Lecture #10
Firebase Services

Mobile Applications 2019-2020

James Tamplin

Andrew Lee

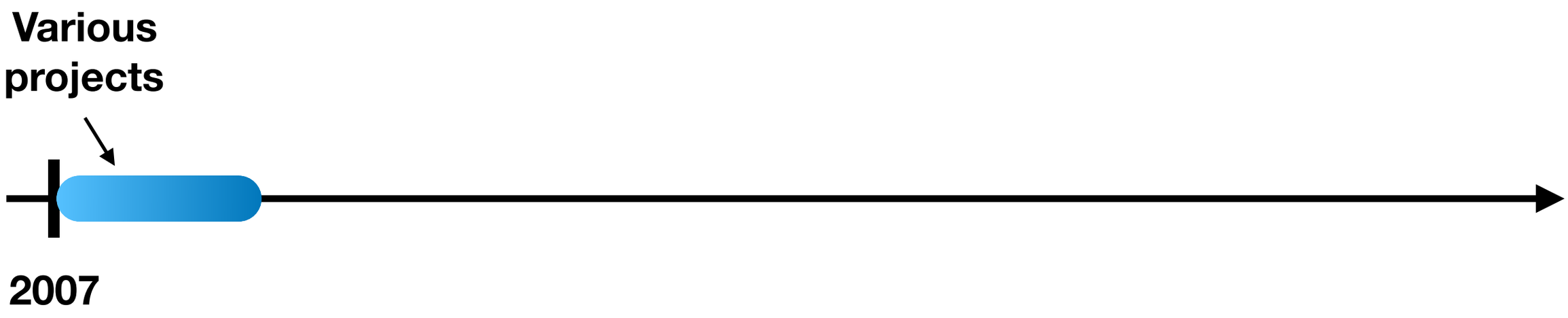
James Tamplin

Andrew Lee

2007

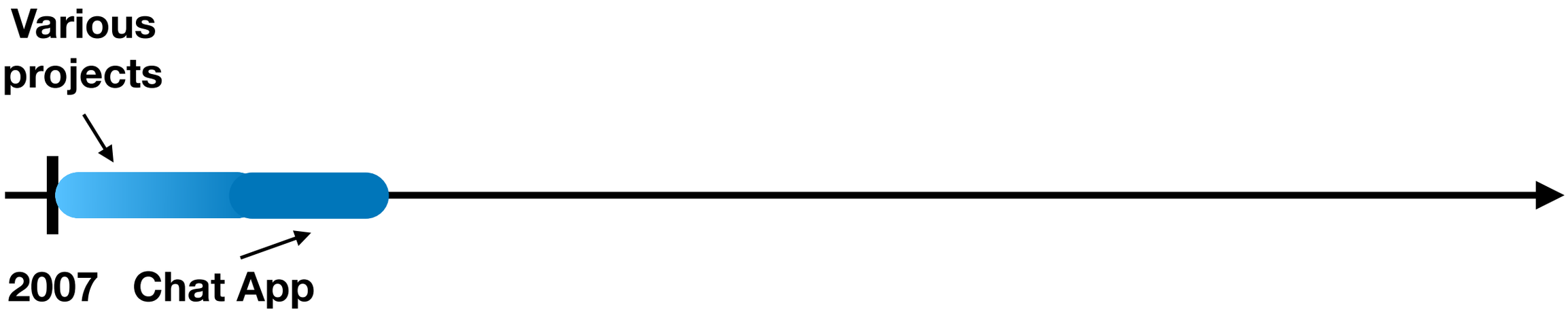
James Tamplin

Andrew Lee



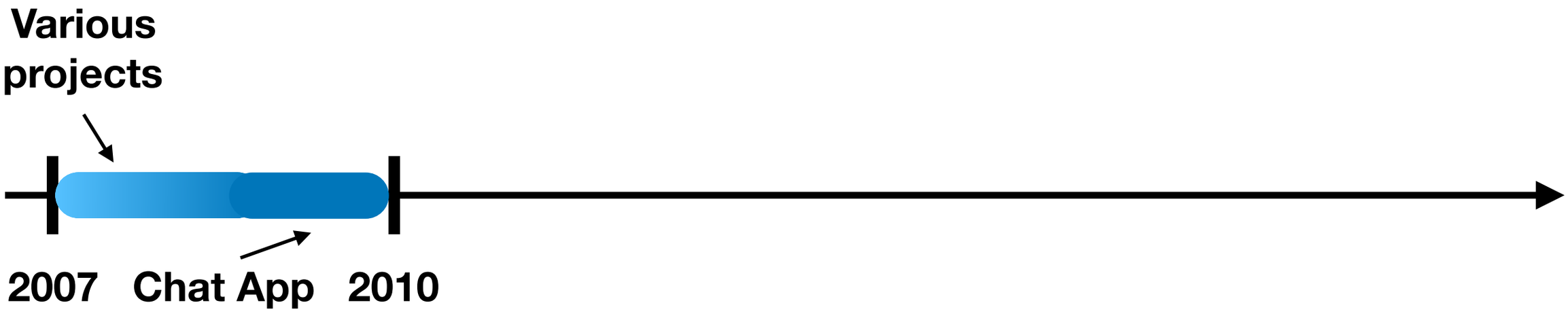
James Tamplin

Andrew Lee

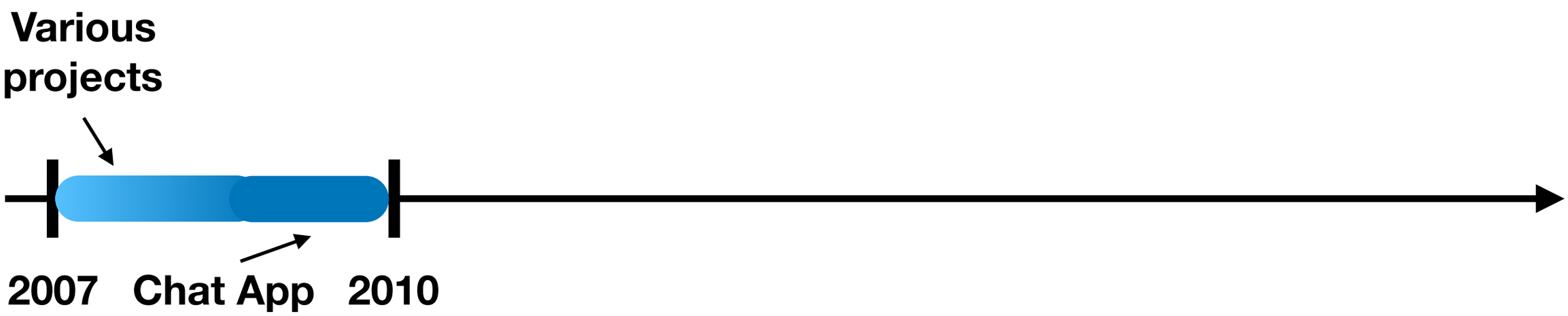


James Tamplin

Andrew Lee

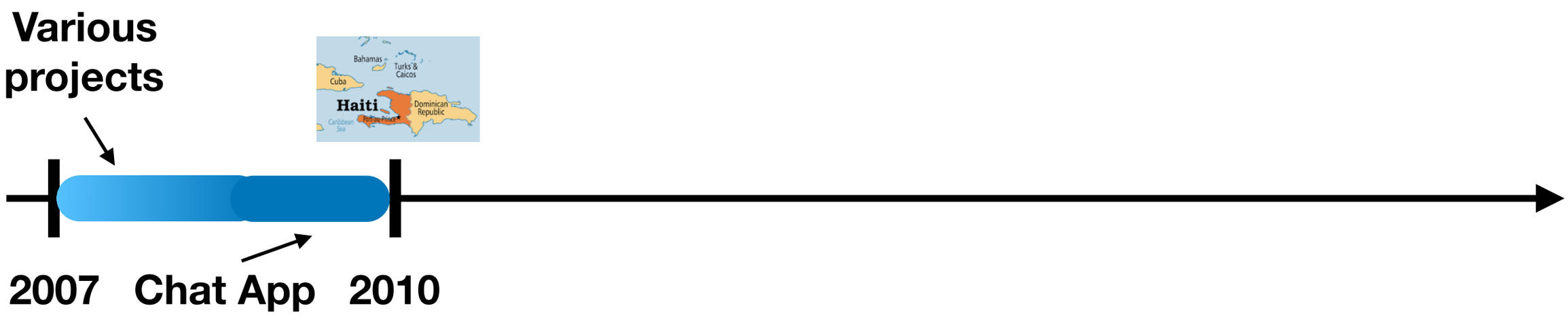


James Tamplin **Andrew Lee**



James Tamplin

Andrew Lee



James Tamplin

Andrew Lee

Various projects

2007 Chat App 2010

James Tamplin

Andrew Lee

Various projects

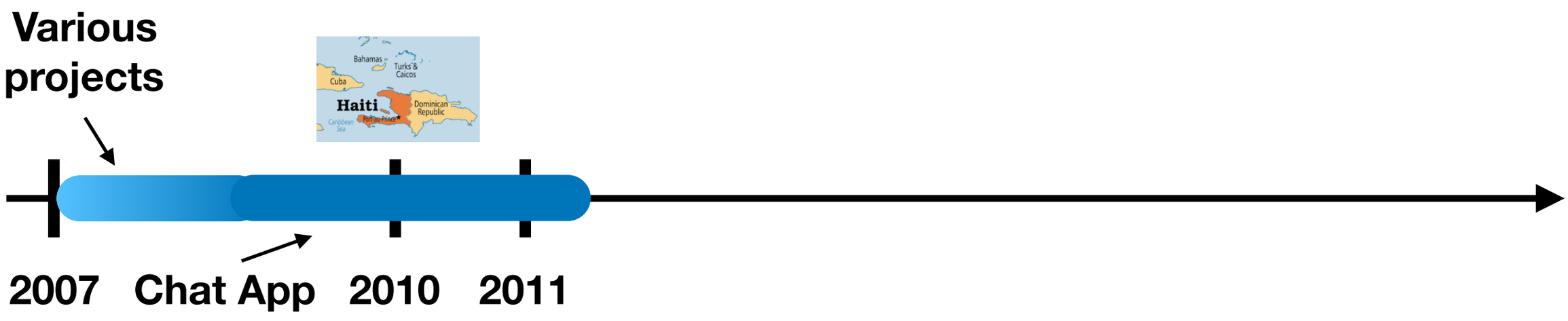
2007 Chat App 2010

https://startupandrew.com/posts/my-experiences-in-the-haitian-earthquake/

envolve

James Tamplin

Andrew Lee

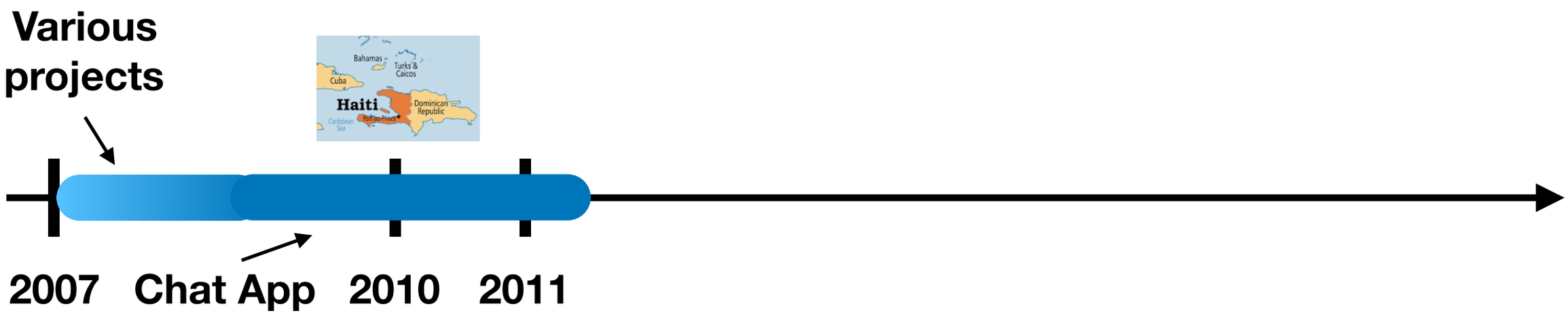


envolve

facebook

James Tamplin

Andrew Lee

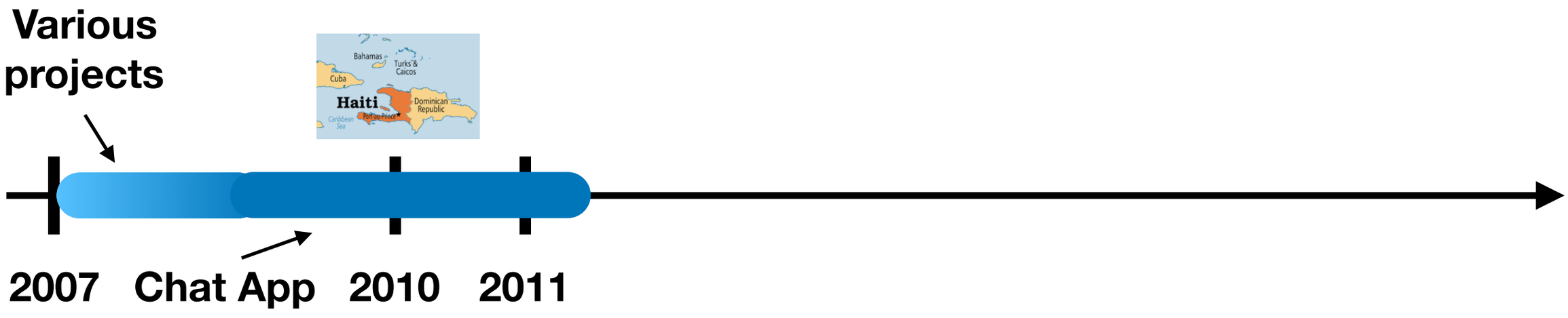


envolve

OTHER THINGS

James Tamplin

Andrew Lee

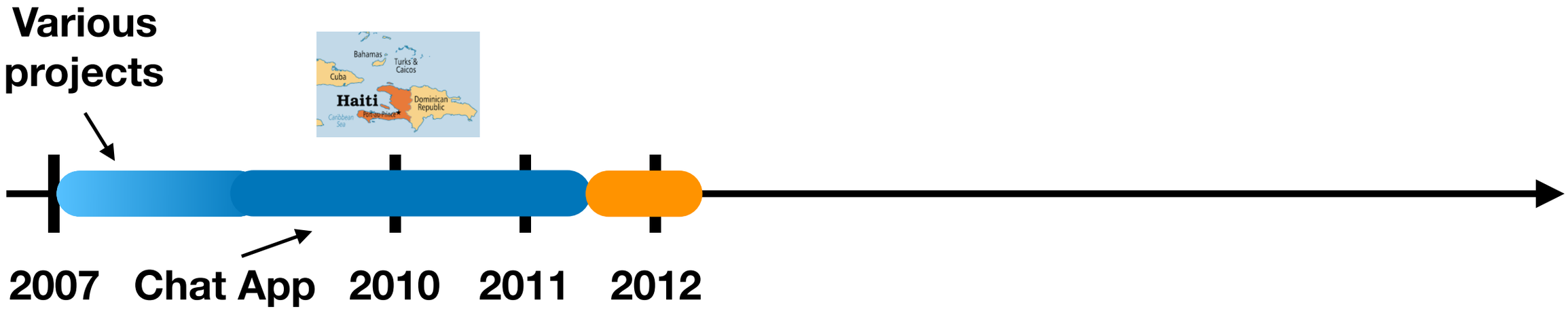


envolve

OTHER THINGS

James Tamplin

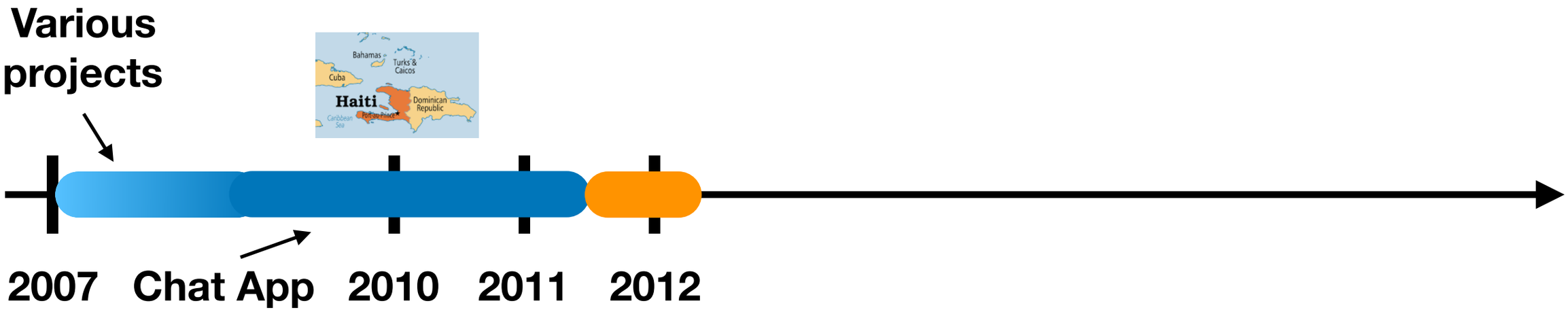
Andrew Lee



Firebase

James Tamplin

Andrew Lee

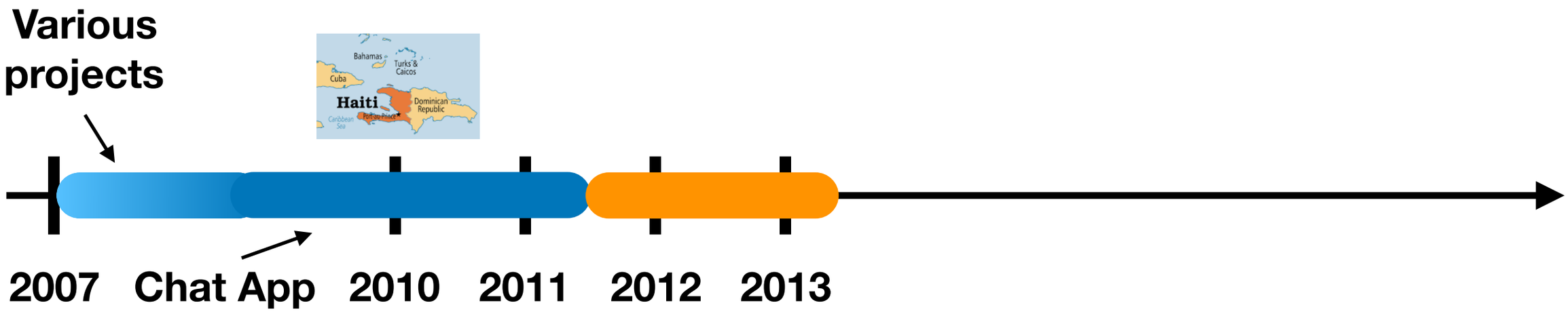


Firebase

Built with Firebase

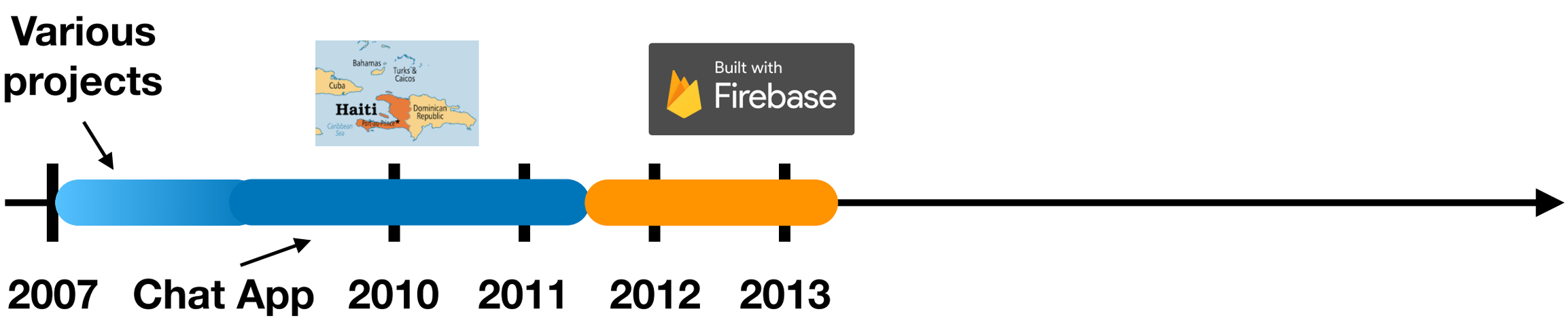
James Tamplin

Andrew Lee



Firebase

James Tamplin **Andrew Lee**

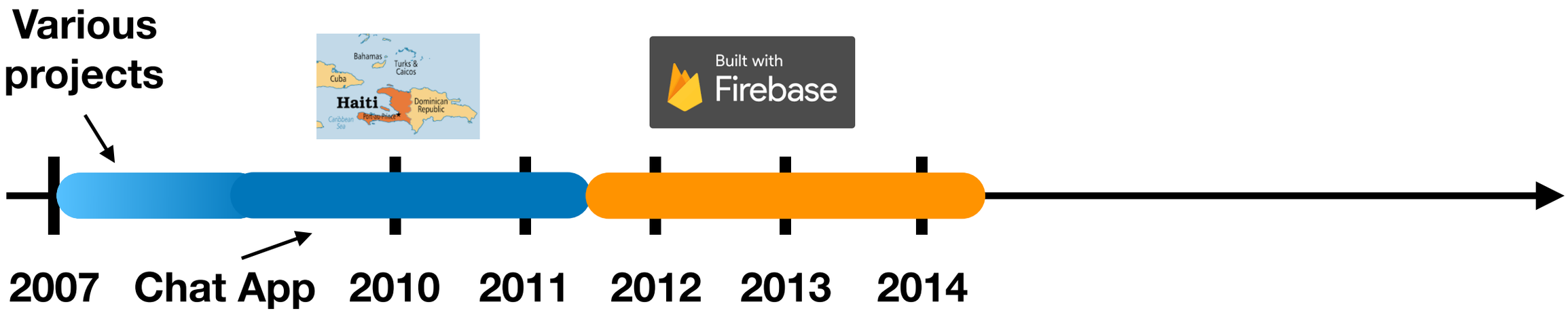


Firebase

Google

James Tamplin

Andrew Lee



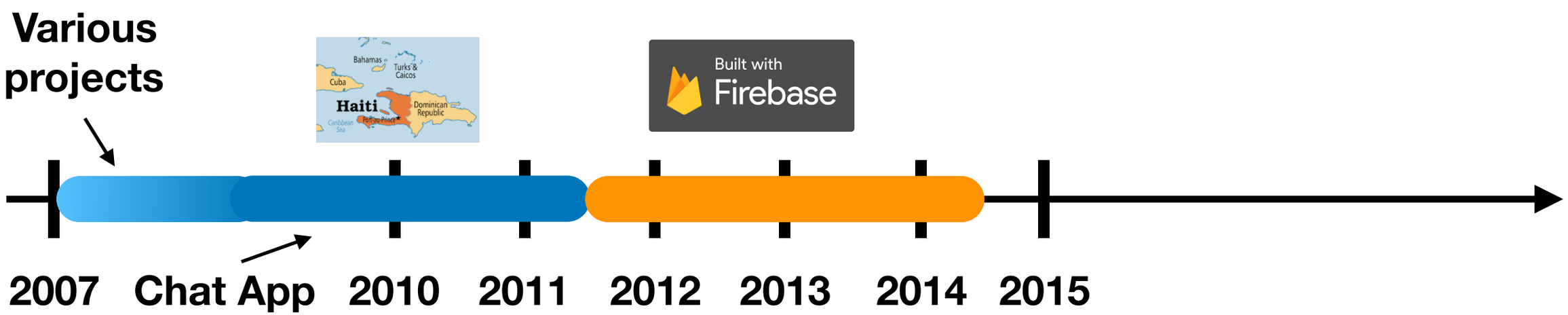
Firebase

James Tamplin

Andrew Lee

Google

DIVSHOT



Firebase

James Tamplin

Andrew Lee

Google

Firebase
Realtime Database

Firebase
Hosting

Various projects

Built with Firebase

2007 Chat App 2010 2011 2012 2013 2014 2015

Firebase

James Tamplin

Andrew Lee

Firebase Cloud Messaging

Google

Cloud Storage for Firebase

Firebase Realtime Database

Firebase Hosting

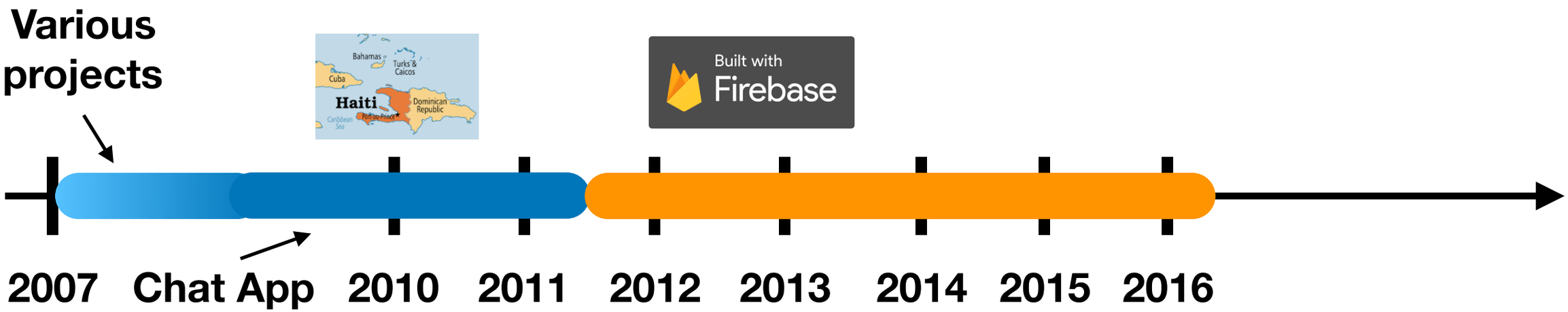
Built with Firebase

Various projects

2007 Chat App 2010 2011 2012 2013 2014 2015 2016

Firebase

Google



Firebase

Firebase Crash Reporting

Firebase Authentication

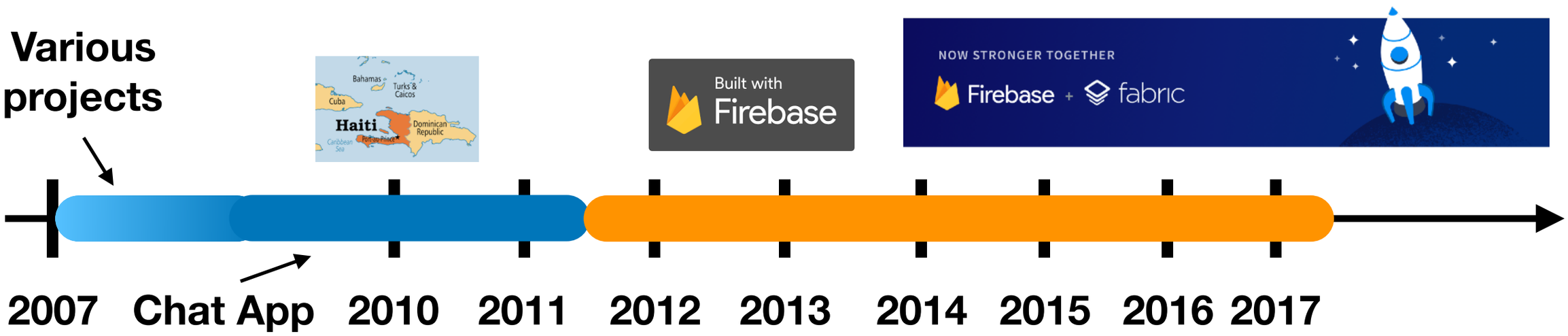
Firebase Cloud Messaging

Google

Cloud Storage for Firebase

Firebase Realtime Database

Firebase Hosting



Firebase

Analytics

Firebase
Authentication

Firebase
Cloud Messaging

Google

Cloud Storage
for Firebase

Firebase
Realtime Database

Firebase
Hosting

Various
projects

Built with
Firebase

2007 Chat App 2010 2011 2012 2013 2014 2015 2016 2017

