

# Desarrollo de aplicación “Mi Agenda”

Amagua Jhoel, Curipoma David  
Escuela Politécnica Nacional (EPN), Quito - Ecuador

**Resumen** – En este documento se presenta el desarrollo de una aplicación móvil en Ionic-Angular conectada a un Realtime Database (Base de datos en tiempo real) en Firebase, donde su funcionalidad permite al usuario interactuar de una manera rápida con la aplicación, permitiendo ingresar notas, tareas, pendientes, etc., en una fecha determinada, de igual manera permite eliminar información antes mencionada, interactuando como una agenda personal. El documento también muestra el modo de uso de la aplicación y contiene capturas de pantalla como evidencia y guía de la misma. Finalmente, el documento contiene conclusiones del desarrollo y está debidamente referenciado.

**Índices** – Aplicación móvil, desarrollo, Realtime Database, Ionic, Angular, Firebase.

## I. INTRODUCCIÓN

El presente artículo muestra la creación de una aplicación móvil, la cual es una de las herramientas más utilizadas en la actualidad ya que permiten realizar muchas tareas y facilitar la vida de las personas, esta aplicación está desarrollada en Ionic, el cual nos permite desarrollar aplicaciones móviles híbridas, estas son construidas con tecnologías web basadas en HTML, CSS y JavaScript, este tipo de aplicaciones son almacenadas en una aplicación nativa que usa un WebView de la plataforma móvil. Ionic se puede ver como un framework front-end y de interfaz de usuario porque contempla estos 2 aspectos de interacción con funcionalidad y visualización [1]. Una de las herramientas que van de la mano con la creación de aplicaciones móviles es Firebase, la cual es una plataforma que utiliza la infraestructura de Google para mejorar nuestras aplicaciones móviles, ofrece algunas herramientas muy útiles donde se destacan Hosting, Authentication, Realtime Database, este último permite almacenar y sincronizar datos en tiempo real los cuales están alojados en la nube, sin necesidad de contar con servidores [2]. A continuación, se muestra el desarrollo de la aplicación móvil desarrollada en Ionic con conexión a una base de datos en tiempo real alojada en Firebase.

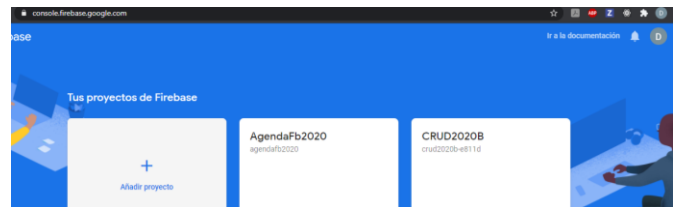
## II. DESARROLLO

Como ya es de conocimiento las aplicaciones móviles son programas diseñados para que tengan funcionalidad en diferentes dispositivos tales como teléfonos, tabletas entre otros, estas aplicaciones permiten al usuario realizar diferentes tipos de actividades tanto profesionales como normales que ayudan en la vida cotidiana [3].

### 1. Creación de la base de datos en tiempo real en Firebase

Para el desarrollo de esta aplicación se creó la base de datos en tiempo real en firebase, ingresamos a la página principal de Firebase con nuestra cuenta de Google *Ilustración 1*, donde

crearemos nuestro proyecto con Realtime Database



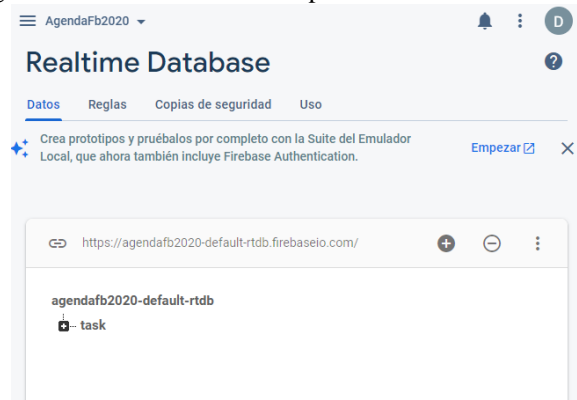
*Ilustración 1. Página Firebase*

Luego de haber creado el proyecto añadimos una aplicación la cual nos dará la secuencia de comandos que deben ser colocados en nuestro proyecto de Ionic para la conexión con nuestra base de datos en Firebase, estos comandos deben ser copiados.

```
<script>
// Your web app's Firebase configuration
// For Firebase JS SDK v7.20.0 and later, measurementId is optional
var firebaseConfig = {
  apiKey: "AIzaSyDYsHoNYt32pMUTExMoof2kdxBFwMrFQIo",
  authDomain: "agendafb2020.firebaseio.com",
  databaseURL: "https://agendafb2020-default-rtdb.firebaseio.com",
  projectId: "agendafb2020",
  storageBucket: "agendafb2020.appspot.com",
  messagingSenderId: "990541511767",
  appId: "1:990541511767:web:0974919fb2d569caa3b712",
  measurementId: "G-3CND13PQ6"
};
// Initialize Firebase
```

