

# Building Mobile Backends

With the Mobile Backend Starter

[about.me/mandywaite](https://about.me/mandywaite) [[@tekgrrrl](https://twitter.com/tekgrrrl)]





# Mobile Backend Starter: Sample

alexismp - January 20, 2014

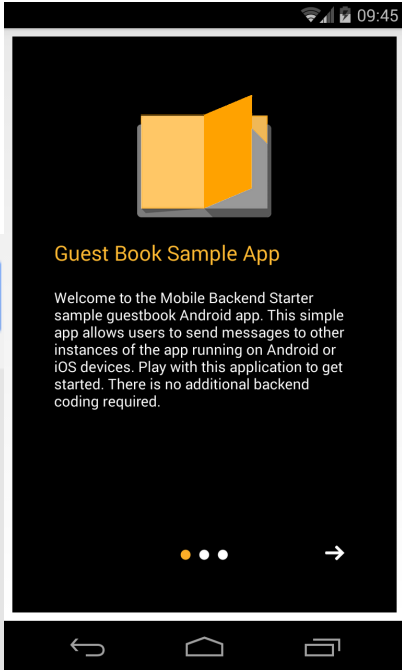
Libraries & Demo

Installed

*i* This app is compatible with all of your devices.

★★★★★ (2)

**g+** Recommend this on Google



<http://goo.gl/Tf0o6J>

# Done !



<http://goo.gl/Tf0o6J>

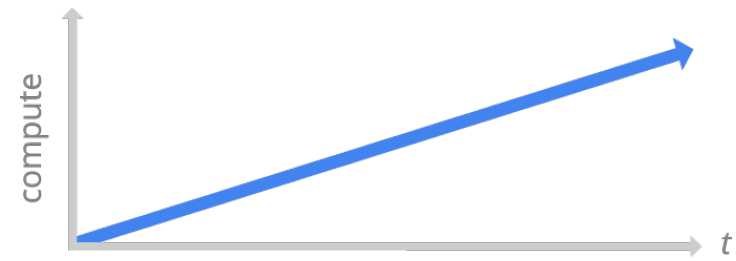
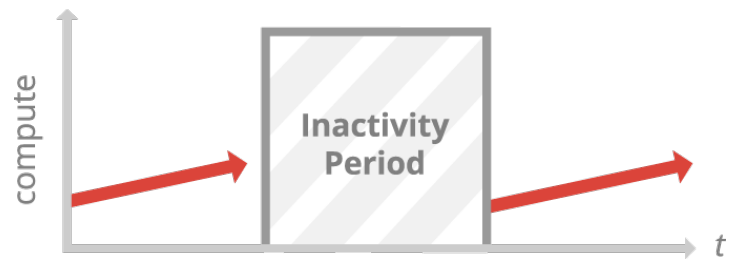
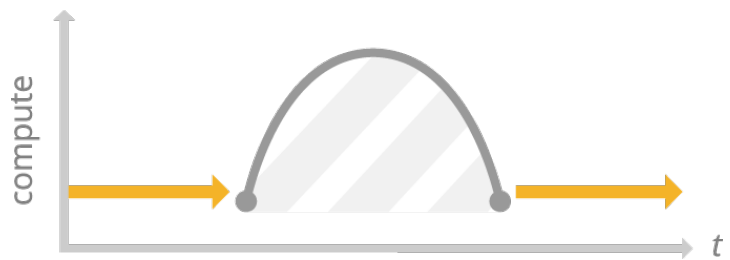
# Yeah but server-side is {boring|hard} !

```
newPost = CloudEntity ("GuestBook"); // onClick method  
newPost.put ( "message", etMessage.getText().toString() );  
getCloudBackend().insert(newPost);
```

Does this seem simple enough?







**Storing Data**

*(preferences, shared state)*

**Sharing messages**

*(between mobile users)*

# why build mobile backend services when we can build them for you?

**Authentication**

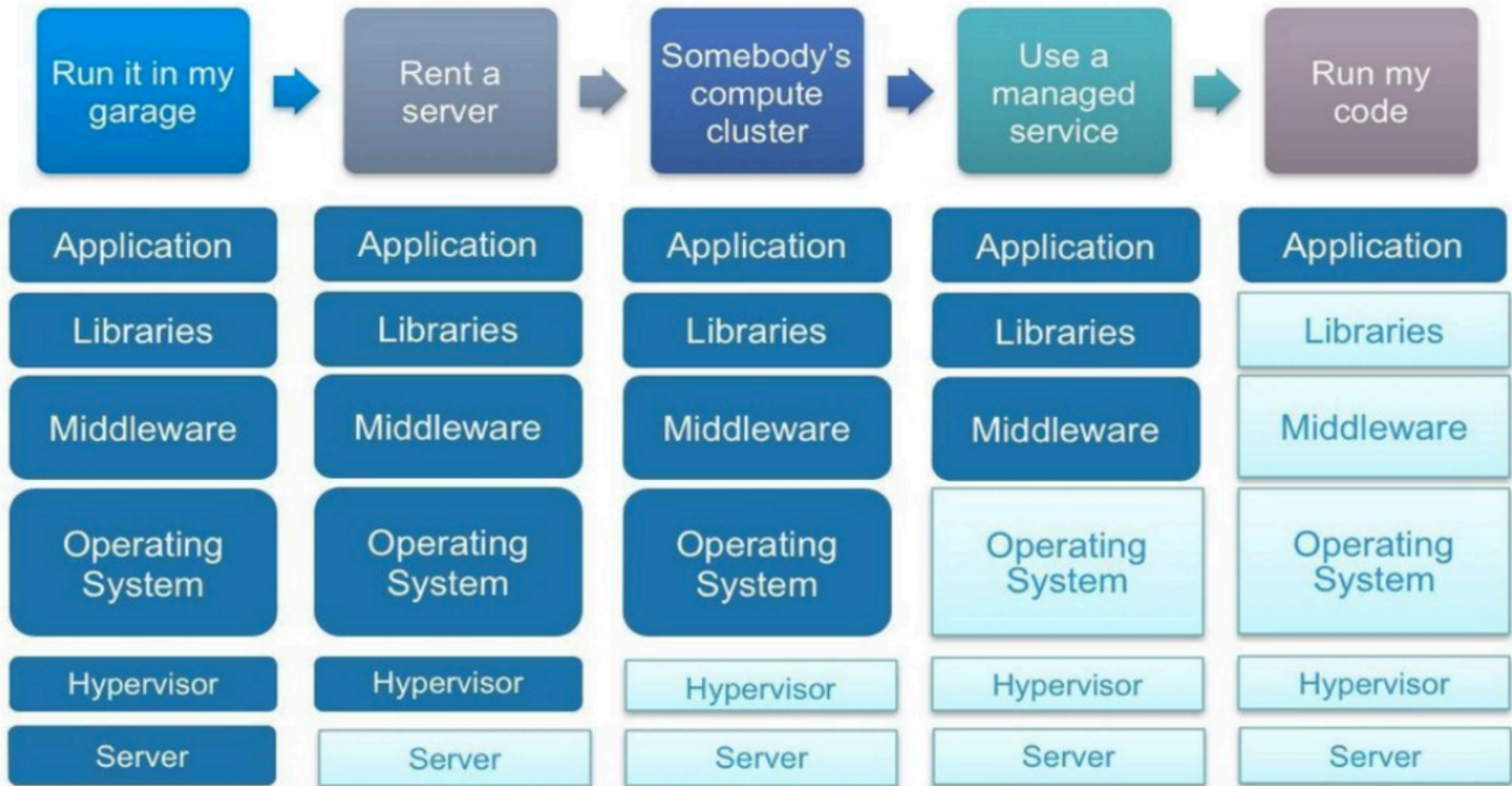
*(OAuth 2)*

**Event Notification**

*(state change, client or server)*

**AutoScale**

*(app engine)*



Life of a Startup ...

