AWS Academy Cloud Architecting

Module 11: Caching Content

Module overview

Sections

1. Architectural need
2. Overview of caching
3. Edge caching
4. Caching web sessions
5. Caching databases

Lab

- Guided Lab: Streaming Dynamic Content Using Amazon CloudFront

Knowledge check

Module objectives

At the end of this module, you should be able to:

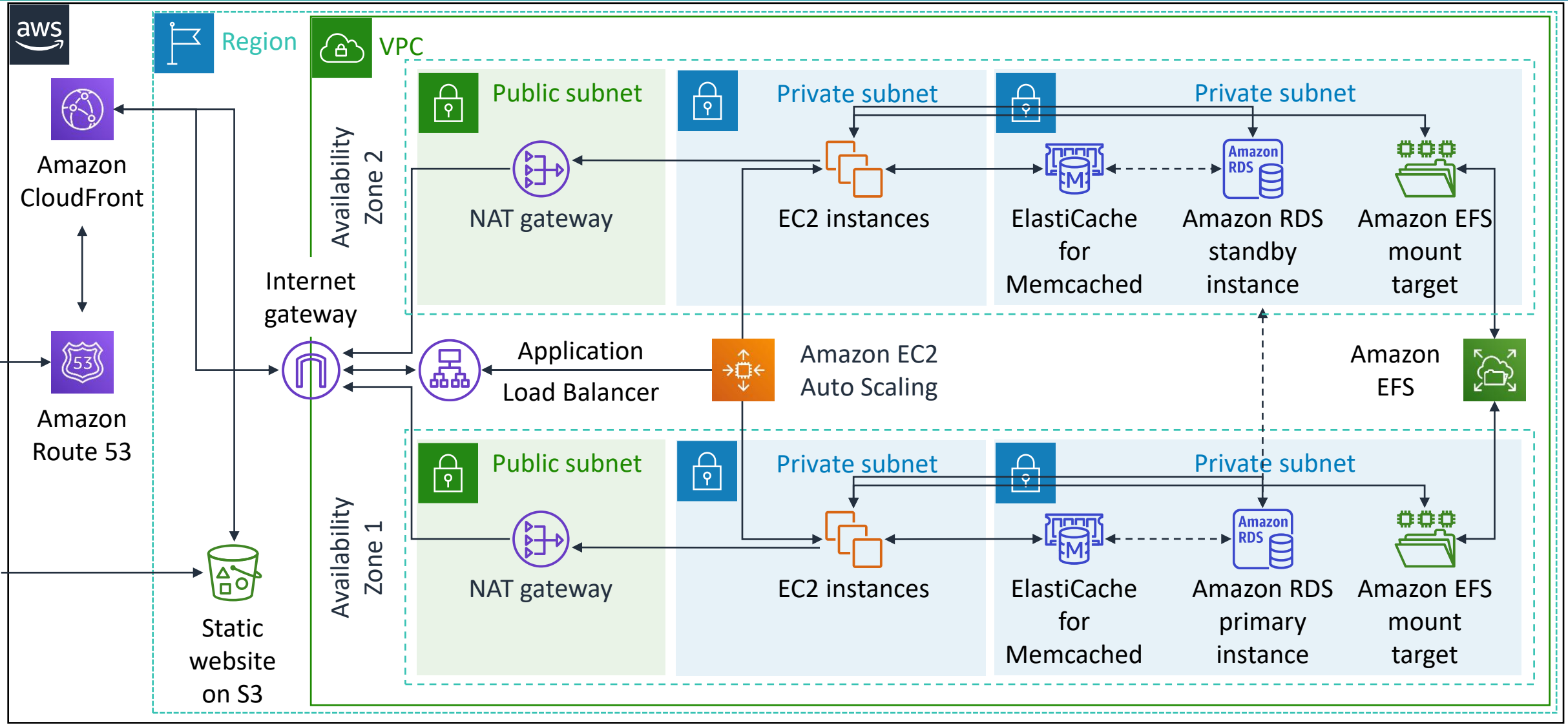
- Identify how caching content can improve application performance and reduce latency
- Identify how to design architectures that use edge locations for distribution and distributed denial of service (DDoS) protection
- Create architectures that use Amazon CloudFront to cache content
- Recognize how session management relates to caching
- Describe how to design architectures that use Amazon ElastiCache

Module 11: Caching Content

Section 1: Architectural need

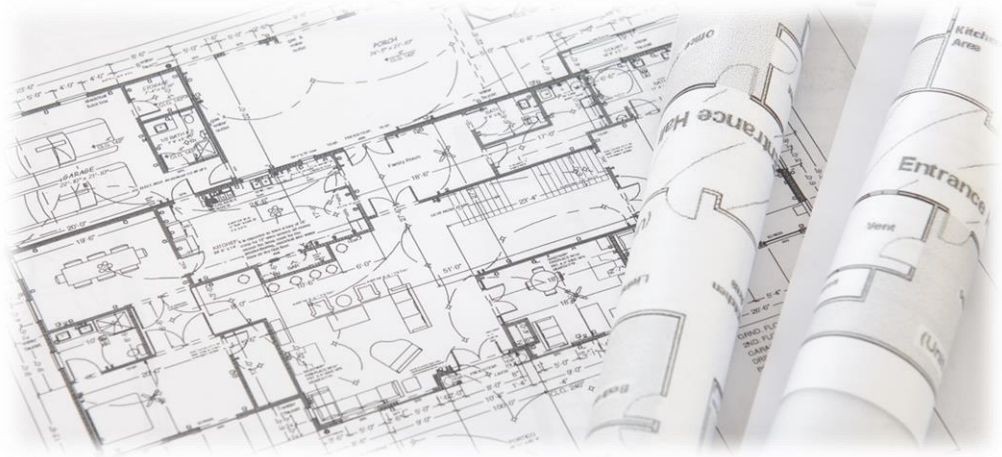
aws academy

Caching as part of a larger architecture



Café business requirement

The capacity of the café's infrastructure is constantly being overloaded with the same requests. This inefficiency is increasing cost and latency.

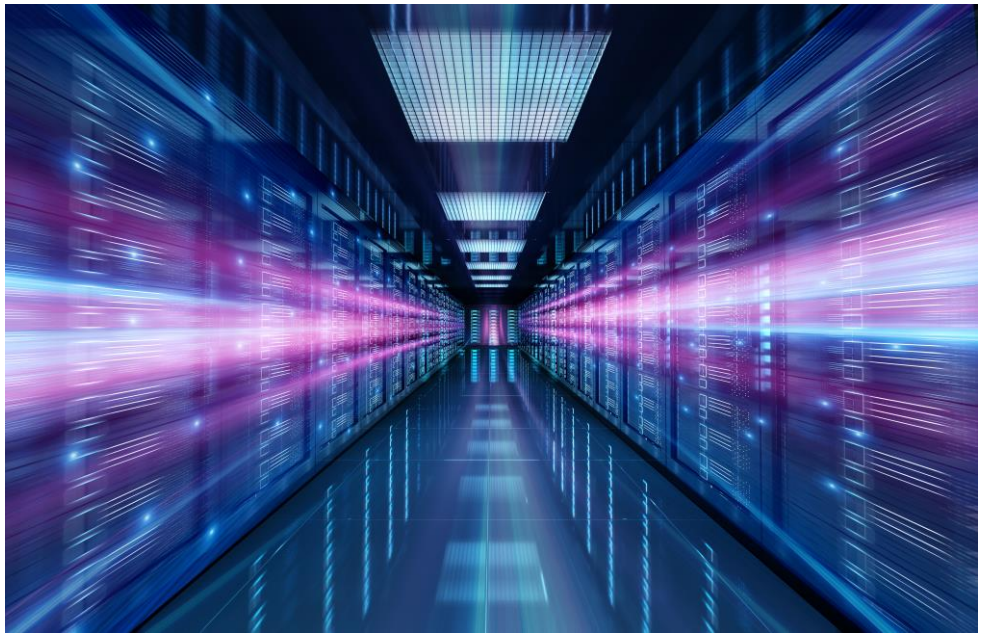


Module 11: Caching Content

Section 2: Overview of caching

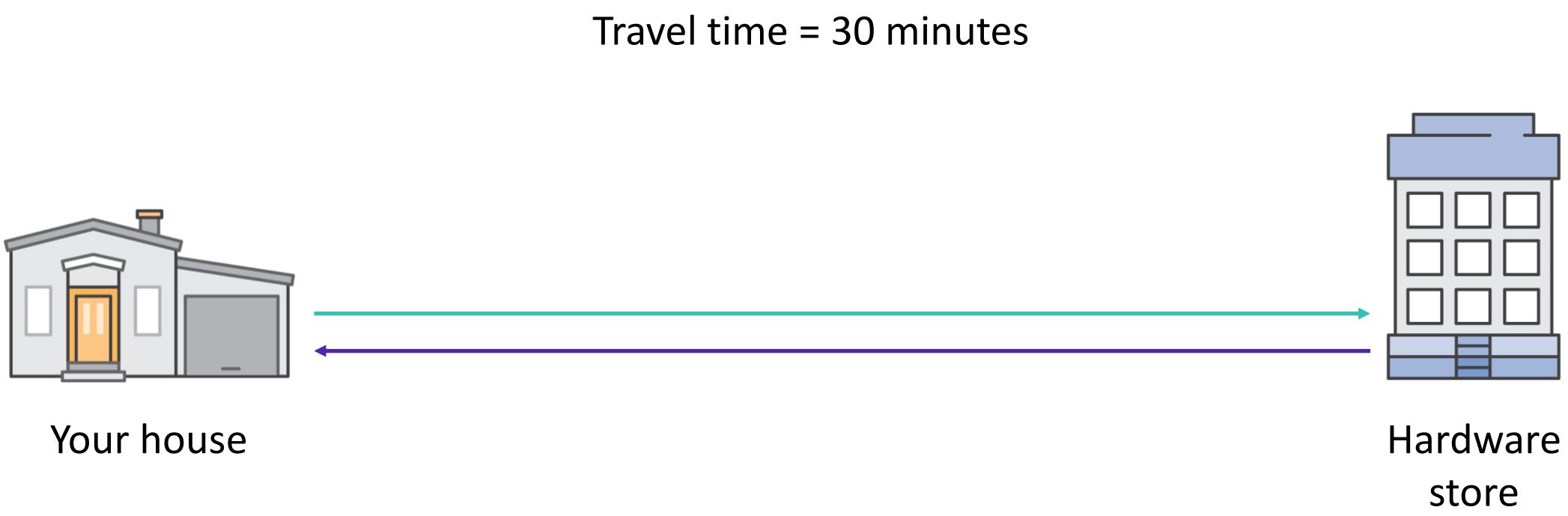
aws academy

Caching: Trading capacity for speed



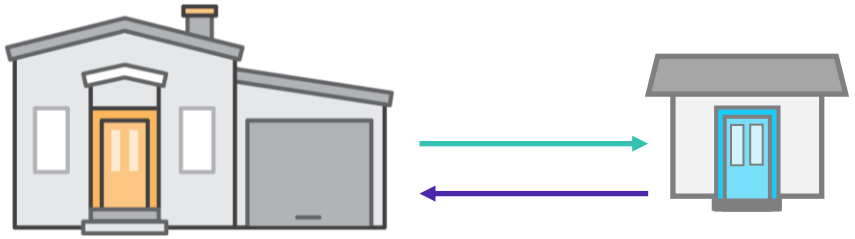
- Is a high-speed data storage layer
- Stores a subset of data
- Increases data retrieval performance
- Reduces the need to access the underlying slower storage layer

Cache example (1 of 2)



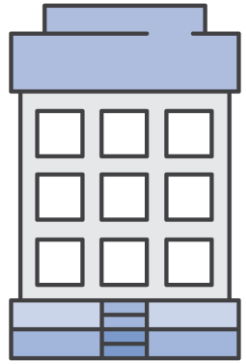
Cache example (2 of 2)

Travel time = 2 minutes



Your house

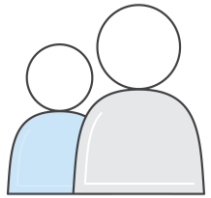
Storage unit



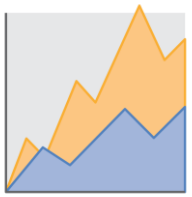
Hardware store

What should you cache?

