PWA Push Notification in Ionic

Author – Shivansh Srivastava.

Introduction – In this document I am going to tell you that how can you implement push notifications in your pwa ionic app with help of firebase messaging and an API call to fcm with NodeJs as backend to trigger the push notifications when needed.

Setting up Firebase Messaging –

```
ng add @angular/pwa
ng add @angular/fire
```

Now generate a service for our messaging logic –

```
ionic g service services/messaging
```

Now go to environments/environment.ts to include the configuration:

```
export const environment = {
  production: false,
  firebase: {
    apiKey: "",
    authDomain: "",
    databaseURL: "",
    projectId: "",
    storageBucket: "",
    messagingSenderId: "",
    appId: ""
  }
};
```

To make Firebase messaging work, we need to create a new file and include our information in here again.

Go ahead and create a file at src/firebase-messaging-sw.js(exactly same name)

```
importScripts('https://www.gstatic.com/firebasejs/7.16.1/firebase-app.js');
importScripts('https://www.gstatic.com/firebasejs/7.16.1/firebase-messaging.js');

firebase.initializeApp({
   apiKey: '<your-key>',
   authDomain: '<your-project-authdomain>',
   databaseURL: '<your-database-URL>',
   projectId: '<your-project-id>',
   storageBucket: '<your-storage-bucket>',
   messagingSenderId: '<your-messaging-sender-id>'
});

const messaging = firebase.messaging();
```

It is important to use the **same Firebase version** in this file like in your package.json!

Now create a new file at src/combined-sw.js and then simply import both service workers like this:

```
importScripts('ngsw-worker.js');
importScripts('firebase-messaging-sw.js');
```

In **angular.json** find the **webmanifest** entry and add your two new service workers below:

```
"assets":[
  ..
 "src/manifest.webmanifest",
 "src/combined-sw.js",
 "src/firebase-messaging-sw.js"
]
```

Now we need to add a generic ID to our src/manifest.webmanifest, which is in fact the same for every app as it's the GCM ID of Google.

```
"gcm_sender_id": "103953800507",
```

Last step is to change the main module file of our app so we can load our new combined service worker instead of the default Angular service worker.

In **app/app.module.ts** :

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { RouteReuseStrategy } from '@angular/router';

import { IonicModule, IonicRouteStrategy } from '@ionic/angular';
import { SplashScreen } from '@ionic-native/splash-screen/ngx';
import { StatusBar } from '@ionic-native/status-bar/ngx';

import { AppComponent } from './app.component';
import { AppRoutingModule } from './app-routing.module';
import { ServiceWorkerModule } from '@angular/service-worker';
import { environment } from '../environments/environment';

import { AngularFireModule } from '@angular/fire';
import { AngularFireMessagingModule } from '@angular/fire/messaging';

@NgModule({
 declarations: [AppComponent],
 entryComponents: [],
 imports: [
  BrowserModule,
  IonicModule.forRoot(),
  AppRoutingModule,
  ServiceWorkerModule.register('combined-sw.js', {
   enabled: environment.production,
  }),
  AngularFireModule.initializeApp(environment.firebase),
  AngularFireMessagingModule,
 ],
 providers: [
  StatusBar,
  SplashScreen,
  { provide: RouteReuseStrategy, useClass: IonicRouteStrategy },
 ],
```

```
  bootstrap: [AppComponent],
})
export class AppModule {}
```

Listening to Push Notifications –

Open the **services/messaging.service.ts** and insert :

```
import { Injectable } from '@angular/core';
import { AngularFireMessaging } from '@angular/fire/messaging';
import { tap } from 'rxjs/operators'

@Injectable({
  providedIn: 'root'
})
export class MessagingService {
  token = null;

  constructor(private afMessaging: AngularFireMessaging) {}

  requestPermission() {
    return this.afMessaging.requestToken.pipe(
      tap(token => {
        console.log('Store token to server: ', token);
      })
    );
  }

  getMessages() {
    return this.afMessaging.messages;
  }
}
```

Now we can put this service to use within our page, and we simple subscribe to the different functions and display some toasts and alerts about the data. Now simply go to the page where you want to have this service, in this document I am taking example of home page:

So, go to **home/home.page.ts** :

```
import { Component } from '@angular/core';
import { MessagingService } from '../services/messaging.service';
import { AlertController, ToastController } from '@ionic/angular';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

  constructor(
    private messagingService: MessagingService,
    private alertCtrl: AlertController,
    private toastCtrl: ToastController
  ) {
    this.listenForMessages();
  }
  listenForMessages() {
    this.messagingService.getMessages().subscribe(async (msg: any) => {
      const alert = await this.alertCtrl.create({
        header: msg.notification.title,
        subHeader: msg.notification.body,
        message: msg.data.info,
        buttons: ['OK'],
      });
      await alert.present();
    });
  }
  requestPermission() {
    this.messagingService.requestPermission().subscribe(
      async token => {
        const toast = await this.toastCtrl.create({
          message: 'Got your token',
          duration: 2000
        });
        toast.present();
      },
      async (err) => {
        const alert = await this.alertCtrl.create({
          header: 'Error',
```

```
        message: err,
        buttons: ['OK'],
      });
      await alert.present();
    });
  }
}
```

Note : Here you got a token in requestPermission function you can use this token to send notification backend code to that particular device.

There are two cases , first that if you want to send notification to a single user only you can just paste that string token to nodejs code (which is rare case), second case is that you want to send the notifications to multiple users so you can store these tokens into your database and when you write the code for pushing the notification you can get those tokens from database and send notifications to them.

Sending Notifications from NodeJs Backend :

Now to trigger the notification from backend we need to call the API of firebase to call the push:

In this documentation we are using **axios** library to make the API call

Your nodeJs code will look like the below code…

In notification title you can give your desired title and in body the body of your message, you can also set the action you want the user to perform on clicking your notification.

In **Authorization:** provide your firebase authorization key which you can get under cloud messaging tab in firebase.

In **to:** provide the token of the device to which you want to send notification which was received in requestPermission() function above.

```
const payload = {
    notification: {
      title: msg_title,
      body: msg_body,
      click_action: 'https://app.rayeye.in/vm-review-folders',
    },
    to: i.device_token,
  };
  const options = {
    headers: {
      Authorization: firebaseCloudMessagingAppAuthKey,
      'Content-Type': 'application/json',
    },
  };
axios
.post('https://fcm.googleapis.com/fcm/send', payload, options)
.then(response => {
    console.log(`statusCode: ${response.status}`);
})
.catch(error => {
    console.error(error);
})
```

I hope this documentation will help you in implementing push notifications in pwa ionic app.

References :

- How to Create an Ionic PWA with Web Push Notifications | Devdactic - Ionic Tutorials
- For any queries you can contact me - www.linkedin.com/in/shivansh-srivastava88