# App Engine



---

## Simple to Scale

- AutoScale



---

## Easy to develop

- Free to start
- Local dev environment
- Service abstractions

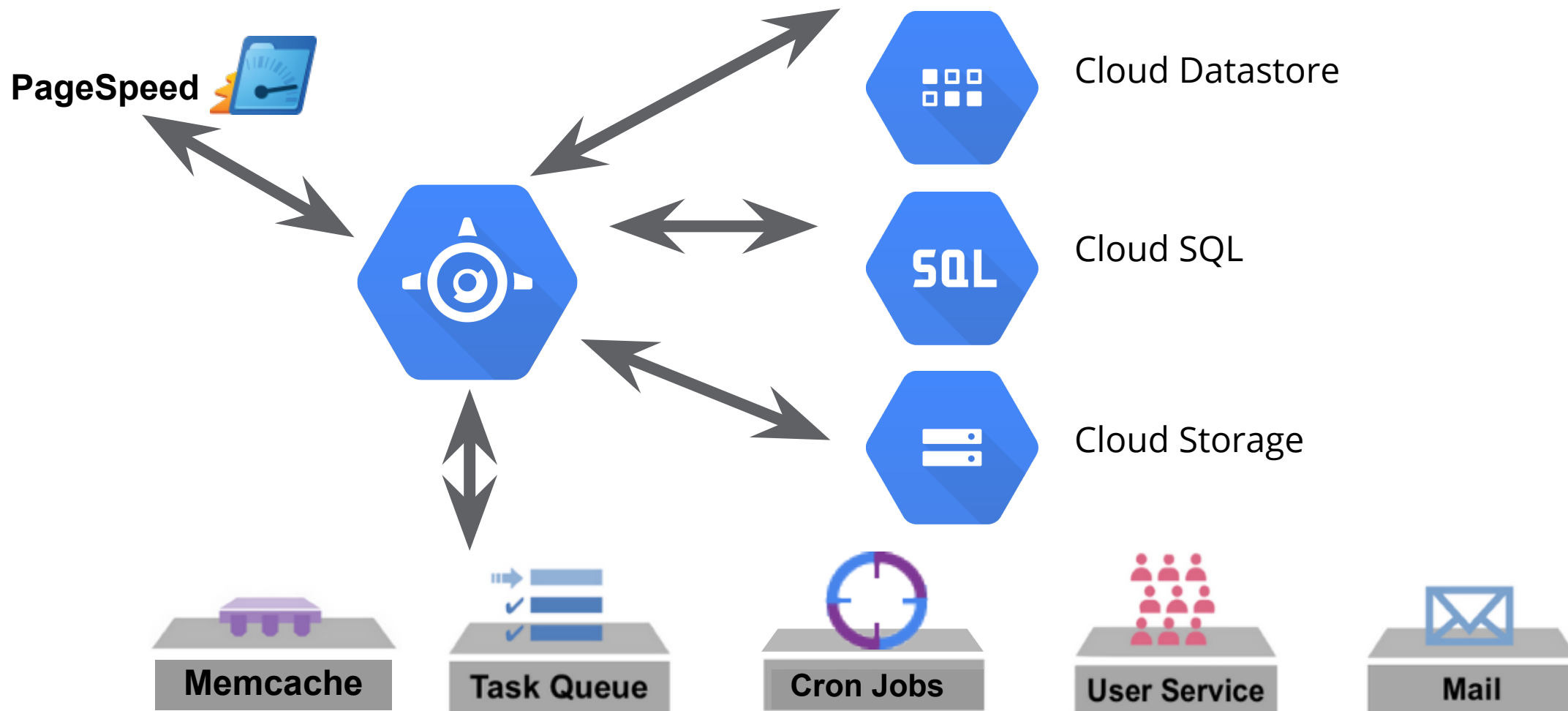


---

## Trivial to manage

- Fully managed
- No patches
- 24x7 operation by Google **SREs**

# App Engine's Cloud Scale Services



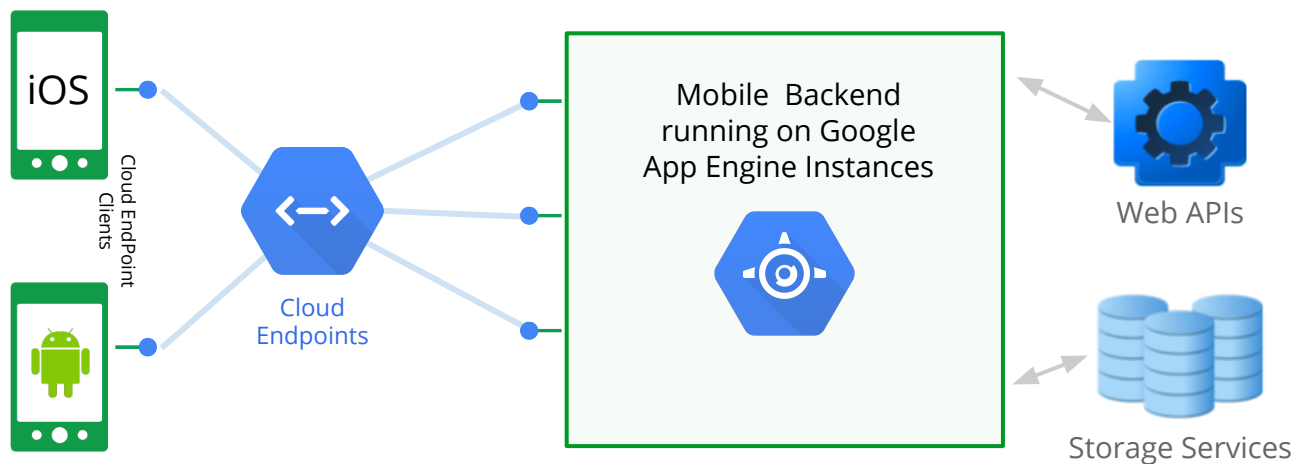
...and more



# Google Cloud Platform Mobile Backend Services

# Cloud Endpoints

## APIs for Mobile Backends Made Easy



- Create APIs for Mobile Apps to communicate with Mobile Backends
- Add annotations to client interface application code or generate Endpoint classes from Models.
- Discoverable, Restful APIs implemented on top of Google's API Infrastructure
- Tools available for generating Client Libraries for Android, iOS and JavaScript
- Built-In Authentication Support



# Cloud Endpoints - Custom API

## Submit Game Answers

```
@Api(name = "gameendpoint", version = "V1")  
public class GameEndpoint {
```

Java - App Engine

```
    @ApiMethod(httpMethod = "PUT", path = "game/{id}/answers")  
    public void submitAnswers(@Named("id") long gameId, GameplayStatus answers, User player)  
        throws UnauthorizedException, NotFoundException {  
        if (user == null)  
            throw new UnauthorizedException("The user is not authorized.");  
        // do important work here  
    }  
}
```

```
protected void executeEndpointCall() {  
    service.gameEndpoint().submitAnswers(gameId, answers).execute();  
}
```

Java - Android App

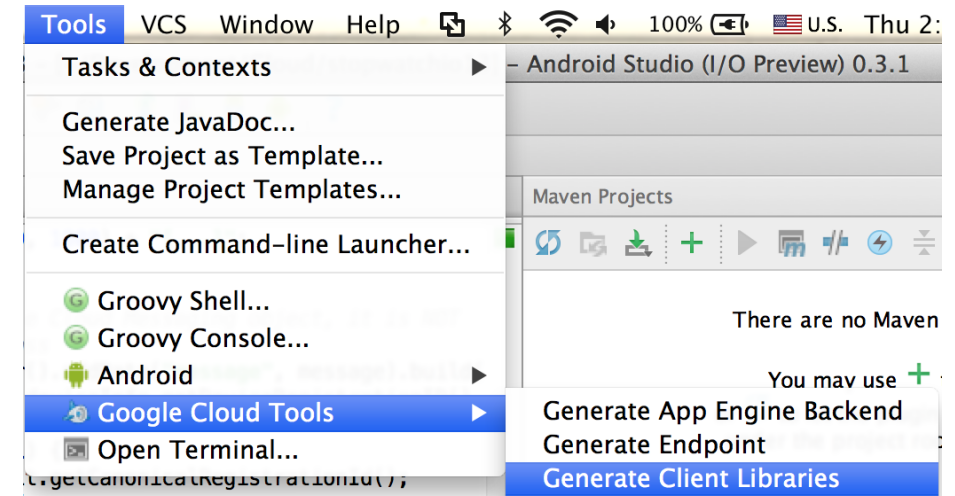
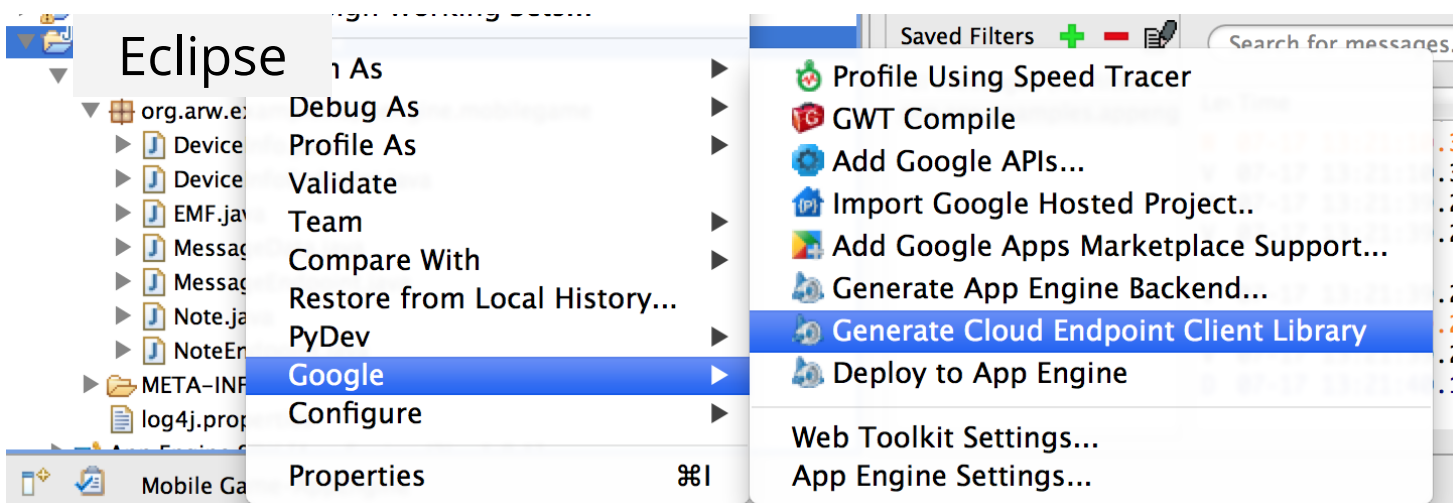
# Cloud Endpoints Development