*Ilustración 2. Código de Firebase*

Finalmente crearemos una base de datos en tiempo real en la sección de Realtime Database, seleccionamos las opciones según lo que queremos y crearemos la base de datos en donde se ingresará la información de la aplicación *Ilustración 3*.



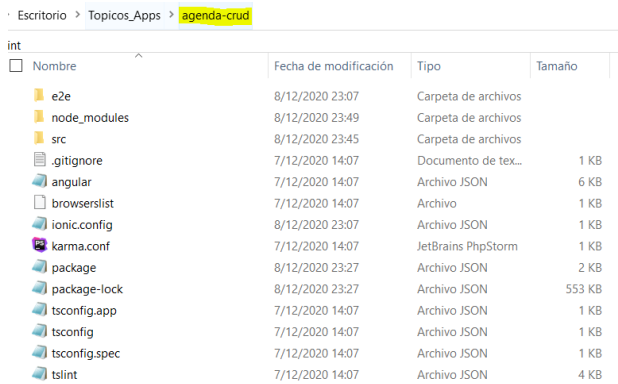
*Ilustración 3. Base de datos en Firebase*

### 2. Creación de proyecto en Ionic

Para crear el respectivo proyecto debemos tener una carpeta y dentro de ella abrir una terminal del sistema, aquí mandaremos a correr el siguiente código: `ionic start name-proyect blank --`

*type=angular*, con esto tendremos nuestro proyecto en blanco creado. A partir de este proyecto crearemos nuestra aplicación, primero realizamos la conexión con firebase, para esto necesitamos instalar esta dependencia con el siguiente comando: *npm install firebase @angular/fire --save*, y una vez que termine de instalar, podremos empezar a crear nuestro App.

Escritorio > Tópicos\_Apps > agenda-crud



Nombre	Fecha de modificación	Tipo	Tamaño
e2e	8/12/2020 23:07	Carpeta de archivos	
node_modules	8/12/2020 23:49	Carpeta de archivos	
src	8/12/2020 23:45	Carpeta de archivos	
.gitignore	7/12/2020 14:07	Documento de texto	1 KB
angular	7/12/2020 14:07	Archivo JSON	6 KB
browserslist	7/12/2020 14:07	Archivo	1 KB
ionic.config	8/12/2020 23:07	Archivo JSON	1 KB
karma.conf	7/12/2020 14:07	JetBrains PhpStorm	1 KB
package	8/12/2020 23:27	Archivo JSON	2 KB
package-lock	8/12/2020 23:27	Archivo JSON	553 KB
tsconfig.app	7/12/2020 14:07	Archivo JSON	1 KB
tsconfig	7/12/2020 14:07	Archivo JSON	1 KB
tsconfig.spec	7/12/2020 14:07	Archivo JSON	1 KB
tslint	7/12/2020 14:07	Archivo JSON	4 KB

Ilustración 4. Carpeta del proyecto

### 3. Creación de Pantallas

Para nuestra App tendremos dos pantallas, una llamada REGISTROS donde se tendrá todos los registros de las notas, deberes, noticias, etc. que se encuentren en la base de datos, y otra pantalla llamada AÑADIR, en la cual se realizará el registro de alguna tarea, en esta segunda pantalla se tendrá que llenar los campos de Tarea, Descripción y escoger una Fecha.

Una vez creado el proyecto debemos ingresar en el mismo para instalar el módulo de Firebase en el proyecto con el comando *"npm install firebase @angular/fire --save"*, hecha la instalación debemos configurar las claves proporcionadas por Firebase dentro de nuestro proyecto en los archivos *environment.ts* y *environment.prod.ts*.

