Serverless Geospatial Mobile Apps with AWS

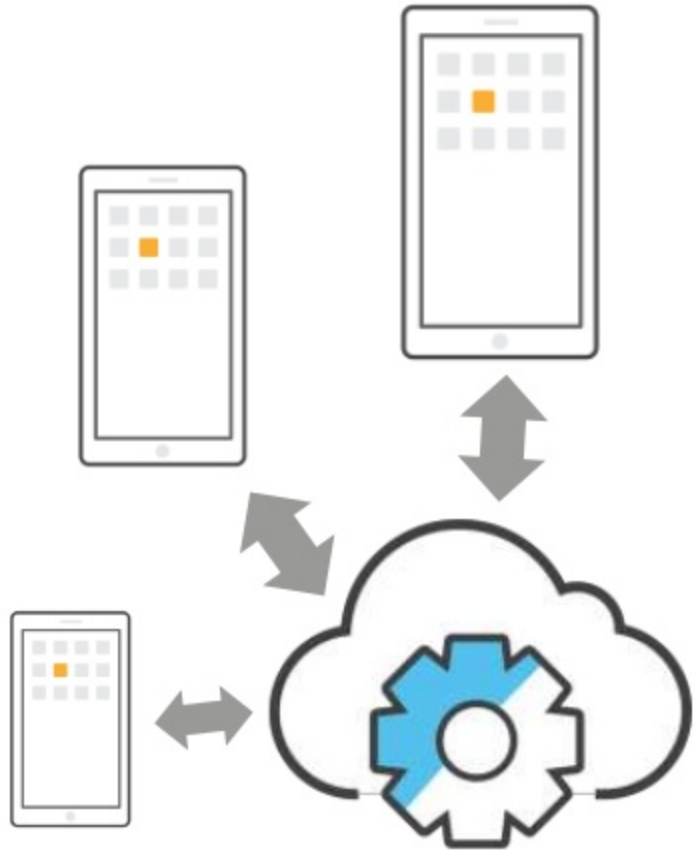
John Chang,
Ecosystem Solutions Architect
Amazon Web Services
September 2016

amazon web services

Trend: Cloud backend for mobile apps

Why Cloud Backend?

- React in the **background** onto events and user-generated content
- Connect **multiple** app users together through common logic
- Share functionality **across** multiple platforms
- Make seamless **code modifications** without pushing out new versions to the app store
- Scale!

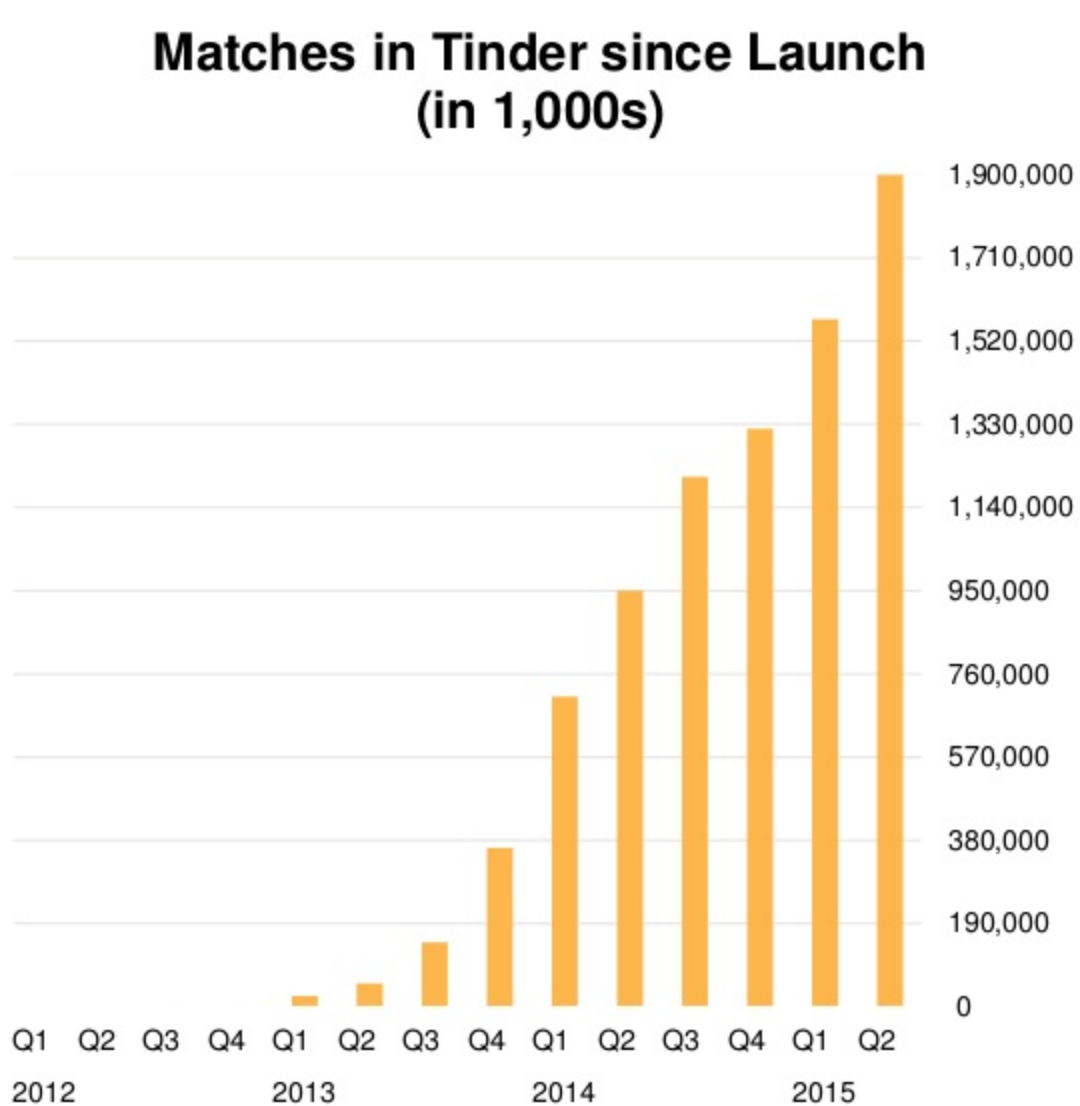


Back end architecture on AWS

tinder™


CP&G 3G
4:20 PM
Christopher
32
Distance: Within 1 mile
Last Checkin: 2 hours ago
Shared Interests
Shared Friends
AT&T
5:00 PM
100%
tinder
It's a Match!
You and Sarah have liked each other.
Send a Message
or
Keep Playing
CP&G 3G
4:20 PM
tinder
Jennifer, 23
4
25
8

tinder™


Matches in Tinder since Launch
(in 1,000s)
1,900,000
1,710,000
1,520,000
1,330,000
1,140,000
950,000
760,000
570,000
380,000
190,000
0
Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4 Q1 Q2
2012
2013
2014
2015

1.7 Billion

swipes a day

October 2015

7 Petabytes

of Data Transfer per Month

October 2015

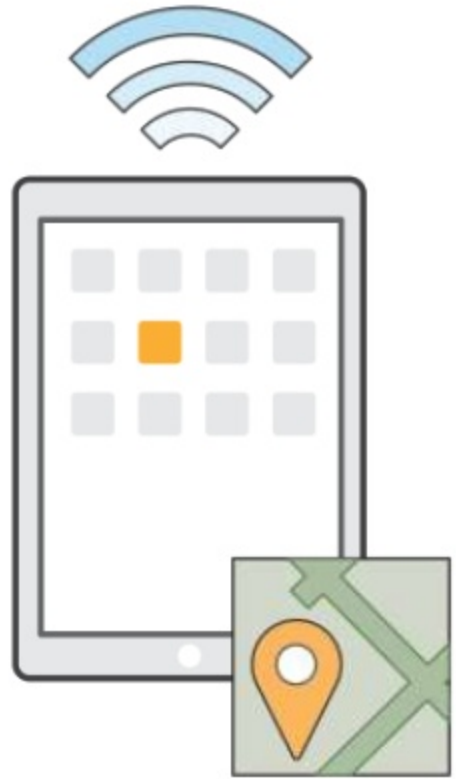
1.2 Trillion+

Total Swipes

October 2015

Sample app: “Find-a-Like”

