Gestor de notificacions híbrid per a múltiples aplicacions.

Aleix Casanovas Pratdesaba, *1292776, UAB*

Abstract—Els dispositius mòbils i les aplicacions formen part de la vida diària de les persones. Actualment, hi han una gran varietat de dispositius i aplicacions que permeten rebre notificacions push en qualsevol moment. L'enviament d'aquestes notificacions pot arribar a ser un mal de cap per una persona no tècnica que ha de fer front a les aplicacions web dels proveïdors de notificacions push. Per tractar aquesta problemàtica, aquest treball presenta una solució per enviar notificacions push duna forma àgil i senzilla. A més, la solucó permet enviar aquestes notificacions push a multi-dispositius i multi-aplicacions a partir d'una API transparent i una aplicació web per poder gestionar i enviar cada una de les notifiacions.

Paraules clau—Notificació Push, FCE, Android, iOS, aplicacions mòbils.

Abstract—Mobile devices and applications are part of people's daily lives. Currently, there are a great heterogeneity of devices and applications that need to be able to receive push notifications in every moment. Send this notifications could be a nightmare for individuals who have to deal with the notification providers. To address this issue, the present work presents a solution for sending push notifications. Furthermore, this solution will work with multi-devices and multi-applications, for that an API is developed together with a website for handle and send every single notification.

Index Terms—Push notification, FCE, Android, iOS, mobile applications.

1 Introducció

Tenint en compte l'època en què vivim, no és d'extranyar que cada dia apareguin noves aplicacions mòbils, que usen tecnologies i termes diferents de les que veníem coneixent. Un d'aquests termes són les notificacions push (tecnologia de tramesa automàtica) [1]. Avui en dia tothom les utilitza i les veu a diari però no tothom sap en què consisteixen. La tecnologia push és una forma de comunicació en la que una aplicació servidor envia un missatge a un client-consumidor. És a dir, és un missatge que un servidor envia a una persona avisant-lo de que té una nova informació. La característica principal d'aquesta tecnologia és que sempre és el servidor qui inicia aquesta comunicació, un altre aspecte clau és la immediatesa d'aquesta tecnologia, donat que no fa falta estar executant l'aplicació client perquè arribi la notificació.

Actualment podem trobar gran quantitat d'aplicacions mòbils al mercat, moltes empreses tenen més d'una aplicació, en alguns casos desenes. Per aquest motiu el fet d'enviar les notificacions directament pel proveïdor de notificacions és una gestió costosa i difícil per una persona no tècnica, ja que en cas d'enviar una notificació a varies aplicacions haurà de canviar de contexte en cada aplicació. El projecte que s'ha desenvolupat pretèn resoldre aquesta problemàtica tot centralitzant l'enviament i gestió de notificacions en un punt.

L'objectiu principal d'aquest treball ha estat el de vestir un projecte per una empresa de notícies que té varies aplicacions mobils segons les seves seccions editorials. En aquest cas, dues aplicacions:

- *E-mail: aleix.casanovas@e-campus.uab.cat*
- *Menció: Tecnologies de la informació.*
- *Tutor: Jordi Duran Cals. Professor associat al DEiC.*

Febrer de 2017. Escola d'Enginyeria (UAB)

- La primera aplicació mòbil desenvolupada permetrà rebre les alertes meteorològiques segons les províncies de Catalunya en què l'usuari estigui subscrit.
- En la segona aplicació mòbil es rebran les alertes esportives segons les provincies de Catalunya en què l'usuari estigui subscrit.

Per tant, en la solució hi haurà un gestor web on els periodistes encarregats d'enviar les notificacions podran accedir i publicar una nova alerta meteorològica o esportiva (triant l'aplicació on es vol enviar) i associar-la a una província de Catalunya, aquesta alerta serà rebuda pels usuaris en format de notificació push en els dispositius que tinguin installada l'aplicació i estiguin subscrits a la província.

1.1 Estructura de l'article

L'article es troba estructurat amb 7 seccions. Primerament s'exposa el projecte en el bloc d'introducció on també es troba descrit els objectius i l'estat de l'art del projecte. Seguidament en el punt 2 hi han exposats els requisits sobre els quals s'han treballat. En el punt 3 es troba la meteodologia utilitzada i com s'ha planificat el projecte. L'apartat 4 detalla ja l'aplicació i les diferents parts: arquitectura, API, base de dades, gestor web i aplicació mòbil. Tot seguit, en el punt 5 la descripció més purament tècnica del projecte amb el llistat dels mòduls més rellevants i l'apartat referent a la seguretat. En el punt 6 es troben els resultats que s'han obtingut a més d'una explicació sobre la posibles fallades que hi poden haver i com es resoldran i una secció sobre com dimensionar la solució. Per acabar en el punt 7 es troben les conclusions del projecte.