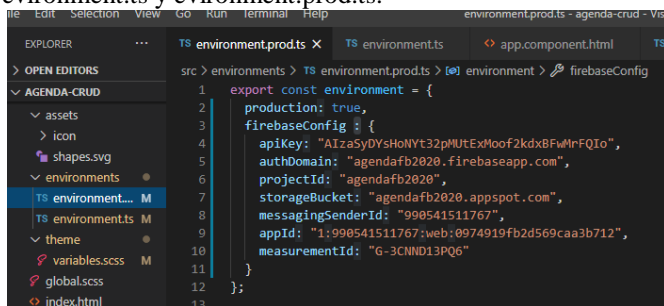


Ilustración 5. Comandos Firebase

Para completar la configuración con Firebase debemos añadir los servicios de Firebase y el objeto de entorno dentro del archivo *app.module.ts*

```

p > TS app.module.ts > AppModule
import { AppComponent } from './app.component';
import { AppRoutingModule } from './app-routing.module';

//firebase librerías
import {AngularFireModule} from '@angular/fire';
import {AngularFireAuthModule} from '@angular/fire/auth';
import {AngularFireDatabaseModule} from '@angular/fire/database';
import {AngularFireStorageModule} from '@angular/fire/storage';

//environment
import {environment} from '../environments/environment';

@NgModule({
  declarations: [AppComponent],
  entryComponents: [],
  imports: [
    BrowserModule,
    IonicModule.forRoot(),
    AppRoutingModule,
    AngularFireModule.initializeApp(environment.firebaseConfig),
    AngularFireAuthModule,
    AngularFireDatabaseModule,
    AngularFireStorageModule
  ],
  providers: [
    StatusBar,
    SplashScreen,
    { provide: RouteReuseStrategy, useClass: IonicRouteStrategy }
  ],
  bootstrap: [AppComponent]
})
export class AppModule {}

```

Ilustración 6. Configuración app.module

Realizada la configuración con Firebase empezamos con el desarrollo de la aplicación, primero creando 2 páginas que servirán de ayuda en la navegación de la aplicación, la primera página es "register" la cual nos permite crear registros de nuevas tareas, donde podremos colocar nombre, descripción y una fecha de la misma, también creamos la página "add" en la cual se mostrará cada tarea con su respectiva información y contará con un botón para poder eliminar las mismas.

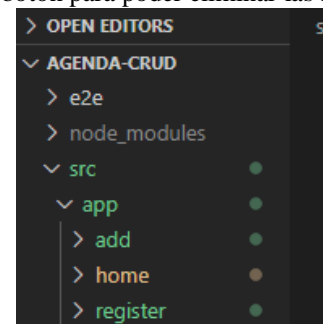


Ilustración 7. Páginas creadas

Configuramos las rutas de las páginas creadas en *app-routing.ts* y agregamos *ion-router-outlet* para activar los elementos de navegación.

```

app > TS app-routing.modules.ts > ...
import { NgModule } from '@angular/core';
import { PreloadAllModules, RouterModule, Routes } from '@angular/router';

const routes: Routes = [
  {
    path: 'home',
    loadChildren: () => import('../home/home.module').then( m => m.HomePageModule)
  },
  {
    path: '',
    redirectTo: 'home',
    pathMatch: 'full'
  },
  {
    path: 'register',
    loadChildren: () => import('../register/register.module').then( m => m.RegisterPageModule)
  },
  {
    path: 'add',
    loadChildren: () => import('../add/add.module').then( m => m.AddPageModule)
  },
];

```

*Ilustración 8. Configuración rutas*

Continuamos creando el CRUD de nuestra aplicación donde utilizamos el comando “ionic generate service shared/task”, esto nos creara una carpeta con diferentes componentes para el CRUD, configuramos el código del archivo task.ts dentro de la carpeta shared donde se definen las operaciones de CRUD con Firebase.

```

shared > TS task.service.ts > taskService > updateBooking
// Create
createBooking(task: Task) {
  return this.bookingListRef.push({
    name: task.name,
    description: task.description,
    date: task.date
  })
}

// Get Single
getBooking(id: string) {
  this.bookingRef = this.db.object('/task/' + id);
  return this.bookingRef;
}

// Get List
getBookingList() {
  this.bookingListRef = this.db.list('/task');
  return this.bookingListRef;
}

// Update
updateBooking(id, task: Task) {
  return this.bookingRef.update({
    name: task.name,
    description: task.description,
    date: task.date
  })
}