## Generate Client Library

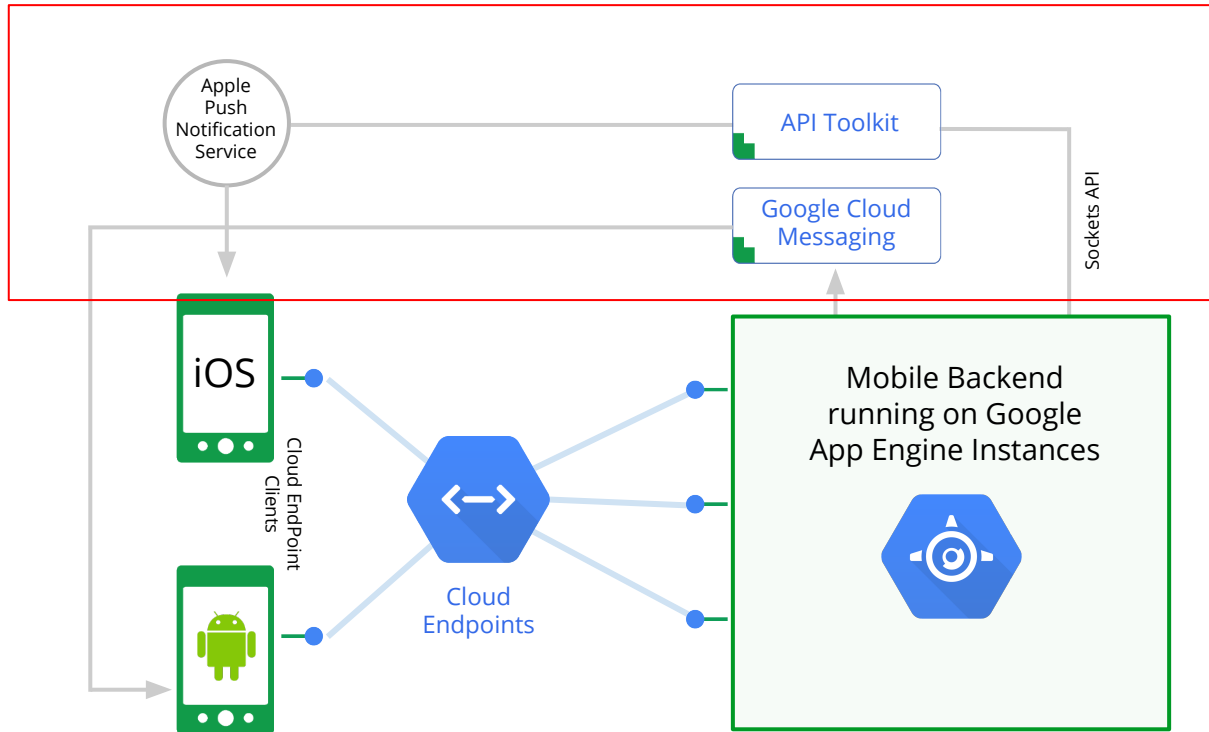
Java - App Engine

```
Mobile Game-AppEngine $ endpoints.sh get-client-lib org.arw.examples.appengine.mobilegame.GameInviteEndpoint
```

```
Jul 17, 2013 3:05:32 PM com.google.apphosting.utils.config.AppEngineWebXmlReader readAppEngineWebXml
INFO: Successfully processed ./war/WEB-INF/appengine-web.xml
API configuration written to ./gameinviteendpoint-v1.api
API Discovery Document written to ./gameinviteendpoint-v1-rpc.discovery
API Discovery Document written to ./gameinviteendpoint-v1-rest.discovery
API client library written to ./gameinviteendpoint-v1-java.zip
```

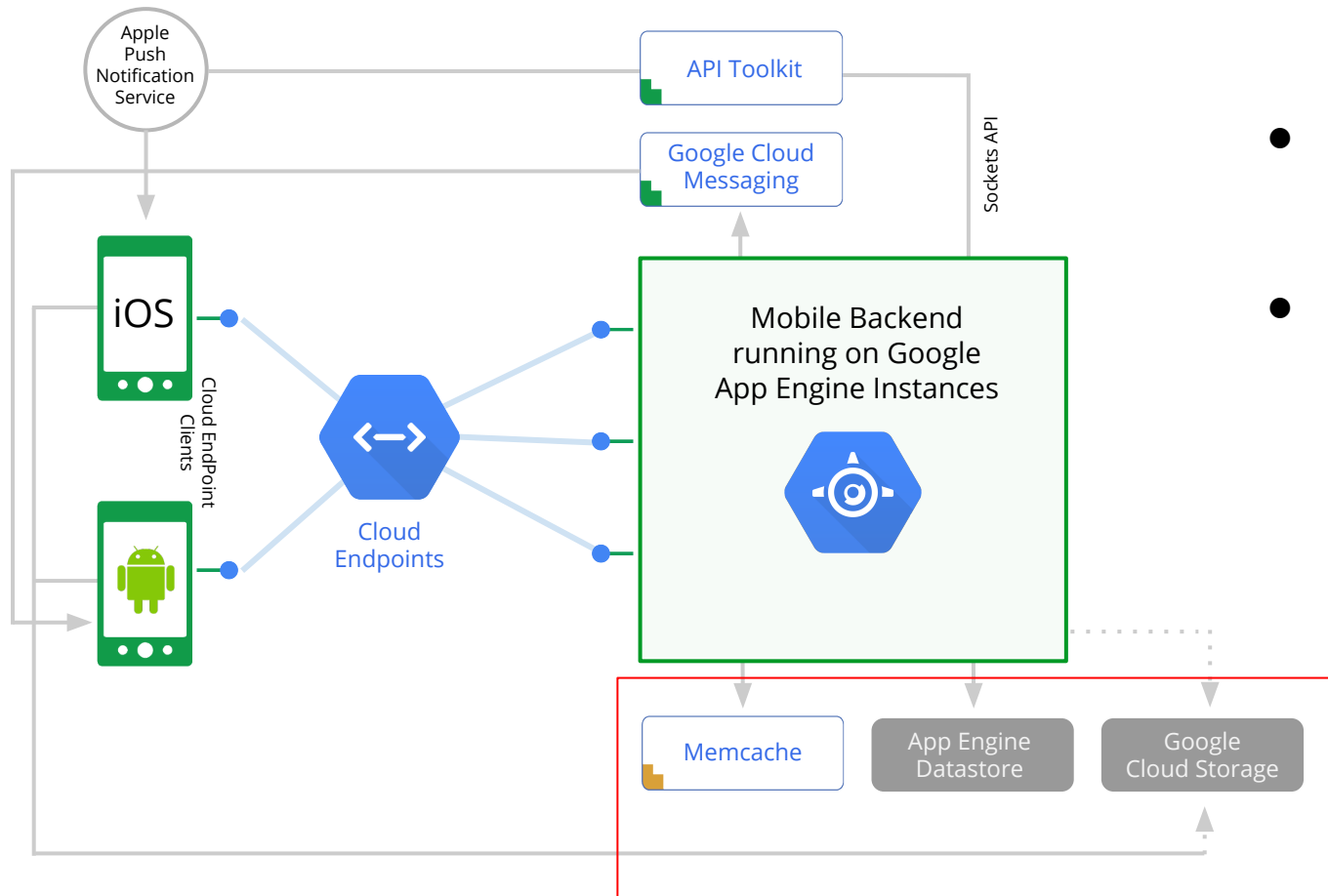


# Push Notifications



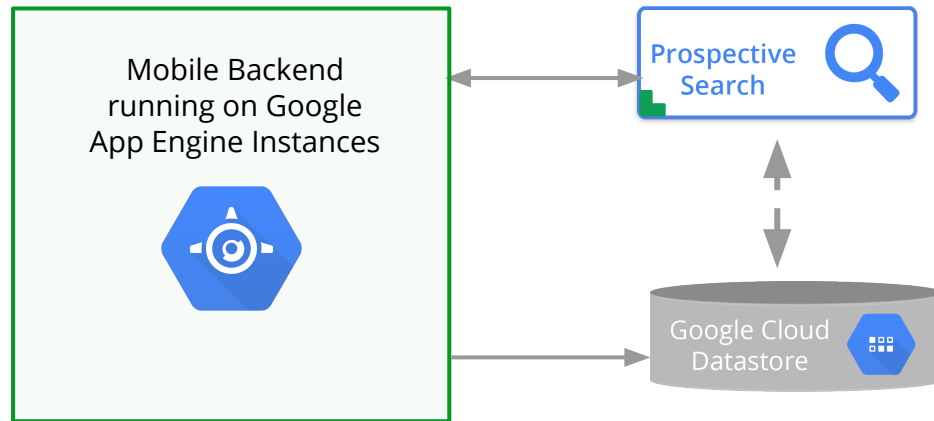
- App Engine supports mobile device messaging
  - Apple Push Notifications (using Sockets API)\*
  - Google Cloud Messaging (using GCM Server)
- Examples
  - Notification of new emails on Server
  - Send game invites from friends
  - Display 'toasts' with important messages