1.2 Objectius

A continuació es llisten els objectius del projecte en la taula 1:

TABLE 1: Llistat d'objectius del projecte

Objectius	Prioritat
Analitzar l'estat de l'art les diferents eines i serveis d'enviament de notificacions push	Prioritari
Desenvolupar un gestor de notificacions web per donar servei a diverses aplicacions	Prioritari
Desenvolupar una API que gestionarà les diferents notificacions i serà l'encarregada d'interoperar amb el servei d'enviament de missatges push escollit per fer els enviaments	Prioritari
Desenvolupar diverses aplicacions mòbils per poder rebre les corresponents notificacions	Prioritari
Realitzar el projecte en el temps especificat	Crític
Sistemes de seguretat en el gestor	Secundari

1.3 Estat de l'art

Conèixer l'estat de l'art de les tecnologies d'enviament de notificacions push és vital alhora d'abordar aquest projecte. Així doncs, en el moment de fer la tria del sistema d'enviament ens hem centrat en els tres principals sistemes de notificacions híbrids (que funcionen tant amb Android com iOS):

- Amazon SNS [6]
- Google FCM/GCM [7]
- Azure NH [8]

Entrant més en detall aquests sistemes funcionen seguint el següent esquema (Fig. 1):

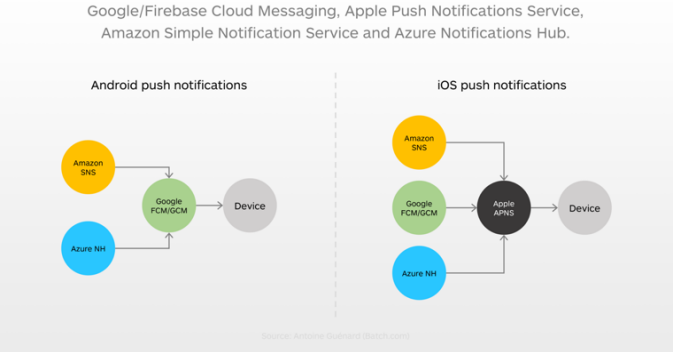

Fig. 1: Sistemes de notificació push

Aquest esquema ha estat clau alhora de decidir quin proveïdor fer servir. En el cas de les notificacions d'Android és directament el servei Google FCM/GCM qui envia les notificacions i en dispositius iOS el Apple APNS. El servei d'Apple APNS directament s'ha descartat al no tenir suport en dispositius Android. Per tant, els únics proveïdors interessants pel projecte serien: Amazon SNS, Azure NH i Google FCM/GCM sent els dos primers en el cas d'Android un servei més per sobre que s'acaba comunicant amb el Google FCM/GCM. Per tant, tant per Amazon SNS com Azure NH al no aportar cap avantatge significatiu als objectius del treball realitzat se'ls ha descartat. Un altre fet diferencial és que tant Amazon SNS com Azure NH són serveis de pagament, mentre que en Google FCM/GCM el servei de notificacions és totalment gratuït [9], a més Google FCM/GCM proporciona altres característiques tant gratuïtes com de pagament que no són utilitzades en aquest projecte.

El projecte desenvolupat a part de fer ús de les tecnologies d'enviament de notificacions push, també emmagetzema les notificacions i permet fer un enviament àgil i senzill de cada notificació a múltiples aplicacions. Amb aquest mateix propòsit es troben gran quantitat de projectes i empreses, com poden ser Kumulos, Carnival, Urban Airship o Push Woosh. Aquestes quatre empreses ofereixen diferents funcionalitats, per exemple, Kumulos i Carnival permeten enviar notificacions push de forma automàtica en funció del comportament de l'usuari, a més de gran quantitat d'analítiques. Pel que fa Urban Airship es centra més en l'optimització afegint filtres de localització, plataforma i preferencies per tal d'enviar notificacions a un segment d'usuaris més concret. Totes elles però són de pagament i tenen la seva pròpia infraestructura, de forma que en cas de utilitzar-les es depén del seu servei i disponibilitat.

2 Requisits

Els requisits funcionals i no funcionals que s'han establert pel present treball són:

2.1 Requisits funcionals

- R1: Un periodista ha de poder accedir al sistema mitjançant un login i una paraula de pas.
- R2: Un periodista ha de poder veure un històric de les notificacions enviades.
- R3: Com ha director de l'empresa el sistema m'ha de permetre afegir noves aplicacions i províncies (tòpics).
- R4: Els usuaris de l'aplicació mòbil han de rebre les notificacions de les províncies que està subscrit.
- R5: Els usuaris de l'aplicació mòbil han de poder subscriure's i desubscriure's de les provincíes.

