Utveckling av en mobil applikation i React Native med Wordpress Back-end

Kim French

Examensarbete

Informationsteknik

2019

EXAMENSARBETE	
Arcada	
Utbildningsprogram:	Informationsteknik
Identifikationsnummer:	17447
Författare:	Kim French
Arbetets namn:	Utveckling av en mobil applikation i React Native med Wordpress Back-end
Handledare (Arcada):	Jonny Karlsson
Uppdragsgivare:	

Sammandrag:
Tekniska Läroverkets Kamratförbund r.f. är en studerandeförening med ca 120 medlemmar. Föreningen har en längre tid haft problem med att få ut information till sina medlemmar via sin Wordpress-webbsida och e-postlista. Examensarbetet behandlar hur en mobil applikation av deras webbsida som kan ta emot mobila notifikationer utvecklades.

Syftet med den praktiska delen av examensarbetet är att producera en mobil applikation med hjälp av Javascript-ramverket React Native som är utvecklat av Facebook. Det här gjordes med hjälp av en samling olika bibliotek som underlättar utvecklingen. Innehållet till applikationen hämtades via sidans Wordpress REST API och notifikationerna gjordes med Google Firebase.

I den teoretiska delen beskrives hur utvecklingen av mobila applikationen utfördes med React Native. Det ges även en översikt över React Native och vad dess fördelar och nackdelar är när det gäller mobil utveckling. Här tas även upp hur alla bibliotek installeras och konfigureras för att fungera med React Native.

Till examensarbetet hör det inte hur utvecklingen av Wordpress sidan gjordes. Det tas endast upp hur utvecklingen av själva applikationen gjordes.

Som resultat fick föreningen en mobil applikation som tar inspiration av webbsidans användargränssnitt. Till applikationen går det att skicka notifikationer via konsolen Google Firebase om inlägg på webbsidan, evenemang eller möten.

Till framtidsplanerna hör att ladda upp applikationen till Google play store för att enkelt kunna distribuera och köra ut uppdateringar. En instickningsmodul till Wordpress-sidan skulle behöva skapas för att automatisera skickandet av notifikationer via Google Firebase REST API till applikationen när nya inlägg publiceras. Det här skulle minska mängden manuellt jobb för styrelsen.

Nyckelord:	React Native, Wordpress, Mobil utveckling, Notifikationer
Sidantal:	30
Språk:	Svenska
Datum för godkännande:	27.05.2019

DEGREE THESIS	
Arcada	
Degree Programme:	Information technology
Identification number:	17447
Author:	Kim French
Title:	Mobile development using React Native with Wordpress Back-end
Supervisor (Arcada):	Jonny Karlsson
Commissioned by:	

Abstract:
Tekniska Läroverkets Kamratförbund r.f. is a student association with about 120 members. For a long time, the association has had problems with getting information to its members via its Wordpress website or mailing list.

The purpose of the practical part of the thesis is to produce a mobile application using the Javascript framework React Native, which is developed by Facebook. This was done using a collection of different libraries that simplified the development process. The content for the application was fetched via the Wordpress REST API and the notifications were handled by Google Firebase.

The theoretical part describes how the development of the mobile application was performed with React Native. It also provides an overview of React Native and its advantages and disadvantages when it comes to mobile development. It is also described how all the libraries are installed and configured to work with React Native.

The degree project does not outline how the development of the Wordpress page was done. It only defines how the development of the application itself was performed.

As a result, the association got a mobile application that takes inspiration from their website's user interface. The application allows you to send notifications via the Google Firebase console about new posts, events or meetings.

Some future plans include uploading the application to Google play store to easily distribute and push out updates. A plugin for the Wordpress page would also have to be created to automate the sending of notifications via the Google Firebase REST API to the application when new posts are published. This would reduce the amount of manual work for the board members.

Keywords:	React Native, Wordpress, Mobile Development, Notifications
Number of pages:	30
Language:	Swedish
Date of acceptance:	27.05.2019

INNEHÅLL

1 INLEDNING 7

1.1 Bakgrund 7
1.2 Syfte och mål 7
1.3 Metoder och avgränsningar 8
1.4 Struktur 8

2 REACT NATIVE 8

2.1 Hur fungerar React Native? 9
2.2 För och nackdelar med React Native 9

3 Wordpress 11

3.1 Wordpress Rest API 11

4 installation av utvecklingsmiljö 12

4.1 Basen för utveckling med React Native 12
4.1.1 React Native installation i Linux 13
4.1.2 Android Studio Installation 13
4.2 Bibliotek 15
4.2.1 Installation av React Native Firebase 15
4.2.2 Installation av React Navigation 17
4.2.3 Installation av de övriga biblioteken 18

5 Utveckling av Mobila applikationen 19

5.1 Applikationsflöde 19
5.2 Filstrukturen 19
5.3 Navigationen 20
5.4 Mobila vyer 21
5.4.1 Hämtande av innehåll 22
5.5 Mobila notifikationer 23
5.5.1 Firebase Cloud Messaging Konsol 23
5.5.2 I applikationen 25
5.1 Byggande av installationsfil 25

6 Resultat 27

6.1 Framtidsplaner 27

7 Slutledning 28

Källor 30

Figurer

Figur 1. Hur vyn byggs up med hjälp av React Native bryggan (Eisenman 2015)..... 9
Figur 2. Vad som skall väljas i SDK Platorms i Android Studio. 14
Figur 3. Vad som skall väljas i SDK Tools i Android Studio. 14
Figur 4. Applikationens flöde 19
Figur 5. Hur applikationens filstruktur ser ut. 20
Figur 6. Hur drawer navigationen skrivs i koden. 21
Figur 7. Hur framsidan ser ut i webbläsare och i applikationen på en mobil telefon. 22
Figur 8. Page.js klassen 22
Figur 9. Koden som visar upp vyn i Page.js 23
Figur 10. Var man laddar ner google-services.json. 24
Figur 11. Google firebase konsol 24
Figur 12. Bild av koden som hanterar notifikationerna i applikationen. 25
Figur 13. Applikationens slutresultat. 27

Tabeller

Tabell 1. Wordpress REST API ändpunkter 11
Tabell 2. Paket som måste installeras för utveckling av applikationen 12
Tabell 3. Installation av Node.js 13
Tabell 4. Installation av React Native CLI 13
Tabell 5. Installation av Java från terminalen 13
Tabell 6. Kommandona för att installera React Native Firebase 15
Tabell 7. Hur MainApplication skall se ut i android/app/src/main/java/com/tlk/ 16
Tabell 8. Vilka tillbehör som behövs i build.gradle 16
Tabell 9. android/app/src/main/AndroidManifest.xml 16
Tabell 10. android/app/build.gradle 17
Tabell 11. Terminalkommandon för att installera react navigation och gesture handler 17
Tabell 12. Hur MainActivity skall se ut 18
Tabell 13. Hur settings.gradle måste fixas efter gesture handler installation 18
Tabell 14.Installations kommandon för de övriga biblioteken 18
Tabell 15. Kommandot som skall köras i terminalen för att bygga en signerad nyckel 26
Tabell 16. Vad som borde läggas till i applikationens gradle.properties 26
Tabell 17. Vad som borde läggas till i applikationens build.gradle 26

Definitioner

Content Management System (CMS) – Innehållshanteringssystem
Representational State Transfer Application Programming Interface (REST API) - Applikationsprogrammeringsgränssnitt
Firebase Cloud Messaging (FCM) – En kross-plattform lösning för att skicka meddelanden
Tema - Hur wordpress definierar hur sidan skall se ut
Javascript Extensible Markup Language (JSX) - Programmeringsspråk
Hypertext Preprocessor (PHP) - Programmeringsspråk
JAVA – Programmeringsspråk
iOS – Operativsystemet Apple telefoner använder sig av
Android – Operativsystem som många telefoner använder sig av
Virtual Document Object Model (Virtual DOM) – Dokumentträd som förklarar relationer mellan element.

