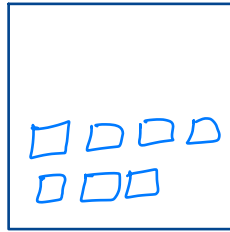


## Agenda

5 min demos →

- ✓ → upload pictures
- ✓ [ → generate caption ]
- [ → store your pics ]
- [ → organise ]
- [ → search ]
- + generating tags



google photos / Apple photos

## First step: Upload handling

API Gateway → front door for mobile or web app.

Lambda (Upload handler)

- ✓ + validate image
- ✓ + Downsizing & Storage in S3

✓ + Send the processing task  
↓  
SQS queue

S3 bucket

photo-original → Raw version

photo-thumbnails → Smaller versions

## 2. Real-time processing

SQS Queue → hold the pending image processing tasks.

Lambda  
↓

Send images to Aws Rekognition

+ handle API response

Amazon Rekognition [Heavy CV lifting]

+ Object detection  
("ipad", "widget")

+ Face recognition (bounding boxes)

+ Text extraction (OCR)

JSON Response

```
{  
  "photo_id": "ABC123",  
  "user_id": "user_456",  
  "objects": [{"dog", "beach", "sunset"}],  
  "faces": [{"box": [x1,y1,x2,y2]}],  
  "text": ["Santa Monica"],  
  "timestamp": "2023-06-15T12:30:00"  
}
```

### 3. Metadata

+ NoSQL Database  
DynamoDB

### 4. Batch Processing

+ AWS Glue

+ Recognition  
✓ + Grouping of photos & videos  
✓ + Generate best photo score  
✓ + Geotagging



+ Sagemaker

+ Host your custom model

+ personalized pet recognition  
+ Domain specific learning

### 5. Searching

Open Search → index metadata for fast querying

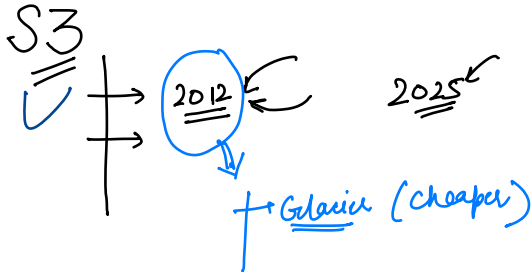
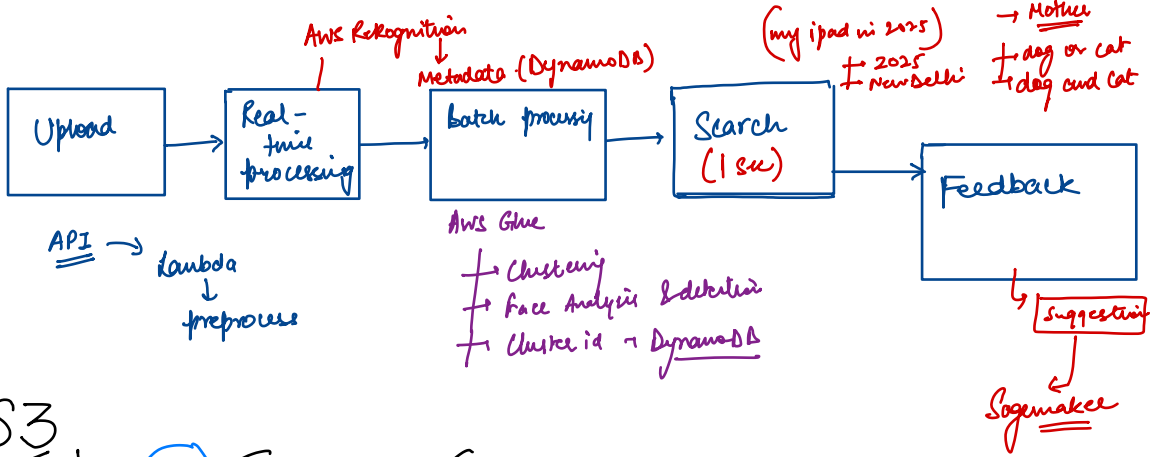
✓ + Text index → "dog", "beach", "2020"  
✓ + Vector index → embedding

<https://docs.aws.amazon.com/opensearch-service/latest/developerguide/search-example.html>