Firebase

Analytics

Cloud Function for Firebase

Firebase Authentication

Firebase Cloud Messaging

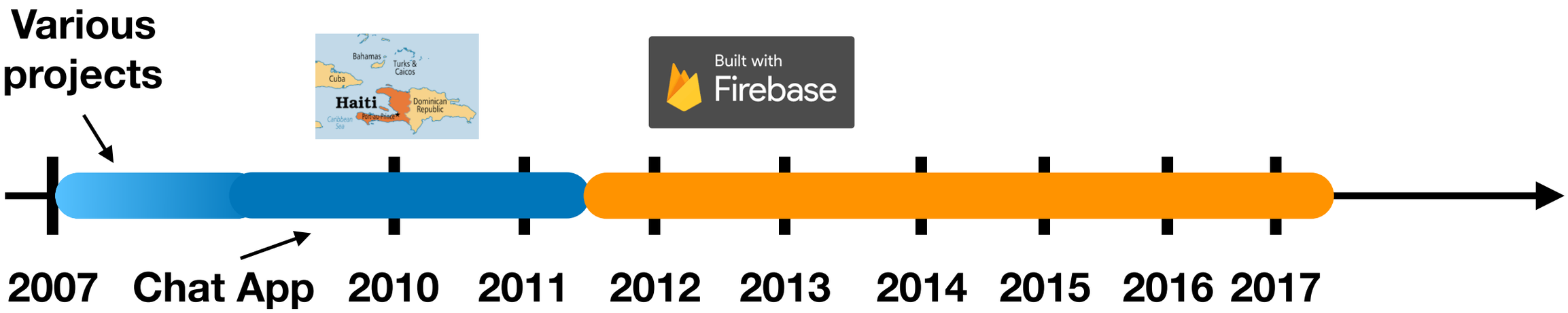
Google

Cloud Storage for Firebase

Firebase Realtime Database

Cloud Firestore

Firebase Hosting



Firebase

Analytics

Cloud Function for Firebase

Firebase Authentication

Firebase Cloud Messaging

Google

Cloud Storage for Firebase

Firebase Realtime Database

Cloud Firestore

Firebase Hosting

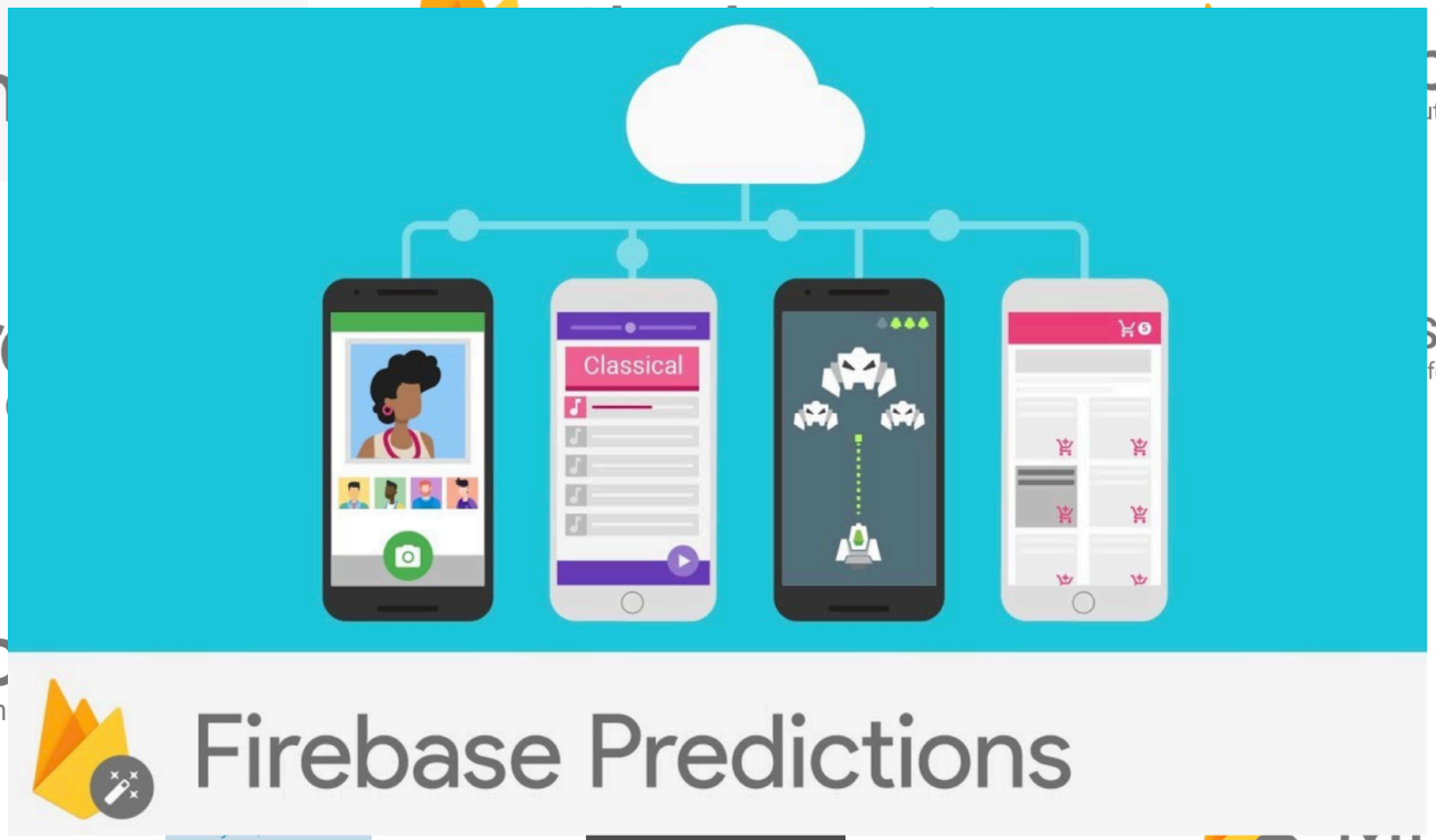
ML Kit for Firebase

Built with Firebase

Various projects

2007 Chat App 2010 2011 2012 2013 2014 2015 2016 2017 2018

Firebase



Firebase Predictions

Various projects

Built with Firebase

ML Kit for Firebase

2007 Chat App 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019

Firebase App Distribution

2007 Chat App 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019

Firebase

Analytics

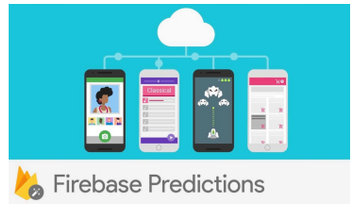
Cloud Function for Firebase

Firebase Authentication

Firebase Cloud Messaging

Google

Cloud Storage for Firebase



Firebase Realtime Database

Cloud Firestore

Firebase Hosting

ML Kit for Firebase

Various projects

Built with Firebase

2007 Chat App 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019

https://firebase.google.com/support/releases


Firebase
android
iOS

Firebase

Flutter

www

ANDROID

iOS

C++

unity

Firebase

Flutter

www

android

iOS

golang

NODEjs

Java

C++

C#

unity

python

Firebase

Flutter

Over 1.5 million apps actively using Firebase every month!

android

iOS

python


Firebase
Over 1.5 million apps
actively using Firebase
every month!
2018
android
iOS

Firebase

Over 2 million apps
actively using Firebase
every month!

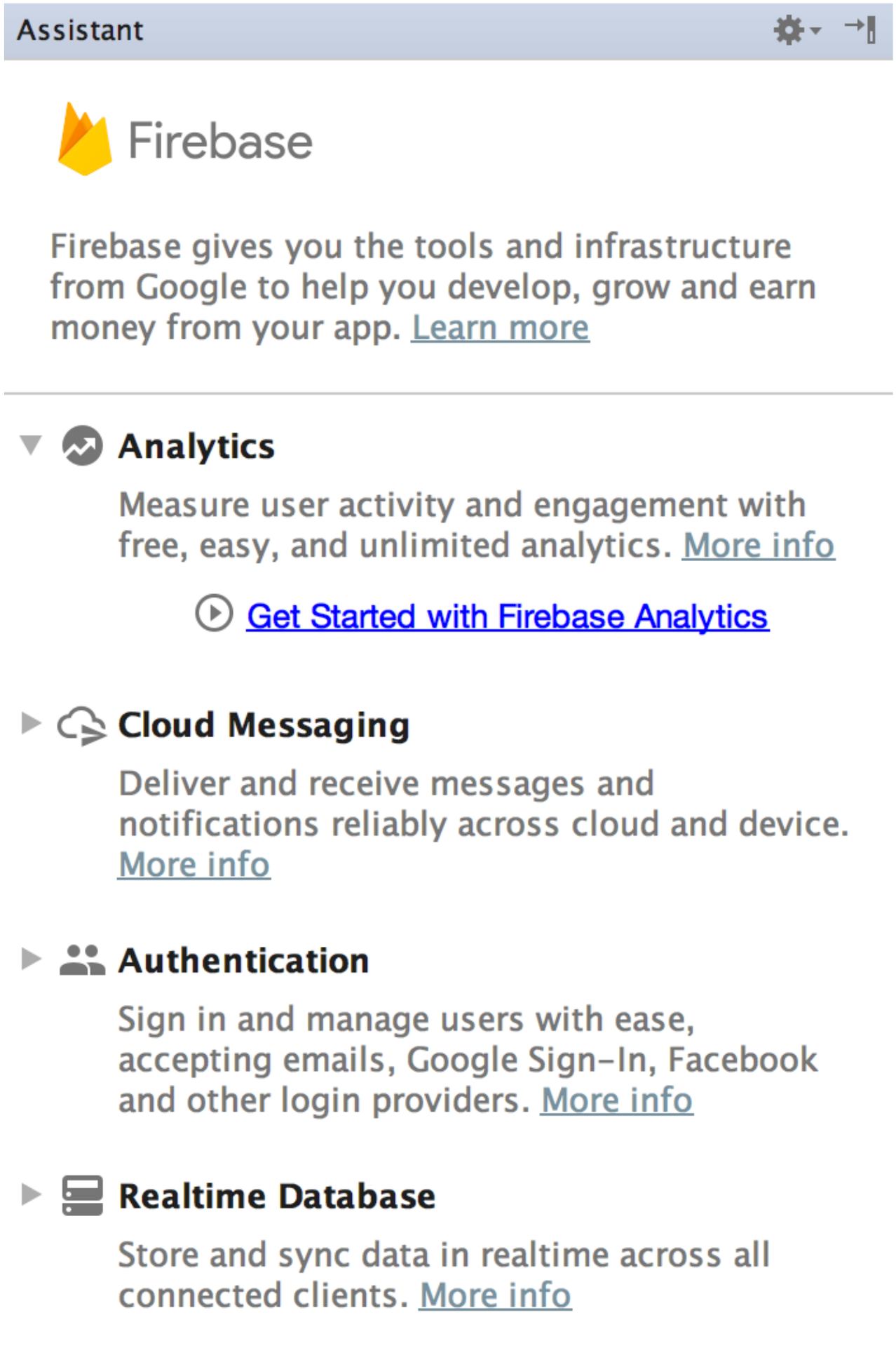
Add Firebase to Your Project

- Using the assistant, in Android Studio:

  1. Tools -> Firebase.

  2. Select the service.

  3. Connect to Firebase.



- Using the assistant, in Android Studio:

  1. Tools -> Firebase.

  2. Select the service.

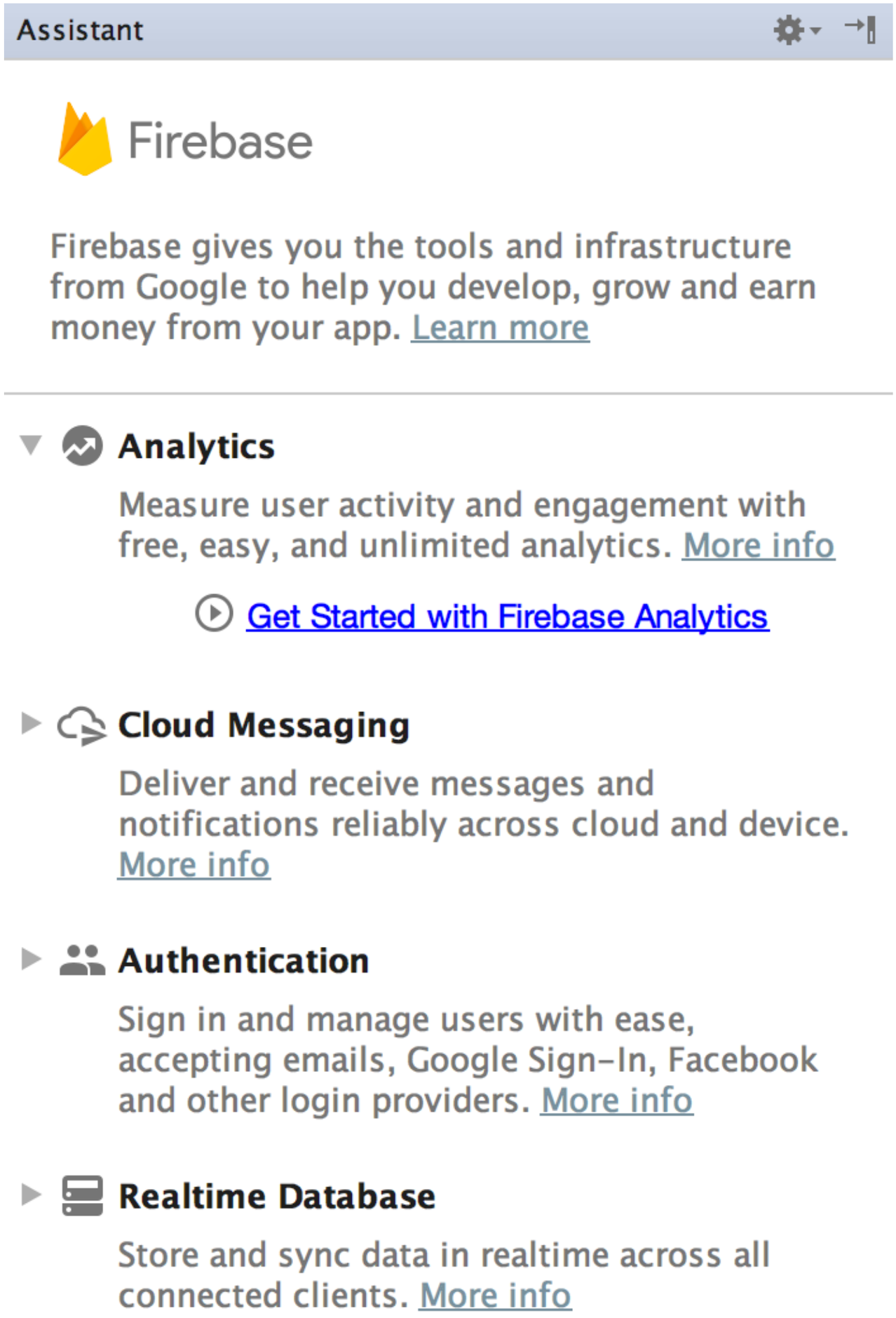
  3. Connect to Firebase.

- Manually:

  - Create a project in console.firebase.google.com

  - Download the config file.

  - Add the SDK



Add the SDK

Root-level build.gradle:

```
buildscript {
  // ...
  dependencies {
    // ...
    classpath ‘com.google.gms:google-services:4.3.3' // google-services plugin
  }
}
```

```
allprojects {
  // ...