# Data and Object Storage



- Cloud Storage
  - App Generated Content
  - Large Binary Objects
  - Served Directly via URL
- Cloud DataStore
  - Finer grained properties (Device registration, in-app purchases, etc)
- Memcache
  - Improved performance and scalability
  - Lower costs

# Prospective Search



- Inverse of traditional search engines
  - Which Queries would retrieve this document?
- Queries registered in advance of documents (hence **Prospective**)
- Documents
  - Python dictionary
  - datastore.Entity
  - db.Model derived class
- Query Rules
  - Simple Queries: 'rose'
  - Field Queries: 'author:rose'
  - Search Operators
    - Numeric
    - AND, OR, NOT



# Mobile Backend Starter (Android)

# What is the Mobile Backend Starter?

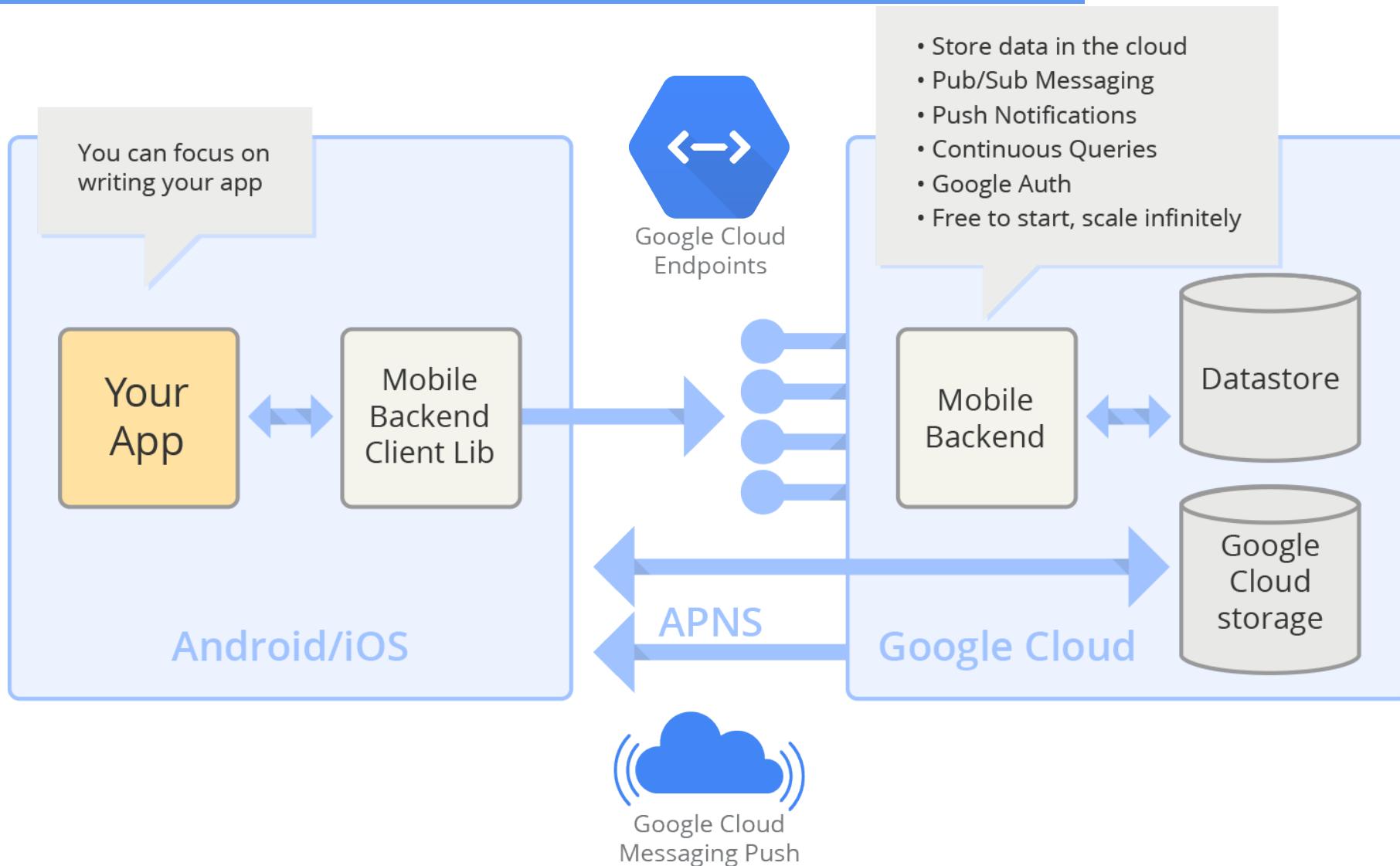
Ready to deploy, sample Cloud Backend

+

Sample Client side Framework (iOS and Android)

<https://developers.google.com/cloud/samples/mbs> (<http://goo.gl/VIbmV>)

# Mobile Backend Starter Features







# Mobile Backend Starter Demo

# Create a Cloud Project

cloud.google.com/console

Google Developers Console

Projects

CREATE PROJECT

New Project

Project name ?

My Mobile Backend

Project ID ?

animated-flow-374

Create

Cancel

Welcome. Not sure what to do next?

Get started with...



[Photofeed Java sample app](#)

A simple web app run on App Engine, Cloud Storage and your choice of Cloud Datastore or Cloud SQL.



[Mobile Backend Starter](#)

A ready to deploy, general purpose cloud backend with Android and iOS client libraries.

# Backend Setup

## 1 Deploy the backend

### Mobile Backend Starter

**Note:** This project is designed for Android and iOS developers.

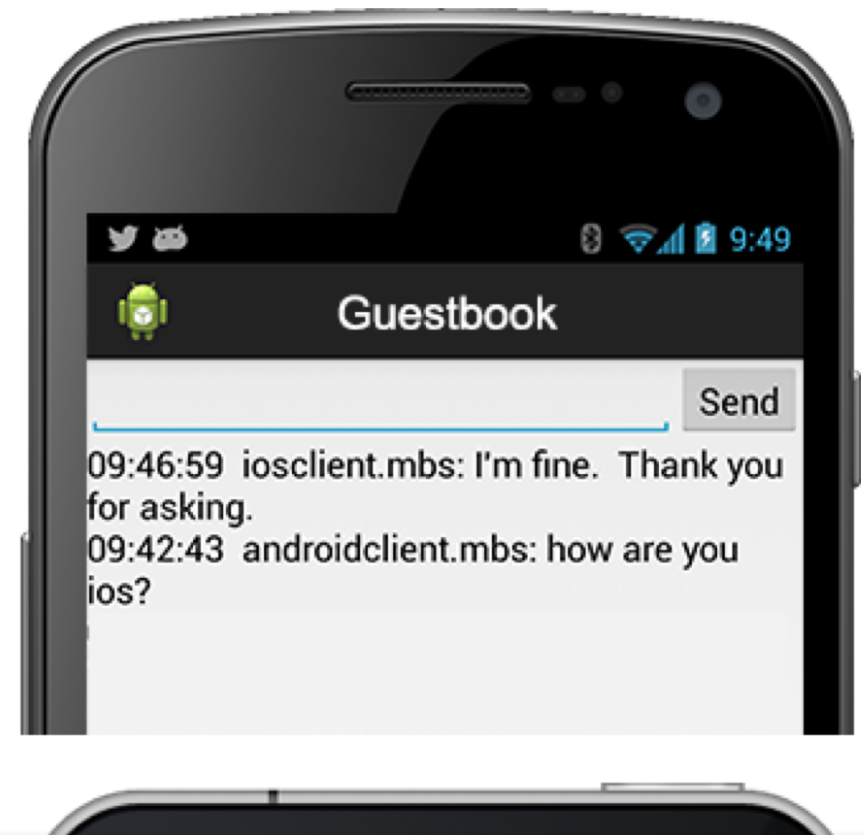
This sample is a ready-to-deploy backend and client libraries for Android and iOS to power your mobile app ([Learn more](#)):

- Store objects to the cloud from a mobile device
- Send real-time messages between devices
- Subscribe to changes in data stored on the cloud
- Authenticate users with Google accounts authentication

Deployment steps:

#### 1 Deploy the backend

Deploy





# Testing the API

← → ↻

**Google**

## APIs Explorer

- Services**
- All Versions
- Request History

	APIs Discovery Service	v1	Lets you discover information about other Google APIs, such as what APIs are available, the resource and method details for each API.
	mobilebackend API	v1	This is an API

# Testing the API

Services > mobilebackend API v1 > mobilebackend.endpointV1.get

**kind**  (string)

**id**  (string)

This parameter was URL encoded.