```

*Ilustración 9. Configuración CRUD*

Continuamos con las configuraciones de cada página que se crearon anteriormente donde modificamos los archivos “page.ts” y “page.html” para la llamada de las diferentes operaciones CRUD que se crearon anteriormente y el diseño de las cada una de las páginas de nuestra aplicación. Para la configuración de estos archivos se debe tener en cuenta que se realizara realizar en cada una de estas páginas para hacer las llamadas a las diferentes operaciones del CRUD.

```

app > register > TS register.page.ts > ...
@Component({
  selector: 'app-register',
  templateUrl: './register.page.html',
  styleUrls: ['./register.page.scss'],
})

export class RegisterPage implements OnInit {
  bookingForm: FormGroup;

  constructor(
    private tskService: TaskService,
    private router: Router,
    public fb: FormBuilder
  ) { }

  ngOnInit() {
    this.bookingForm = this.fb.group({
      name: [''],
      description: [''],
      date: ['']
    })
  }

  formSubmit() {
    if (!this.bookingForm.valid) {
      return false;
    } else {
      this.tskService.createBooking(this.bookingForm.value).then(res => {
        console.log(res)
        this.bookingForm.reset();
        this.router.navigate(['/home']);
      })
    }
  }
}

```

*Ilustración 10. Configuración register.page.ts*

Para la configuración de la visualización primero configuramos la página para realizar registros nuevos donde contiene secciones para ingresar el nombre del asunto, una descripción y una fecha para ese asunto. Además, cuenta con un botón que permiten agregar todo lo que se mencionó anteriormente en la base de datos.

```

<!-- Register -->
<ion-header>
  <ion-title>Registrar</ion-title>
</ion-header>

<ion-content>
  <ion-list lines="full" style="height: 100%; background-image: url('https://cdn.pixabay.com/photo/2020/05/17/10/04/pen-518');>
    <form [formGroup]="bookingForm" (ngSubmit)="formSubmit()">
      <ion-item>
        <ion-label position="floating">Tarea</ion-label>
        <ion-input formControlName="name" type="text" required ></ion-input>
      </ion-item>
      <ion-item>
        <ion-label position="floating">Descripción</ion-label>
        <ion-input formControlName="description" type="text" required ></ion-input>
      </ion-item>
      <ion-item>
        <ion-label position="floating">Fecha</ion-label>
        <ion-input formControlName="date" type="date" required style="color: #lightslategray">
      </ion-item>
      <ion-row>
        <ion-col>
          <ion-button type="submit" shape="full" expand="block" style="width: max-content;">Agregar</ion-button>
        </ion-col>
      </ion-row>
    </form>
  </ion-list>
</ion-content>

```

*Ilustración 11. Configuración page.html*

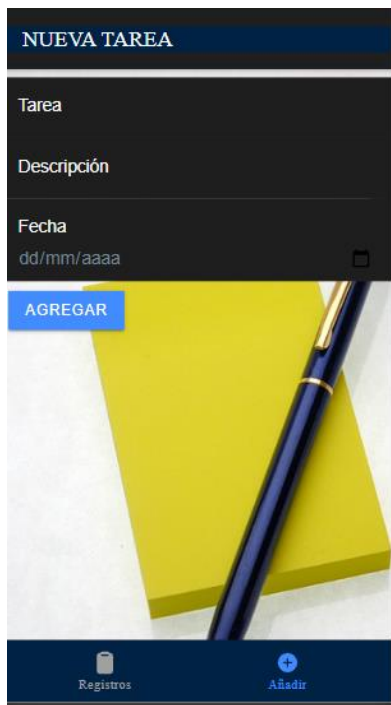


Ilustración 12. Visualización página register

La página home cuenta con secciones que se agregaran automáticamente cada vez que se ingrese un nuevo registro a la aplicación, estos registros se guardaran en la base de datos y en estas secciones se podrá ver toda la información de esa tarea y también se podrá eliminar la misma.

```

<ion-header style="background-color: #002244; color: white; padding: 5px; font-family: sans-serif; font-weight: bold;">
  <ion-title>
    MI AGENDA
  </ion-title>
</ion-header>

<ion-content>
  <ion-list class="list-item" style="background-color: #f0f0f0; padding: 10px;">
    <ion-list-header class="list-item" style="background-color: #002244; color: white; padding: 5px; font-weight: bold;">
      TAREAS
    </ion-list-header>

    <ion-item *ngFor="let booking of Bookings" class="user-list" style="margin-bottom: 10px;">
      <ion-label>
        <div>
          <ion-icon name="person" style="color: #002244; margin-right: 10px;"/> {{booking.name}}
        </div>
        <div>
          <ion-icon name="book" style="color: #002244; margin-right: 10px;"/> {{booking.description}}
        </div>
        <div>
          <ion-icon name="calendar" style="color: #002244; margin-right: 10px;"/> {{booking.date}}
        </div>
      </ion-label>

      <div class="item-note" item-end>
        <button ion-button clear (click)="deleteBooking(booking.$key)" style="color: red; font-size: 0.8em;">
          <ion-icon name="trash" style="width: 1em; height: 1em;"/>
        </button>
      </div>
    </ion-item>
  </ion-list>
</ion-content>