1 INLEDNING

I dagens läge kommer 47,2% av världens webbtrafik från mobila enheter. (Statista 2018) 99% av de framgångsrikaste företagen i Finland har webbsidor anpassade för mobila enheter. Mobila applikationer är dock inte lika vanliga trots att de fungerar som ypperliga informationskanaler. (Bearingpoint 2019)

1.1 Bakgrund

Tekniska Läroverkets Kamratförbund r.f. (TLK) är en studerandeförening i yrkeshögskolan Arcada med ca 120 medlemmar (TLK 2019). Förening har länge haft problem med att få medlemmarna aktiva på TLK's hemsida. Deras webbsida är byggd med Wordpress och det enda sättet som medlemmar får någon information om nya inlägg på sidan just nu är via en mailinglista som man måste själv registrera sig till.
Dessa brister gav upphov till en idé om att bygga en mobil applikation som skulle kunna ta emot notifikationer om nya inlägg och visa upp samma innehåll som finns på webbsidan. Jag valde att bygga applikationen med hjälp av React Native som är ett Javascript ramverk utgiven av Facebook. (React Native 2019) Det här biblioteket valdes på grund av att det går att utveckla för flera miljöer samtidigt och har en stor gemenskap av utvecklare som producerar öppen källkod.

1.2 Syfte och mål

Syftet med den praktiska delen av examensarbetet har varit att producera en mobil applikation av TLK's webbsida som skall fungera minst lika bra som själva webbsidan och meddela medlemmar med applikationen att nya inlägg kan läsas via notifikationer när ett nytt inlägg kommer till sidan.
Målsättningen med examensarbetes teoretiska del är att beskriva hur utvecklingen av TLK's mobila applikation utfördes med React Native. Utöver det här ges även en översikt över React Native och vad dess fördelar och nackdelar är när det kommer till mobil utveckling gentemot nativ.

1.3 Metoder och avgränsningar

Applikationen som har utvecklats bygger på en Wordpress-sida som har utvecklats tidigare av en alumn i föreningen. Det kommer inte att tas upp hur den här utvecklingen har gjorts.
Jag har valt att utveckla applikationen med hjälp av React Native för att den ger möjligheten att utveckla för både Android och iOS-miljöerna samtidigt och har många färdiga bibliotek som gör att utvecklingen går snabbare.
Applikationen har planerats att likna webbsidan mycket och ta inspiration från den när det kommer till funktionalitet. Applikationens testning kommer utföras under några månaders tid med medlemmar från föreningens informationsutskott för att kolla att allting fungerar som det ska före man lanserar den till alla medlemmar.

1.4 Struktur

Resten av examensarbete är strukturerat enligt följande:
I nästa kapitel gås det igenom vad React Native är och hur det fungerar samt varför ramverket valts för applikationsutvecklingen. I kapitel 3 tas det en titt på Wordpress som TLK har som back-end till sin webbsida. Utvecklingen av mobila applikationen beskrivs i kapitel 4 med focus på hur den gjordes och vilka problem som uppstod. De sista två kapitlen av slutarbete utvärderar den utvecklade applikationen och vad som skulle kunna förbättras och utvecklas vidare.

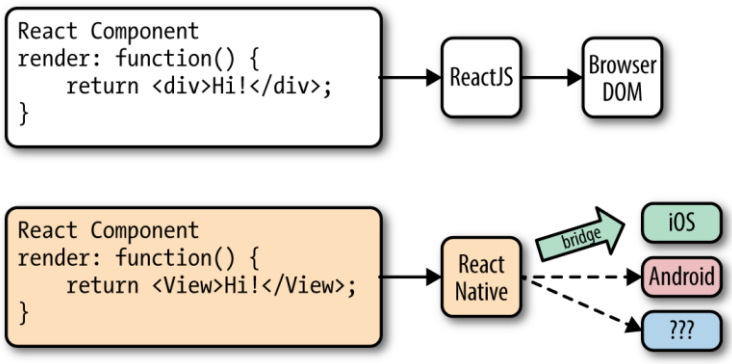
2 REACT NATIVE

React Native är ett Javascript-ramverk utvecklat av Facebook som bygger upp nativa vyer för både iOS- och Android-miljöer. Man kan utveckla för båda operativsystemen samtidigt på grund av att man skriver största delen av koden i Javascript. Det här gör det också enklare för webbutvecklare att lära sig React Native på grund av att det använder sig av ett språk som är menat för webben och som många redan kan från förut.

Ramverket använder sig också av JavaScript XML (JSX) som liknar lite märkspråket XML. Det används för att beskriva hur vyerna skall byggas upp av de olika komponenterna man skriver i Javascript.
React Native fungerar med att kommunicera med operativsystemets kod genom en API brygga som sedan tar hand om hur vyerna skall visas. Det här betyder att man inte bygger upp webbvyer utan koden körs med nativa komponenter som om man skulle ha skrivit den i Java eller Objective-C. Det här betyder att man också har tillgång till telefonens alla funktioner inklusive kamera, lagringsmedia eller fingeravtryckssensor. (Eisenman 2015)

2.1 Hur fungerar React Native?

React Native använder sig av en Virtual Document Object Model (Virtual DOM) för att beskriva hur en vy skall se ut. En DOM är ett dokumentträd som förklarar hur olika element relaterar sig till varandra.
Eftersom det handlar om en virtuell DOM kan React Native kalla på Objective-C eller Java API:t som skriver ut vyn med dess egna komponenter.
Det här betyder att om man skriver ”<View>” i React Native kan bryggan förvandla det till motsvarande ”<UIView>” som finns i Objective-C (se figur 1).



Figur 1. Hur vyn byggs up med hjälp av React Native bryggan (Eisenman 2015)

2.2 För och nackdelar med React Native

React Native har många för- och nackdelar när det kommer till utveckling av en mobil applikation jämfört med att skriva med nativ kod. Till de största fördelarna hör att man kan utveckla för båda mobila operativsystemen samtidigt och återanvända till och med

87% av koden. (Eisenman 2015) Det här betyder att kodbasen hålls i stort sett lika för båda operativsystemen och man behöver inte ha två team som jobbar på två olika versioner av applikationen.

Att kunna snabbt iterera och testa sin kod är en stor fördel och det har React Native med sin ”Live Reload” egenskap som gör att man kan se förändringar på några sekunder i emulatorn. Att göra det här med nativ kod går också men det förutsätter kompilering av koden som Javascript inte behöver och kan ta mellan 15 sekunder till 20 minuter. (Peal 2018, Arora 2018)

För att React Native använder sig av Javascript kan man använda sig mycket av färdiga bibliotek och kod som finns på internet. En stor del kan laddas ner från NPM som ger dig möjligheten att försnabba utvecklingen av applikationen. (Peal 2019)

React Native ger möjligheten att snabbt komma igång och utveckla en simpel prototyp av en mobil applikation jämfört med att skriva koden i både Java och Objective-C.

> “React Native is a great choice for simple applications where the APIs have a clear bridge between two platforms. In the earlier days of the Airbnb mobile apps, the platform was a great accelerator for the simpler tools within the mobile application. Eventually, the APIs will not operate exactly the way you want and you will have to dive into the native libraries to make the necessary adjustments.” (Conrad 2018)

Till nackdelarna hör att större och mera krävande applikationer kan kräva att man ändå måste skriva nativ kod för att tillämpa funktionalitet som inte går ännu att göra med React Native. Det här betyder att som utvecklare måste man kanske ändå behöva behärska Java eller Objective-C.

Från en prestandas synvinkel är inte React Native lika bra som nativ kod som körs på telefonen. Javascript-motorn måste läggas med i koden på Android som gör att applikationen blir större än den skulle behöva vara.

Bryggan mellan Javascript och nativa koden tar också en del prestanda som i nativa applikationer inte skulle finnas. Om allting är skrivet bra är det här inget problem men det kan uppstå flaskhalsar på grund av bryggan. (Arora 2018, Peal 2018)

Biblioteket är mycket ungt och har inte ännu kommit ut ur alfa-stadiet och har många små problem. Det kommer många små förändringar till biblioteket som utvecklar det vidare men det kommer att ta tid före alla problem är lösta. (Peal 2018)

3 WORDPRESS

Wordpress är ett innehållshanteringssystem (CMS) skrivet i PHP och har givits ut under GPL-licens sedan år 2003 (Wordpress 2019). Ursprungligen har Wordpress varit populärt bland bloggare, men används i dagens läge både av företag och privatpersoner. Som plattform är Wordpress mycket flexibel tack vare teman och insticknings moduler som gör det möjligt för administratörer att enkelt ändra utseende och funktionalitet på deras sidor.