**fields**  Selector specifying wh  
[Use fields editor](#)

**bold red** = required

Execute

mobilebackend.endpointV1.get executed moments ago time to execute: 1767 ms

## Request

```
GET https://[REDACTED].appspot.com/_ah/api/mobilebackend/v1/CloudEntities
Authorization: Bearer [REDACTED]
X-JavaScript-User-Agent: Google APIs Explorer
```

## Response

200 OK

- Show headers -

```
-{
  "id": "CE:03e984ae-1939-4727-b0eb-bb5fbde8e0d1",
  "createdAt": "2014-02-01T20:59:00.083Z",
  "updatedAt": "2014-02-01T20:59:00.083Z",
  "createdBy": "mandywaite@google.com",
  "updatedBy": "mandywaite@google.com",
  "kindName": "Guestbook",
  "properties": {
    "message": "hello"
  },
  "owner": "USER:[REDACTED]",
  "kind": "mobilebackend#endpointV1Item",
  "etag": "\"\`RK6JF8e5PpU17QgLVQOf9GSELWs/TnxIOSjJpKAbYVW2vvOQXEardBg\`\""
}
```

# Backend Admin

2 Enable the backend to accept incoming requests via [Settings](#)

## Mobile Backend settings

This page lets you configure the authentication and other options for your mobile backend. If you don't have a client application yet, download the [Android](#) or [iOS](#) sample client application.

### Authentication / Authorization

Locked Down (Access disabled)

All requests will be rejected.

Open (for development use only)

All unauthenticated requests will be allowed. The backend will not be taking advantage of the integrated authentication to know the identity of the callers. For Android sample client app you can use either emulator or a physical device.

Secured by Client IDs (Recommended)

### Google Cloud Messaging and iOS Push Notification

Disabled

Enabled

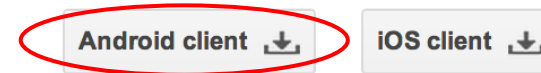
Save

# Android Sample Client Project

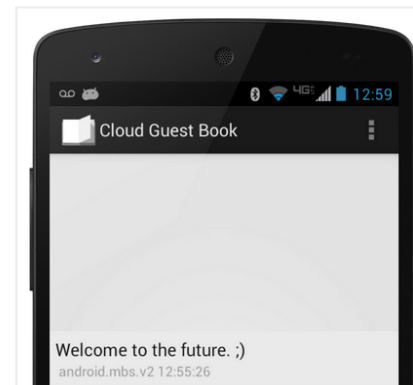
## 3 Open a mobile client

1. Next, download the **Android client project**
2. Requires Android level 15 and above with Google API, including GCM
3. Open the project in your Android IDE. Locate the Consts.java file and set the PROJECT\_ID to that of your Cloud Project
4. Now just build and run the project and you have a very simple cloud enabled Android application.

### 3 Open a mobile client:



```
/**  
 * Set Project ID of your Google APIs Console Project.  
 */  
public static final String PROJECT_ID = "river-pointer-309";
```



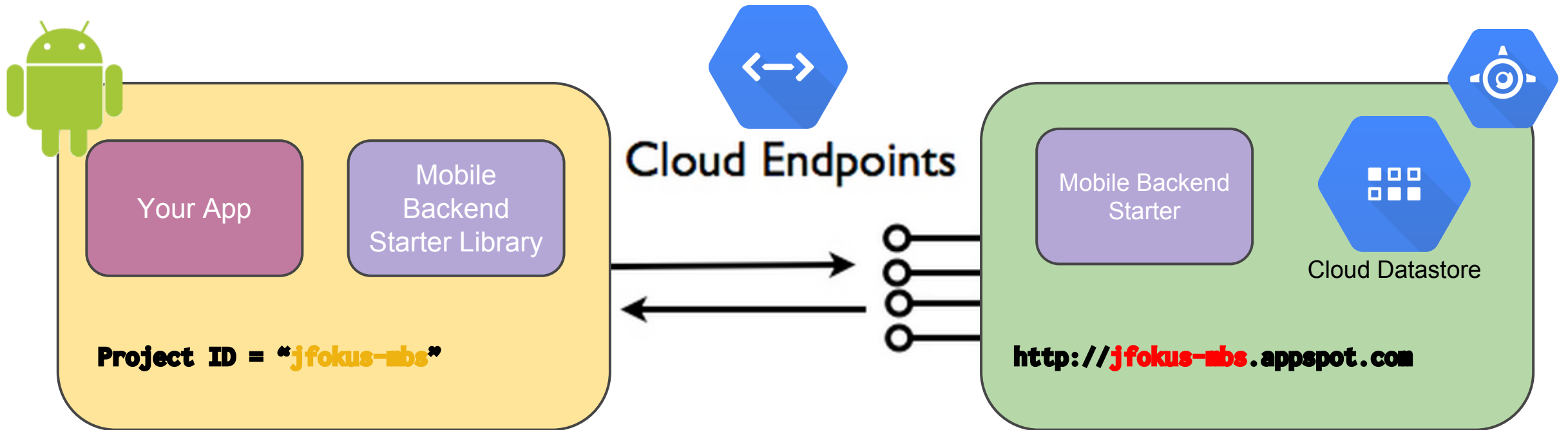
# Client Setup

Android - Consts.java

```
public static final String PROJECT_ID = "jfokus-mbs";  
/**  
 * Set Project Number of your Google APIs Console Project.  
 */  
public static final String PROJECT_NUMBER = "*** ENTER YOUR PROJECT NUMBER ***";  
/**  
 * Set your Web Client ID for authentication at backend.  
 */  
public static final String WEB_CLIENT_ID = "*** ENTER YOUR WEB CLIENT ID ***";  
/**  
 * Set default user authentication enabled or disabled.  
 */  
public static final boolean IS_AUTH_ENABLED = false;
```



# Connecting to App Engine



# Notifications: Project Setup

## < Mobile Backend V2

Overview

APIs & auth

APIs

Credentials

Consent screen

NAME

STATUS

BigQuery API



ON

Google Cloud Messaging for Android

ON

Google Cloud SQL

ON

Google Cloud Storage

ON

ON

OFF

## < Mobile Backend V2

Overview

APIs & auth

APIs

Credentials

Consent screen

Notification endpoints

Permissions

### Public API access

Use of this key does not require any user action or consent, does not grant access to any account information, and is not used for authorization.

[Learn more](#)

[CREATE NEW KEY](#)

### Key for server applications

API key

AKIAIOSFODU7KBG...

IPs

Any IP allowed

Activation date

Jan 5, 2014 1:38 PM

Activated by

alexis.mp@gmail.com (you)

Edit allowed IPs

Regenerate key

Delete

# Notifications: Backend Setup

**Google Cloud Messaging and iOS Push Notification**

- Open (for development use only)  
All unauthenticated requests will be allowed. The backend will not be taking advantage of the integrated authentication to know the identity of the callers. For Android sample client app you can use either emulator or a physical device.
- Secured by Client IDs (Recommended)
- Disabled
- Enabled

**Android**

**Google Cloud Messaging API Key** [Learn how to obtain an API Key](#)

[Redacted API Key]