Data that requires a slow and expensive query to acquire



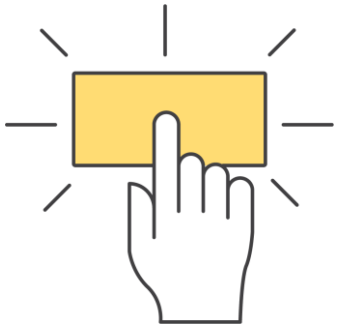
Relatively static and frequently accessed data—for example, a profile for your social media website



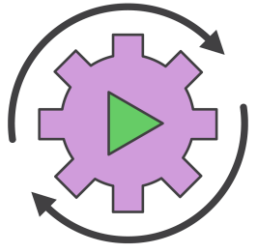
Information that can be stale for some time, such as a publicly traded stock price

Benefits of caching

Improves application speed

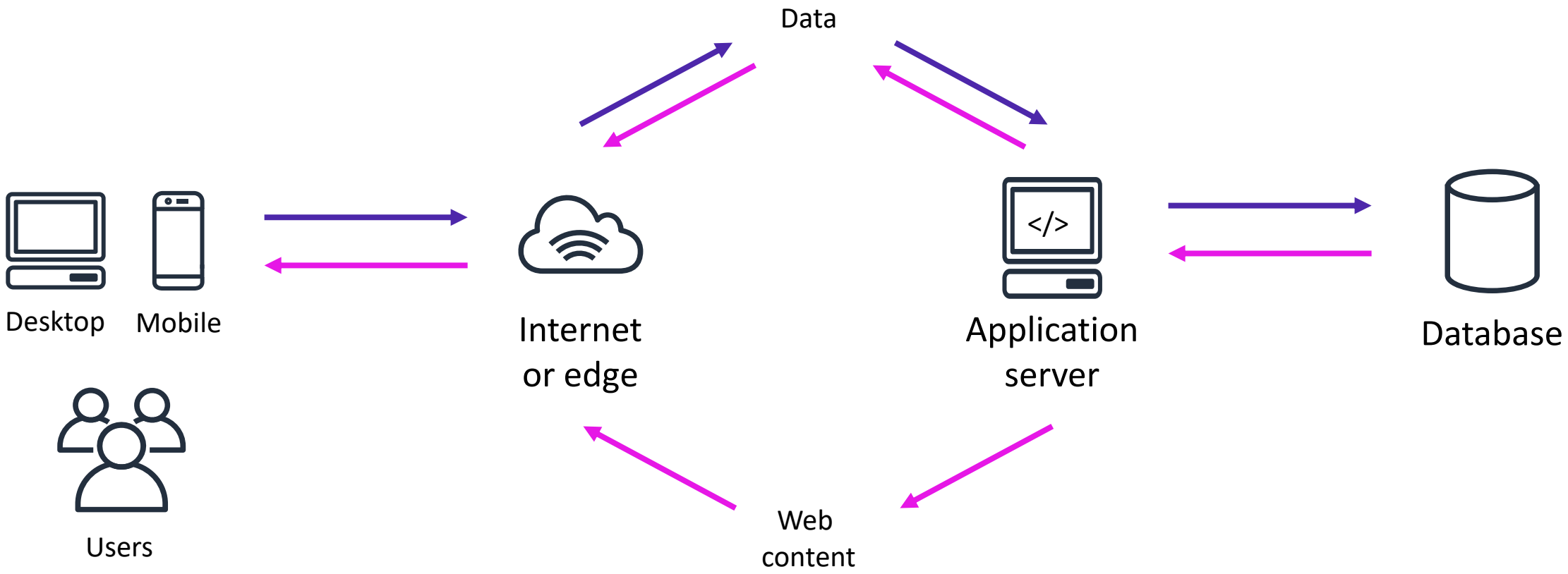


Reduces response latency



Reduces database access time

Caching throughout the data journey



Section 2 key takeaways

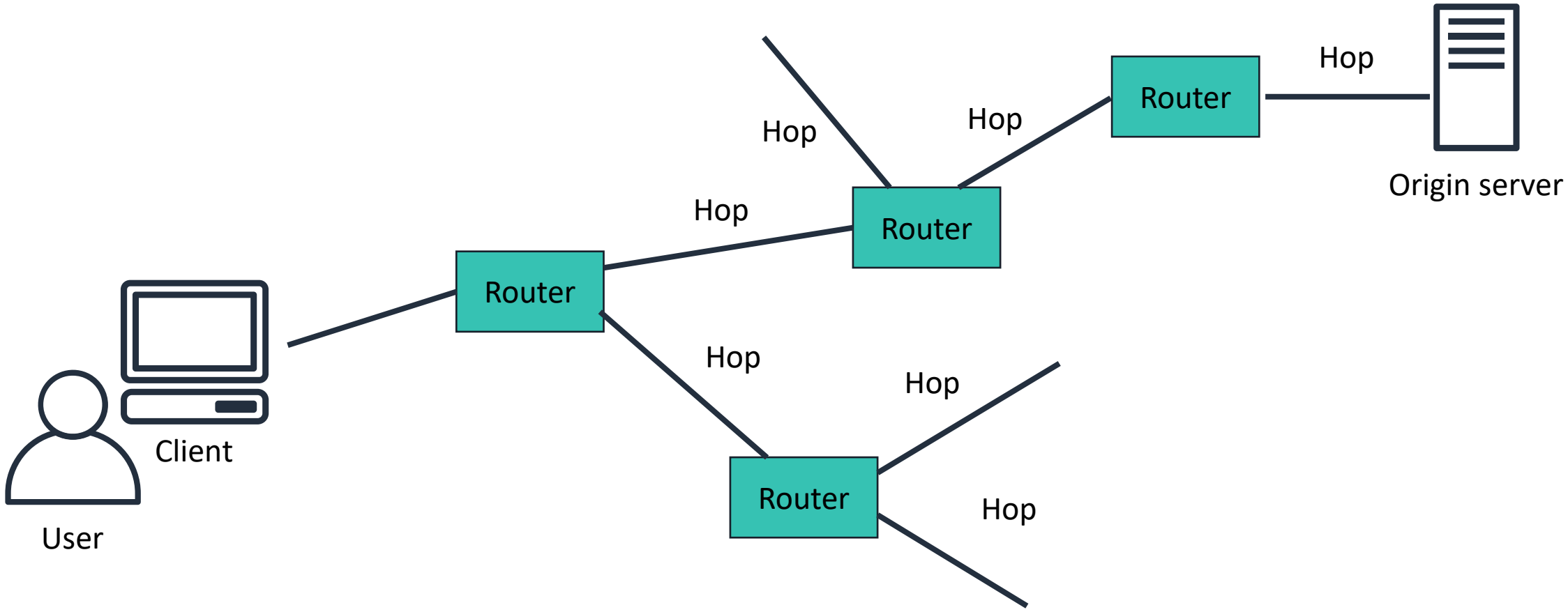
- A cache provides high throughput, low-latency access to commonly accessed application data by storing the data in memory
- When you decide what data to cache, consider speed and expense, data and access patterns, and your application's tolerance for stale data
- Caches can be applied and used throughout various layers of technology, including operating systems, networking layers, web applications, and databases

Module 11: Caching Content

Section 3: Edge caching

aws academy

Network latency



Content delivery network (CDN)

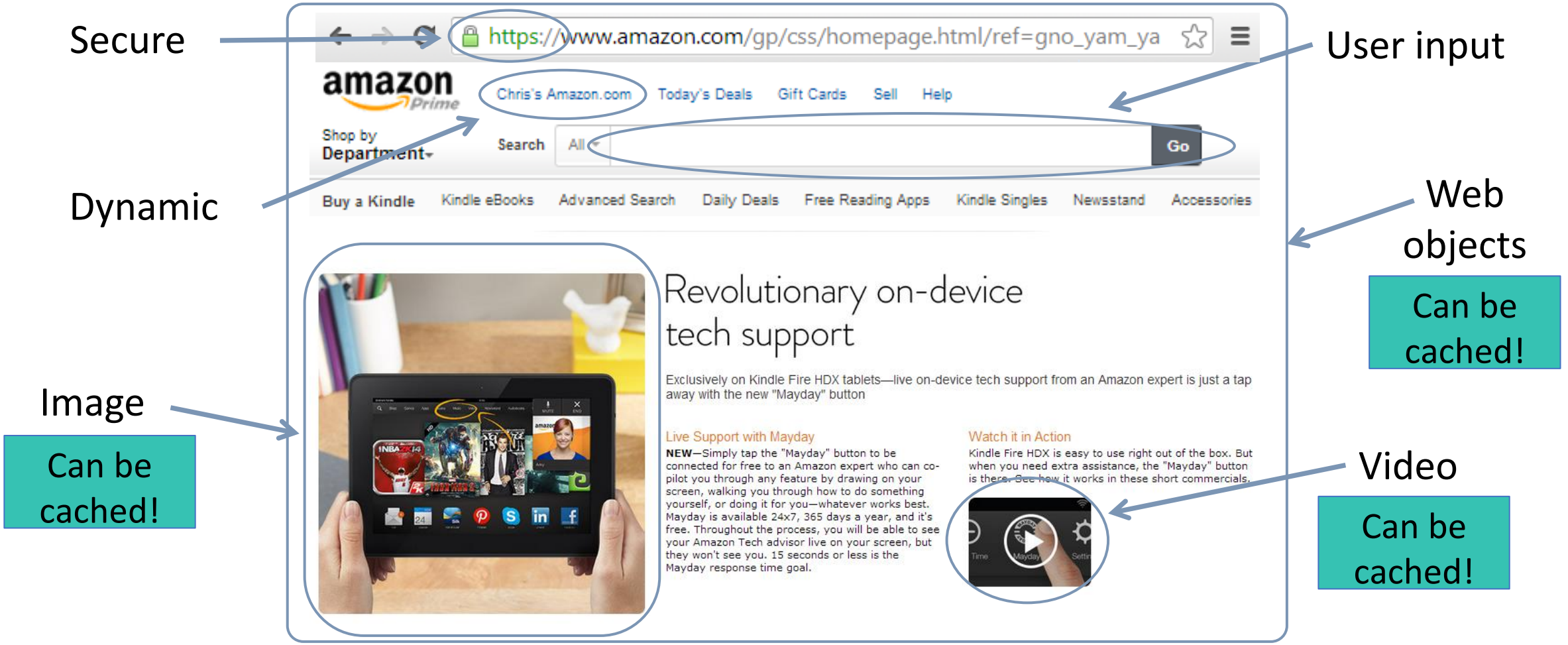
- Is a globally distributed system of caching servers
- Caches copies of commonly requested files (static content)
- Delivers a local copy of the requested content from a nearby cache edge or Point of Presence
- Improves application performance and scaling

