Intense Gaming

Chris Elliott
Solutions Architect
http://about.me/elliottchris

Google Cloud Platform

Build Scale Survive

Session Game Plan

- Gaming on GCP Overview
- Mobile Gaming Architecture
- FreshPlanet and SongPop
- Dedicated Game Server Architecture
- Wrap Up

App Engine   Compute Engine   Cloud Storage   BigQuery   Cloud SQL

Gaming on Google Cloud Platform

Google™ Cloud Platform

Google Services

YouTube

Compute

- Compute Engine (IaaS)
- App Engine (PaaS)

Storage

- Cloud Storage (Object)
- Cloud SQL (Relational)
- DataStore (NoSQL)

Services

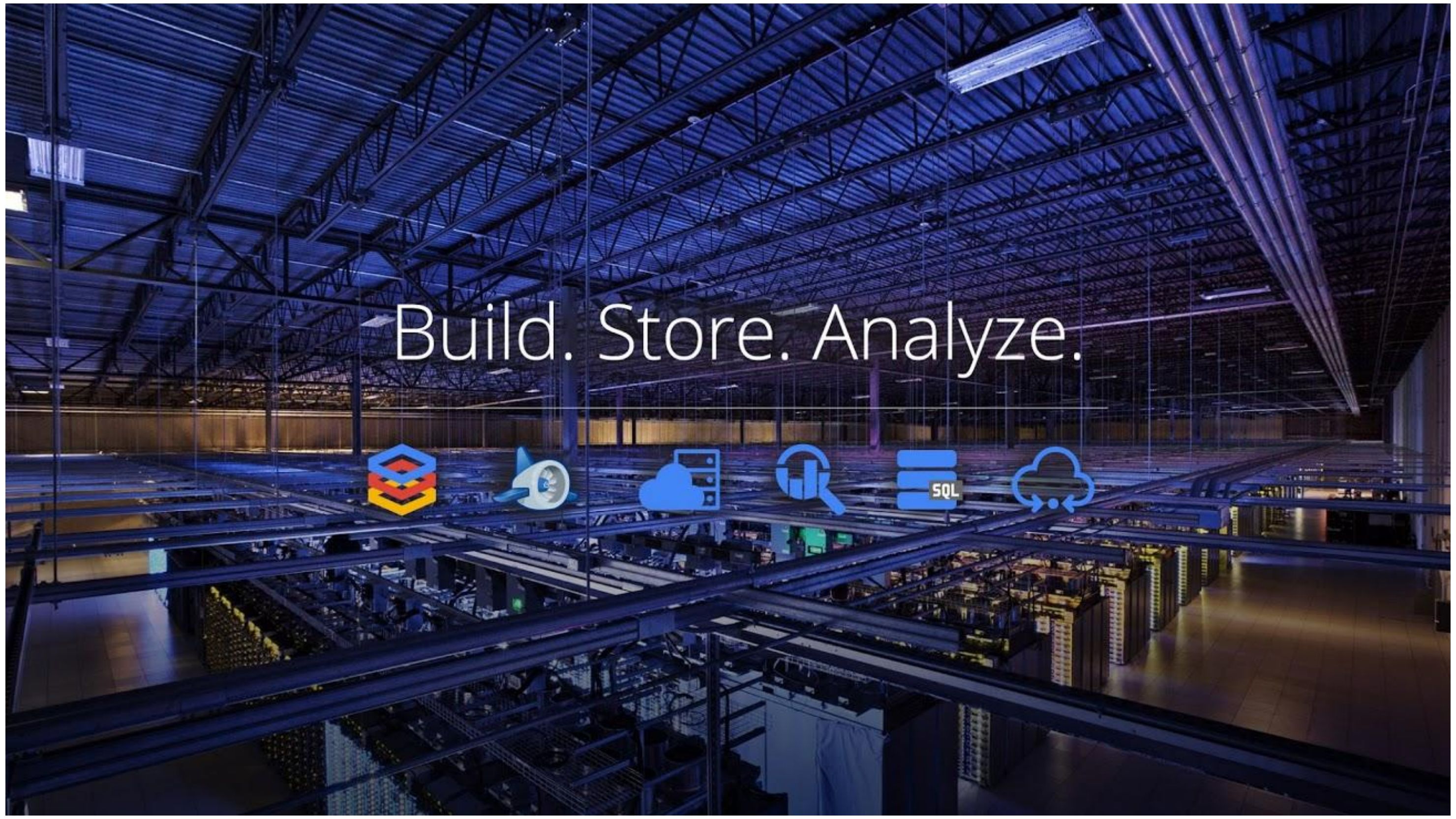
- BigQuery
- Cloud EndPoints
- ✓ Caching
- ✓ Queues
- ✓ and more...

Google Infrastructure

✓ Global Data Centers	✓ Performance	✓ Disaster Recovery	✓ Security
✓ 99.95% Uptime SLA	✓ Redundancy	✓ Audits & Certifications	✓ Energy Efficient

Google Cloud Platform by The Numbers

- 3M Active Applications
- 300,000 Unique Developers per Month



Gaming on Google Cloud Platform

FreshPlanet

EA

ROVIO

EPIC GAMES

Applibot®

UBISOFT®

PHYKEN MEDIA

gamesys

BACKFLIP STUDIOS

MAG interactive

chillingo

PocketGems

OUTFIT7

TETRIS
BLITZ
EA

WORLD SERIES OF
Yahtzee
EA

W
S
EA

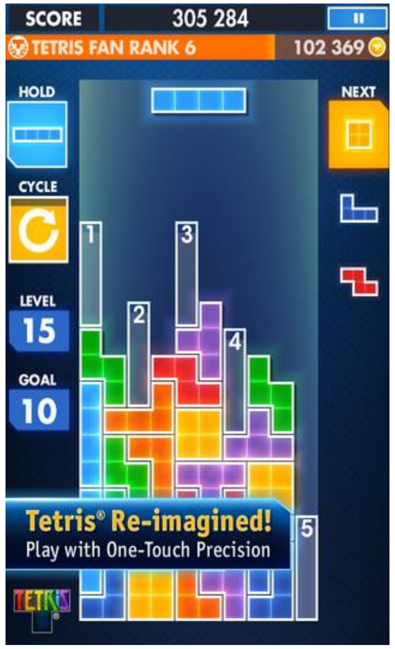
TETRIS
PUBLISHED BY
EA

TETRIS BLITZ
965,377
Scott 256,000 SHARE
Joe 214,120 BRAG
Brett 180,720 BRAG
Amy 155,000 BRAG
Adam 120,250 BRAG
PLAY YOUR FRIENDS
Soar to the top of the leaderboards
Jeremy 61,320 BRAG
Andy 34,050 BRAG
Leaderboard Reset in: 2 Hours
PLAY

WORLD SERIES Yahtzee
240 Chips
Get More
Play
Random Game
Your Turn
Gina
UR IT! It's your turn to play.
5 minutes ago
Matt
UR IT! It's your turn to play.
30 minutes ago
UR IT!™
TAKE TURNS WITH YOUR FRIENDS
Their Turn
Nichole
It's Nichole's turn
2 hours ago

The Simpsons
Set Complete
Level up & unlock the entire Simpsons family!
Original content from the writers & animators of The Simpsons!
Build your very own Springfield!

SCORE 2650
GUESSES 5
THEME
TEACH
TRUCE
GUESS THE WORDS
BEAT YOUR FRIENDS
QWERTYUIOP
ASDFGHJKL
ZXCVBNM


SCORE 305 284
TETRIS FAN RANK 6 102 369
HOLD
NEXT
CYCLE
LEVEL 15
GOAL 10
Tetris® Re-imagined!
Play with One-Touch Precision

Epic Games

- Compelling social features powered by App Engine
- Scaled to millions of users engaging in challenges
- Core development completed by 1 person in 6 weeks

Infinity Blade 2

Vote!!! The Game

App Engine

Easy to build
Easy to scale
Easy to maintain

Compute Engine

Open and Flexible
Consistent Speed
Global Network

“Your latency is crazy low. You should pat the Compute Engine team on the back for that.”

Edward Goose
PA Consulting

MapR Terasort Record

	MapR World Record	Previous Record
Sort Time (s)	54	62
Number of Servers	1003	1460
Number of Cores	4012	11680
Number of Disks	1003	5840
Time to Build Cluster	Minutes	Months

"Everything is noticeably faster on Compute Engine"

Kunal Patel
Phyken Media

Mobile Gaming Architecture

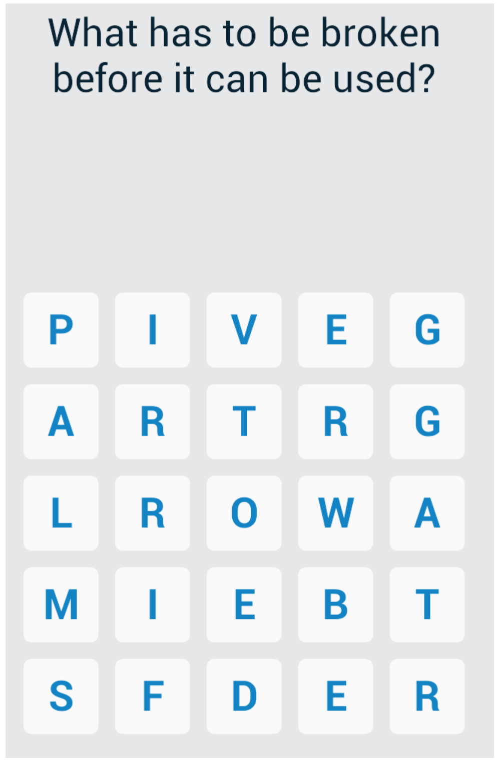
Mobile Gaming Reference Architecture

- Support iOS and Android Devices
- Scalable to Millions of Users
- Engaging Social Components

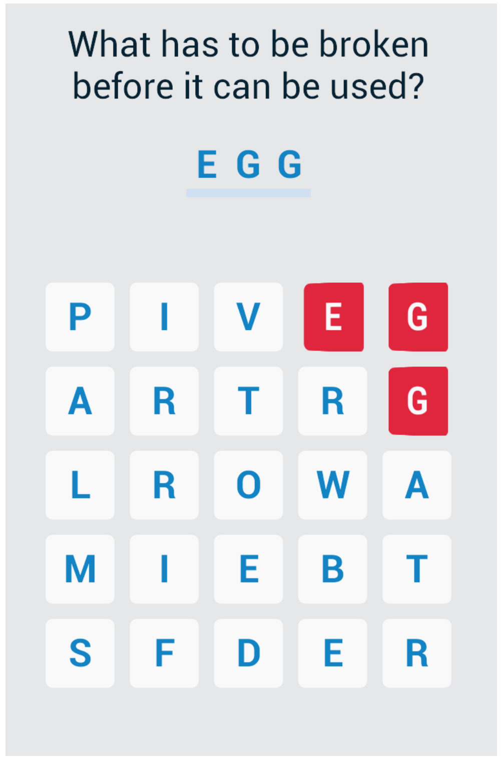
App Engine Compute Engine Cloud Storage BigQuery Cloud SQL

Sample Application - Griddler

- Async Multiplayer Riddle Mobile Game

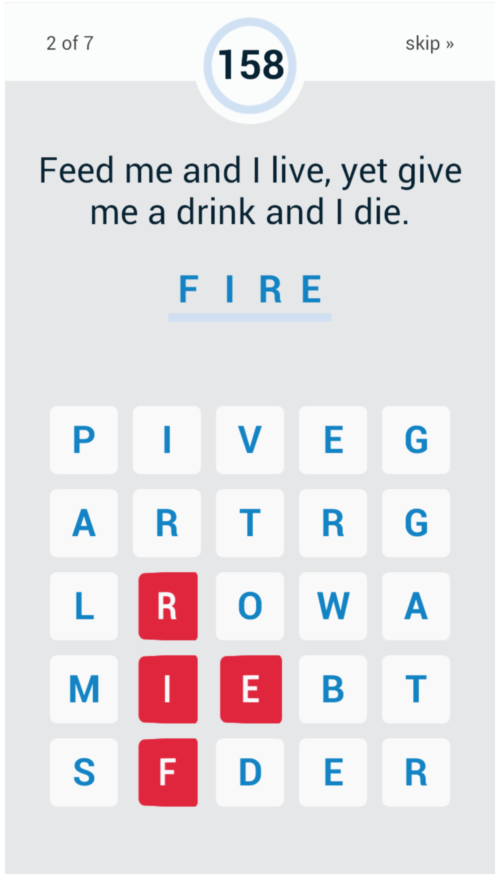

What has to be broken before it can be used?

Riddle


What has to be broken before it can be used?

E G G

Answer



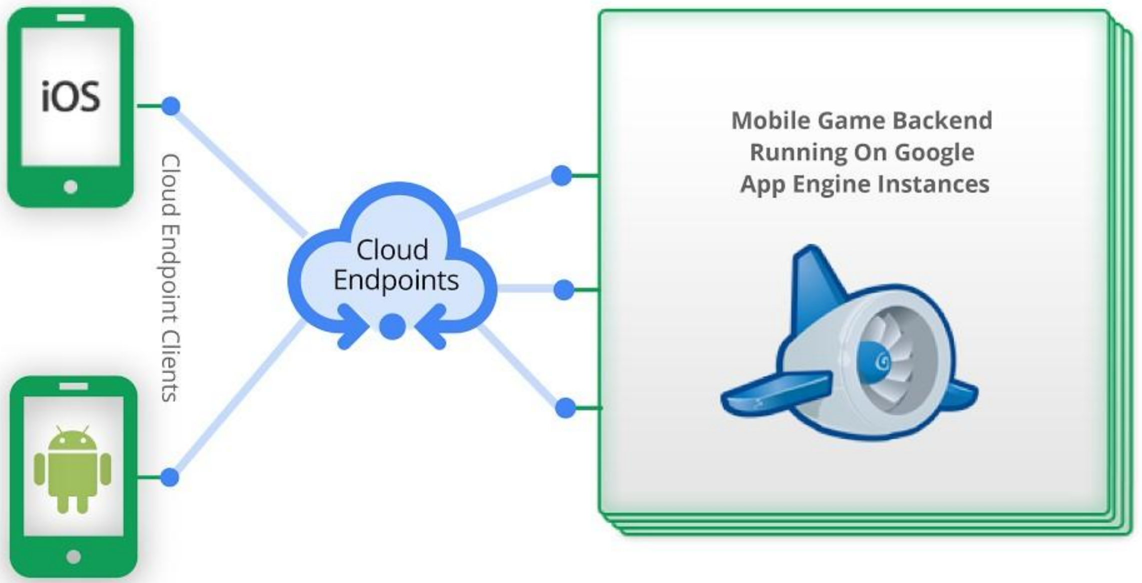
Feed me and I live, yet give me a drink and I die.