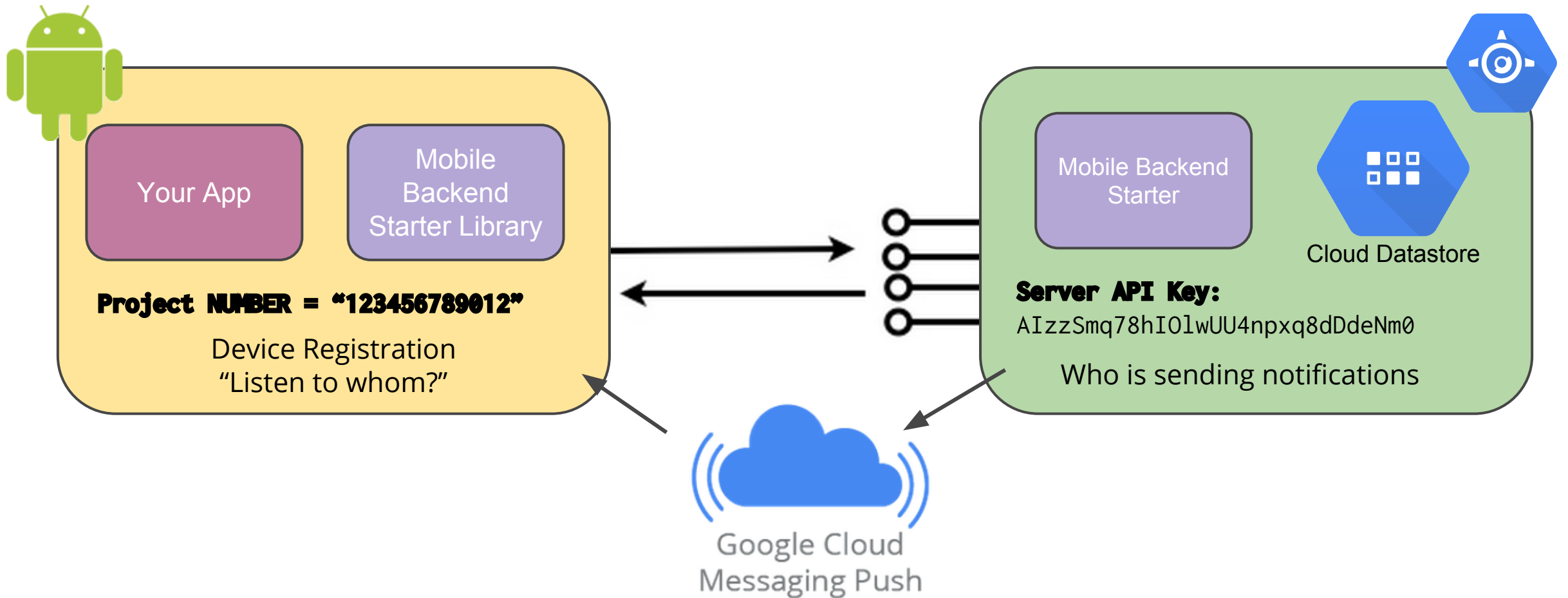


# Client Setup

Android - Consts.java

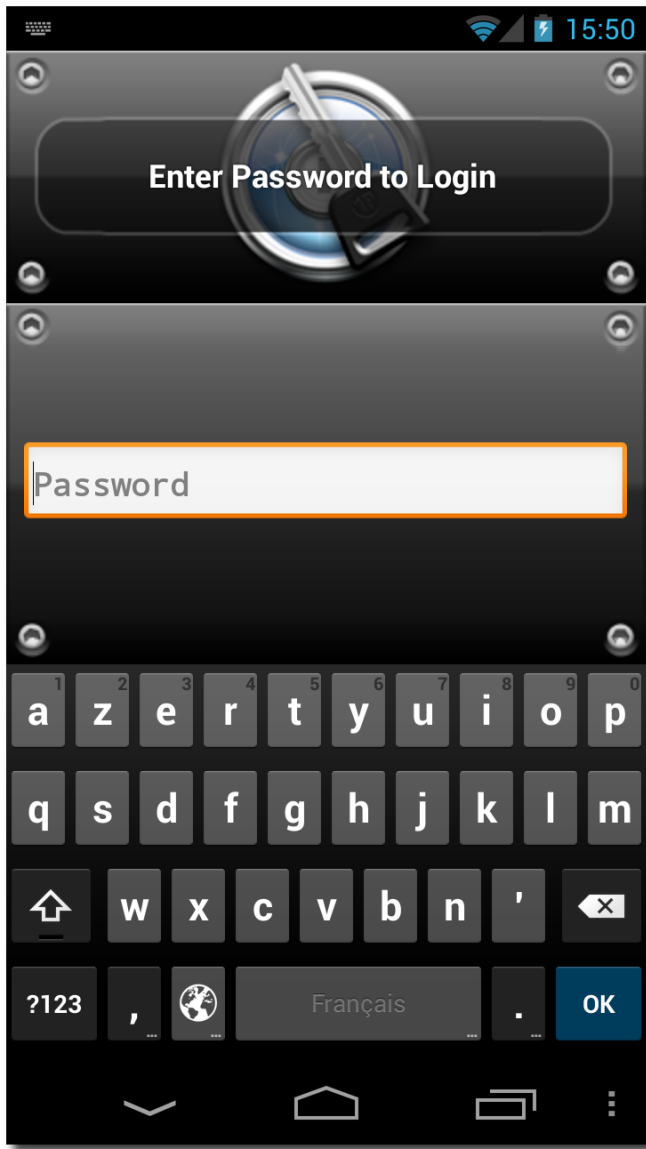
```
public static final String PROJECT_ID = "jfokus-mbs";  
/**  
 * Set Project Number of your Google APIs Console Project.  
 */  
public static final String PROJECT_NUMBER = "123456789012";  
/**  
 * Set your Web Client ID for authentication at backend.  
 */  
public static final String WEB_CLIENT_ID = "*** ENTER YOUR WEB CLIENT ID ***";  
/**  
 * Set default user authentication enabled or disabled.  
 */  
public static final boolean IS_AUTH_ENABLED = false;
```

# Notifications: Identifying the Server

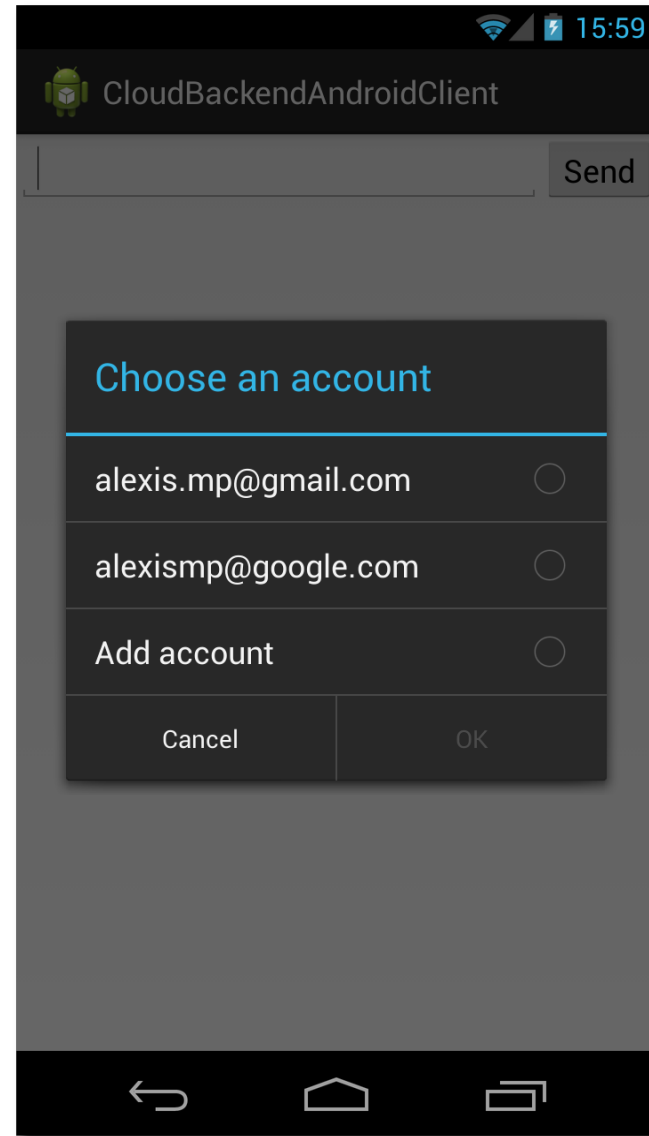


# Authentication

- Establish a strong identity for our Android Client
- Lock down access to the backend APIs to our Android Client
- Associate users with their messages when persisted in the Datastore
- Share user data across multiple devices
- Share messages with a selected set of users



Hasbeen

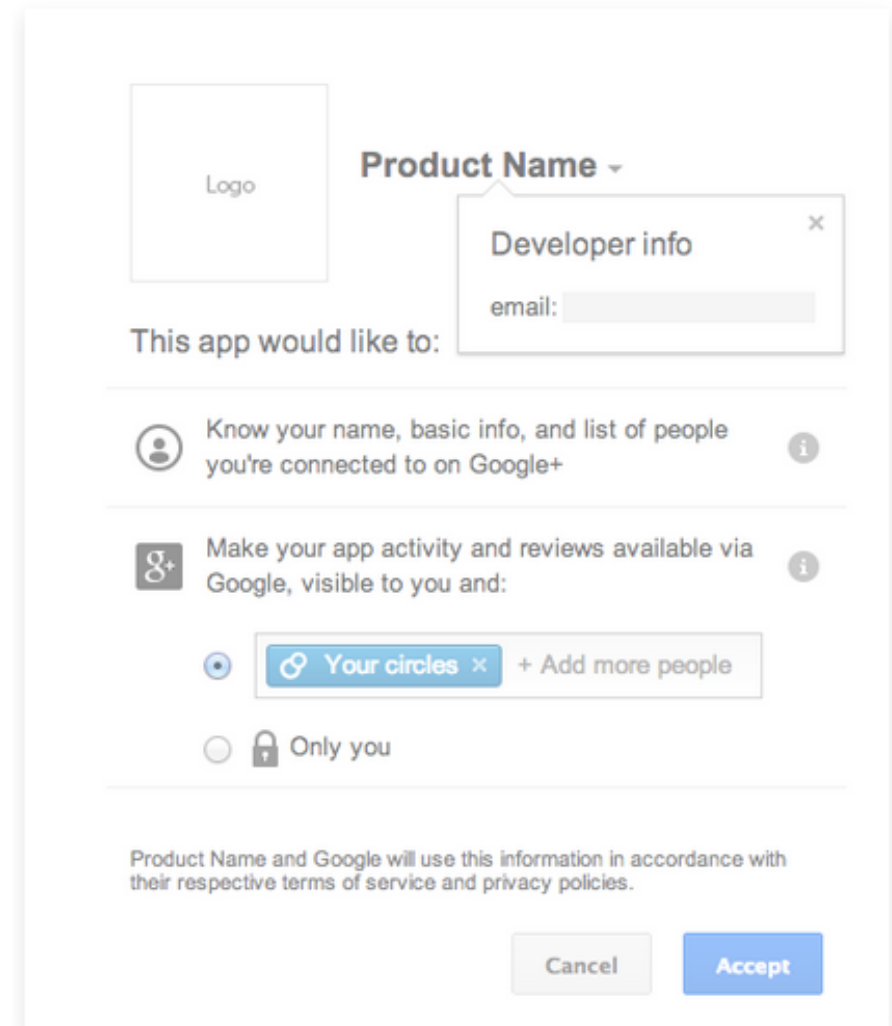


Cool



# Authentication: Consent Screen (?)

- OAuth 2
- Standard and powerful
- Useful for 3rd party apps
- ... but not required for our Android application





# Authentication: ClientIDs

A **Web ClientID** establishes that **one developer is responsible both the backend and the client**, thus removing the need for the OAuth2 consent screen.

