Mobilni i distribuirani informacioni sistemi

- Mobilni servisi, razmena poruka i notifikacija –

Katedra za računarstvo
Elektronski fakultet u Nišu

Literatura

- *Mobile Developer's Guide To The Galaxy,* 18th Edition, 2019
- *Android Programming The Big Nerd Ranch Guide ,* 3rd ed, Bill Phillips, Chris Stewart, Brian Hardy And Kristin Marsicano, 2017
  - https://www.bignerdranch.com/books/android-programming/

Web servisi

- SOAP Web servisi
  - SOAP, WSDL, UDDI
  - XML poruke (SOAP)
- RESTful Web servisi
  - HTTP (GET, POST, PUT, DELETE,...)
  - JSON

Android i Web servisi

- Za implementaciju mrežnih aplikacija Android koristi *java.net* paket
  - *Low level API* (*Addresses, Sockets, Interfaces)*
  - *High level API* (*URIs, URLs, Connections*)
- Za pristup Web resursima/servisima
  - *HttpURLConnection*
- Za pristup Internetu je potrebna dozvola (*permission*) u *AndroidManifest.xml*

```
<uses-permission android:name="android.permission.INTERNET" />
```

- Sva mrežna komunikacija obavlja se u posebnoj niti, na primer korišćenjem *AsyncTask-a*

Android i Web servisi (2)

- Primer pristupa URL-u i čitanje sadržaja kao niza bajtova

```
public byte[] getUrlBytes(String urlSpec) throws IOException {
    URL url = new URL(urlSpec);
    HttpURLConnection connection = (HttpURLConnection)url.openConnection();

    try {
        ByteArrayOutputStream out = new ByteArrayOutputStream();
        InputStream in = connection.getInputStream();

        if (connection.getResponseCode() != HttpURLConnection.HTTP_OK) {
            throw new IOException(connection.getResponseMessage() +
                    ": with " +
                    urlSpec);
        }

        int bytesRead = 0;
        byte[] buffer = new byte[1024];
        while ((bytesRead = in.read(buffer)) > 0) {
            out.write(buffer, 0, bytesRead);
        }
        out.close();
        return out.toByteArray();
    } finally {
        connection.disconnect();
    }
```

- Sadržaj resursa se može prevesti u *String* i parsirati korišćenjem JSON API (*org.json*) ili DOM API (*org.w3c.dom*)

Android HTTP biblioteke

- Google Volley API
  - https://developer.android.com/training/volley/index.html
- OkHttp - HTTP+HTTP/2 client for Android and Java applications
  - https://github.com/square/okhttp
  - Retrofit - type-safe HTTP client for Android and Java
    - http://square.github.io/retrofit/
  - Picasso - powerful image downloading and caching library for Android
    - http://square.github.io/picasso/

Razmena poruka (*messaging*)

- Odnos pošiljaoca i primaoca poruke
  - Sinhrona – pošiljalac poruke se blokira i čeka odgovor na poruku pre nego što nastavi sa radom (RPC, CORBA, RMI, DCOM, HTTP *request/response*, itd.)
  - Asinhrona - pošiljalac poruke nakon slanja nastavlja sa radom, a kad odgovor stigne vrši njegovu obradu
- Inicijator razmene poruka
  - ***Pull*** – Klijent periodično ispituje da li postoji raspoloživa informacija na serveru, i ako postoji zahteva slanje od servera
  - ***Push*** – Server inicira komunikaciju šaljući informaciju klijentu bez njegove eksplicitne interakcije ili zahteva (HDML *notification*, WAP Push, *application-to-application messaging*)

Tipovi razmene poruka

- E-mail
- *Paging* i pager sistemi
- Short Message Servise (SMS)
  - 160 znakova teksta
- *Enhanced Message Service* (EMS)
  - tekst, slike, animacije, zvuk i formatiran tekst
- *Multimedia Message Service* (MMS)
  - Pored formatiranog teksta, slike i zvuka, podrška za prenos audio i video sadržaja i glasa veličine do 300KB
  - Standardizovan od strane Open Mobile Alliance (OMA) i 3rd Generation Partnership Project (3GPP)
- HDML notifikacija
- WAP Push
- *Mobile Instant Messaging*
  - Po funkcionalnosti odgovara tehnologiji dvosmerne razmene tekstualnih, slikovnih i video poruka uz informaciju o prisustvu – Skype, Viber, WhatsApp
- Razmena poruka između aplikacija (*Application-to-Application Messaging*)
- *Push notification service* (APNS, FCM/GCM, MPNS)