- **Premise:** Create a profile with interests and get notified when like-minded users are nearby
- **Functionalities:**
  - Create a profile with interests and upload content
  - Track location continuously
  - Notify when users with similar interests are close by
  - Log and analyze app usage



Let’s think in layers

1.  Create profile, upload content, and track usage 

App-centric
“You”

2.  Track location and user interests 

Activity-centric
“What you do”

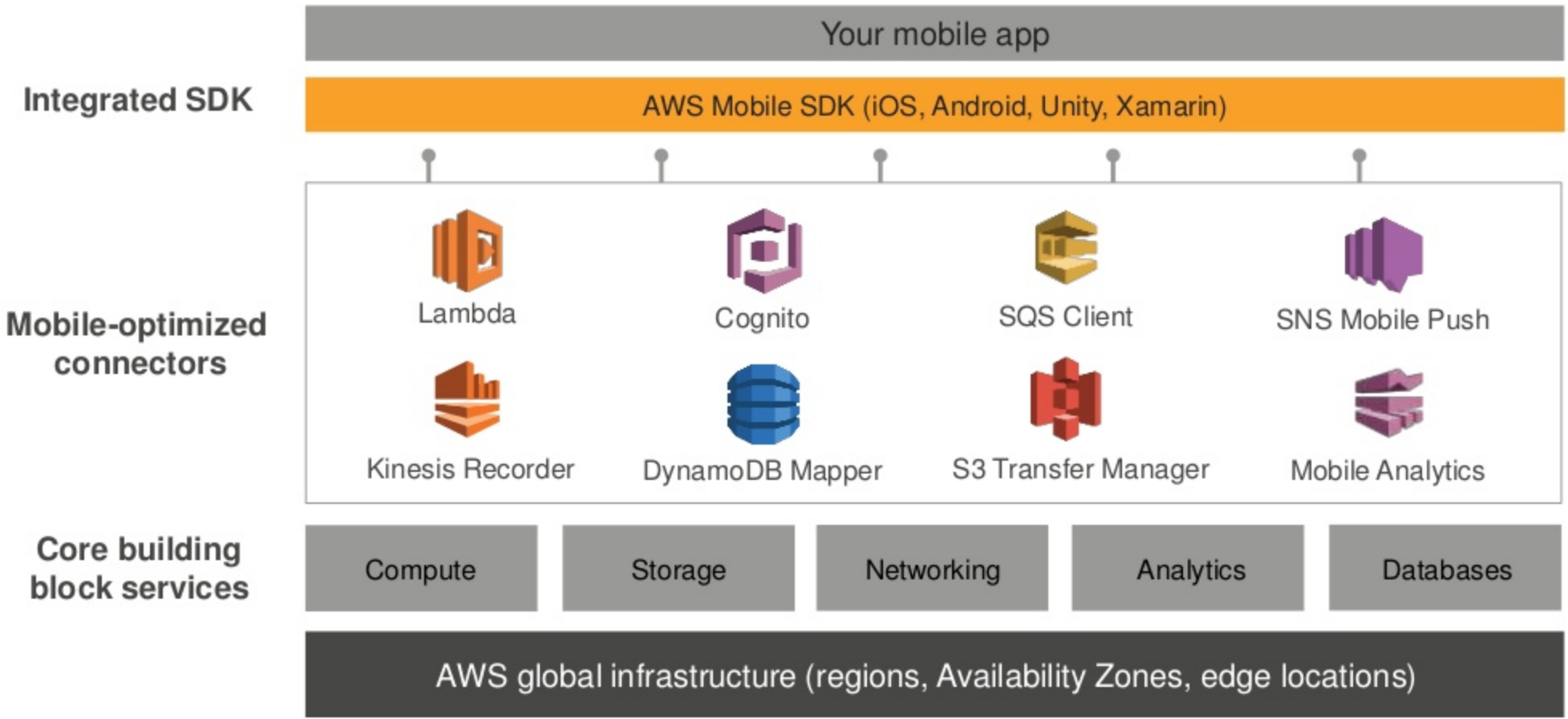
3.  Match and alert users 

User base-centric
“Them & me”