  repositories {
    // ...
    google() // Google's Maven repository
  }
}
```

Add the SDK

Module-level build.gradle:

```
apply plugin: 'com.android.application'

android {
  // ...
}

dependencies {
  // ...
  implementation ‘com.google.firebase:firebase-auth:19.2.0'

  // Getting a "Could not find" error? Make sure you have
  // added the Google maven respository to your root build.gradle
}

// ADD THIS AT THE BOTTOM
apply plugin: 'com.google.gms.google-services'
```

Available Libraries

ads

com.google.android.gms.ads	Contains classes for Google Mobile Ads.
com.google.android.gms.ads.appopen	
com.google.android.gms.ads.doubleclick	Contains classes for DoubleClick for Publishers.
com.google.android.gms.ads.formats	Contains classes for native ads functionality within Google Mobile Ads.
com.google.android.gms.ads.initialization	Contains classes related to SDK initialization.
com.google.android.gms.ads.instream	Contains classes related to the instream ad format.
com.google.android.gms.ads.mediation	Contains classes for Google Mobile Ads mediation adapters.
com.google.android.gms.ads.mediation.admob	Contains classes for the Google AdMob mediation adapter.
com.google.android.gms.ads.mediation.customevent	Contains classes for Google Mobile Ads mediation custom events.
com.google.android.gms.ads.mediation.rtb	Contains classes for Google Mobile Ads RTB mediation adapters.
com.google.android.gms.ads.reward	Contains classes for Rewarded Video Ads.
com.google.android.gms.ads.reward.mediation	Contains classes for Rewarded Video Ads mediation adapters.
com.google.android.gms.ads.rewarded	Contains classes for Rewarded Ads.
com.google.android.gms.ads.search	Contains classes for Search Ads for Apps.

Available Libraries

appindexing

com.google.android.gms.appindexing	
com.google.firebase.appindexing	The Firebase App Indexing API lets apps index personal content and log user actions with Google.
com.google.firebase.appindexing.builders	This package contains builders and convenience methods to help construct common types of indexable objects and actions.

appinvite

com.google.android.gms.appinvite	
com.google.firebase.appinvite	

measurement

com.google.android.gms.measurement	

measurement.impl

com.google.android.gms.measurement	Contains classes that configure Firebase Analytics core services.
com.google.firebase.analytics	Contains public API classes for Firebase Analytics.

