UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Título del Proyecto:

Fray Luis AAA

Autor:

Hernáiz Izquierdo, Ignacio

Director:

Puga Sabio, Gonzalo

TESINA PARA LA OBTENCIÓN DEL TÍTULO DE:

Máster en Desarrollo de Aplicaciones sobre Dispositivos Móviles

Septiembre del 2017

Contenido

Introducción

Descripción del problema

La idea de crear esta aplicación surge de mantener informados a los usuarios de una liga privada de futbol sala.

Me encargo desde hace un año a gestionar dicha liga y después de adquirir los conocimientos a través del master de aplicaciones móviles que he realizado vi la oportunidad de darle una vuelta a la liga y de "modernizarla" introduciéndola en un app para que los participantes de la misma tenga más a mano la información de la misma.

Objetivos

El objetivo principal es el desarrollo de una aplicación para dispositivos móviles Android que permita tener informados a los usuarios de una liga privada de fútbol sala.

Con esta aplicación se informará al usuario de todo lo relevante a la liga, resultados, calendarios, estadísticas, localización del pabellón donde se realizan los partidos y un contacto directo desde la aplicación para ponerse en comunicación con la organización sobre dudas o problemas que surjan al equipo.

Se incluyeron notificaciones para que el usuario esté atento de los diferentes cambios así como de las noticias más relevantes. Se informará de cuando haya cambios en las estadísticas e informando de los horarios días antes de que comience la jornada.

Motivación

Mi motivación como bien indicaba en las líneas de arriba es que hace ya un año me hice cargo de la organización de esta liga en la cual llevo participando desde el año 2005. Soy antiguo alumno del colegio Fray Luis de León de Madrid.

El poder tener toda la información en la palma de la mano y en tiempo real gracias al sistema de base de datos de Firebase y estar informado de los cambios gracias a las notificaciones me pareció muy motivante y me lancé en este proyecto (mi primera app).

Informar a los usuarios de esta manera me pareció más cómoda, cosa que siendo yo usuario de la misma agradezco.

Tecnologías utilizadas

Las tecnologías que se han utilizado en el proyecto han sido las siguientes:

- **Android**: La aplicación ha sido desarrollada para dispositivos Android y se ha utilizado el IDE de Android Studio para la codificación de la misma.
- **Firebase**: La aplicación se apoya en los siguientes módulos:

  -**Firebase Realtime Database**: Almacena y sincroniza datos con nuestra base de datos NoSQL alojada en la nube. Los datos se sincronizan con todos los clientes tiempo real y se mantienen disponibles cuando tu app está sin conexión.

  -**Firebase Cloud Messaging (FCM):** Solución de mensajería multiplataforma que te permite envíar mensajes de forma segura y gratuita. Enviar mensajes de notificación al usuario.

- **GPS**: Muestra la ubicación del campo donde se disputan los partidos.
- **Google Maps API v2** para la integración de los mapas en la aplicación.
- **Git** para el control de versiones del código y repositorio.
- **SourceTree** como herramienta para gestionar las ramas de git y sus correspondientes commits, push y merges.

Arquitectura de la aplicación

Esquema del diseño

La aplicación se basa sobre un menú principal desde el cual accederemos a los diferentes submenús e informaciones de la liga.

La clase MainActivity.java contiene las diferentes llamadas a las actividades que nos darán que precisemos e incluye la función onStart() para la visualización del cuadro de dialogo cuando recibimos notificaciones.

- **MainActivity.java**

```
public class MainActivity extends AppCompatActivity {

    private static MainActivity current;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_2);
    }

    public void lanzarAcercaDe(View view){
        Intent i = new Intent(this, AcercaDeActivity.class);
        startActivity(i);
    }

    public void lanzarCalendario(View view) {
        Intent i = new Intent(this, Calendario.class);
        startActivity(i);
    }

    public void lanzarEstadisticas(View view) {
        Intent i = new Intent(this, Estadisticas.class);
        startActivity(i);
    }

    public void lanzarContacto(View view) {
        Intent i = new Intent(this, Contacto.class);
        startActivity(i);
    }

   public void lanzarClasificacion(View view) {
       Intent i = new Intent(this, Info.class);
       startActivity(i);
   }

   @Override
   protected void onStart() {
       super.onStart();
       current=this;
   }
   public static MainActivity getCurrentContext() { return current; }
```

A partir de aquí realizó las clases para las diferentes actividades las cuales se dan de alta en el AndroidManifest.xml.

A continuación se detallan las diferentes clases incluidas en las diferentes opciones de menú.