2.2 Requisits no funcionals

- R6: Les interfícies d'usuari del gestor i l'aplicacitó han de ser intuitius.
- R7: L'usuari ha de rebre les notificacions de forma instantània.
- R8: Els periodistes han de poder navegar pel gestor de forma ràpida i sense d'errors.
- R9: El sistema ha de ser segur, cap persona àliena ha de poder enviar una notificació.

3 Meteodologia

Per tal d'organitzar les diferents tasques i desenvolupaments a realitzar s'ha fet servir la metodologia SCRUM. Com que només hi han dues persones en el projecte (el projectista i el tutor), els rols es reperteixen de la següent forma:

- El tutor del projecte ocupa el rol de Product owner i Customer, indicant els requeriments que vol i prioritzant les tasques del backlog.
- El projectista té el rol de Scrum Master i Scrum Team, de forma que organitza les tasques en el Kanban i produeix els entregables del projecte. Per fer l'organització de les tasques s'utilitza el mètode Kanban, més concretament el programari Trello que té la següent representació (Fig. 2):

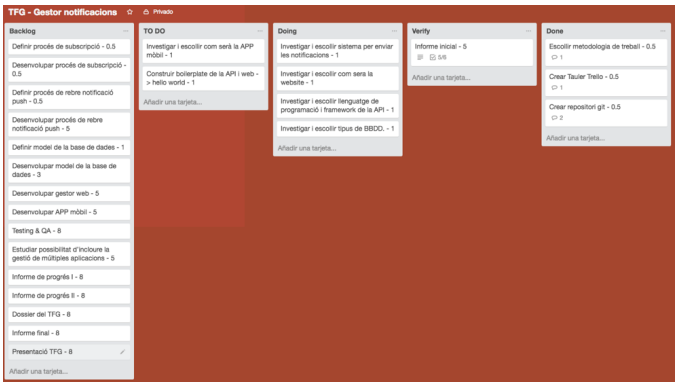

Fig. 2: Kanban board

La gestió i control del codi font es troba dispositat en un repositori Git [4], concretament Bitbucket [5] de l'empresa Atlassian. Aquest programari online, permet crear respositoris gratuïts per equips de treball petits (fins a 5 usuaris) i té un ampli reconeixement a nivell global. Així doncs, gràcies a aquest programari s'han fent commits a mesura que es va desenvolupant les històries del sprint.

3.1 Planificació

Les regles de SCRUM són la següents:

- Sprints de 2 setmanes.
- Una tasca no pot tenir un esforç superior a dos setmanes.
- L'esforç serà avaluat a partir de números de la sèrie Fibonacci, a més del 0.5 (Planning poker [10]). On la història més gran com a molt pot tenir 13 punts d'esforç. Les valoracions seran fetes en base a la experiència de 4 anys com a enginyer informàtic.

El backlog està compost per les següents històries, i s'han anat introduint a cada començament d'sprint. Juntament amb cada història hi ha l'esforç associat. La planificació aproximada per sprints és la següent:

3.1.1 SPRINT I (16/09 - 30/09)

- Informe inicial. - 5
- Crear repositori git. - 0.5
- Investigar i escollir com serà la APP. - 1
- Construir el projecte base de la API a partir del qual es comencarà a desenvolupar la web fins a fer un "hola món". - 1
- Investigar i escollir sistema per enviar les notificacions. - 1
- Investigar i escollir com serà el gestor website. - 1
- Investigar i escolir llenguatge de programació i la infraestructura digital de la API. - 1
- Investigar i escollir tipus de BBDD. - 1
- Escollir metodologia de treball. - 0.5
- Crear Tauler trello - 0.5

3.1.2 SPRINT II (02/10 - 14/10)

- Definir procés de subscripció. - 0.5
- Definir procés de rebre notificació push. - 0.5
- Desenvolupar procés de subscripció. - 5
- Definir model de la base de dades. - 1

3.1.3 SPRINT III (17/10 - 27/10)

- Desenvolupar procés de rebre notificació push. - 5
- Desenvolupar model de la base de dades. - 3

3.1.4 SPRINT IV (31/10 - 11/11)

- Informe de progrés I. - 8

3.1.5 SPRINT V (14/10 - 25/11)

- Desenvolupar gestor web. - 5

3.1.6 SPRINT VI (28/10 - 09/12)

- Desenvolupar APP mòbil. - 5

3.1.7 SPRINT VII (12/11 - 23/12)

- Testing & QA - 8
- Informe de progrés II. - 8

3.1.8 SPRINT VIII (02/01 - 12/01)

- Estudiar possibilitat d'incloure la gestió de múltiples aplicacions. - 5

3.1.9 SPRINT IX (02/01 - 13/01)

- Informe final. - 8
- Dossier del TFG. - 8

3.1.10 SPRINT X (16/01 - 26/01)

- Presentació TFG. - 8

3.2 Gràfic "Burn up"

Es pot veure l'evolució de l'esforç completat a mesura que s'assolien els diferents "sprints" en el següent gràfic (Fig. 3):