Razmena poruka između aplikacija

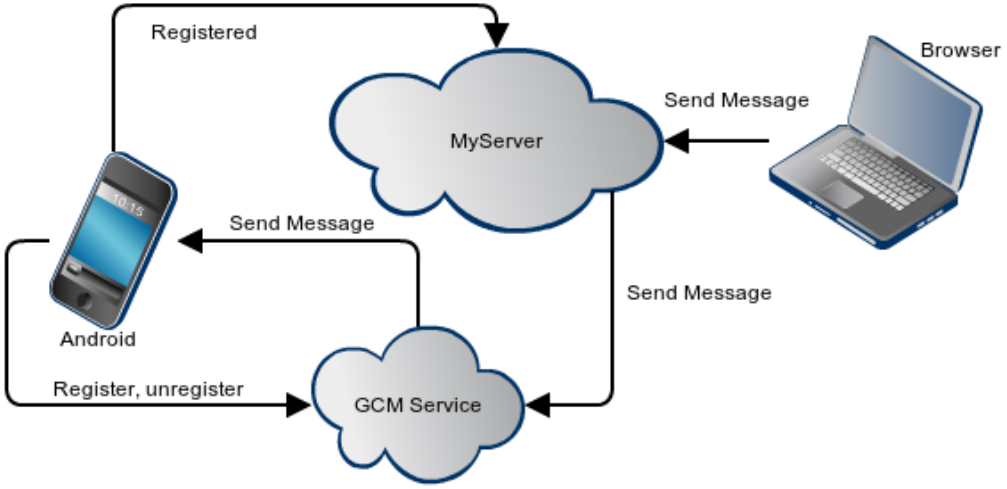
- Slanje i prijem poruka integrisani u konkretnu aplikaciju, klijentski i serverski deo
- Može se odvijati preko:
  - Servera (XMPP, MQTT, COAP, AMQP,…), ili
  - Direktno aplikacija sa aplikacijom korišćenjem nekog oblika *peer-to-peer , ad-hoc* (MANET, VANET) bežične mreže i protokola (*Bluetooth*, *WiFi Direct/Aware*).
- *Developer* može izabrati protokol, kompresione tehnike i sigurnosne mehanizme za komunikaciju mobilnog klijenta sa *messaging* serverom
- Preko *messaging* servera, serverska komponenta aplikacije može poslati poruku mobilnoj aplikaciji da ažurira lokalne podatke – sinhronizacija korišćenjem *push* mehanizma u obliku serverski inicirane sinhronizacije

Push Notification Service

- FCM (GCM) za Android, APNS za iOS, MPNS za Windows
- Veličina poruke koja može biti poslata uređaju se razlikuje:
  - 4KB - Android, 256 B - iOS, 4096 B- Windows Phone
- Broj poruka koje mogu biti poslate jednom mobilnom uređaju
  - 200,000 za GCM, 500 za MPNS bez autentifikacije
- Održavanje specifične infrastrukture za slanje poruka (npr. aplikacioni server, mobilni klijent) koji se zasnivaju na karakteristikama mobilne platforme (npr. Broadcast receivers for Android, URI-channels for Windows Phone, itd)
- Zasnovane na cloud tehnologijama odgovarajuće kompanije (npr. autentifikacioni mehanizam, komunikacioni protokoli, itd).

