🧭 AWS Service → Usage Hint Cheat Sheet

Context: Dining Concierge / Real-time Restaurant Recommendation System
(Assignment 1 → Q1 → Q2 → Q8 progression)

Service	Used For / Typical Question Context	When to Use It (Hint)	Example Phrase Trigger in Question
Amazon Lex	Chatbot frontend → understands user text (e.g., *"Find Italian food near me"*)	Use whenever question mentions **chat interface, NLP, conversation, or intent extraction**.	"User speaks to bot / chat interface" → Lex for intent + slots
AWS Lambda	Core logic processing unit for everything (API handlers, aggregators, background jobs)	Always use for **stateless compute** – POST handlers, rating aggregators, trending fetchers, scheduled jobs.	"Function to process data / trigger by event / schedule" → Lambda
API Gateway	Unified entry point for REST + WebSocket APIs	Always front end your Lambdas with API Gateway for auth, throttling, and CORS.	"Expose API / GET endpoint / handle HTTP requests" → API Gateway
Amazon SQS	Decouples producers and consumers; buffers requests for async processing	Use when you see **asynchronous, event-driven, avoid**	"Prevent bottlenecks / decouple flows" → SQS

		overload, batch processing, queue.	
Amazon SNS	Sends notifications / emails / mobile alerts to users	Mention when question adds “notify users / real-time alerts / publish-subscribe”.	“Send update to user / publish message” → SNS (or SES for email)
Amazon SES	Email delivery of recommendations or reports	Use for **sending recommendation emails or system alerts**.	“Send email / confirmation / daily recommendations” → SES
Amazon Cognito	Handles authentication and authorization with JWT tokens	Include whenever question mentions **login, user auth, profiles, secure endpoints**.	“User sign-in / token / auth required” → Cognito
Amazon DynamoDB	Primary data store for users, restaurants, likes, ratings, trending	Always when you need **fast, scalable NoSQL** reads/writes < 10 ms.	“Store user / rating / trending data / aggregate” → DynamoDB
Amazon OpenSearch (ElasticSearch)	Full-text and geo-spatial search engine for queries	Use when question mentions **search, filter, geo-based, ranking by distance or rating**.	“Find nearby restaurants / search by keyword” → OpenSearch
Amazon S3 + CloudFront	Frontend hosting, asset delivery, CDN caching for static site	Always present at top of architecture. Mention for **frontend UI / CDN / low-latency delivery**.	“Static website / UI / frontend content” → S3 + CloudFront

ElastiCache (Redis)	In-memory cache for hot data (trending, ratings, nearby restaurants)	Use for **low-latency (<1 ms)** re-reads of popular data / avoid DB hot keys.	“Cache / hot data / reduce latency / store trending” → Redis
Amazon CloudWatch	Monitoring + Metrics + Triggers (Lambda logs, schedules)	Include for **metrics, alarms, scheduled tasks, monitoring**.	“Every 5 min / daily job / monitor errors / logs” → CloudWatch
Amazon EventBridge	Scheduler & event bus for periodic Lambda jobs	Mention when you see **‘run every 15 min / 2 AM daily’ / trigger Lambda on event**.	“Scheduled job / event-based workflow” → EventBridge
AWS WAF + Shield + KMS	Security layer (attack protection + encryption)	Add whenever you discuss **security / data encryption / DDoS protection**.	“Secure data / protect API / encrypt PII” → WAF + KMS
AWS X-Ray	Distributed tracing and latency debugging	Optional for questions on **latency bottlenecks / monitoring**.	“Trace requests / identify slow Lambda” → X-Ray

---

💬 How to Think During the Exam

If the question asks for… → instantly map to these:

Question Theme / Keyword	Use These Services	Why

Chatbot / NLP	Lex → Lambda → SQS	Conversational intent handling pipeline
User Feedback / Ratings / Likes	API Gateway → Lambda → DynamoDB → SQS → Aggregator Lambda → DynamoDB update	Async writes + aggregation without blocking
Trending / Popularity / Realtime Data	External API → Lambda → DynamoDB TrendingCache → Redis → OpenSearch	Cache hot data, reduce latency
Search / Nearby Results	OpenSearch + DynamoDB + Redis	Combine geo search with cached aggregates
Wait-Time or Offers (API Integration)	External API → Lambda → DynamoDB Cache + TTL + EventBridge schedule	Cached third-party data to save API costs
Email / Alerts	SES / SNS + SQS → Lambda	Asynchronous notifications to users
Auth / Profiles / Users	Cognito + DynamoDB	Manage users securely with JWT
Scalability / Burst Traffic	SQS + Lambda auto-scale + Redis cache + DynamoDB on-demand	Decouple + cache + auto-scale
Monitoring / Automation	CloudWatch + EventBridge + X-Ray	Scheduled tasks + performance visibility

---

🧩 Quick Example Mapping

Question Type	Core Services Involved
Q1 – Personalized Recommendations	Lex, API Gateway, Lambda, SQS, DynamoDB (UserProfiles, UserLikes, TrendingCache), OpenSearch, SES
Q2 – Wait-Times & Offers	Q1 + Lambda Aggregator + EventBridge + Extra DynamoDB Caches + External APIs
Q8 – Ratings & Trending	Q1 + Q2 + ElastiCache (Redis) + DynamoDB (Ratings, TrendingCache) + External Trending API + Multi-Region Scaling
Assignment 1 (Core)	Lex → Lambda → SQS → DynamoDB → SES (+ API Gateway + Cognito)

🎯 Exam Hints (Memory Triggers)

If Question Mentions…	Immediately Think of…
“Chatbot”	**Lex + Lambda**
“High Traffic / Async / Queue”	**SQS**
“Schedule / every 5 minutes”	**EventBridge or CloudWatch Event**

“Trending / Cache / Hot Data”	**ElastiCache (Redis)**
“Personalized / User Based”	**Cognito + UserProfiles + OpenSearch**
“Search Nearby / Geo Queries”	**OpenSearch (geo_point)**
“Rating / Feedback / Write Ops”	**Lambda + DynamoDB Streams + Aggregator**
“Real-Time Notification / Alert”	**SNS / WebSocket / SES**
“Authentication / Token”	**Cognito**
“Monitoring / Logs / Metrics”	**CloudWatch + X-Ray**
“Secure Endpoints”	**WAF + KMS**