Available Libraries

firebase

- com.google.firebase
- com.google.firebase.auth
- com.google.firebase.provider

firebase.auth

- com.google.firebase.auth

crashlytics

- crashlytics
- crashlytics.core
- fabric

firebase.database

- com.google.firebase.database

Available Libraries

firebase.dynamiclinks

com.google.firebase.dynamiclinks

firebase.firestore

com.google.firebase

com.google.firebase.firestore

firebase.functions

com.google.firebase.functions

firebase.iid

com.google.firebase.iid

firebase.inappmessaging

com.google.firebase.inappmessaging

Available Libraries

firebase.inappmessaging.display

com.google.firebase.inappmessaging.display	

firebase.messaging

com.google.firebase.messaging	Contains public API classes for Firebase Cloud Messaging.

firebase.ml.common

com.google.firebase.ml.common	
com.google.firebase.ml.common.modeldownload	

firebase.ml.interpreter

com.google.firebase.ml.custom	

firebase.ml.naturallanguage

com.google.firebase.ml.naturallanguage	
com.google.firebase.ml.naturallanguage.languageid	
com.google.firebase.ml.naturallanguage.smartreply	

Available Libraries

firebase.perf

com.google.firebase.perf
com.google.firebase.perf.metrics

firebase.remoteconfig

com.google.firebase.remoteconfig

firebase.storage

com.google.firebase.storage

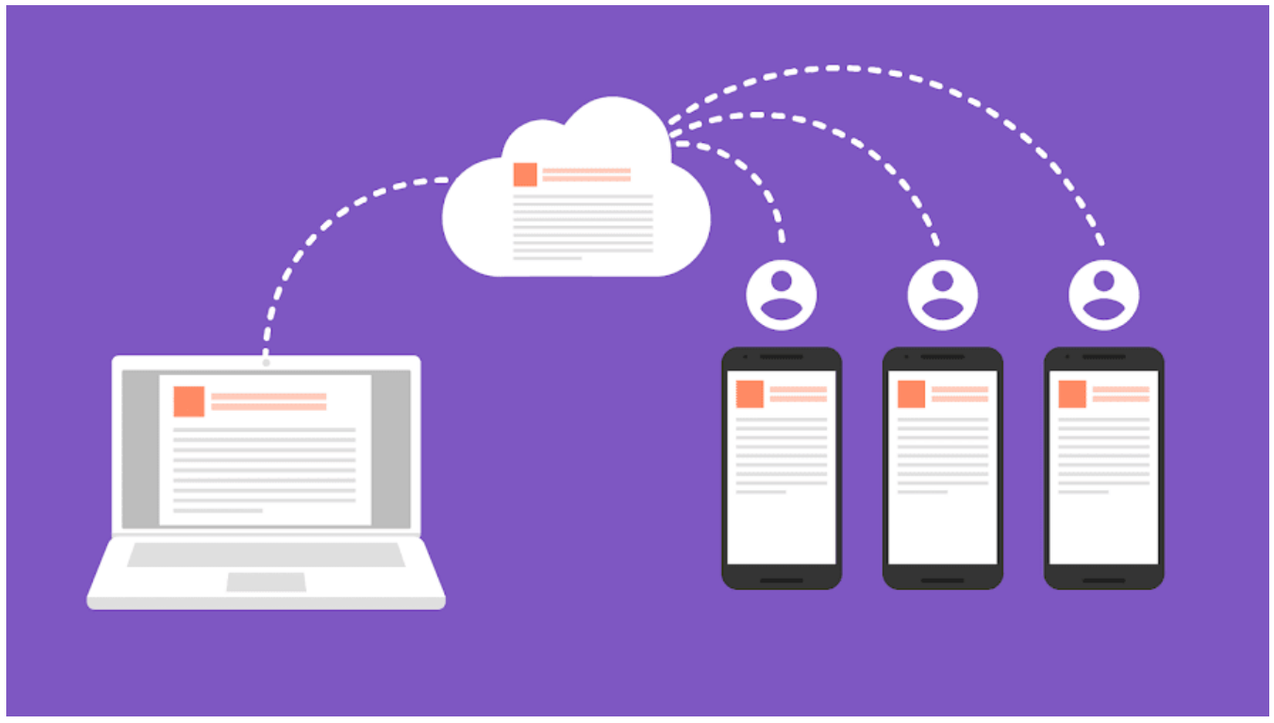
Inter-operational packages

com.google.firebase.auth.internal

firebase.google.com/docs/reference/android/packages

Realtime Database

- Realtime.
- Offline.
- Collaborate across devices with ease.
- Scale across multiple regions.



Installation & Setup

Add the Realtime Database to your app

```
implementation 'com.google.firebase:firebase-database-ktx:19.2.0'
```

Installation & Setup

Add the Realtime Database to your app

```
implementation 'com.google.firebase:firebase-database-ktx:19.2.0'
```

Secure Your Data

```
{
  "messages": {
    "message0": {
      "content": "Hello",
      "timestamp": 1405704370369
    },
    "message1": {
      "content": "Goodbye",
      "timestamp": 1405704395231
    },
    ...
  }
}
```

Installation & Setup

Add the Realtime Database to your app

```
implementation 'com.google.firebase:firebase-database-ktx:19.2.0'
```

Secure Your Data

```
{
  "messages": {
    "message0": {
      "content": "Hello",
      "timestamp": 1405704370369
    },
    "message1": {
      "content": "Goodbye",
      "timestamp": 1405704395231
    },
    ...
  }
}
```

```
{
  "rules": {
    "messages": {
      "$message": {
        // only messages from the last ten minutes can be read
        ".read": "data.child('timestamp').val() > (now - 600000)",
        // new messages must have a string content and a number timestamp
        ".validate": "newData.hasChildren(['content', 'timestamp'])
            && newData.child('content').isString()
            && newData.child('timestamp').isNumber()"
      }
    }
  }
}
```

Installation & Setup

Add the Realtime Database to your app

```
implementation 'com.google.firebase:firebase-database-ktx:19.2.0'
```

Secure Your Data

```
{
  "messages": {
    "message0": {
      "content": "Hello",
      "timestamp": 1405704370369
    },
    "message1": {
      "content": "Goodbye",
      "timestamp": 1405704395231
    },
    ...
  }
}
```

```
{
  "rules": {
    "messages": {
      "$message": {
        // only messages from the last ten minutes can be read
        ".read": "data.child('timestamp').val() > (now - 600000)",
        // new messages must have a string content and a number timestamp
        ".validate": "newData.hasChildren(['content', 'timestamp'])
            && newData.child('content').isString()
            && newData.child('timestamp').isNumber()"
      }
    }
  }
}
```

Installation & Setup

Add the Realtime Database to your app

```
implementation 'com.google.firebase:firebase-database-ktx:19.2.0'
```

Secure Your Data

```
{
  "messages": {
    "message0": {
      "content": "Hello",
      "timestamp": 1405704370369
    },
    "message1": {
      "content": "Goodbye",
      "timestamp": 1405704395231
    },
    ...
  }
}
```

```
{
  "rules": {
    "messages": {
      "$message": {
        // only messages from the last ten minutes can be read
        ".read": "data.child('timestamp').val() > (now - 600000)",
        // new messages must have a string content and a number timestamp
        ".validate": "newData.hasChildren(['content', 'timestamp'])
            && newData.child('content').isString()
            && newData.child('timestamp').isNumber()"
      }
    }
  }
}
```

Installation & Setup

Add the Realtime Database to your app

```
implementation 'com.google.firebase:firebase-database-ktx:19.2.0'
```

Secure Your Data

```
{
  "messages": {
    "message0": {
      "content": "Hello",
      "timestamp": 1405704370369
    },
    "message1": {
      "content": "Goodbye",
      "timestamp": 1405704395231
    },
    ...
  }
}
```

```
{
  "rules": {
    "messages": {
      "$message": {
        // only messages from the last ten minutes can be read
        ".read": "data.child('timestamp').val() > (now - 600000)",
        // new messages must have a string content and a number timestamp
        ".validate": "newData.hasChildren(['content', 'timestamp'])
            && newData.child('content').isString()
            && newData.child('timestamp').isNumber()"
      }
    }
  }
}
```

Installation & Setup

Add the Realtime Database to your app

```
implementation 'com.google.firebase:firebase-database-ktx:19.2.0'
```

Secure Your Data

```
{
  "messages": {
    "message0": {
      "content": "Hello",
      "timestamp": 1405704370369
    },
    "message1": {
      "content": "Goodbye",
      "timestamp": 1405704395231
    },
    ...
  }
}
```

```
{
  "rules": {
    "messages": {
      "$message": {
        // only messages from the last ten minutes can be read
        ".read": "data.child('timestamp').val() > (now - 600000)",
        // new messages must have a string content and a number timestamp
        ".validate": "newData.hasChildren(['content', 'timestamp'])
            && newData.child('content').isString()
            && newData.child('timestamp').isNumber()"
      }
    }
  }
}
```

https://firebase.google.com/docs/database/security/securing-data

Data Access

Write to your database

```
// Write a message to the database
val database = FirebaseDatabase.getInstance()
val myRef = database.getReference("message")

myRef.setValue("Hello, World!")
```

Data Access

Write to your database

```
// Write a message to the database
val database = FirebaseDatabase.getInstance()
val myRef = database.getReference("message")

myRef.setValue("Hello, World!")
```

Read from your database

```
// Read from the database
myRef.addValueEventListener(object : ValueEventListener {
    override fun onDataChange(dataSnapshot: DataSnapshot) {
        // This method is called once with the initial value and again
        // whenever data at this location is updated.
        val value = dataSnapshot.getValue(String::class.java)
        Log.d(TAG, "Value is: $value")
    }

    override fun onCancelled(error: DatabaseError) {
        // Failed to read value
        Log.w(TAG, "Failed to read value.", error.toException())
    }
})
```

Data Access

Write to your database

```
// Write a message to the database
val database = FirebaseDatabase.getInstance()
val myRef = database.getReference("message")

myRef.setValue("Hello, World!")
```

Read from your database

```
// Read from the database
myRef.addValueEventListener(object : ValueEventListener {
    override fun onDataChange(dataSnapshot: DataSnapshot) {
        // This method is called once with the initial value and again
        // whenever data at this location is updated.
        val value = dataSnapshot.getValue(String::class.java)
        Log.d(TAG, "Value is: $value")
    }

    override fun onCancelled(error: DatabaseError) {
        // Failed to read value
        Log.w(TAG, "Failed to read value.", error.toException())
    }
})
```

https://firebase.google.com/docs/database/android/read-and-write

Update Data

```
private fun writeNewPost(
    userId: String,
    username: String,
    title: String,
    body: String
  ){
  // Create new post at /user-posts/$userid/$postid and at
  // /posts/$postid simultaneously
  val key = database.child("posts").push().key
  if (key == null) {
    Log.w(TAG, "Couldn't get push key for posts")
    return
  }

  val post = Post(userId, username, title, body)
  val postValues = post.toMap()

  val childUpdates = HashMap<String, Any>()
  childUpdates["/posts/$key"] = postValues
  childUpdates["/user-posts/$userId/$key"] = postValues

  database.updateChildren(childUpdates)
}
```

Using Transactions

```
private fun onStarClicked(postRef: DatabaseReference) {
  postRef.runTransaction(object : Transaction.Handler {
    override fun doTransaction(mutableData: MutableData): Transaction.Result {
      val p = mutableData.getValue(Post::class.java)
          ?: return Transaction.success(mutableData)
      if (p.stars.containsKey(uid)) {
        // Unstar the post and remove self from stars
        p.starCount = p.starCount - 1
        p.stars.remove(uid)
      } else {
        // Star the post and add self to stars
        p.starCount = p.starCount + 1
        p.stars[uid] = true
      }
      // Set value and report transaction success
      mutableData.value = p
      return Transaction.success(mutableData)
    }
    override fun onComplete(
      databaseError: DatabaseError?,
      b: Boolean,
      dataSnapshot: DataSnapshot?
    ) {
      // Transaction completed
      Log.d(TAG, "postTransaction:onComplete:" + databaseError!!)
```

```
private fun onStarClicked(postRef: DatabaseReference) {
  postRef.runTransaction(object : Transaction.Handler {
    override fun doTransaction(mutableData: MutableData): Transaction.Result {
      val p = mutableData.getValue(Post::class.java)
          ?: return Transaction.success(mutableData)
      if (p.stars.containsKey(uid)) {
        // Unstar the post and remove self from stars
        p.starCount = p.starCount - 1
        p.stars.remove(uid)
      } else {
        // Star the post and add self to stars
        p.starCount = p.starCount + 1
        p.stars[uid] = true
      }
      // Set value and report transaction success
      mutableData.value = p
      return Transaction.success(mutableData)
    }
    override fun onComplete(
      databaseError: DatabaseError?,
      b: Boolean,
      dataSnapshot: DataSnapshot?
    ) {
      // Transaction completed
      Log.d(TAG, "postTransaction:onComplete:" + databaseError!!)
    }
  })
}
```

Enabling Offline Capabilities

```
FirebaseDatabase.getInstance().setPersistenceEnabled(true);
```

Enabling Offline Capabilities

```
FirebaseDatabase.getInstance().setPersistenceEnabled(true);
```

Keeping Data Fresh

```
DatabaseReference scoresRef = FirebaseDatabase.getInstance()
    .getReference("scores");
scoresRef.keepSynced(true);
```

Enabling Offline Capabilities

```
FirebaseDatabase.getInstance().setPersistenceEnabled(true);
```

Keeping Data Fresh

```
DatabaseReference scoresRef = FirebaseDatabase.getInstance()
    .getReference("scores");
scoresRef.keepSynced(true);

scoresRef.keepSynced(false);
```

DEMO

Enabling Offline Capabilities

```
FirebaseDatabase.getInstance().setPersistenceEnabled(true);
```

Keeping Data Fresh

```
DatabaseReference scoresRef = FirebaseDatabase.getInstance()
    .getReference("scores");
scoresRef.keepSynced(true);

scoresRef.keepSynced(false);
```

Detecting Connection State

```
val connectedRef = FirebaseDatabase.getInstance().getReference(".info/connected")
connectedRef.addValueEventListener(object : ValueEventListener {
  override fun onDataChange(snapshot: DataSnapshot) {
    val connected = snapshot.getValue(Boolean::class.java) ?: false
    if (connected) {
      Log.d(TAG, "connected")
    } else {
      Log.d(TAG, "not connected")
    }
  }

  override fun onCancelled(error: DatabaseError) {
    Log.w(TAG, "Listener was cancelled")
  }
})
```

https://firebase.google.com/docs/database/android/offline-capabilities

Authentication

Key capabilities:

- Email and password based authentication.
- Federated identity provider integration:
  - Google, Facebook, Twitter, Github.
- Phone number authentication.
- Custom auth system integration.
- Anonymous auth.

Authentication

Key capabilities:

- Email and password based authentication.
- Federated identity provider integration:
  - Google, Facebook, Twitter, Github.
- Phone number authentication.
- Custom auth system integration.
- Anonymous auth.

Authentication

Key capabilities:

- Email and password based authentication.
- Federated identity provider integration:
  - Google, Facebook, Twitter, Github, Apple.
- Phone number authentication.
- Custom auth system integration.
- Anonymous auth.

Authenticate Using Google Sign-In

Authenticate Using Google Sign-In

- Dependencies

```
implementation 'com.google.firebase:firebase-auth:19.2.0'
implementation 'com.google.android.gms:play-services-auth:17.0.0'
```

Authenticate Using Google Sign-In

- Dependencies
- Integrate Google Sign-In

```
// Configure Google Sign In
val gso = GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
        .requestIdToken(getString(R.string.default_web_client_id))
        .requestEmail()
        .build()
```

Authenticate Using Google Sign-In

- Dependencies
- Integrate Google Sign-In

```
// Configure Google Sign In
val gso = GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
        .requestIdToken(getString(R.string.default_web_client_id))
        .requestEmail()
        .build()

private fun signIn() {
    val signInIntent = gso.signInIntent
    startActivityForResult(signInIntent, RC_SIGN_IN)
}
```

Authenticate Using Google Sign-In

- Dependencies

- Integrate Google Sign-In

```
public override fun onActivityResult(requestCode: Int, resultCode: Int,
      data: Intent?) {
  super.onActivityResult(requestCode, resultCode, data)

  // Result returned from launching the Intent from
  // GoogleSignInApi.getSignInIntent(...);
  if (requestCode == RC_SIGN_IN) {
    val task = GoogleSignIn.getSignedInAccountFromIntent(data)
    try {
      // Google Sign In was successful, authenticate with Firebase
      val account = task.getResult(ApiException::class.java)
      firebaseAuthWithGoogle(account!!)
    } catch (e: ApiException) {
      // Google Sign In failed, update UI appropriately
      Log.w(TAG, "Google sign in failed", e)
      // ...
    }
  }
}
```

Authenticate Using Google Sign-In

- Dependencies
- Integrate Google Sign-In
- Use shared auth

```
private lateinit var auth: FirebaseAuth
// ...
// Initialize Firebase Auth
auth = FirebaseAuth.getInstance()

// ...

public override fun onStart() {
    super.onStart()
    // Check if user is signed in (non-null) and update UI accordingly.
    val currentUser = auth.currentUser
    updateUI(currentUser)
}
```

Authenticate Using Google Sign-In

- Dependencies
- Integrate Google Sign-In
- Use shared auth
- Use the token

```
private fun firebaseAuthWithGoogle(acct: GoogleSignInAccount) {
  Log.d(TAG, "firebaseAuthWithGoogle:" + acct.id!!)

  val credential = GoogleAuthProvider.getCredential(acct.idToken, null)
  auth.signInWithCredential(credential)
    .addOnCompleteListener(this) { task ->
      if (task.isSuccessful) {
        // Sign in success, update UI with the signed-in user's information
        Log.d(TAG, "signInWithCredential:success")
        val user = auth.currentUser
        updateUI(user)
      } else {
        // If sign in fails, display a message to the user.
        Log.w(TAG, "signInWithCredential:failure", task.exception)
```