```

Ilustración 13. Configuración home.page.html

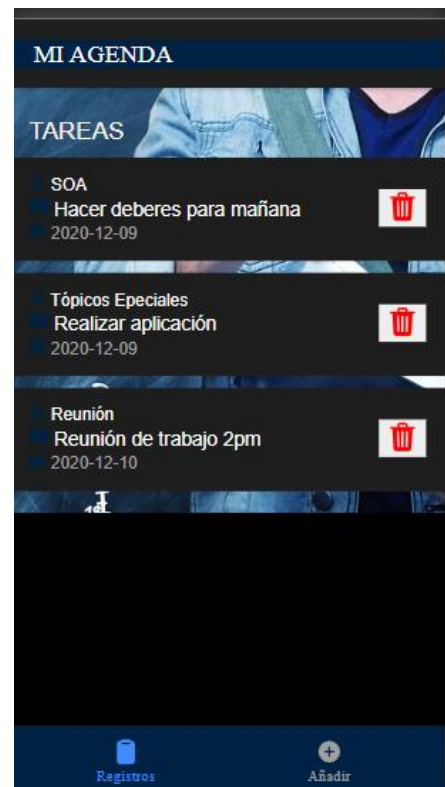


Ilustración 14. Visualización página home

Para finalizar podremos observar cómo se guardan los datos en firebase de manera instantánea al momento de utilizar la aplicación.

✦ Crea prototipos y pruébalos por completo con la Suite del Emulador Local, con



Ilustración 15. Base de datos Firebase

#### 4. Desarrollo de icono personalizado y splash screen

Para el desarrollo del icono y el splash screen personalizado, primero se debe construir la aplicación a APK para poder modificar los iconos en los archivos que se crean al momento de construir la APK.

e2e	8/12/2020 23:07	Carpeta de archivos
node_modules	9/12/2020 19:11	Carpeta de archivos
platforms	9/12/2020 19:08	Carpeta de archivos
plugins	9/12/2020 19:10	Carpeta de archivos
resources	9/12/2020 19:18	Carpeta de archivos
src	8/12/2020 23:45	Carpeta de archivos
www	9/12/2020 19:28	Carpeta de archivos
.gitignore	7/12/2020 14:07	Documento de tex...
angular	9/12/2020 19:10	Archivo JSON
browserslist	7/12/2020 14:07	Archivo
config	9/12/2020 19:27	Documento XML
ionic.config	9/12/2020 19:06	Archivo JSON
karma.conf	7/12/2020 14:07	JetBrains PhpStorm
package	9/12/2020 19:10	Archivo JSON
package-lock	9/12/2020 19:09	Archivo JSON
tsconfig.app	7/12/2020 14:07	Archivo JSON
tsconfig	7/12/2020 14:07	Archivo JSON
tsconfig.spec	7/12/2020 14:07	Archivo JSON
tslint	7/12/2020 14:07	Archivo JSON

Ilustración 16. Archivos generados por APK

En el archivo resources se encuentra el icono y el splash screen por defecto que se crean al momento de generar la APK, estas 2 imágenes las cambiamos por nuestras propias imágenes las cuales deben tener una dimensión de 1024 x 1024 para el icono y de 2732 x 2732 para el screen splash, en este caso se creó 2 imágenes con Canva [4], hecho esto se cargó las nuevas imágenes al archivo.

