده.

**4. میکروسرویس پرداخت (Payment Service)**

* **وظایف:**
  + ادغام با درگاه‌های پرداخت متعدد (Stripe, PayPal, درگاه‌های محلی).
  + مدیریت کیف پول دیجیتال کاربران.
  + پردازش بازپرداخت‌ها و تراکنش‌های ناموفق.
  + اعتبارسنجی امنیتی (مثل 3D Secure).
  + لاگ تراکنش‌ها برای حسابرسی.
* **دیتابیس پیشنهادی:**
  + **PostgreSQL**: برای تراکنش‌های اتمیک و قابلیت ردیابی.
  + **Redis**: برای کش کردن تراکنش‌های موقت.
* **زبان/فریمورک:**
  + **.NET Core**: با HttpClient برای اتصال به APIهای پرداخت.
  + **جایگزین پیشنهادی:** Go برای سرعت بالا و پردازش هم‌زمان تراکنش‌ها.
* **تکنولوژی‌های کمکی:**
  + **Stripe SDK**: برای پرداخت امن و سریع.
  + **OpenTelemetry**: برای ردیابی تراکنش‌ها.

**5. میکروسرویس انبار و موجودی (Inventory Service)**

* **وظایف:**
  + مدیریت موجودی چندانباره (انبارهای محلی و جهانی).
  + به‌روزرسانی لحظه‌ای موجودی با استفاده از قفل‌گذاری توزیع‌شده.
  + پیش‌بینی نیاز به موجودی با تحلیل داده‌ها.
  + ارسال هشدار برای کمبود یا مازاد موجودی.
* **دیتابیس پیشنهادی:**
  + **Redis**: برای سرعت بالا در مدیریت موجودی لحظه‌ای.
  + **Cassandra**: برای مقیاس‌پذیری افقی در انبارهای متعدد.
* **زبان/فریمورک:**
  + **.NET Core**: با StackExchange.Redis.
  + **جایگزین پیشنهادی:** Elixir برای مدیریت هم‌زمانی و Real-Time.
* **تکنولوژی‌های کمکی:**
  + **ZooKeeper**: برای هماهنگی قفل‌های توزیع‌شده.
  + **Kafka Streams**: برای پردازش جریان موجودی.

**6. میکروسرویس جستجو (Search Service)**

* **وظایف:**
  + جستجوی پیشرفته با فیلترهای پویا (قیمت، برند، رتبه‌بندی).
  + پشتیبانی از جستجوی متنی کامل (Full-Text Search).
  + پیشنهادات هوشمند با Auto-Complete و Spell Correction.
  + شخصی‌سازی نتایج بر اساس رفتار کاربر.
* **دیتابیس/موتور پیشنهادی:**
  + **Elasticsearch**: برای جستجوی سریع و تحلیل داده‌ها.
* **زبان/فریمورک:**
  + **.NET Core**: با NEST برای تعامل با Elasticsearch.
  + **جایگزین پیشنهادی:** Python با Elasticsearch DSL برای انعطاف‌پذیری بیشتر.
* **تکنولوژی‌های کمکی:**
  + **Apache Solr**: به‌عنوان جایگزین برای Elasticsearch در سناریوهای خاص.
  + **NGINX**: برای Load Balancing درخواست‌ها.

**7. میکروسرویس توصیه‌گر (Recommendation Service)**

* **وظایف:**
  + پیشنهاد محصولات بر اساس رفتار کاربر، تاریخچه خرید، و الگوهای جمعی.
  + پیاده‌سازی الگوریتم‌های ML (Collaborative Filtering, Deep Learning).
  + به‌روزرسانی لحظه‌ای پیشنهادات با داده‌های جدید.
  + A/B Testing برای بهینه‌سازی مدل‌ها.
* **دیتابیس پیشنهادی:**
  + **MongoDB**: برای ذخیره‌سازی تعاملات کاربر.
  + **Neo4j**: برای تحلیل روابط گرافی و توصیه‌های پیچیده.
* **زبان/فریمورک:**
  + **Python**: با TensorFlow/PyTorch برای مدل‌های ML.
  + **جایگزین پیشنهادی:** Rust برای پردازش سریع مدل‌های سنگین.
* **تکنولوژی‌های کمکی:**
  + **Apache Airflow**: برای زمان‌بندی اجرای مدل‌ها.
  + **Kubeflow**: برای مدیریت پایپ‌لاین‌های ML.

**8. میکروسرویس تحویل (Shipping Service)**

* **وظایف:**
  + محاسبه هزینه و زمان تحویل با الگوریتم‌های بهینه‌سازی مسیر.
  + ادغام با سرویس‌های حمل‌ونقل (UPS, DHL, پست محلی).
  + رهگیری مرسولات و ارسال اعلان به کاربران.
  + مدیریت مرجوعی‌ها و تعویض‌ها.
* **دیتابیس پیشنهادی:**
  + **PostgreSQL**: برای ذخیره‌سازی اطلاعات حمل‌ونقل و روابط.
* **زبان/فریمورک:**
  + **.NET Core**: با HttpClient برای APIهای خارجی.
  + **جایگزین پیشنهادی:** Node.js برای پردازش غیرهمزمان.
* **تکنولوژی‌های کمکی:**
  + **Google OR-Tools**: برای بهینه‌سازی مسیر تحویل.
  + **Twilio**: برای ارسال اعلان‌های رهگیری.

