Mobile Systems and Applications Assignment One.

Student Name: Connor Haines

SID: 17025931

SCHOOL OF COMPUTING AND MATHEMATICS

*An assignment submitted for the degree of   
Bachelor of Science (BSc) in Computer Science.*

January 2020 – University of South Wales

INTRODUCTION

For this assignment, the task is clear. An Android database management application must be built, tested and documented to show knowledge of the localised database features that can be used within Android platforms. SQLite knowledge will also be on display and tested to fully utilise the database.   
  
This document will guide you through the planning and design stage, into the implantation stage and finally will prove the work with testing.

A close up of text on a white background

Description automatically generated

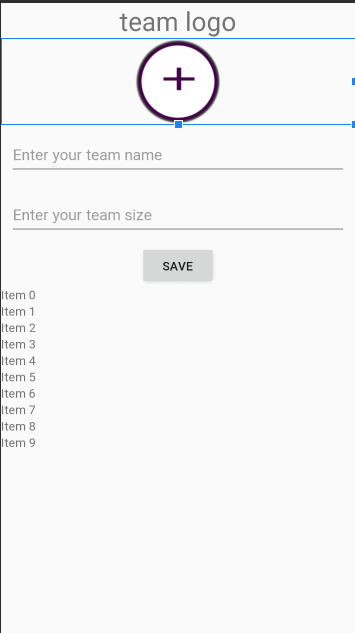
Figure

PLANNING AND DESIGN

The first thing that needed to be done was to decide what the   
application was going to be about; a mind map was created as seen   
in figure one to the right and the theme of the Six Nations Rugby  
Tournament was picked. The application planning began in a   
physical sense with crude drawings, designs and mock-ups on paper.  
An abundance of drafts was made before the final design was picked  
 and the applications design started getting made in Android Studio using *XML files*.

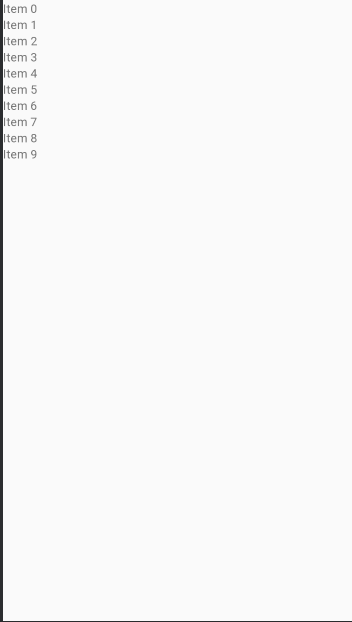
Diagram Three


Figure

The first *layout* designed was the main screen, this was where the user would originally be when they clicked the app icon from the home page of the android device. The idea was that, on first-time running, it should be blank as there will be nothing in the database to show within the layout. Therefore, because the *layout* was planned before the local *SQLite database* was integrated it was kept as simple as possible with a *recycler view*that would eventually show the Teams of the Six Nations Tournament alongside a *floating action button* that would allow the User to add a new Team to the recycler view. Improvements were made to make the recycler view more visually appealing, this will be further discussed later in this document.

Figure

Secondly, the next design that should be discussed should be the *layout* that links to the last one. This is the layout that allows users to add a team to the *recycler view*. Again, simplicity is key when it comes to ease of use for the User. This *layout* is comprised of *TextViews*, an *ImageView* and *Labels*. The *Labels* describe what the user should be inserting into the *database*, e.g. ‘Team Name’ or ‘Team Logo’. The *ImageView* is connected to a function that will allow the user to take a picture or select an image to be used as the team’s logo. Finally, the *TextViews* are where the user will insert the relevant information before clicking the save button to confirm, save the data to the database and return to the *recycler view*.  
  
 The next layout that needed to be worked on was the one below:

  
 This is the screen within the *application* where the user will be able to view the players that are in the team that they clicked on. E.g. if the user clicks on Wales, they could see Alun Wyn Jones and his position. This was a reasonably difficult task as I knew from previous projects and from some early testing that for this layout to work as intended, *the database* and the *application* would have to work seamlessly together to ensure that the layout shows the correct players for the team that the user had clicked in the last screen. This means there would have to be *cross-referencing* between the two tables using *a foreign key constraint*. Thanks to research, I know that this is the best approach for enforcing *referential integrity*. This is because both tables are in the same *database* and if the parent table is in a different *database*, there is nothing in place to stop someone restoring the parent *database* to an earlier time and that creates *orphaned records*.   
The *layout* is made up of a simple recycler view and the data is being fetched using a *DataAdapter* that uses an *ArrayList* as a constraint.

Figure

A picture containing screenshot

Description automatically generatedA screenshot of a cell phone

Description automatically generated As aforementioned, the first layout had slightly more added to it later on in the design process as it looked too plain and uninteresting. The layout now makes great use of LinearLayouts (both Horizontal and Vertical) to create a clean and aesthetically pleasing area for the Team details to be shown. This is used in conjunction with a CardView to wrap the layout and to differentiate between items in the recycler view. This is a good idea as CardViews are specifically designed for heterogenous content and is preferred as the boundaries help the user quickly scan a list to find the specified data. The CardView is split into five areas, this is for the name of the team, the team logo, the team size, an edit button and a delete button. As a side note, the edit layout is exactly the same as the insert layout but it updates the data set for the specific team that’s already in the database.   
  
 Finally, as the first layout was updated, it made the ViewPlayers layout looks outdated, simple and boring. As time was dwindling, the decision was made to make it look better but not to the same standard as the earlier updated layout. This decision made the application look more professional but left enough time to get on with the rest of the implementation.

Figure

Figure

IMPLEMENTATION

AndroidManifest.xml

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 package="info.teams.sqlitedbwithimages">  
  
 <application  
 android:allowBackup="true"  
 android:icon="@drawable/teamlogo"  
 android:label="@string/app\_name"  
 android:theme="@style/AppTheme"  
 tools:ignore="GoogleAppIndexingWarning">  
 <activity android:name=".activities.DisplayPlayersList"></activity>  
 <activity android:name=".activities.PlayersActivity" />  
 <activity android:name=".activities.MainActivity">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 <activity  
 android:name=".activities.InsertDataActivity"  
 android:label="@string/app\_name" />  
 <activity  
 android:name=".activities.editActivity"  
 android:label="Edit Team" />  
 </application>  
  
</manifest>

DisplayPlayersList.java

package info.teams.sqlitedbwithimages.activities;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.os.Bundle;  
  
import java.util.ArrayList;  
  
import info.teams.sqlitedbwithimages.R;  
import info.teams.sqlitedbwithimages.adapters.RecordAdapter;  
import info.teams.sqlitedbwithimages.adapters.playerDataAdapter;  
import info.teams.sqlitedbwithimages.adapters.playerRecordAdapter;  
import info.teams.sqlitedbwithimages.helper.DatabaseHandler;  
import info.teams.sqlitedbwithimages.models.Team;  
import info.teams.sqlitedbwithimages.models.PlayerModel;  
  
public class DisplayPlayersList extends AppCompatActivity {  
 RecyclerView playerrv;  
 playerRecordAdapter playerDataAdapter;  
 LinearLayoutManager linearLayoutManager;  
 ArrayList<PlayerModel> arrayList;  
 DatabaseHandler db;  
 String fid="";  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_display\_players\_list*);  
 playerrv=findViewById(R.id.*playerrv*);  
 db=new DatabaseHandler(getApplicationContext());  
 linearLayoutManager=new LinearLayoutManager(getApplicationContext());  
 playerDataAdapter=new playerRecordAdapter(arrayList,DisplayPlayersList.this);  
 playerrv.setLayoutManager(linearLayoutManager);  
 playerrv.setAdapter(playerDataAdapter);  
 fid=getIntent().getStringExtra("teamid");  
 ShowRecords();  
 }  
 private void ShowRecords() {  
 arrayList = new ArrayList<>(db.getAllPlayers(Integer.*parseInt*(fid)));  
 playerDataAdapter.setdata(arrayList);  
 }  
}

editActivity.java

package info.teams.sqlitedbwithimages.activities;  
  
import android.annotation.TargetApi;  
import android.app.Activity;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.net.Uri;  
import android.os.Build;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.EditText;  
import android.widget.ImageView;  
import android.widget.ListView;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import java.io.ByteArrayOutputStream;  
  
import info.teams.sqlitedbwithimages.R;  
import info.teams.sqlitedbwithimages.helper.DatabaseHandler;  
import info.teams.sqlitedbwithimages.models.Team;  
  
public class editActivity extends Activity {  
  
 private EditText fname,teasize;  
 private ImageView pic;  
 private DatabaseHandler db;  
 private String f\_name,teamsize;  
 private ListView lv;  
 // private dataAdapter data;  
// private PlayerModel dataModel;  
 private Bitmap bp;  
 private byte[] photo;  
 String rid;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_insert*);  
  
 //Instantiate database handler  
 db=new DatabaseHandler(this);  
  
  
// lv = (ListView) findViewById(R.id.list1);  
 pic= findViewById(R.id.*pic*);  
 fname= findViewById(R.id.*txt1*);  
 teasize=findViewById(R.id.*txt2*);  
 rid=getIntent().getStringExtra("rid");  
  
 }  
  
 public void buttonClicked(View v){  
 int id=v.getId();  
  
 switch(id){  
  
 case R.id.*save*:  
  
 if(fname.getText().toString().trim().equals("")){  
 Toast.*makeText*(getApplicationContext(),"Name edit text is empty, Enter name", Toast.*LENGTH\_LONG*).show();  
 } else{  
 UpdateTeam(id);  
 }  
  
 break;  
  
// case R.id.display:  
//  
// // ShowRecords();  
// break;  
 case R.id.*pic*:  
 selectImage();  
 break;  
 }  
 }  
  
 public void selectImage(){  
 Intent photoPickerIntent = new Intent(Intent.*ACTION\_PICK*);  
 photoPickerIntent.setType("image/\*");  
 startActivityForResult(photoPickerIntent, 2);  
 }  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
 switch(requestCode) {  
 case 2:  
 if(resultCode == *RESULT\_OK*){  
 Uri choosenImage = data.getData();  
  
 if(choosenImage !=null){  
  
 bp=decodeUri(choosenImage, 400);  
 pic.setImageBitmap(bp);  
 }  
 }  
 }  
 }  
  
  
 //COnvert and resize our image to 400dp for faster uploading our images to DB  
 protected Bitmap decodeUri(Uri selectedImage, int REQUIRED\_SIZE) {  
  
 try {  
  
 // Decode image size  
 BitmapFactory.Options o = new BitmapFactory.Options();  
 o.inJustDecodeBounds = true;  
 BitmapFactory.*decodeStream*(getContentResolver().openInputStream(selectedImage), null, o);  
  
 // The new size we want to scale to  
 // final int REQUIRED\_SIZE = size;  
  
 // Find the correct scale value. It should be the power of 2.  
 int width\_tmp = o.outWidth, height\_tmp = o.outHeight;  
 int scale = 1;  
 while (true) {  
 if (width\_tmp / 2 < REQUIRED\_SIZE  
 || height\_tmp / 2 < REQUIRED\_SIZE) {  
 break;  
 }  
 width\_tmp /= 2;  
 height\_tmp /= 2;  
 scale \*= 2;  
 }  
  
 // Decode with inSampleSize  
 BitmapFactory.Options o2 = new BitmapFactory.Options();  
 o2.inSampleSize = scale;  
 return BitmapFactory.*decodeStream*(getContentResolver().openInputStream(selectedImage), null, o2);  
 }  
 catch (Exception e){  
 e.printStackTrace();  
 }  
 return null;  
 }  
  
 //Convert bitmap to bytes  
 @TargetApi(Build.VERSION\_CODES.*HONEYCOMB\_MR1*)  
 private byte[] profileImage(Bitmap b){  
  
 ByteArrayOutputStream bos = new ByteArrayOutputStream();  
 b.compress(Bitmap.CompressFormat.*PNG*, 0, bos);  
 return bos.toByteArray();  
  
 }  
  
  
  
 // function to get values from the Edittext and image  
 private void getValues(){  
 f\_name = fname.getText().toString();  
 photo = profileImage(bp);  
 teamsize=teasize.getText().toString();  
  