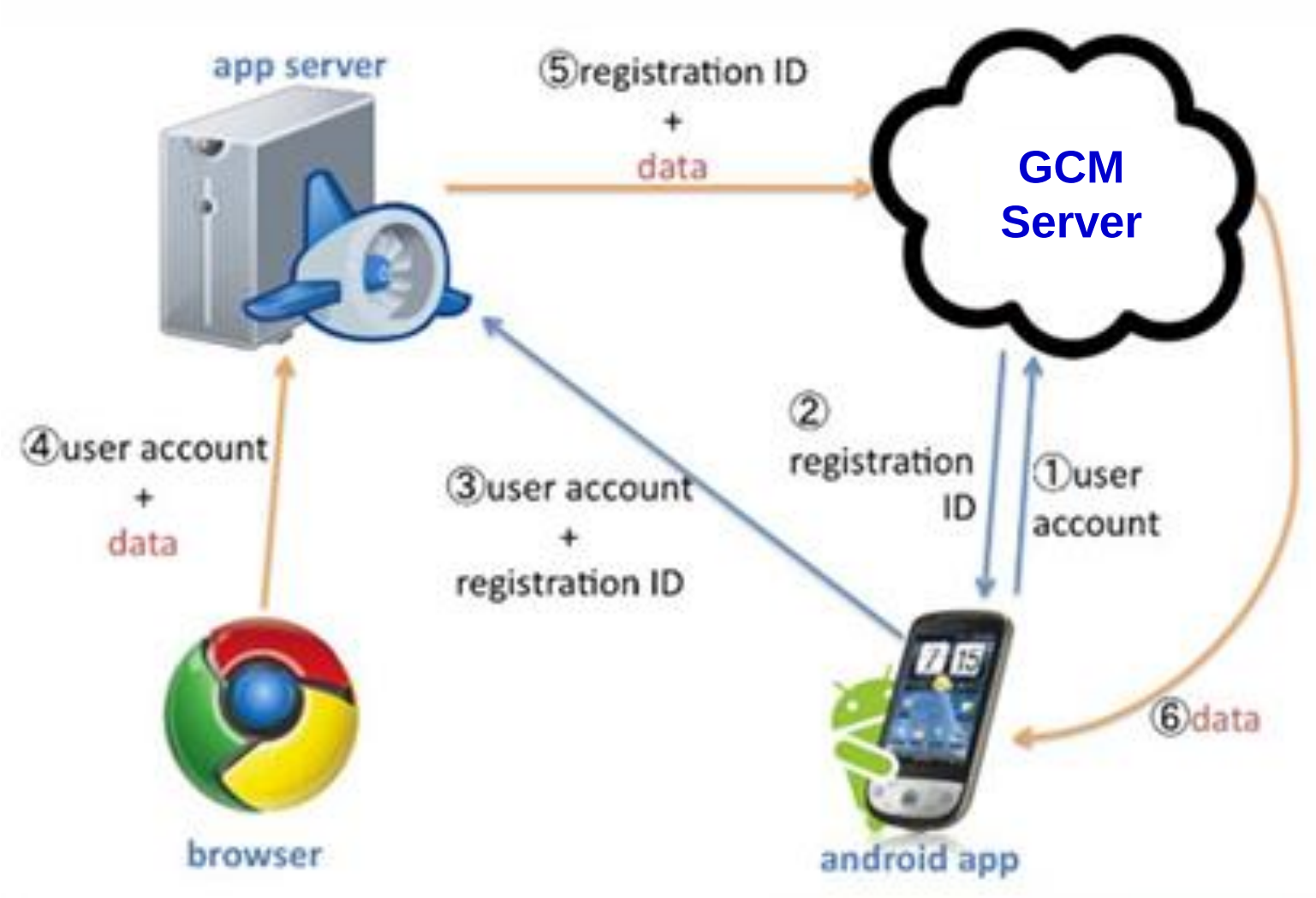
Google Cloud Messaging for Android

- Google Cloud Messaging for Android (GCM) je servis koji omogućava slanje podataka sa našeg servera na Android mobilni uređaj, a takođe i primanje poruka sa uređaja preko iste konekcije.
  - http://developer.android.com/google/gcm/index.html
- GCM servis upravlja svim aspektima formiranja i održavanja redova poruka (*message queues)* i njihove isporuke ciljnoj Android aplikaciji koja se izvršava na mobilnom uređaju.



Mobilni servisi, razmena poruka i notifikacija

GCM – princip rada



1 – Manifest aplikacije

```
<manifest package="com.example.gcm" ...>

    <uses-sdk android:minSdkVersion="8" android:targetSdkVersion="17"/>
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.GET_ACCOUNTS" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE" />

    <permission android:name="com.example.gcm.permission.C2D_MESSAGE"
        android:protectionLevel="signature" />
    <uses-permission android:name="com.example.gcm.permission.C2D_MESSAGE" />

    <application ...>
        <receiver
            android:name=".MyBroadcastReceiver"
            android:permission="com.google.android.c2dm.permission.SEND" >
            <intent-filter>
                <action android:name="com.google.android.c2dm.intent.RECEIVE" />
                <category android:name="com.example.gcm" />
            </intent-filter>
        </receiver>
        <service android:name=".MyIntentService" />
    </application>

</manifest>
```

2 – Registrovanje na GCM

- Android aplikacija koja se izvršava na mobilnom uređaju registruje se u cilju primanja notifikacionih poruka pozivanjem metoda *register(senderID...)*, servera *GoogleCloudMessaging*
- Ovaj metod registruje aplikaciju na GCM i vraća registracioni ID.

3 – Razvoj mobilne i server aplikacije

- Za server poruka može da se koristi novi GCM *Cloud Connection Server* (CCS), stari *GCM HTTP server*, ili oba istovremeno.
- Radi razvoja klijentske aplikacije može da se koristi nešto od sledećeg:
  - Pomoćne biblioteke koje su opisane u Demo App Tutorial i Using the GCM Helper Libraries.
  - Pristup opisan u GCM Architectural Overview.
  - Takođe može se koristiti GoogleCloudMessaging APIs posebno ukoliko se izvršava *upstream* (*device-to-cloud*) slanje poruka (*messaging*).