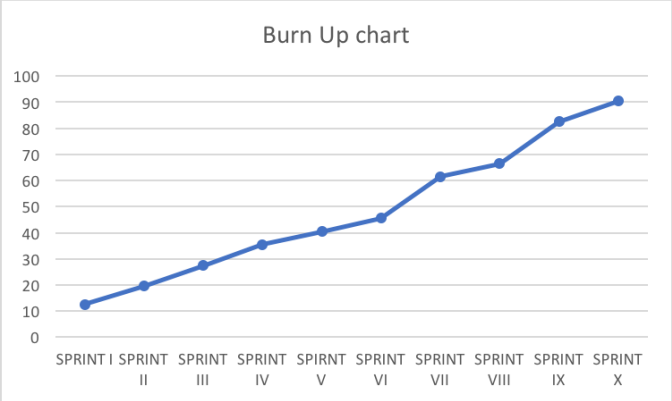

Fig. 3: "Burn up chart"

4 APLICACIÓ

4.1 Arquitectura

Per definició, perquè el servidor enviï el missatge al client, aquest s'ha hagut de subscriure prèviament. Per tant, es capturarà el Registrarion ID d'Android[3] o el Device Token d'iOS[4] i es guardarà per poder identificar el dispositiu i enviar-li la notificació push. és important distingir entre la tecnologia pull de la push. La diferencia cau en qui inicia la comunicació. En la tecnologia pull és el client que l'inicia mentre que en la push és el servidor. Un possible flux per poder rebre una notificació push seria el següent (Fig. 4):

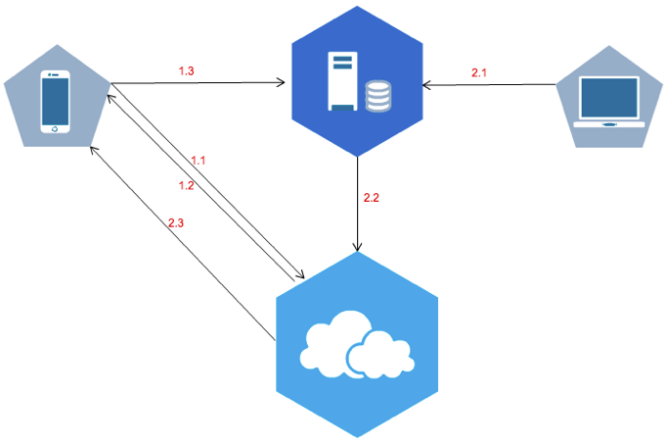

Fig. 4: Flux notificació push

(1) Procés de subscripció:

- 1.1 La aplicació mòbil envia una petició al sistema d'enviament de missatges push amb el Registration ID d'Android o el Device Token Id d'iOS.
- 1.2 El sistema d'enviament de missatges push (per exemple: FCM, APNs, Amazon SNS, etc.) respon amb el token identificatiu del dispositiu.
- 1.3 Aquest token és enviat a la API juntament amb les dades d'identificació de l'aplicació i dispositiu i del tòpic on es vol subscriure. Finalment, l'API guardarà la relació de: aplicació-dispositiu-token-tòpic a la base de dades.

(2) Procés d'enviament d'una notificació push:

- 2.1 S'introdueix una nova notificació a partir d'una pàgina web, que fa la pertinent crida a la API.
- 2.2 L'API processa aquesta petició i li indica al sistema d'enviament de missatges push a quins tokens (dispositius) o a quin tòpic ha d'enviar la notificació.
- 2.3 El sistema d'enviament de missatges push envia la notificació a cada dispositiu o tòpic que l'API li ha ordenat.

4.2 Estructura de la base de dades

S'ha desenvolupat una base de dades relacional MySQL per gestionar els usuaris del gestor web i per guardar les relacions de les aplicacions subscrites a cada una de les províncies (que estan descrites en el model com a tòpics) i els seus tokens corresponents. Més en detall es pot veure l'estructura de la base de dades en la imatge (Fig. 5):