 }  
  
  
 @Override  
 public void onBackPressed() {  
 super.onBackPressed();  
 Intent intent=new Intent();  
 setResult(*RESULT\_OK*,intent);  
 finish();  
 }  
 //Insert data to the database  
 private void UpdateTeam(int id){  
 getValues();  
  
 db.updateTeam(new Team(f\_name, photo,teamsize),rid);  
 Toast.*makeText*(getApplicationContext(),"updated successfully", Toast.*LENGTH\_LONG*).show();  
 Intent intent=new Intent();  
 setResult(*RESULT\_OK*,intent);  
 finish();  
 }  
  
}

InsertDataActivity.java

package info.teams.sqlitedbwithimages.activities;  
  
import android.annotation.TargetApi;  
import android.app.Activity;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.net.Uri;  
import android.os.Build;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.EditText;  
import android.widget.ImageView;  
import android.widget.ListView;  
import android.widget.Toast;  
  
import java.io.ByteArrayOutputStream;  
  
import info.teams.sqlitedbwithimages.helper.DatabaseHandler;  
import info.teams.sqlitedbwithimages.R;  
import info.teams.sqlitedbwithimages.models.Team;  
  
*/\*\*  
\* \*\*/*public class InsertDataActivity extends Activity {  
  
 private EditText fname,teasize;  
 private ImageView pic;  
 private DatabaseHandler db;  
 private String f\_name,teamsize;  
 private ListView lv;  
// private dataAdapter data;  
// private PlayerModel dataModel;  
 private Bitmap bp;  
 private byte[] photo;  
  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_insert*);  
  
 //Instantiate database handler  
 db=new DatabaseHandler(this);  
  
// lv = (ListView) findViewById(R.id.list1);  
 pic= findViewById(R.id.*pic*);  
 fname= findViewById(R.id.*txt1*);  
 teasize=findViewById(R.id.*txt2*);  
  
 }  
  
 public void buttonClicked(View v){  
 int id=v.getId();  
  
 switch(id){  
  
 case R.id.*save*:  
  
 if(fname.getText().toString().trim().equals("")){  
 Toast.*makeText*(getApplicationContext(),"Name edit text is empty, Enter name", Toast.*LENGTH\_LONG*).show();  
 } else{  
 addTeam();  
 }  
  
 break;  
  
// case R.id.display:  
//  
// // ShowRecords();  
// break;  
 case R.id.*pic*:  
 selectImage();  
 break;  
 }  
 }  
  
 public void selectImage(){  
 Intent photoPickerIntent = new Intent(Intent.*ACTION\_PICK*);  
 photoPickerIntent.setType("image/\*");  
 startActivityForResult(photoPickerIntent, 2);  
 }  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
 switch(requestCode) {  
 case 2:  
 if(resultCode == *RESULT\_OK*){  
 Uri choosenImage = data.getData();  
  
 if(choosenImage !=null){  
  
 bp=decodeUri(choosenImage, 400);  
 pic.setImageBitmap(bp);  
 }  
 }  
 }  
 }  
  
  
 //COnvert and resize our image to 400dp for faster uploading our images to DB  
 protected Bitmap decodeUri(Uri selectedImage, int REQUIRED\_SIZE) {  
  
 try {  
  
 // Decode image size  
 BitmapFactory.Options o = new BitmapFactory.Options();  
 o.inJustDecodeBounds = true;  
 BitmapFactory.*decodeStream*(getContentResolver().openInputStream(selectedImage), null, o);  
  
 // The new size we want to scale to  
 // final int REQUIRED\_SIZE = size;  
  
 // Find the correct scale value. It should be the power of 2.  
 int width\_tmp = o.outWidth, height\_tmp = o.outHeight;  
 int scale = 1;  
 while (true) {  
 if (width\_tmp / 2 < REQUIRED\_SIZE  
 || height\_tmp / 2 < REQUIRED\_SIZE) {  
 break;  
 }  
 width\_tmp /= 2;  
 height\_tmp /= 2;  
 scale \*= 2;  
 }  
  
 // Decode with inSampleSize  
 BitmapFactory.Options o2 = new BitmapFactory.Options();  
 o2.inSampleSize = scale;  
 return BitmapFactory.*decodeStream*(getContentResolver().openInputStream(selectedImage), null, o2);  
 }  
 catch (Exception e){  
 e.printStackTrace();  
 }  
 return null;  
 }  
  
 //Convert bitmap to bytes  
 @TargetApi(Build.VERSION\_CODES.*HONEYCOMB\_MR1*)  
 private byte[] profileImage(Bitmap b){  
  
 ByteArrayOutputStream bos = new ByteArrayOutputStream();  
 b.compress(Bitmap.CompressFormat.*PNG*, 0, bos);  
 return bos.toByteArray();  
  
 }  
  
  
  
 // function to get values from the Edittext and image  
 private void getValues(){  
 f\_name = fname.getText().toString();  
 photo = profileImage(bp);  
 teamsize=teasize.getText().toString();  
  
 }  
  
 //Insert data to the database  
 private void addTeam(){  
 getValues();  
  
 db.addTeams(new Team(f\_name, photo,teamsize));  
 Toast.*makeText*(getApplicationContext(),"Saved successfully", Toast.*LENGTH\_LONG*).show();  
 Intent intent=new Intent();  
 setResult(*RESULT\_OK*,intent);  
 finish();  
 }  
  
 @Override  
 public void onBackPressed() {  
 super.onBackPressed();  
 Intent intent=new Intent();  
 setResult(*RESULT\_OK*,intent);  
 finish();  
 }  
  
}

MainActivity.java

package info.teams.sqlitedbwithimages.activities;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import com.google.android.material.floatingactionbutton.FloatingActionButton;  
  
import java.util.ArrayList;  
  
import info.teams.sqlitedbwithimages.helper.DatabaseHandler;  
import info.teams.sqlitedbwithimages.R;  
import info.teams.sqlitedbwithimages.adapters.RecordAdapter;  
import info.teams.sqlitedbwithimages.interfaces.MyInterfaces;  
import info.teams.sqlitedbwithimages.models.Team;  
  
public class MainActivity extends AppCompatActivity {  
 RecyclerView teamrecyclerview;  
 RecordAdapter recordAdapter;  
 LinearLayoutManager linearLayoutManager;  
 ArrayList<Team>arrayList;  
 private DatabaseHandler db;  
 FloatingActionButton add;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main\_activity*);  
 teamrecyclerview=findViewById(R.id.*teamrecyclerview*);  
 add=findViewById(R.id.*addrecord*);  
 arrayList=new ArrayList<>();  
 db=new DatabaseHandler(this);  
 linearLayoutManager=new LinearLayoutManager(getApplicationContext());  
 teamrecyclerview.setLayoutManager(linearLayoutManager);  
 recordAdapter=new RecordAdapter(arrayList, MainActivity.this);  
 teamrecyclerview.setAdapter(recordAdapter);  
 recordAdapter.setdata(arrayList);  
 recordAdapter.setDeleteRecord(new MyInterfaces.deleteRecord() {  
 @Override  
 public void onDeleteRecord(int id) {  
 db.deleteTeam(id);  
 ShowRecords();  
 recordAdapter.setdata(arrayList);  
 recreate();  
 // recordAdapter.notifyDataSetChanged();  
 }  
 });  
 recordAdapter.setUpdateRecord(new MyInterfaces.updateRecord() {  
 @Override  
 public void onUpdateRecord(int id) {  
 Intent intent=new Intent(MainActivity.this,editActivity.class);  
 intent.putExtra("rid",String.*valueOf*(id));  
 startActivityForResult(intent,2);  
 ShowRecords();  
 recordAdapter.setdata(arrayList);  
 recreate();  
  
 }  
 });  
 recordAdapter.notifyDataSetChanged();  
  
 ShowRecords();  
 add.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Intent intent=new Intent(MainActivity.this, InsertDataActivity.class);  
 startActivityForResult(intent,1);  
 }  
 });  
 }  
  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
 if(requestCode==1)  
 {  
 arrayList = new ArrayList<>(db.getAllTeams());  
  
 recordAdapter.setdata(arrayList);  
  
 }  
 else if(resultCode==2)  
 {  
 arrayList = new ArrayList<>(db.getAllTeams());  
 recordAdapter.setdata(arrayList);  
  
 }else{  
 recreate();  
 }  
 }  
 private void ShowRecords() {  
  
 arrayList = new ArrayList<>(db.getAllTeams());  
 recordAdapter.setdata(arrayList);  
  
 }  
}

PlayersActivity.java

package info.teams.sqlitedbwithimages.activities;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.Toast;  
  
import info.teams.sqlitedbwithimages.R;  
import info.teams.sqlitedbwithimages.helper.DatabaseHandler;  
import info.teams.sqlitedbwithimages.models.PlayerModel;  
  
public class PlayersActivity extends AppCompatActivity implements View.OnClickListener {  
EditText playername,playerpositon;  
Button saveplayer;  
String teamid,player,position;  
private DatabaseHandler db;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_new*);  
 playername=findViewById(R.id.*etpname*);  
 playerpositon=findViewById(R.id.*etppos*);  
 saveplayer=findViewById(R.id.*saveplayer*);  
 db=new DatabaseHandler(getApplicationContext());  
 saveplayer.setOnClickListener(this);  
 teamid=getIntent().getStringExtra("teamid");  
 getValues();  
 }  
 private void getValues(){  
 player = playername.getText().toString();  
 position=playerpositon.getText().toString();  
  
 }  
 @Override  
 public void onClick(View view) {  
 boolean error=false;  
 player=playername.getText().toString();  
 position=playerpositon.getText().toString();  
 if(player.equals(""))  
 {  
 Toast.*makeText*(this, "player name must be selected", Toast.*LENGTH\_SHORT*).show();  
 error=true;  
 }  
 if(position.equals(""))  
 {  
 Toast.*makeText*(this, "player positon should not be empty", Toast.*LENGTH\_SHORT*).show();  
 error=true;  
 }  
 if(error)  
 return;  
 adPlayer(player,position,teamid);  
 }  
 private void adPlayer(String player,String position,String teamid){  
 getValues();  
  
 db.addPlayers(new PlayerModel(player, position,Integer.*parseInt*(teamid)));  
 Toast.*makeText*(getApplicationContext(),"Saved player successfully", Toast.*LENGTH\_LONG*).show();  
  
 finish();  
 }  
}

dataAdapter.java

package info.teams.sqlitedbwithimages.adapters;  
  
 import android.content.Context;  
 import android.graphics.Bitmap;  
 import android.graphics.BitmapFactory;  
 import android.view.LayoutInflater;  
 import android.view.View;  
 import android.view.ViewGroup;  
 import android.widget.ArrayAdapter;  
 import android.widget.ImageView;  
 import android.widget.TextView;  
 import android.widget.Toast;  
  
 import java.util.ArrayList;  
  
 import info.teams.sqlitedbwithimages.R;  
 import info.teams.sqlitedbwithimages.models.Team;  
  
*/\*\*  
 \* \*\*/*public class dataAdapter extends ArrayAdapter<Team>{  
  
 Context context;  
 ArrayList<Team> mteam;  
  
  
 public dataAdapter(Context context, ArrayList<Team> team){  
 super(context, R.layout.*listteams*, team);  
 this.context=context;  
 this.mteam=team;  
 }  
  
 public class Holder{  
 TextView nameFV;  
 ImageView pic;  
 }  
  
 @Override  
 public View getView(int position, View convertView, ViewGroup parent) {  
 // Get the data item for this position  
  
 Team data = getItem(position);  
 // Check if an existing view is being reused, otherwise inflate the view  
  
 Holder viewHolder; // view lookup cache stored in tag  
  
 if (convertView == null) {  
  
  
 viewHolder = new Holder();  
 LayoutInflater inflater = LayoutInflater.*from*(getContext());  
 convertView = inflater.inflate(R.layout.*listteams*, parent, false);  
  
 viewHolder.nameFV = (TextView) convertView.findViewById(R.id.*txtViewer*);  
 viewHolder.pic = (ImageView) convertView.findViewById(R.id.*imgView*);  
  
  
 convertView.setTag(viewHolder);  
 } else {  
 viewHolder = (Holder) convertView.getTag();  
 }  
  
  
 viewHolder.nameFV.setText("Team Name: "+data.getFName());  
 viewHolder.pic.setImageBitmap(convertToBitmap(data.getImage()));  
 Toast.*makeText*(context, ""+data.getTeamsize(), Toast.*LENGTH\_SHORT*).show();  
  
  
 // Return the completed view to render on screen  
 return convertView;  
 }  
 //get bitmap image from byte array  
  
 private Bitmap convertToBitmap(byte[] b){  
  
 return BitmapFactory.*decodeByteArray*(b, 0, b.length);  
  