- Authentication
- Realtime Database
- Storage
- Cloud Messaging
- Hosting
- Cloud functions
- Dynamic links
- Analytics
- Indexing
- AdWords
- Notifications
- Prediction
- Test Lab
- Remote config

Firebase

Build better apps

Improve app quality

Grow your business

The Lifecycle of an App

Build better apps

Cloud Firestore
Store and sync app data at global scale

ML Kit BETA
Machine learning for mobile developers

Cloud Functions
Run mobile backend code without managing servers

Authentication
Authenticate users simply and securely

Hosting
Deliver web app assets with speed and security

Cloud Storage
Store and serve files at Google scale

Realtime Database
Store and sync app data in milliseconds

Improve app quality

Crashlytics
Prioritize and fix issues with powerful, realtime crash reporting

Performance Monitoring
Gain insight into your app's performance

Test Lab
Test your app on devices hosted by Google

Grow your business

In-App Messaging
Engage active app users with contextual messages

Google Analytics
Get free and unlimited app analytics

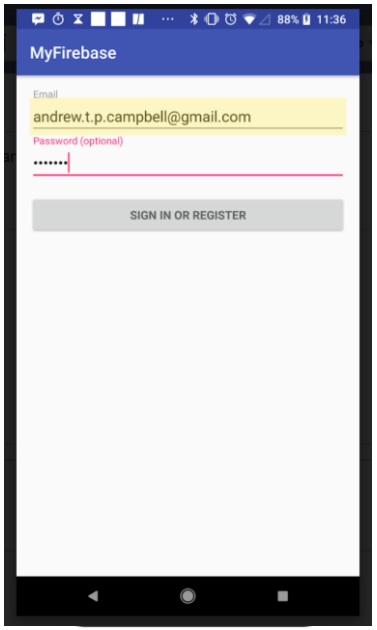
Predictions
Smart user segmentation based on predicted behavior

A/B Testing BETA
Optimize your app experience through experimentation

Cloud Messaging
Send targeted messages and notifications

Remote Config
Modify your app without deploying a new version

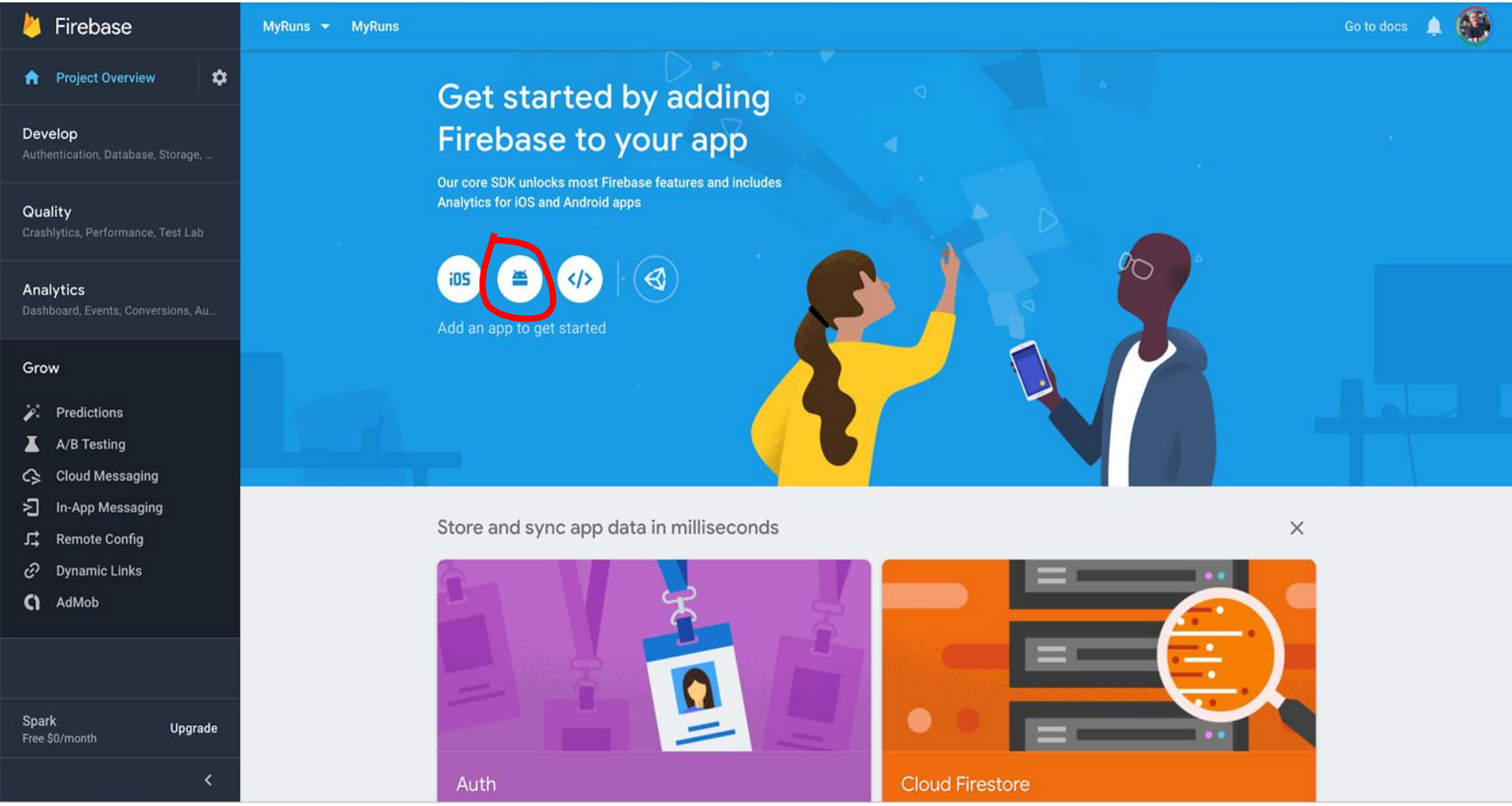
Dynamic Links
Drive growth by using deep links with attribution