=> required in the backend configuration and in the **Android app**

A **Android ClientID** for the mobile app strongly identifies it. Computed from package name, and SHA1 fingerprint (debug or release keystores).

=> required in the backend configuration only

## Client ID for web application

Client ID	818378024343-44e8b7e321eae43111
Email address	818378024343-44e8b7e321eae43111
Client secret	90882e4f2a2e09877a0c2a2f8a2e0987
Redirect URIs	https://mbs2alexis.appspot.com
Javascript Origins	https://mbs2alexis.appspot.com

Edit settings

Download JSON

Delete

## Client ID for Android application

Client ID	818378024343-44e8b7e321eae43111
Redirect URIs	urn:ietf:wg:oauth:2.0:oob http://localhost
Package name	org.alexismp.cloud.backend
Certificate fingerprint (SHA1)	43:24:78:28:33:33:47:84:47:33:47:1
Deep linking	Disabled

Edit settings

Download JSON

Delete

# Authentication: Backend Setup

## Authentication / Authorization

- Locked Down (Access disabled)  
All requests will be rejected.
- Open (for development use only)  
All unauthenticated requests will be allowed. The backend will not be taking advantage of the integrated authentication to know the identity of the callers. For Android sample client app you can use either emulator or a physical device.
- Secured by Client IDs (Recommended)  
Only authenticated calls using the registered Client IDs will be allowed. For Android sample client app you need to use a physical device.

## Android

**Android Client ID** [Learn how to obtain an Android Client ID](#)

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.apps.googleusercontent.com

**Web Client ID** [Learn how to obtain a Web Client ID](#)

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.apps.googleusercontent.com

## iOS

**iOS Client ID** [Learn how to obtain an iOS Client ID](#)

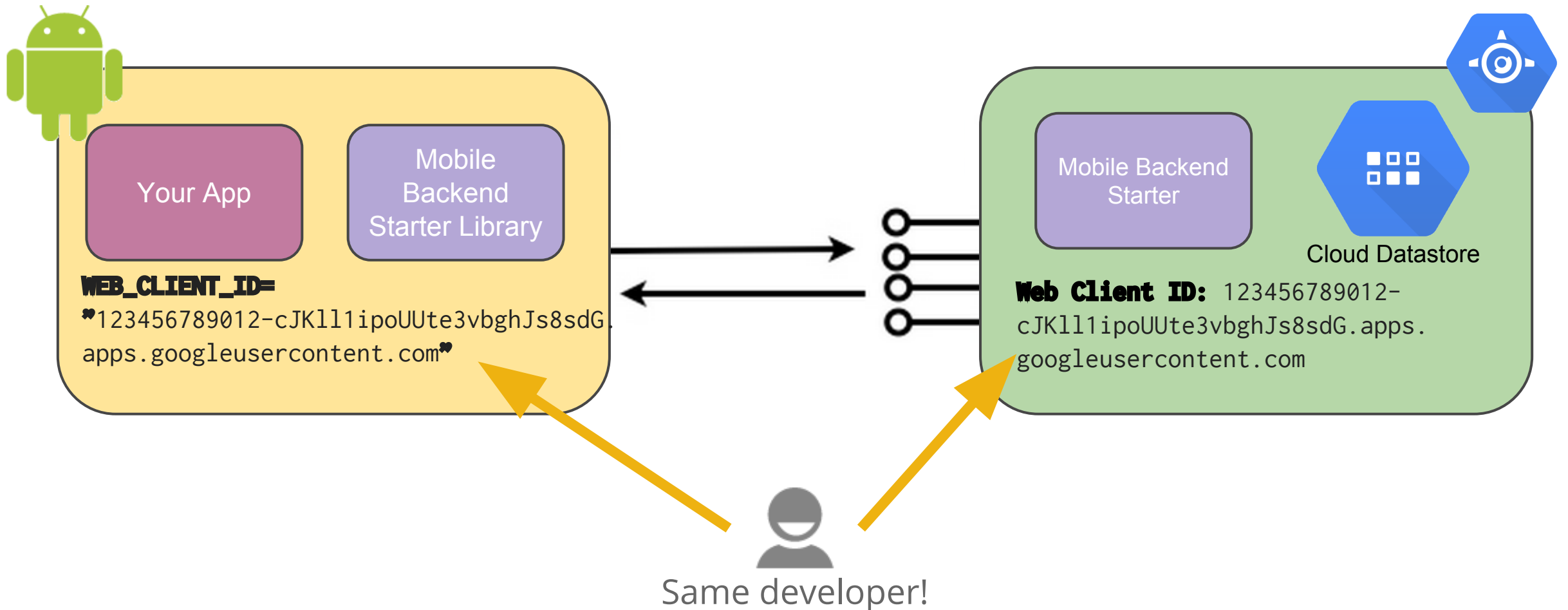
# Authentication: Client Setup

Android - Consts.java

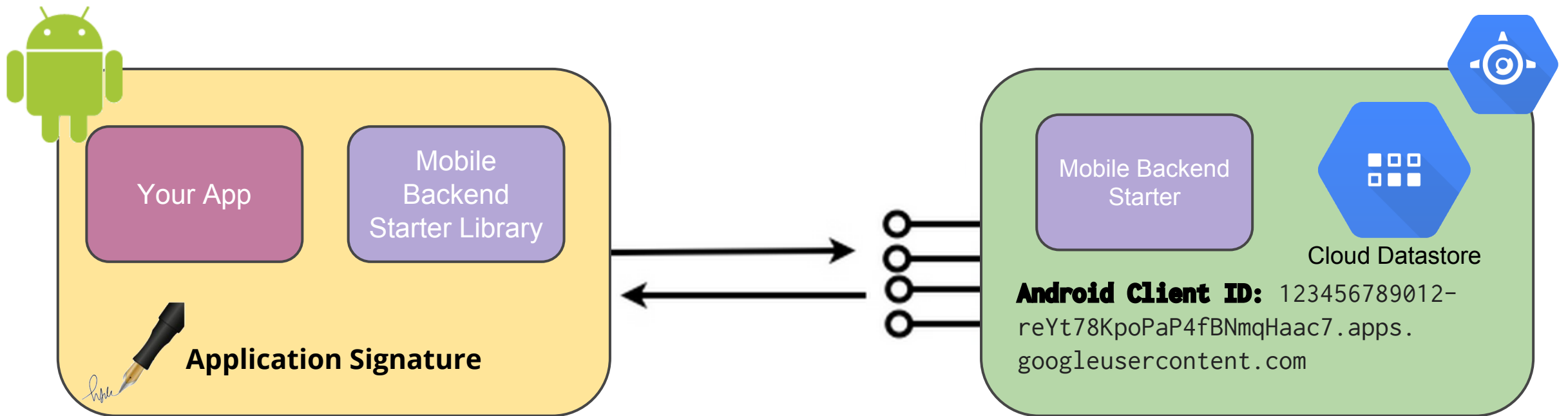
```
public static final String PROJECT_ID = "jfokus-mbs";
/**
 * Set Project Number of your Google APIs Console Project.
 */
public static final String PROJECT_NUMBER = "123456789012";
/**
 * Set your Web Client ID for authentication at backend.
 */
public static final String WEB_CLIENT_ID =
    "123456789012-xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.apps.googleusercontent.com";
/**
 * Set default user authentication enabled or disabled.
 */
public static final boolean IS_AUTH_ENABLED = true;
```