 }  
  
}

playerDataAdapter.java

package info.teams.sqlitedbwithimages.adapters;  
  
import android.content.Context;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ArrayAdapter;  
import android.widget.TextView;  
  
import java.util.ArrayList;  
  
import info.teams.sqlitedbwithimages.R;  
  
import info.teams.sqlitedbwithimages.models.PlayerModel;  
  
*/\*\*  
 \* \*\*/*public class playerDataAdapter extends ArrayAdapter<PlayerModel>{  
  
 Context context;  
 ArrayList<PlayerModel> mplayers;  
  
  
 public playerDataAdapter(Context context, ArrayList<PlayerModel> player){  
 super(context, R.layout.*listplayer*, player);  
 this.context=context;  
 this.mplayers=player;  
 }  
  
  
  
 public class Holder{  
 TextView name;  
 TextView pos;  
 }  
  
 @Override  
 public View getView(int position, View convertView, ViewGroup parent) {  
 // Get the data item for this position  
  
 PlayerModel data = getItem(position);  
 // Check if an existing view is being reused, otherwise inflate the view  
  
 Holder viewHolder; // view lookup cache stored in tag  
  
 if (convertView == null) {  
  
  
 viewHolder = new Holder();  
 LayoutInflater inflater = LayoutInflater.*from*(getContext());  
 convertView = inflater.inflate(R.layout.*listplayer*, parent, false);  
  
 viewHolder.name = (TextView) convertView.findViewById(R.id.*txtViewer*);  
 viewHolder.pos = (TextView) convertView.findViewById(R.id.*txtViewer*);  
  
  
 convertView.setTag(viewHolder);  
 } else {  
 viewHolder = (Holder) convertView.getTag();  
 }  
  
  
 viewHolder.name.setText("Player: "+data.getPName());  
 viewHolder.name.setText("Position: "+data.getPposition());  
  
  
  
 // Return the completed view to render on screen  
 return convertView;  
 }  
 //get bitmap image from byte array  
  
  
}

playerRecordAdapter.java

package info.teams.sqlitedbwithimages.adapters;  
  
import android.app.Activity;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.TextView;  
  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
  
import java.util.ArrayList;  
  
import info.teams.sqlitedbwithimages.R;  
import info.teams.sqlitedbwithimages.models.PlayerModel;  
  
public class playerRecordAdapter extends RecyclerView.Adapter<playerRecordAdapter.MyViewHolder> {  
 ArrayList<PlayerModel>arrayList;  
 Activity activity;  
  
 public playerRecordAdapter(ArrayList<PlayerModel> arrayList, Activity activity) {  
 this.arrayList = arrayList;  
 this.activity = activity;  
 }  
  
 @NonNull  
 @Override  
 public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view= LayoutInflater.*from*(activity).inflate(R.layout.*listplayer*,parent,false);  
 return new MyViewHolder(view);  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull MyViewHolder holder, int position) {  
 PlayerModel players=arrayList.get(position);  
 holder.pname.setText(players.getPName());  
 holder.pposition.setText(players.getPposition());  
  
 }  
  
 @Override  
 public int getItemCount() {  
 return arrayList.size();  
 }  
 private Bitmap convertToBitmap(byte[] b){  
  
 return BitmapFactory.*decodeByteArray*(b, 0, b.length);  
  
 }  
  
 public void setdata(ArrayList<PlayerModel> arrayList) {  
 this.arrayList=arrayList;  
 notifyDataSetChanged();  
 }  
  
  
  
 class MyViewHolder extends RecyclerView.ViewHolder  
 {  
 TextView pname,pposition;  
 MyViewHolder(@NonNull View itemView) {  
 super(itemView);  
 pname=itemView.findViewById(R.id.*pname*);  
 pposition=itemView.findViewById(R.id.*pposition*);  
  
 }  
 }  
}

recordAdapter.java

package info.teams.sqlitedbwithimages.adapters;  
  
import android.app.Activity;  
import android.content.DialogInterface;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ImageButton;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AlertDialog;  
import androidx.cardview.widget.CardView;  
import androidx.recyclerview.widget.RecyclerView;  
  
import java.util.ArrayList;  
  
import info.teams.sqlitedbwithimages.R;  
import info.teams.sqlitedbwithimages.activities.DisplayPlayersList;  
import info.teams.sqlitedbwithimages.activities.PlayersActivity;  
import info.teams.sqlitedbwithimages.activities.editActivity;  
import info.teams.sqlitedbwithimages.interfaces.MyInterfaces;  
import info.teams.sqlitedbwithimages.models.Team;  
  
public class RecordAdapter extends RecyclerView.Adapter<RecordAdapter.MyViewHolder> {  
 ArrayList<Team>arrayList;  
 Activity activity;  
 MyInterfaces.deleteRecord deleteRecord;  
 MyInterfaces.updateRecord updateRecord;  
 public RecordAdapter(ArrayList<Team> arrayList, Activity activity) {  
 this.arrayList = arrayList;  
 this.activity = activity;  
 }  
  
 @NonNull  
 @Override  
 public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view= LayoutInflater.*from*(activity).inflate(R.layout.*listteams*,parent,false);  
 return new MyViewHolder(view);  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull MyViewHolder holder, int position) {  
 final Team team=arrayList.get(position);  
 holder.teamLogo.setImageBitmap(convertToBitmap(team.getImage()));  
 holder.temaname.setText("Team Name "+team.getFName());  
 holder.teamsize.setText("Team Size "+team.getTeamsize());  
 holder.teamLogo.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
  
 playersDialog(String.*valueOf*(team.getID()));  
 }  
 });  
 holder.delete.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 deleteRecord.onDeleteRecord(team.getID());  
 }  
 });  
 holder.edit.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 updateRecord.onUpdateRecord(team.getID());  
  
 }  
 });  
 }  
  
 @Override  
 public int getItemCount() {  
 return arrayList.size();  
 }  
 private Bitmap convertToBitmap(byte[] b){  
  
 return BitmapFactory.*decodeByteArray*(b, 0, b.length);  
  
 }  
  
 public void setdata(ArrayList<Team> arrayList) {  
 this.arrayList=arrayList;  
 notifyDataSetChanged();  
 }  
  
 class MyViewHolder extends RecyclerView.ViewHolder  
 {  
 ImageView teamLogo;  
 TextView temaname,teamsize;  
 CardView cardinfo;  
 ImageButton edit,delete;  
 MyViewHolder(@NonNull View itemView) {  
 super(itemView);  
 temaname=itemView.findViewById(R.id.*txtViewer*);  
 teamsize=itemView.findViewById(R.id.*teamsize*);  
 teamLogo=itemView.findViewById(R.id.*imgView*);  
 cardinfo=itemView.findViewById(R.id.*cardinfo*);  
 edit=itemView.findViewById(R.id.*edit*);  
 delete=itemView.findViewById(R.id.*delete*);  
 }  
 }  
  
 public void setDeleteRecord(MyInterfaces.deleteRecord deleteRecord) {  
 this.deleteRecord = deleteRecord;  
 }  
  
 public void setUpdateRecord(MyInterfaces.updateRecord updateRecord) {  
 this.updateRecord = updateRecord;  
 }  
 private void playersDialog(final String cid)  
 {  
 AlertDialog.Builder builder1 = new AlertDialog.Builder(activity);  
 builder1.setMessage("Add or display players");  
 builder1.setCancelable(true);  
  
 builder1.setPositiveButton(  
 "Diplay Players",  
 new DialogInterface.OnClickListener() {  
 public void onClick(DialogInterface dialog, int id) {  
 Intent intent=new Intent(activity, DisplayPlayersList.class);  
 intent.putExtra("teamid",cid);  
 activity.startActivity(intent);  
 }  
 });  
 builder1.setNegativeButton(  
 "Add Players",  
 new DialogInterface.OnClickListener() {  
 public void onClick(DialogInterface dialog, int id) {  
 Intent intent=new Intent(activity, PlayersActivity.class);  
 intent.putExtra("teamid",cid);  
 activity.startActivity(intent);  
 }  
 });  
  
 AlertDialog alert11 = builder1.create();  
 alert11.show();  
 }  
}

DatabaseHandler.java

package info.teams.sqlitedbwithimages.helper;  
  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
import android.util.Log;  
  
import java.util.ArrayList;  
import java.util.List;  
  
import info.teams.sqlitedbwithimages.models.Team;  
import info.teams.sqlitedbwithimages.models.PlayerModel;  
  
*/\*\*  
 \* \*\*/*public class DatabaseHandler extends SQLiteOpenHelper {  
  
 // Database Version  
 private static final int *DATABASE\_VERSION* = 1;  
  
 // Database Name  
 private static final String *DATABASE\_NAME* = "teamsManager.db";  
  
 // teams table name  
 private static final String *TABLE\_TEAMS* = "teams";  
  
  
 // teams Table Columns names  
 private static final String *KEY\_ID* = "id";  
 private static final String *KEY\_FNAME* = "fname";  
 private static final String *KEY\_POTO* = "poto";  
 private static final String *KEY\_TEAMSIZE* = "teamsize";  
  
 //Player table name  
 private static final String *TABLE\_PLAYERS* = "players";  
  
 // teams Table Columns names  
 private static final String *KEY\_ID\_PLAYERS* = "id";  
 private static final String *KEY\_PNAME* = "pname";  
 private static final String *KEY\_POSITION* = "position";  
 private static final String *KEY\_FID* = "fid";  
  
  
 public DatabaseHandler(Context context) {  
 super(context, *DATABASE\_NAME*, null, *DATABASE\_VERSION*);  
 }  
  
 //Create tables  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
 String CREATE\_TABLE\_TEAMS="CREATE TABLE " + *TABLE\_TEAMS* + "("  
 + *KEY\_ID* +" INTEGER PRIMARY KEY,"  
 + *KEY\_FNAME* +" TEXT,"  
 + *KEY\_TEAMSIZE* +" TEXT,"  
 + *KEY\_POTO* +" BLOB" + ")";  
  
 String CREATE\_TABLE\_PLAYERS="CREATE TABLE " + *TABLE\_PLAYERS* + "("  
 + *KEY\_ID\_PLAYERS* +" INTEGER PRIMARY KEY,"  
 + *KEY\_PNAME* +" TEXT,"  
 + *KEY\_POSITION* +" TEXT,"  
 + *KEY\_FID* +" INTEGER" + ")";  
  
  
 db.execSQL(CREATE\_TABLE\_TEAMS);  
 db.execSQL(CREATE\_TABLE\_PLAYERS);  
 }  
  
 // Upgrading database  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
  
 // Drop older table if existed  
 db.execSQL("DROP TABLE IF EXISTS " + *TABLE\_TEAMS*);  
 db.execSQL("DROP TABLE IF EXISTS " + *TABLE\_PLAYERS*);  
  
  
 // Create tables again  
 onCreate(db);  
 }  
  
 */\*\*  
 \* All CRUD(Create, Read, Update, Delete) Operations  
 \*/* //Insert values to the table teams  
 public void addTeams(Team team){  
 SQLiteDatabase db = this.getReadableDatabase();  
 ContentValues values=new ContentValues();  
  
 values.put(*KEY\_FNAME*, team.getFName());  
 values.put(*KEY\_TEAMSIZE*, team.getTeamsize());  
 values.put(*KEY\_POTO*, team.getImage() );  
  
  
 db.insert(*TABLE\_TEAMS*, null, values);  
 db.close();  
 }  
  
 public void addPlayers(PlayerModel players){  
 SQLiteDatabase db = this.getReadableDatabase();  
 ContentValues values=new ContentValues();  
  
 values.put(*KEY\_PNAME*, players.getPName());  
 values.put(*KEY\_POSITION*, players.getPposition());  
 values.put(*KEY\_FID*, players.getFID());  
  
  
 db.insert(*TABLE\_PLAYERS*, null, values);  
 db.close();  
 }  
  
  
 */\*\*  
 \*Getting All Teams  
 \*\*/* public List<Team> getAllTeams() {  
 List<Team> teamList = new ArrayList<Team>();  
 // Select All Query  
 String selectQuery = "SELECT \* FROM " + *TABLE\_TEAMS*;  
  
 SQLiteDatabase db = this.getWritableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery, null);  
  
 // looping through all rows and adding to list  
 if (cursor!=null&&cursor.moveToFirst()) {  
 do {  
 Team team = new Team();  
 team.setID(Integer.*parseInt*(cursor.getString(0)));  
 team.setFName(cursor.getString(1));  
 team.setTeamsize(cursor.getString(2));  
 team.setImage(cursor.getBlob(3));  
  