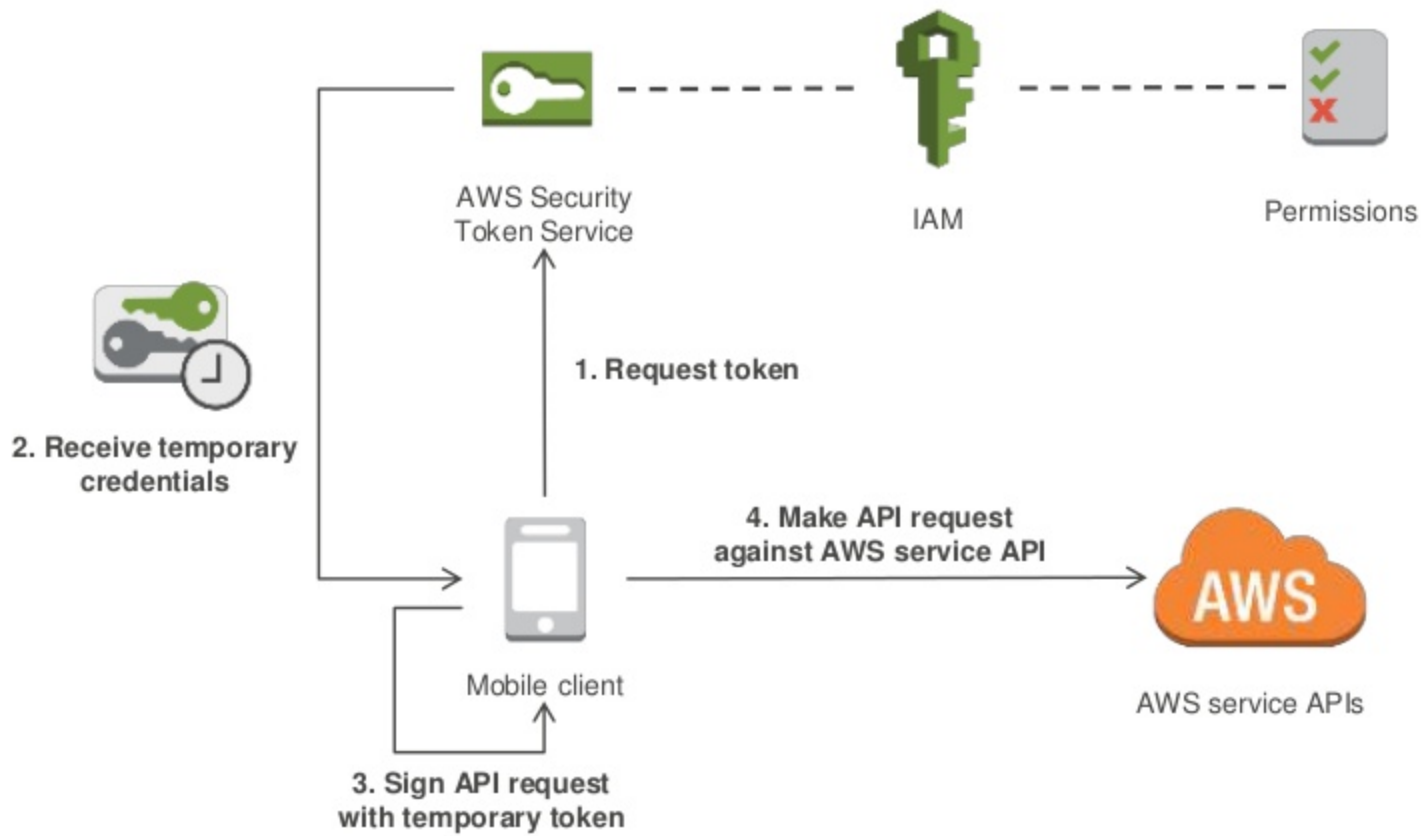
1

Create a profile, upload content, and track usage

AWS Mobile SDKs



Security model for AWS API calls



Authenticate your user: Amazon Cognito

- Generate **temporary credentials** and enforce rotation to limit **credential lifetime**
- Authenticate user through third-party **authentication provider**
- Unique users across **multiple** devices and identity providers
- Allows **anonymous** user access
- Enables security best practices through **IAM roles**

Use Cognito for authentication on iOS

```
//Create and configure Cognito credentials provider
AWSCognitoCredentialsProvider *credentialsP = [AWSCognitoCredentialsProvider
    credentialsWithRegionType:AWSRegionUSEast1
    accountId:@"0123456789"
    identityPoolId:@"us-east-1:beeeeeef-beef-beef-beef-beef"
    unauthRoleArn:@"arn:aws:iam::0123456789:role/Unauth"
    authRoleArn:@"arn:aws:iam::0123456789:role/Auth"];

//Set Cognito as default credentials provider for all AWS service calls
AWSServiceConfiguration *configuration =  [AWSServiceConfiguration
    configurationWithRegion:AWSRegionUSEast1
    credentialsProvider:credentialsP];

[AWSServiceManager defaultServiceManager].defaultServiceConfiguration =
configuration;
```

Create your profile: Cognito Sync

- Create your app profile and save it **locally** in the Cognito data store
- Cognito will **synchronize** the data sets across all your user's devices
- Cognito data sets are **key/value** pairs

```
AWSCognito *syncClient = [AWSCognito defaultCognito];
AWSCognitoDataset *subs = [syncClient
openOrCreateDataset:@"UserProfile"];
[dataset setString:"Oli" forKey:@"name"];
[dataset setString:"50km" forKey:@"interestRadius"];
[dataset synchronize];
```

Upload a profile picture: Amazon S3

- Highly available object storage
- Designed for **99.999999999%** annual data durability
- Replicated across **3 facilities**
- Virtually **unlimited scale**
- Pay only for what you use, you **don't need to pre-provision**
- Allows **event notifications** to trigger further action

Amazon S3

Upload a profile picture: S3 Transfer Utility

- Amazon S3 to store and share UGC **directly** from the mobile device
- S3 Transfer Utility provides:
  - Ability to continue transferring data **in the background** when your app is not running
  - Ability to upload **binary data** instead of having to first save it as a file on the device

Amazon S3

S3 Transfer Utility: iOS code

```
NSData *dataToUpload = // The data to upload

AWSS3TransferUtility *transferUtility = [AWSS3TransferUtility
defaultS3TransferUtility];
[[transferUtility uploadData:dataToUpload
                      bucket:@"YourBucketName"
                         key:@"YourObjectKeyName"
                 contentType:@"text/plain"
                  expression:expression
            completionHander:completionHandler]
continueWithBlock:^id(AWSTask *task) {
    if (task.result) {
        AWSS3TransferUtilityUploadTask *uploadTask = task.result;
        // Do something with uploadTask
    }
}
```

Track app usage: Amazon Mobile Analytics

- Allows you to **collect, visualize, and understand** your mobile app usage
- Scales seamlessly to **billions** of events per day
- You retain **full control** and **ownership** of the data

```
AWSMobileAnalytics *analytics =
[AWSMobileAnalytics
    mobileAnalyticsForAppId:@"yourAppId"
    identityPoolId: @"cognitoId"];
```

Amazon Mobile Analytics | Demo App | Filters: 06/11/14 - 06/17/14 | All Platforms

Overview | Active Users | Sessions | Revenue | Retention | Custom Events

Lifetime Users

505,616 iOS: 84,547 Android: 382,456 Fire OS: 38,613

Lifetime Value Per User

$0.65 USD iOS: $2.15 USD Android: $0.38 USD Fire OS: $0.05 USD

Daily Active Users (DAU)

Avg. **2,914** | Change ▲4.7%

Monthly Active Users (MAU)