3.1 Wordpress Rest API

Wordpress Rest API lades med i version 4.7 av Wordpress för att göra det enkelt för tredje parter att kommunicera och hämta information från Wordpress webbsidor (Barron 2018). Detta API har använts för att få dynamisk information från webbsidan för att sedan visa upp informationen i mobilapplikationen.
API:t har 12 olika ändpunkter för att hämta information, men den utvecklade mobilapplikationen använder enbart posts, pages, media (se tabell 1).
Från *posts* och *pages* får vi alla inlägg och sidor som har lagats på webbsidan medan från media får vi bilder som har eventuellt länkats i de här sidorna. Informationen kommer ut i JSON-format som är sedan enkelt att hantera.

Tabell 1. Wordpress REST API ändpunkter

Resurser	Rutter
Posts	/wp/v2/posts
Post Revisions	/wp/v2/revisions
Categories	/wp/v2/categories
Tags	/wp/v2/tags
Pages	/wp/v2/pages
Comments	/wp/v2/comments
Taxonomies	/wp/v2/taxonomies
Media	/wp/v2/media

Users	/wp/v2/users
Post Types	/wp/v2/types
Post Statuses	/wp/v2/statuses
Settings	/wp/v2/settings

4 INSTALLATION AV UTVECKLINGSMILJÖ

Detta kapitel beskriver på en grundläggande nivå vilka verktyg som användes för utvecklingen av den mobila applikationen samt hur dessa installeras och konfigureras. De olika verktygen som installeras listas och beskrivs kort i tabell 2.

Tabell 2. Paket som måste installeras för utveckling av applikationen

React Native Firebase	Bibliotek som hanterar mobila notifikationer i applikationen.
React Navigation	Bibliotek som hanterar navigationen mellan vyer i applikationen.
React Native Render HTML	Bibliotek som förvandlar HTML till React Native komponenter.
WP-API	Bibliotek med färdiga funktioner för att hämta information från Wordpress REST API.
Moment	Bibliotek som hanterar förvandlingen mellan tidsenheter.

4.1 Basen för utveckling med React Native

För applikationsutvecklingen installerades alla utvecklingsverktyg på en dator med Ubuntu Linux som operativsystem. Det finns små skillnader i installationsprocessen för olika operativsystem vilka framgår ur dokumentationen i React Native (React Native 2019). Vid installationen av utvecklingsmiljön valdes att inte installera med Expo som är en färdig samling av bibliotek som innehåller React Native. Expo kan göra utvecklingen lättare men gör applikationens storlek större när man inte själv väljer vilka andra bibliotek man behöver för utvecklingen. (Expo 2019) Därför valde jag att köra med en ren installation av endast React Native biblioteket.

4.1.1 React Native installation i Linux

För installation av Node.js körs följande kommandon i terminalen. (se tabell 3).

Tabell 3. Installation av Node.js

curl -sL https://deb.nodesource.com/setup_11.x | sudo -E bash -
sudo apt-get install -y nodejs

Efter det kan man med hjälp av node package manager (NPM) som kommer med *Node.js* installera React Native CLI från terminalen (se tabell 4). Det här kommer också att förenkla installationen av olika bibliotek senare.

Tabell 4. Installation av React Native CLI

npm install -g react-native-cli

För att installera JDK lägger man först till datakatalogen och sedan uppdaterar det till datorn och installerar den (se tabell 5).

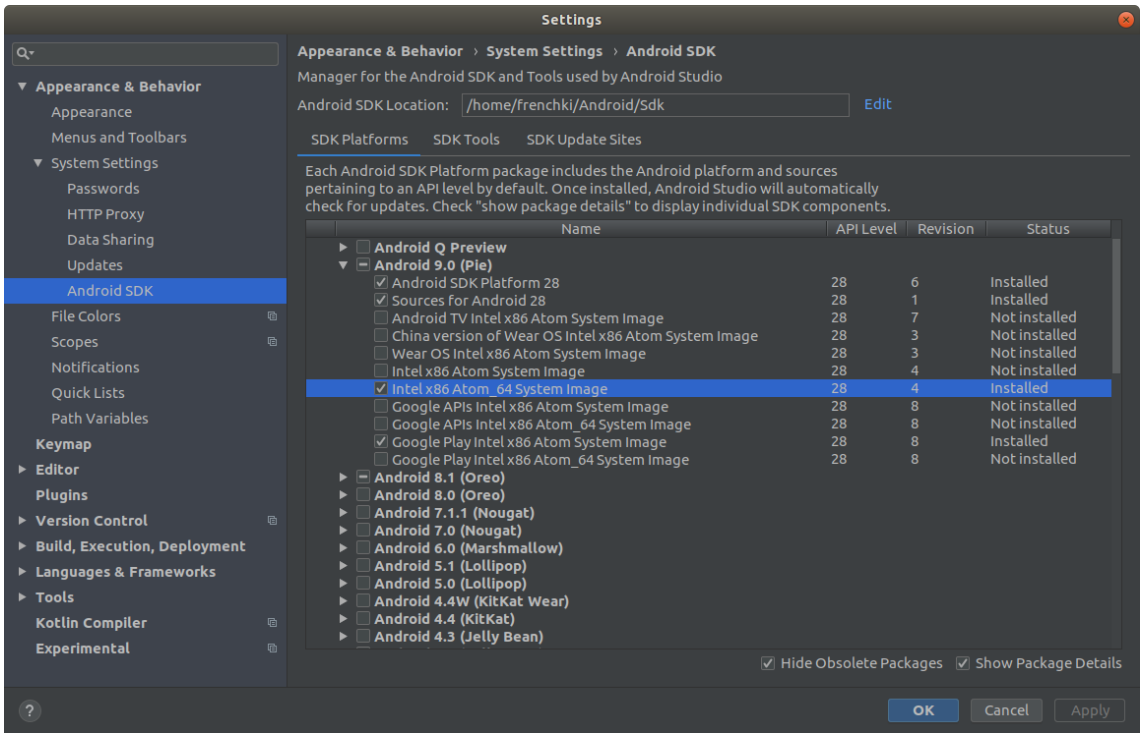
Tabell 5. Installation av Java från terminalen

sudo add-apt-repository ppa:webupd8team/java
sudo apt-get update
sudo apt install oracle-java8-installer