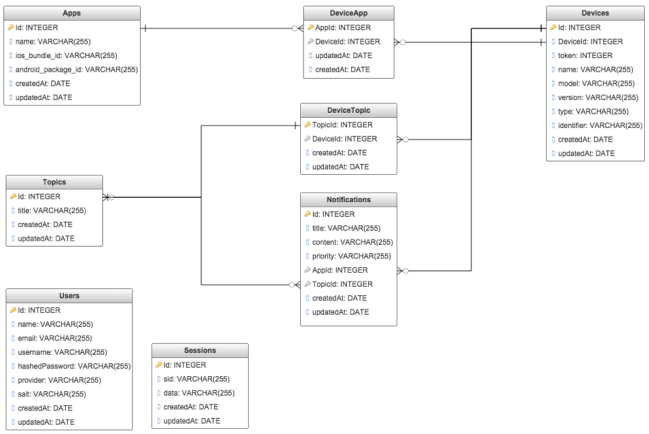

Fig. 5: Esctructura de la base de dades relacional

4.3 API

En la comunicació entre el gestor web i el sistema d'enviament de notificacions push s'ha desenvolupat una API rest usant el llenguatge Node.js [11] ja que és un llenguatge modern event-driven [12] i asíncron. Fet que ens ajudarà a construir un sistema com el que volem fer. La infraestructura digital usada és express [13] ja que és un dels framworks Node.js més reconeguts i madurs. Totes les funcionalitats de l'API desenvolupada es troben en el següent llistat:

- GET /apps - Llistat de totes les aplicacions registrades al sistema
- GET /apps/:appId - Informació d'una aplicació en concret
- GET /topics - Llistat de tots els topics registrats al sistema
- GET /topics/:topicId - Informació d'un topic en concret
- GET /notifications - Llistat de totes les notificacions creades al sistema
- GET /notifications/:notificationId - Informació d'una notificació en concret
- POST /register - Registre d'un dispositiu
- POST /subscribe - Subscripció d'un dispositiu a un tòpic
- POST /unsubscribe - Desubscripció d'un dispositiu a un tòpic
- GET /users/me - Obtenció de l'informació de l'usuari gestor del web
- POST /users/session - Obtenció d'un token de sessó pel gestor web
- POST /users - Creació d'un usuari pel gestor web.

4.4 Gestor web

El gestor web s'ha desenvolupat usant l'infraestructura digital AngularJs desenvolupada amb Javascript. Més en detall s'han realitzat les següents pantalles on els periodistes podran accedir per tal de gestionar i crear tant aplicacions, províncies (tòpics) i notificacions. Més en concret el llistat de pantalles realitzades és el següent:

- Homepage amb les opcions per accedir al gestor
- Pàgina de login per accedir al gestor
- Pàgina de registre per crear un usuari
- Llistat per veure l'històric de notificacions
- Pàgina per crear una notificació
- Pàgina detall d'una notificació
- Llistat per veure tots els tòpics introduits al sistema
- Pàgina per crear un tòpic
- Pàgina detall d'un tòpic
- Llistat per veure totes les aplicacions introduïdes al sistema
- Pàgina per crear una aplicació
- Pàgina detall d'una aplicació

4.5 Aplicació mòbil

Donat que es vol crear una aplicació senzilla que solament rebi les notificacions s'ha obtat per crear una aplicació hibrida amb cordova [14]. Més concretament, a nivell de tecnologies s'ha usat ionic2, aquest es un framework per crear aplicacions híbirdes a partir d'angular2 i cordova.

Per simplificar l'aplicació s'ha creat solament una pantalla per permetre a l'usuari subscriure's als tòpics desitjats dels quals rebrà les notificacions. Com que un dels requisits és fer un sistema multi plataforma s'han creat dues aplicacions, una on és rebran notificacions esportives (Fig. 7) i una altre on es rebran les alertes meteorològiques (Fig. 6).

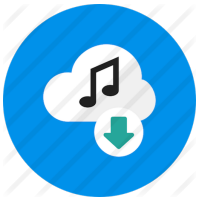

Fig. 6: Icona de l'aplicació d'alertes Meteorològiques

Fig. 7: Icona de l'aplicació d'alertes esportives

5 Descripció tècnica

5.1 Mòduls i llibreries

Per desenvolupar cada part del projecte s'ha fet ús d'una gran quantitat de tecnologies agrupades en tres grans blocs (API, Gestor web i Aplicació mòbil). Per aquest motiu s'han hagut de prendre moltes decisions tècniques, les més rellevants es troben justificades en els següents apartats:

5.1.1 API

- Expressjs [13]: Infraestructura digital Open-Source per desenvolupar en Nodejs
- FCM-push [15]: Mòdul Javascript que envia les peticions al servidor de notificacions Firebase (FCM).
- Mysql [16]: Tecnologia de base de dades que s'utilitzarà.
- Passport [17]: Infraestructura digital d'autenticació per Nodejs.
- Sequelize [18]: Mapatge d'objectes relacional (ORM) per Nodejs que suporta PostgreSQL, MySQL, MariaDB, SQLite i MSSQL.

5.1.2 Gestor web

- Angular [19]: Infraestructura digital en Javascript per desenvolupar web-apps.
- Bootstrap [20]: Llibreria que permet utilitzar components HTML i CSS.