Amazon CloudFront

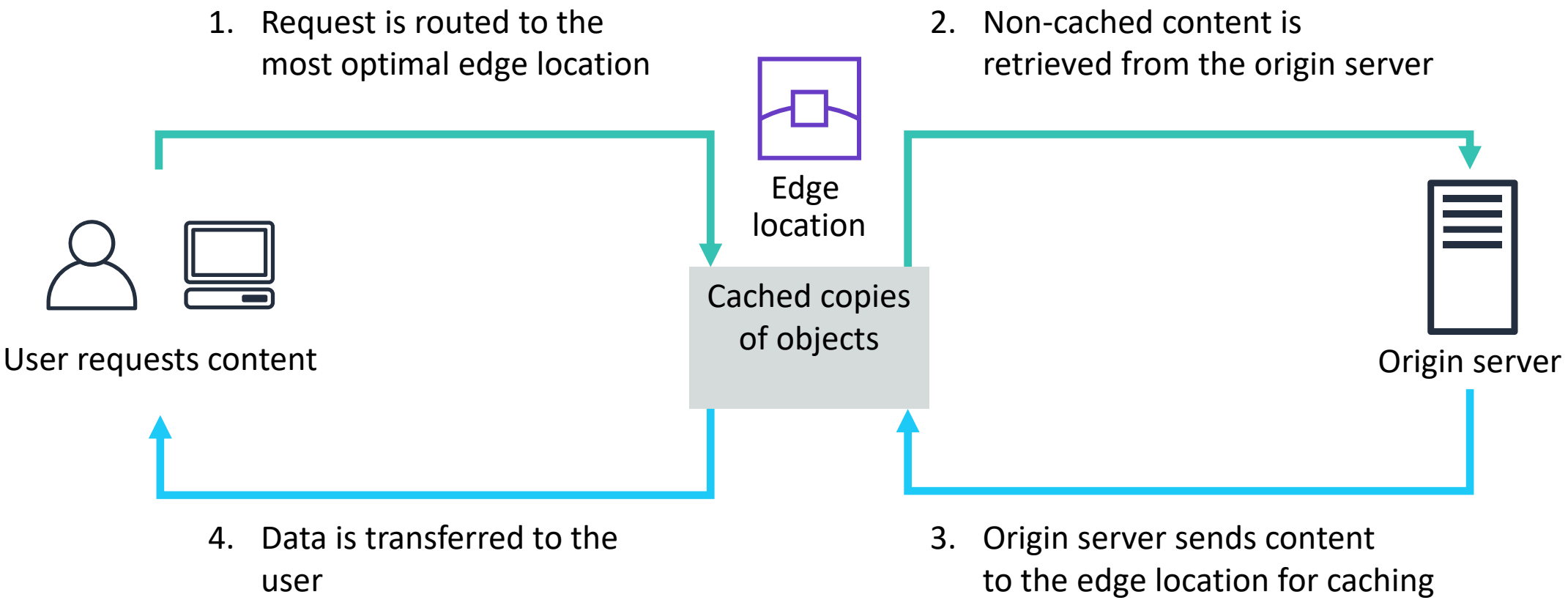
Amazon CloudFront

- Is the Amazon global CDN
- Is optimized for all delivery use cases, with a multi-tier cache by default and extensive flexibility
- Provides an extra layer of security for your architectures
- Supports WebSockets and HTTP or HTTPS methods

What type of content can you cache in an edge cache?

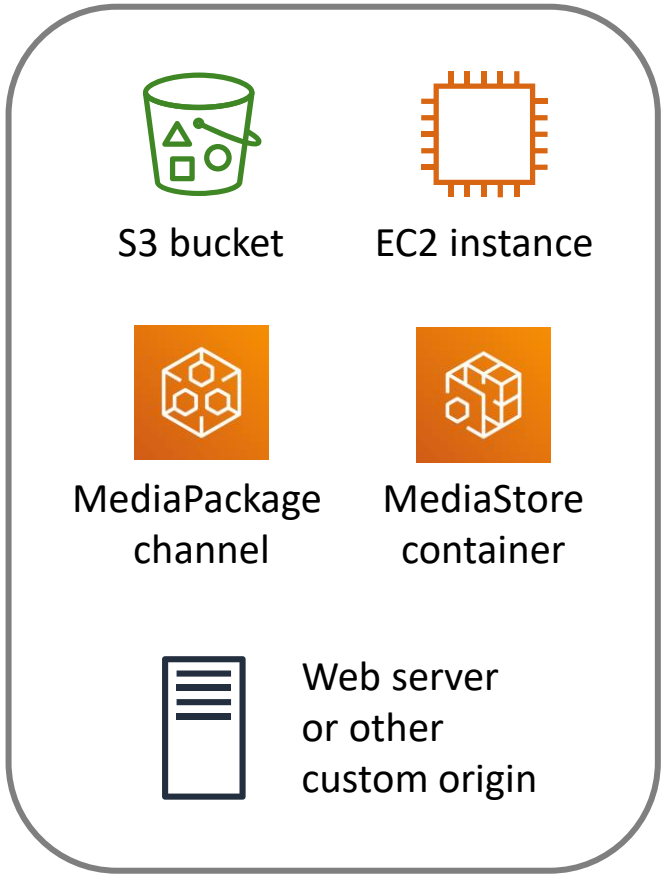


How caching works in Amazon CloudFront



How to configure a CloudFront distribution

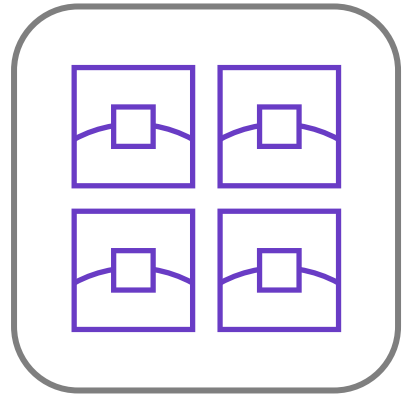
1. You specify an origin server



2. You configure the distribution

3. CloudFront assigns a domain name

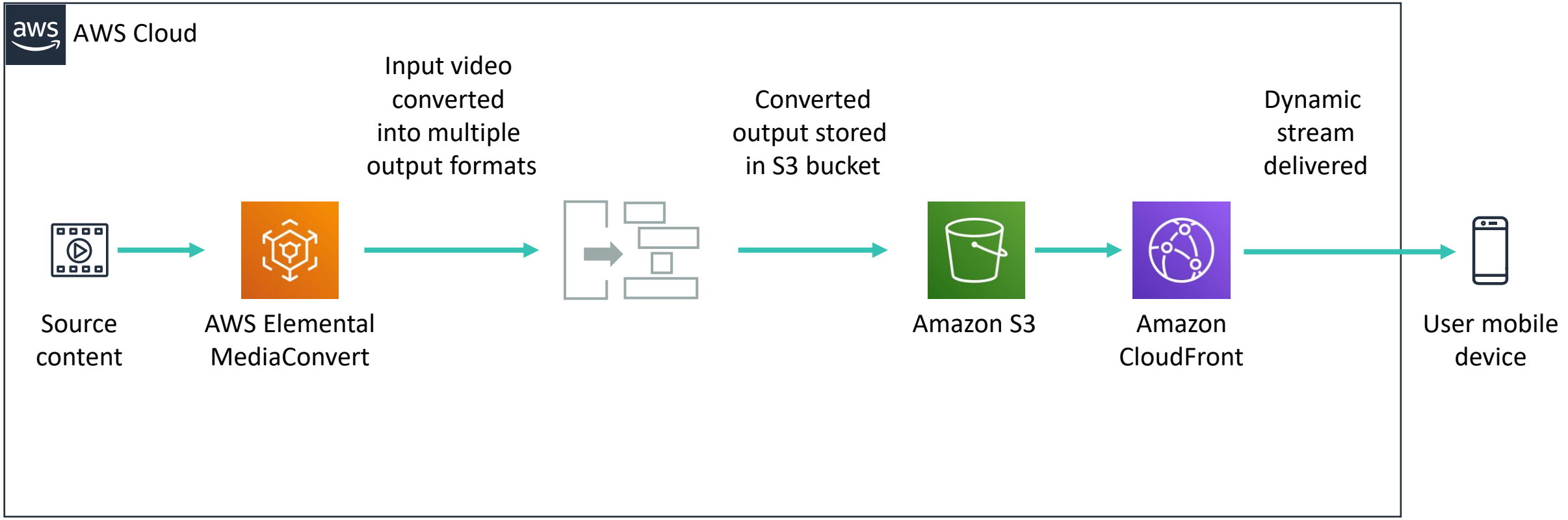
4. CloudFront sends your distribution's configuration to edge locations



How to expire content

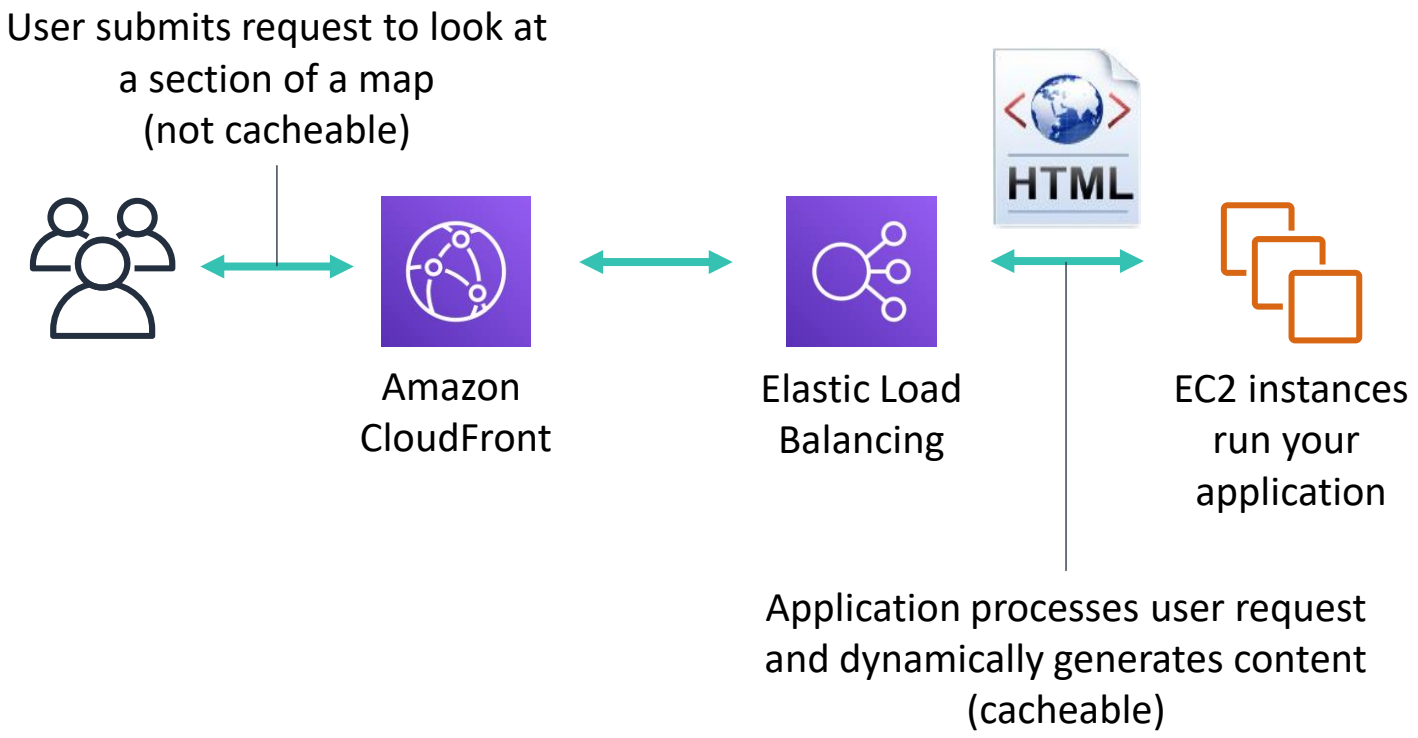
- Time to Live (TTL) –
  - Fixed period of time (expiration period)
  - Set by you
  - GET request to origin from CloudFront uses If-Modified-Since header
- Change object name –
  - Header-v1.jpg becomes Header-v2.jpg
  - New name forces immediate refresh
- Invalidate object –
  - Last resort: inefficient and expensive