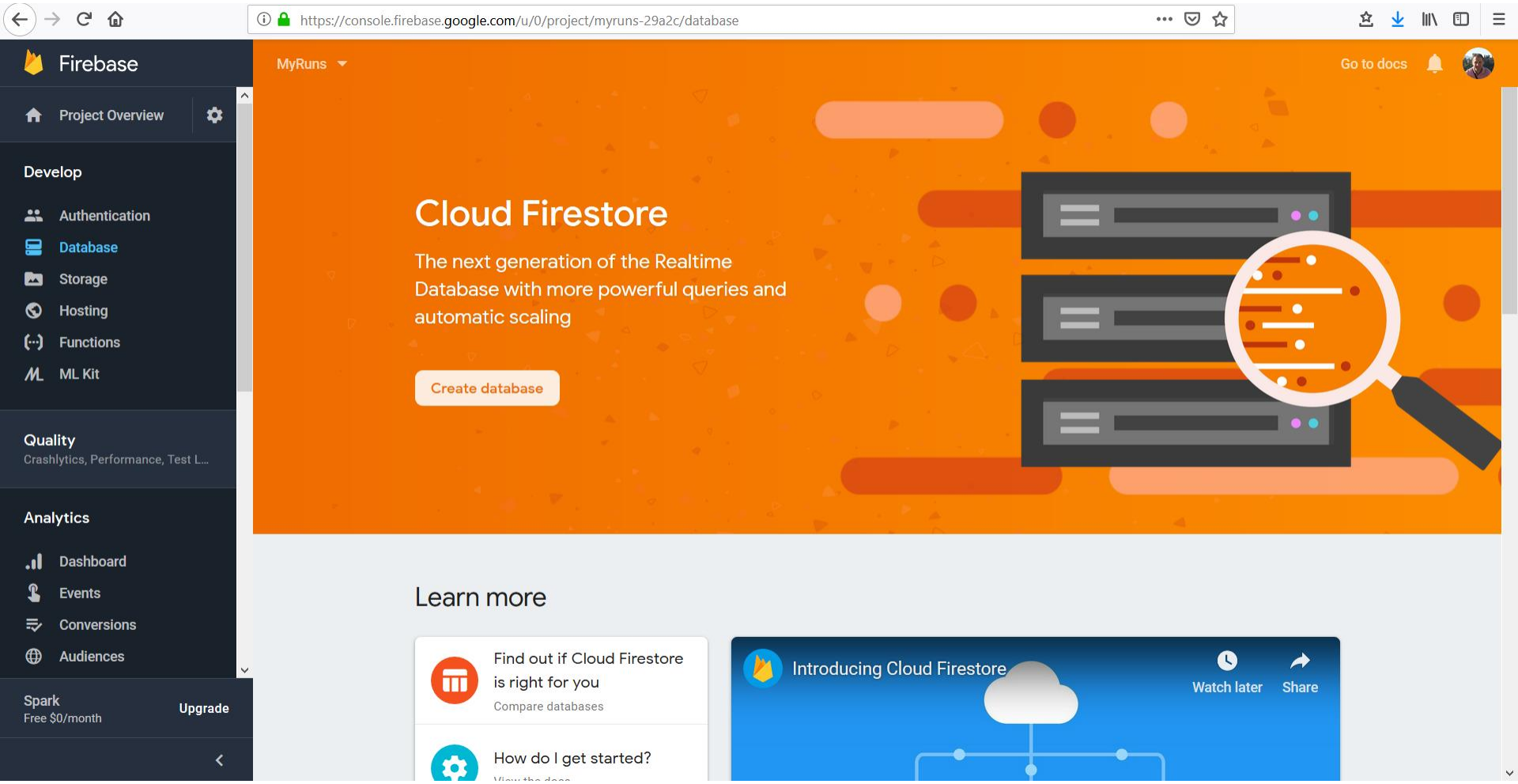
Authentication

Realtime Database/
Cloud Firestore

Firebase console

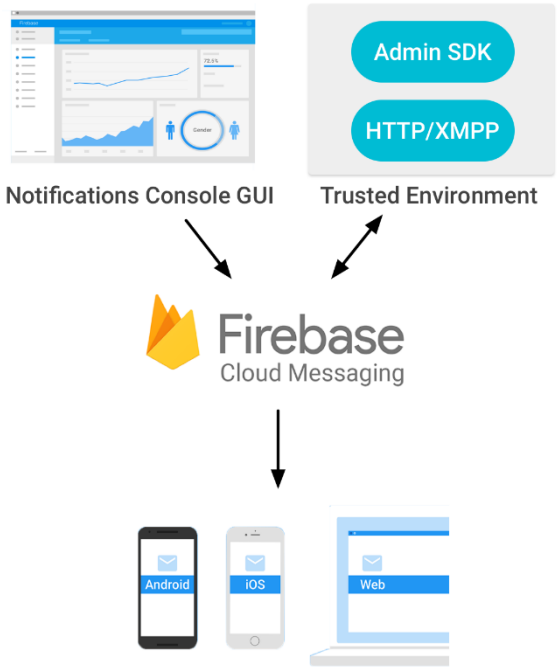


Firebase console



Firebase Cloud Messaging

- *Firebase Cloud Messaging* (FCM) je kros-platformsko rešenje za pouzdanu razmenu poruka, pri tome i besplatno (GCM je *deprecated* od aprila 2018, a biće u upotrebi do aprila 2019)