5.1.3 APP mòbil

- Ionic2 [22]: Infraestructura digital basada amb Angular2 [21] per crear aplicacions de mòbils híbrides.
- Cordova [23]: Utilitat que permet la compilació d'una aplicació web a mòbil, tant per Android, iOS com Windows phone.

5.2 Seguretat

Les seguretat és un aspecte que s'ha tingut molt present en la realització del projecte. Ja que el fet que una suplantació d'identitat al enviar una notificació pot ser un punt molt crític. Així doncs veiem que s'ha tractat amb especial cura la seguretat en els següents punts clau:

5.2.1 Accés al gestor web i API

Per poder accedir al gestor web i per tant, fer ús de diferents peticions de la API és necessàri d'un token de sessió. Aquest token és obtingut a partir del formulari d'inici de sessió on es demana usuari i un mot de pas. En la base de dades no es guarda la contrasenya en clar, sinó que es guarda un "one way" hash amb l'algoritme pbkdf2 (de forma que no es pot saber la contrasenya inicial), a més se li aplica un "salt" de 32bytes de forma que dos usuaris amb la mateixa contrasenya no tindran el mateix hash. Les crides de la API doncs, requereixen obligatoriament d'un token valid per poder ser executades.

5.2.2 Comunicacions al sistema d'enviament de notificacions push

- Comunicació amb la API:
  La comunicació entre el sistema d'enviament de notificacions push i la API es a partir de missatges HTTP (POST) aquests són autenticats a través de la capçalera Authorization. Aquesta "autorization key" és generada pel pròpi sistema d'enviament, en el nostre cas FCM, i per tant, es vital que sigui guardada de forma segura i només sigui transportada a través de comunicacions segures.
- Comunicació amb els dispositius:

Per tal de rebre una notificació els dispositius s'utilitza el sistema estàndard d'Android i iOS. Aquest es basa amb una comunicacio TLS entre el dispositiu i el proveïdor de notificacions push de la plataforma corresponent. Però al moment de rebre una notificació aquesta s'envia en text clar per això és important conèixer que els missatges poden ser interceptats i llegits per tercers i és important no posar informació confidencial. En el nostre cas al ser un sistema pensat per noticies aquest no suposa un problema, tot i així si es volgués seguretat s'hauria de fer servir la notificació push com un sistema de sincronització. De forma que al haver-hi un contingut nou s'envia una notificació push però "sense el contingut" i seria la pròpia aplicació que s'encargaria de sincronitzar-se per rebre el nou contingut d'una altre forma més segura.

6 RESULTATS

6.1 Evidències

Al ser un projecte amb una finalitat molt concreta, la millor forma de demostrar les evidències del desenvolupament fet és mostrar per sobre els diferents punts claus:

- Subscripció per part d'un usuari a una província (tòpic) a través d'un dispositiu:

L'usuari es subscriu a una província a partir de dues crides: una a la API indicant el token i la província, i un altre directament al sistema de notificacions. La pantalla de subscripció a províncies serà doncs com la seguent (Fig. 8):

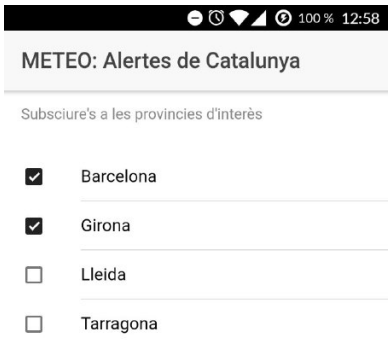

Fig. 8: Pantalla de subscripció a una província (tòpic)

- Enviament d'una notificació a través del gestor per part d'un periodista:

En el gestor es pot crear una notificació indicant l'aplicació on es vol enviar i/o la província (tòpic), com veiem en la següent imatge s'ha triat tant l'aplicació com el tòpic, de forma que només els usuaris que tinguin l'aplicació d'alertes meteorològiques i estiguin subscrits a la província Barcelona rebran la notificació (Fig. 9).

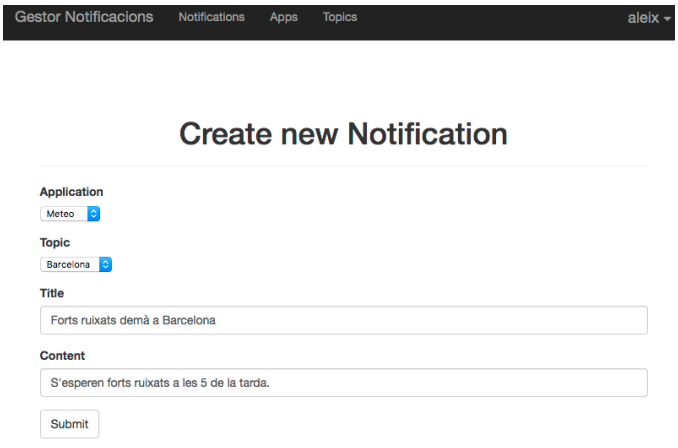

Fig. 9: Pantalla de creació d'una notificació

- L'usuari rep una notificació al dispositiu:

El sistema d'enviaments de notificació enviarà directament la notificació als dispositius, un exemple de notificació es pot veure en la següent imatge (Fig. 10):

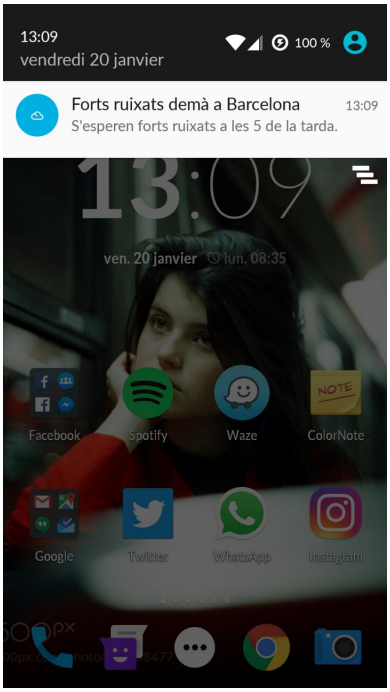

Fig. 10: Recepció de la notificació al dispositiu

6.2 Tolerància a fallades

En cas que el sistema de notificacions push estigui caigut o saturat el gestor i la API continuaran funcionant. Per aconseguir-ho al crear una notificació al sistema primer s'inserta a la base de dades i la columna "send" es posa com a "false", de forma que no fa res més que insertar una fila a la base de dades. I per darrera hi ha un "cron" executant-se que va enviant les notificacions de forma desatesa i un cop enviades les marca amb la columna send a "true".

D'aquesta forma en cas que el sistema d'enviament estigui caigut o saturat les notificacions es podran crear des del gestor i s'aniran apilant a la base de dades com a pendents per enviar i un cop és restebleixi el sistema de notificacions push aquestes aniran sent enviades.

6.3 Infraestructura i dimensionament del HW

Pel que fa el HW que es necessita per prossar en marxa el projecte, es recomana una màquina similar a la de per exemple la instància d'us general "T2.medium" d'amazon per fer correr la API i el Amazon RDS "db.m4.large" per la base de dades. Pel que fa a les aplicacions mobils no requereixen d'infraestructura ja que són distribuïdes pels mercats d'aplicacions d'iOS i Android.

A més, l'arquitectura desenvolupada pot ser fàcilment dimensionada gràcies a la flexibilitat que se l'hi ha donat. Una forma fàcil de dimensionar seria a partir de tecnologia Docker [24], ja que es podria encapsular l'API en contenidors docker i usant el servei de contenidors d'amazon, Elastic compute Cloud (EC2) [25] a partir del qual es podrien dimensionar les instancies d'API segons la càrrega del moment. Tot i així, faria falta introduir un "job scheduler", com podria ser el node-cron [26] per tal d'executar les tasques en background de forma coordinada amb cadascuna de les instàncies que es trobin corrent.

6.4 Llista de millores

L'execució d'aquest treball ha complert amb els objectius marcats. Tot i així sempre es poden aplicar moltes millores, a continuació es troben llistades les més interessants:

- Llistat d'analítiques (nombre d'enviaments, notificacions obertes, etc).
- Usuaris amb rols al gestor, de forma que una persona amb rol de "Periodista esportiu" només pot enviar notificacions a l'aplicació d'esports.
- Habilitar la possibilitat de programar l'enviament d'una notificació a una determinada hora.
- Afegir propietats opcionals a les notificacions tals com: imatge relacionada amb la notificació, icona de la notificació, so de la notificació, etc.

7 Conclusions

El sistema d'enviaments de notificacions push és un sistema que ja esta molt integrat a la nostre vida i que diàriament utilitzem sense donarnos compte. És doncs, un sistema àmpliament

utilitzat i de formes molt diverses. Amb aquest projecte s'ha pogut veure una de les possibles aplicacions basada en poder enviar notificacions push de forma àgil i senzilla. A més, gràcies al sistema desenvolupat s'ha donat molta més flexibilitat al propi sistema de notificacions push triat (FCM), ja que, al fet de tenir una base de dades pròpia amb els dispositius registrats pot fer que, en casos concrets, es vulgui enviar una notificació a només alguns dispositius específics (per exemple tots els registrats en data X). Una mica més en profunditat ens trobem que el sistema desenvolupat pot permetre amb un cost de desenvolupament molt petit altres avantatges i funcionalitats que el pròpi sistema de notificacions (FCM) no ens donava, com el de planificar una notificació, gestionar les prioritats de les notificacions i enviar dades extres com; icona, missatges desplegables, temps de vida de la notificació, array de dades, etc.. Per últim, una de les altres avantatges que aporta la solució és l'històric de notificacions, aplicacions i províncies (òpics) en un gestor web corporatiu pròpi que no depèn d'un tercer i que, per tant, les dades generades estan directament en la base de dades permetent fer qualsevol tipus de tractament a posteriori.