- 

  Se trata del botón de información de la liga donde podemos encontrar los resultados de las jornadas y la clasificación.

  NOTA: *Mostraré como ejemplo las clases de Info.java del menú, debido a que las demás son similares donde solo cambia los nombres de las clases. Todos los adaptadores se han realizado con RecyclerViews.*

  Las clases utilizadas han sido las siguientes:

  - ***Info.java***, dentro de esta clase nos encontramos con las funciones que nos lanzaran las actividades de la clasificación y los resultados de las jornadas.

```
public class Info extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.info);
    }

    public void lanzarClasificacion(View view) {
        Intent i = new Intent(this, Clasificacion.class);
        startActivity(i);
    }

    public void lanzarResultados(View view) {
        Intent i = new Intent(this, Resultado.class);
        startActivity(i);
    }
}
```

- ***Clasificacion.java***, lanza la clasificación. Esta clase tiene una clase asociada que se llama ***ClasificacionTabla.java***, la cual tiene los diferentes campos que tiene el JSON almacenado en Firebase. Esta clase es utilizada en su adaptador ***AdaptadorClasificacion.java***.

```
public class Clasificacion  extends AppCompatActivity {

    private RecyclerView recyclerView;
    private RecyclerView.LayoutManager layoutManager;
    private DatabaseReference clasificacionReference;
    private FirebaseRecyclerAdapter adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.clasificacion2);
        recyclerView = (RecyclerView) findViewById(R.id.recycler_view);
        Aplicacion app = (Aplicacion) getApplicationContext();
        clasificacionReference = app.getClasificacionReference();
        adapter = new AdaptadorClasificacion(this,clasificacionReference);
        layoutManager = new LinearLayoutManager(this);
        recyclerView.setLayoutManager(layoutManager);
        recyclerView.setAdapter(adapter);
    }
}
```

```java
public class ClasificacionTabla {

    private String equipo;
    private String jugados;
    private String ganados;
    private String empatados;
    private String perdidos;
    private String golesfavor;
    private String golescontra;
    private String puntos;

    public ClasificacionTabla() {}

    public ClasificacionTabla(String equipo, String jugados, String ganados, String empatados,
                              String perdidos, String golesfavor, String golescontra, String puntos) {
        this.equipo = equipo;
        this.jugados = jugados;
        this.ganados = ganados;
        this.empatados = empatados;
        this.perdidos = perdidos;
        this.golesfavor = golesfavor;
        this.golescontra = golescontra;
        this.puntos = puntos;
    }

    public String getEquipo() { return equipo; }

    public void setEquipo(String equipo) { this.equipo = equipo; }

    public String getJugados() { return jugados; }

    public void setJugados(String jugados) { this.jugados = jugados; }

    public String getGanados() { return ganados; }

    public void setGanados(String ganados) { this.ganados = ganados; }

    public String getEmpatados() { return empatados; }

    public void setEmpatados(String empatados) { this.empatados = empatados; }

    public String getPerdidos() { return perdidos; }

    public void setPerdidos(String perdidos) { this.perdidos = perdidos; }

    public String getGolesfavor() { return golesfavor; }

    public void setGolesfavor(String golesfavor) { this.golesfavor = golesfavor; }

    public String getGolescontra() { return golescontra; }

    public void setGolescontra(String golescontra) { this.golescontra = golescontra; }

    public String getPuntos() { return puntos; }

    public void setPuntos(String puntos) { this.puntos = puntos; }
}
```

```
public class AdaptadorClasificacion extends FirebaseRecyclerAdapter<ClasificacionTabla,AdaptadorClasificacion.ViewHolder> {

    private LayoutInflater inflador;
    protected DatabaseReference clasificacionReference;
    private Context contexto;

    public AdaptadorClasificacion(Context context, DatabaseReference reference) {
        super(ClasificacionTabla.class, R.layout.elemento_clasificacion,ViewHolder.class,reference);
        inflador = (LayoutInflater) context
                .getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.clasificacionReference = reference;
        this.contexto = context;
    }

    @Override
    public AdaptadorClasificacion.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View view = inflador.inflate(R.layout.elemento_clasificacion, parent, false);
        return new AdaptadorClasificacion.ViewHolder(view);
    }

    @Override
    protected void populateViewHolder(final AdaptadorClasificacion.ViewHolder holder, ClasificacionTabla item, int position){
        String equipo = item.getEquipo();
        String jugados = item.getJugados();
        String ganados = item.getGanados();
        String empatados = item.getEmpatados();
        String perdidos = item.getPerdidos();
        String golesfavor = item.getGolesfavor();
        String golescontra = item.getGolescontra();
        String puntos = item.getPuntos();
        if (equipo.equals("EQUIPO")) {
            holder.equipo.setBackgroundColor(Color.CYAN);
            holder.equipo.setText(equipo);
            holder.jugados.setBackgroundColor(Color.YELLOW);
            holder.jugados.setText(jugados);
            holder.ganados.setBackgroundColor(Color.GREEN);
            holder.ganados.setText(ganados);
            holder.empatados.setBackgroundColor(Color.LTGRAY);
            holder.empatados.setText(empatados);
            holder.perdidos.setBackgroundColor(Color.RED);
            holder.perdidos.setText(perdidos);
            holder.golesFavor.setBackgroundColor(Color.GRAY);
            holder.golesFavor.setText(golesfavor);
            holder.golesContra.setBackgroundColor(Color.WHITE);
            holder.golesContra.setText(golescontra);
            holder.puntos.setBackgroundColor(Color.YELLOW);
            holder.puntos.setText(puntos);
        }
        else{
            holder.equipo.setText(equipo);
            holder.jugados.setText(jugados);
            holder.ganados.setText(ganados);
            holder.empatados.setText(empatados);
            holder.perdidos.setText(perdidos);
            holder.golesFavor.setText(golesfavor);
            holder.golesContra.setText(golescontra);
            holder.puntos.setText(puntos);
        }

    }

    public class ViewHolder extends RecyclerView.ViewHolder {

        public TextView equipo;
        public TextView jugados;
        public TextView ganados;
        public TextView empatados;
        public TextView perdidos;
        public TextView golesFavor;
        public TextView golesContra;
        public TextView puntos;

        public ViewHolder(View itemView) {
            super(itemView);
            equipo = (TextView) itemView.findViewById(R.id.equipo);
            jugados = (TextView) itemView.findViewById(R.id.jugados);
            ganados = (TextView) itemView.findViewById(R.id.ganados);
            empatados = (TextView) itemView.findViewById(R.id.empatados);
            perdidos = (TextView) itemView.findViewById(R.id.perdidos);
            golesFavor = (TextView) itemView.findViewById(R.id.golesFavor);
            golesContra = (TextView) itemView.findViewById(R.id.golesContra);
            puntos = (TextView) itemView.findViewById(R.id.puntos);

        }
    }
}
```