F I R E

Score

Mobile Application

Cloud Endpoints and App Engine



Griddles Game API

Send Game Invitation

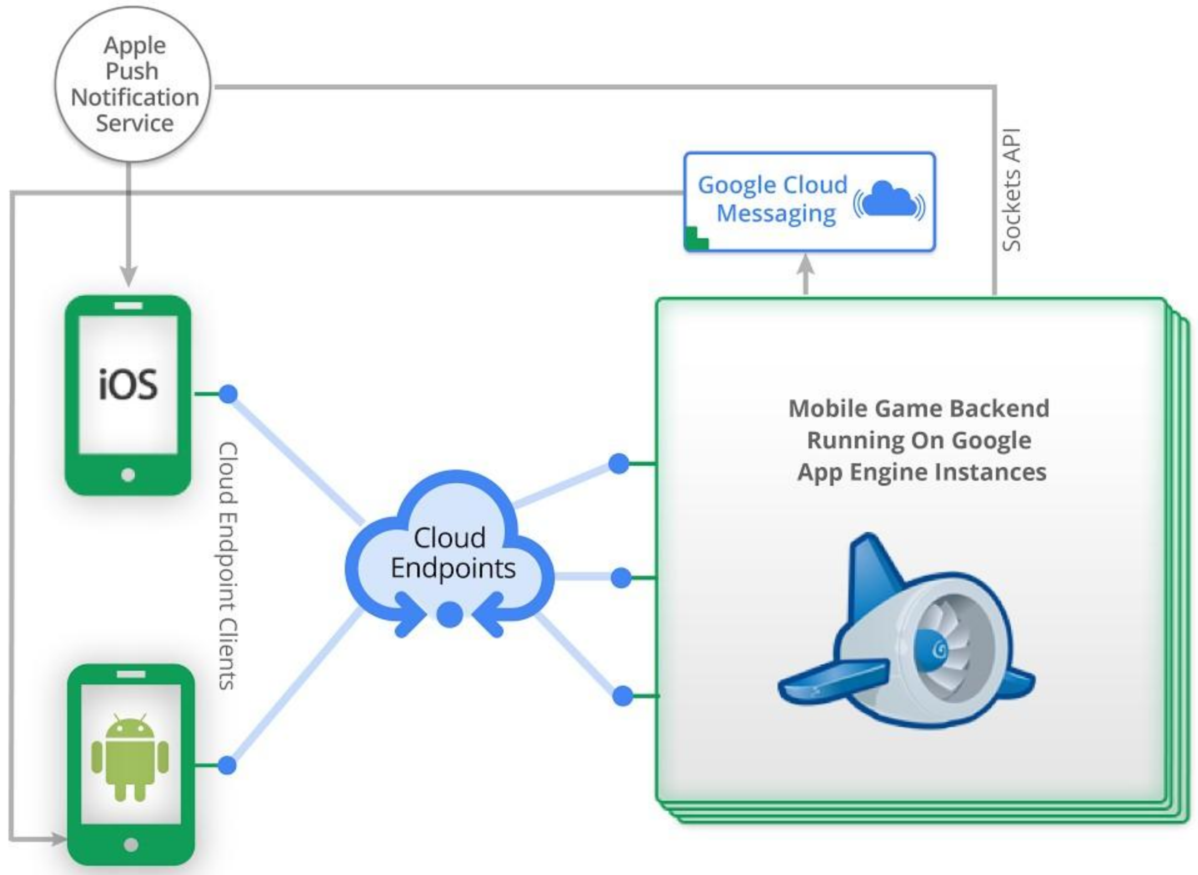
Java - App Engine

```
@ApiMethod(httpMethod = "PUT", path ="game/{gameId}/invitation/{playerId}")
  public InvitationResult sendInvitation(
       @Named("gameId") Long gameId,
       @Named("playerId") Long playerId,
       User user)
throws ServiceException { }
```

Java - Android App

```
InvitationResult invitation = gameBackend.sendInvitation(
                                 gameId, playerId)).execute();
```

Push Notifications


Apple Push Notification Service
Google Cloud Messaging
Sockets API
iOS
Cloud Endpoint Clients
Cloud Endpoints
Mobile Game Backend Running On Google App Engine Instances

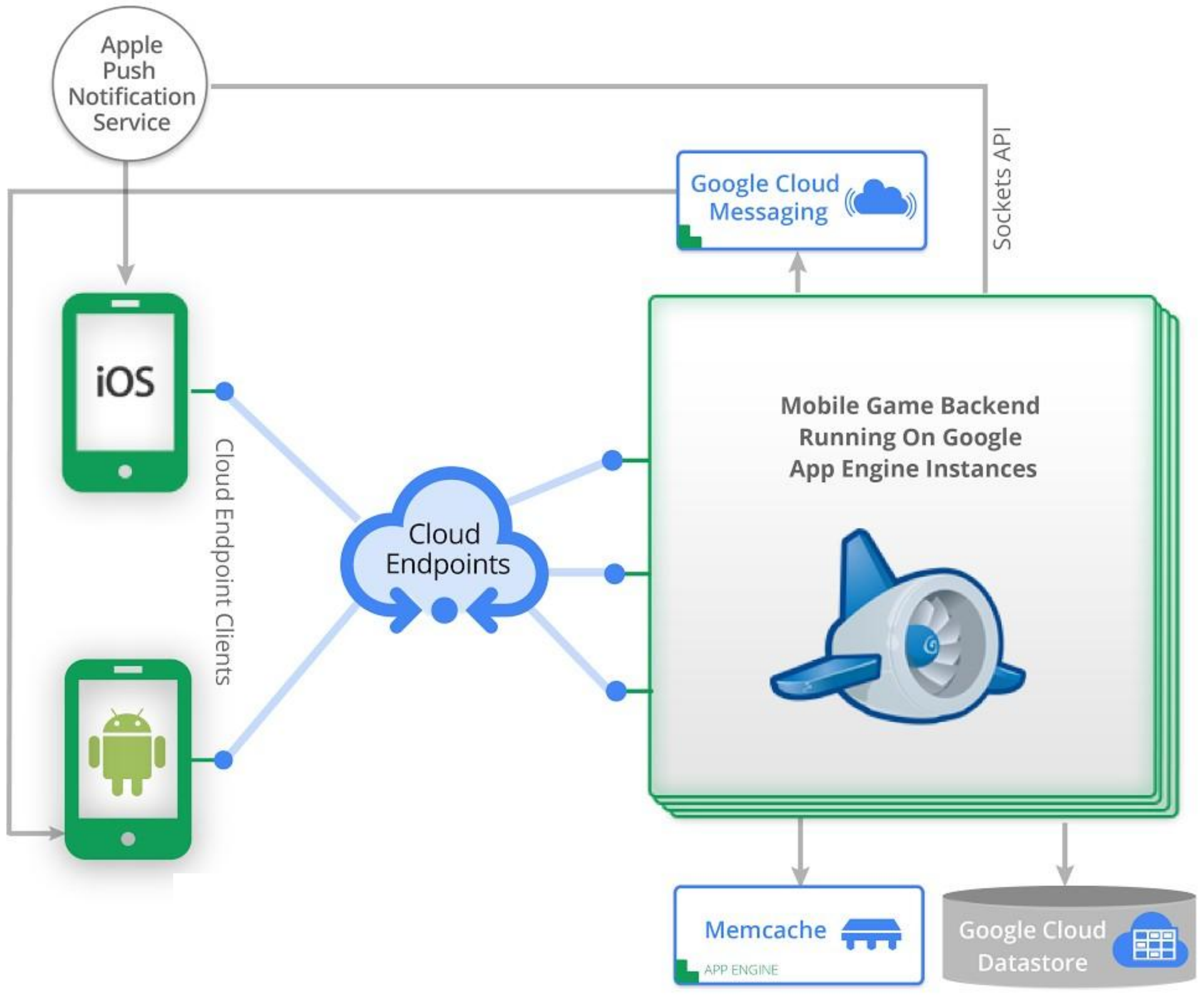
Apple Push Notification Example

Java - App Engine

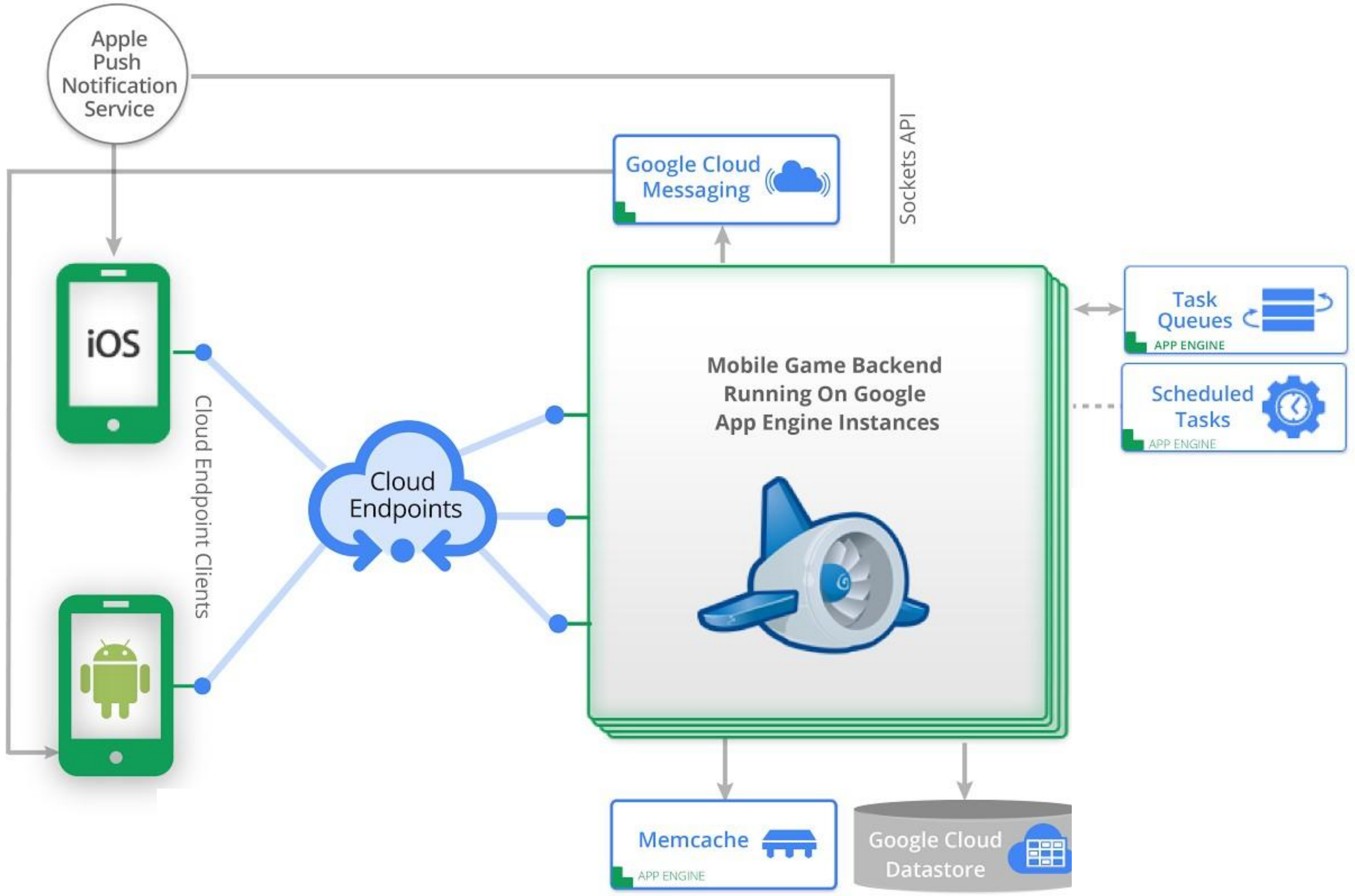
```
import javapns.Push;

private void sendAlert(String alertMessage, String[] deviceTokens) {
        try {
                PushedNotifications notifications = Push.alert(
                        alertMessage,
                        Configuration.getCertificateStream(),
                        Configuration.CERTIFICATE_PASSWORD,
                        Configuration.USE_PRODUCTION_APNS_SERVICE,
                        deviceTokens);
        //Check for Success, Catch, etc...
```