Avg. **5,577** | Change ▲0.6%

New Users

Avg. **1,108** | Change ▲28.1%

Sticky Factor

Wt. Avg. **0.52** | Change ▲4.1%

Total Sessions

Avg. **4,112** | Change ▲5.7%

Day 1 Retention

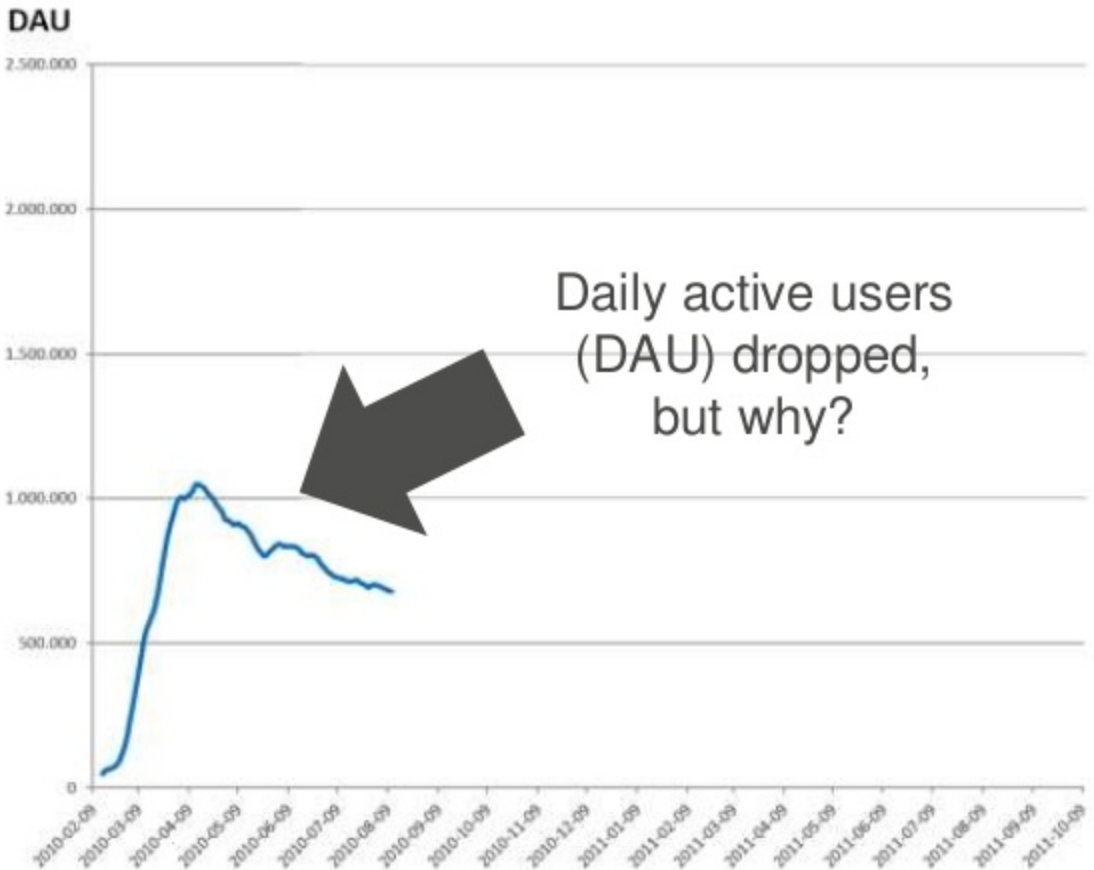
Avg. **140** | Change ▲2.9%

User Retention: Bubble Island

Case Study

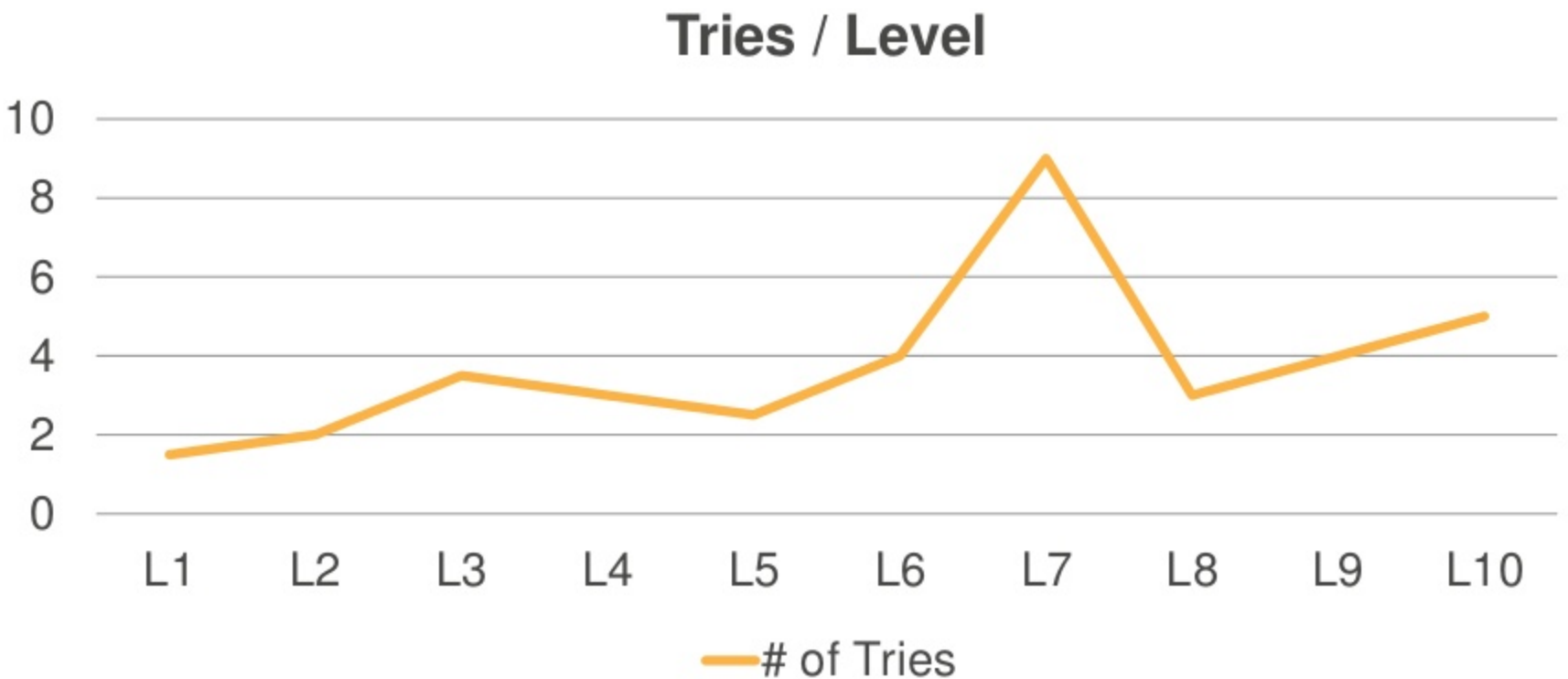
User Retention: Bubble Island

Case Study



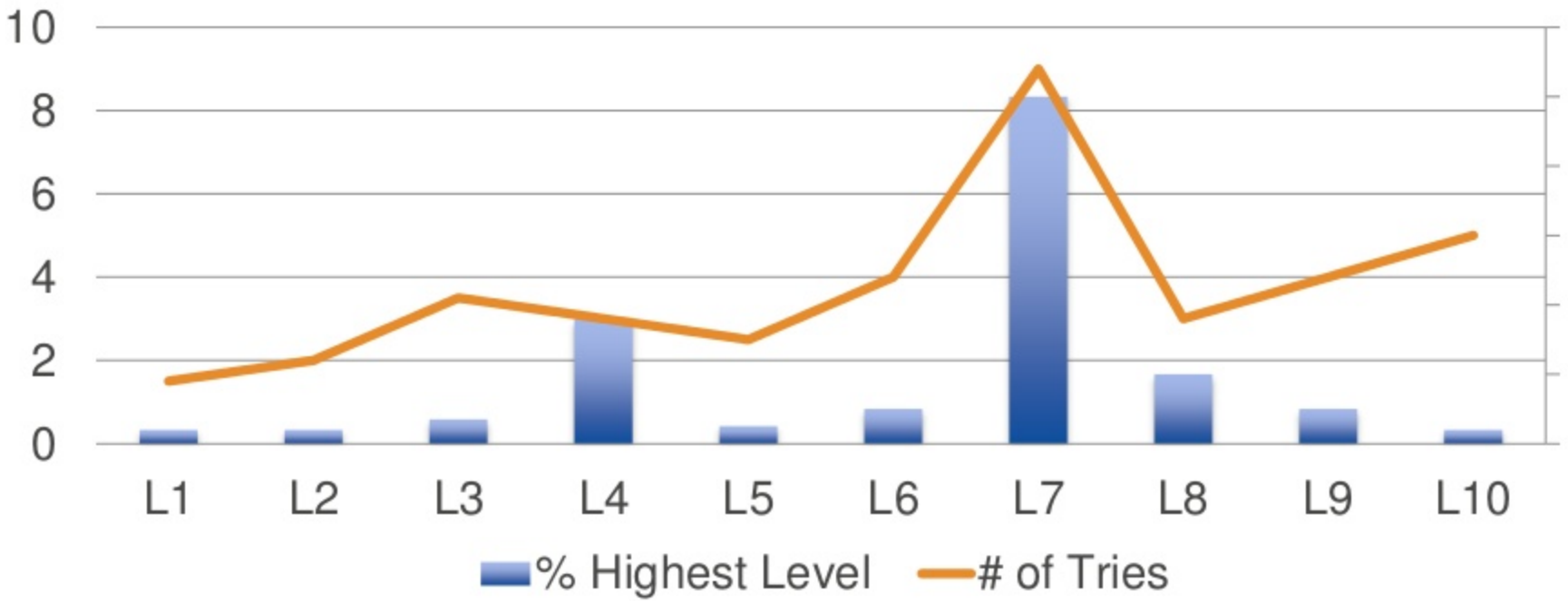
Level Progression (One Metric)

Case Study



Level Progression (Two Metrics)

Case Study



User Retention: Bubble Island

Case Study

After a few weeks of playing:

All levels were played
OR
Level became too difficult

Let’s think in layers

1

Create profile, upload content, and track usage

App-centric
“You”