**9. میکروسرویس نوتیفیکیشن (Notification Service)**

* **وظایف:**
  + ارسال اعلان چندکاناله (ایمیل، SMS، Push Notification).
  + مدیریت الگوهای پیام پویا و چندزبانه.
  + اولویت‌بندی و صف‌بندی اعلان‌ها.
  + تحلیل نرخ تعامل با اعلان‌ها.
* **دیتابیس پیشنهادی:**
  + **MongoDB**: برای لاگ‌ها و الگوهای پیام.
  + **Redis**: برای صف‌بندی اعلان‌ها.
* **زبان/فریمورک:**
  + **.NET Core**: با Hangfire برای مدیریت صف‌ها.
  + **جایگزین پیشنهادی:** Go با Goroutines برای پردازش Real-Time.
* **تکنولوژی‌های کمکی:**
  + **AWS SNS**: برای ارسال اعلان‌های مقیاس‌پذیر.
  + **Firebase Cloud Messaging**: برای Push Notification.

**10. میکروسرویس تحلیل و گزارشات (Analytics Service)**

* **وظایف:**
  + جمع‌آوری داده‌های کلان (Big Data) از رفتار کاربران و فروش.
  + تولید گزارش‌های Real-Time و تاریخی (فروش، ترافیک، KPIها).
  + پیش‌بینی روندها با استفاده از ML.
  + داشبوردهای تعاملی برای تیم‌های داخلی.
* **دیتابیس پیشنهادی:**
  + **ClickHouse**: برای پردازش سریع داده‌های حجیم.
  + **TimescaleDB**: برای داده‌های سری زمانی.
* **زبان/فریمورک:**
  + **.NET Core**: با ادغام Power BI.
  + **جایگزین پیشنهادی:** Python با Pandas و Superset.
* **تکنولوژی‌های کمکی:**
  + **Apache Spark**: برای پردازش کلان‌داده.
  + **Grafana Tempo**: برای ردیابی و تحلیل.

**11. میکروسرویس سبد خرید (Cart Service)**

* **وظایف:**
  + مدیریت سبد خرید کاربران (افزودن، حذف، تغییر تعداد).
  + اعمال تخفیف‌ها و محاسبه قیمت نهایی.
  + ذخیره موقت سبد برای کاربران مهمان.
  + ادغام با موجودی و سفارشات.
* **دیتابیس پیشنهادی:**
  + **Redis**: برای سرعت و ذخیره‌سازی موقت.
* **زبان/فریمورک:**
  + **.NET Core**: با StackExchange.Redis.
  + **جایگزین پیشنهادی:** Node.js برای عملکرد سبک‌تر.
* **تکنولوژی‌های کمکی:**
  + **Memcached**: به‌عنوان جایگزین Redis در برخی سناریوها.
  + **Kafka**: برای همگام‌سازی سبد با سرویس سفارشات.

**12. میکروسرویس بازخورد (Review Service)**

* **وظایف:**
  + مدیریت نظرات، امتیازات، و پرسش و پاسخ محصولات.
  + مودریشن خودکار با NLP (تشخیص محتوای نامناسب).
  + تحلیل احساسات (Sentiment Analysis) نظرات.
  + گزارش تخلفات کاربران.
* **دیتابیس پیشنهادی:**
  + **MongoDB**: برای داده‌های پویا و غیرساختاریافته.
* **زبان/فریمورک:**
  + **.NET Core**: با MongoDB.Driver.
  + **جایگزین پیشنهادی:** Python با Flask و Hugging Face برای NLP.
* **تکنولوژی‌های کمکی:**
  + **TensorFlow Lite**: برای مودریشن سبک‌تر.
  + **Elasticsearch**: برای جستجوی نظرات.

**13. میکروسرویس بازاریابی (Marketing Service)**

* **وظایف:**
  + مدیریت کمپین‌های تبلیغاتی هدفمند.
  + تولید و اعتبارسنجی کوپن‌ها و کدهای تخفیف.
  + تحلیل ROI و اثربخشی کمپین‌ها.
  + ادغام با سیستم توصیه‌گر برای تبلیغات شخصی‌سازی‌شده.
* **دیتابیس پیشنهادی:**
  + **PostgreSQL**: برای روابط پیچیده و گزارش‌گیری.
* **زبان/فریمورک:**
  + **.NET Core**: با Entity Framework Core.
  + **جایگزین پیشنهادی:** Python برای ادغام با ابزارهای تحلیلی.
* **تکنولوژی‌های کمکی:**
  + **Segment**: برای ردیابی کمپین‌ها.
  + **Redis**: برای کش کردن کدهای تخفیف.