  - https://firebase.google.com/docs/cloud-messaging/



ML Kit for Firebase

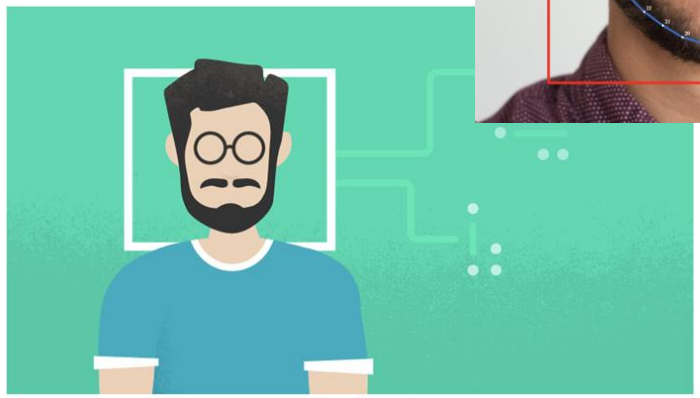
Firebase > Docs > Guides

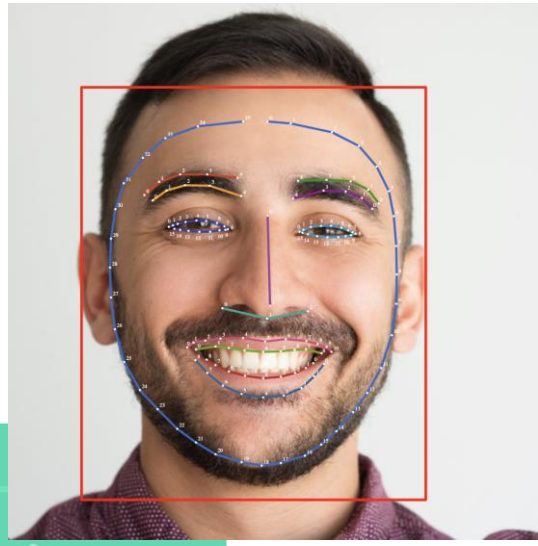
Face Detection

With ML Kit's face detection API, you can detect faces in an image, identify key facial features, and get the contours of detected faces.