2

Track location and user interests

Activity-centric
“What you do”

3

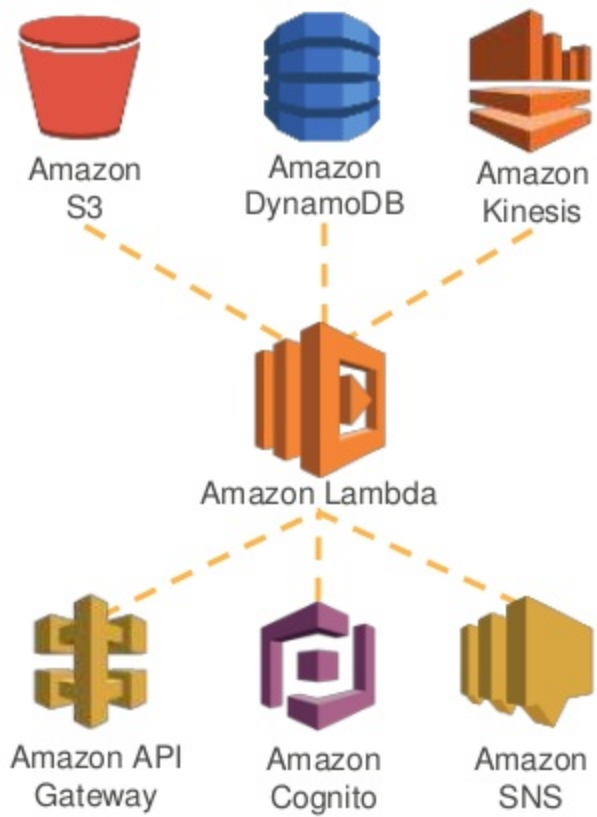
Match and alert users

User base-centric
“Them & me”

Track location and user interests

AWS Lambda

- Run your arbitrary cloud functions using **AWS Lambda**
- Triggered through **invocation** from the mobile app or **state changes** in your setup
- Executed immediately and scales automatically to **match** the incoming request rate
- Sub second granular billing based on **execution time**



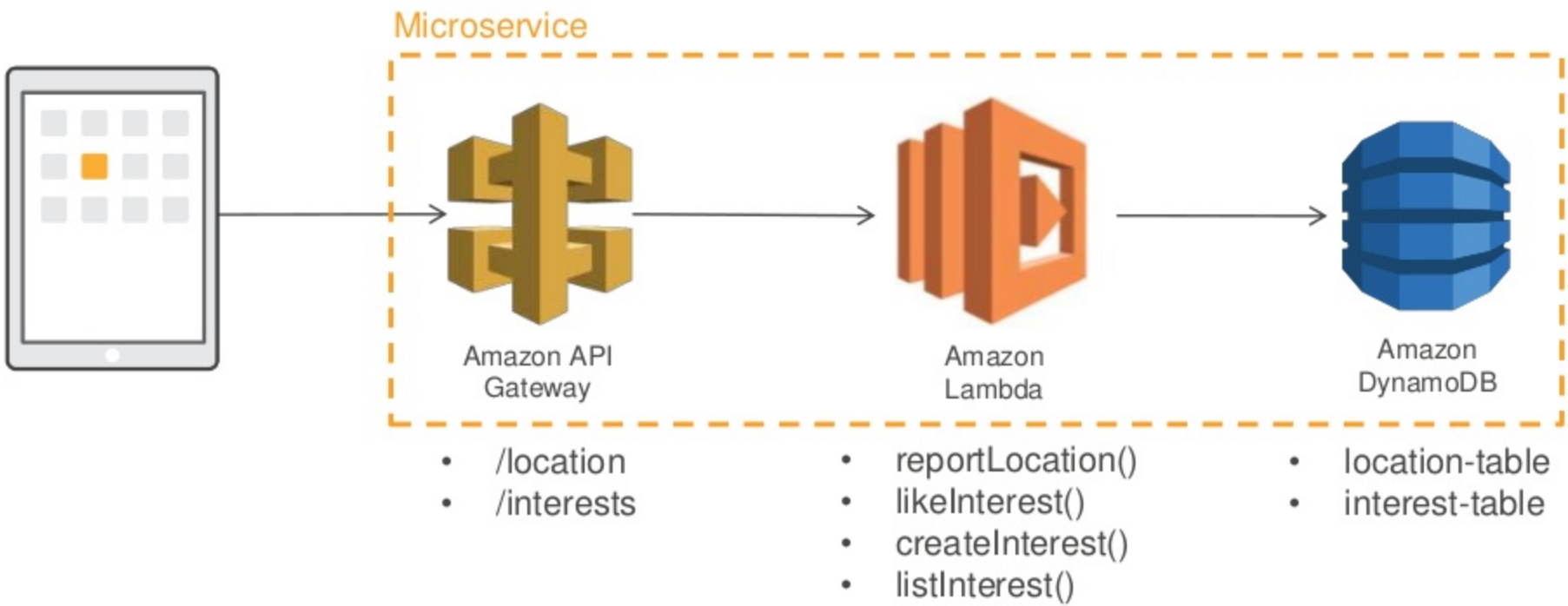
How to collect location and interests?

Mobile

Back-end logic

Database

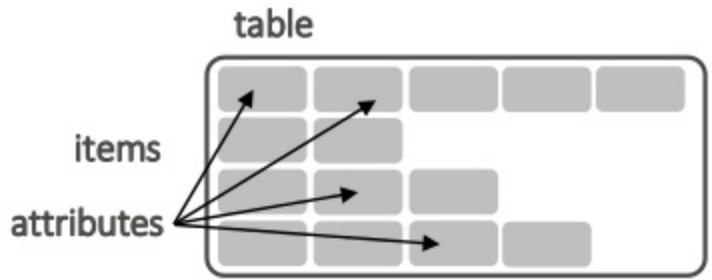
“Location Tracker” and “Interest” microservice



Amazon DynamoDB

Fully Managed NoSQL Database Service

- **Schemaless** Data Model
- Seamless **scalability**
- **No** storage or throughput limits
- Consistent **low latency** performance
- High **durability** and availability
- Replicated across **3 facilities**



SHAZAM

Search Music

Experience more

Discover, explore and share more music, TV shows and ads you love

Get Shazam now

SHAZAM FOR YOUR

- iPhone & iPod touch
- Android
- Windows Phone
- Windows 8
- BlackBerry
- iPad
- Other devices

GET SHAZAM | SHAZAM MUSIC | CAREERS

500,000 writes / second to their Amazon DynamoDB tables

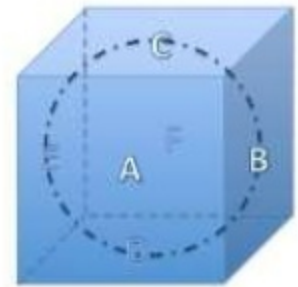
Concepts first: Geohash

GeoHash is a lat/long **geocode system** that subdivides space into buckets on a grid.

Can be numerical e.g. *6093522776912656819*

Works with DynamoDB!

How does it work?



Divide the planet earth into six cells (A,B,C,D,E,F) like the six faces of a cube.

Divide each cell into child cells, and divide child cells into more child cells. The red dot here would thus be A224.