  
 // Adding team to list  
 teamList.add(team);  
 } while (cursor.moveToNext());  
 }  
  
 // return team list  
 return teamList;  
 }  
  
 public List<PlayerModel> getAllPlayers(int fid) {  
 List<PlayerModel> playerList = new ArrayList<PlayerModel>();  
 // Select All Query  
 // String selectQuery = "SELECT \* FROM " + TABLE\_PLAYERS ;  
 String selectQuery = ("SELECT " +*KEY\_ID* +","+*KEY\_PNAME*+ ","+*KEY\_POSITION*+","+*KEY\_FID*+" FROM "  
 + *TABLE\_PLAYERS* + " where `" + *KEY\_FID* + "`="  
 + fid);  
  
 SQLiteDatabase db = this.getWritableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery, null);  
  
 // looping through all rows and adding to list  
 if (cursor!=null&&cursor.moveToFirst()) {  
 do {  
 PlayerModel player = new PlayerModel();  
 player.setID(Integer.*parseInt*(cursor.getString(0)));  
 player.setPName(cursor.getString(1));  
 player.setPposition(cursor.getString(2));  
 player.setFID(cursor.getInt(3));  
  
  
 // Adding team to list  
 playerList.add(player);  
 } while (cursor.moveToNext());  
 }  
  
 // return team list  
 return playerList;  
 }  
  
  
  
  
 */\*\*  
 \*Updating single team  
 \*\*/* public void updateTeam(Team team,String rid) {  
 SQLiteDatabase db = DatabaseHandler.this.getWritableDatabase();  
  
 try {  
 db.beginTransaction();  
 ContentValues values = new ContentValues();  
 values.put(*KEY\_FNAME*, team.getFName());  
 values.put(*KEY\_TEAMSIZE*, team.getTeamsize());  
 values.put(*KEY\_POTO*, team.getImage());  
  
  
 // updating row  
 db.update(*TABLE\_TEAMS*, values, "id=?", new String[]{String.*valueOf*(rid)});  
 db.setTransactionSuccessful();  
 db.endTransaction();  
 db.close();  
 db.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 Log.*d*("e",e.getMessage());  
 }  
  
 }  
  
 public int updatePlayers(PlayerModel player, int id) {  
 SQLiteDatabase db = this.getWritableDatabase();  
  
 ContentValues values = new ContentValues();  
 values.put(*KEY\_PNAME*, player.getPName());  
 values.put(*KEY\_POSITION*, player.getPposition());  
  
  
 // updating row  
 return db.update(*TABLE\_PLAYERS*, values, *KEY\_ID* + " = ?",  
 new String[] { String.*valueOf*(id) });  
 }  
  
 */\*\*  
 \*Deleting single team  
 \*\*/* public void deleteTeam(int Id) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.delete(*TABLE\_TEAMS*, *KEY\_ID* + " = ?",  
 new String[] { String.*valueOf*(Id) });  
 db.delete(*TABLE\_PLAYERS*, *KEY\_ID* + " = ?",  
 new String[] { String.*valueOf*(Id) });  
 db.close();  
 }  
  
 public void deletePlayer(int Id) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.delete(*TABLE\_PLAYERS*, *KEY\_ID* + " = ?",  
 new String[] { String.*valueOf*(Id) });  
 db.close();  
 }  
  
}

myInterfaces.java

package info.teams.sqlitedbwithimages.interfaces;  
  
public class MyInterfaces {  
 public interface deleteRecord  
 {  
 void onDeleteRecord(int id);  
 }  
 public interface updateRecord  
 {  
 void onUpdateRecord(int id);  
 }  
}

playerModel.java

package info.teams.sqlitedbwithimages.models;  
  
*/\*\*  
 \* \*\*/*public class PlayerModel {  
  
 //private variables  
 int \_id;  
 String \_pname,pposition;  
 int \_fid;  
  
  
 // Empty constructor  
 public PlayerModel(){  
  
 }  
  
 // constructor  
 public PlayerModel(String pname, String pposition , int fid){  
  
 this.\_pname = pname;  
 this.pposition = pposition;  
 this.\_fid = fid;  
 }  
  
 public String getPposition() {  
 return pposition;  
 }  
  
 public void setPposition(String pposition) {  
 this.pposition = pposition;  
 }  
  
 // getting ID  
 public int getID(){  
 return this.\_id;  
 }  
  
 // setting id  
 public void setID(int id){  
 this.\_id = id;  
 }  
  
 // getting name  
 public String getPName(){  
 return this.\_pname;  
 }  
  
 // setting name  
 public void setPName(String pname){  
 this.\_pname = pname;  
 }  
  
 //getting profile pic  
 public int getFID(){  
 return this.\_fid;  
 }  
  
 //setting profile pic  
  
 public void setFID(int fid) {  
 this.\_fid = fid;  
 }  
}

teamModel.java

package info.teams.sqlitedbwithimages.models;  
  
import java.io.Serializable;  
  
*/\*\*  
 \* \*\*/*public class Team implements Serializable {  
  
 //private variables  
 int \_id;  
 String \_fname,teamsize;  
 byte[] \_img;  
  
  
  
 // Empty constructor  
 public Team(){  
  
 }  
 // constructor  
 public Team(int id, String fname,String teamsize, byte[] img){  
 this.\_id = id;  
 this.\_fname = fname;  
 this.teamsize=teamsize;  
 this.\_img = img;  
  
 }  
  
 // constructor  
 public Team(String fname, byte[] img,String teamsize){  
  
 this.\_fname = fname;  
 this.\_img = img;  
 this.teamsize=teamsize;  
 }  
  
 public String getTeamsize() {  
 return teamsize;  
 }  
  
 public void setTeamsize(String teamsize) {  
 this.teamsize = teamsize;  
 }  
  
 // getting ID  
 public int getID(){  
 return this.\_id;  
 }  
  
 // setting id  
 public void setID(int id){  
 this.\_id = id;  
 }  
  
 // getting first name  
 public String getFName(){  
 return this.\_fname;  
 }  
  
 // setting first name  
 public void setFName(String fname){  
 this.\_fname = fname;  
 }  
  
 //getting profile pic  
 public byte[] getImage(){  
 return this.\_img;  
 }  
  
 //setting profile pic  
  
 public void setImage(byte[] b){  
 this.\_img=b;  
 }  
  
}

BuildConfig.java

*/\*\*  
 \* Automatically generated file. DO NOT MODIFY  
 \*/*package info.teams.sqlitedbwithimages;  
  
public final class BuildConfig {  
 public static final boolean *DEBUG* = Boolean.*parseBoolean*("true");  
 public static final String *APPLICATION\_ID* = "info.teams.sqlitedbwithimages";  
 public static final String *BUILD\_TYPE* = "debug";  
 public static final String *FLAVOR* = "";  
 public static final int *VERSION\_CODE* = 1;  
 public static final String *VERSION\_NAME* = "1.0";  
}

Activity\_display\_players\_list.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".activities.DisplayPlayersList">  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/playerrv"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"/>  
</LinearLayout>

Activity\_edit.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context=".activities.editActivity">  
  
 <TextView android:text="team logo"  
 android:layout\_gravity="center"  
 android:textSize="30dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content" />  
  
 <ImageView  
 android:layout\_width="match\_parent"  
 android:layout\_height="100dp"  
 android:onClick="buttonClicked"  
 android:id="@+id/pic"  
 android:src="@drawable/choose\_up"/>  
  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/txt1"  
 android:hint="Enter your team name"  
 android:layout\_margin="10dp"/>  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/txt2"  
 android:hint="Enter your team size"  
 android:inputType="number"  
 android:layout\_margin="10dp"/>  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:orientation="horizontal"  
 android:gravity="center"  
 android:layout\_height="wrap\_content">  
  
 <Button  
 android:id="@+id/save"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="buttonClicked"  
 android:text="Save" />  
  
  
 </LinearLayout>  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/playerRV"  
 android:layout\_width="220dp"  
 android:layout\_height="414dp" />  
 </LinearLayout>  
  
</LinearLayout>

Activity\_insert.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context=".activities.InsertDataActivity">  
  
 <TextView android:text="team logo"  
 android:layout\_gravity="center"  
 android:textSize="30dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content" />  
  
 <ImageView  
 android:layout\_width="match\_parent"  
 android:layout\_height="100dp"  
 android:onClick="buttonClicked"  
 android:id="@+id/pic"  
 android:src="@drawable/choose\_up"/>  
  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/txt1"  
 android:hint="Enter your team name"  
 android:layout\_margin="10dp"/>  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/txt2"  
 android:hint="Enter your team size"  
 android:inputType="number"  
 android:layout\_margin="10dp"/>  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:orientation="horizontal"  
 android:gravity="center"  
 android:layout\_height="wrap\_content">  
  
 <Button  
 android:id="@+id/save"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="buttonClicked"  
 android:text="Save" />  
  
 <!--<Button  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:onClick="buttonClicked"  
 android:id="@+id/display"  
 android:text="Display"/>-->  
  
 </LinearLayout>  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" />  
 </LinearLayout>  
  
</LinearLayout>

Activity\_main\_activity.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".activities.MainActivity">  
 <FrameLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
<androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/teamrecyclerview"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"/>  
 <com.google.android.material.floatingactionbutton.FloatingActionButton  
 android:id="@+id/addrecord"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:src="@android:drawable/ic\_input\_add"  
 android:layout\_gravity="bottom|end"  
 android:layout\_margin="10dp"/>  
  
 </FrameLayout>  
</LinearLayout>

Activity\_new.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".activities.PlayersActivity"  
 android:orientation="vertical">  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/etpname"  
 android:hint="Enter player name"  
 android:layout\_margin="10dp"/>  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:id="@+id/etppos"  
 android:hint="Enter player position"  
 android:layout\_margin="10dp"/>  
 <Button  
 android:id="@+id/saveplayer"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="save"  
 android:textAllCaps="false"  
 android:layout\_gravity="center"/>  
</LinearLayout>

Listplayer.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical" android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
<LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_margin="7dp">  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Player Name"  
 android:textStyle="bold"  
 android:fontFamily="sans-serif-light"/>  
 <TextView  
 android:id="@+id/pname"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content" />  
</LinearLayout>  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:layout\_margin="7dp">  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Player Position"  
 android:textStyle="bold"  
 android:fontFamily="sans-serif-light"/>  
 <TextView  
 android:id="@+id/pposition"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 />  
 </LinearLayout>  
 <View android:background="#C0BCBC"  
 android:layout\_width = "match\_parent"  
 android:layout\_height="1dp"/>  
</LinearLayout>

Listteams.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 <androidx.cardview.widget.CardView  
 android:id="@+id/cardinfo"  
 android:layout\_width="match\_parent"  
 android:layout\_height="109dp"  
 android:layout\_margin="10dp"  
 app:cardCornerRadius="5sp">  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
  
 <ImageView  
 android:id="@+id/imgView"  
 android:layout\_width="100dp"  
 android:layout\_height="100dp"  
 android:layout\_gravity="center"  
 android:layout\_margin="5dp" />  
 </LinearLayout>  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical">  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
  
 <TextView  
 android:id="@+id/txtViewer"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_margin="5dp"  
 android:gravity="center"  
 android:text="name"  
 android:textAlignment="center" />  
 </LinearLayout>  
  
 <TextView  
 android:id="@+id/teamsize"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:layout\_margin="5dp"  
 android:gravity="center"  
 android:text="size"  
 android:textAlignment="center" />  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
  
 <ImageButton  
 android:id="@+id/delete"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_margin="5dp"  
 android:src="@drawable/delete" />  
  
 <ImageButton  
 android:id="@+id/edit"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_margin="5dp"  
 android:src="@drawable/edit" />  
 </LinearLayout>  
 </LinearLayout>  
  
 </LinearLayout>  
 </androidx.cardview.widget.CardView>  
</LinearLayout>

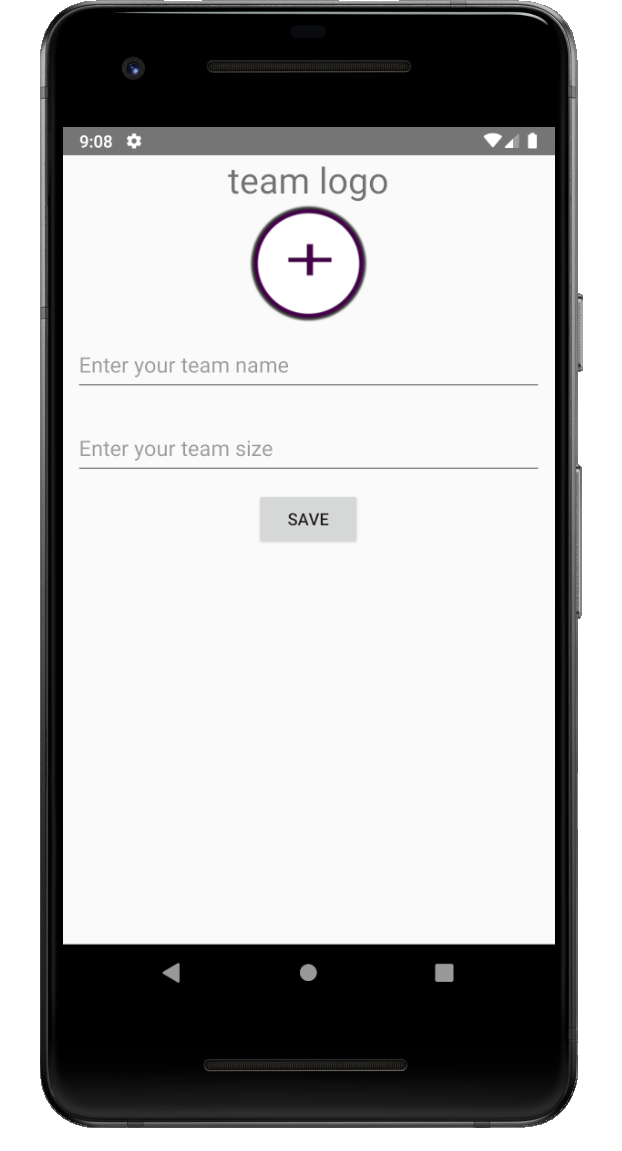
TESTING

All evidence for the tests will be below the table and labelled with the corresponding test number.