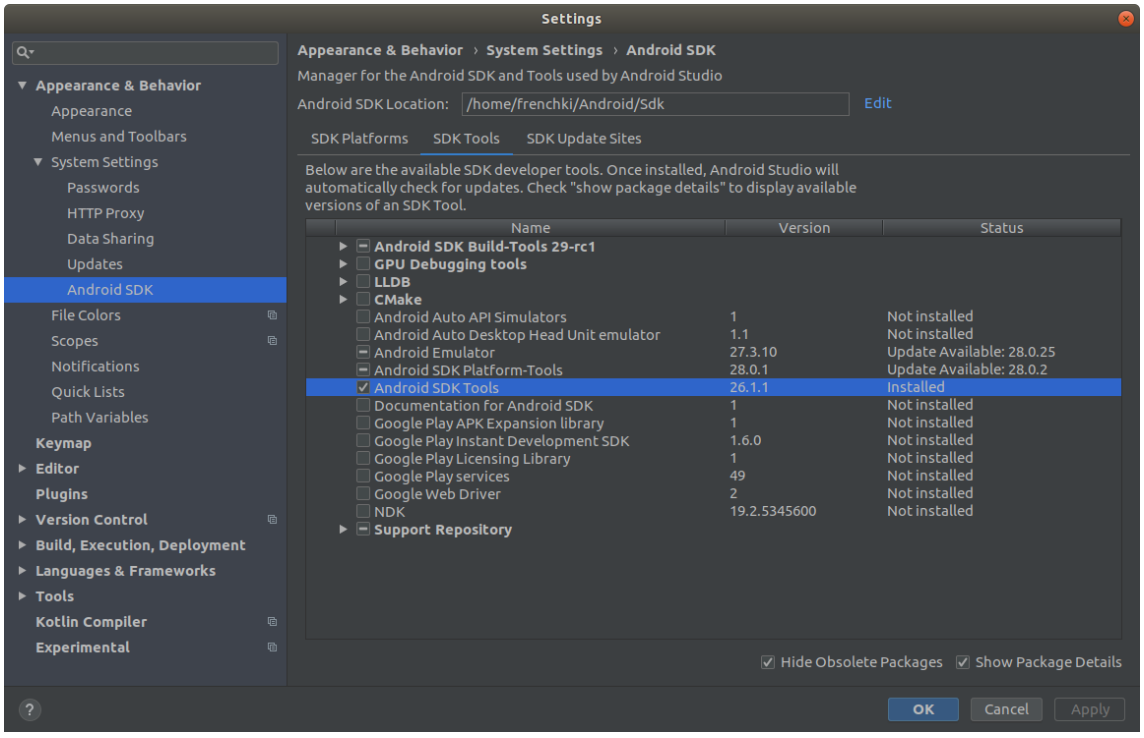
4.1.2 Android Studio Installation

Android Studio går att laddas ner och installeras via deras hemsida (Android Studio, 2019). Efter att den har installerats måste den konfigureras i inställningarna under “appearance & behavior -> System Settings -> Android SDK” Man skall se till att nyaste versionen av Android och korresponderande Android SDK Platform är vald. Det här är på grund av att React Native behöver den nyaste versionen av Android för att man skall kunna simulera applikationen med hjälp av emulatorn.

Intel x86 Atom_64 System Image eller Google APIs Intel x86 Atom System Image skall också vara valda och installerade under SDK Platforms (se figur 2). I SDK Tools skall android emulator, android sdk tools och android sdk platform-tools (se figur 3).



Figur 2. Vad som skall väljas i SDK Platorms i Android Studio.



Figur 3. Vad som skall väljas i SDK Tools i Android Studio.

Efter det här kan man göra i ”Android Virtual Device Manager” emulatorer av olika mobila enheter med olika versioner av Android. Det här använder sig React Native av när man bygger en testversion av applikationskoden.

4.2 Bibliotek

För att göra utvecklingen snabbare och smidigare användes några färdiga bibliotek för att få viss funktionalitet.

För att kunna navigera i applikationen används react navigation 3.3.0 som är ett bibliotek som specialiserar sig på att bygga upp olika typs navigationer (React Navigation 2019). För den här applikationen används två olika färdiga navigationsmöjligheter. Det här biblioteket valdes för navigationen för att React Native rekommenderar den i sin dokumentation.

WP-API Javascript biblioteket valdes för att det gör det enklare att hämta information från de olika REST-ändpunkter i Wordpress via färdiga funktioner istället för att programmera en egen klass för ändamålet. Det fanns inte många bibliotek att välja mellan för denna funktionalitet men WP-API har en bra dokumentation som är enkel att förstå.

Det minst använda biblioteket i projektet är Moment.js som används endast på ett ställe för att förvandla en tidssträng till i finsk format.

React Native Render Html biblioteket används mycket på grund av att det förvandlar HTML kod till 100% React Native komponenter. Det här behövdes på grund av att man får HTML från de olika Wordpress ändpunkterna och det skulle ha tagit mycket tid att skriva en egen klass som skulle ta hand om hur det här borde återges på skärmen.

4.2.1 Installation av React Native Firebase

React Native Firebase installeras med följande kommandon i terminalen (se tabell 6).

Tabell 6. Kommandona för att installera React Native Firebase

```
npm install --save react-native-firebase
react-native link react-native-firebase
```

Firebase-paketen måste läggas med i MainApplication.java filen. (se tabell 7).

Tabell 7. Hur MainApplication skall se ut i android/app/src/main/java/com/tlk/

```
// ...
import io.invertase.firebase.RNFirebasePackage;
import io.invertase.firebase.notifications.RNFirebaseNotificationsPackage;
public class MainApplication extends Application implements ReactApplication {
   // ...
   @Override
   protected List<ReactPackage> getPackages() {
     return Arrays.<ReactPackage>asList(
         new MainReactPackage(),
         new RNFirebasePackage(),
         new RNFirebaseNotificationsPackage()
     );
   }
 };
 // ...
}
```

”build.gradle” är en fil som hanterar hur Java-kod skall kompileras och vilka tillgångar den kommer att behöva för att göra det. Här måste vi definiera Googles instickningsmodul som hanterar notifikationer för mobila applikationer. (se tabell 8, Android Developer 2019b).

Tabell 8. Vilka tillbehör som behövs i build.gradle

```
buildscript {
 // ...
 dependencies {
   // ...
   classpath 'com.google.gms:google-services:4.2.0'
 }
}
```

“AndroidManifest.xml” är en fil som definierar olika paket, aktiviteter, komponenter och rättigheter som applikationen kommer behöva för att kunna kompileras. I den här filen måste vi definiera att applikationen kommer behöva tillgång till att komma åt internet, telefonen har blivit startad och lov att vibrera vid notifikationer för att biblioteket skall fungera och uppfylla sin uppgift. (se tabell 9)

Tabell 9. android/app/src/main/AndroidManifest.xml

```
<manifest ...>
```

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

"android/app/build.gradle" är mycket lik den tidigare nämnda "build.gradle" filen men hanterar istället konfigurationen av applikationens APK-fil. Den måste konfigureras att lägga med biblioteks kod och resurser som kommer från Google i applikationens färdiga kod för att notifikationerna skall fungera. (se tabell 10, Android Developer 2019a)

Tabell 10. android/app/build.gradle

```
dependencies {
  //…
  implementation "com.google.android.gms:play-services-base:16.1.0"
  implementation "com.google.firebase:firebase-core:16.0.8"
}
apply plugin: 'com.google.gms.google-services'
```

4.2.2 Installation av React Navigation

React navigation kan installeras via npm med --save flaggan för att lägga den med i *package.json* filen som ett bibliotek som behövs för applikationen (se tabell 11). Det här går också att göra manuellt men tar mycket mera tid.

Tabell 11. Terminalkommandon för att installera react navigation och gesture handler

npm install --save react-navigation
npm install --save react-native-gesture-handler
react-native link react-native-gesture-handler

Efter det måste MainActivity.java modifieras som finns i "android/app/src/main/java/com/tlk/" katalogen. Det måste läggas med imports för paketet som används (se tabell 12).

Tabell 12. Hur MainActivity skall se ut

```
package com.tlk;

import com.facebook.react.ReactActivity;
import com.facebook.react.ReactActivityDelegate;
import com.facebook.react.ReactRootView;
import com.swmansion.gesturehandler.react.RNGestureHandlerEnabledRootView;

public class MainActivity extends ReactActivity {
  @Override
  protected String getMainComponentName() {
    return "Tlk";
}
@Override
protected ReactActivityDelegate createReactActivityDelegate() {
    return new ReactActivityDelegate(this, getMainComponentName()) {
      @Override
      protected ReactRootView createRootView() {
       return new RNGestureHandlerEnabledRootView(MainActivity.this);
      }
    };
  }
}
```

Det dyker upp ett problem med installationen av gesture handler i Linux som gör att det inte går att starta projektet efter installationen. Det här fixas med att modifiera ”android/settings.gradle” (se tabell 13, React Navigation 2019).

Tabell 13. Hur settings.gradle måste fixas efter gesture handler installation

```
project(':react-native-gesture-handler').projectDir = new File(rootProject.projectDir,
'..\node_modules\react-native-gesture-handler\android')
project(':react-native-gesture-handler').projectDir = new File(rootProject.projectDir,
'../node_modules/react-native-gesture-handler/android')
```

4.2.3 Installation av de övriga biblioteken

React Native Render Html, wpapi och moment kräver inte någon extra konfiguration och kan installeras enkelt med de här kommandona (se tabell 14, WP API 2019).

