**REPORT**

**Mobile Programming Challenge-**

**Android**

SUBMITTED BY: SWATI JHA

Table of Contents

[Project Requirements: 3](#_Toc528023530)

[Design Concept: 4](#_Toc528023531)

[Technologies Used: 6](#_Toc528023532)

[Database schema: 6](#_Toc528023533)

[Project Structure: 7](#_Toc528023534)

[Classes: 8](#_Toc528023535)

## **Project Requirements:**

1. Use​ ​Firebase​ ​as​ ​your​ ​REST​ ​API​ ​and​ ​database.​ ​​http://firebase.io/

2. You​ ​will​ ​need​ ​to​ ​create​ ​a​ ​list​ ​view​ ​of​ ​profiles.  
3. Each​ ​profile​ ​should​ ​have​ ​all​ ​of​ ​the​ ​following​ ​information:

a. A unique integer ID

b. Gender (Male/Female)

c. Name

d. Age

e. Profile Image

f. Hobbies

4. Each​ ​profile​ ​should​ ​display​ ​its​ ​respective​ ​background​ ​color

a. Males​ ​profiles​ ​should​ ​have​ ​a​ ​blue​ ​background  
b. Female​ ​profiles​ ​should​ ​have​ ​a​ ​pink​ ​background

5. A​ ​user​ ​should​ ​be​ ​able​ ​to:  
 a. Filter​ ​the​ ​list​ ​to​ ​show​ ​only​ ​male​ ​or​ ​female​ ​profiles.

b. Clear​ ​the​ ​filter​ ​to​ ​show​ ​all​ ​profiles.  
c. Sort​ ​the​ ​list​ ​by​ ​age​ ​ascending​ ​or​ ​descending.  
d. Sort​ ​the​ ​list​ ​by​ ​name​ ​ascending​ ​or​ ​descending.  
e. Remove​ ​any​ ​sorts​ ​and​ ​go​ ​back​ ​to​ ​the​ ​default.  
f. Add​ ​a​ ​new​ ​profile​ ​using​ ​an​ ​overlay.  
g. Tap​ ​on​ ​a​ ​profile​ ​and​ ​go​ ​to​ ​a​ ​profile​ ​view.  
h. Remove​ ​a​ ​profile​ ​from​ ​the​ ​profile​ ​view.  
i. Update​ ​a​ ​profile’s​ ​hobbies​ ​from​ ​the​ ​profile​ ​view

6. The​ ​profile​ ​view​ ​should​ ​display​ ​the​ ​following​ ​information:

a. Gender (Male/Female)

b. Name

c. Age

d. Profile Image

e. Hobbies

7. The​ ​list​ ​should​ ​be​ ​sorted​ ​by​ ​ID,​ ​ascending​ ​by​ ​default.  
8. Any​ ​changes​ ​to​ ​the​ ​profiles​ ​should​ ​be​ ​reflected​ ​across​ ​all​ ​running​ ​instances​ ​of​ ​the​ ​app​ ​in

real​ ​time​ ​without​ ​requiring​ ​user​ ​interaction.

## **Design Concept:**

Through this project I have tried to create a simple and easy to use interface that allows the user to create a new profile using a dialog, view the profile list. Following are the views available in the app:

1. **Profile List View**

This is the main view of the application which will be displayed to the user on application launch. This view has the list of user profiles showing name, age, gender, hobbies and image for each profile. The list has been created using recycler view. In the profile list view page, users have options to sort and filter the list. User can filter the list based on gender by selecting the checkboxes. If female is selected, list of females will be displayed and if male is selected list of males will be displayed. If both or none are selected, all profiles will be displayed. User can sort the list based on name, age in ascending and descending order, default sort is by ID. User can create new profile by clicking on the plus icon. User can also go to profile detail view by clicking on the profile in the list view.