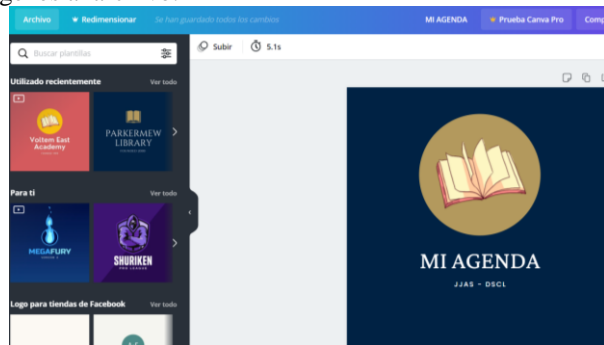


Ilustración 17. Creación de imagen en Canva

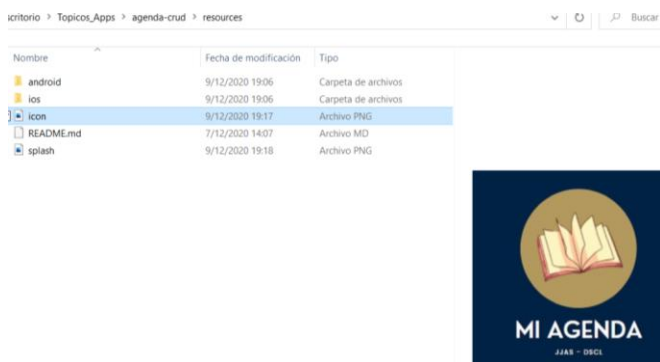


Ilustración 18. Cambio de imágenes

A continuación, se debe ejecutar el comando “ionic cordova resources” el cual nos crea todos los aspectos y tamaños de nuestras imágenes ingresadas automáticamente en los dispositivos Android y iOS, generando los diferentes tipos de vistas.

```
C:\Windows\System32\cmd.exe
C:\Users\jhoel\Desktop\Temas_Apps\agenda-crud>ionic cordova resources
> cordova-res.cmd
[cordova-res] Generated 18 resources for Android
[cordova-res] WARN: Source icon "resources/icon.png" contains alpha channel, g
generated icons for iOS will not.
[cordova-res] Apple recommends avoiding transparency. See the App Icon Human Int
interface Guidelines[1] for details. Any transparency in your icon will be filled in
with white.
[cordova-res] [1]: https://developer.apple.com/design/human-interface-guidelines
/ios/icons-and-images/app-icon/
[cordova-res] Generated 47 resources for iOS
[cordova-res] Wrote to config.xml
```

Ilustración 19. Comando para actualizar recursos

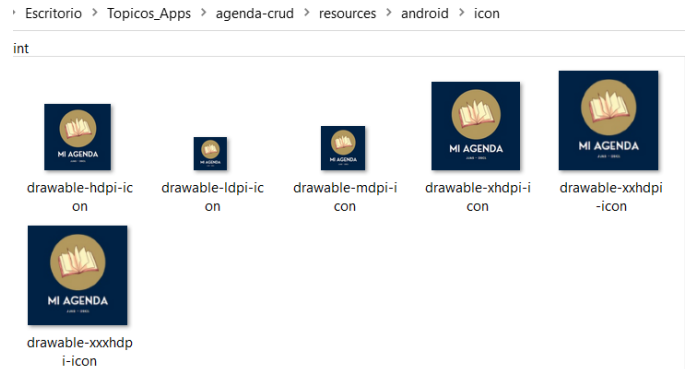


Ilustración 20. Iconos generados

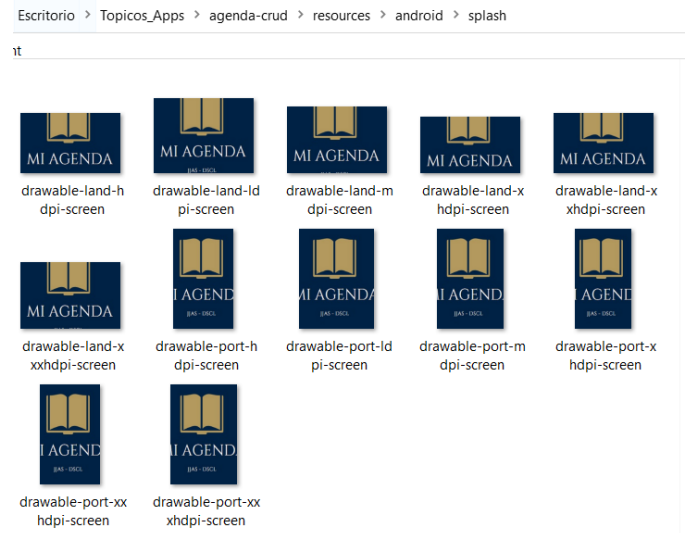
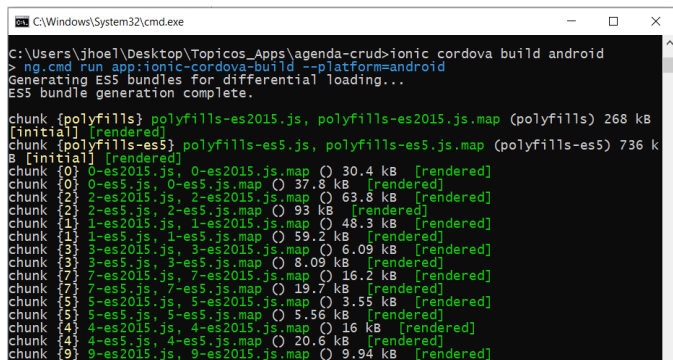