Geo library for Amazon DynamoDB

- Java library to easily **create** and **query** **geospatial** data in DynamoDB using GeoHashes

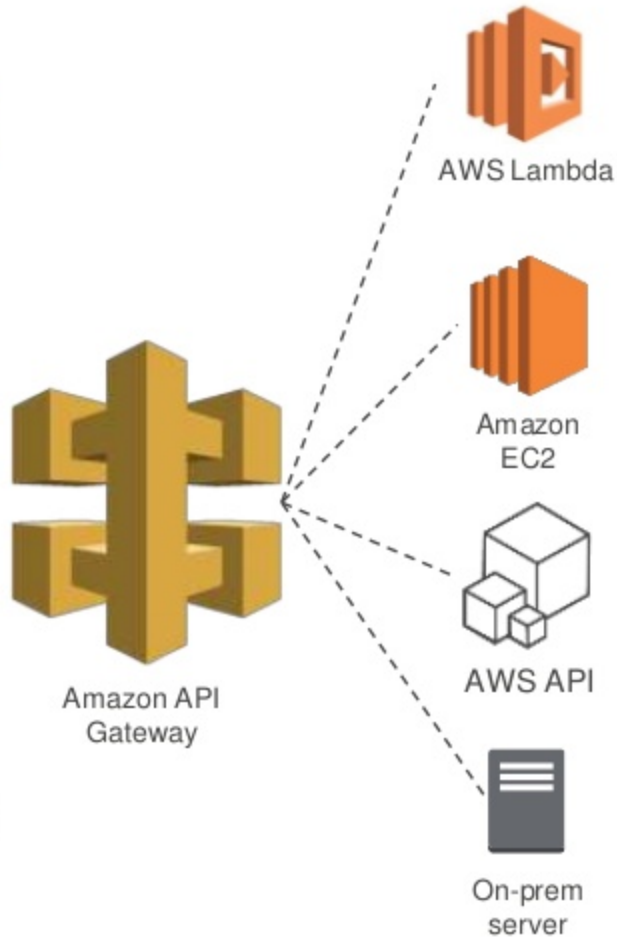
Works with Lambda!

```
GeoPoint point = new GeoPoint(47.62, -122.34);

// find places 250m of Seattle's Space Needle
QueryRadiusRequest request = new
     QueryRadiusRequest(point, 250);

QueryRadiusResult result =
     geoDataManager.queryRadius(request);
```

https://github.com/awslabs/dynamodb-geo

Amazon API Gateway

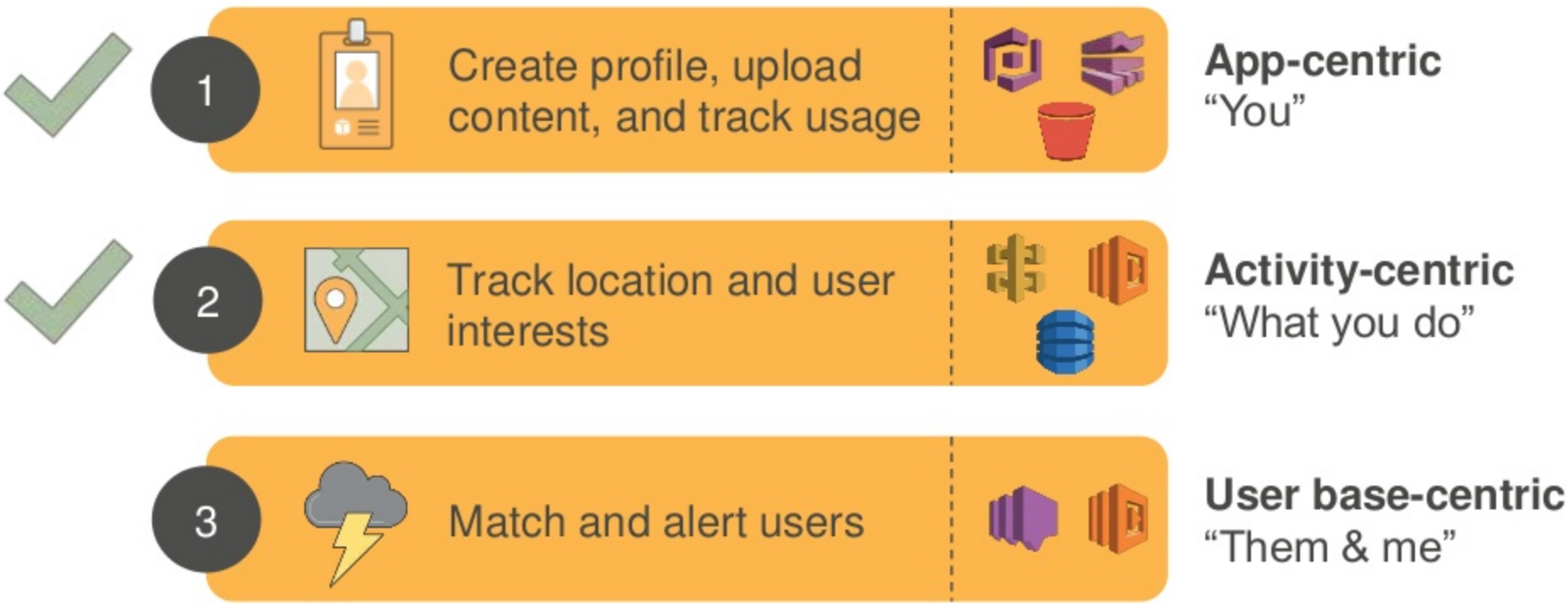
- Fully managed and scalable **RESTful** API gateway service
- Powered through our **content delivery network** via our 55 global edge locations
- Provides **DDoS** protection and **throttling** capabilities
- Multiple **API stages** which you define (e.g. dev, test, prod)



When to choose API Gateway vs. Direct SDK?

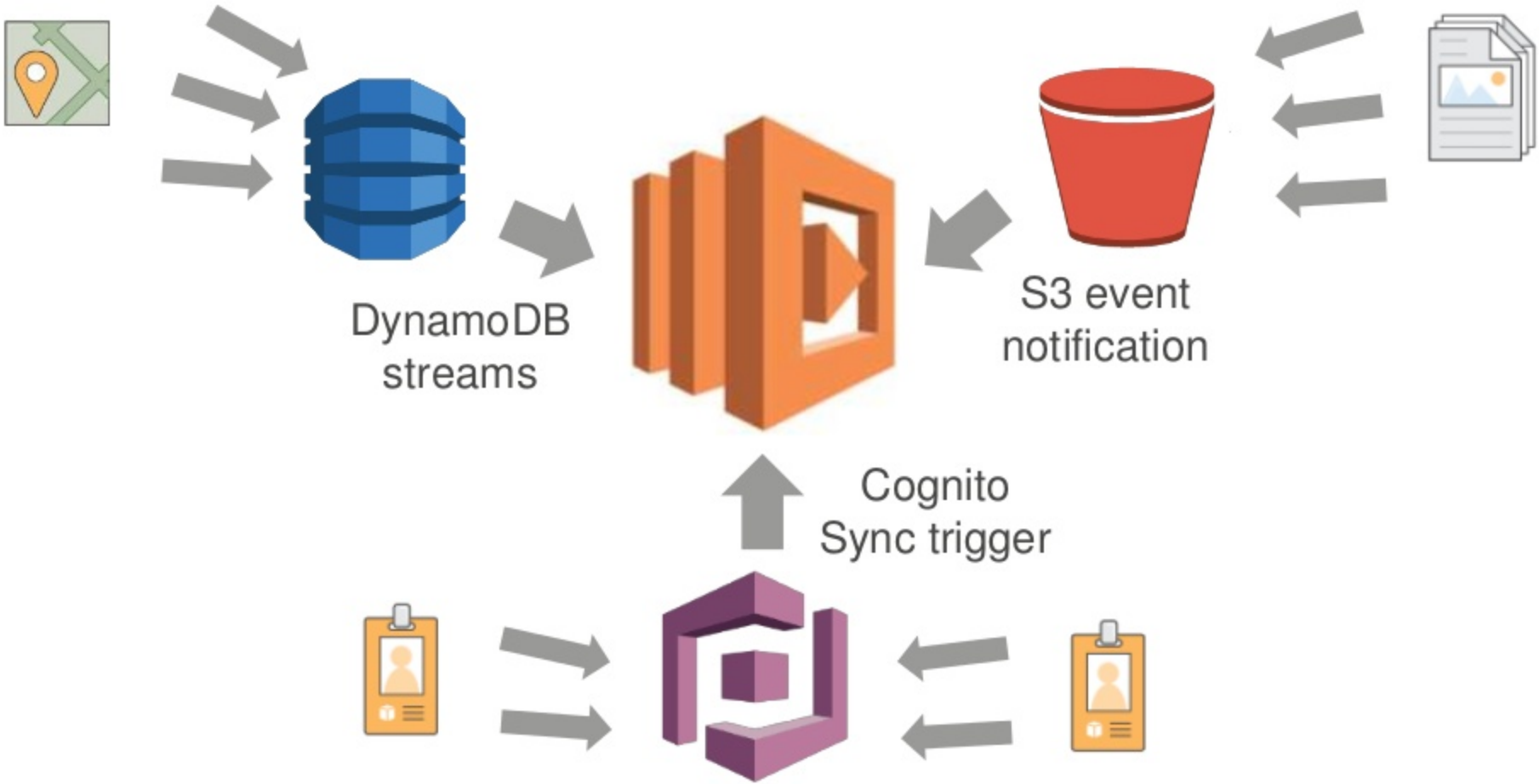
- Amazon API Gateway adds an additional layer between your mobile users and your logic and data stores in order to:
  - Allow back-end logic to be **interchanged** without mobile app code modifications
  - Ability to **throttle** individual users or requests
  - Protect against **DDoS attacks** including counterfeit requests (Layer 7) and SYN floods (Layer 3)
  - Provide a **caching layer** for your calls
  - Enables **CORS** for all AWS service for web apps