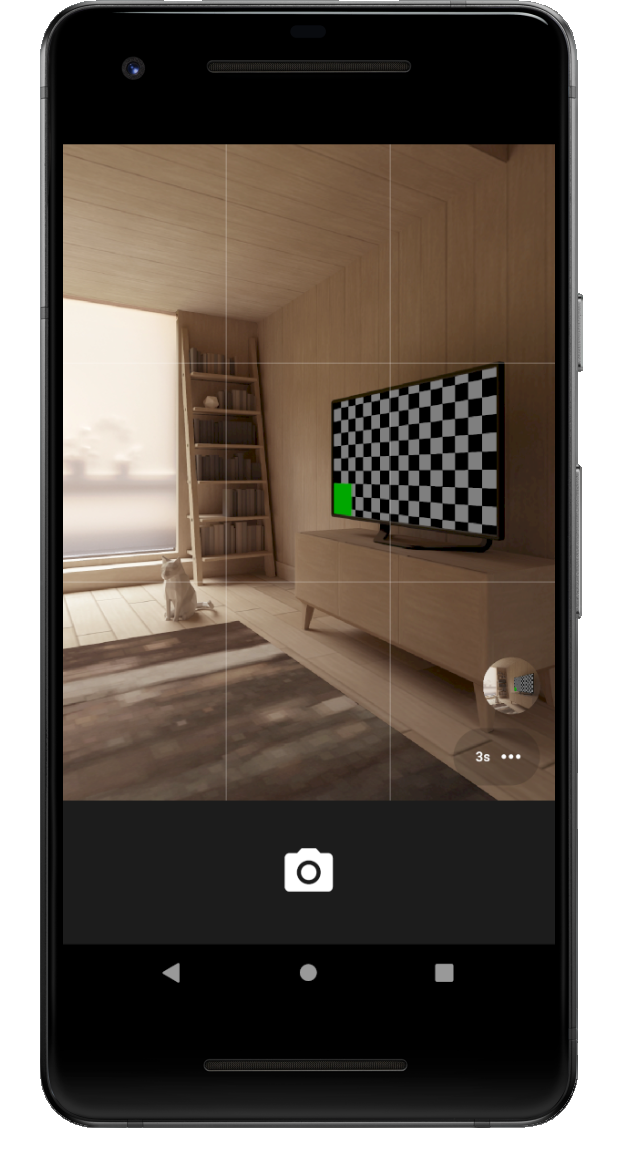
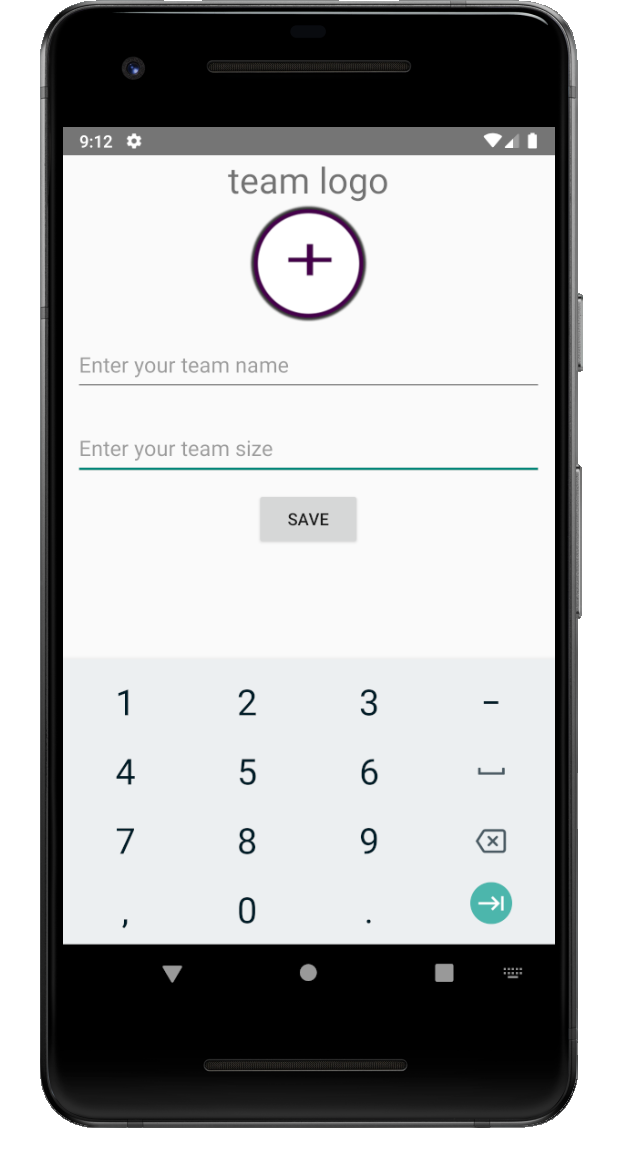
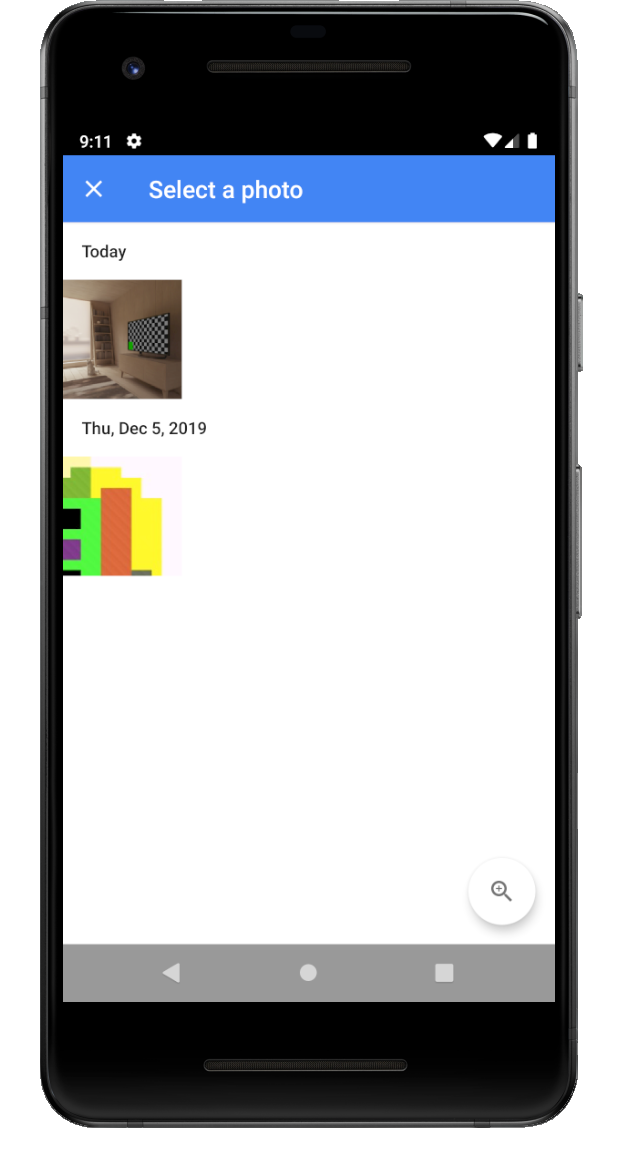
|  |  |  |  |
| --- | --- | --- | --- |
| Test Number | Test Summary | Expected Result | Actual Result |
| 1 | Application Start Up | When pressing the run icon in Android Studio I expect the application to start up and show the recycler view. | After pressing the run icon, the application started up as it should, and the recycler view was visible. |
| 2 | Application Icon Start Up | When the application is closed on the emulator and you click the icon the application should start up and show the recycler view. | After the application is closed on the emulator and you click the icon the application starts up and the recycler view is visible. |
| 3 | Add Team Button | When the add team button is clicked I expect to see the add team layout. | The application responds quickly and the add team layout is shown. |
| 4 | Choose Team Logo | I expect the gallery to open and the images I can choose from to be visible and are on the layout when clicked. | The gallery opens and the images that I can choose from are there. Then, they are on the layout when clicked. |
| 5 | Take Team Logo Picture | The camera application should open and then once the picture is taken the image should show up in the gallery. | The camera application opened and, once taken, the picture showed up in the gallery to choose as a logo picture. |
| 6 | Type String in Team Size | The application should not allow you to type a string into the team size area as it is an integer TextView. | When trying to type a string, nothing appears. |
| 7 | Save New Team | Once clicked, the application should return to the view teams’ layout and the newly saved team should be there. | The application returned the correct layout and the newly saved team was visible along with the correct data. |
| 8 | Edit (pencil) Button | The edit button should take you to the same page as the add team button, however, it should overwrite the team that you chose to edits’ details. | It took me to the correct layout and instead of creating a new team it overwrote the team that I chose to edit. |
| 9 | Delete Button | Should delete the team and its details from the database. | Deleted the team and its details from the database. |
| 10 | Add Players | The player that is meant to be added should be added and then show up into the players table for a specific team. | The player that was added to a specific team was added to the players table and visible. |
| 11 | Display Players | When clicking the display players button the players table for a specific team. | The button showed all of the players for a specific team. |
| 12 | Delete Button (Part 2) | Should delete the entire list of players from the database. | Deleted the entire list of players from the database. |
| 13 | Players Recycler View | The recycler view should move up and down in conjunction with the users swiping. | The recycler view moved up and down in conjunction with the user’s swipes. |
| 14 | Teams Recycler View | The recycler view should move up and down in conjunction with the users swiping. | The recycler view moved up and down in conjunction with the user’s swipes. |
| 15 | Resume Application | When going off the application and resuming via the recent application function everything should be exactly as the user left it. | When resuming the application after leaving it running in the background it remained exactly as the user left it. |
| 16 | Data Remains after shutdown | When closing the application, on restart all data should still remain. | All data that had been entered beforehand was still there on restart. |