Ilustración 21. Splash Screen generados

Para finalizar se debe reconstruir nuevamente la APK con el mismo código anteriormente ingresado para generar la misma, dando así una nueva APK con los cambios realizados, como adicional también se cambió el nombre de nuestra APK en el archivo conf.xml.

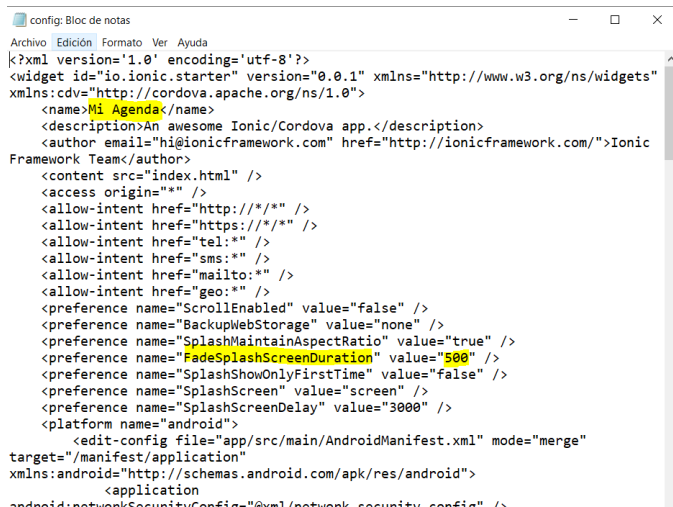




```
C:\Users\jhoel\Desktop\Temas_Appls\agenda-crud>ionic cordova build android
> ng.cmd run app:ionic-cordova-build --platform=android
Generating ES5 bundles for differential loading...
ES5 bundle generation complete.

chunk {polyfills} polyfills-es2015.js, polyfills-es2015.js.map (polyfills) 268 kB
[initial] [rendered]
chunk {polyfills-es5} polyfills-es5.js, polyfills-es5.js.map (polyfills-es5) 736 kB
[initial] [rendered]
chunk {0} 0-es2015.js, 0-es2015.js.map 30.4 kB [rendered]
chunk {0} 0-es5.js, 0-es5.js.map 37.8 kB [rendered]
chunk {2} 2-es2015.js, 2-es2015.js.map 63.8 kB [rendered]
chunk {2} 2-es5.js, 2-es5.js.map 93 kB [rendered]
chunk {1} 1-es2015.js, 1-es2015.js.map 48.3 kB [rendered]
chunk {1} 1-es5.js, 1-es5.js.map 59.2 kB [rendered]
chunk {3} 3-es2015.js, 3-es2015.js.map 6.09 kB [rendered]
chunk {3} 3-es5.js, 3-es5.js.map 8.09 kB [rendered]
chunk {7} 7-es2015.js, 7-es2015.js.map 16.2 kB [rendered]
chunk {7} 7-es5.js, 7-es5.js.map 19.7 kB [rendered]
chunk {5} 5-es2015.js, 5-es2015.js.map 3.55 kB [rendered]
chunk {5} 5-es5.js, 5-es5.js.map 5.56 kB [rendered]
chunk {4} 4-es2015.js, 4-es2015.js.map 16 kB [rendered]
chunk {4} 4-es5.js, 4-es5.js.map 20.6 kB [rendered]
chunk {9} 9-es2015.js, 9-es2015.js.map 9.94 kB [rendered]
```