- ***Resultado.java***, lanza los resultados de las jornadas. Al igual que la clase anterior, tiene una clase asociada ***ResultadoTabla.java*** que se utiliza en su adaptador asociado ***AdaptadorResultado.java***. (Como las imágenes adjuntadas arriba).

➢ 

Se trata del botón de las estadísticas de la liga, en el cual encontramos la clasificación de goleadores y de los porteros menos goleados de la liga.

Las clases utilizadas han sido las siguientes:

- ***Estadisticas.java***, dentro de esta clase nos encontramos las funciones que nos lanzaran las actividades de los goleadores y de los porteros menos goleados.

- ***Goleador.java***, lanza los goleadores. Esta clase tiene una clase asociada que se llama ***GoleadorTabla.java***, la cual tiene los diferentes campos que tiene el JSON almacenado en Firebase. Esta clase es utilizada en su adaptador ***AdaptadorGoleador.java***.

- ***Zamora.java***, lanza a los porteros menos goleados. Esta clase tiene una clase asociada que se llama ***ZamoraTabla.java***, la cual tiene los diferentes campos que tiene el JSON almacenado en Firebase. Esta clase es utilizada en su adaptador ***AdaptadorZamora.java***.

➢ 

Se trata del botón de los horarios de las jornadas de liga y de copa.

Las clases utilizadas han sido las siguientes:

- ***Calendario.java***, en la cual nos encontramos las funciones que nos lanzan las actividades de las jornadas de liga y de copa respectivamente.