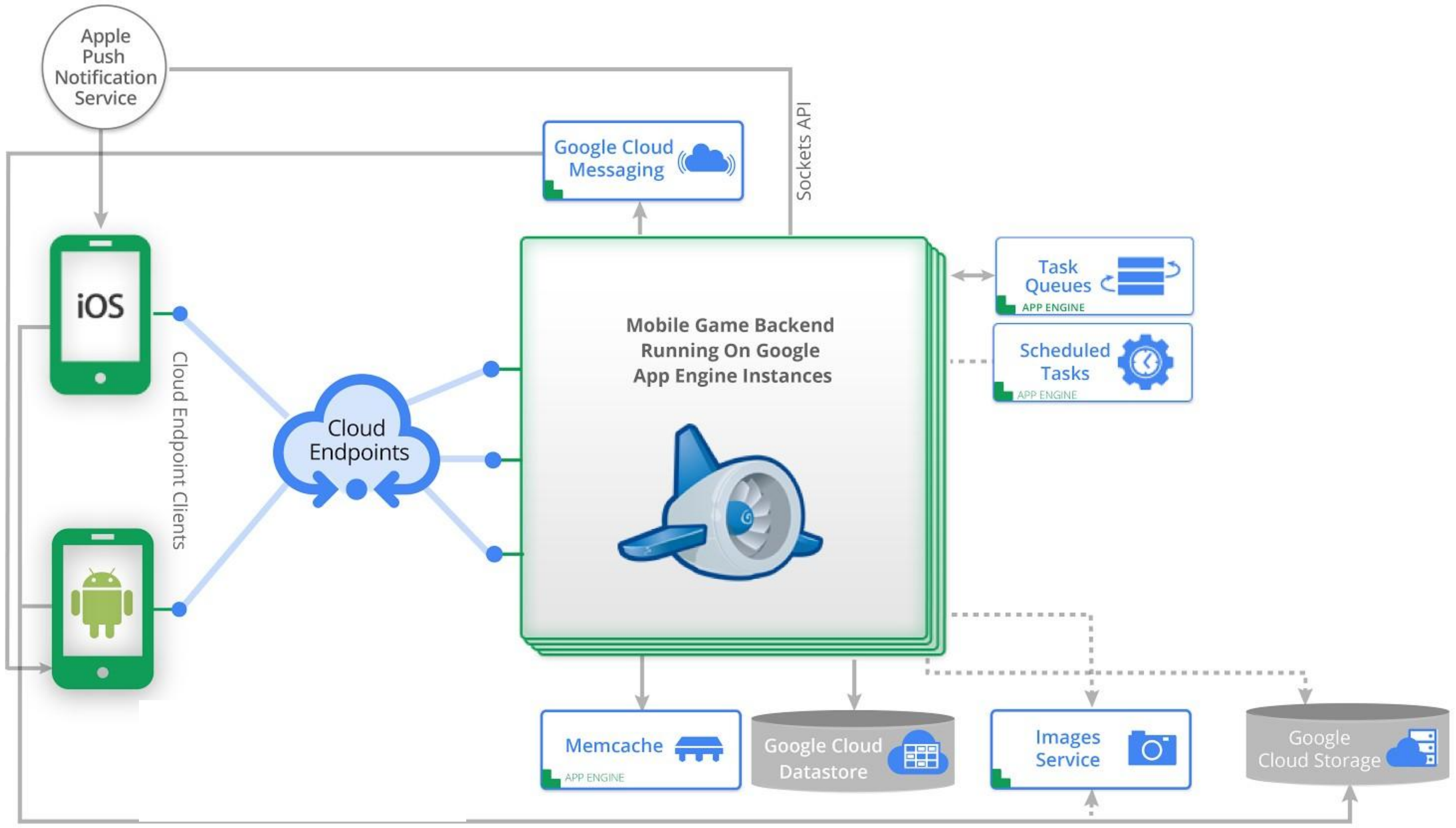
Datastore and Memcache



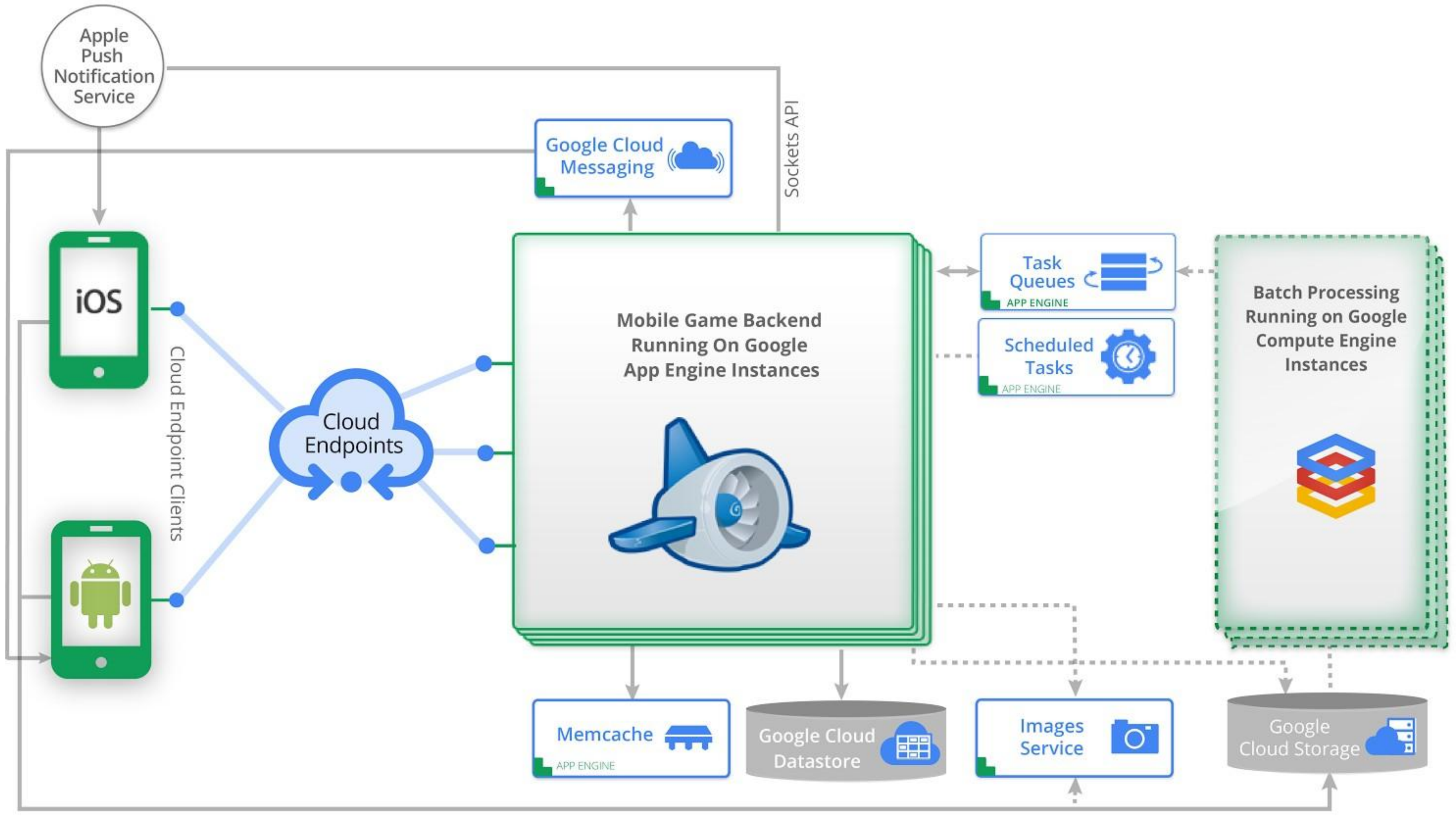
Background Tasks



Object Storage



Compute Engine



Compute Engine Task Queue Access

Instance Creation

```
gcutil --project=123456789 addinstance foobar \
       --service_account_scopes=https://www.googleapis.com/auth/taskqueue
```

GCE Instance

```
curl "http://metadata/computeMetadata/v1beta1/instance/service-accounts/default/token"
```

Python Client Library

```
from oauth2client.gce import AppAssertionCredentials

http = AppAssertionCredentials("https://www.googleapis.com/auth/taskqueue").authorize(httplib2.Http())
```

Compute Engine Task Queue Access

RESTful API

Base URL: https://www.googleapis.com/taskqueue/v1beta2/projects

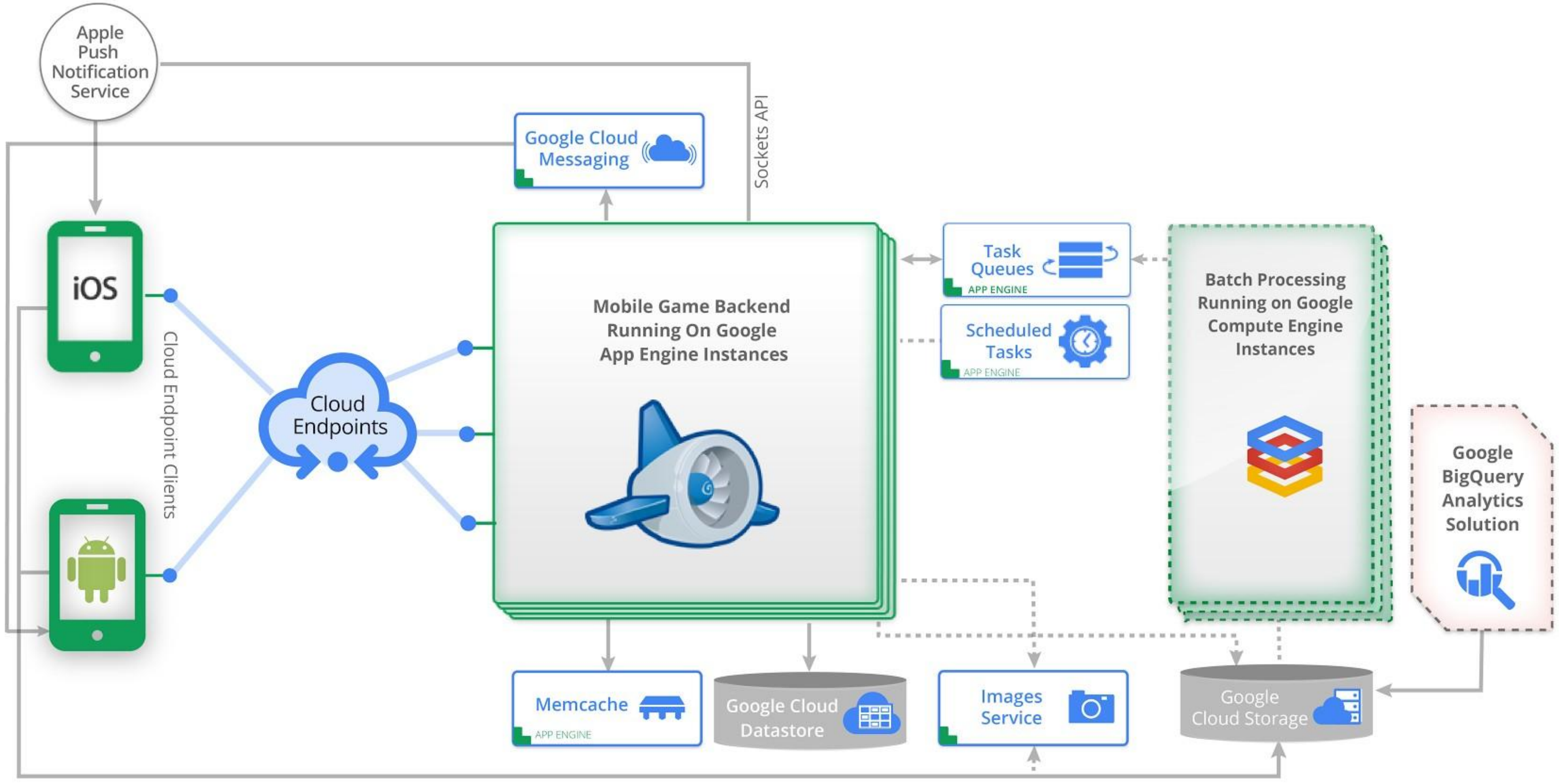
Get Lease: POST /{project}/{taskqueues}/taskqueue/{tasks}/lease

Delete Task: DELETE /{project}/taskqueues/{taskqueue}/tasks/{task}

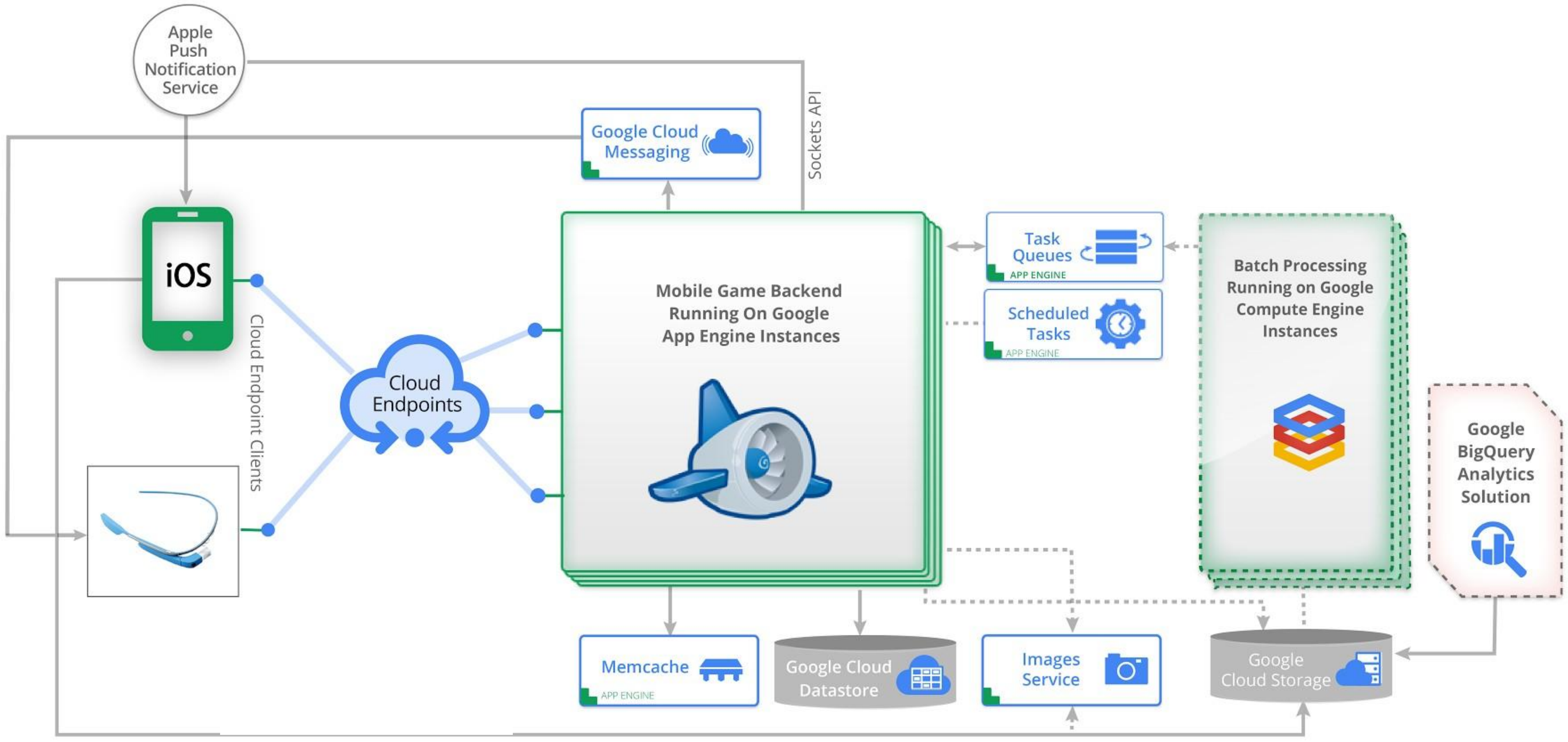
Add Task: POST /{project}/taskqueues/{taskqueue}/tasks

Autoscaling? Use Stats: GET /{project}/taskqueues/{taskqueue}

Big Query


Apple Push Notification Service
Sockets API
Google Cloud Messaging
iOS
Cloud Endpoint Clients
Cloud Endpoints
Mobile Game Backend Running On Google App Engine Instances
Task Queues
APP ENGINE
Scheduled Tasks
APP ENGINE
Batch Processing Running on Google Compute Engine Instances
Google BigQuery Analytics Solution
Memcache
APP ENGINE
Google Cloud Datastore
Images Service
Google Cloud Storage