**14. میکروسرویس مدیریت فروشندگان (Vendor Service) *(اضافه شده برای حرفه‌ای‌تر شدن)***

* **وظایف:**
  + ثبت‌نام و احراز هویت فروشندگان.
  + مدیریت محصولات و موجودی فروشندگان.
  + محاسبه کمیسیون و تسویه‌حساب با فروشندگان.
  + داشبورد اختصاصی برای فروشندگان.
* **دیتابیس پیشنهادی:**
  + **PostgreSQL**: برای روابط و تراکنش‌ها.
  + **MongoDB**: برای داده‌های متغیر فروشندگان.
* **زبان/فریمورک:**
  + **.NET Core**: برای یکپارچگی با سیستم اصلی.
  + **جایگزین پیشنهادی:** Java + Spring Boot برای پیچیدگی‌های بیشتر.
* **تکنولوژی‌های کمکی:**
  + **Stripe Connect**: برای مدیریت پرداخت به فروشندگان.
  + **Grafana**: برای داشبوردهای فروشندگان.

**15. میکروسرویس مدیریت محتوای استاتیک (Content Service) *(اضافه شده برای حرفه‌ای‌تر شدن)***

* **وظایف:**
  + مدیریت صفحات فرود، بلاگ، و محتوای تبلیغاتی.
  + پشتیبانی از چندزبانگی (Localization).
  + بهینه‌سازی SEO برای محتوا.
* **دیتابیس پیشنهادی:**
  + **MongoDB**: برای انعطاف‌پذیری محتوا.
  + **S3**: برای ذخیره‌سازی فایل‌ها.
* **زبان/فریمورک:**
  + **.NET Core**: با Razor Pages.
  + **جایگزین پیشنهادی:** Node.js با Next.js برای SSR و SEO.
* **تکنولوژی‌های کمکی:**
  + **Cloudflare Workers**: برای تحویل سریع محتوا.
  + **Strapi**: برای CMS بدون سر (Headless CMS).

**تکنولوژی‌های کلیدی برای بهترین پرفرمنس**

1. **API Gateway:**
   * **Kong** یا **Envoy**: برای مسیریابی پیشرفته، Rate Limiting، و Authentication مرکزی.
2. **Message Broker:**
   * **Apache Kafka**: برای معماری Event-Driven و پردازش Real-Time.
   * **RabbitMQ**: برای سناریوهای ساده‌تر.
3. **کش:**
   * **Redis Enterprise**: برای کش پیشرفته و مقیاس‌پذیر.
   * **Varnish**: برای کش HTTP در لایه Gateway.
4. **مونیتورینگ و لاگ‌گیری:**
   * **Prometheus + Grafana**: برای مانیتورینگ متریک‌ها.
   * **ELK Stack (Elasticsearch, Logstash, Kibana)**: برای لاگ‌گیری توزیع‌شده.
   * **Jaeger**: برای ردیابی درخواست‌ها (Distributed Tracing).
5. **Containerization & Orchestration:**
   * **Docker + Kubernetes**: برای مقیاس‌پذیری و مدیریت کانتینرها.
   * **Istio**: برای Service Mesh و امنیت بین سرویس‌ها.
6. **پروتکل ارتباطی:**
   * **gRPC**: برای ارتباطات داخلی با پرفرمنس بالا.
   * **GraphQL**: برای APIهای انعطاف‌پذیر با کلاینت‌ها.
   * **REST**: برای سازگاری با سیستم‌های خارجی.
7. **CI/CD:**
   * **GitHub Actions** یا **GitLab CI/CD**: برای اتوماسیون حرفه‌ای.
   * **ArgoCD**: برای GitOps و مدیریت Deployment.
8. **امنیت:**
   * **WAF (Cloudflare)**: برای محافظت در برابر حملات.
   * **OPA (Open Policy Agent)**: برای سیاست‌گذاری دسترسی‌ها.

**جمع‌بندی حرفه‌ای**

* **زبان اصلی:** .NET Core به‌عنوان پایه اصلی مناسب است و برای اکثر سرویس‌ها کارایی بالایی دارد. با این حال، برای سرویس‌های تخصصی (مثل توصیه‌گر، جستجو، یا پردازش Real-Time) زبان‌هایی مثل Python، Go، یا Elixir می‌توانند بهینه‌تر باشند.
* **دیتابیس:** ترکیب حرفه‌ای SQL (PostgreSQL) و NoSQL (MongoDB, Redis, Cassandra) با ابزارهای خاص مثل Elasticsearch و Neo4j بسته به نیاز هر سرویس استفاده می‌شود.
* **پرفرمنس:** استفاده از معماری Event-Driven با Kafka، کش چندلایه (Redis + Varnish)، و ابزارهای مانیتورینگ پیشرفته (Prometheus, Jaeger) تضمین‌کننده Latency کم و مقیاس‌پذیری بالاست.
* **مقیاس‌پذیری و انعطاف:** اضافه کردن سرویس‌هایی مثل Vendor Service و Content Service، این معماری را به سطحی حرفه‌ای‌تر و نزدیک‌تر به آمازون می‌رساند.

این معماری نه‌تنها برای یک سایت فروشگاهی بزرگ مناسب است، بلکه با پیچیدگی‌ها و نیازهای یک پلتفرم جهانی مثل آمازون هم‌راستاست و امکان توسعه مداوم و موازی تیم‌ها را فراهم می‌کند.