- ***JornadaListaLiga.java***, en esta clase nos encontramos con las diferentes funciones que lanzaran la actividad que nos visualizará la jornada de liga. Para ello se utiliza la función intent.putExtra donde les pasamos los parámetros dependiendo de la jornada a mostrar.

```
public void onClick1(View view) {
    Intent i = new Intent(this, CalendarioLiga.class);
    i.putExtra("jornada","jornada1");
    i.putExtra("posicion","0");
    startActivity(i);
}
```

- ***CalendarioLiga.java***, lanza la jornada de liga seleccionada. Esta clase tiene una clase asociada que se llama ***Jornada.java***, la cual tiene los diferentes campos que tiene el JSON almacenado en Firebase. Esta clase es utilizada en su adaptador ***AdaptadorCalendario.java***. Se realiza la búsqueda de los datos solicitados recogiendo los valores introducidos a través del putExtra :

```
jornada = getIntent().getExtras().getString("jornada");
posicion = getIntent().getExtras().getString("posicion");
Aplicacion app = (Aplicacion) getApplicationContext();
calendarioLigaReference = app.getCalendarioLigaReference();
DatabaseReference ref = calendarioLigaReference.getRef().child(posicion).child(jornada);
```

- ***JornadaListaCopa.java***, en esta clase nos encontramos con las diferentes funciones que lanzaran la actividad que nos visualizará la jornada de liga. Para ello se utiliza la función intent.putExtra donde les pasamos los parámetros dependiendo de la jornada a mostrar.

- ***CalendarioCopa.java***, lanza la jornada de liga seleccionada. Esta clase tiene una clase asociada que se llama ***Jornada.java***, la cual tiene los diferentes campos que tiene el JSON almacenado en Firebase. Esta clase es utilizada en su adaptador ***AdaptadorCalendario.java.*** Se realiza la búsqueda de los datos solicitados recogiendo los valores introducidos a través del putExtra.

➢ 

Última opción del menú principal en la cual encontramos un mapa de cómo llegar a las instalaciones y la posibilidad de mandar un correo a la organización.

Las clases utilizadas han sido las siguientes:

- ***Contacto.java***, que incluye las funciones para mostrar el mapa, el permiso que se solicitará para tener acceso a la ubicación y la función de enviar un mail utilizando una de las opciones que tengamos para enviar mails en nuestro móvil. El mail se manda a una dirección de correo que se ha codificado en el código.

```
public class Contacto  extends FragmentActivity implements
        OnMapReadyCallback {

    private GoogleMap mapa;
    private final LatLng FrayLuis = new LatLng(40.42715031352817, -3.7158421999999973);
    private static final int SOLICITUD_PERMISO_ACCESS_FINE_LOCATION = 1;
    private LocationManager manejador;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.contacto);

        manejador = (LocationManager) getSystemService(LOCATION_SERVICE);
        if(ContextCompat.checkSelfPermission(this, android.Manifest.permission.ACCESS_FINE_LOCATION)
                != PackageManager.PERMISSION_GRANTED) {
            solicitarPermisoUbicacion();
        }
        SupportMapFragment mapFragment = (SupportMapFragment)
                getSupportFragmentManager().findFragmentById(R.id.mapa);
        mapFragment.getMapAsync(this);

    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        GoogleMap mapa = googleMap;
        mapa.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
        mapa.moveCamera(CameraUpdateFactory.newLatLngZoom(FrayLuis, 15));
        mapa.addMarker(new MarkerOptions().position(FrayLuis).title("Fray Luis AAA"));
    }

    void solicitarPermisoUbicacion(){
        ActivityCompat.requestPermissions(this,
                new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
                SOLICITUD_PERMISO_ACCESS_FINE_LOCATION);
    }

    public void lanzarEmail(View view) {
        String[] to = { "nacho_celtic@hotmail.com" };
        enviar(to,"AAA Fray Luis de Leon",
                "Esto es un email enviado desde una app de Android");
    }

    private void enviar(String[] to,String asunto, String mensaje) {
        Intent emailIntent = new Intent(Intent.ACTION_SEND);
        emailIntent.setData(Uri.parse("mailto:"));
        emailIntent.putExtra(Intent.EXTRA_EMAIL, to);
        emailIntent.putExtra(Intent.EXTRA_SUBJECT, asunto);
        emailIntent.putExtra(Intent.EXTRA_TEXT, mensaje);
        emailIntent.setType("message/rfc822");
        startActivity(Intent.createChooser(emailIntent, "Email "));
    }

}
```

- **Clases auxiliares**:
  - Para las notificaciones he utilizado dos clases, ***Dialogo.java*** que extiende de Activity que recoge en contenido del mensaje a mostrar. La otra clase es ***NotificacionesFCMService.java*** que extiende de FirebaseMessagingService y mostraría el mensaje.

```
public class Dialogo extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Bundle extras = getIntent().getExtras();
        if (getIntent().hasExtra("mensaje")) {
            AlertDialog alertDialog = new AlertDialog.Builder(this).create();
            alertDialog.setTitle("Información:");
            alertDialog.setMessage(extras.getString("mensaje"));
            alertDialog.setButton(AlertDialog.BUTTON_NEUTRAL, "CERRAR",
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int which) {
                            dialog.dismiss();
                            finish();
                        }
                    });
            alertDialog.show();
            extras.remove("mensaje");
        }
    }
}
```

```
public class NotificacionesFCMService extends FirebaseMessagingService {

    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {
        if (remoteMessage.getNotification() != null) {
            mostrarDialogo(getApplicationContext(),
                    remoteMessage.getNotification().getBody());
        }
    }
}
```