Example: Video on demand streaming

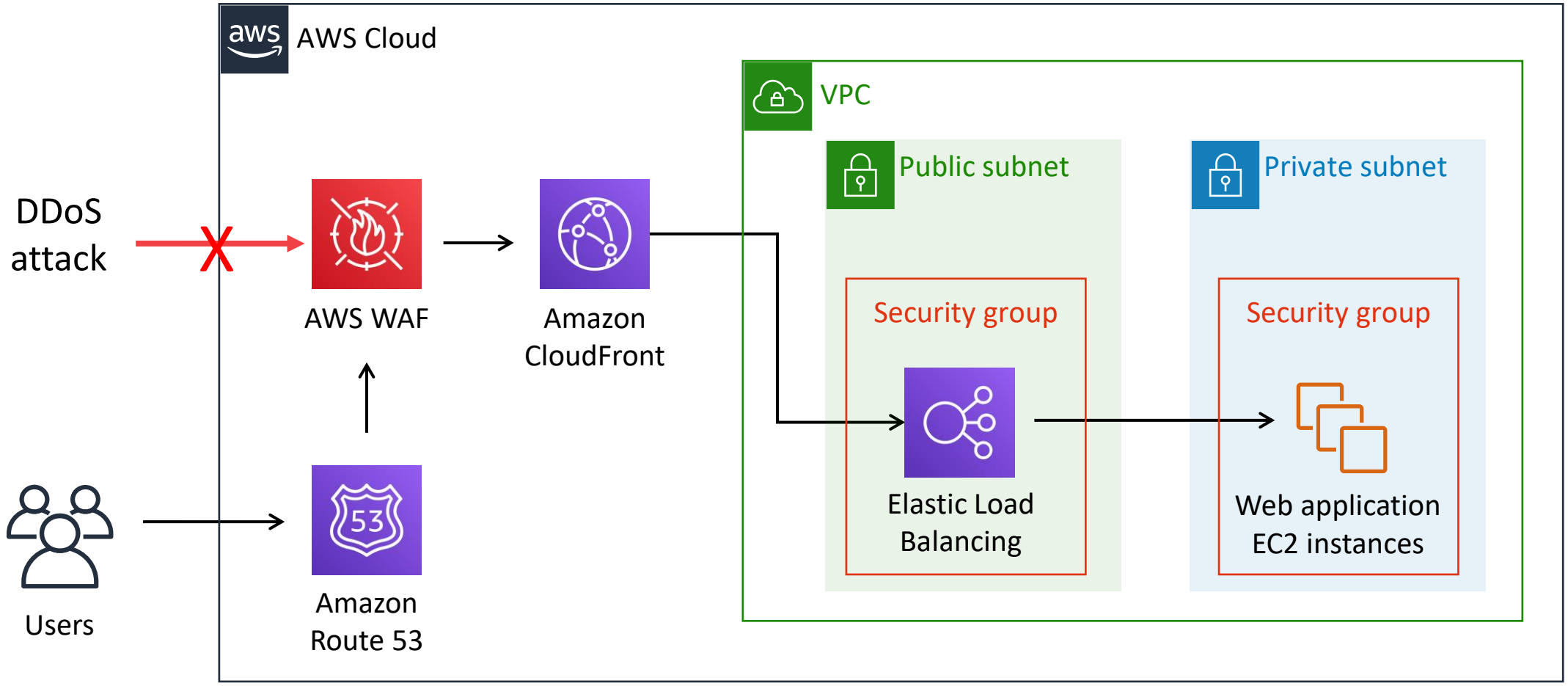


Example: Dynamically generated content

Use case: Map tiles
Problem: Need faster DB response time



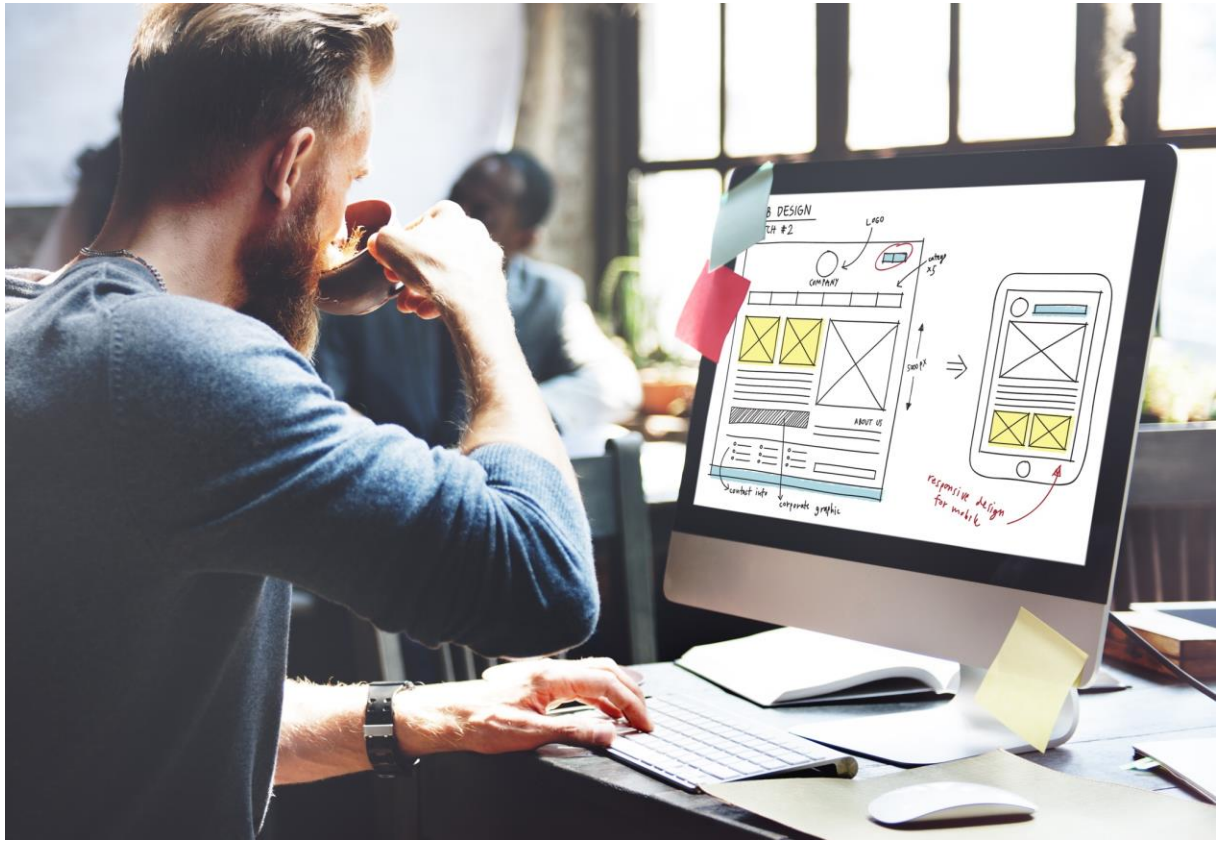
Example: DDoS mitigation



Section 3 key takeaways

- Amazon CloudFront is a global CDN service that accelerates the delivery of content, including static and video, to users with no minimum usage commitments.
- CloudFront uses a global network that comprises edge locations and regional edge caches to deliver content to your users.
- To use CloudFront to deliver your content, you specify an origin server and configure a CloudFront distribution. CloudFront assigns a domain name and sends your distribution's configuration to all of its edge locations.
- You can use Amazon CloudFront to improve the resilience of your applications that run on AWS from DDoS attacks.

Module 11 – Guided Lab: Streaming Dynamic Content Using Amazon CloudFront



Guided lab: Scenario

In this lab, you use Amazon Elastic Transcoder to convert a source video into multiple bitrates. You use Amazon CloudFront to deliver the dynamic, multiple bitrate stream to a connected device by using Apple HTTP Live Streaming (HLS) protocol.

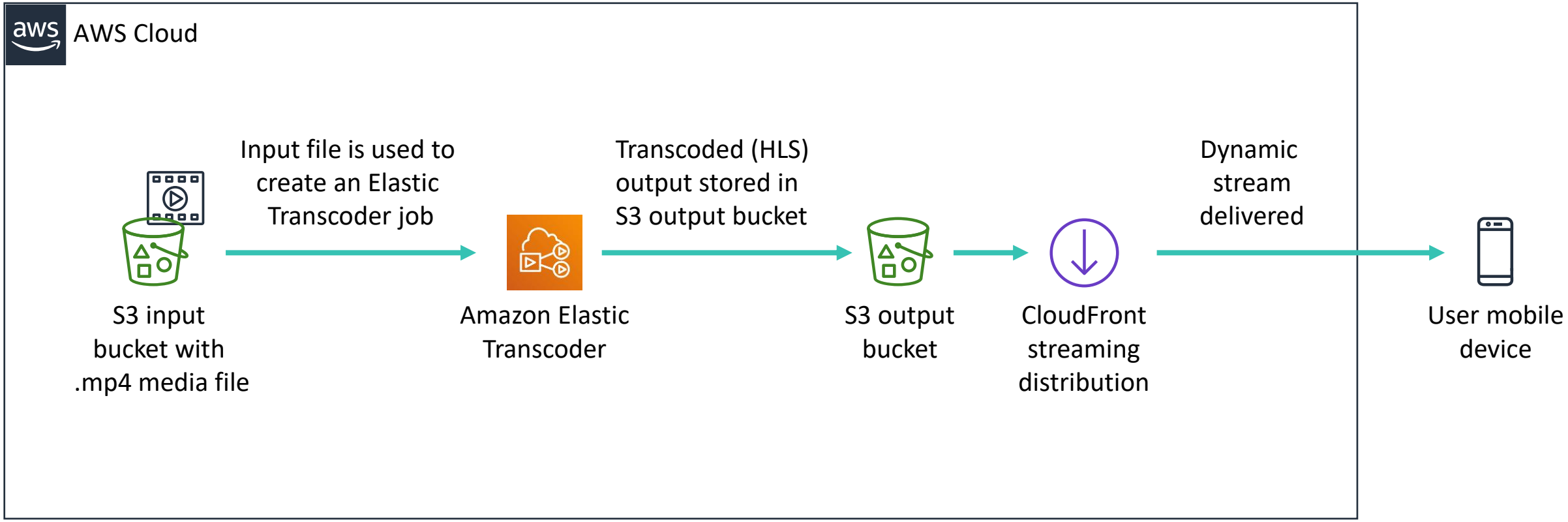
Amazon Elastic Transcoder

Amazon CloudFront

Guided lab: Tasks

1. Create an Amazon CloudFront distribution
2. Create an Amazon Elastic Transcoder pipeline
3. Test playback of the dynamic (multiple bitrate) stream