Glass Integration



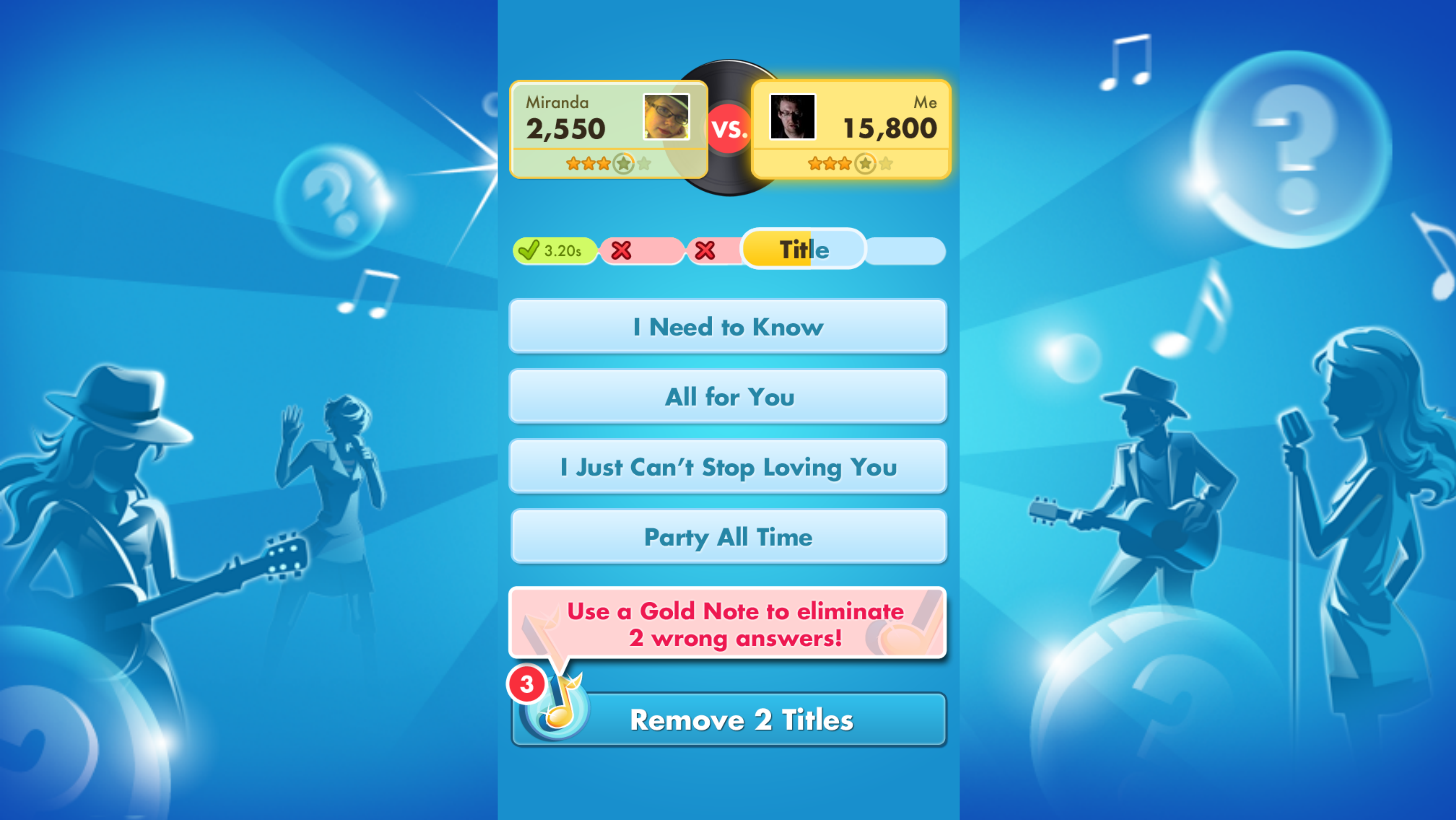
FreshPlanet

Alexis Hanicotte, Lead Back End developer
Olivier Michon, CTO

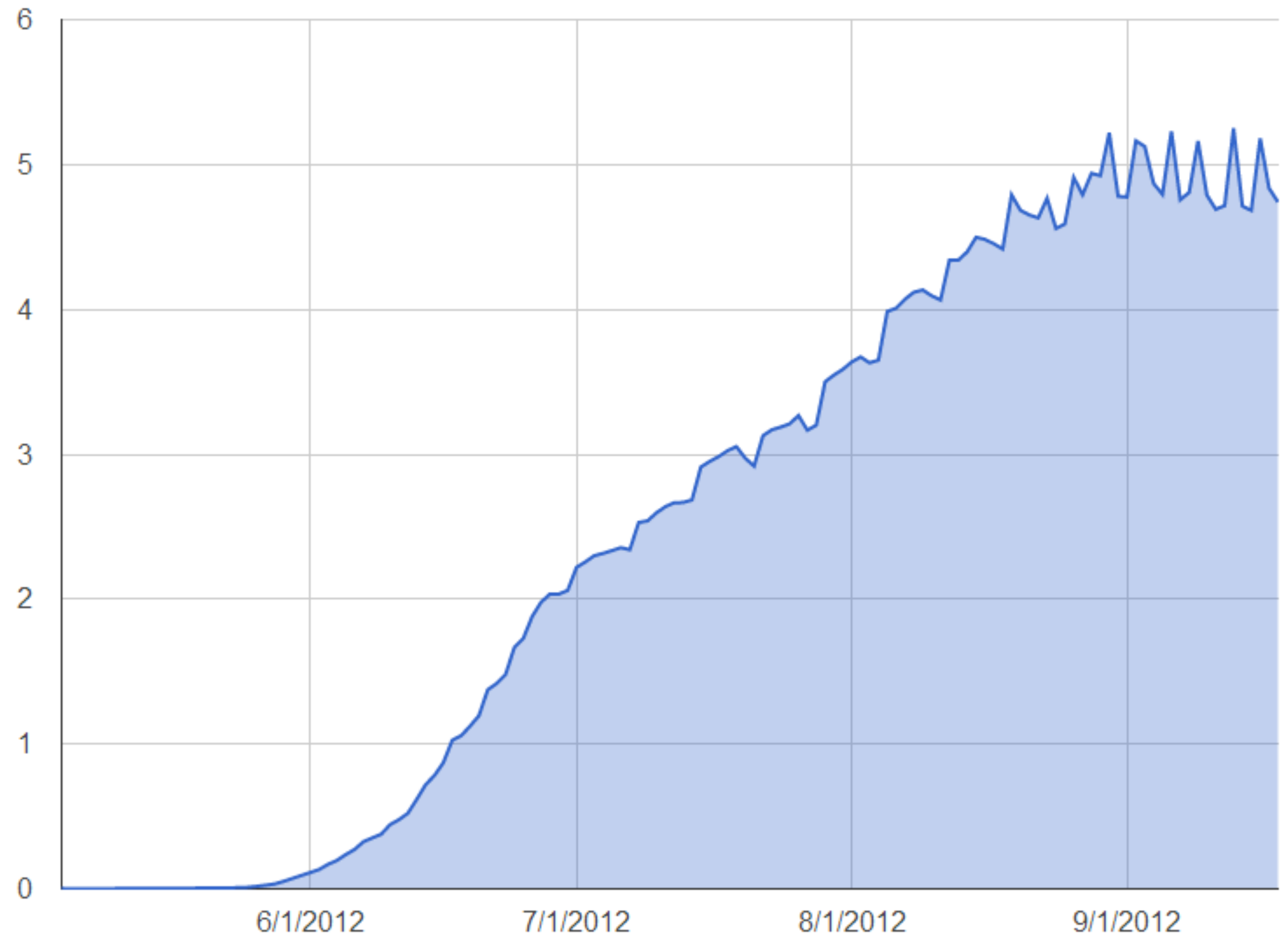
FreshPlanet

- Small startup (~20) based in New York
- Making mobile social games and apps
- Teams of 3-4 developers per project
- Focused on being efficient
- Powered by Google Cloud Platform

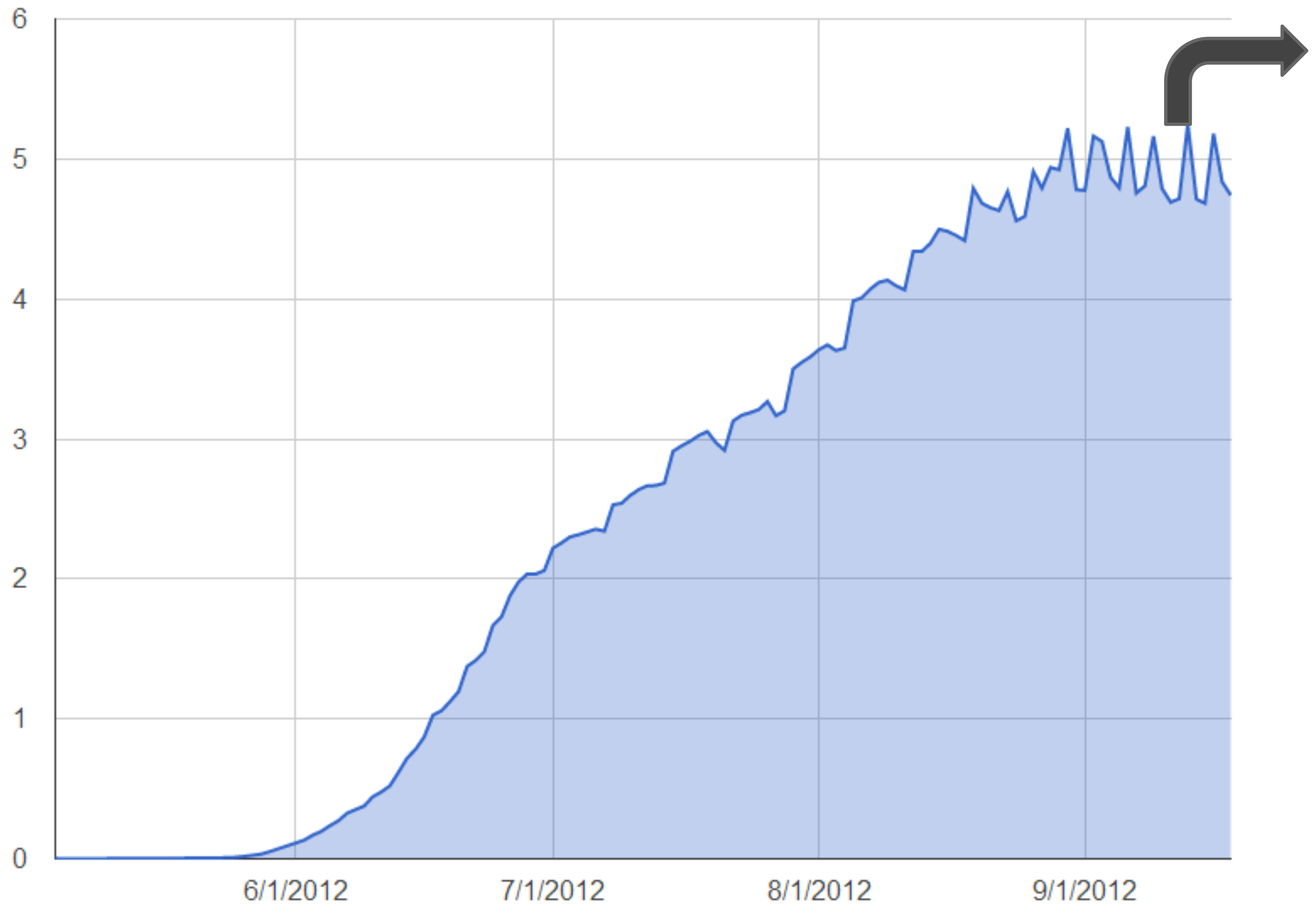
Song
Pop


Miranda
2,550
VS.
Me
15,800
3.20s
Title
I Need to Know
All for You
I Just Can't Stop Loving You
Party All Time
Use a Gold Note to eliminate 2 wrong answers!
3
Remove 2 Titles

SongPop daily active users (millions)



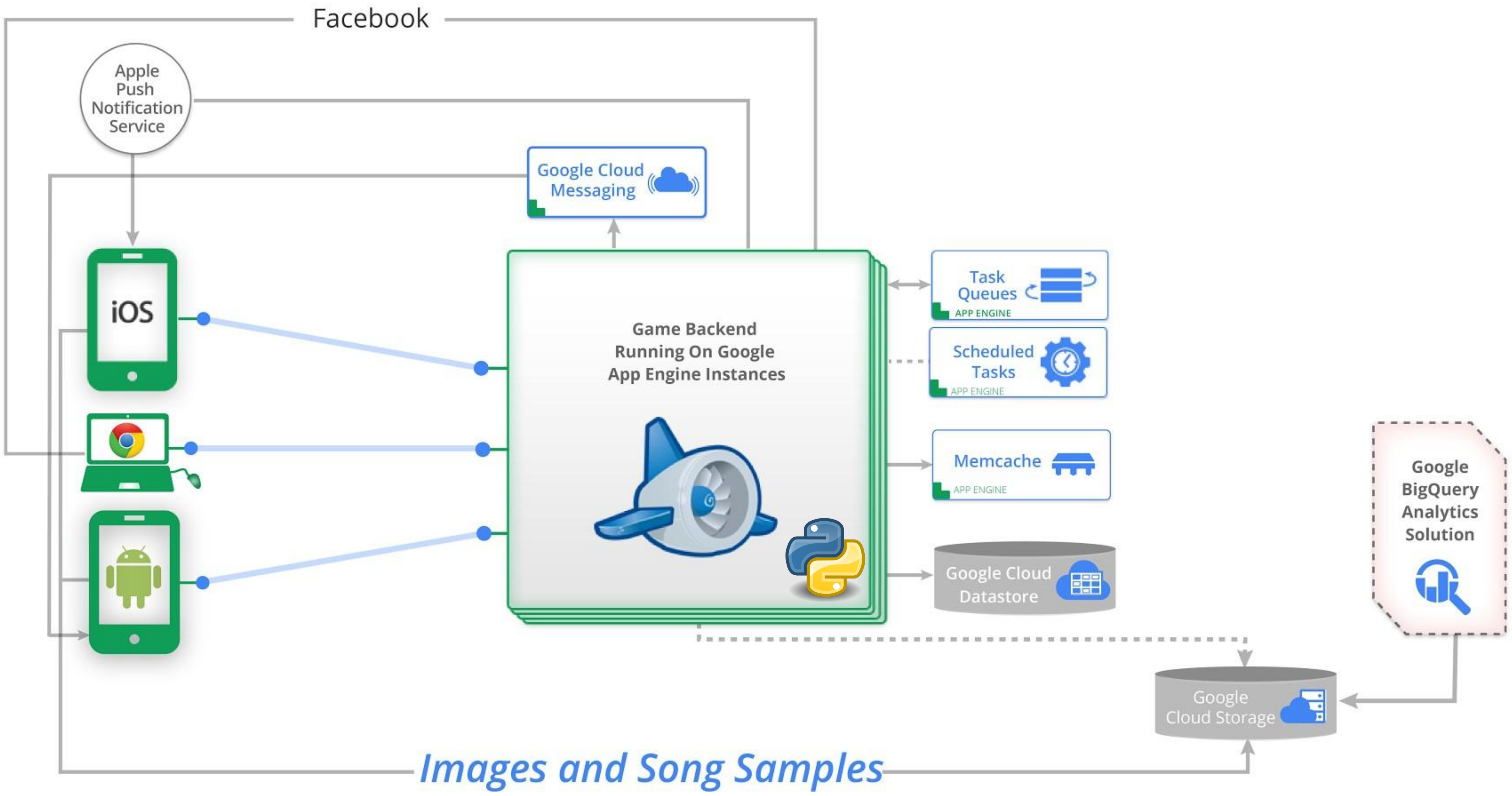
SongPop daily active users (millions)