A nivell de desenvolupament del projecte es pot dir que s'han complert els objectius marcats i el que és més important, en el termini establert. Gràcies a la meteodologia SCRUM utilitzada cada entregable que s'havia planificat s'ha pogut entregar dins els terminis sense retrassos considerables.

Les major dificultat amb que s'han fet front alhora de la realització d'aquest projecte ha estat la escassa documentació que tenen moltes les tecnolgies utilitzades degut a que moltes han sortit en els darrers anys, aquesta dificultat però no ha set tant com per suposar un endarreriment en les entregues.

A nivell personal, la realització d'aquest projecte ha suposat un repte i alhora l'adquisició de noves habilitats tant en l'àmbit de les notificacions push, com en l'àmbit de gestió de projecte, organització i planificació. Tot i així, i per acabar les conclusions cal dir que les tecnologies utilitzades (com la majoria de tecnologies de l'àmbit de les tecnologies de l'informació), encara estan en la fase de maduresa, fet que pot portar a que d'un dia per l'altre surtin al mercat noves tecnologies que milloren i retiren les actuals.

REFERENCES

[1] Wikipedia, "Push technology", 22/09/2016. Disponible en: https://en.wikipedia.org/wiki/Push_technology

[2] Wikipedia, "Scrum", 21/10/2016. Disponible en: https://en.wikipedia.org/wiki/Scrum_(software_development)

[3] Trello, Software per la gestió de tasques a mode Kanban, 21/10/2016. Disponible en (Tauler privat, es requereix de permisos): https://trello.com/b/uPNMcoIR/tfg-gestor-notificacions

[4] Git, Software open source gratuït per la gestió del codi font, 21/10/2016. Disponible en: https://git-scm.com/

[5] Bitbucket, Software de repositoris git, 21/10/2016. Disponible en (Repositori privat, es requereix de permisos): https://bitbucket.org/aleiix24/tfg

[6] Amazon, "Amazon Simple Notification Service (SNS)", 21/10/2016. Disponible en: https://aws.amazon.com/es/sns/

[7] Google FCM, "Firebase Cloud Messaging", 21/10/2016. Disponible en: https://firebase.google.com/docs/cloud-messaging/?hl=es

[8] Microsoft Azure NH, "Notification Hubs documentation", 21/10/2016. Disponible en: https://azure.microsoft.com/en-us/documentation/services/notification-hubs/

[9] Google Firebase, "Firebase Pricing", 04/11/2016. Disponible en: https://firebase.google.com/pricing

[10] Wikipedia, "Planning poker", 26/09/2016. Disponible en: https://en.wikipedia.org/wiki/Planning_poker

[11] Wikipedia, "Node.js", 26/09/2016. Disponible en: https://en.wikipedia.org/wiki/Node.js

[12] Wikipedia, "Event-driven programming", 26/09/2016. Disponible en: https://en.wikipedia.org/wiki/Event-driven_programming

[13] Expressjs website, 26/09/2016. Disponible en: https://expressjs.com/

[14] Cordova website, 26/09/2016. Disponible en: https://cordova.apache.org/

[15] Npm, "FCM push", 26/11/2016. Disponible en: https://www.npmjs.com/package/fcm-push

[16] Mysql website, 26/11/2016. Disponible en: http://www.mysql.com/

[17] Passport, "Passport website", 26/11/2016. Disponible en: http://passportjs.org/

[18] Sequelize, "Sequelize docs website", 26/11/2016. Disponible en: http://docs.sequelizejs.com/en/v3/

[19] Angular, "Angular webiste", 26/11/2016. Disponible en: https://angularjs.org/

[20] Bootstrap, "Bootstrap website", 26/11/2016. Disponible en: http://getbootstrap.com/

[21] Angular, "Angular 2 website", 26/11/2016. Disponible en: https://angular.io/

[22] Ionic, "Ionic website", 26/11/2016. Disponible en: https://ionicframework.com/

[23] Cordova, "Cordova website", 26/11/2016. Disponible en: https://cordova.apache.org/
[24] Docker, "Docker webpage", 07/01/2017. Disponible en: https://www.docker.com/
[25] Amazon, "Amazon Elastic Compute Cloud EC2", 07/01/2017. Disponible en: https://aws.amazon.com/es/ec2/
[26] Npm, "Node cron, simple cron-like task scheduler for node.js", 26/11/2016. Disponible en: https://www.npmjs.com/package/node-cron