Guided lab: Final product



~ 30 minutes

Begin Module 11 – Guided Lab: Streaming Dynamic Content Using Amazon CloudFront

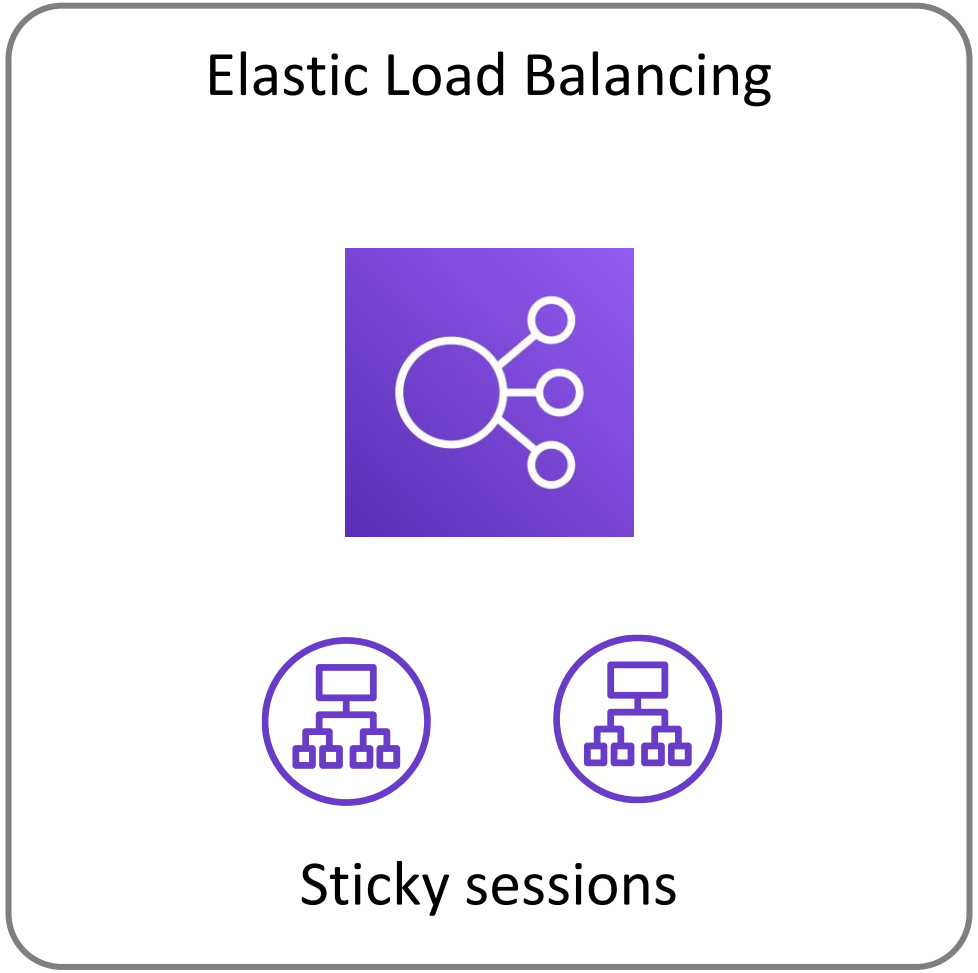
Guided lab debrief: Key takeaways

Module 11: Caching Content

Section 4: Caching web sessions

aws academy

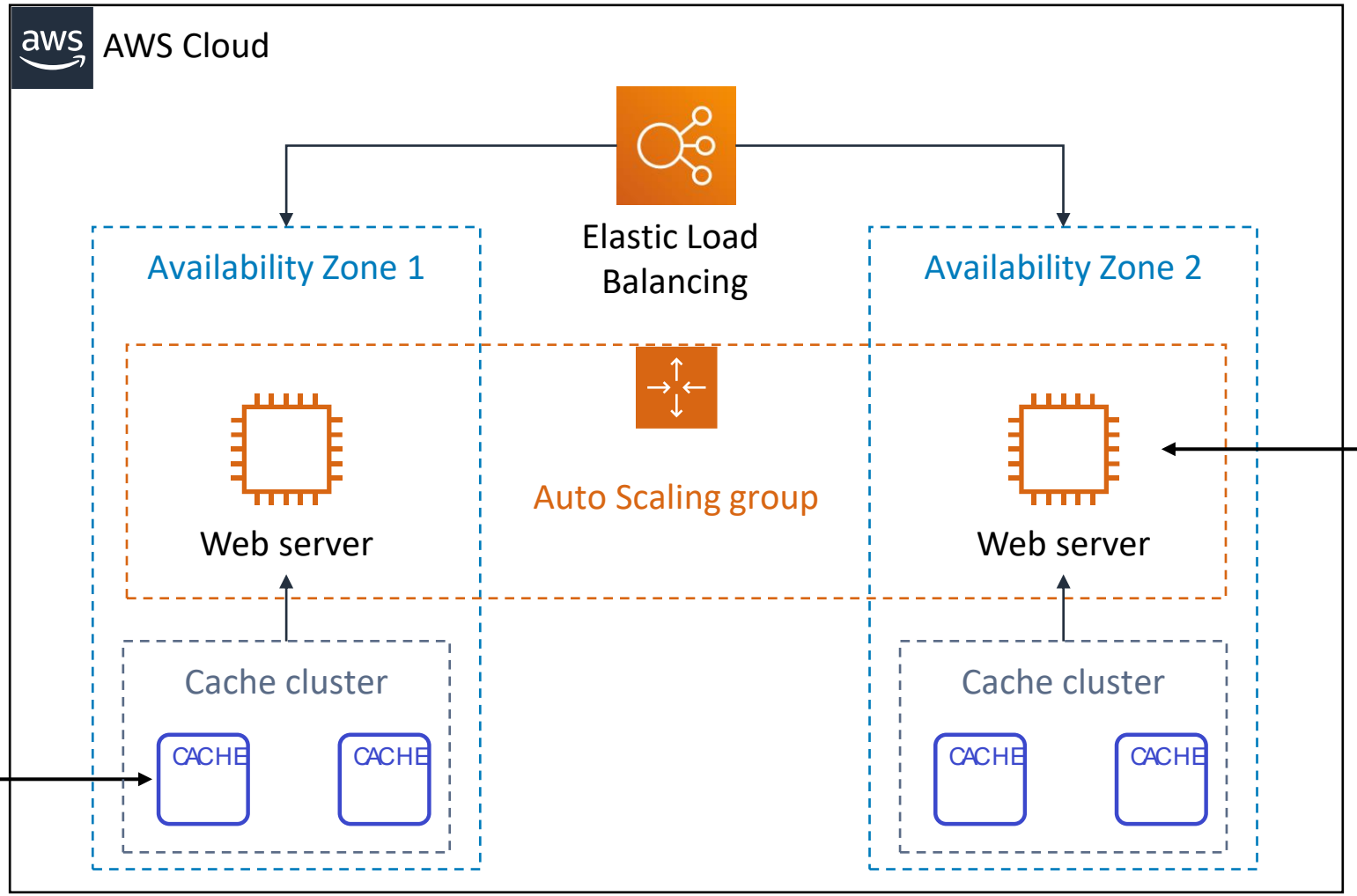
Session management: Sticky sessions



Feature that enables a load balancer to route a request to the specific server that manages the user's session.

- Use client-side cookies
- Are cost-effective
- Speed up retrieval of sessions
- Have disadvantages –
  - Loss of sessions when you have an instance failure
  - Limit scalability: Uneven load distribution and increased latency

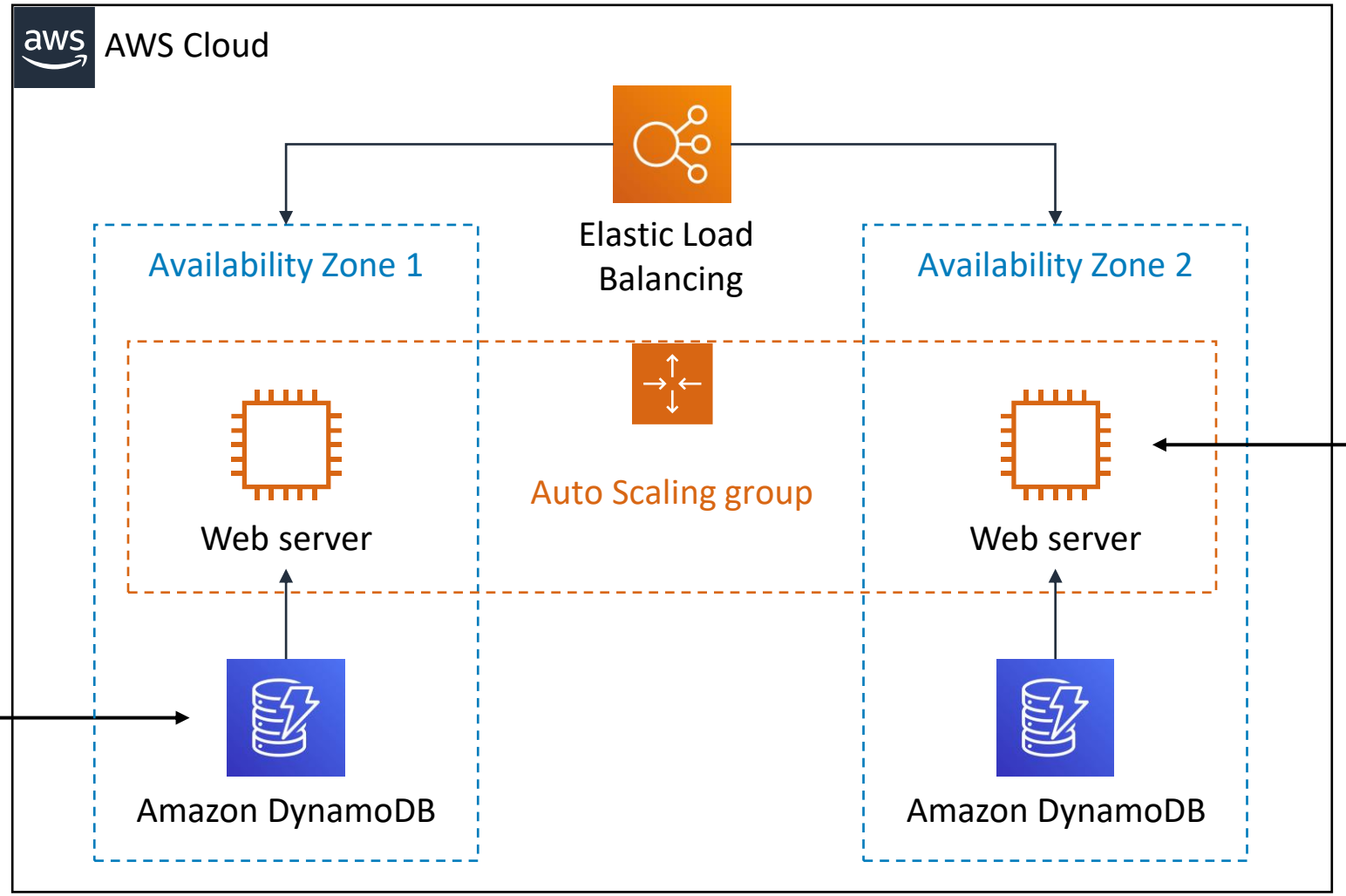
Instead of sticky sessions:
Persist sessions inside a distributed cache