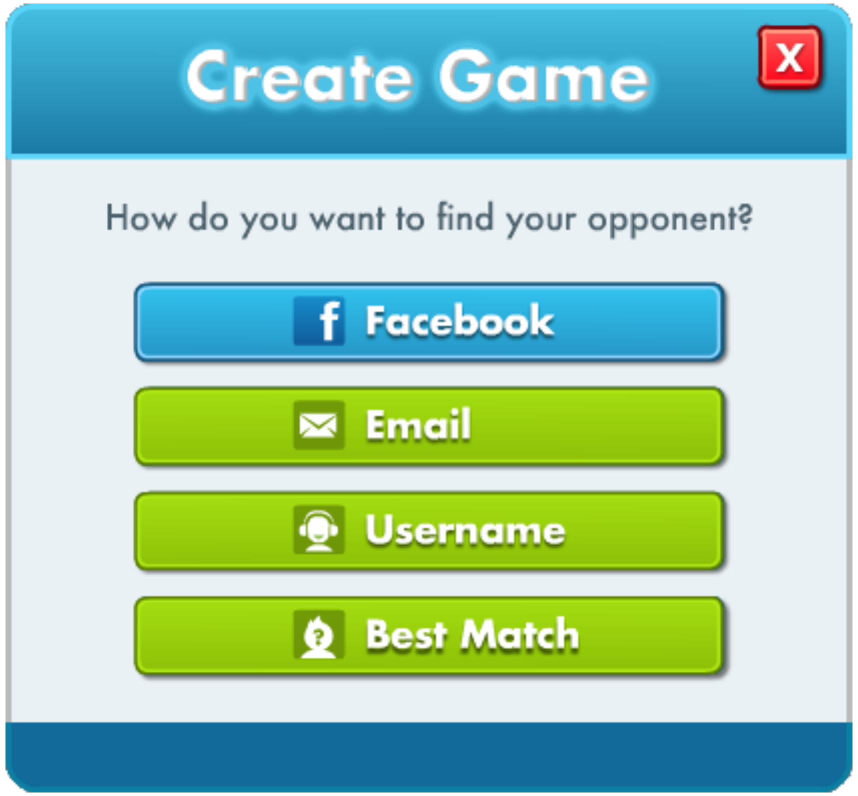
Per Day

- 300 Million Requests
- 2.7 Billion Datastore Ops
- 18TB of Songs and Images

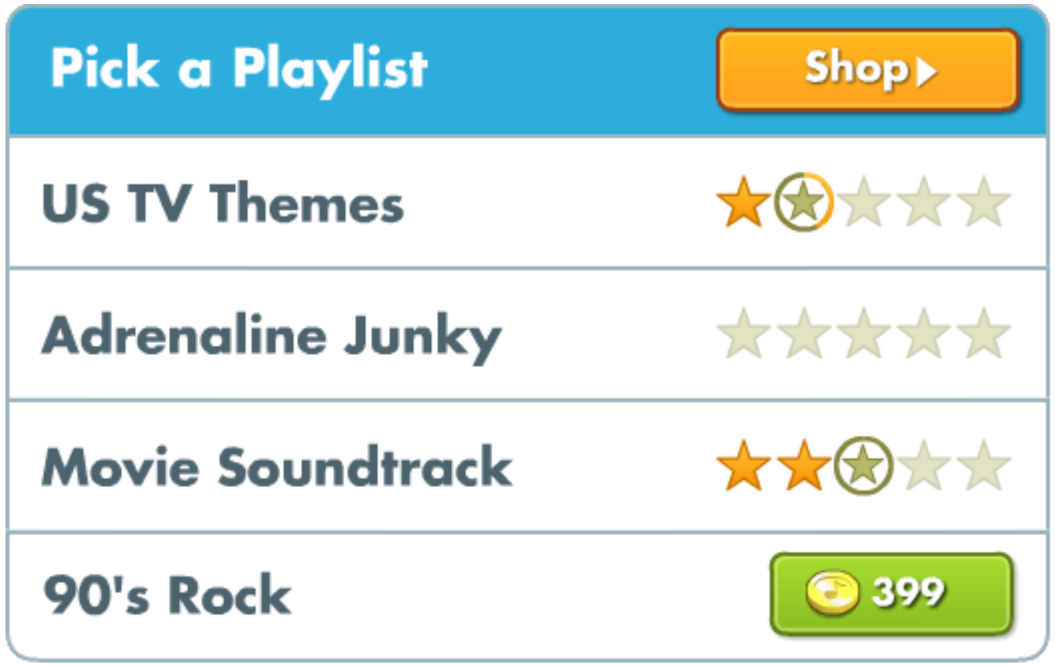
Technical Overview



Technical Overview


Create Game

How do you want to find your opponent?

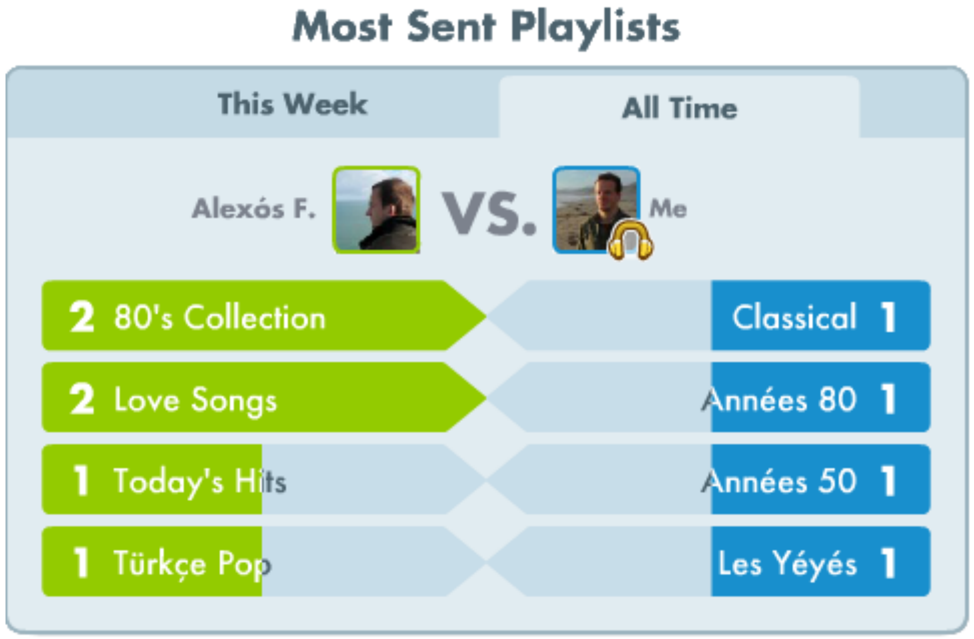

Pick a Playlist

US TV Themes

Adrenaline Junky

Movie Soundtrack

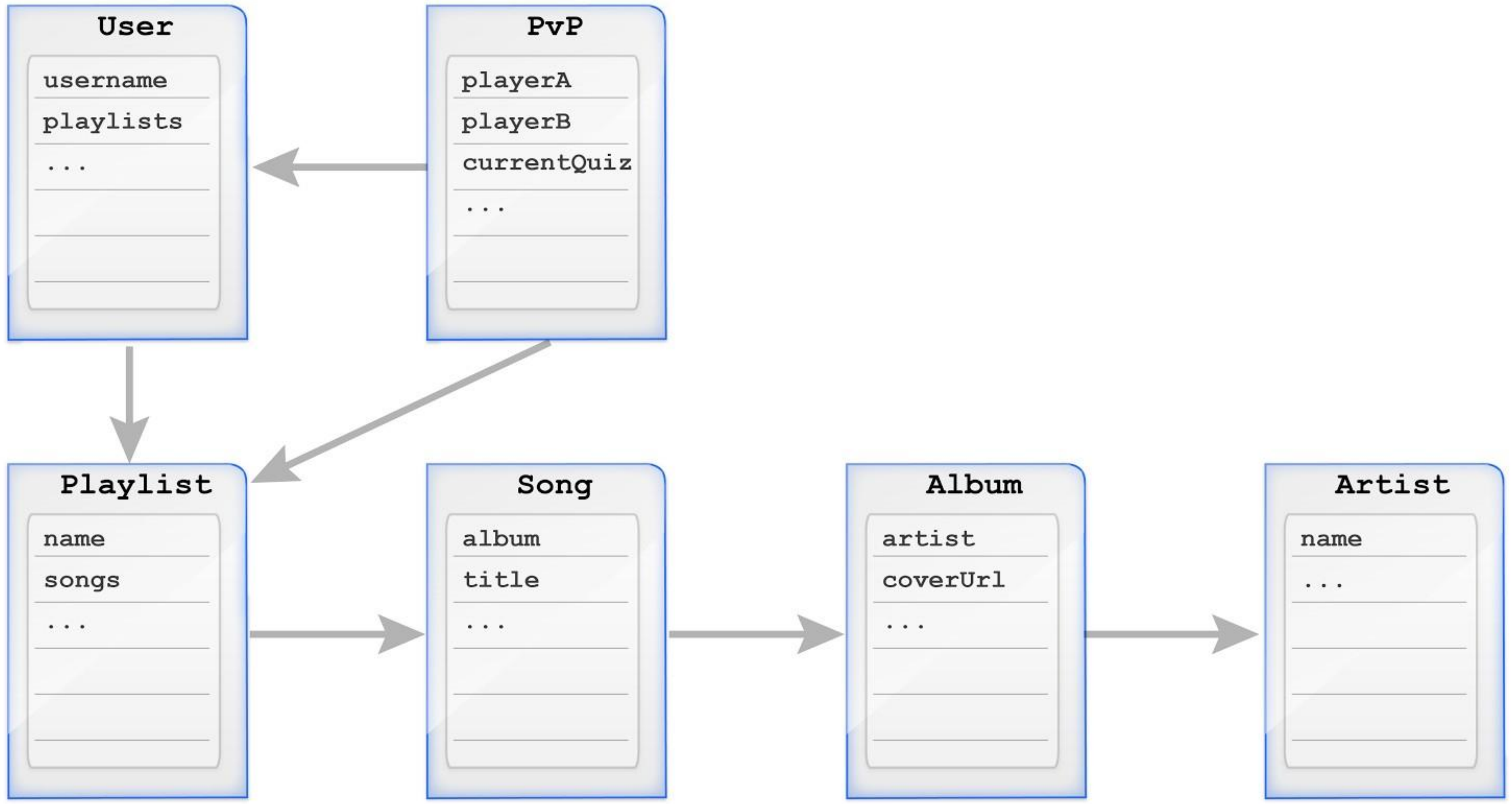
90's Rock


Most Sent Playlists

Alexós F. VS. Me

2	80's Collection	Classical	1
2	Love Songs	Années 80	1
1	Today's Hits	Années 50	1
1	Türkçe Pop	Les Yéyés	1

Modeling



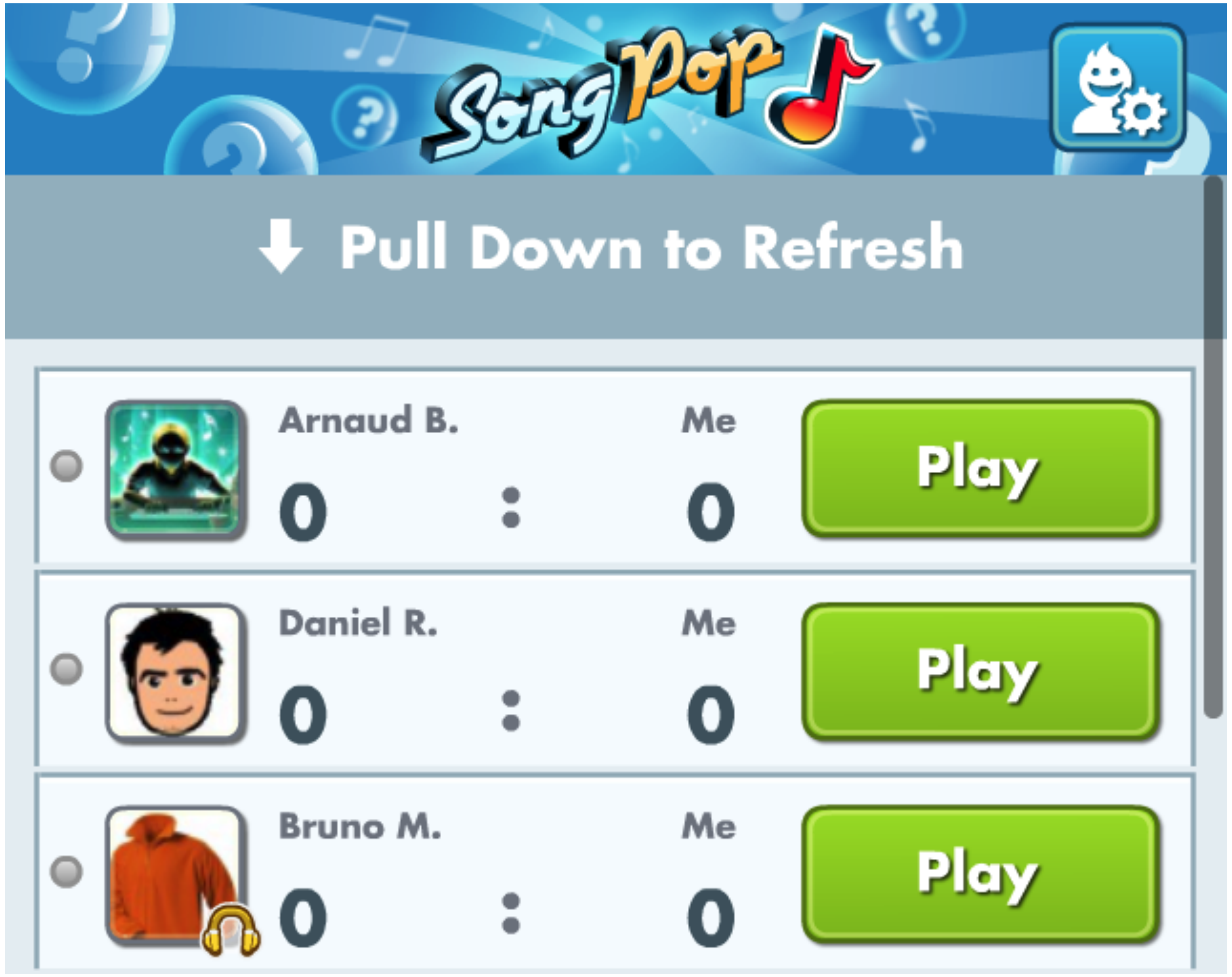
User
username
playlists
...
PvP
playerA
playerB
currentQuiz
...
Playlist
name
songs
...
Song
album
title
...
Album
artist
coverUrl
...
Artist
name
...