### 6. Encryption (privacy)

AWS Cognito → managing user permissions

Feature	Aws Service	Data Scientist
Object Detection	<u>Recognition</u>	Train a <u>CNN</u> (Transfer learning)
Face Analysis	<u>Recognition</u>	Face <u>embeddings</u> → <u>K-means</u>
Text extraction	<u>Recognition</u> / <u>Texttract</u>	<u>OCR pipeline</u>
Personalization	<u>Sagemaker</u> [ ]	[ Fine-tune model on private data ]



Lambda → Autoscales (upload spikes)

Batch-processing → cheaper (60-70%)

Vector Caching → Frequent face vectors Storage (DynamoDB accelerator)


# Prompt to generate the app.

Give end to end implementation of Google Photos: Image Recognition and Organization

Domain: Computer Vision, Data Management

The Problem: Users upload billions of photos, creating a massive, unstructured dataset. How can Google Photos enable users to easily find specific photos (e.g., "photos of my dog at the beach in 2023"), group similar faces, and create intelligent albums without manual tagging?

Key Design Principles & Architecture:

Massive-Scale Ingestion and Indexing:

Handles billions of image uploads, requiring a highly scalable and fault-tolerant ingestion pipeline.

Images are processed and indexed with rich metadata, including features extracted by ML models.

Advanced Computer Vision Models:

Object Recognition: Models trained on vast datasets identify objects, scenes, and activities within images (e.g., "dog," "beach," "sunset").

Facial Recognition and Grouping: Sophisticated models detect and identify faces, clustering similar faces together to enable searching by person. This requires careful handling of privacy.

Landmark Recognition: Identifies famous landmarks.

Optical Character Recognition (OCR): Extracts text from images (e.g., street signs, documents).

Semantic Segmentation: More advanced models can understand different regions of an image (e.g., sky, person, ground).

Real-time and Batch Processing:

Near Real-time Processing: As new photos are uploaded, ML models run relatively quickly to extract basic features and enable immediate search capabilities.

Batch Processing: More intensive analysis, such as deeper semantic understanding or cross-image clustering (e.g., identifying all photos of a specific trip), might be done in batch over time.

Distributed Computing: Leverages Google's vast distributed infrastructure (e.g., MapReduce, BigQuery, TensorFlow) for training large-scale models and processing the immense volume of image data.

Personalization and Privacy:

Features like "People & Pets" grouping are opt-in and processed locally or with strong privacy guarantees.

The system learns from user interactions (e.g., corrections to face labels) to refine its models.

Scalable Storage: Stores images and their extracted features efficiently in distributed file systems.

Impact: Google Photos transformed personal photo management, making it effortless for users to organize, search, and relive memories. It's a prime example of AI solving a widespread user problem at an unprecedented scale.

## HW Projects

1>

<https://aistockanalyser.com/howto>

2>

<https://github.com/ngxson/smolvlm-realtime-webcam>

# FAANG Exercise Questions

## Story Difficulty Estimation (Duolingo)

How would you design a system to assess and quantify the difficulty level of a language-learning story?

→ (Follow-up: What features or linguistic metrics would you use?)

## Editable Story Complexity (Duolingo)

Given a story, how would you automatically modify its complexity to make it easier or more challenging for the learner?

→ (Think: vocabulary, sentence length, grammar sophistication)

## Fraud Detection with Transaction Data

How would you build a machine learning model to detect fraudulent credit card transactions from labeled purchase history?

## Out-of-Stock Recommendation (E-commerce)

When a product is unavailable, how would you build a system to suggest the best possible replacement items to a customer?

## New User Follow Suggestions (Twitter)

How would you design a system to recommend accounts to follow for a new or existing user on a social media platform like Twitter? What are the challenges for cold-start users?

## Answer Ranking on Quora

Quora questions often have multiple answers. How would you build a model to rank the most helpful or relevant answers to the top? What tradeoffs affect computation?

## Trending Hashtag Detection (Twitter)

How would you design a system to automatically detect and surface trending hashtags in real time on Twitter?

## Query-Based Image Retrieval (Google Images)

How would you build a system that returns the most relevant images in response to a text query, like Google Images?

## Search Autocomplete / Sentence Completion

How would you design a model that suggests completions for a partially typed sentence or query? What training data would you use?

## Rental Listing Search (Airbnb)

How would you build a system to return the top 10 most relevant rental listings for a user searching in a particular location on Airbnb?