Tabell 14.Installations kommandon för de övriga biblioteken

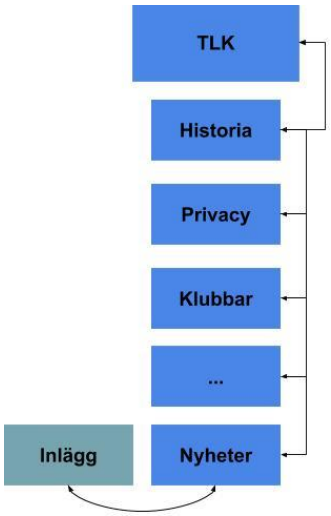
```
npm install --save react-native-render-html
npm install --save wpapi
npm install --save moment
```

5 UTVECKLING AV MOBILA APPLIKATIONEN

Detta kapitel beskriver applikationens struktur och funktionalitet i detalj.

5.1 Applikationsflöde

Efter att en användare har installerat applikationen och startat den är *TLK* vyn den första som hen ser. (se figur 4) Efter det här kan man hoppa fram och tillbaka mellan vyerna vertikalt via navigationen. (se figur 13) Navigationen kommer fram antigen med att klicka på logon i översta högra hörnet eller att svepa på skärmen till höger.
Från *Nyheter* vyn kan man klicka på olika inlägg som har lagts upp på webbsidan. Från dessa inlägg kan man endast navigera bakåt i till nyheter och efter det igen navigera till de andra vyerna.



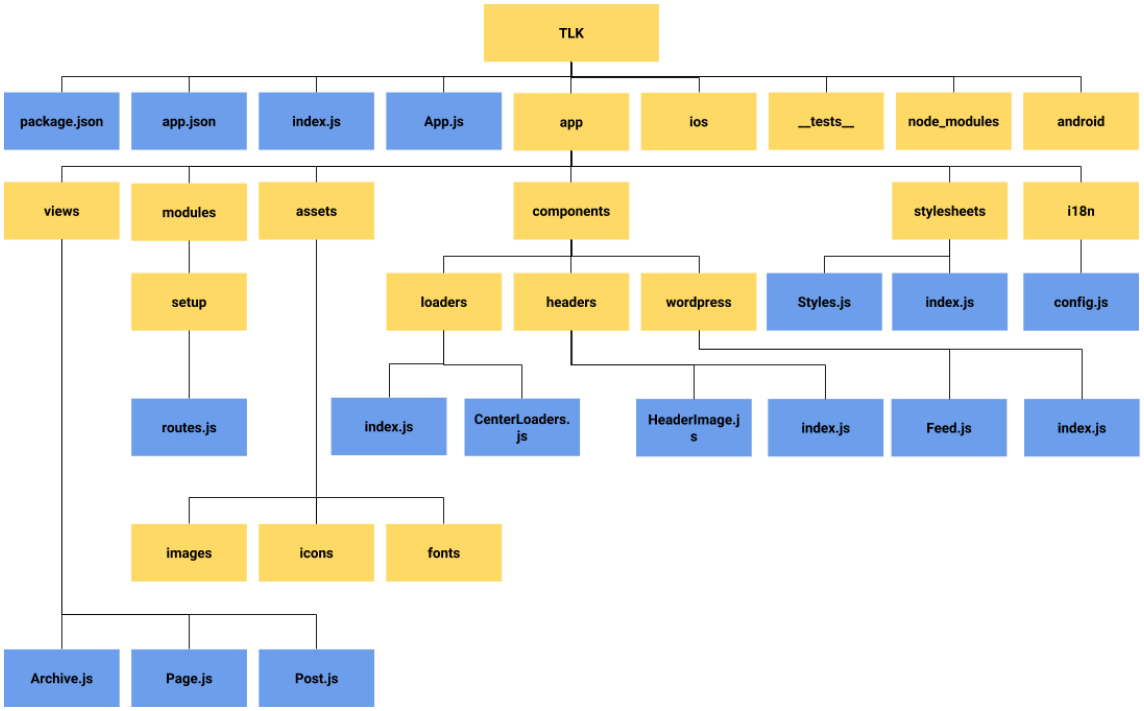
Figur 4. Applikationens flöde

5.2 Filstrukturen

När man initierar ett projekt i React Native bygger den upp en viss filstruktur med filer som bildar en simpel startpunkt för ett projekt. Filstrukturen för den utvecklade applikationen presenteras i figur 5. Katalogen ”ios” används om man skulle skriva kod för det operativsystemet men i det här fallet är den inte relevant eftersom applikationen byggdes för Android. Som namnet säger skulle katalogen ”__tests__” innehålla tester om man skulle vilja göra sådana. Applikationens huvudfil är *App.js* som bygger upp applikationen

från komponenterna som finns i underkatalogerna. Det som jag har förändrat är att lägga till en katalog som heter app som sedan har underkataloger och filer. Det här är för att få en logisk struktur för applikationen som delar upp koden i funktionella delar. Views innehåller vyer som applikationen visar åt användaren. De byggs upp med hjälpkod från *components*-katalogen. Som exempel varje vys bild som kommer i sidhuvudet kommer från koden som finns i *HeaderImage.js*.

Styles.js innehåller stilkod som används för att berätta hur olika komponenter skall visas. *Assets*-katalog innehåller fonter, bilder och ikoner som applikationen använder.



Figur 5. Hur applikationens filstruktur ser ut.

5.3 Navigationen

Navigationen använder sig av React Navigation för att navigera mellan olika vyer i applikationen. Biblioteket är lätt att förstå och gör det enkelt att utveckla olika typs navigationer beroende på vad man behöver. TLK's applikation använder sig av drawer navigation och stacknavigator som är de vanligaste. Valet av drawer navigation basera sig på att få samma känsla som om du vore på websidan för den har också en sådan stils navigation i mobila apparater. Stacknavigationen ger en balk i övre delen av applikationen som igen ger känslan av att det är en applikation och inte en webbsida i en webbläsare.

Navigationen är hårdkodad för tillfället och hämtar information enligt id parametrar som de olika vyerna ger till funktionen som tar hand om att hämta information från Wordpress (se figur 6).

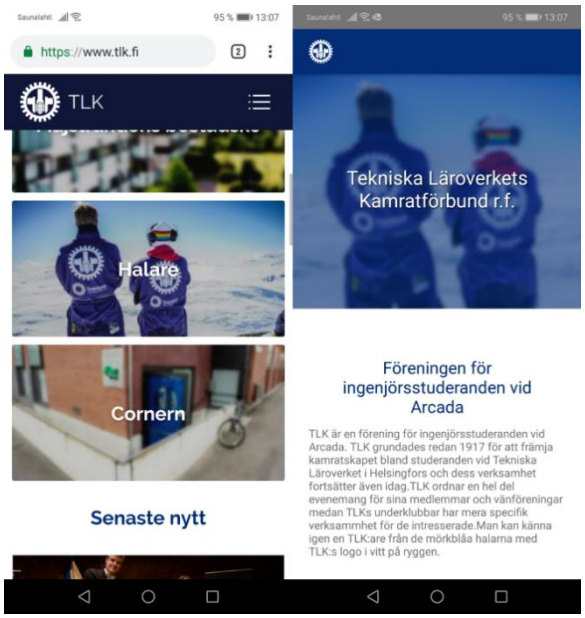
```
const DrawerNavigator = createDrawerNavigator({
    TLK: {screen: Page, params:{id:2,title:"TLK"}},
    Historia: {screen: Page, params:{id:103,title:"Historia"}},
    Privacy:{screen:Page,params:{id:312,title:"Privacy"}},
    /*Mötesprotokoll:{screen:Page,params:{id:404}},*/
    Klubbar:{screen:Page,params:{id:327,title:"Klubbar"}},
    Stadgar:{screen:Page,params:{id:323,title:"Stadgar"}},
    "För företag":{screen:Page,params:{id:325,title:"För Företag"}},
    Medlemmar:{screen:Page,params:{id:349,title:"Medlemmar"}},
    Sångbok:{screen:Page,params:{id:359,title:"Sångbok"}},
    Halare:{screen:Page,params:{id:356,title:"Halare"}},
    Stipendier:{screen:Page,params:{id:354,title:"Stipendier"}},
    Bostäder:{screen:Page,params:{id:351,title:"Bostäder"}},
    /*Evenemang:{screen:Page,params:{id:329}},*/
    Nyheter:{screen: Archive,params:{id:230,title:"Nyheter"}},
    Styrelse:{screen:Page,params:{id:340,title:"Styrelsen"}},
},
{
    initialRouteName: "TLK",
    title:"Test",
    contentComponent: customDrawerComponent,
    contentOptions:{
        activeBackgroundColor: "#032e77",
        activeTintColor: "#FFFF",
        inactiveBackgroundColor: "#FFFF",
        inactiveTintColor: "#032e77",
    }
});
const StackNavigator = createStackNavigator({
    DrawerNavigator:{
        screen: DrawerNavigator
    },
```

Figur 6. Hur drawer navigationen skrivs i koden.