Modeling

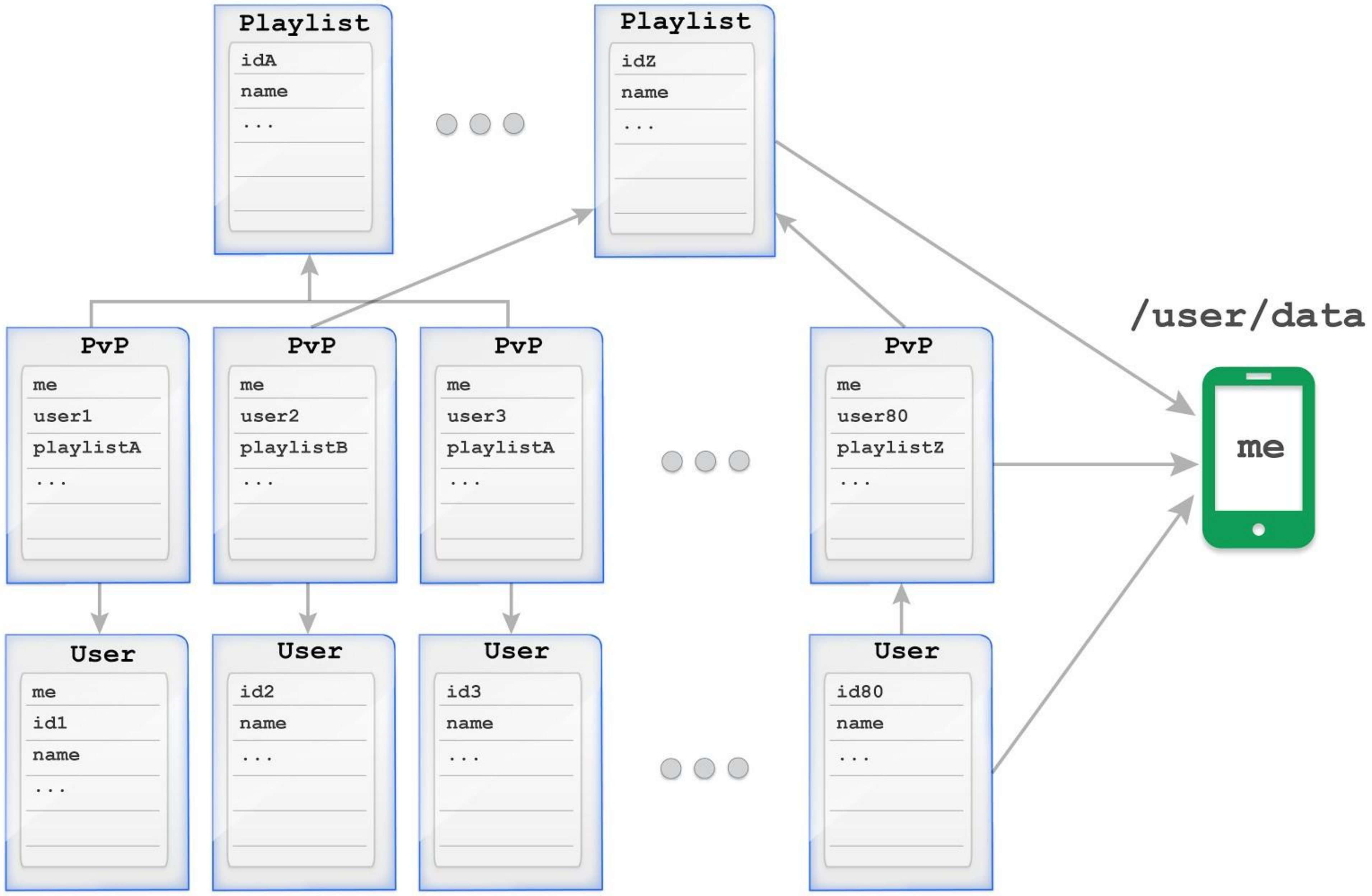
Python

```python
class Playlist(ndb.Model):
    name = ndb.StringProperty()
    picUrl= ndb.StringProperty()
    songs = ndb.KeyProperty( kind = Song, repeated=True )
    locales = ndb.StringProperty( repeated=True )
    price = ndb.IntegerProperty()
```

Case study: /user/data



Case study: /user/data



Case study: “/user/data”

Python - Algo

```
def userDataHandler():
    user =  getCurrentUser()
    games = queryUserGames(user)
    playlists = getPlaylistsFrom(user, games)
    opponents = getOpponents(games)
    return formatToJSON(user, opponents,
                        games, playlists)
```

Zoom on: Fetch by key

- Using NDB:
    - Get from Datastore
    - Automatically added to Memcache layer
    - Added to in-memory cache

Python - Algo

```
def userDataHandler():
    user =  getCurrentUser()
    games = queryUserGames(user)
    playlists = getPlaylistsFrom(user, games)
    opponents = getOpponents(games)
    return formatToJSON(user, opponents,
                        games, playlists)
```

Python

```
userId = int(self.request.get( 'userId', 0 ))
user = User.get_by_id( userId )
if user is None:
    return {'error':errors.UNKNOWN_USER}
```

Zoom on: Query for the games

Python - Algo

```
def userDataHandler():
    user =  getCurrentUser()
    games = queryUserGames(user)
    playlists = getPlaylistsFrom(user, games)
    opponents = getOpponents(games)
    return formatToJSON(user, opponents,
                        games, playlists)
```

Python

```
filter = ndb.query.OR(PvP.playerA == user.key, PvP.playerB == user.key)
games = PvP.query(filter).fetch(MAX_PVP)
```

Zoom on: Query for the games

- Store query results in Memcache
- Invalidate cache with hooks

Python - Algo

```
def userDataHandler():
    user =  getCurrentUser()
    games = queryUserGames(user)
    playlists = getPlaylistsFrom(user, games)
    opponents = getOpponents(games)
    return formatToJSON(user, opponents,
                        games, playlists)
```

Python

```
cacheKey = "queryUserGames(%s)" % user.key.id()
gameKeys = memcache.get(cacheKey)
if gameKeys is None:
    filter = ndb.query.OR(PvP.playerA == user.key, PvP.playerB == user.key)
    gameKeys = PvP.query(filter).fetch(MAX_PVP, keys_only=True)
    memcache.set(cacheKey, gameKeys)
games = ndb.get_multi(gameKeys)
```

Zoom on: Batching

Python - Algo

```
def userDataHandler():
    user =  getCurrentUser()
    games = queryUserGames(user)
    playlists = getPlaylistsFrom(user, games)
    opponents = getOpponents(games)
    return formatToJSON(user, opponents,
                        games, playlists)
```

Python

```
toGet = { user.availablePlaylists }
for pvp in pvps:
    toGet.add( pvp.currentQuiz.playlistKey )
    toGet.add( pvp.getOpponentKey( user.key ) )
ndb.get_multi(toGet)
```

Case study: “/user/data”

- Code
- Deploy
- Enjoy

11 Billion - Number of times these few lines of code have executed for 80 millions players

```
class UserDataHandler(webapp2.RequestHandler):

    @utils.authenticated
    @utils.jsonResponse
    def get(self):
        """
        @return: list of PvP for this player and the player profile.
        """
        userId = int(self.request.get( 'userId', 0 ))
        user = User.get_by_id( userId )
        if user is None:
            return {'error':errors.UNKNOWN_USER}

        pvps = PvPStatus.queryOpponents(user.key).get_result()

        # Batch fetch all data: each playlist we are interested in, each opponent profile
        toGet = { user.availablePlaylists }

        for pvp in pvps:
            quiz = pvp.currentQuiz
            toGet.add( quiz.playlistKey )
            toGet.add( pvp.getOpponentKey( user.key ) )

        # NDB will cache in the instance memory the results,
        # we can access them later by calling key.get()
        ndb.get_multi(toGet)

        pvpList = []
        for pvp in pvps :
            quiz = pvp.currentQuiz

            opponent = pvp.getOpponentKey( user.key ).get()
            if not opponent:
                logging.error("PvP opponent not resolved! pvpKey=%s", pvp.key)
                continue

            pvpInfo = pvp.toJsonObject( opponent=opponent )
            pvpInfo['quiz'] = quiz.toJsonObject( includeQuestions=False )

            pvpList.append( pvpInfo )

        userInfo = user.toJsonObject()

        playlistInfos = []
        for playlistKey in user.availablePlaylists:
            userLevel = user.getPlaylistLevel( playlistKey )
            playlist = playlistKey.get()
            playlistInfos.append( playlist.toJsonObject( userLevel=userLevel ) )

        userInfo['playlists'] = playlistInfos

        return {'pvpList': pvpList, 'user': userInfo }
```

For App Engine developers...

- Use Memcache wisely
- Only index what is truly needed
- App Engine scales a lot more than the official limits
- Responsive support, qualified people

Conclusion

- App Engine allowed us to scale
  - No added complexity
  - Don't spend your time on it
  - Reasonable cost

- Instrumental to SongPop's success

Want more?

- Come meet us at Developers Sandbox
- More projects coming, using Google Cloud Platform
- We are hiring!

Dedicated Game Server Architecture

Dedicated Game Server Reference Architecture

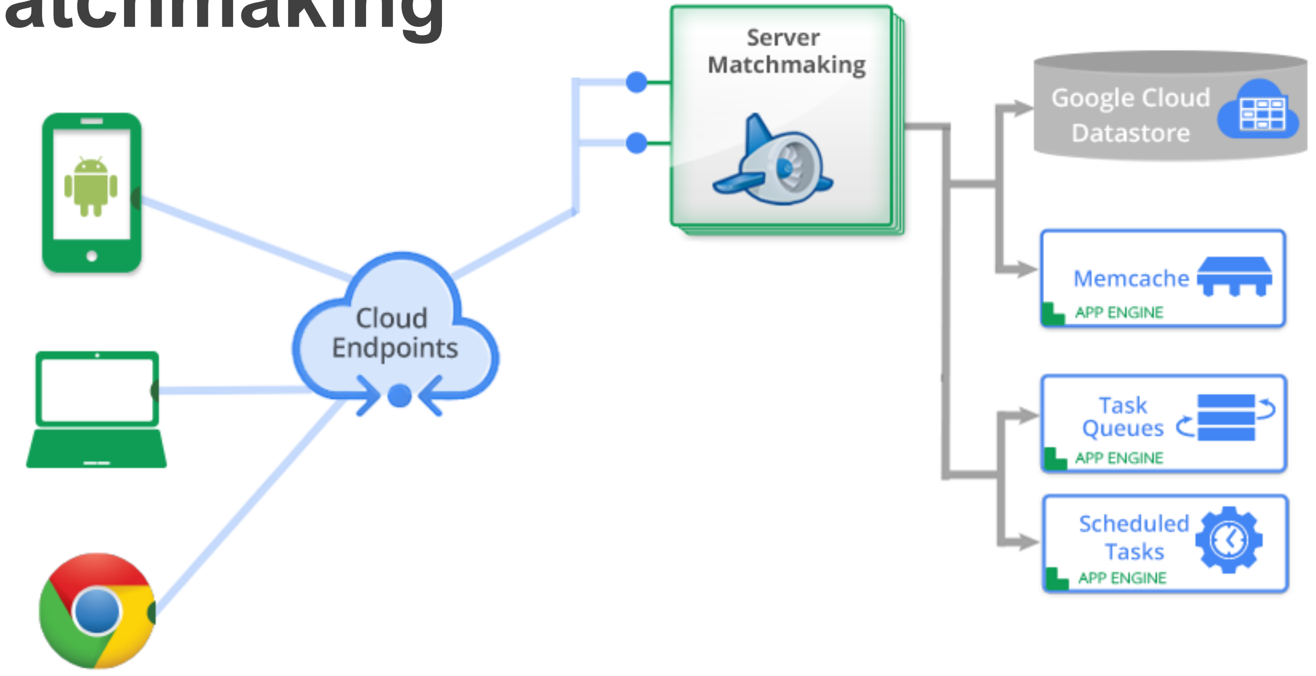
- Real-Time Multiplayer with Dedicated Servers
- Scalable to Millions of Users
- Full Featured Game Experience