Let’s think in layers

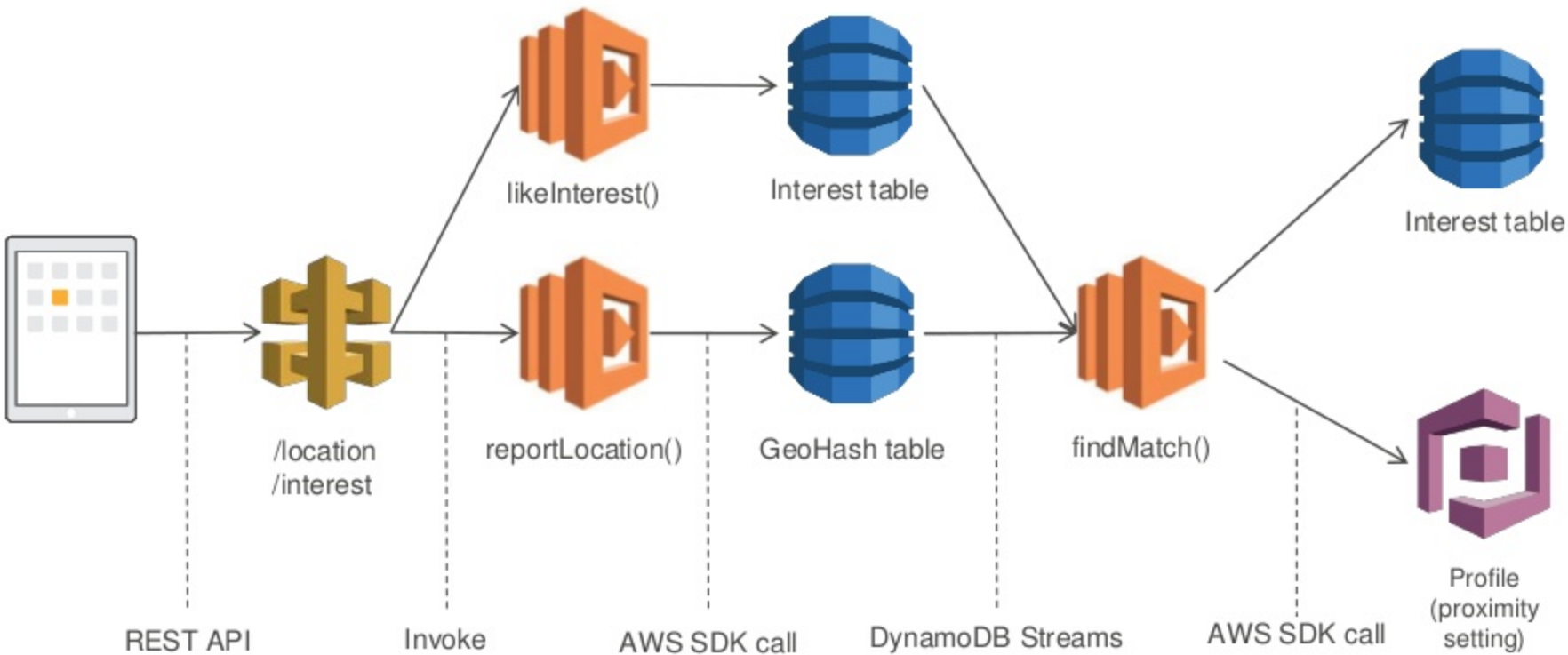


Match and alert users

AWS Lambda: Event-driven compute



Find a proximity match based on interests



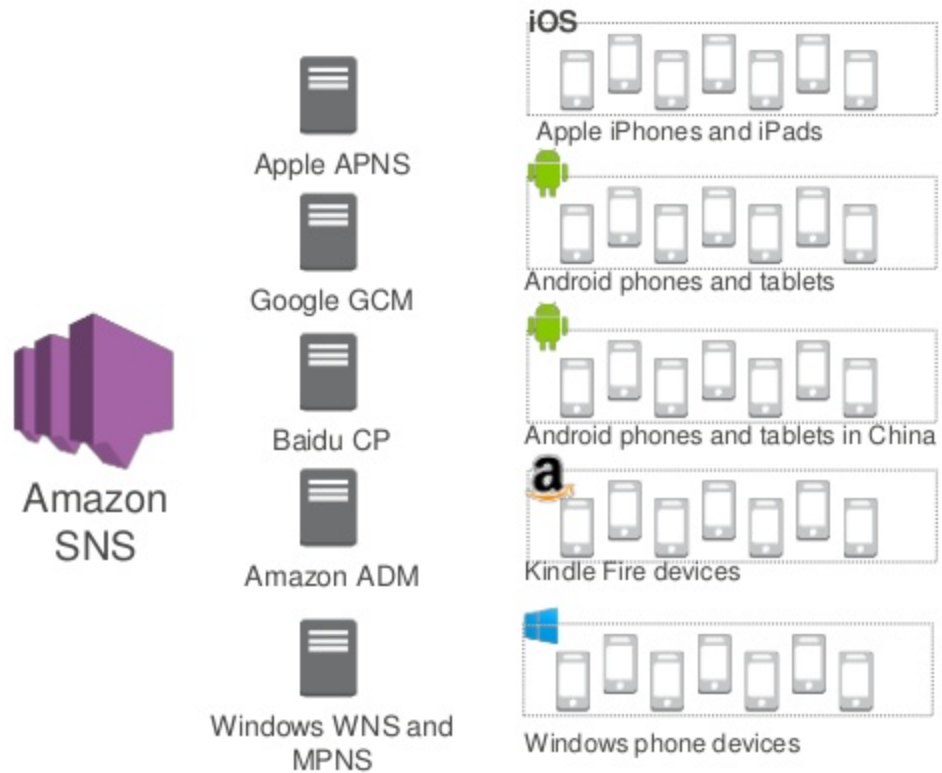
DynamoDB Streams processor: findMatch()

```
exports.handler = function(event, context) {

    // Process all the records in the stream
    event.Records.forEach(function(record) {

        var newLocation = record.dynamodb.NewImage.geohash.S;

        if (findProximityMatch(newLocation)) {
            // Found match!
        }
    });
    context.succeed();
};
```