5.4 Mobila vyer

Jag har skrivit tre olika vyer för att ta hand om hur informationen som kommer från API:t skall visas. De har samma namngivning som Wordpress använder sig. (se figur 5). Det här var med flit för att veta bättre vilka vyer filerna kommer bygga upp om man kommer från Wordpress temautveckling. Det går enkelt att skriva in nya vyer här för att i framtiden kunna lägga till mera funktionalitet i applikationen.

Page och post vyerna använder sig mycket av React Native Render Html på grund av att de har mycket HTML-kod som kommer från API:t. Biblioteket har flera funktioner som låter utvecklare modifiera hur och vad som skall visas av HTML i vyn. Använder mig flitigt av ”ignoreNodesFunction” som ger mig frihet att ignorera html om de har en viss klass eller id. Det här behövs för att ignorera på webbsidans framsida senaste nytt delen och blocken som kan ses i figur 7. Archive vyn består endast av kod och komponenter jag själv har skrivit och skulle ha önskat att göra det på det här viset men på grund av HTML:n var det bara mycket enklare att använda sig av ett bibliotek.



Figur 7. Hur framsidan ser ut i webbläsare och i applikationen på en mobil telefon.

5.4.1 Hämtande av innehåll

Hämtande av applikationens innehåll sker via Wordpress REST API. Det här görs med hjälp av WP-API biblioteket. Som exempel visas *Page.js* som hanterar hur huvudnavigationens vyer visas. (se figur 8) När den här vyn för första gången kallas på konstruerar den en ny *WPAPI* klass som har som parameter TLK's Wordpress REST API webbadress. Här definierar vi också andra variabler som *isLoading* som berättar åt komponenten om den borde visa upp en ikon att applikationen laddar in innehållet.

```
class Page extends Component {
    constructor(props){
        super(props);
        this.wp = new WPAPI({ endpoint: 'http://www.tlk.fi/wp-json' });
        this.state = {
            data: null,
            isLoading: true,
            id: this.props.navigation.getParam('id', 'NO-ID'),
        }
    }
    static navigationOptions = ({ navigation }) => ({
        title: navigation.state.params.title || 'default title',
    });
    async componentWillReceiveProps(newProps){
        this.setState({id: newProps.navigation.getParam('id','NO-ID')});
    }
    async shouldComponentUpdate(nextProps, nextState){
        return nextProps.navigation.getParam('id', 'NO-ID') != this.state.id;
    }
    async componentWillUpdate(willUpdate){
        this.state.isLoading = true;
    }
    async componentDidUpdate(didUpdate){
        this.state.isLoading = false;
    }
    async componentDidMount(){
        const data = await this.wp.pages().id(this.state.id);
        this.setState({data:data,isLoading:false});
    }

    render() {
```

Figur 8. Page.js klassen

Funktionen *componentDidMount* som kallas varje gång den här komponenten laddas och här använder vi bibliotekets färdiga funktioner för att berätta vad vi vill hämta från REST API:t. I detta fall hämtar vi en sida med en viss id som kommer från navigationen. Dessa idn är för tillfället hårdkodade till de specifika sidorna som finns just nu på webbsidan.

Som svar får vi ett JSON-objekt som vi sparar och kan sedan använda för att visa innehållet i vyn. Det här gör vi med att ge *content.rendered* från vårt objekt till React Native Render HTML biblioteket som hanterar hur html koden skall visas. (se figur 9)

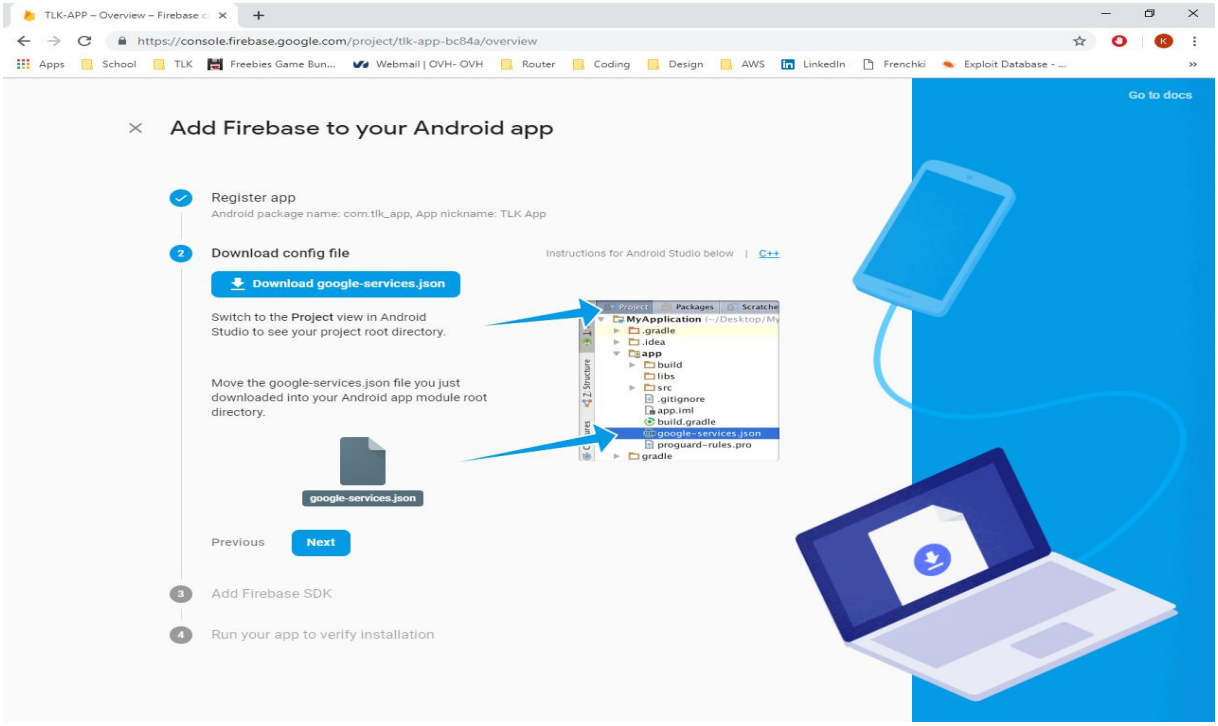
Figur 9. Koden som visar upp vyn i Page.js

5.5 Mobila notifikationer

Google Firebase är en samling av olika produkter som specialiserar sig i att fungera som backend för olika applikationer. Det här kan vara för t.ex. analytik, databaser eller moln funktioner för att nämna några. Till den här applikationen används Firebase Cloud Messaging (FCM) och mera specifikt notifikationsdelen. Det här används för att det skall gå att skicka notifikationer till mobila apparaten.