Authenticate Using Google Sign-In

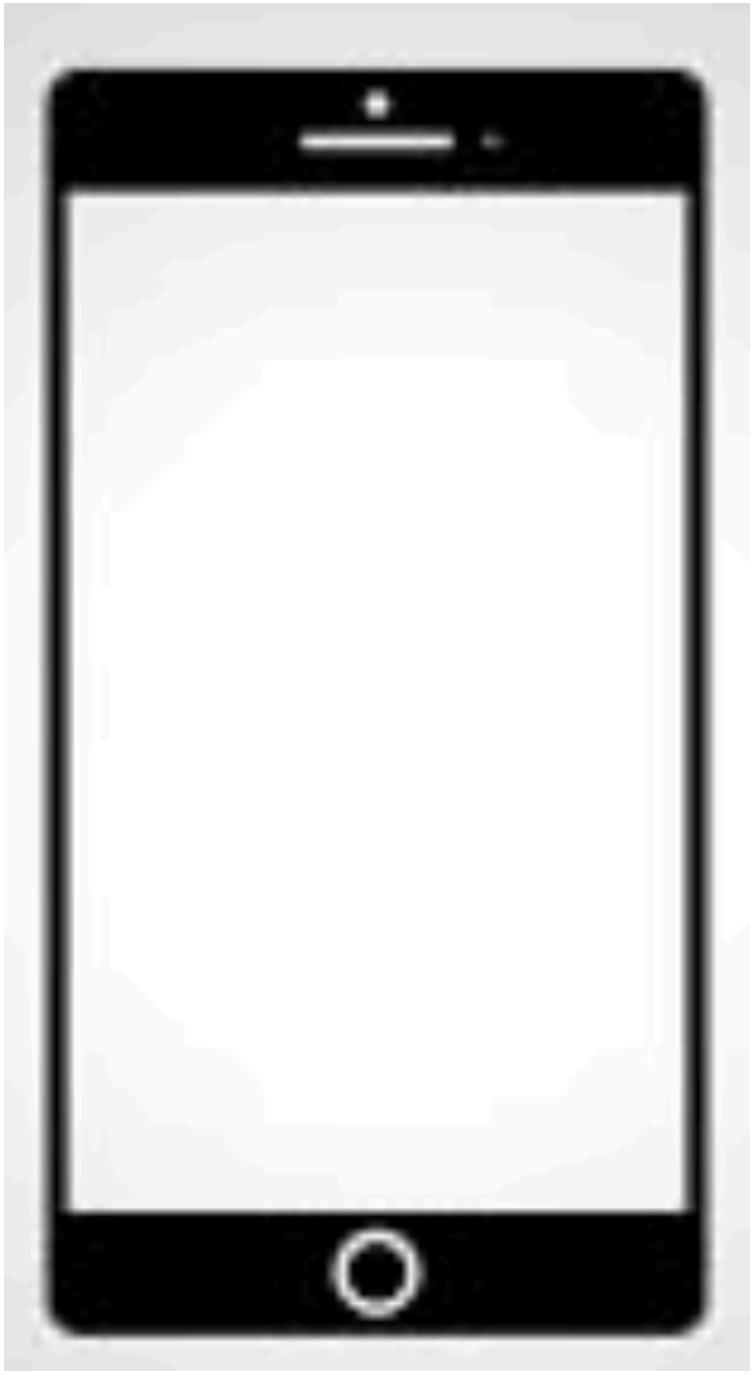
```
private fun firebaseAuthWithGoogle(acct: GoogleSignInAccount) {
  Log.d(TAG, "firebaseAuthWithGoogle:" + acct.id!!)

  val credential = GoogleAuthProvider.getCredential(acct.idToken, null)
  auth.signInWithCredential(credential)
    .addOnCompleteListener(this) { task ->
      if (task.isSuccessful) {
        // Sign in success, update UI with the signed-in user's information
        Log.d(TAG, "signInWithCredential:success")
        val user = auth.currentUser
        updateUI(user)
      } else {
        // If sign in fails, display a message to the user.
        Log.w(TAG, "signInWithCredential:failure", task.exception)
        Snackbar.make(main_layout, "Authentication Failed.",
Snackbar.LENGTH_SHORT).show()
        updateUI(null)
      }

  // ...
  }
}
```

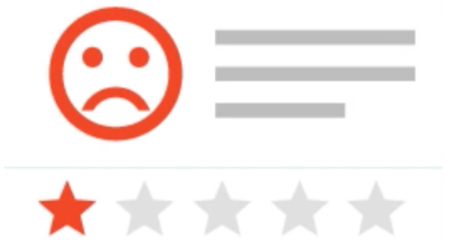
Sign out a User

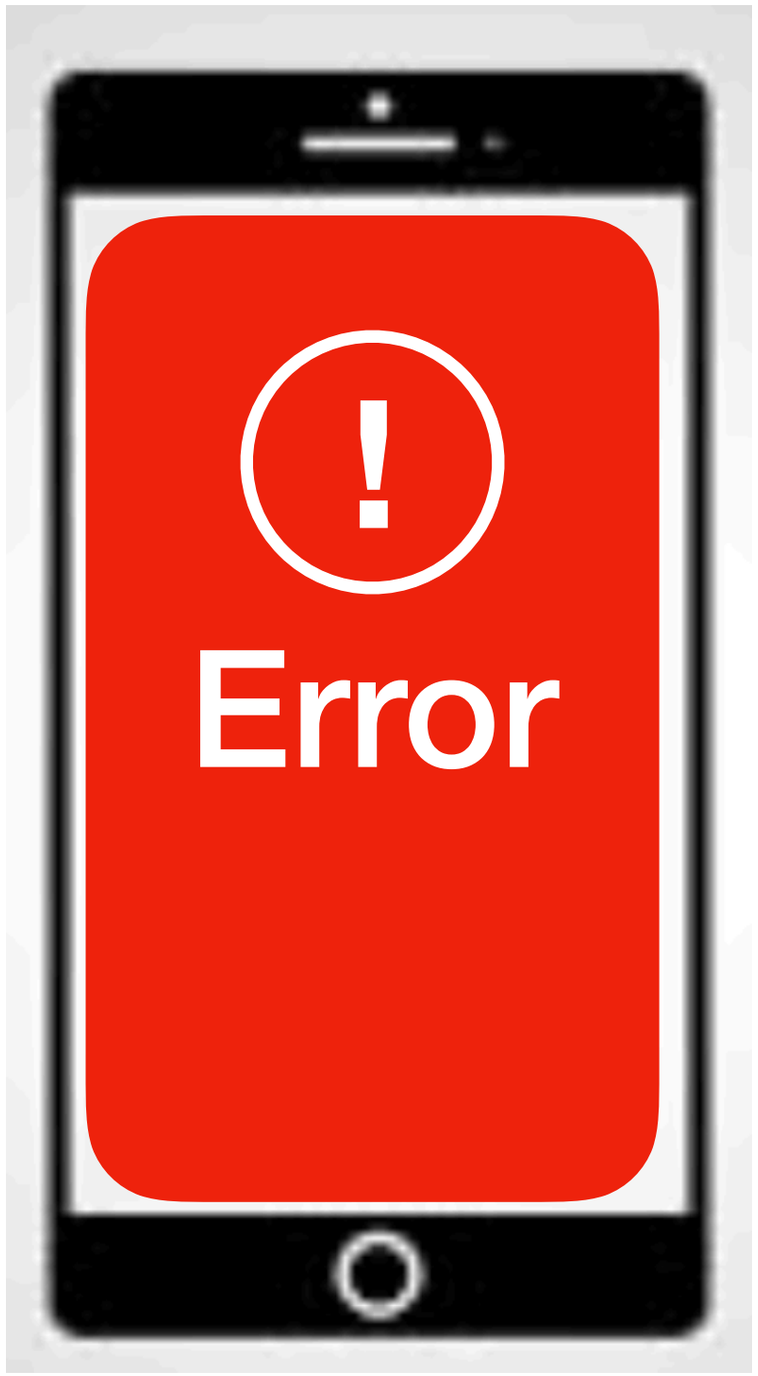
```
FirebaseAuth.getInstance().signOut()
```

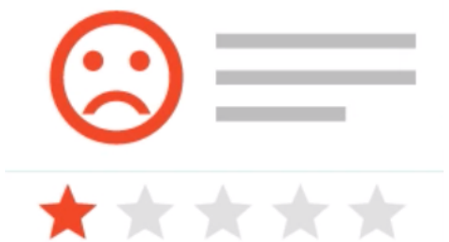


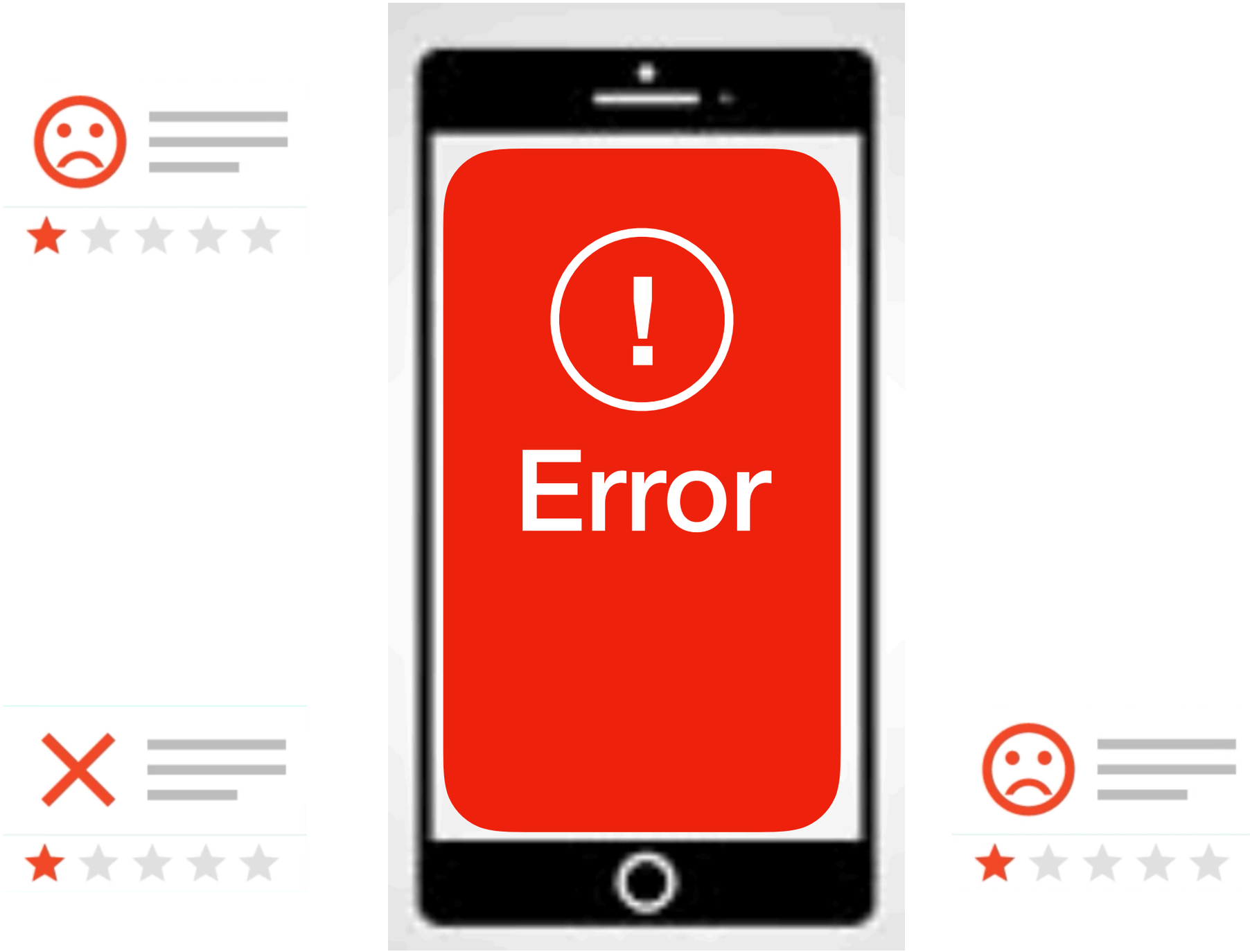

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacus. Phasellus auctor faucibus facilisis.
Sed nec arcu nibh. Maecenas nulla velit, hendrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.