App Engine | Compute Engine | Cloud Storage | BigQuery | Cloud SQL

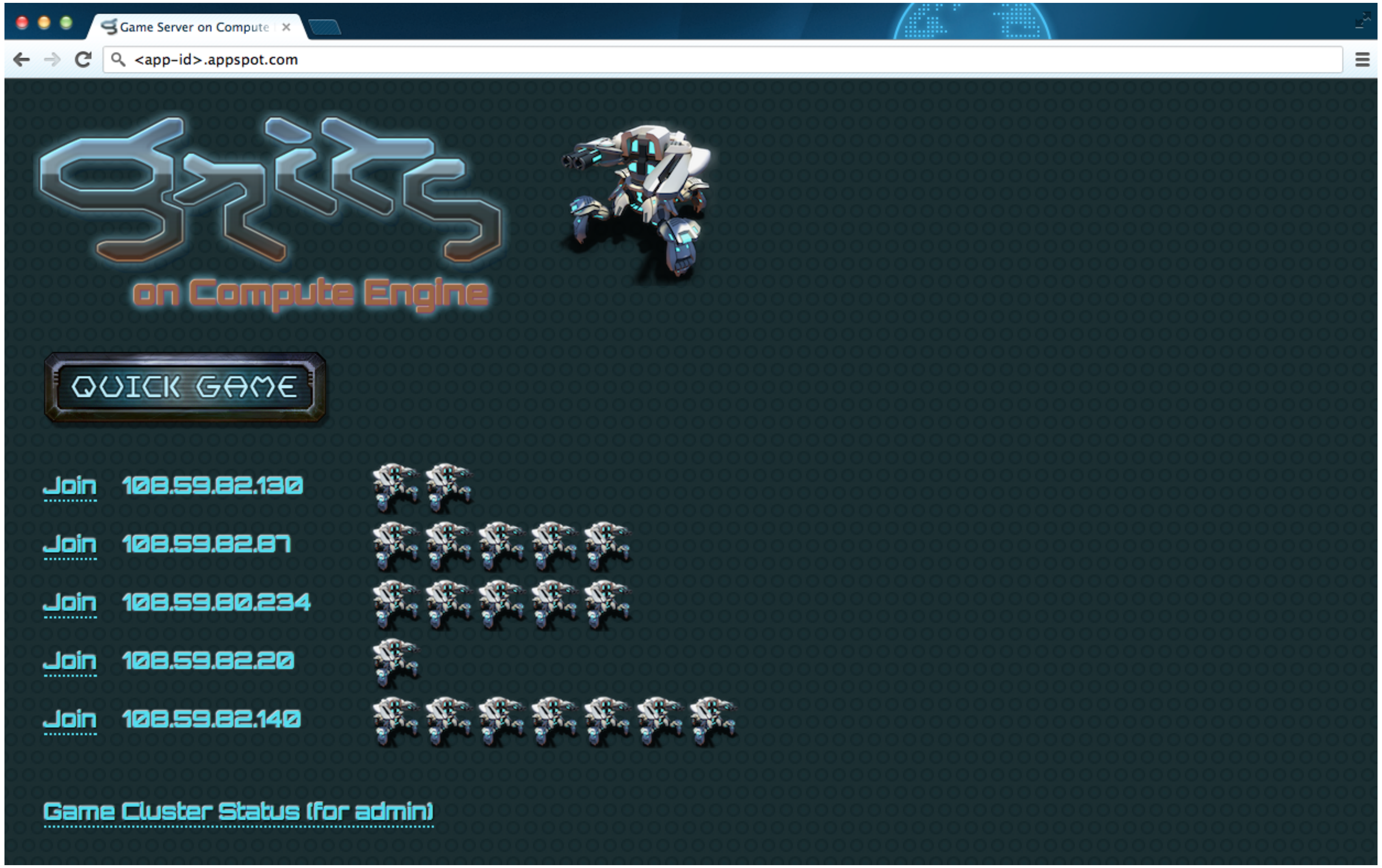
GRITS: PvP Gaming with HTML5

Game Client

Server Matchmaking



GRITS: Server Matchmaking



Join 108.59.82.130

Join 108.59.82.87

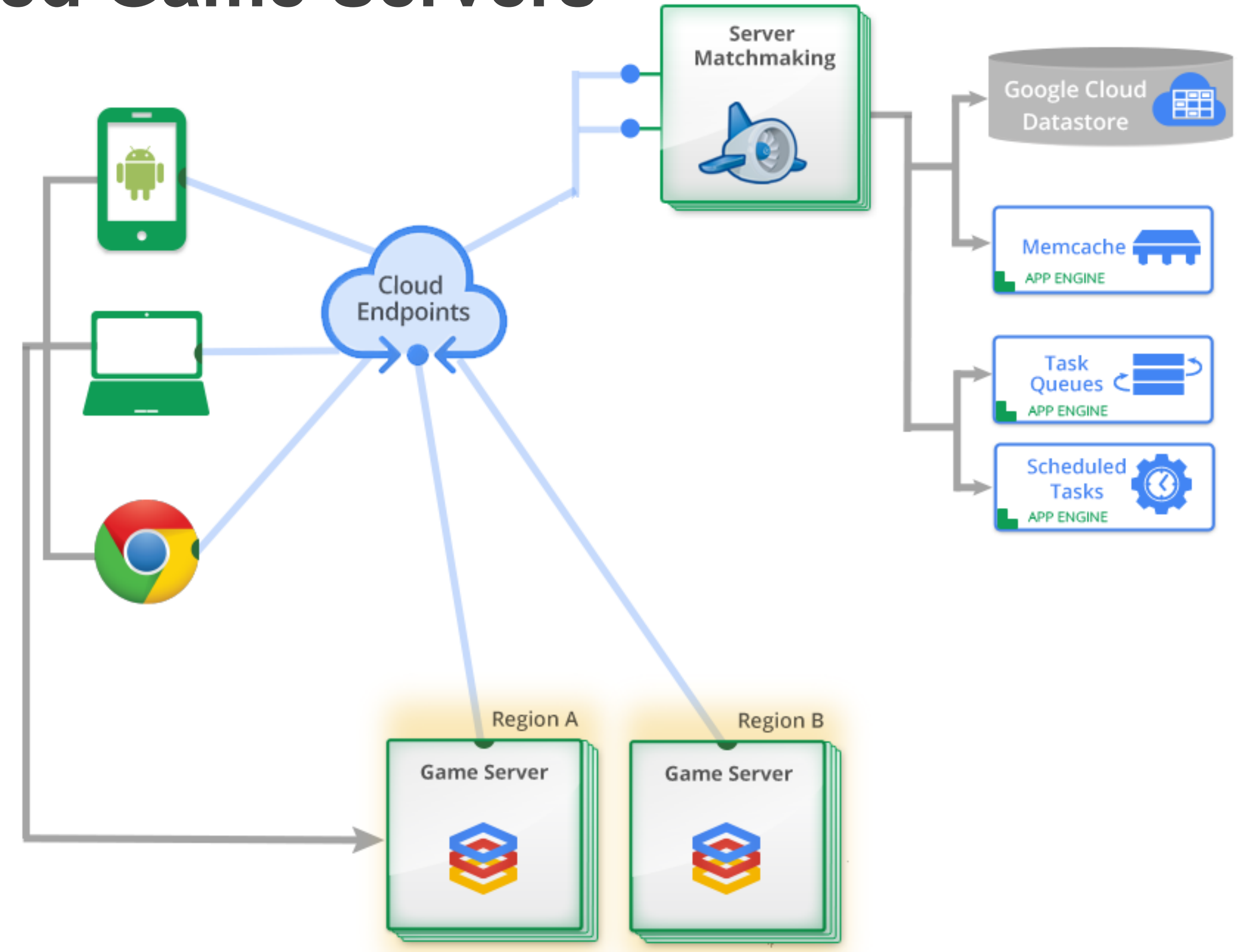
Join 108.59.80.234

Join 108.59.82.20

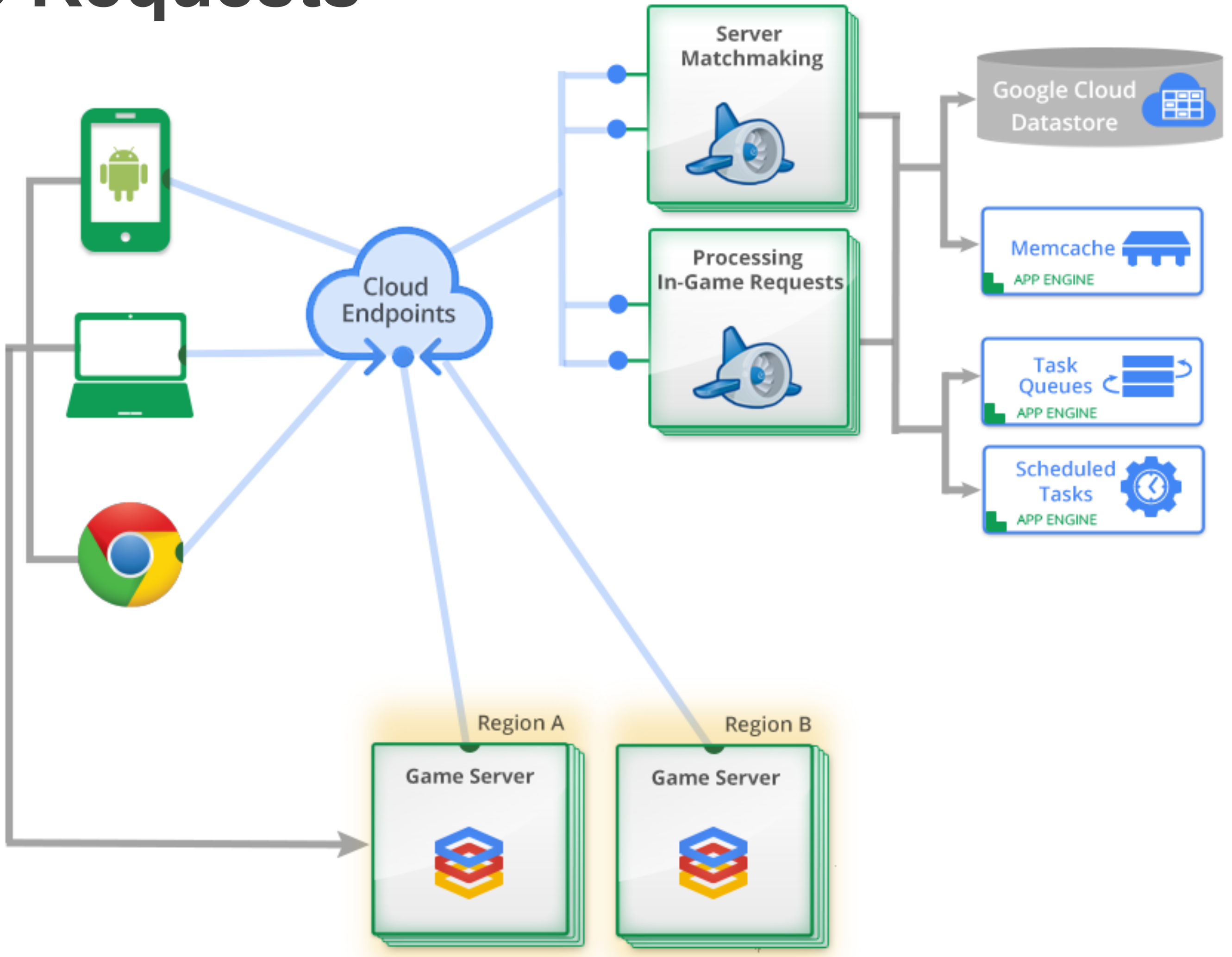
Join 108.59.82.140

Game Cluster Status (for admin)