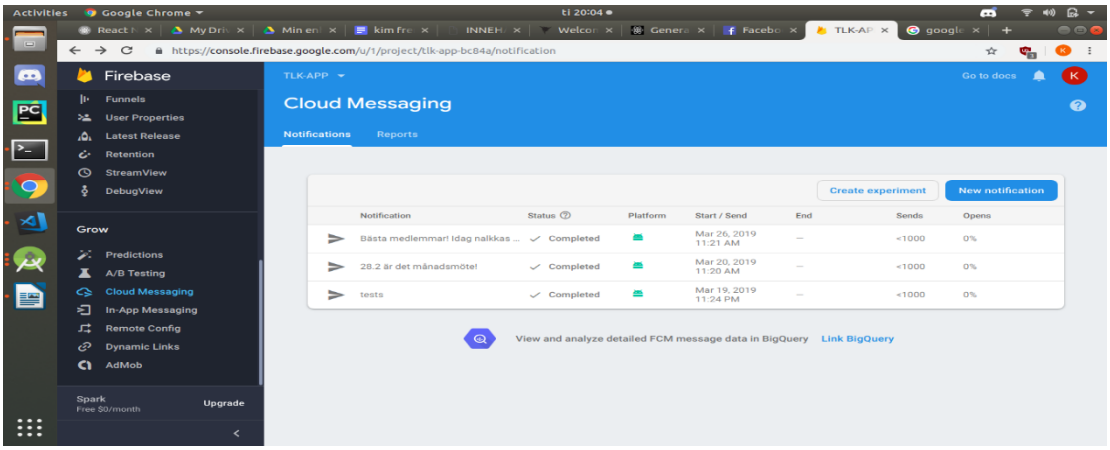
5.5.1 Firebase Cloud Messaging Konsol

Först måste man göra ett konto till Google Firebase och göra ett projekt för Android i konsolen. (se figur 11)



Figur 10. Var man laddar ner google-services.json.

Härifrån får man en fil som heter *google-services.json* som man skall lägga med i projektet under android/app (se figur 10).

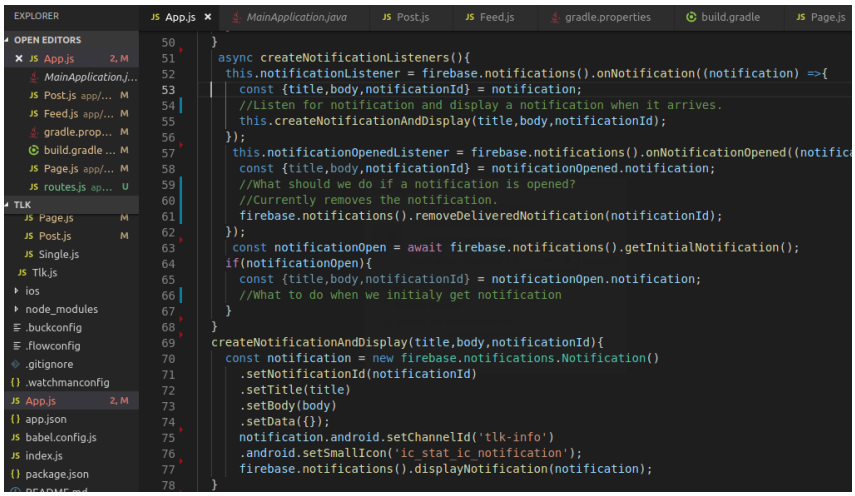


Figur 11. Google firebase konsol

Från den här konsolen kan man sedan skicka till alla som har mobila applikationen notifikationer. Hur man hanterar notifikationerna i applikationen behandlas i nästa avsnitt (se figur 12). (Google Firebase 2019)

5.5.2 I applikationen

I applikationen måste vi hantera vad som kommer att hända med notifikationerna som kommer från Firebase.



Figur 12. Bild av koden som hanterar notifikationerna i applikationen.

React Native Firebase har färdiga funktioner som lyssnar efter notifikationer så vi måste bara skriva kod som hanterar dem (se figur 12). Vi kan modifiera vad som händer med hjälp av våra egna funktioner och modifiera hur notifikationerna skall se ut och vad som ska hända när man öppnar dem. Tillsvidare öppnas applikationen av att man trycker på en notifikation men det händer inget i applikationen. Man skulle kunna som exempel navigera användaren till ett inlägg via notifikationerna.

5.1 Byggande av installationsfil

För att kunna installera och distribuera mobila applikationen måste man bygga en Android Application Package (APK) fil som fungerar lite som en ZIP-fil som kan installera applikationer.

Det här görs mycket likt som om man skulle göra det i Android studio med att först generera en signerad nyckel som skall länkas till applikationen. Det här krävs för Android installerar inte osignerade applikationer. Google Playstore kräver också det här för att man skall kunna distribuera applikationer via dem. Det är viktigt att man håller den här nyckeln hemlig för utan den kan man inte uppdatera applikationen i framtiden. För att göra en nyckel måste kommandot i tabell 15 köras.

Tabell 15. Kommandot som skall köras i terminalen för att bygga en signerad nyckel

```
sudo keytool -genkeypair -v -keystore tlk-app-release.keystore -alias tlk-app -keyalg RSA -keysize 2048 -validity 10000
```

Det här lagar en fil som kommer att heta tlk-app-release.keystore som ska läggas i android/app. Sedan måste android/gradle.properties modifieras på det här viset (se tabell 16).

Tabell 16. Vad som borde läggas till i applikationens gradle.properties

```
MYAPP_RELEASE_STORE_FILE=tlk-app-release.keystore
MYAPP_RELEASE_KEY_ALIAS=tlk-app
MYAPP_RELEASE_STORE_PASSWORD=******
MYAPP_RELEASE_KEY_PASSWORD=******
```

Lösenordet skall bytas med det som har lagts in när man genererade nyckeln.

För att gradlew skall veta hur den skall bygga applikationen måste också android/app/build.gradle förändras lite. (se tabell 17)

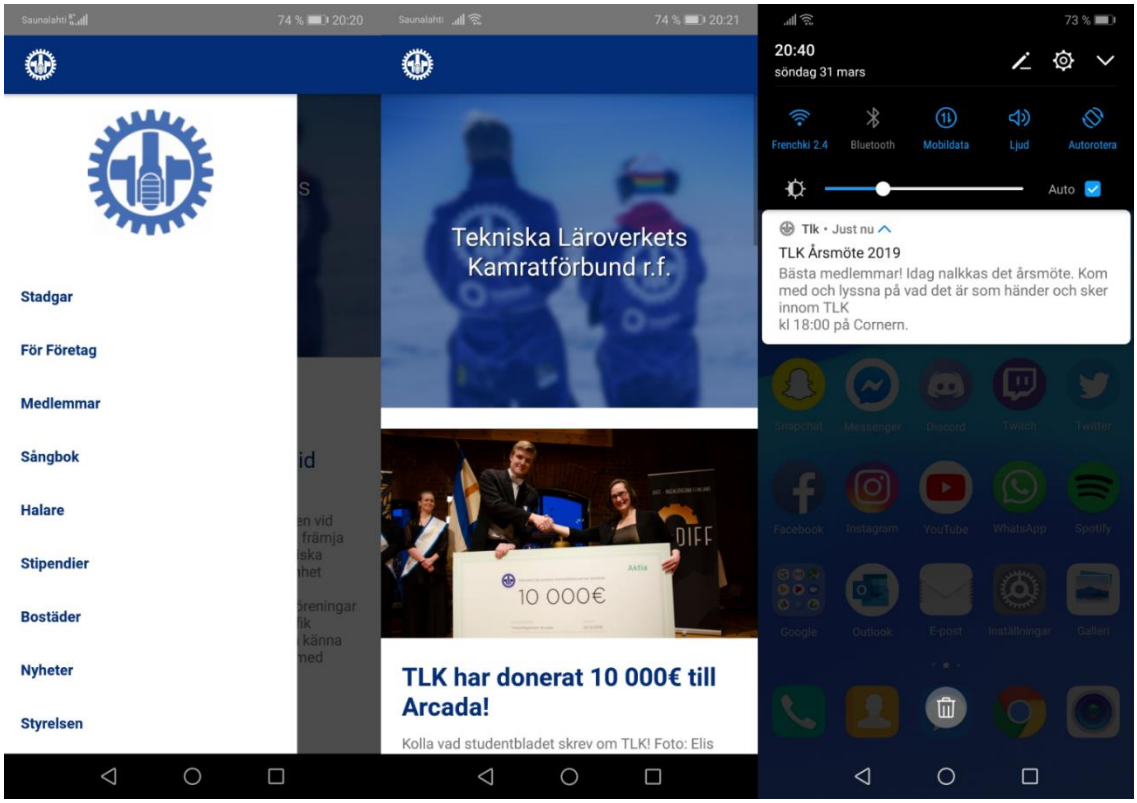
Tabell 17. Vad som borde läggas till i applikationens build.gradle

```
...
android {
    ...
    defaultConfig { ... }
    signingConfigs {
        release {
            if (project.hasProperty('MYAPP_RELEASE_STORE_FILE')) {
                storeFile file(MYAPP_RELEASE_STORE_FILE)
                storePassword MYAPP_RELEASE_STORE_PASSWORD
                keyAlias MYAPP_RELEASE_KEY_ALIAS
                keyPassword MYAPP_RELEASE_KEY_PASSWORD
            }
        }
    }
    buildTypes {
        release {
            ...
            signingConfig signingConfigs.release
        }
    }
}
...
```