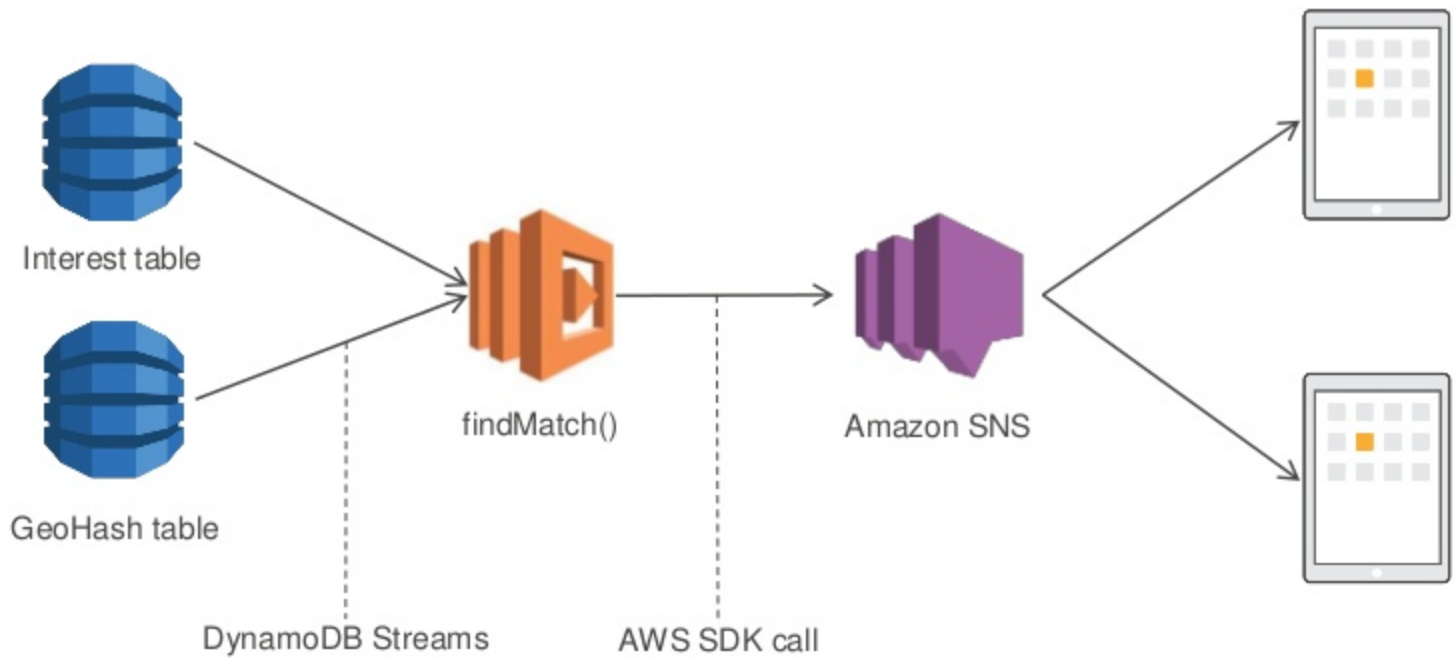
We found a match. Now what?

Amazon SNS mobile push notifications

- Amazon SNS is a fully managed, **cross-platform mobile push** intermediary service
- Fully scalable to **millions** of devices
- Allows you to create **topics** (e.g. per geo, interest, usage pattern, etc.)



Found a match: Notify user!



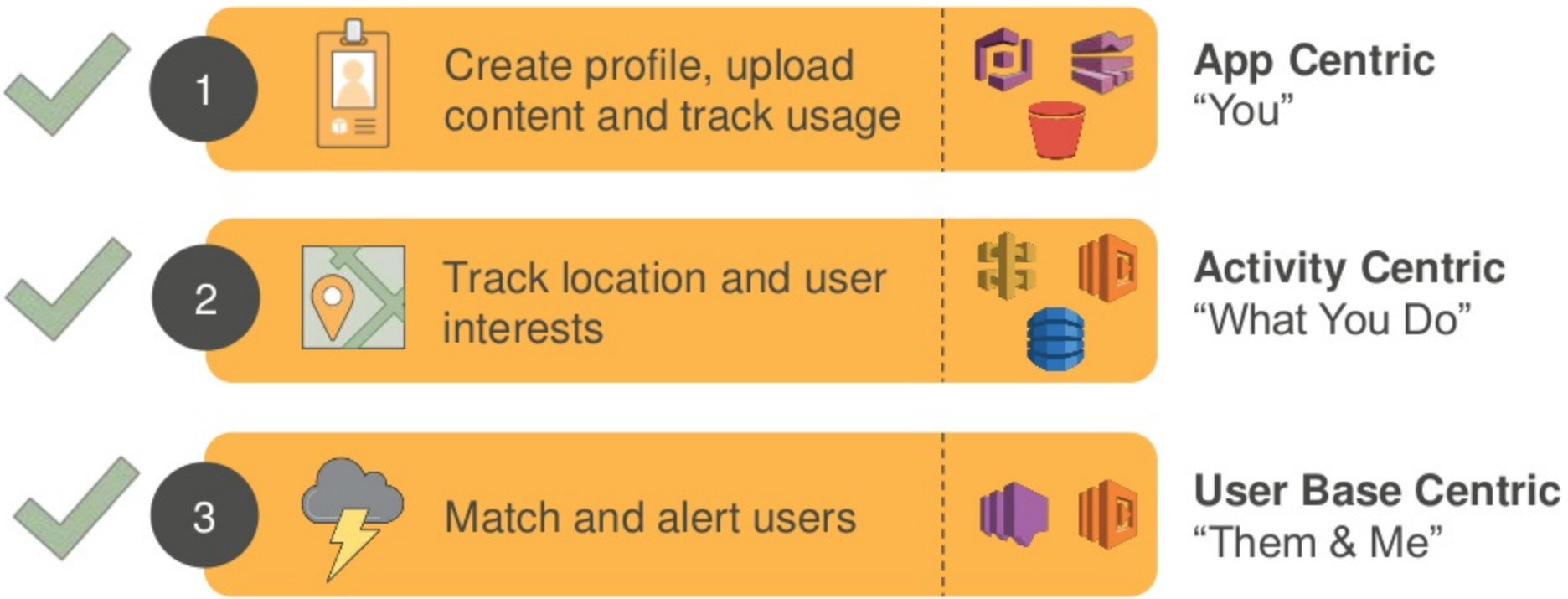
Interest Radius

But what if I adjust my profile?

Cognito Sync Trigger – AWS Lambda Code

```
exports.handler = function(event, context) {
    if (event.eventType === 'SyncTrigger') {
        event.datasetRecords.forEach(function(item) {
            if (item.interestRadius.op == 'replace') {
                // New interest radius set - process findMatch()
                var params = {
                    FunctionName: 'findMatch',
                    InvocationType: 'Event', //makes it async
                    Payload: '{"user":'+ item.identityId +'}'};
                lambda.invoke(params, function(err, data) {[..]});
            }
        }
    }
    context.succeed(event);
};
```

Let's think in Layers



Final architecture

Thank you!