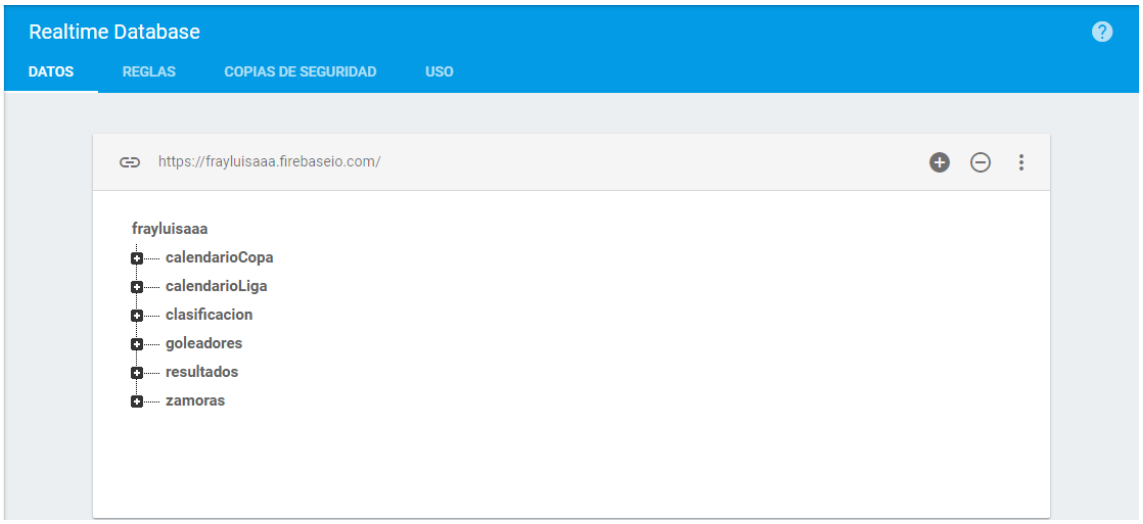
- ***Aplicación.java***, clase importante sobre la que gira la creación de la instancia a la bbdd y sus diferentes referencias a sus child. Incluye la función mostrarDialogo que lanza la actividad Dialogo.java explicada anteriormente para las notificaciones.

```
public class Aplicacion extends Application {

    private final static String GOLEADOR_CHILD = "goleadores";
    private final static String ZAMORAS_CHILD = "zamoras";
    private final static String CALENDARIOLIGA_CHILD = "calendarioLiga";
    private final static String CALENDARIOCOPA_CHILD = "calendarioCopa";
    private final static String CLASIFICACION_CHILD = "clasificacion";
    private final static String RESULTADOS_CHILD = "resultados";

    private static DatabaseReference goleadorReference;
    private static DatabaseReference zamoraReference;
    private static DatabaseReference calendarioLigaReference;
    private static DatabaseReference calendarioCopaReference;
    private static DatabaseReference clasificacionReference;
    private static DatabaseReference resultadosReference;

    private static Context context;

    @Override
    public void onCreate(){
        super.onCreate();
        Aplicacion.context = getApplicationContext();
        FirebaseDatabase database = FirebaseDatabase.getInstance();
        database.setPersistenceEnabled(true);
        goleadorReference = database.getReference(GOLEADOR_CHILD);
        zamoraReference = database.getReference(ZAMORAS_CHILD);
        calendarioLigaReference = database.getReference(CALENDARIOLIGA_CHILD);
        calendarioCopaReference = database.getReference(CALENDARIOCOPA_CHILD);
        clasificacionReference = database.getReference(CLASIFICACION_CHILD);
        resultadosReference = database.getReference(RESULTADOS_CHILD);

    }

    public static Context getAppContext() { return Aplicacion.context; }

    public static DatabaseReference getGoleadorReference() { return goleadorReference; }

    public static DatabaseReference getZamoraReference() { return zamoraReference; }

    public static DatabaseReference getCalendarioLigaReference() { return calendarioLigaReference; }

    public static DatabaseReference getCalendarioCopaReference() { return calendarioCopaReference; }

    public static DatabaseReference getClasificacionReference() { return clasificacionReference; }

    public static DatabaseReference getResultadosReference() { return resultadosReference; }

    static void mostrarDialogo (final Context context, final String mensaje){
        Intent intent = new Intent(context, Dialogo.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        intent.putExtra("mensaje", mensaje);
        context.startActivity(intent);
    }
}
```