With face detection, you can get the information you need to perform tasks like embellishing selfies and portraits, or generating avatars from a user's photo. Because ML Kit can perform face detection in real time, you can use it in applications like video chat or games that respond to the player's expressions.

iOS Android





Base APIs: seamlessly build machine learning into your apps

Image labeling

Identify objects, locations, activities, animal species, products, and more

Text recognition (OCR)

Recognize and extract text from images

Face detection

Detect faces and facial landmarks

Barcode scanning

Scan and process barcodes

Landmark detection

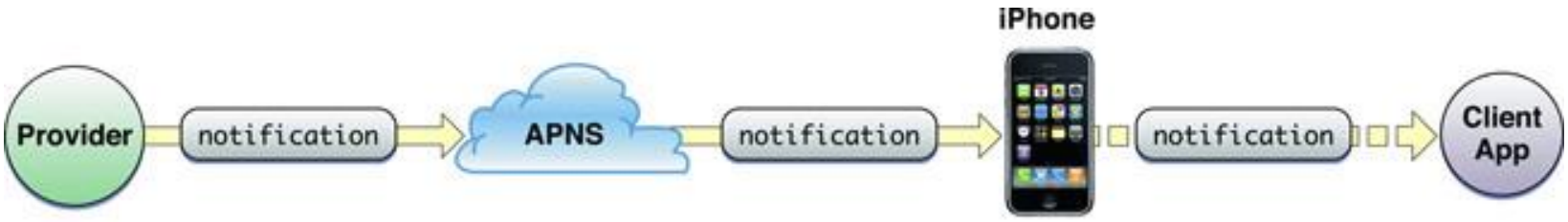
Identify popular landmarks in an image

Smart reply (coming soon)

Provide suggested text snippet that fits context

Apple Push Notification Service

- Apple Push Notification service (APNS) je efikasan servis za slanje podataka u formi *push* notifikacija iOS i OS X uređajima.
  - https://developer.apple.com/library/content/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/APNSOverview.html
  - Svaki uređaj uspostavlja akreditovanu i enkriptovanu IP konekciju sa servisom i prima notifikacije preko te perzistentne konekcije.
- Ako stigne notifikacija za aplikaciju koja trenutno nije aktivna, mobilni uređaj šalje upozorenje korisniku da postoje podaci koji su upućeni toj aplikaciji.



Apple Push notifikacije

- Da bi se napravila push notifikacija za aplikaciju potrebno je uraditi sledeće:
  1. Aplikacija mora da omogući tj. podržava push notifikaciju. Korisnik mora da potvrdi da želi da prima notifikacije za tu aplikaciju.
  2. Aplikacija prima "device token".
  3. Aplikacija šalje *device token* serveru.
  4. Kada se desi dogadjaj od interesa za za aplikaciju, server šalje push notifikaciju Apple Push Notification Service-u (APNS).
  5. APNS prosledjuje push notifikaciju korisnikovom uredjaju.
- Kada korisnikov uredjaj primi notifikaciju, može prikazati upozorenje, emitovati neki zvuk, itd. Korisnik može pokrenuti aplikaciju sa upozorenja, tada se aplikaciji prosledjuje sadržaj push notifikacije i ona može da je obradi na odgovarajući način.

Neophodno za *push* notifikacije

- Da bi dodali push notifikacije aplikaciji, potrebni su:
  - iPhone ili iPad, jer notifikacije ne rade na simulatoru, pa je potreban stvaran uređaj
  - Članstvo u iOS Developer programu, jer za svaku aplikaciju, koja će koristiti push notifikaciju, treba napraviti novi App ID i profil, kao i SSL sertifikat za server.
  - Server koji je povezan na Internet, koji je u stvari zadužen za slanje notifikacija. Na serveru je potrebno pokrenuti background process, instalirati SSL sertifikat i kreirati odlazeće TSL konekcije na odredjenim portovima.

Izgled push notifikacije

- Push notifikacija je kratka poruka koja se sastoji od device tokena, sadržaja tj. *payloada*. *Payload* sadrži podatke koji će biti poslati i koji su od interesa mobilnoj aplikaciji.
- Server treba da *payload* šalje kao JSON dictionary, koji, na primer izgleda ovako:

```
{
    "aps":
    {
        "alert": "Hello, world!",
        "sound": "default
    }
}
```

Kada se ovakva push notifikacija primi:

- prikazaće se alert view sa tekstom "Hello, world!" i
- oglasiće se standardni zvučni signal.

Aktiviranje push notifikacija

- Da bi kreirali aplikaciju koja može da prima push notifikacije, prilikom kreiranja aplikacije u **Xcode** okruženju potrebno je otvoriti **<ime_aplikacije>AppDelegate.m** i izmeniti ***didFinishLaunchingWithOptions*** metodu tako da izgleda ovako:

```
(BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    self.window.rootViewController = self.viewController;
    [self.window makeKeyAndVisible];
    // Let the device know we want to receive push notifications
    [[UIApplication sharedApplication]
    registerForRemoteNotificationTypes:
    (UIRemoteNotificationTypeBadge | UIRemoteNotificationTypeSound |
    UIRemoteNotificationTypeAlert)];

     return YES;
}
```