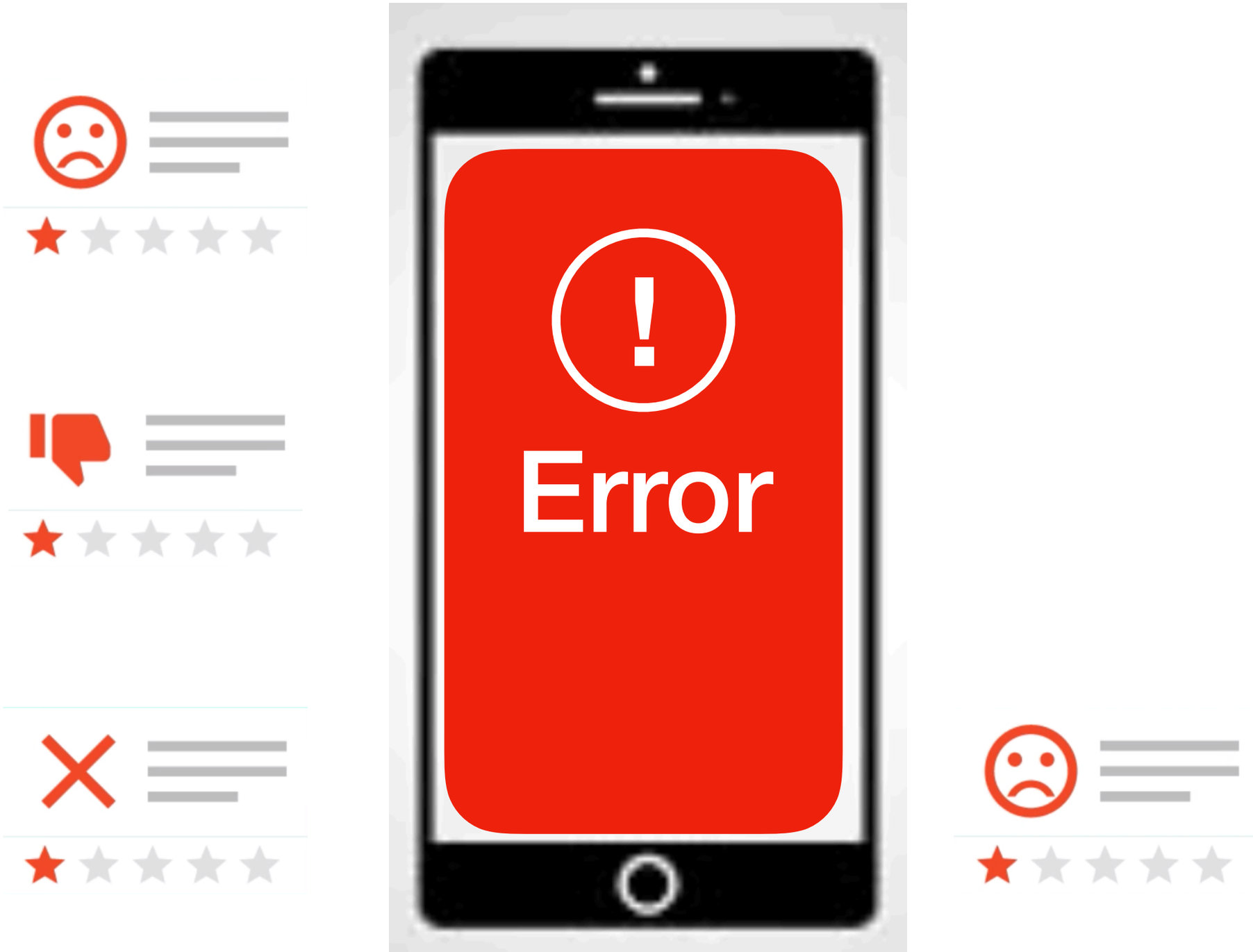

!
Error

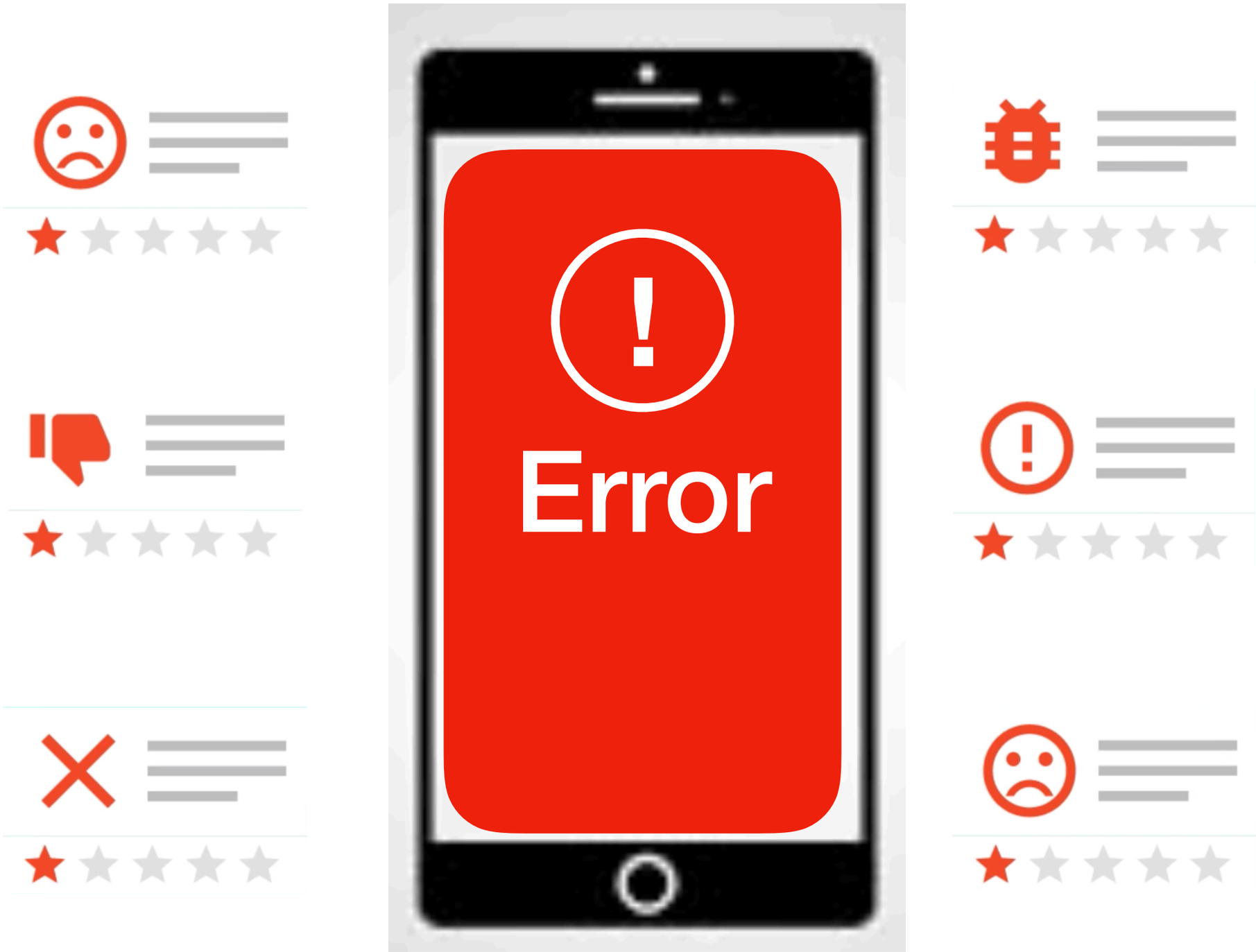


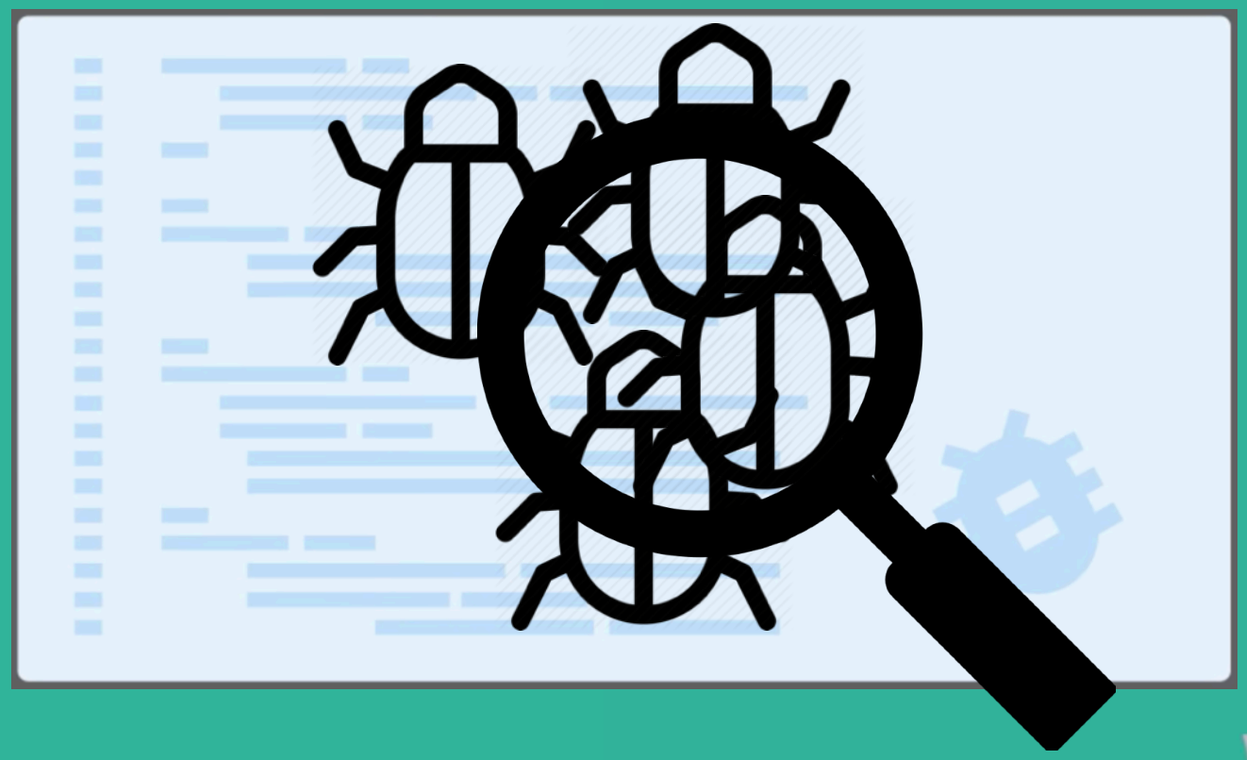

!
Error



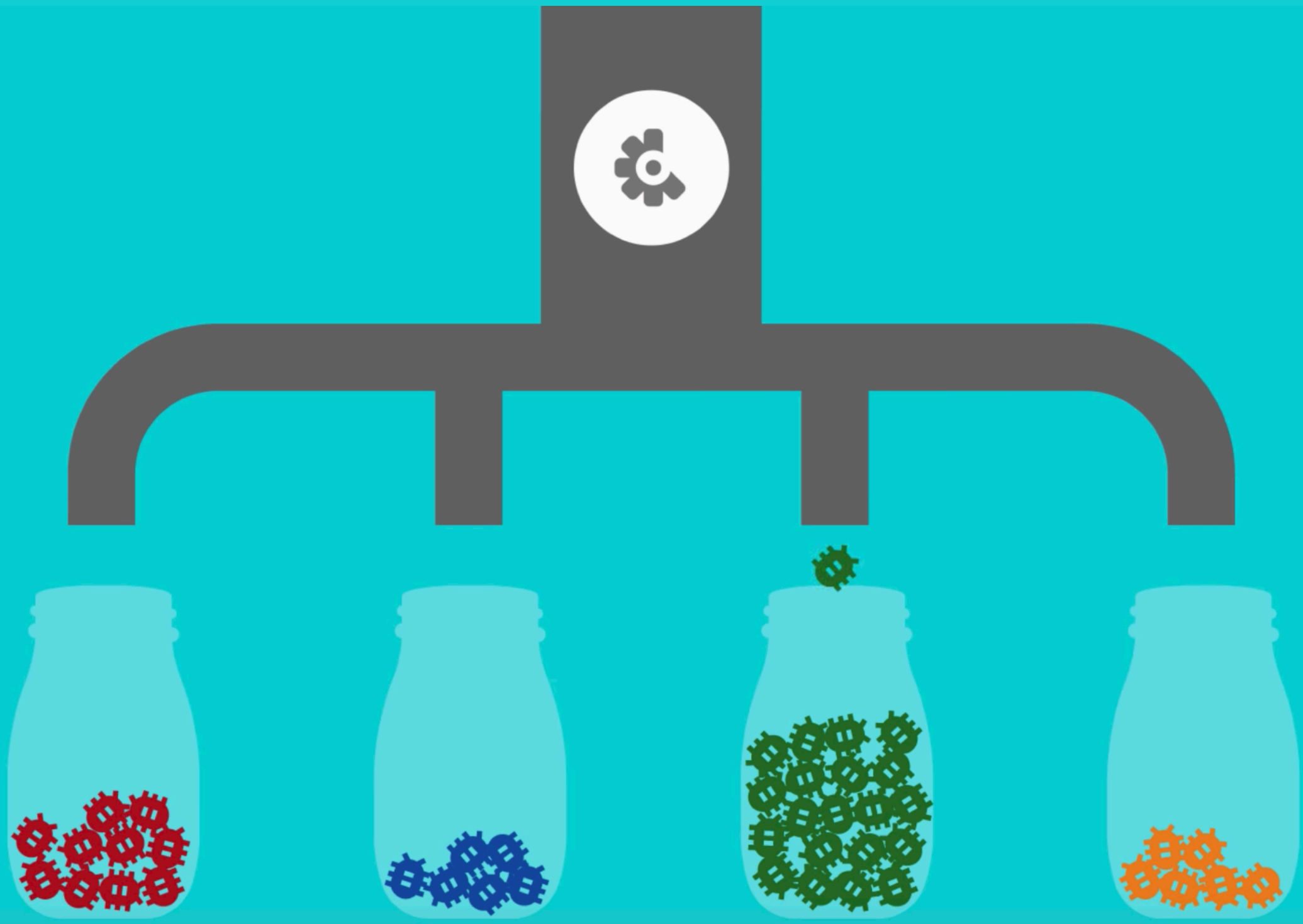

!
Error


!
Error


Error



Crashlytics



Enable the SDK

```
buildscript {
    repositories {
        // Add the following repositories:
        google()  // Google's Maven repository
        maven {
          url 'https://maven.fabric.io/public'
        }
    }
    dependencies {
        // ...
        classpath 'com.google.gms:google-services:4.3.3'  // Google Services plugin
        // Add dependency
        classpath 'io.fabric.tools:gradle:1.31.2'  // Crashlytics plugin
    }
}
allprojects {
    repositories {
      // Check that you have the following line (if not, add it):
      google()  // Google's Maven repository
    }
}
```

Enable the SDK

```
apply plugin: 'com.android.application'
apply plugin: 'io.fabric'

dependencies {
    // ...
    // (Recommended) Add Analytics
    implementation 'com.google.firebase:firebase-analytics:17.2.1'
    // Add dependency
    implementation 'com.crashlytics.sdk.android:crashlytics:2.10.1'
}
```

Test Implementation

```
val crashButton = Button(this)
crashButton.text = "Crash!"
crashButton.setOnClickListener {
    Crashlytics.getInstance().crash() // Force a crash
}
```

Customize

AndroidManifest.xml

```
<meta-data
android:name="firebase_crashlytics_collection_enabled"
android:value="false" />
```

Customize

AndroidManifest.xml

```
<meta-data
    android:name="firebase_crashlytics_collection_enabled"
    android:value="false" />
```

Enable collection for selected users:

```
Fabric.with(this, Crashlytics())
```

Customize

AndroidManifest.xml

```
<meta-data
    android:name="firebase_crashlytics_collection_enabled"
    android:value="false" />
```

Enable collection for selected users:

```
Fabric.with(this, Crashlytics())
```

Crash report and `Log.println:`

```
Crashlytics.log(Log.DEBUG, "tag", "message")
```

Customize

`AndroidManifest.xml`

```
<meta-data
    android:name="firebase_crashlytics_collection_enabled"
    android:value="false" />
```

Enable collection for selected users:

```
Fabric.with(this, Crashlytics())
```

Crash report and `Log.println`:

```
Crashlytics.log(Log.DEBUG, "tag", "message")
```

Crash report only:

```
Crashlytics.log("message")
```

Customize

`AndroidManifest.xml`

```
<meta-data
    android:name="firebase_crashlytics_collection_enabled"
    android:value="false" />
```

Enable collection for selected users:

```
Fabric.with(this, Crashlytics())
```

Crash report and `Log.println`:

```
Crashlytics.log(Log.DEBUG, "tag", "message")
```

Crash report only:

```
Crashlytics.log("message")
```

Add custom keys:

```
Crashlytics.setString(key, "foo" /* string value */)
Crashlytics.setBool(key, true /* boolean value */)
Crashlytics.setDouble(key, 1.0 /* double value */)
Crashlytics.setFloat(key, 1.0f /* float value */)
Crashlytics.setInt(key, 1 /* int value */)
```

Customize

`AndroidManifest.xml`

```
<meta-data
    android:name="firebase_crashlytics_collection_enabled"
    android:value="false" />
```

Enable collection for selected users:

```
Fabric.with(this, Crashlytics())
```

Crash report and `Log.println:`

```
Crashlytics.log(Log.DEBUG, "tag", "message")
```

Crash report only:

```
Crashlytics.log("message")
```

Add custom keys:

```
Crashlytics.setInt("current_level", 3)
Crashlytics.setString("last_UI_action", "teamed the dragon!")
```

Customize

AndroidManifest.xml

```
<meta-data
    android:name="firebase_crashlytics_collection_enabled"
    android:value="false" />
```

Enable collection for selected users:

```
Fabric.with(this, Crashlytics())
```

Crash report and `Log.println:`

```
Crashlytics.log(Log.DEBUG, "tag", "message")
```

Crash report only:

```
Crashlytics.log("message")
```

Add custom keys:

```
Crashlytics.setInt("current_level", 3)
Crashlytics.setString("last_UI_action", "teamed the dragon!")
```

Set user IDs

```
Crashlytics.setUserIdentifier("user123456789")
```

Customize

AndroidManifest.xml

```
<meta-data
    android:name="firebase_crashlytics_collection_enabled"
    android:value="false" />
```

Enable collection for selected users:

```
Fabric.with(this, Crashlytics())
```

Crash report and `Log.println:`

```
Crashlytics.log(Log.DEBUG, "tag", "message")
```

Crash report only:

```
Crashlytics.log("message")
```

Add custom keys:

```
Crashlytics.setInt("current_level", 3)
Crashlytics.setString("last_UI_action", "teamed the dragon!")
```