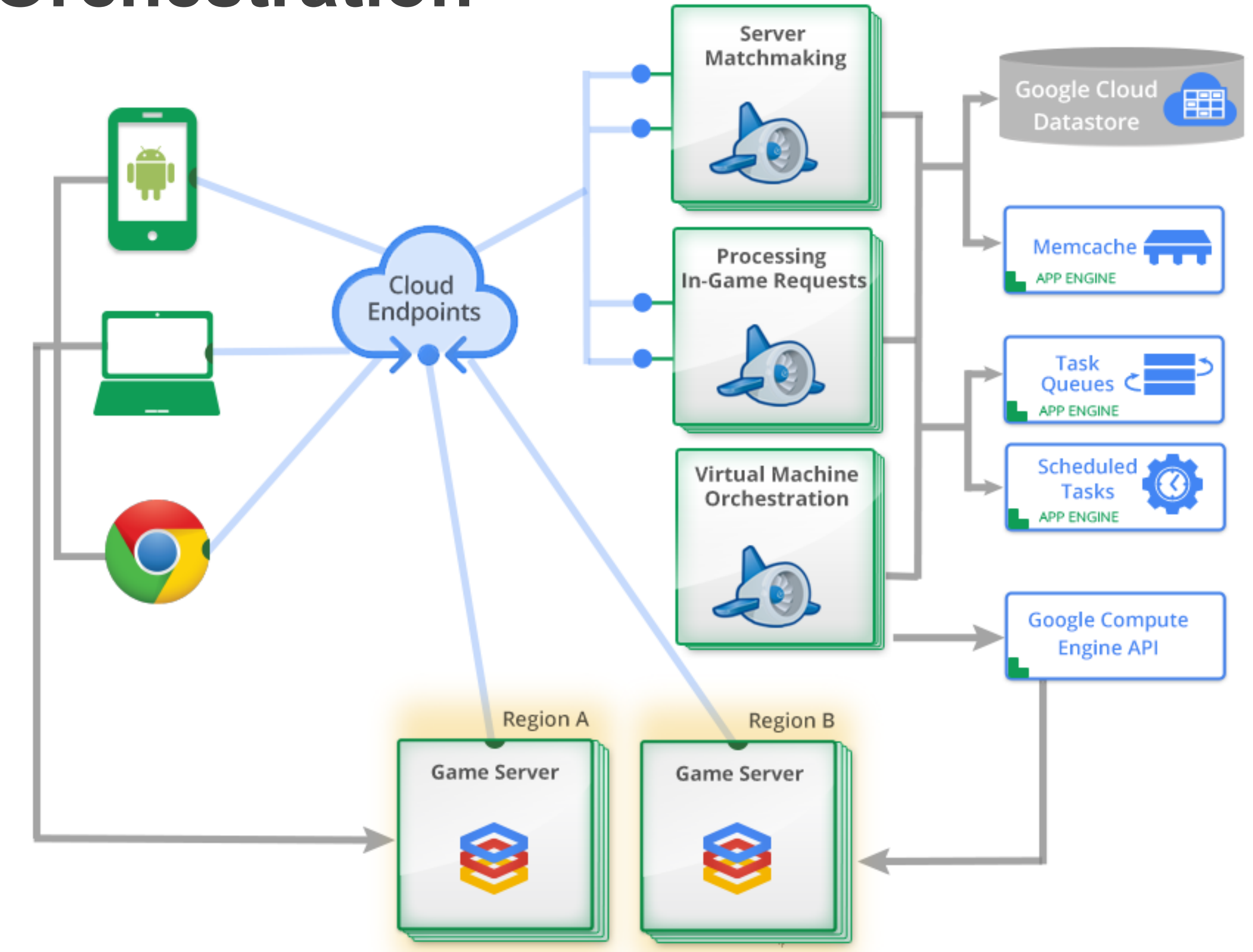
Dedicated Game Servers



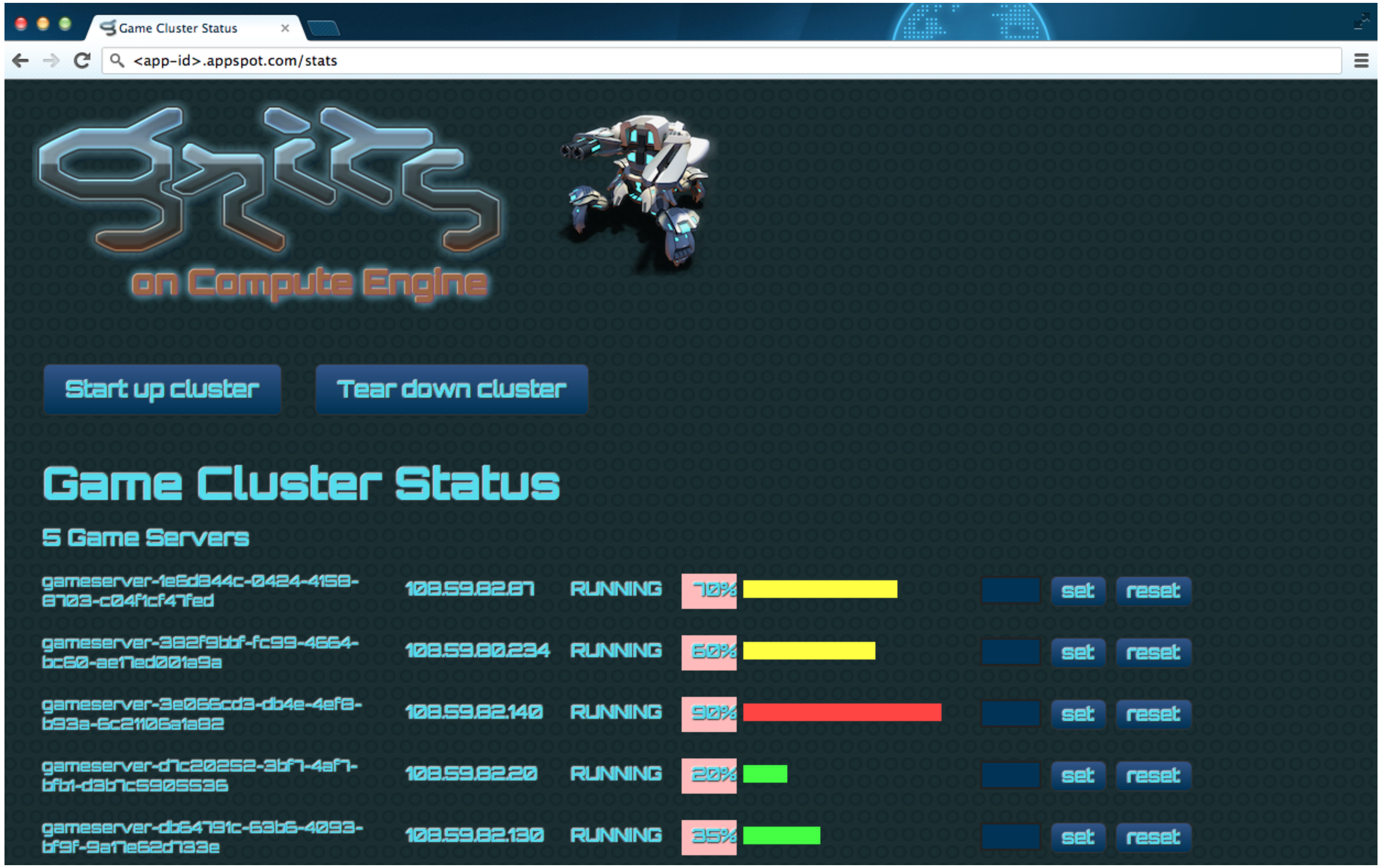
In-Game Requests



Server Orchestration



GRITS: Server Orchestration

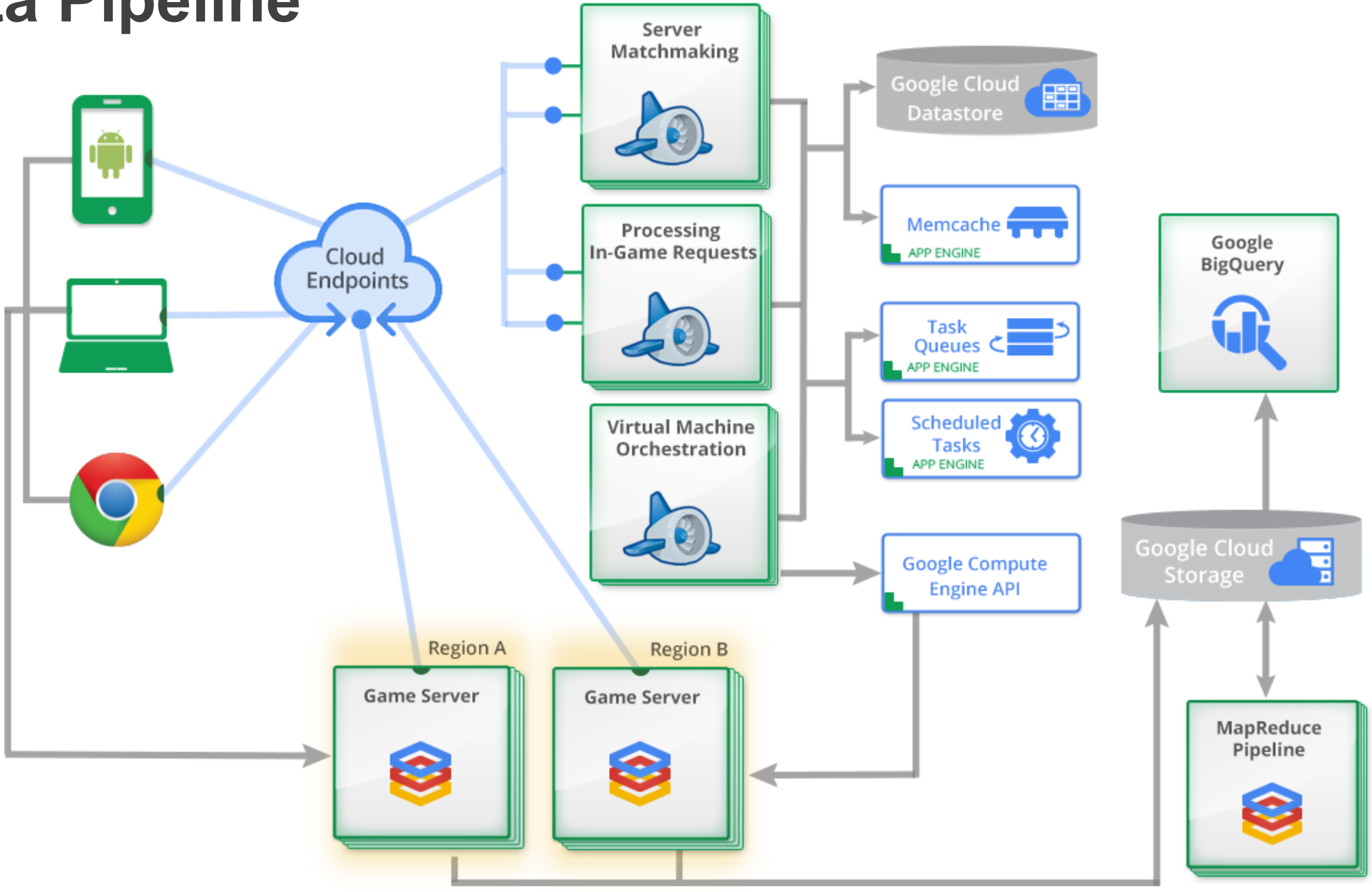


Game Cluster Status

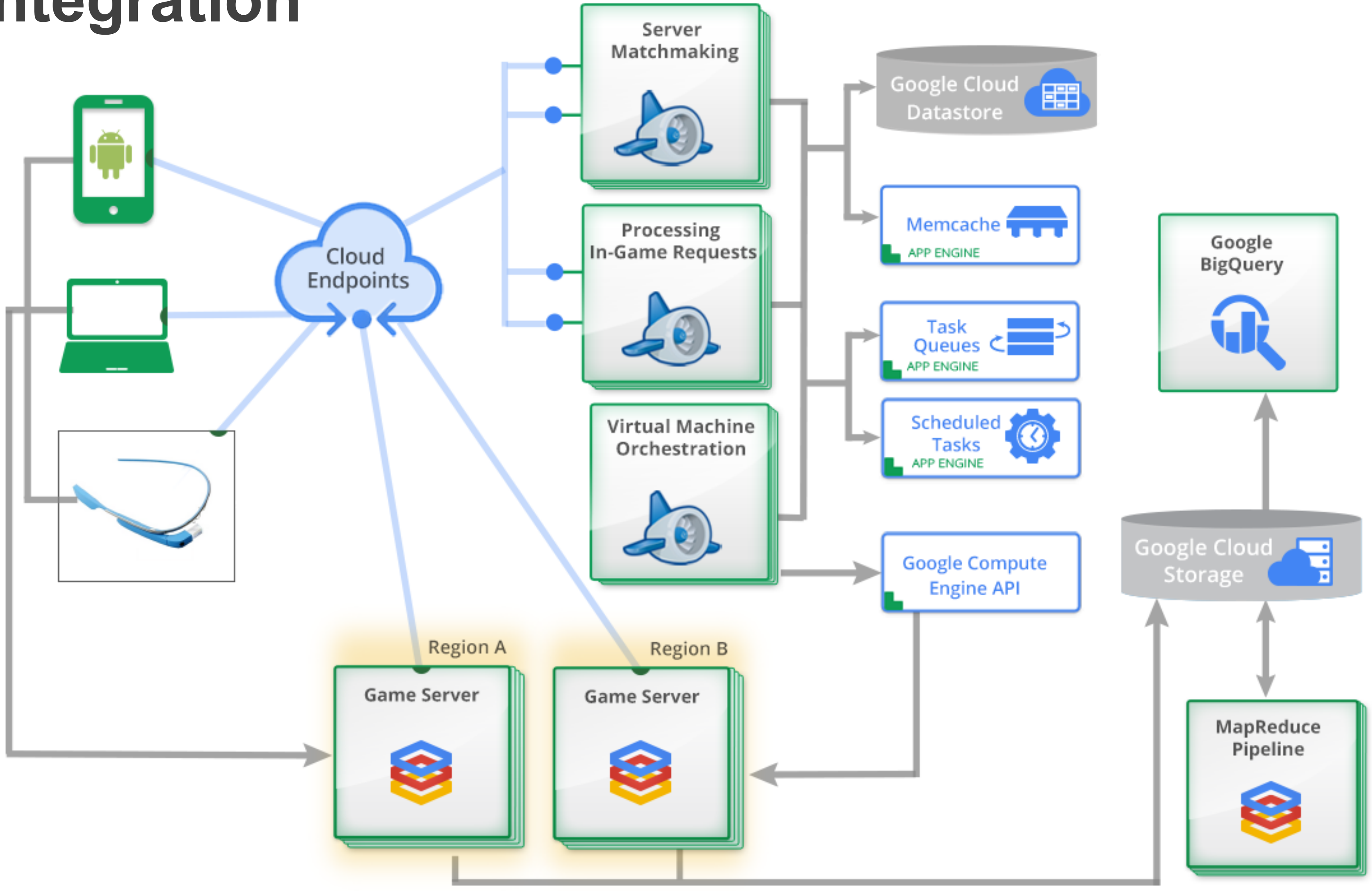
5 Game Servers

gameserver-1e6d844c-0424-4158-8703-c04f1cf47fed	108.59.82.87	RUNNING	70%
gameserver-382f9bbf-fc99-4664-bc60-ae17ed001a9a	108.59.80.234	RUNNING	60%
gameserver-3e066cd3-db4e-4ef8-b93a-6c21106a1a82	108.59.82.140	RUNNING	90%
gameserver-d7c20252-3bf7-4af7-bfb1-d3b7c5905536	108.59.82.20	RUNNING	20%
gameserver-db64791c-63b6-4093-bf9f-9a17e62d733e	108.59.82.130	RUNNING	35%

Big Data Pipeline

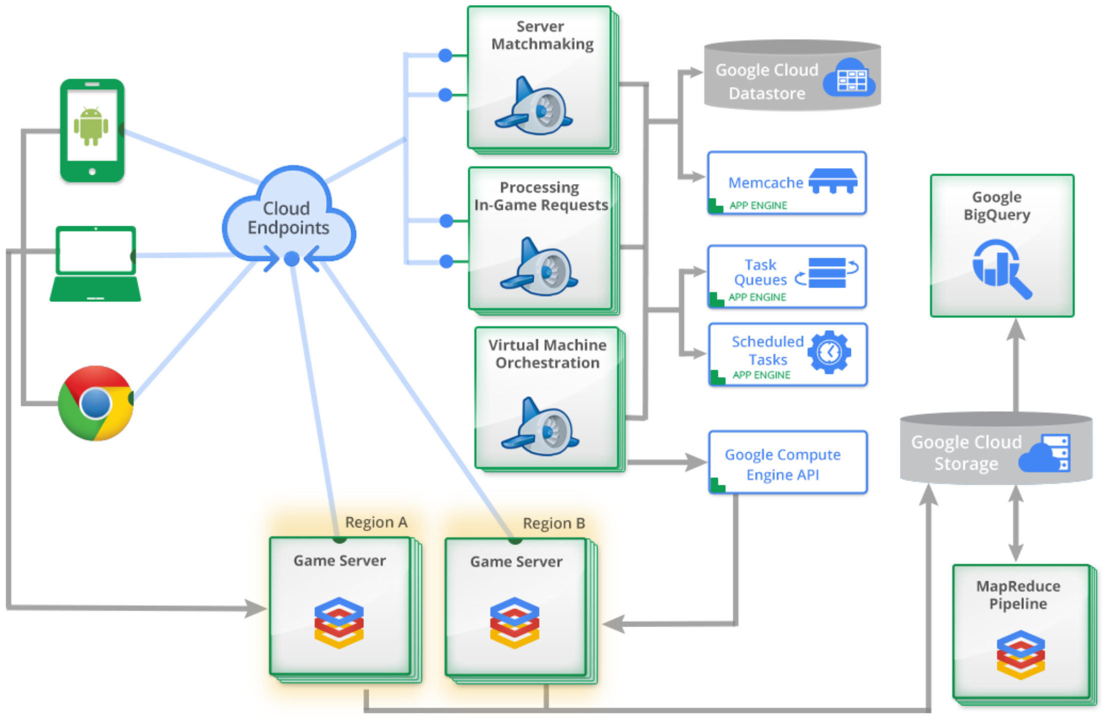


Glass Integration



Build a Full Featured Game Experience

- User profile and configuration
- Player customization
- Event stream and news feed
- Market place and daily promotions
- Friends and favorites



Wrap Up

Best Place for Cutting Edge Developers

From Indie to AAA Game Studios, Google has the Platform to Build On

App Engine Compute Engine Cloud Storage BigQuery Cloud SQL

Architecture and Sample Resources

- Solution Papers
  https://cloud.google.com/resources/tutorials-articles

- Sample Applications
  https://github.com/googlecloudplatform

Relevant I/O Sessions

- **Today**
  - 3:30 PM - From Nothing to Nirvana in Minutes: Cloud Backend for Your Android Application
- **Tomorrow**
  - 9 AM - Keys to the Kingdom: Design Patterns for Using OAuth in the Cloud
  - 10 AM - Here Be BigQuery: Building Social Gaming Infrastructure on the Google Cloud Platform
- **Yesterday**
  - JAM with Chrome: How We Built a Massive Multiplayer Music Application Using Only Web Technology

Thank You! Questions?

chriselliott@google.com
http://about.me/elliottchris