Modelo de datos

Esquema de base de datos

El modelo de datos esta realizado con *Firebase Realtime Database*, los datos se almacenan en formato JSON y se sincronizan en tiempo real con cada cliente conectado. Cuando compilas apps multiplataforma con nuestros SDK de iOS, Android y JavaScript, todos tus clientes comparten una instancia de *Realtime Database* y reciben actualizaciones de forma automática con los datos más recientes.



En la imagen se observan las diferentes tablas que componen la base de datos:

- **calendarioCopa**: Almacena los horarios de las diferentes jornadas de copa que tiene la competición.
- **calendarioLiga**: Almacena los horarios de las diferentes jornadas de liga que tiene la competición.
- **clasificación**: Almacena un registro por cada equipo donde se incluyen los partidos jugados, ganados, perdidos, empatados, goles a favor, goles en contra y puntos.
- **goleadores**: Almacena la información de los participantes que más goles llevan jornada tras jornada.
- **zamoras**: Almacena la información de los participantes que menos goles reciben en su portería jornada tras jornada.
- **resultados**: Almacena los resultados de las diferentes jornadas una vez se hayan disputado los partidos.

Se trata de una aplicación que no modificará los datos de la base de datos, sólo obtendrá los datos de la misma. Al tratarse de varias referencias a tablas cree una

clase *Aplicacion.java* que incluye la referencia a todas ellas y que haciendo uso de ella pueda ser utilizado por las diferentes clases.

APIs utilizadas

Se ha utilizado la *API de Google Maps v2* con el fin de mostrar al usuario en el mapa donde se encuentra la localización del mismo.

Para la utilización del mismo se necesita una clave de Google Maps la cual se obtiene al dar de alta el proyeto en https://code.google.com/apis/console/. A continuación se introduce dicha clave en el metadata del AndroidManifest.xml:

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="xxxxxxxxxxxxxxxxxxxxxxxxx"/>
```

Permite al usuario abrir Google Maps desde la aplicación para que le guie hasta el campo.

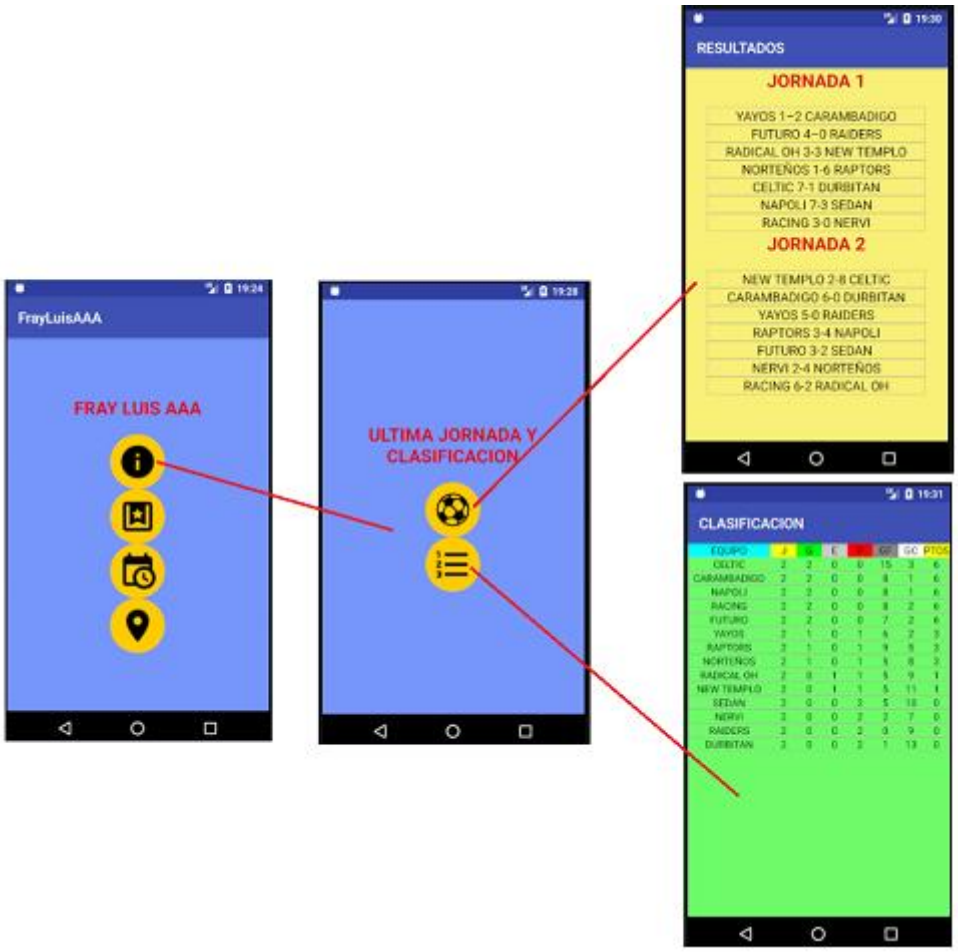
Vistas

A continuación se muestras las diferentes pantallas de la aplicación con una breve explicación de las mismas.

- **PANTALLA PRINCIPAL**

Este es el menú principal desde el cual se accede a las diferentes opciones nos da la aplicación.
Utilicé iconos descriptivos para la identificación de los diferentes submenús que desarrollare a continuación.

- **INFORMACIÓN DE LA LIGA**

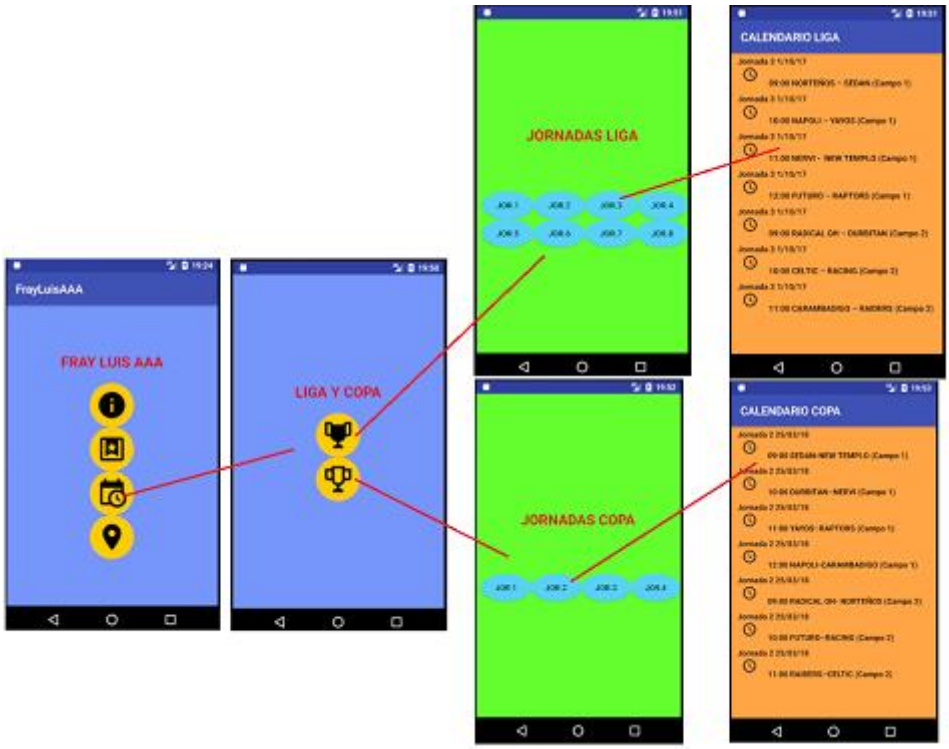


El primer botón del menú principal nos lleva a otro submenú donde podemos elegir entre ver los resultados de las jornadas disputadas y la clasificación actual.

- **ESTADISTICAS**

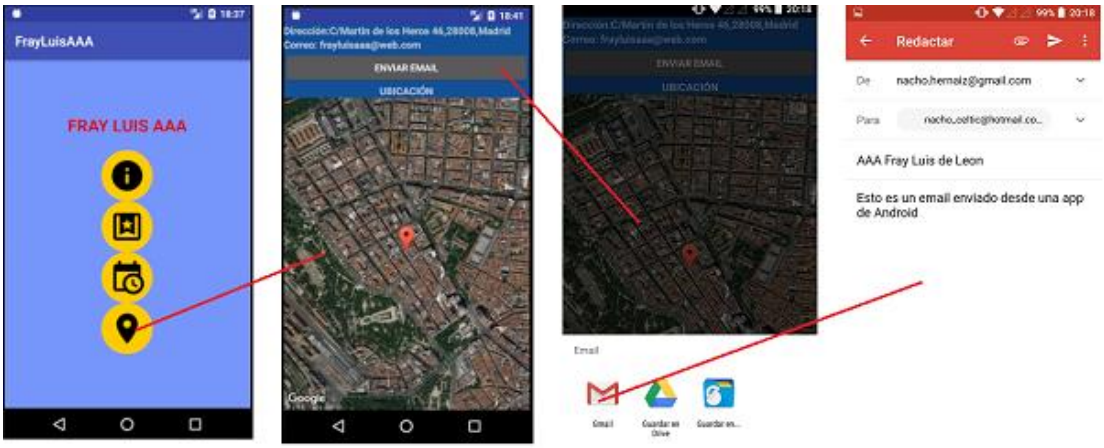
El segundo botón del menú principal nos lleva a otro submenú donde podemos elegir entre ver los goleadores y los porteros menos goleados de la liga.

- **CALENDARIO**



El tercer botón del menú principal nos lleva a otro submenú donde podemos elegir entre ver las jornadas de liga o de copa. Dependiendo la que pulsemos nos llevará a otro menú en el cual podremos elegir la jornada a ver en la cual se nos especificará la fecha y la hora a la que juega nuestra equipo.

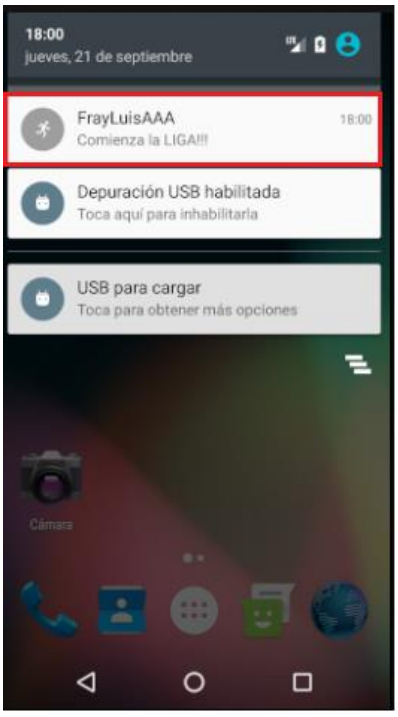
- **CONTACTO**



Es el cuarto y último botón del menú principal. Desde el accedemos a ver el mapa de donde se encuentra el terreno de juego (pabellón con dos campos) y la posibilidad de mandar un correo a la organización para preguntas o dudas.

- **NOTIFICACIONES**

La aplicación puede recibir notificaciones tanto si la aplicación está en un segundo plano como si la aplicación está abierta. A continuación adjunto las dos imágenes.