Set user IDs

```
Crashlytics.setUserIdentifier("user123456789")
```

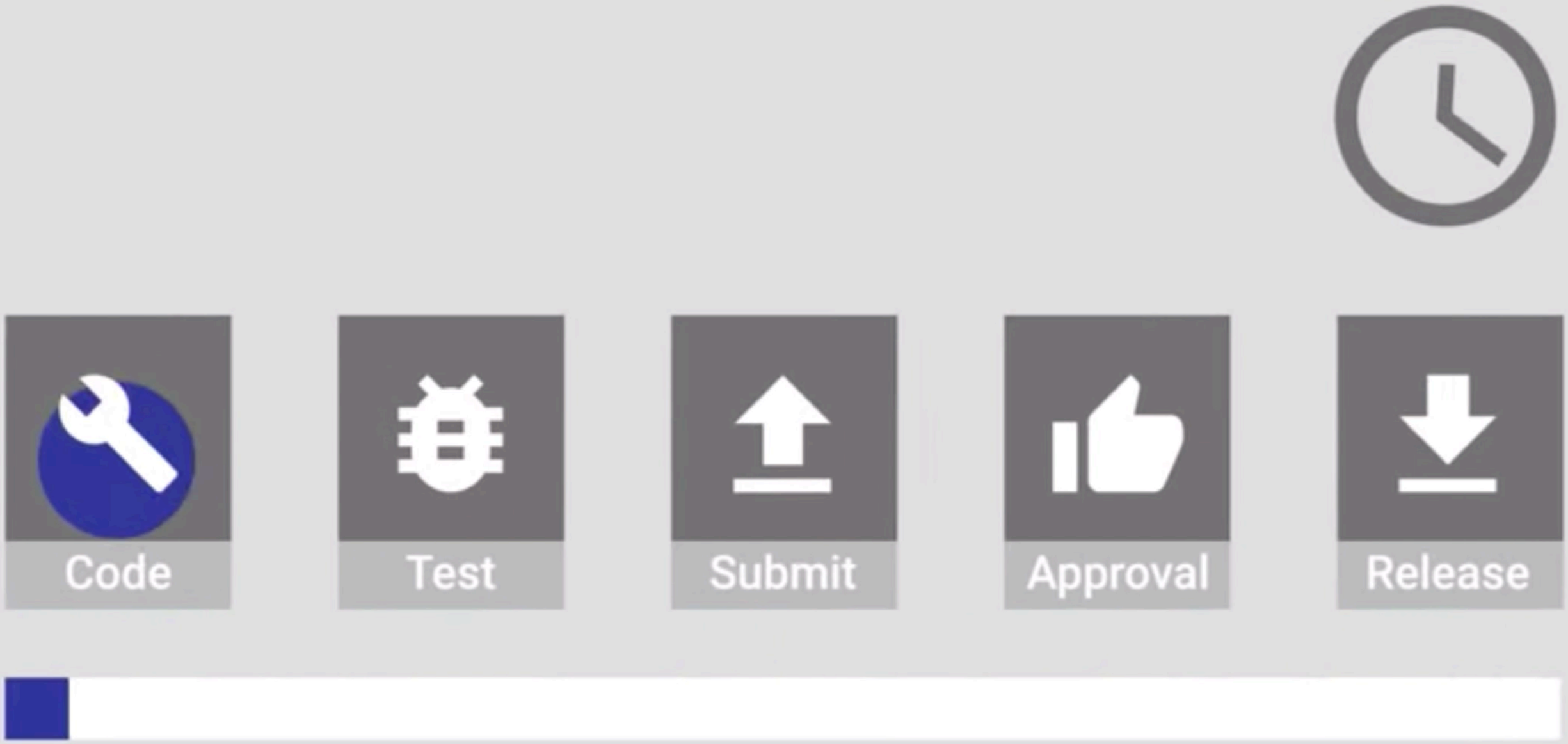
Log non-fatal exceptions

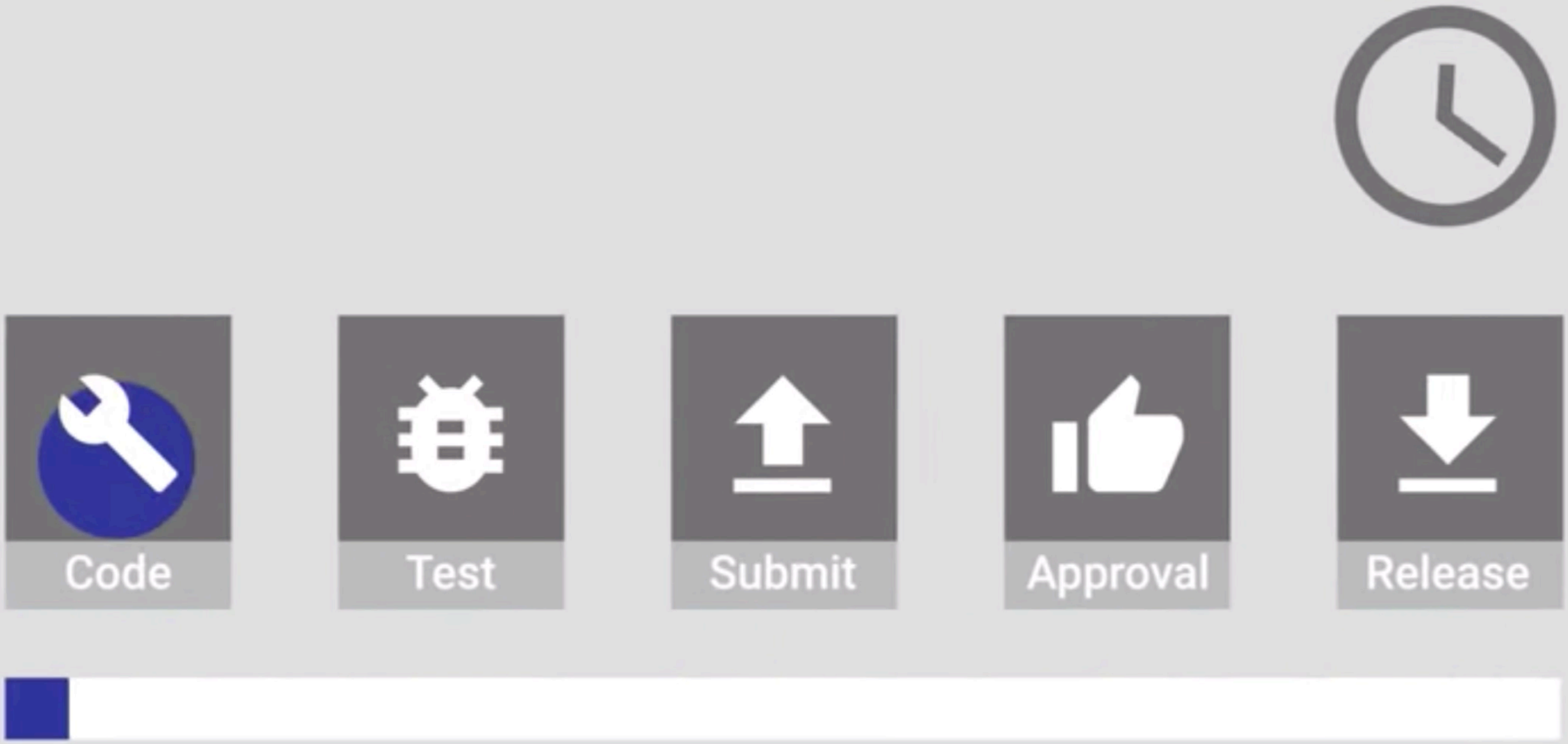
```
Crashlytics.logException(e)
```

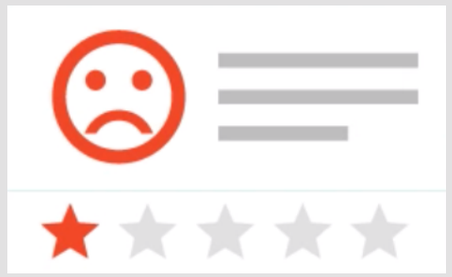
https://firebase.google.com/docs/crashlytics/customize-crash-reports

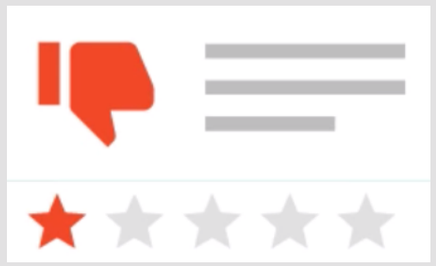

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacus. Phasellus auctor faucibus facilisis.
Sed nec arcu nibh. Maecenas nulla velit, hendrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.

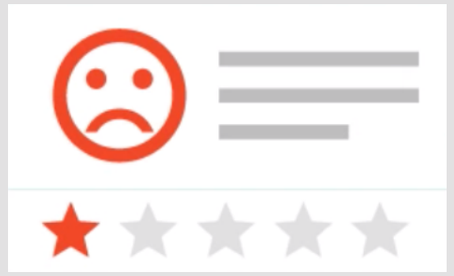

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacus. Phasellus auctor faucibus facilisis.
Sed nec arcu nibh. Maecenas nulla velit, hendrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.


Code
Test
Submit
Approval
Release


Code
Test
Submit
Approval
Release








Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacus. Phasellus auctor faucibus facilisis.
Sed nec arcu nibh. Maecenas nulla velit, hendrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacus. Phasellus auctor faucibus facilisis.
Sed nec arcu nibh. Maecenas nulla velit, hendrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacus. Phasellus auctor faucibus facilisis.
Sed nec arcu nibh. Maecenas nulla velit, hendrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacus. Phasellus auctor faucibus facilisis.
ed nec arcu nibh. Maecenas nulla velit, endrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.

Realtime

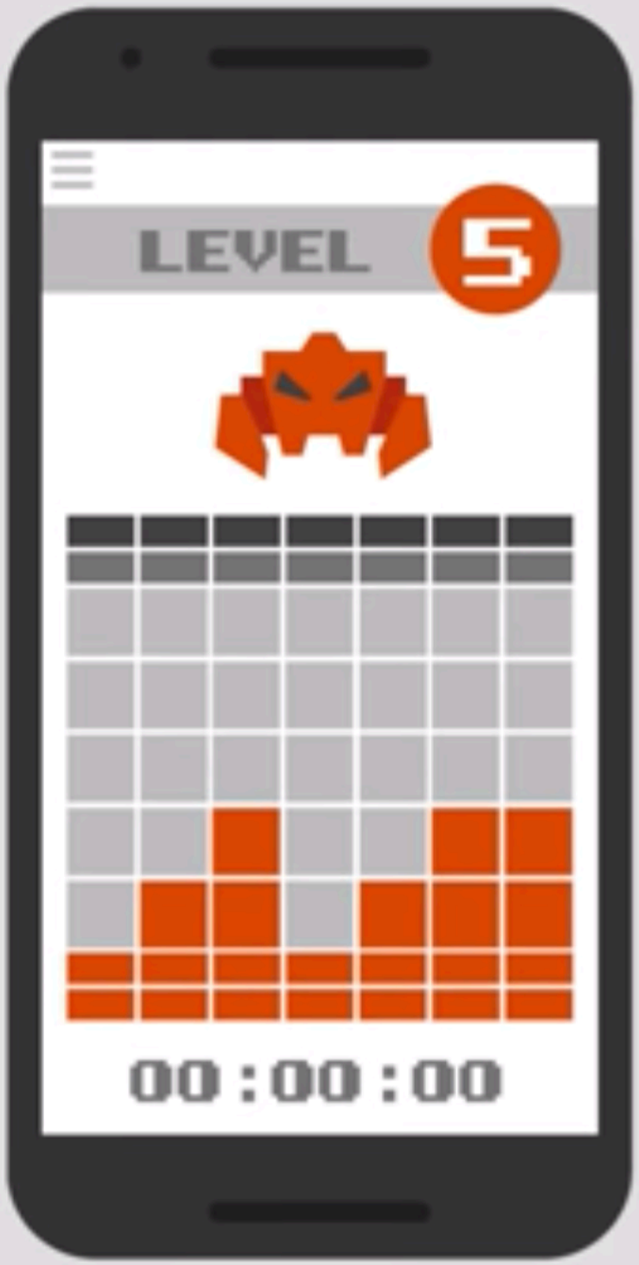

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada eros. In nec mi consequat, laoreet massa et, molestie nisi. Proin eget dui lacus. Phasellus auctor faucibus facilisis.
ed nec arcu nibh. Maecenas nulla velit, endrerit at arcu ut, viverra ullamcorper eros. Proin efficitur libero in urna gravida, dictum tempor nibh bibendum. Ut eu libero non leo elementum molestie. Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.

Realtime

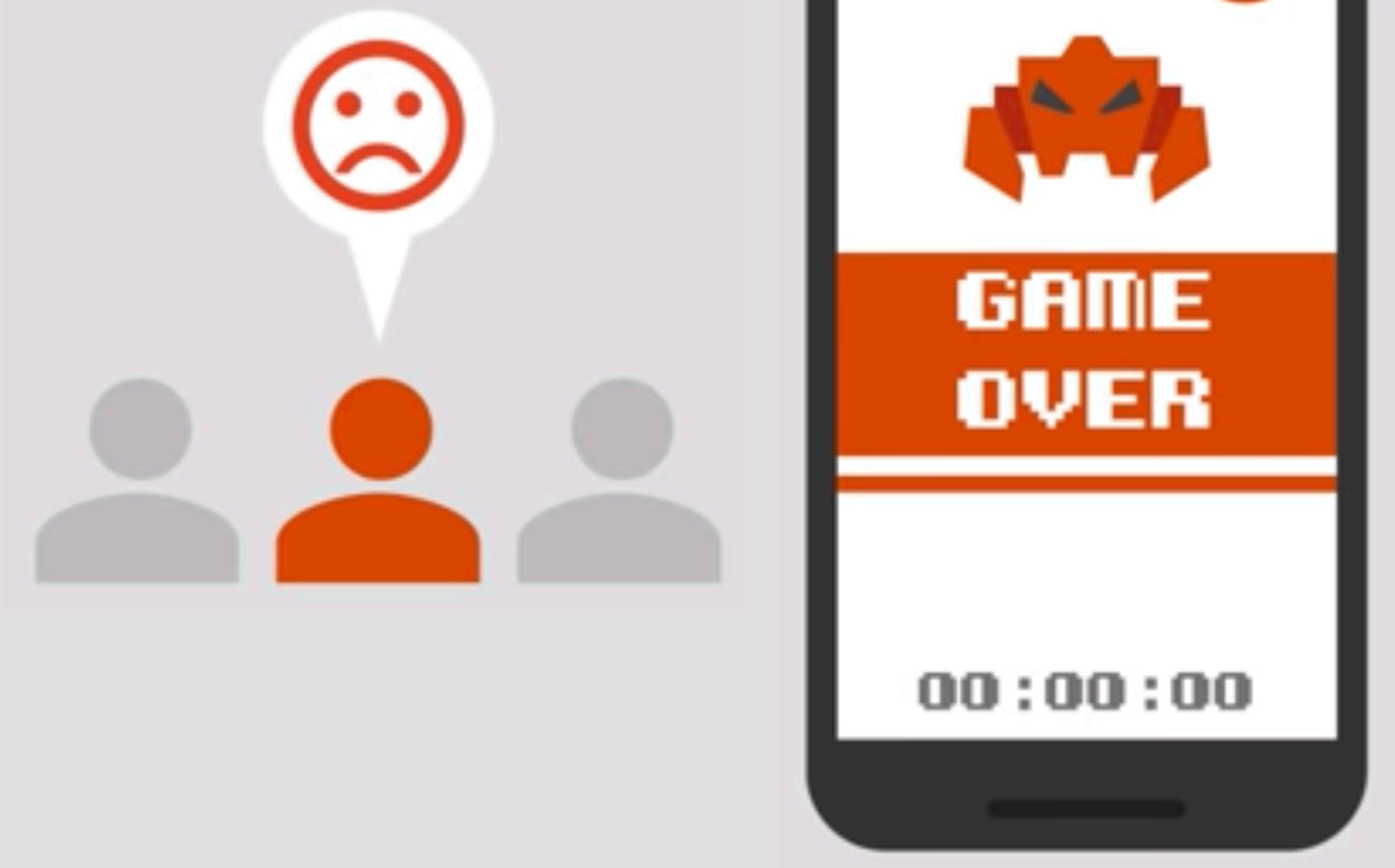

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada
ros. In nec mi consequat, laoreet massa
molestie nisi. Proin eget dui lacus.
us auctor faucibus facilisis.
rcu nibh. Maecenas nulla velit,
at arcu ut, viverra ullamcorper
roin efficitur libero in urna gravida,
um tempor nibh bibendum. Ut eu
ibero non leo elementum molestie.
Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.