Persist sessions in a distributed cache

Loss of sessions is a risk if sessions are stored locally on nodes

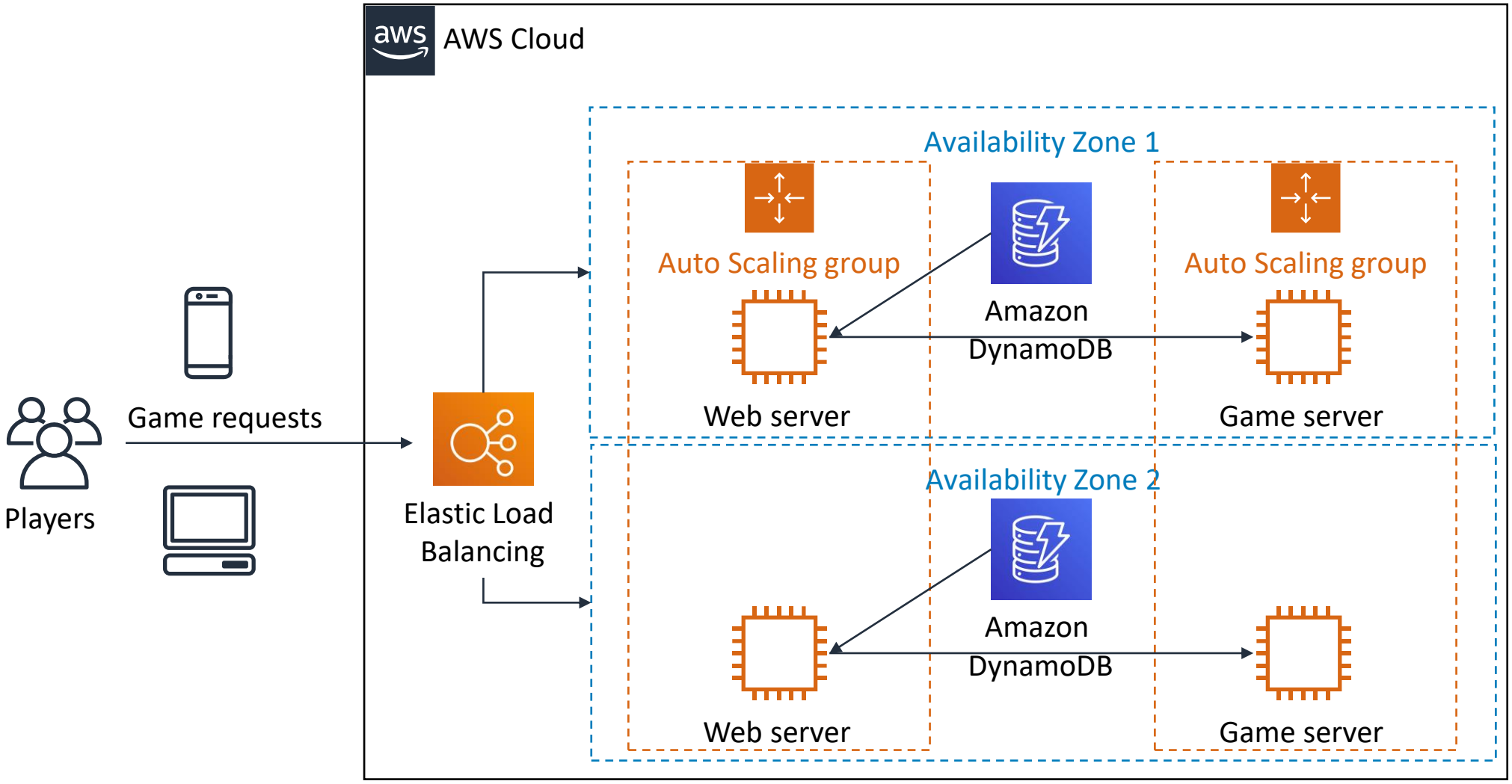
Instead of sticky sessions: Persist sessions inside a DynamoDB table



Persist sessions in an Amazon DynamoDB database

Loss of sessions is a risk if sessions are stored locally on nodes

Example: Storing session states for an online gaming application



Section 4 key takeaways

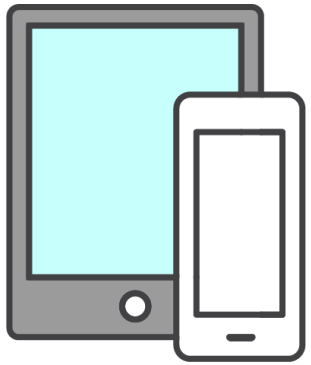
- Sessions are used to manage user authentication and store user data while the user interacts with the application.
- You can manage sessions with sticky sessions, which is a feature of Elastic Load Balancing load balancers. Sticky sessions route requests to the specific server that's managing the user's session.
- You can also manage sessions by persisting session data outside the web server instance—for example, in a distributed cache or DynamoDB table.

Module 11: Caching Content

Section 5: Caching databases

aws academy

When should you cache your database?



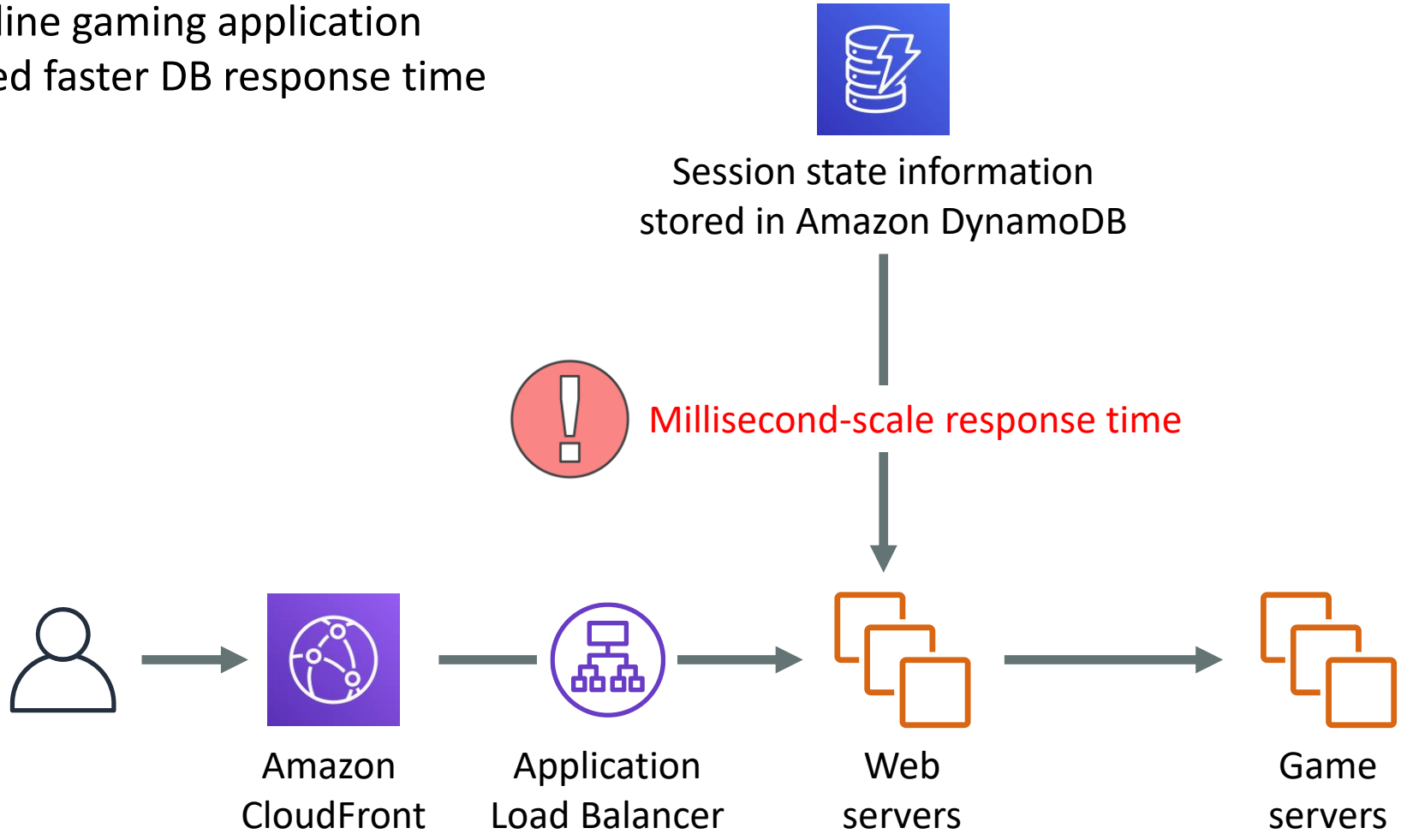
You are concerned about response times for your customer.

You have a high volume of requests that are inundating your database.

You would like to reduce your database costs.