Sista steget är att gå till android foldern och köra bygg kommandot "./gradlew assembleRelease" för att bygga applikationen.

6 RESULTAT

Som slutresultat fick Tekniska Läroverkets Kamratförbund r.f. en mobilapplikation som tar sin inspiration från webbsidans användargränssnitt. Från Firebase konsolen kan styrelsen skicka ut notifikationer om nya inlägg på webbsidan, evenemang eller möten (se figur 13). Det här borde hjälpa i aktiveringen av föreningens medlemmar. Mobila applikationen har inte under skrivandets stund givits ut till medlemmarna utan kommer gå igenom en testningsfas med några medlemmar för att kolla att allting fungerar som det ska och möjligen göra små uppdateringar till användargränssnittet.



Figur 13. Applikationens slutresultat.

6.1 Framtidsplaner

Den mobila applikationen kommer säkert laddas upp till Google play store efter att testningen är klar och små förändringar i användargränssnittet gjorts. Det här kommer göra det enkelt att distribuera till medlemmarna och köra ut uppdateringar till applikationen. Planer för längre i framtiden är att göra en instickningsmodul till Wordpress sidan som skulle automatisera skickande av notifikationer när ett nytt inlägg läggs upp till sidan. Det här skulle göras med att skicka data till Firebase REST API:t som sedan skulle skicka ut

det till mobila applikationens användare. I instickningsmodulen skulle man också kunna lägga till ett fält i admin panelen var styrelsen skulle kunna skicka ut små meddelanden med info utan att behöva logga in på Firebase konsolen för att göra det. En möjlig utvecklingsidé skulle vara att fixa så att medlemmar skulle kunna visa upp att de har ett aktivt medlemskap i föreningen via applikationen. Det här skulle kunna ske via en inloggning som sedan kollar från en databas om de har betalat medlemskapsavgiften för läsåret. Det här skulle göra att man inte skulle måsta ha ett kort med föreningens klistermärke för att bevisa sitt medlemskap vid evenemang.

7 SLUTLEDNING

I detta arbete har det tagits fram vad React Native är och hur man utvecklar en Android mobil applikation med ramverket. Som praktiskt exempel används det hur Tekniska Läroverkets Kamratförbunds r.f's mobila applikation utvecklades.

TLK har en längre tid haft problem att få medlemmar att läsa inlägg på deras webbsida och få på det viset ut information. Som lösning till dessa problem utvecklades en mobil applikation med React Native var man kan få notifikationer om nya inlägg och kan läsa dem i applikationen. Det här görs med att hämta information från webbsidan via Wordpress REST API och Google Firebase. För dessa funktionaliteter användes olika bibliotek för att försnabba utvecklingen av applikationen.

Det här är en av dom stora fördelar med att utveckla med React Native. Det finns en stor gemenskap bakom React Native som producerar öppen källkod som kan användas i projekt. Dessutom går det med "Live Reload" att snabbt se förändringarna med sin kod som inte är möjligt i nativ utveckling. En annan fördel med React Native är att det går att utveckla för både Android och iOS-miljöerna samtidigt men det här tas inte upp i det här arbete.

Nackdelar med React Native är att applikationen blir större när Javascript motorn måste läggas med i kompilerade koden. Dålig skriven kod kan göra att prestandan blir sämre på grund av att den kommunicerar med den nativa koden på telefonen som kan göra större applikationer krångliga.

Slutprodukten av arbete är en mobil applikation som hämtar sitt innehåll via Wordpress REST API och får notifikationer med hjälp av Google Firebase. Notifikationerna skickas

via konsolen i Firebase men som framtidsplan är det meningen att automatisera det här så att notifikationerna genast skickas ut vid publiceringen av nya inlägg på webbsidan.
Personligen tycker jag att React Native fungerar bra för små applikationer som inte är komplexa. För större applikationer med större team och resurser skulle jag överväga att istället utveckla nativt. Det här är på grund av att React Native är ännu ett ungt ramverk och får konstant nya uppdateringar som kan ställa till med problem vid övergången till en ny version av ramverket.

KÄLLOR

Android Studio, 2019, Android Studio Tillgänglig: https://developer.android.com/studio, Hämtad: 29.4.2019

Android Developer, 2019a, *Configure your build* Tillgänglig: https://developer.android.com/studio/build Hämtad: 10.4.2019

Android Developer 2019b, *The Google Services Gradle Plugin* Tillgänglig: https://developers.google.com/android/guides/google-services-plugin Hämtad: 10.4.2019

WP API, 2019, Tillgänglig: http://wp-api.org/node-wpapi/ Hämtad 10.4.2019

NPM JS, 2019, Tillgänglig: https://www.npmjs.com/ Hämtad: 9.4.2019

Peal, G., 2018 *React Native at Airbnb: The Technology* Tillgänglig: https://medium.com/airbnb-engineering/react-native-at-airbnb-the-technology-dafd0b43838 Hämtad: 9.4.2019

Eisenman, B., 2015, *Learning React Native*, O'Reilly Media Inc Tillgänglig: https://www.oreilly.com/library/view/learning-react-native/9781491929049/ Hämtad: 13.3.2019

Conrad, A., 2018, *Why Airbnb is Moving Off of React Native* Tillgänglig: https://softwareengineeringdaily.com/2018/09/24/show-summary-react-native-at-airbnb/ Hämtad: 8.4.2019

Arora, G., 2018, *Android to React Native, The good, Bad & Ugly* Tillgänglig: https://medium.com/@gauravsapiens/android-to-react-native-the-good-bad-ugly-40e581f92c64 Hämtad: 8.4.2019

BearingPoint 2019 *Digital Leaders In Finland* Tillgänglig: https://www.bearingpoint.com/en-fi/our-success/digital-leaders/ Hämtad: 8.4.2019

Wordpress, 2019, Wordpress Tillgänglig: https://wordpress.org/ Hämtad: 8.1.2019

React Native, 2019, React Native Tillgänglig: https://facebook.github.io/react-native/ Hämtad: 8.1.2019

Google Firebase, 2019, Google Firebase Tillgänglig: https://firebase.google.com/ Hämtad: 1.8.2019

Statista, 2018, *Percentage of mobile device website traffic worldwide from 1st quarter 2015 to 4th quarter 2018* Tillgänglig: https://www.statista.com/statistics/277125/share-of-website-traffic-coming-from-mobile-devices/ Hämtad: 16.1.2019

Expo, 2019, *Why not Expo?* Tillgänglig https://docs.expo.io/versions/latest/introduction/why-not-expo/ Hämtad: 1.4.2019

Barron, B., 2018, A Quick Start Guide To The WordPress REST API, Tillgänglig: https://www.wpsuperstars.net/wordpress-rest-api/bilagor Hämtad: 31.03.2019

TLK, 2019, Tillgänglig: https://www.tlk.fi/presentation-historia/ Hämtad: 31.3.2019

React Navigation, 2019, Tillgänglig: https://reactnavigation.org/, Hämtad 24.4.2019