Realtime


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas rutrum est ac ultrices porttitor. Suspendisse pellentesque risus vehicula, vehicula erat a, tempor nisi. Donec vel suscipit sapien, vitae sollicitudin dui. Fusce vitae nulla ornare, sollicitudin nibh et, malesuada
ros. In nec mi consequat, laoreet massa
molestie nisi. Proin eget dui lacus.
us auctor faucibus facilisis.
rcu nibh. Maecenas nulla velit,
at arcu ut, viverra ullamcorper
roin efficitur libero in urna gravida,
um tempor nibh bibendum. Ut eu
ibero non leo elementum molestie.
Nullam convallis convallis tellus, vitae ultrices nisi bibendum non. Suspendisse placerat lorem augue, nec bibendum nisi mollis non. Mauris viverra posuere diam in laoreet. Sed non erat rutrum, bibendum risus a, molestie arcu. Praesent sit amet arcu ut libero faucibus lacinia.


LEVEL
5
00 : 00 : 00

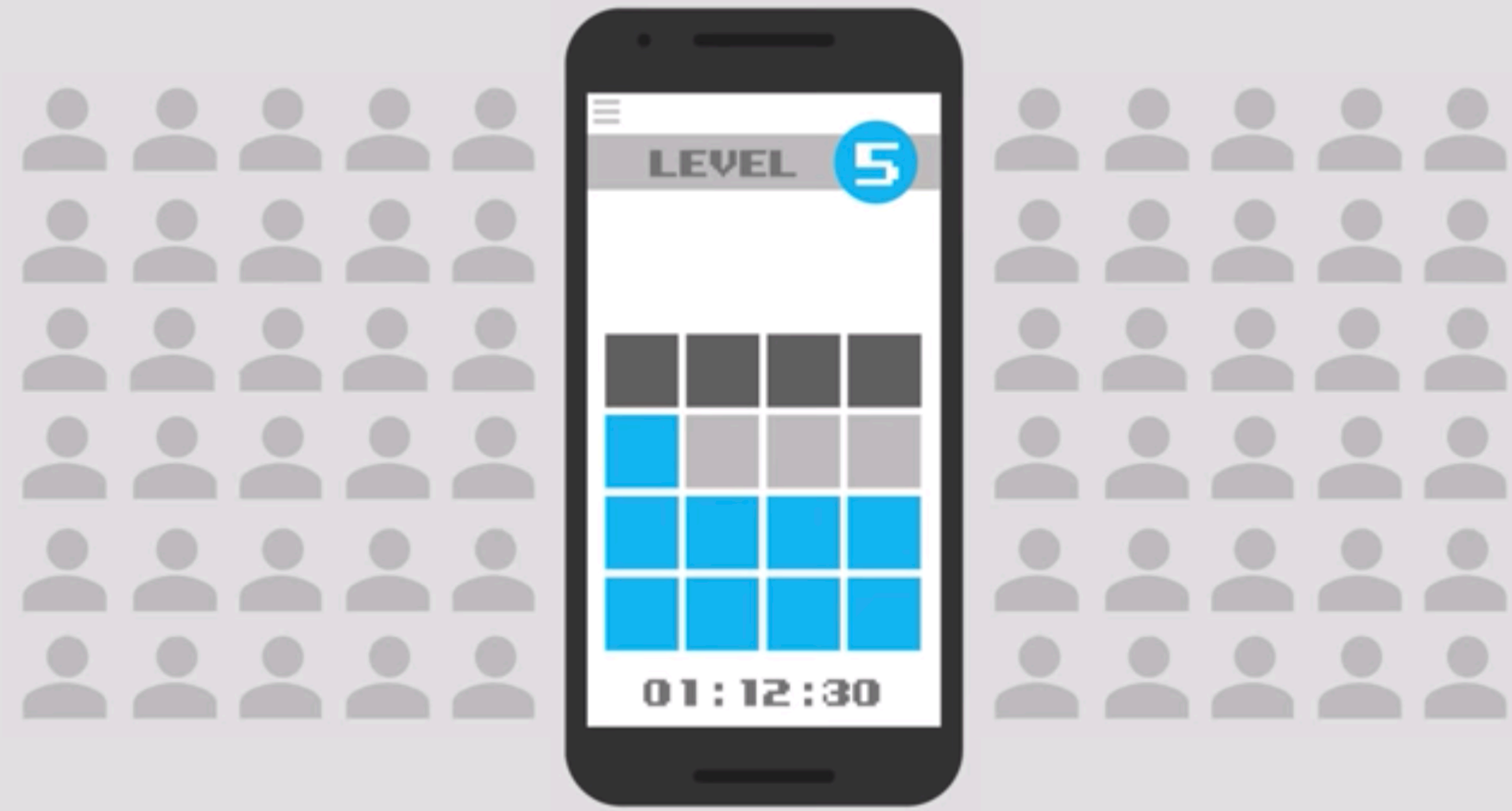

LEVEL
5
GAME
OVER
00 : 00 : 00

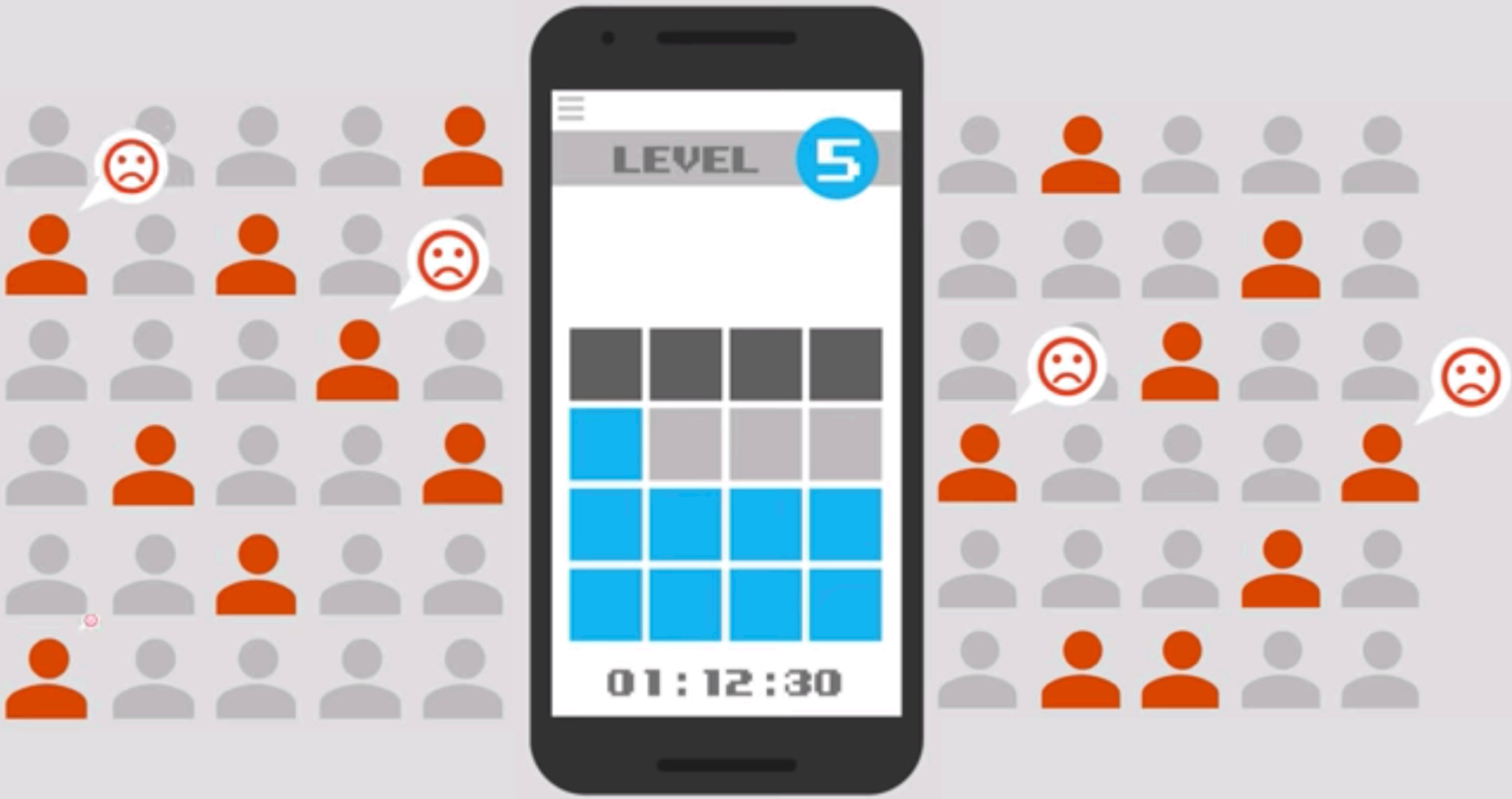

LEVEL
5
GAME
OVER
00 : 00 : 00

Realtime

Realtime

Realtime


LEVEL
5
01:12:30


LEVEL
5
01:12:30

A

B

Remote Config

Using Remote Config

1 Add Firebase to your app

```
implementation ‘com.google.firebase:firebase-config:19.0.3'
```

Using Remote Config

1 Add Firebase to your app

```
implementation ‘com.google.firebase:firebase-config:19.0.3'
```

2 Get the Remote Config singleton object

```
FirebaseRemoteConfig.getInstance()
```

Using Remote Config

1 Add Firebase to your app

```
implementation ‘com.google.firebase:firebase-config:19.0.3'
```

2 Get the Remote Config singleton object

```
FirebaseRemoteConfig.getInstance()
```

3 Set in-app default parameter values

Using Remote Config

1 Add Firebase to your app

```
implementation ‘com.google.firebase:firebase-config:19.0.3'
```

2 Get the Remote Config singleton object

```
FirebaseRemoteConfig.getInstance()
```

3 Set in-app default parameter values

4 Get parameter values to use in your app

Using Remote Config

1. Add Firebase to your app

```
implementation 'com.google.firebase:firebase-config:19.0.3'
```

2. Get the Remote Config singleton object

```
FirebaseRemoteConfig.getInstance()
```

3. Set in-app default parameter values

4. Get parameter values to use in your app

5. Set parameter values in the service (as needed)

Using Remote Config

1. Add Firebase to your app

```
implementation 'com.google.firebase:firebase-config:19.0.3'
```

2. Get the Remote Config singleton object

```
FirebaseRemoteConfig.getInstance()
```

3. Set in-app default parameter values

4. Get parameter values to use in your app

5. Set parameter values in the service (as needed)

6. Fetch and activate values from the service (as needed)

Using Remote Config

DEMO

1. Add Firebase to your app

```
implementation 'com.google.firebase:firebase-config:19.0.3'
```

2. Get the Remote Config singleton object

```
FirebaseRemoteConfig.getInstance()
```

3. Set in-app default parameter values

4. Get parameter values to use in your app

5. Set parameter values in the service (as needed)

6. Fetch and activate values from the service (as needed)

https://firebase.google.com/docs/remote-config/use-config-android

Make $

- In-app purchases.
- Subscriptions.
- Advertising.
- Paid apps.
- E-commerce.

Make $

- In-app purchases.
- Subscriptions.
- Advertising.
- Paid apps.
- E-commerce.

https://developer.android.com/distribute/best-practices/earn/monetization-options

AdMob

Import the Mobile Ads SDK

```
implementation 'com.google.firebase:firebase-ads:18.3.0'
```

AdMob

Import the Mobile Ads SDK

```
implementation 'com.google.firebase:firebase-ads:18.3.0'
```

Update your AndroidManifest.xml

```
<manifest>
  <application>
    <!-- Sample AdMob App ID: ca-app-pub-3940256099942544~3347511713 -->
    <meta-data
      android:name="com.google.android.gms.ads.APPLICATION_ID"
      android:value="[ADMOB_APP_ID]"/>
  </application>
</manifest>
```

AdMob

Import the Mobile Ads SDK

```
implementation 'com.google.firebase:firebase-ads:18.3.0'
```

Update your AndroidManifest.xml

```
<manifest>
  <application>
    <!-- Sample AdMob App ID: ca-app-pub-3940256099942544~3347511713 -->
    <meta-data
      android:name="com.google.android.gms.ads.APPLICATION_ID"
      android:value="[ADMOB_APP_ID]"/>
  </application>
</manifest>
```

Initialize the SDK

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    // Sample AdMob app ID: ca-app-pub-3940256099942544~3347511713
    MobileAds.initialize(this, "YOUR_ADMOB_APP_ID")
}
```

Ad Formats

Ad Formats

- Banner

```
<com.google.android.gms.ads.AdView
    xmlns:ads="http://schemas.android.com/apk/res-auto"
    android:id="@+id/adView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_alignParentBottom="true"
    ads:adSize="BANNER"
    ads:adUnitId="ca-app-pub-3940256099942544/6300978111">
</com.google.android.gms.ads.AdView>
```

Ad Formats

- Banner

```
<com.google.android.gms.ads.AdView
    xmlns:ads="http://schemas.android.com/apk/res-auto"
    android:id="@+id/adView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_alignParentBottom="true"
    ads:adSize="BANNER"
    ads:adUnitId="ca-app-pub-3940256099942544/6300978111">
 </com.google.android.gms.ads.AdView>
```

```
val adView = AdView(this)
adView.adSize = AdSize.BANNER
adView.adUnitId = "ca-app-pub-3940256099942544/6300978111"
// TODO: Add adView to your view hierarchy.
```

https://developers.google.com/admob/android/banner

Ad Formats

- Banner
- Interstitial

```
class MainActivity : Activity() {

    private lateinit var mInterstitialAd: InterstitialAd

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        MobileAds.initialize(this,
            "ca-app-pub-3940256099942544~3347511713")

        mInterstitialAd = InterstitialAd(this)
        mInterstitialAd.adUnitId = "ca-app-pub-3940256099942544/1033173712"
        mInterstitialAd.loadAd(AdRequest.Builder().build())
    }
}
```

https://developers.google.com/admob/android/interstitial

Ad Formats

- Banner
- Interstitial
- Native

```
val adLoader = AdLoader.Builder(this, "ca-app-pub-3940256099942544/2247696110")
        .forUnifiedNativeAd { ad : UnifiedNativeAd ->
            // Show the ad.
        }
        .withAdListener(object : AdListener() {
            override fun onAdFailedToLoad(errorCode: Int) {
                // Handle the failure by logging, altering the UI, and so on.
            }
        })
        .withNativeAdOptions(NativeAdOptions.Builder()
                // Methods in the NativeAdOptions.Builder class can be
                // used here to specify individual options settings.
                .build())
        .build()
```

https://developers.google.com/admob/android/native/start

Ad Formats

- Banner
- Interstitial
- Native
- Rewarded Video

```
import com.google.android.gms.ads.AdRequest
import com.google.android.gms.ads.MobileAds
import com.google.android.gms.ads.reward.RewardedVideoAd

class MainActivity : AppCompatActivity(), RewardedVideoAdListener {

private lateinit var mRewardedVideoAd: RewardedVideoAd

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        MobileAds.initialize(this, "ca-app-pub-3940256099942544~3347511713")

        // Use an activity context to get the rewarded video instance.
```

DEMO

Ad Formats

```
import com.google.android.gms.ads.AdRequest
import com.google.android.gms.ads.MobileAds
import com.google.android.gms.ads.reward.RewardedVideoAd

class MainActivity : AppCompatActivity(), RewardedVideoAdListener {

    private lateinit var mRewardedVideoAd: RewardedVideoAd

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        MobileAds.initialize(this, "ca-app-pub-3940256099942544~3347511713")

        // Use an activity context to get the rewarded video instance.
        mRewardedVideoAd = MobileAds.getRewardedVideoAdInstance(this)
        mRewardedVideoAd.rewardedVideoAdListener = this
    }
    ...
}
```

https://developers.google.com/admob/android/rewarded-video

DEMO

Google Play Billing Overview

- Types of in-app products:
  - One-time products.
  - Subscriptions.
- In-app product configuration options:
  - Title.
  - Description.
  - Product ID.
  - Price / Default Price.

https://developer.android.com/google/play/billing/billing_overview

Lecture outcomes

- Use Firebase Realtime Database.
- How to use Remove Config with A-B testing.
- Authenticate your users using Firebase, with GoogleSignIn and EmailPassword methods.
- How to use Firebase Realtime Database.
- Using AdMob to display ads.