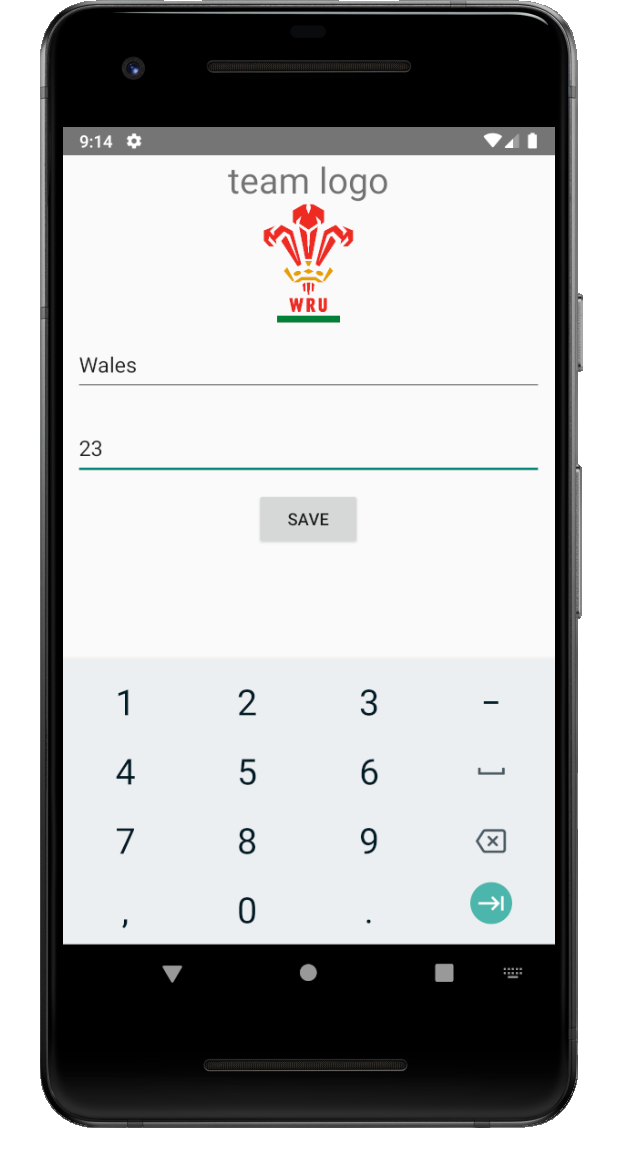
 Test 1 Test 2

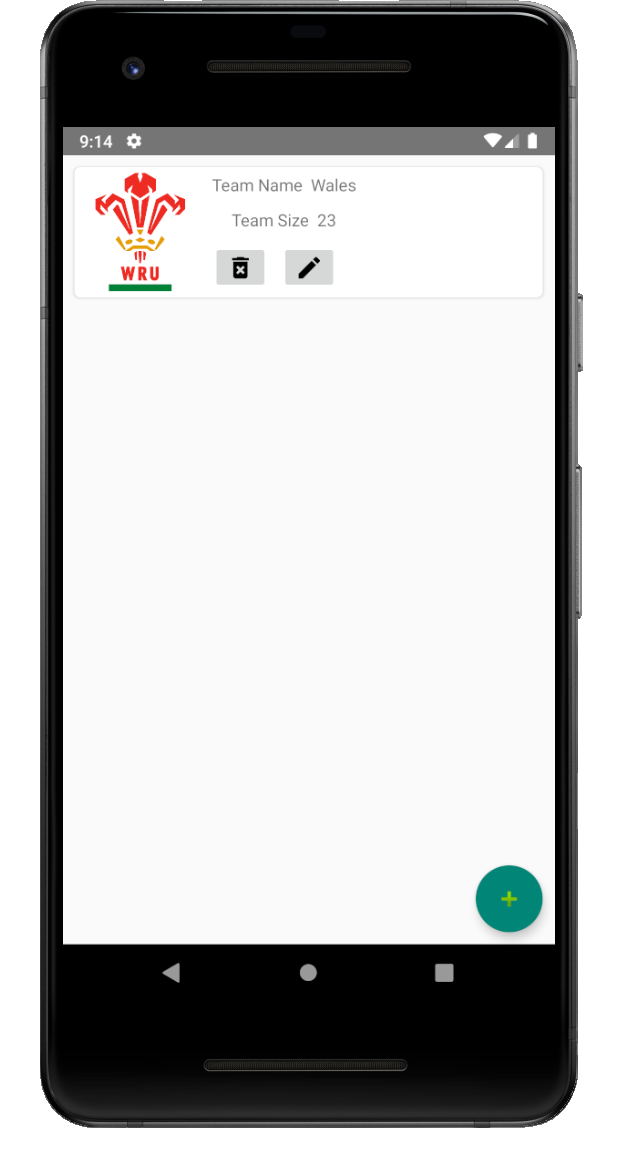
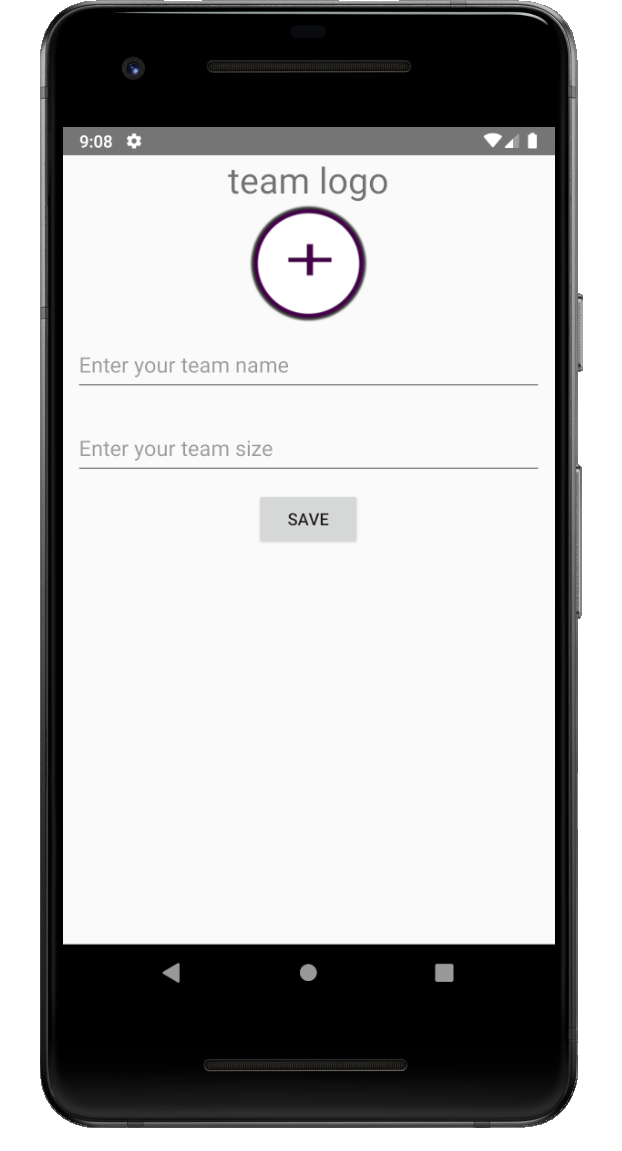


 Test 3 Test 4

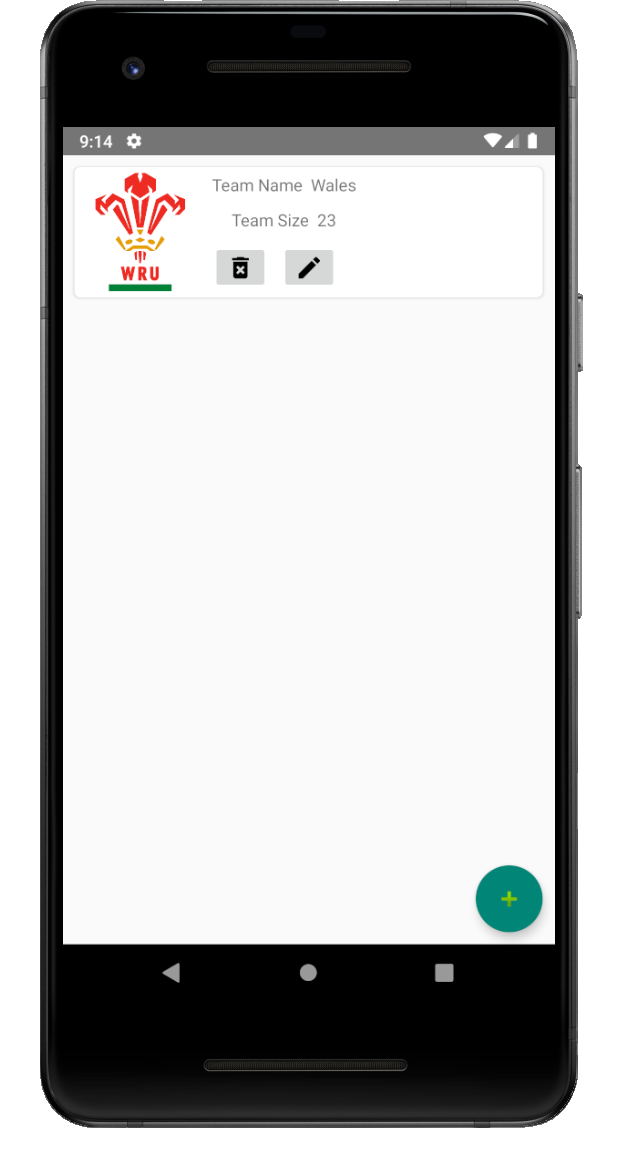


 Test 5 Test 6

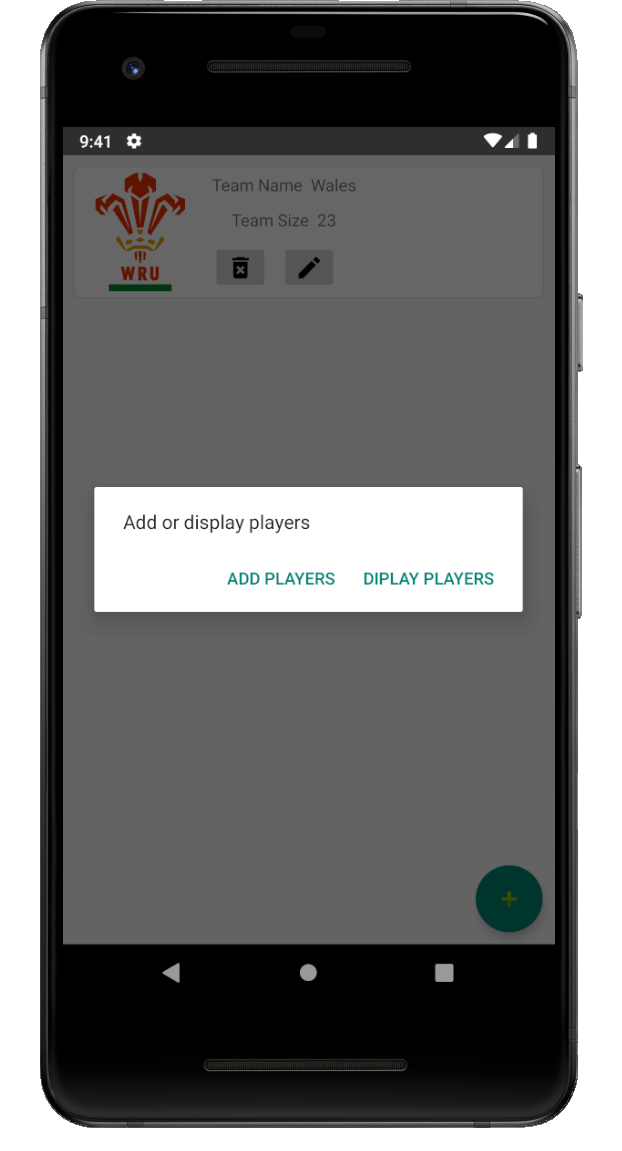
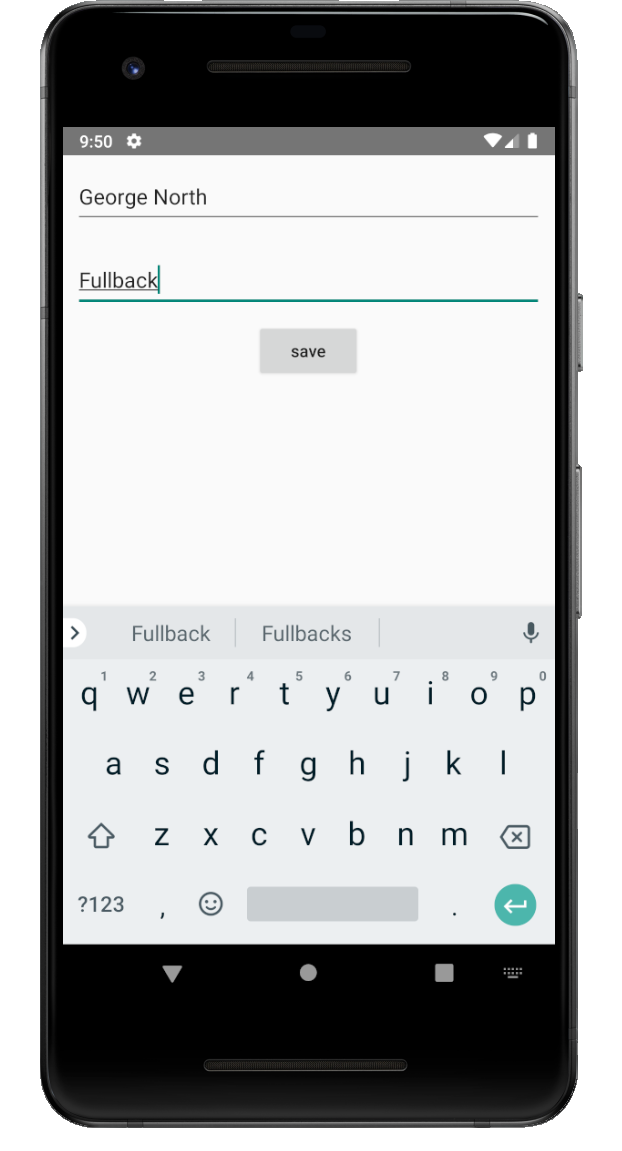
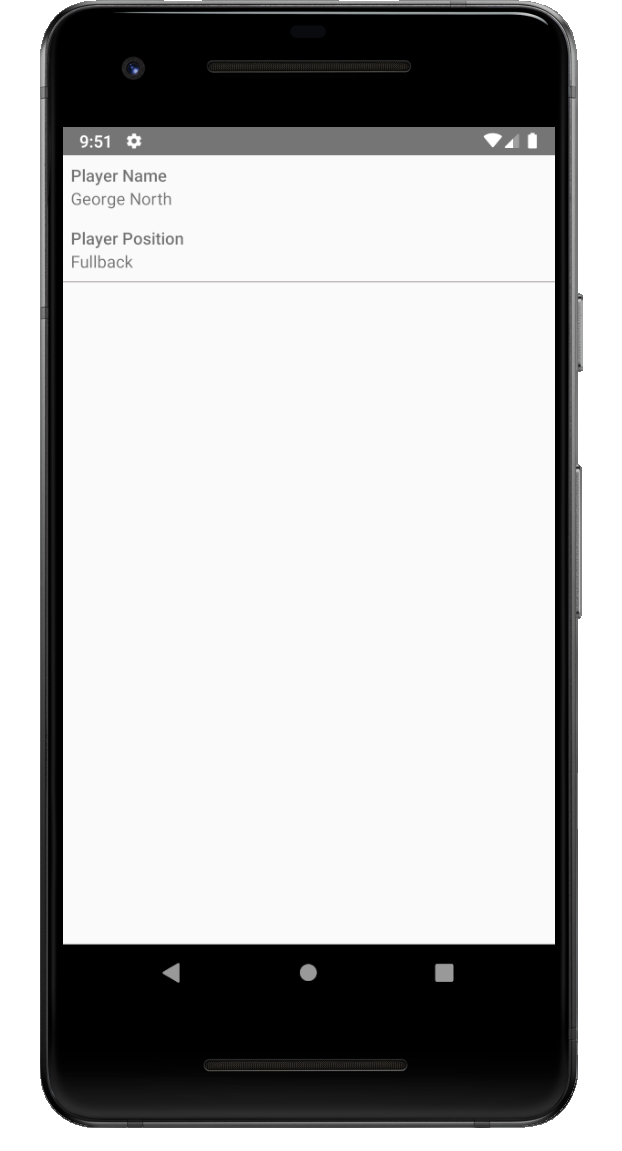
 Test 7 Test 8