Ilustración 22. Reconstrucción de la APK



```
config: Bloc de notas
Archivo Edición Formato Ver Ayuda
<?xml version='1.0' encoding='utf-8'?>
<widget id="io.ionic.starter" version="0.0.1" xmlns="http://www.w3.org/ns/widgets"
xmlns:cdv="http://cordova.apache.org/ns/1.0">
  <name>Mi Agenda</name>
  <description>An awesome Ionic/Cordova app.</description>
  <author email="hi@ionicframework.com" href="http://ionicframework.com/">Ionic
Framework Team</author>
  <content src="index.html" />
  <access origin="*" />
  <allow-intent href="http://*/*" />
  <allow-intent href="https://*/*" />
  <allow-intent href="tel:*" />
  <allow-intent href="sms:*" />
  <allow-intent href="mailto:*" />
  <allow-intent href="geo:*" />
  <preference name="ScrollEnabled" value="false" />
  <preference name="BackupWebStorage" value="none" />
  <preference name="SplashMaintainAspectRatio" value="true" />
  <preference name="FadeSplashScreenDuration" value="500" />
  <preference name="SplashShowOnlyFirstTime" value="false" />
  <preference name="SplashScreen" value="screen" />
  <preference name="SplashScreenDelay" value="3000" />
  <platform name="android">
    <edit-config file="app/src/main/AndroidManifest.xml" mode="merge"
target="/manifest/application"
xmlns:android="http://schemas.android.com/apk/res/android">
      <application
android:networkSecurityConfig="@xml/network_security_config" />
    </edit-config>
  </platform>
</widget>
```

Ilustración 23. Configuración archivo conf.xml

El APK está cargada en el repositorio de GitHub para su descarga.



Ilustración 24. Repositorio en GitHub

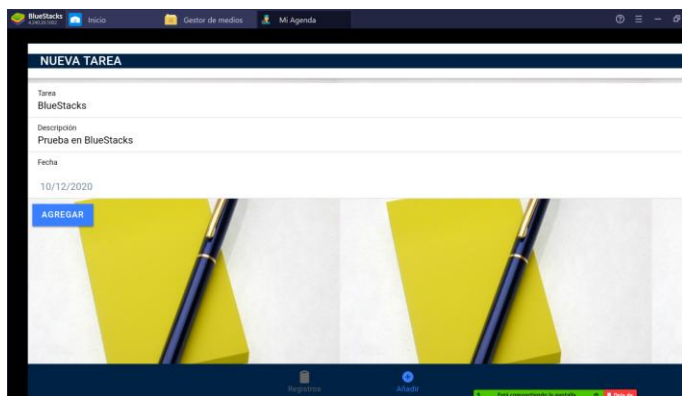


Ilustración 25. Visualización de la aplicación en BlueStacks

### III. CONCLUSIONES

Las funciones desarrolladas para realizar las actividades de añadir y eliminar, deben constar con las variables correctas, del mismo modo estas variables deben estar bien definidas y con el formato correspondiente, de no ser así, la presentación de estos datos puede ser errónea, o la ejecución de la aplicación puede llegar a fallar.

El diseño de la aplicación debe realizarse tomando en cuenta las variables de dispositivos en los que la App puede ser usada, en los diferentes tipos de fondos de colores y demás, para lograr que la aplicación sea entendible para el mayor número de usuarios posible.

Uno de los problemas suscitados en el desarrollo de la apk, fue el no tener los complementos instalados de Cordova, es por eso que se debe verificar que las dependencias se encuentren correctamente para poder ejecutar la aplicación y obtener el ejecutable.

Se debe tener en cuenta las dimensiones de las imágenes, tanto como para el icono personalizado, como para desarrollar el Splash Screen, ya que puede traer problemas en presentación, ejecución y/o diseño.

### IV. REFERENCIAS

- [1] A. Gonzalez, «Swapps,» 04 Abril 2017. [En línea]. Available: <https://swapps.com/es/blog/desarrollo-de-aplicaciones-moviles-con-ionic/>. [Último acceso: 09 Diciembre 2020].
- [2] «Firebase,» [En línea]. Available: <https://firebase.google.com/>. [Último acceso: 09 Diciembre 2020].
- [3] «Softcorp,» [En línea]. Available: <https://servissoftcorp.com/definicion-y-como-funcionan-las-aplicaciones-moviles/>. [Último acceso: 21 Noviembre 2020].
- [4] «Canva,» [En línea]. Available: [https://www.canva.com/es\\_419/](https://www.canva.com/es_419/). [Último acceso: 09 Diciembre 2020].
- [5] C. D. Amagua Jhoel, «GitHub,» 09 Diciembre 2020. [En línea]. Available: <https://github.com/JhoelAmagua/Topicos.git>. [Último acceso: 09 12 2020].