Using DynamoDB for state information

Use case: Online gaming application
Problem: Need faster DB response time



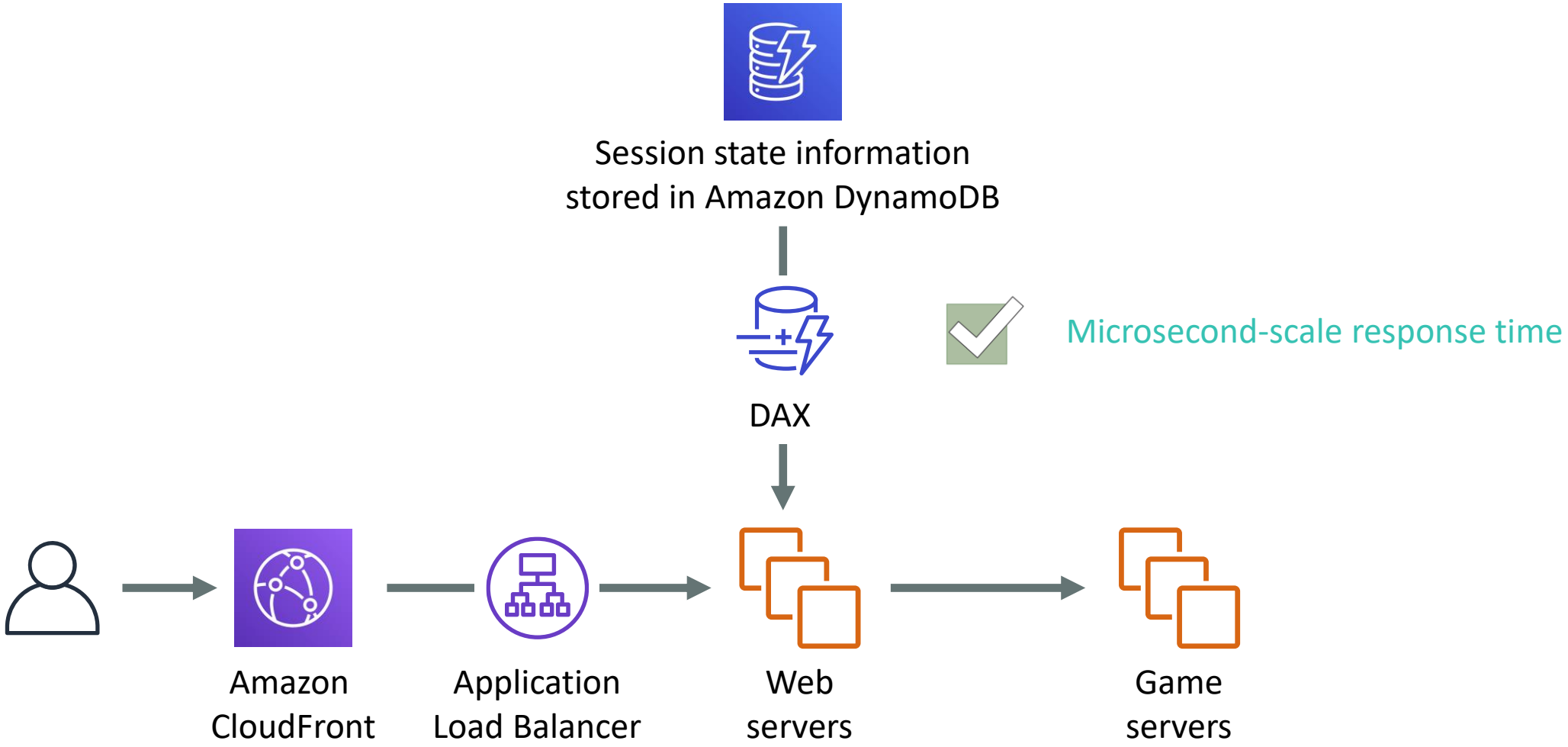
Amazon DynamoDB Accelerator

Amazon DynamoDB Accelerator

Fully managed, highly available, in-memory cache for DynamoDB

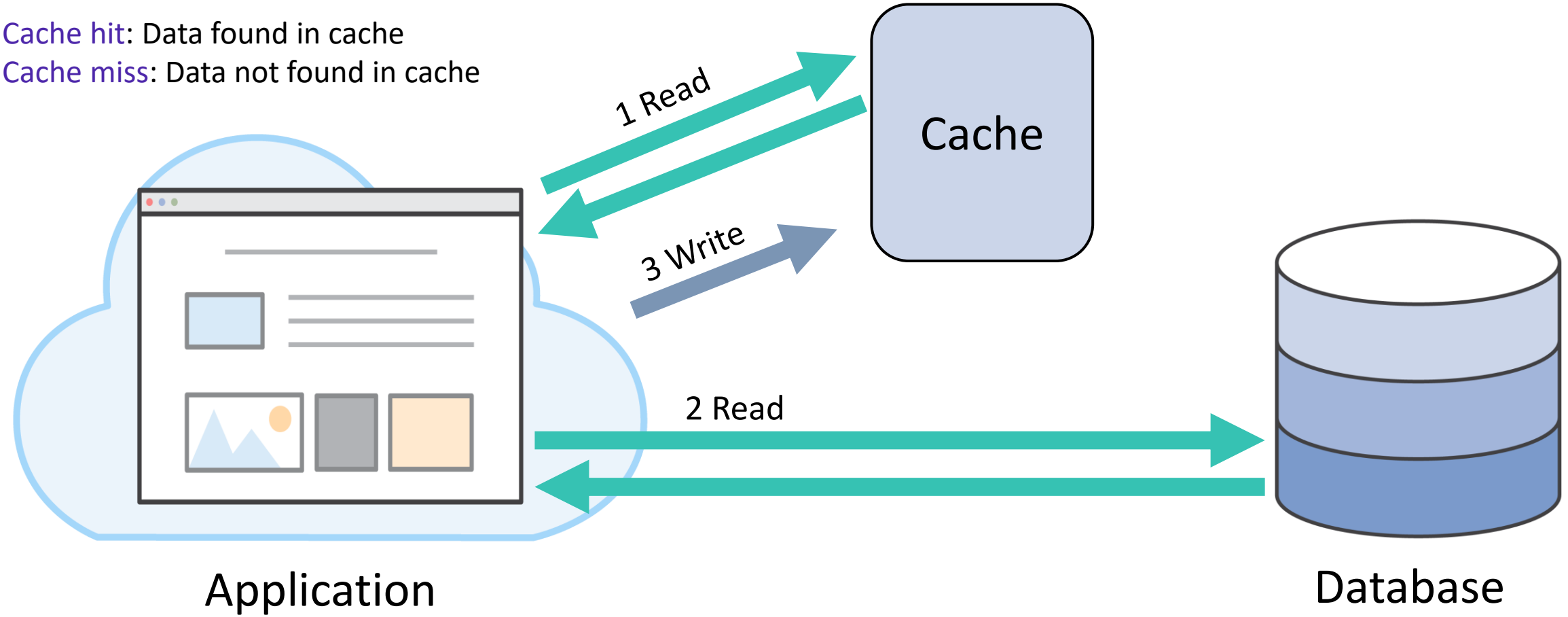
- Extreme performance (microsecond-scale response time)
- Highly scalable
- Fully managed
- Integrated with DynamoDB
- Flexible
- Secure

Using DynamoDB with DAX to accelerate response time



Remote or side caches

Cache hit: Data found in cache
Cache miss: Data not found in cache



Amazon ElastiCache

Amazon ElastiCache

ElastiCache provides web applications with an in-memory data store in the cloud.

- Works an in-memory data store and cache
- Offers high performance
- Is fully managed
- Is scalable
- Supports Redis and Memcached

Redis and Memcached

Amazon
ElastiCache

ElastiCache
for Memcached

Scales up to 20 nodes
per cluster

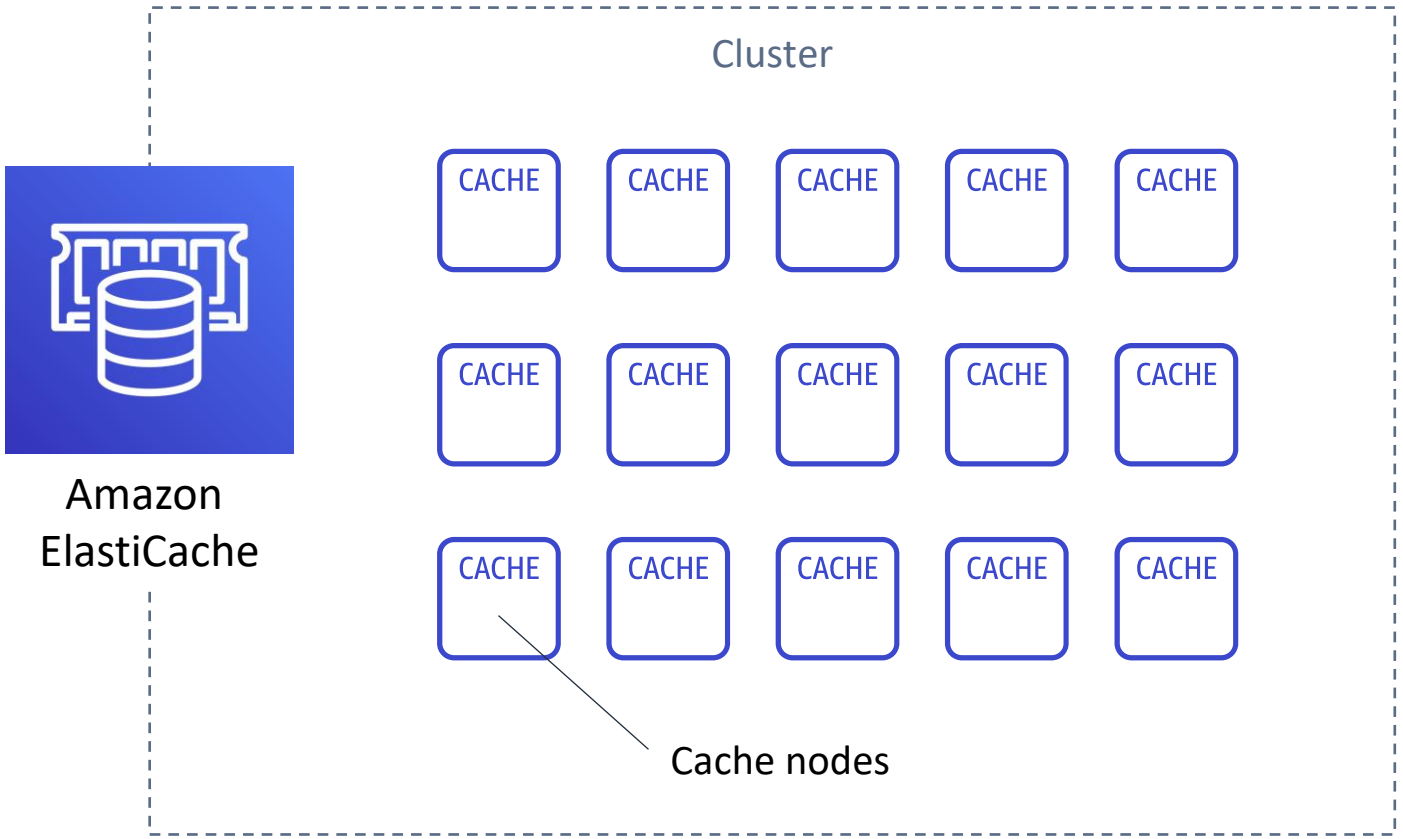
ElastiCache
for Redis

Scales up to 250 nodes
per cluster

Memcached versus Redis comparison

Feature	Memcached	Redis
Sub-millisecond latency	Yes	Yes
Ability to scale horizontally for writes and storage	Yes	No
Multi-threaded performance	Yes	No
Advanced data structures	No	Yes
Sorting and ranking datasets	No	Yes
Publish/subscribe messaging	No	Yes
Multi-AZ deployments with automatic failover	No	Yes
Persistence	No	Yes

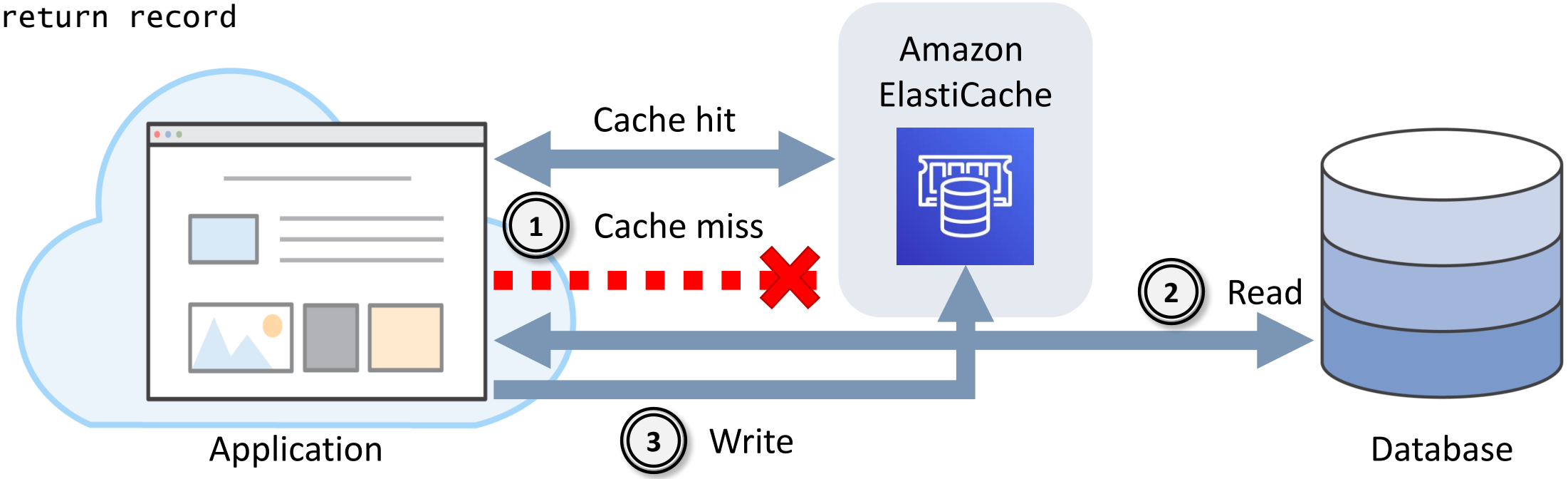
ElastiCache components



- A node is the smallest block of an ElastiCache deployment
- Each node has its own DNS name and port
- A cluster is a logical grouping of one or more nodes