Aktiviranje push notifikacija

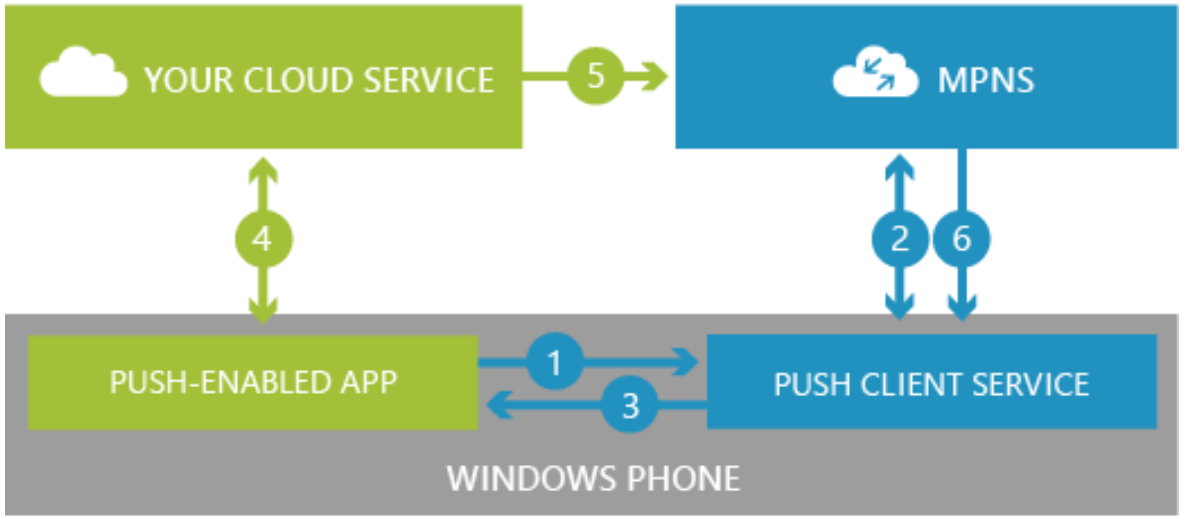
- Poziv ***registerForRemoteNotificationTypes*** označava da aplikacija želi da prima push notifikacije. Kada se ova aplikacija startuje i registruje za push notifikacije, prikazaće poruku kojom obaveštava korisnika da želi da mu prosleđuje push notifikacije.
- Da bi omogućili ovoj aplikaciji da prima push notifikacije potrebno je u **<ime_aplikacije>AppDelegate.m** dodati sledeće linije koda:

```
(void)application:(UIApplication*)application
didRegisterForRemoteNotificationsWithDeviceToken:(NSData*)deviceTo
ken
{
     NSLog(@"My token is: %@", deviceToken);
}

(void)application:(UIApplication*)application
didFailToRegisterForRemoteNotificationsWithError:(NSError*)error
 {
     NSLog(@"Failed to get token, error: %@", error);
}
```

Microsoft *Push Notification Service* - Windows Phone

- *Microsoft Push Notification Service* za Windows Phone je asinhroni servis koji obezbeđuje third-party developer-ima mogućnost slanja podataka Windows Phone aplikaciji preko *Microsoft Azure* cloud servisa na efikasan način.
- *Push notifications* obezbeđuje način da Web servis/server, desktop PC-a, ili drugi Windows Phone uređaj pošalje podatke klijentu po *push* principu bez potrebe da klijent izda web zahtev.



MPNS – princip rada

1. Mobilna aplikacija zahteva *push notification URI* od strane *Push client service*.
2. *Push client service* kontaktira *Microsoft Push Notification Service* (MPNS), i MPNS vraća *notification URI* za *Push client service*.
3. *Push client service* vraća *notification URI* *v*ašoj mobilnoj aplikaciji.
4. Vaša aplikacija tada šalje *notification URI odgovarajućem* cloud servisu .
5. Kada cloud servis ima podatke/informacije koje treba asinhrno da pošalje vašoj aplikaciji, koristi *notification URI* da pošalje *push* notifikaciju MPNS - u.
6. MPNS preusmerava i šalje *push* notifikaciju mobilnoj aplikaciji.

Mehanizam komunikacije

- Aplikacija može imati samo jedan komunikacioni kanal.
- Komunikacioni kanal se vezuje za jedan tip notifikacija. Ovo je moguće menjati u toku rada aplikacije.
- U zavisnosti od tipa notifikacija za koji je vezan, komunikacioni kanal može ostati otvoren i nakon gašenja aplikacije koja ga je otvorila.

Tipovi *push* notifikacija

- Tri tipa:
  - *Toast* notifikacije.
  - *Tile* notifikacije.
  - *Raw data* notifikacije.
- Osnovne razlike leže u načinu prikaza primljenih notifikacija i u načinu njihovog primanja od strane aplikacije.

Pitanja i komentari