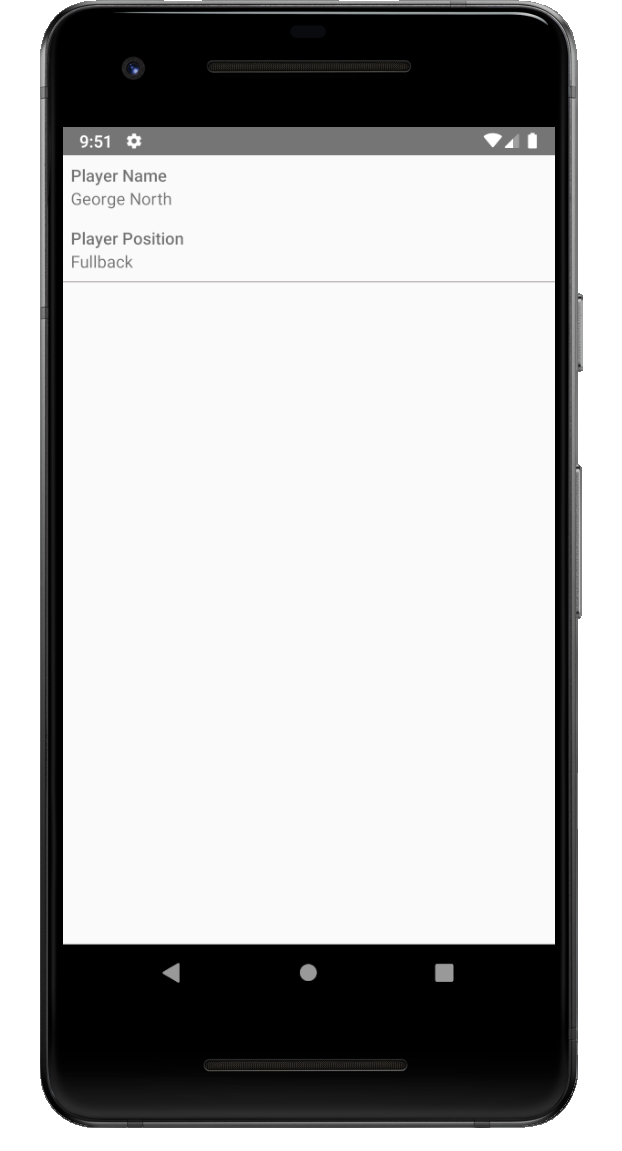
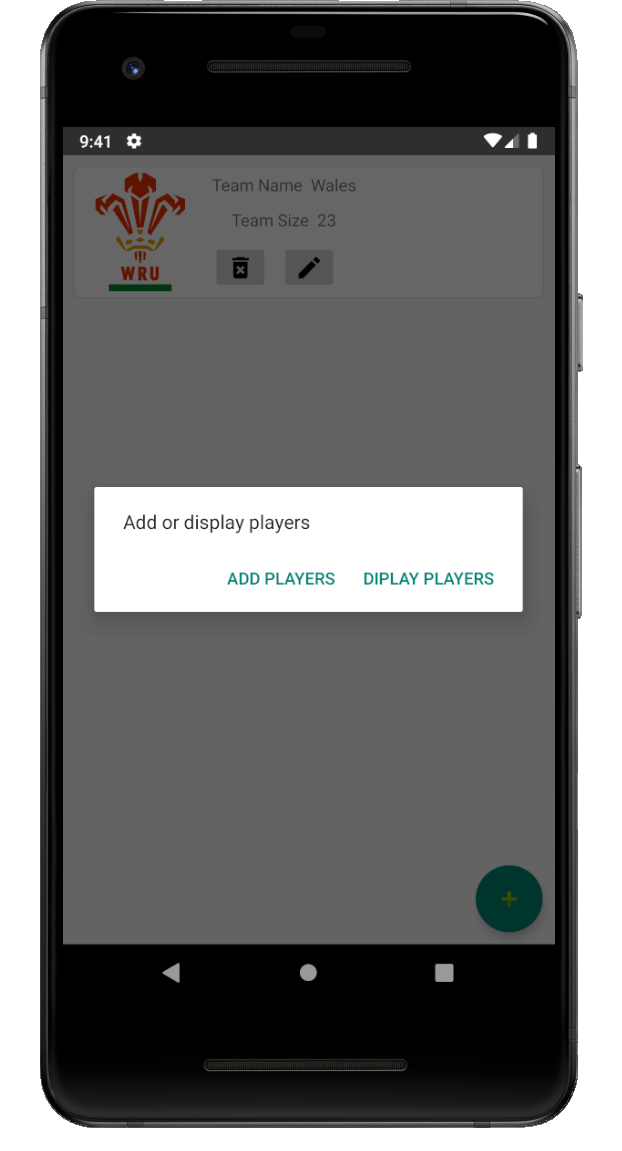
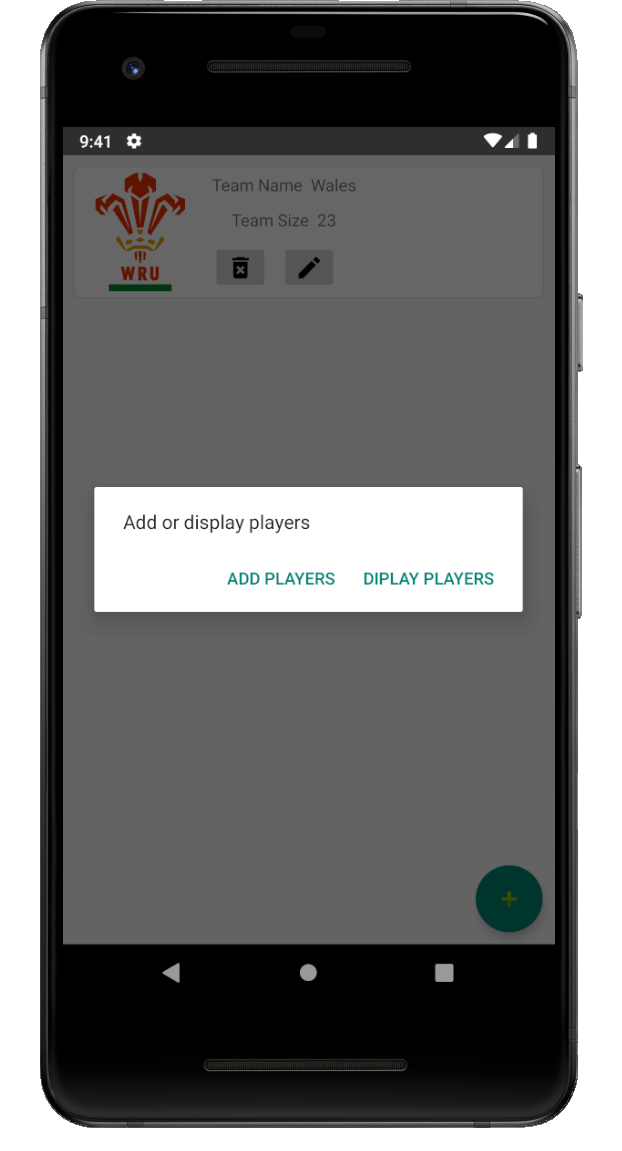
Test 9