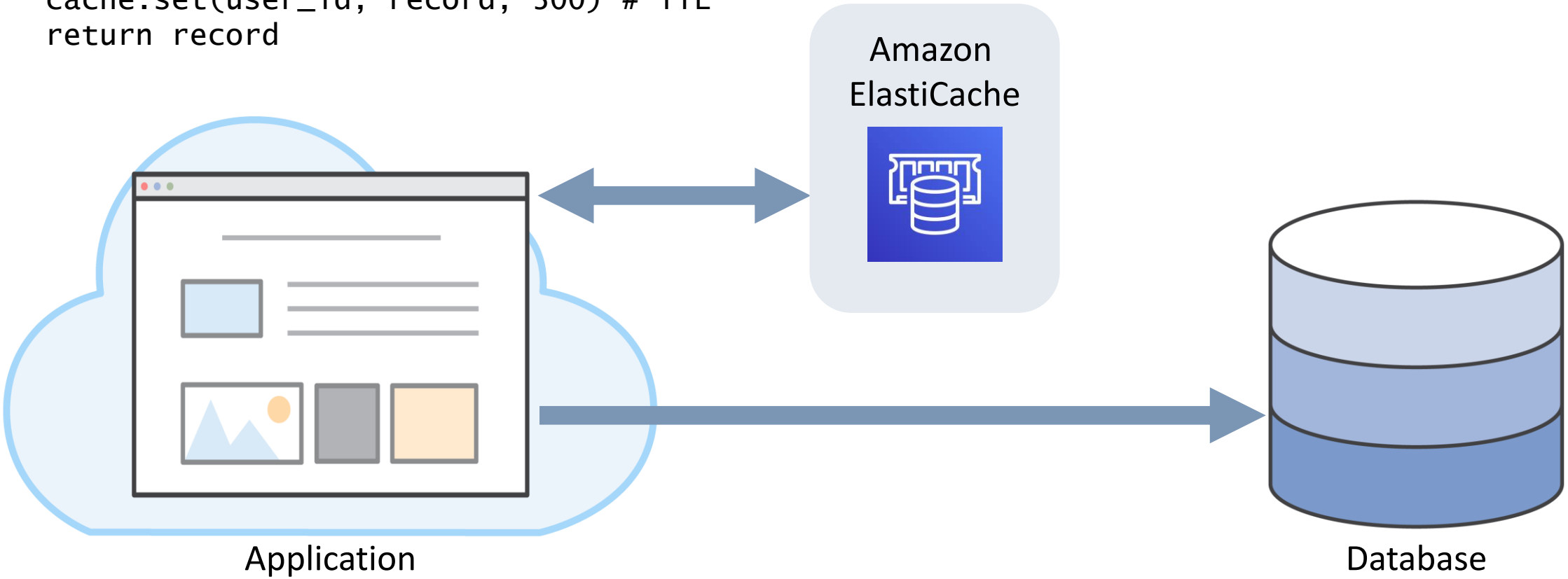
Caching strategies: Lazy loading

```
def get_user(user_id):
    # Check the cache
    record = cache.get(user_id)
    if record is None:
        # Run a DB query
        record = db.query("select * from users where id = ?", user_id)
        # Populate the cache
        cache.set(user_id, record)
    return record
```

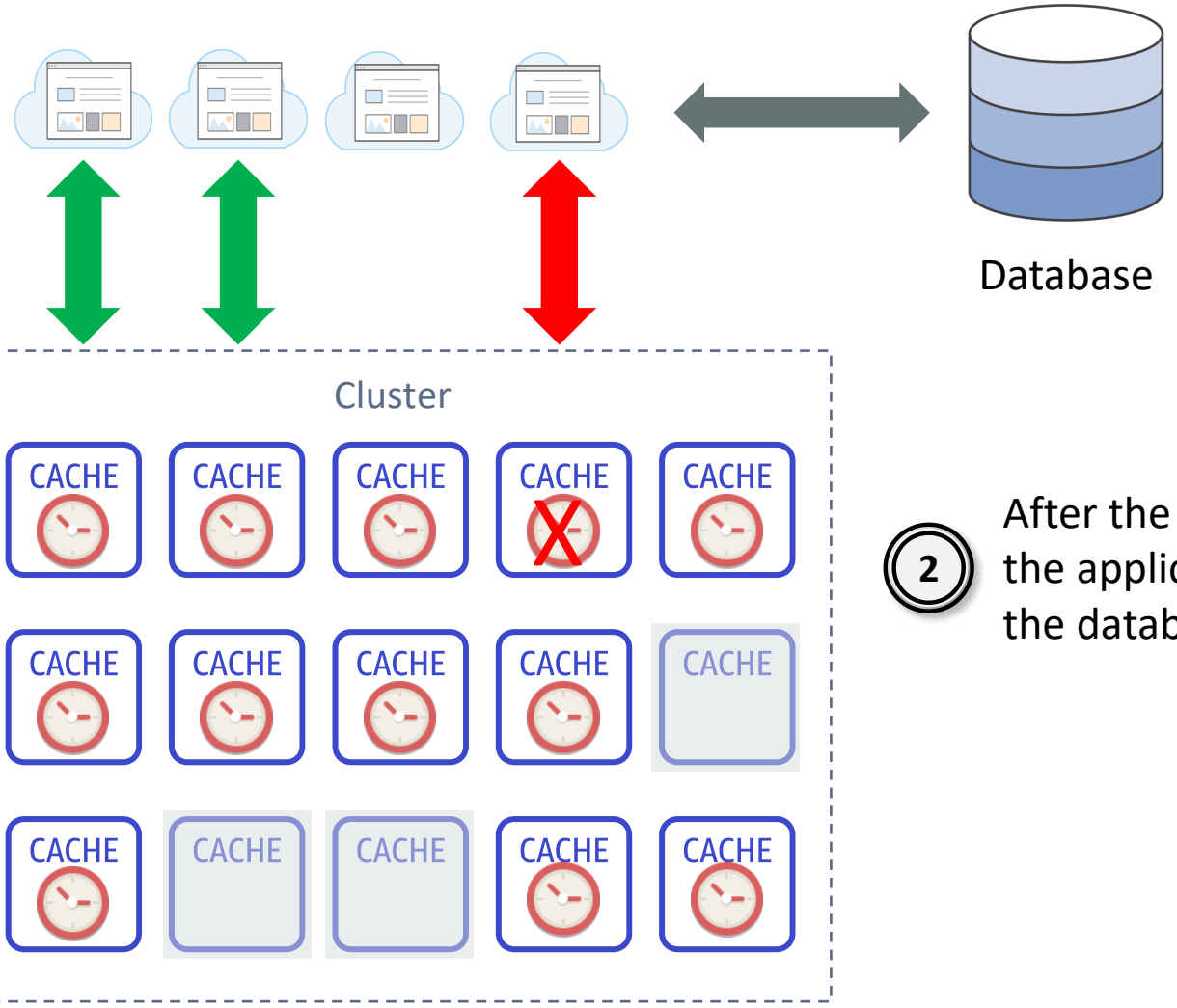


Caching strategies: Write-through

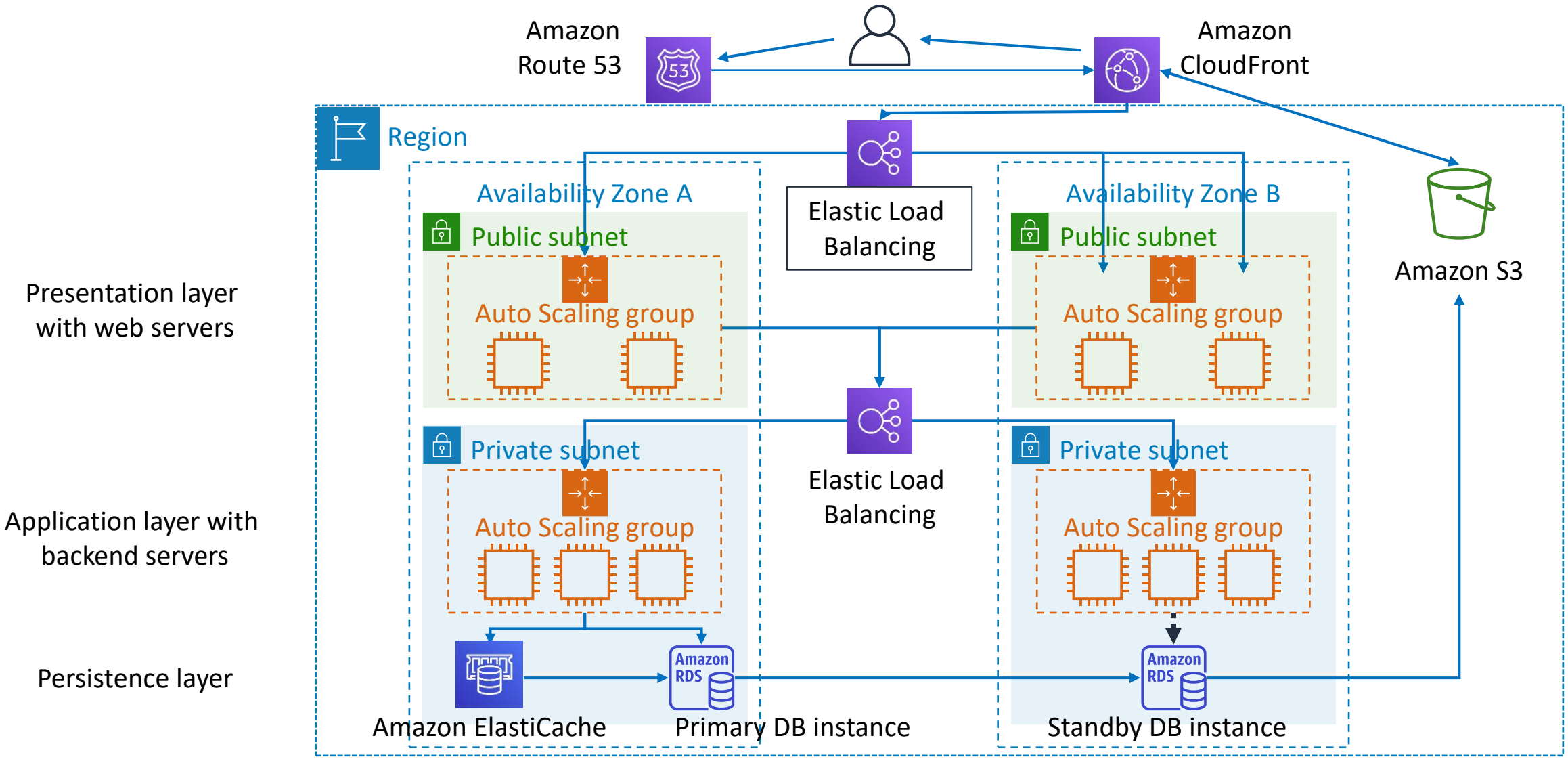
```
def save_user(user_id, values):
    # Save to DB
    record = db.query("update users…where id = ?", user_id, values)
    # Push into cache
    cache.set(user_id, record, 300) # TTL
    return record
```



Adding TTL



Three-tier web hosting architecture



Section 5 key takeaways

- A database cache supplements your primary database by removing unnecessary pressure on it, typically in the form of frequently accessed read data
- DAX is a fully managed, highly available, in-memory cache for DynamoDB that delivers a performance improvement of up to 10 times—from milliseconds to microseconds
- Amazon ElastiCache is a side cache that works as an in-memory data store to support the most demanding applications that require sub-millisecond response times

Module 11: Caching Content

Module wrap-up

Module summary

In summary, in this module, you learned how to:

- Identify how caching content can improve application performance and reduce latency
- Create architectures that use Amazon CloudFront to cache content
- Identify how to design architectures that use edge locations for distribution and distributed denial of service (DDoS) protection
- Recognize how session management relates to caching
- Describe how to design architectures that use Amazon ElastiCache

Complete the knowledge check

Sample exam question

A company is developing a highly available web application that uses stateless web servers. Which services are suitable for storing session state data? (Select TWO.)

A. Amazon CloudWatch
B. Amazon DynamoDB
C. Elastic Load Balancing
D. Amazon ElastiCache
E. AWS Storage Gateway

Thank you

aws academy