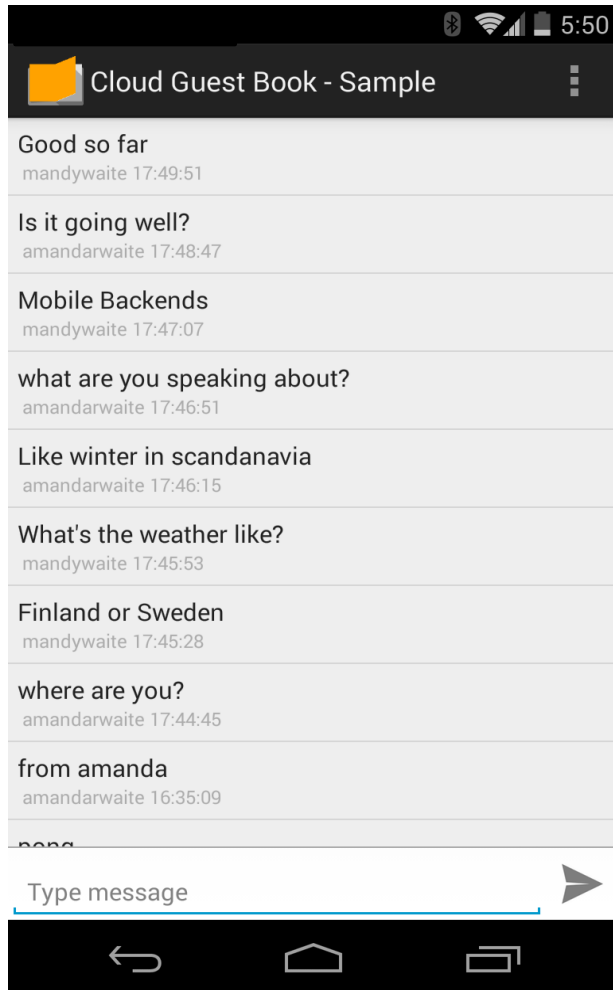
# Authentication: Same Developer



# Authentication: Android ClientID



# Cloud Guest Book App



## 1. Shared Guest Book

- Storing messages

## 2. Notifications

- Update UI with no polling
- Push messaging with GCM

## 3. Authentication

- Propagate the Android logged-in user
- Hide the Oauth2 consent screen



# Coding the Mobile Backend Starter

# Pub/Sub - Subscribe

```
CloudCallbackHandler<List<CloudEntity>> cloudCallBackHandler =  
    new CloudCallbackHandler<List<CloudEntity>>() {  
        @Override  
        public void onComplete(List<CloudEntity> messages) {  
            /* message processing logic */  
        }  
    };
```

Define Handler



```
CloudBackendMessaging cloudBackendMessaging = new CloudBackendMessaging(activity);  
cloudBackendMessaging.subscribeToCloudMessage("JFokus", cloudCallBackHandler);
```

Subscribe





# Pub/Sub - Publish

Publish Message

```
CloudBackendMessaging cloudBackendMessaging = new CloudBackendMessaging(activity);  
  
CloudEntity ce = cloudBackendMessaging.createCloudMessage("JFokus");  
ce.put("chatMessage", "<chat message about how great JFokus is>");  
cloudBackendMessaging.sendCloudMessage(ce);
```



# Continuous Queries

Register Query

```
CloudBackendAsync cloudBackendAsync = new CloudBackendAsync(activity);
```

```
CloudQuery cq = new CloudQuery("MyEvents");
```

```
cq.setFilter(F.eq("event", "JFokus"));
```

```
cq.setScope(Scope.FUTURE_AND_PAST);
```

```
List<CloudEntity> results = cloudBackendAsync.list(cq, cloudCallbackHandler);
```



# Storing and Retrieving Data

Store Data (Insert)

```
// Cloud Datastore example for storing data
CloudBackendAsync cloudBackendAsync = new CloudBackendAsync(activity);

CloudEntity newPost = CloudEntity("GuestBook");
newPost.put("message", etMessage.getText().toString());
cloudBackendAsync.insert(newPost, cloudCallbackHandler);

// Cloud Storage: get a secure URL for uploading a private file
URL url = blobEndpoint.getUploadUrl("receipts", getReceiptFileName(), "PRIVATE")
    .execute().getShortLivedUrl();

// Cloud Storage: get a secure URL for downloading a file
URL url = blobEndpoint.getDownloadUrl("receipts",getReceiptFileName())
    .execute().getShortLivedUrl();
```



- Cloud Datastore or Cloud Storage
- Abstraction classes in Android MBS Client Framework

# CRUD operations

## Crud Operations

```
CloudBackendAsync cloudBackendAsync = new CloudBackendAsync(activity);  
cloudBackendAsync.insert(ce, cloudCallbackHandler);  
cloudBackendAsync.get(ce, cloudCallbackHandler);  
cloudBackendAsync.update(ce, cloudCallbackHandler);  
cloudBackendAsync.delete(ce, cloudCallbackHandler);
```





# Customising the Backend

# Backend Code + API Reference

Open Source - Github

GoogleCloudPlatform/**solutions-mobile-backend-starter-java**

<http://goo.gl/FgKRI>

API Reference

<https://cloud.google.com/resources/articles/mobile-backend-starter-api-reference>

<http://goo.gl/VUsX42>



Wrap Up

Q & A





---

# Thanks!

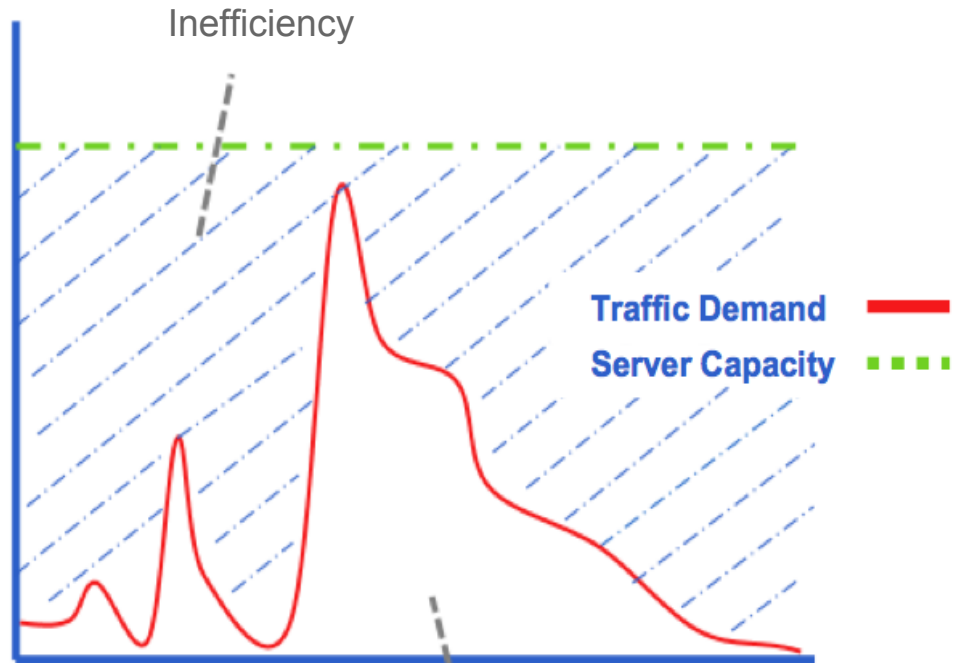
Mandy Waite  
Cloud Platform Developer Advocate  
<http://about.me/mandywaite>

# Messaging

- Messages are Datastore entities
  - Represented by the CloudEntity class on the client side
- Continuous queries
  - Run against any Datastore Kind
  - Can return past messages (with Scope.FUTURE\_AND\_PAST)
- Pub/Sub messaging
  - Uses Continuous queries
  - Uses a specific Datastore Kind for storing and retrieving messages
  - Can retrieve past messages of that Kind
  - Subscription is by TopicId not by Message content

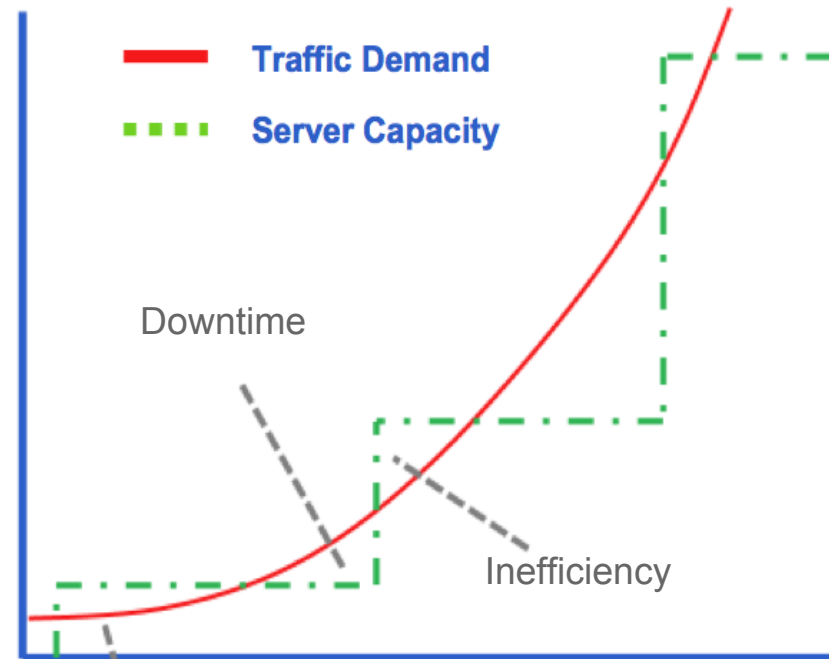
# App Engine AutoScale

## Volatile Demand Fluctuation



With App Engine  
only pay for what you use

## Steady Demand Growth



With App Engine  
scale with efficiency and reliability