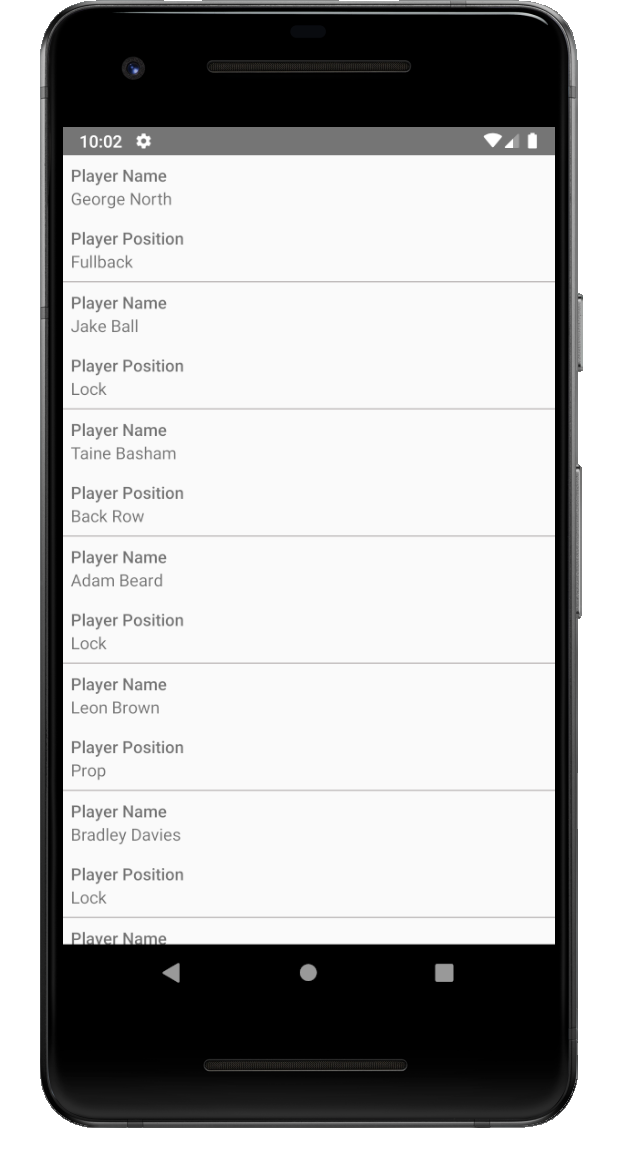
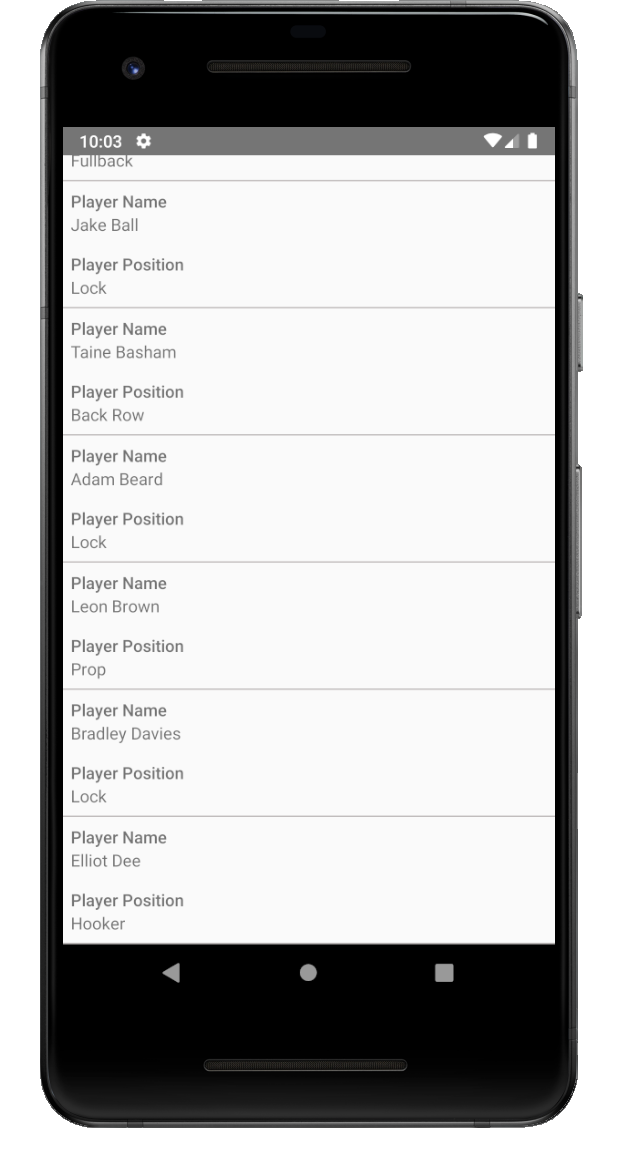


Test 10 & 11

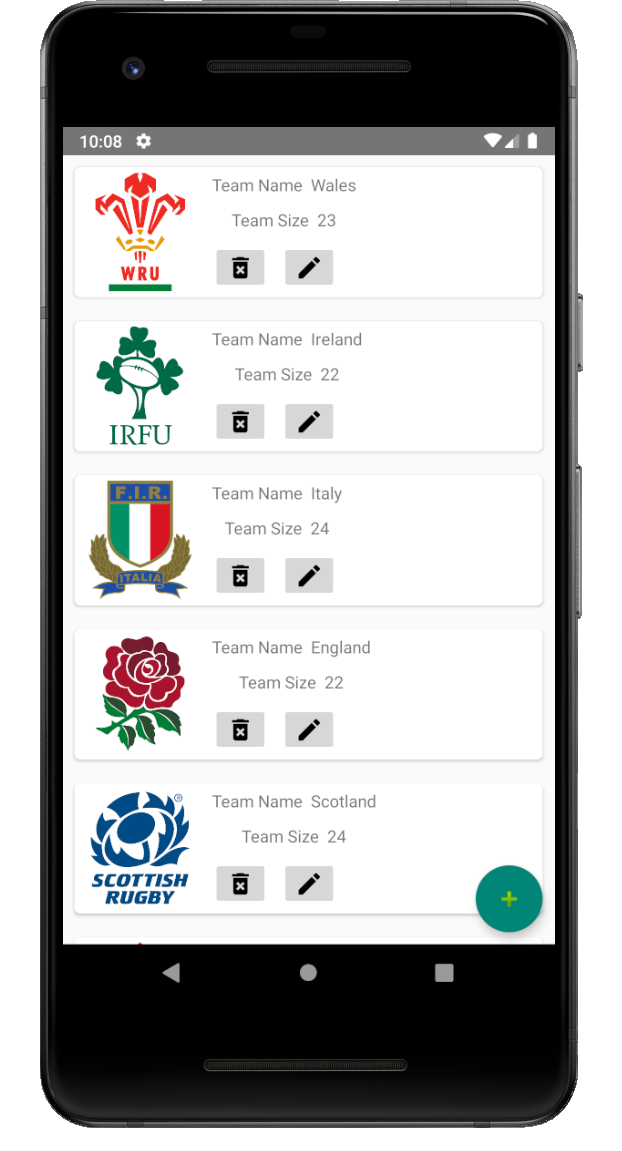
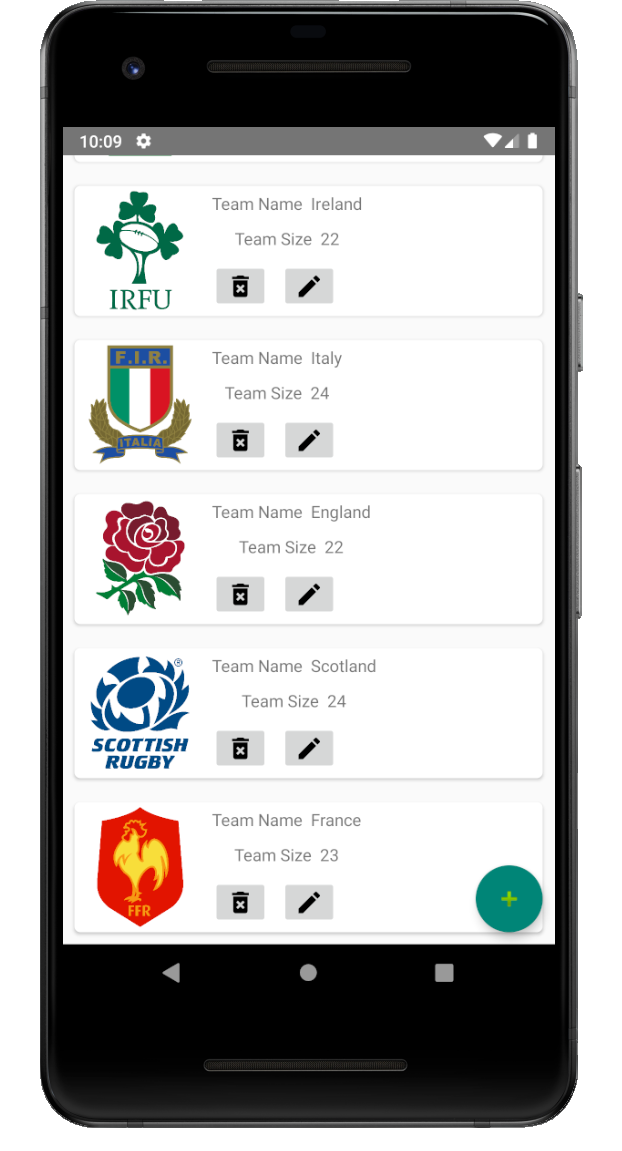


Test 12

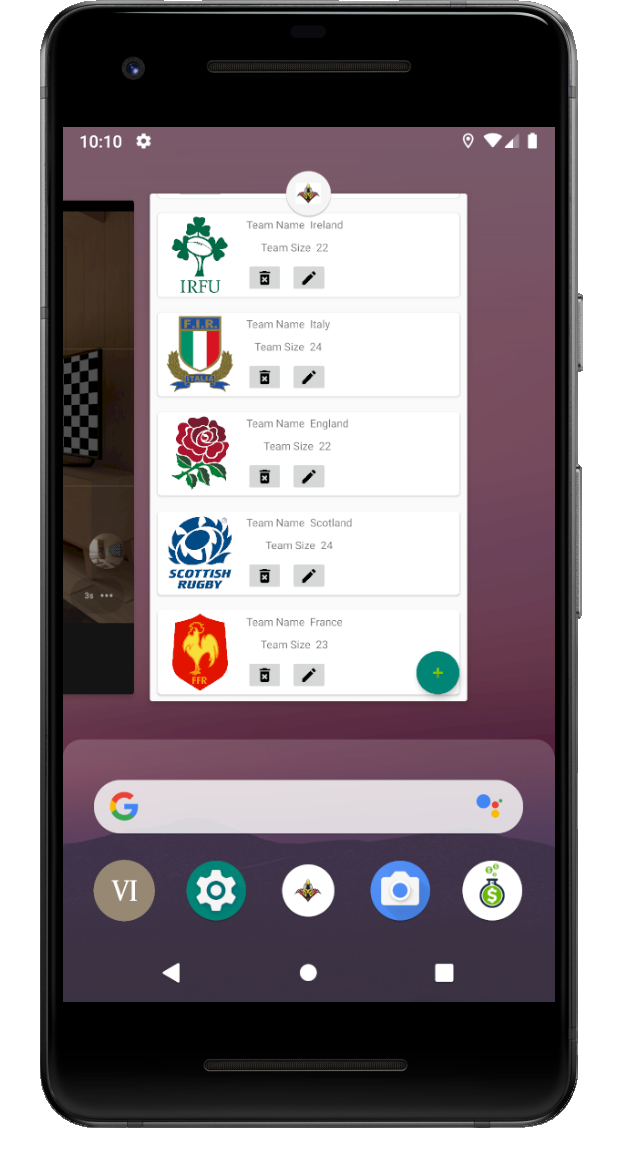
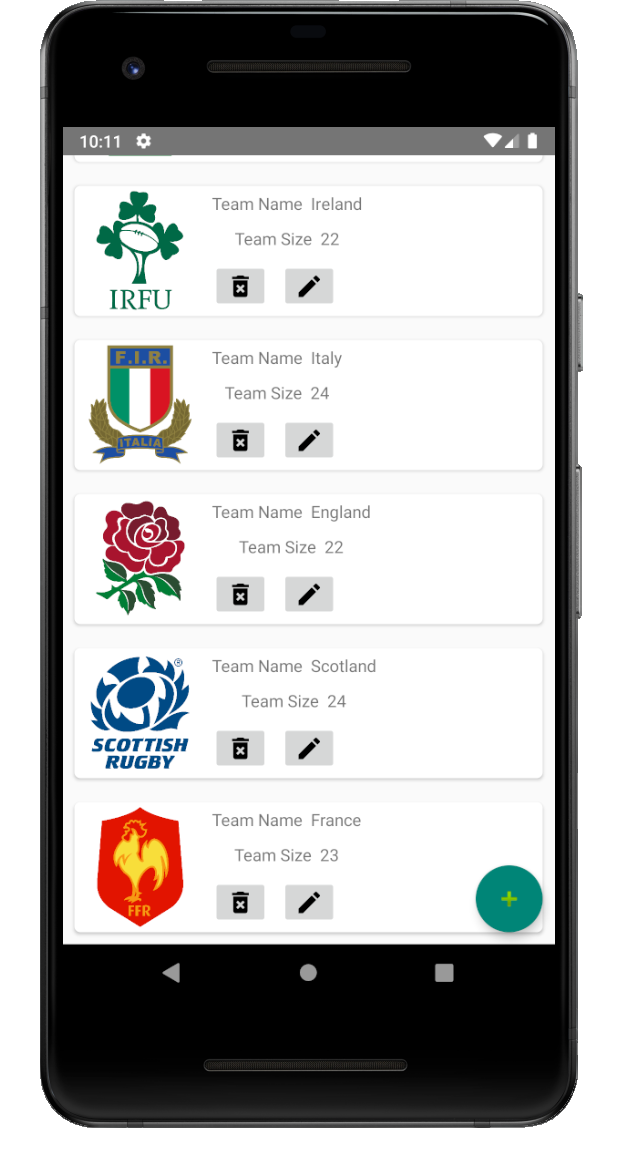


Test 13

Test 14



Test 15



Test 16