Conclusiones

Grado de cumplimiento de los objetivos planteados.

Lo primero de todo indicar que nunca había programado nada relacionado con el mundo móvil hasta que me matriculé en el master.

Me plantee como reto la realización de esta aplicación desde la asignatura de Fundamentos y se han cumplido mis expectativas.

Realizar una aplicación desde cero y ver como va tomando forma y funcionalidad fue una sensación muy satisfactoria. Sé que me queda mucho por mejorar pero la base adquirida durante este año ha sido buena.

Líneas abiertas.

El proyecto para mí no ha finalizado aún.

Me gustaría introducirle más cosas que hagan más visual y atractiva a la aplicación que por motivos laborales me ha sido implementarlas.

Como por ejemplo:

- Introducir las equipaciones de los equipos al lado de sus nombres en la pantalla de resultados como en la de clasificación. Las camisetas se podrías cargar gracias al servicio host que nos proporciona Firebase.
- Fotos de los jugadores goleadores y zamoras.
- Suscripciones de notificaciones de tu equipo en concreto.

Son ideas que tengo y que quiero seguir desarrollando para que la aplicación siga creciendo y mejorando.

Consideraciones personales.

Agradecerles a todos vosotros (los profesores) su dedicación durante todo este año, ha sido realmente increíble y que para mí esto no es un punto final si no un punto y seguido para poder seguir creciendo dentro del mundo móvil.

Anexos

Código fuente en GitLab

El código se encuentra alojado en GITLAB en:

https://gitlab.com/nacher8/frayLuisAAA.git.

El proyecto es público y pueden descargárselo. También adjuntaré el código en .zip a través de la plataforma junto con la apk.

Pruebas/Testeo

Las pruebas de la aplicación se han realizado sobre:

- Emulador: Nexus 5 Android v.6.0.
- Terminal: BQ Aquaris E5 Android v.6.0.1.