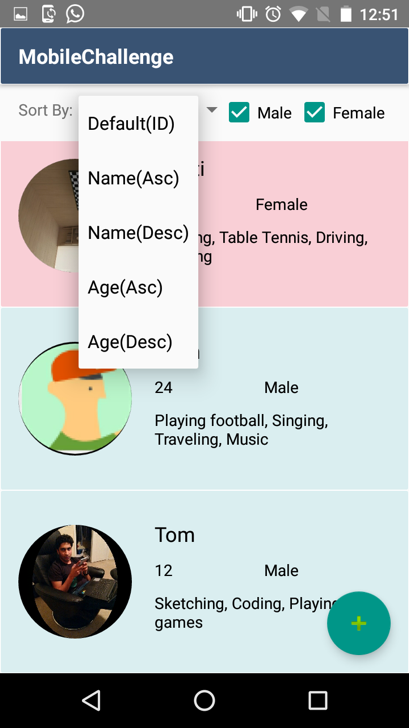
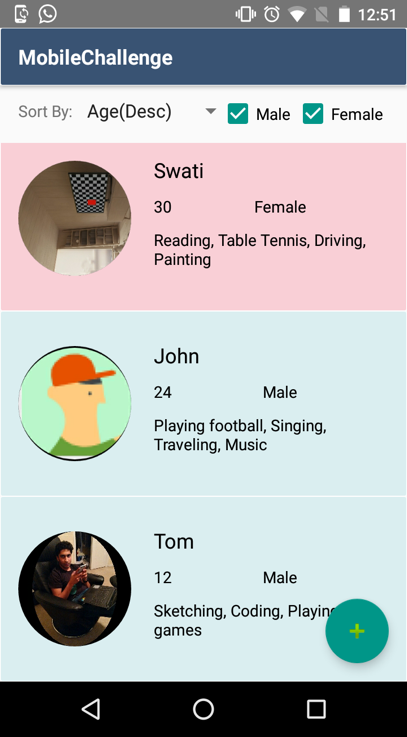


Fig 1. Profile List View

1. **Add Profile View**

By clicking on the + button, user will be shown a dialog box where he can enter the details of the new profile, select profile picture from galary and save it by clicking on the save button. The overlay is created using DialogFragment. If the user clicks on the cancel button, dialog box will be closed, and user will be navigated to the profile list view. The images are stored in the firebase storage.

The following validations are done before creating the profile in the database.

1. User has entered name
2. User has entered age
3. User has entered hobbies
4. User has selected gender

When the validations are done, profile is created in the database. If the user has not uploaded a profile image, default image is displayed.

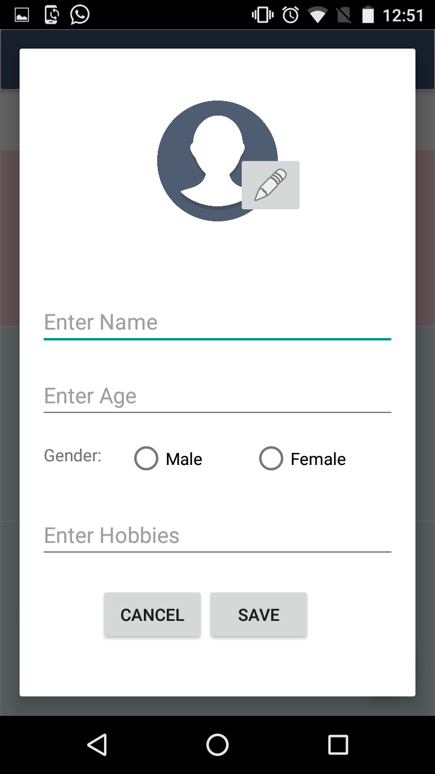


Fig 2. Add Profile View

1. **Profile Detail View**

The user can view the profile detail by clicking on the any of the profiles in the profile list view. The details of the profile like image, name, age, gender, hobbies are displayed on the detail page. User can click on the hobbies and edit it as well by clicking on the update button. User also has an option to delete profile. If the user clicks on the delete button a confirmation alert dialog is shown, if the user clicks on the yes button, the profile is deleted. If the user clicks on the no button, the action is discarded. The validation for hobbies not empty is done when the user clicks on the update button.

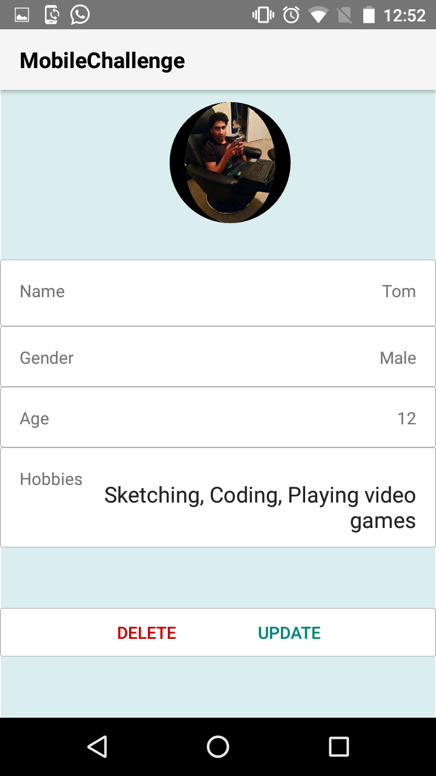


Fig 3. Profile Detail View

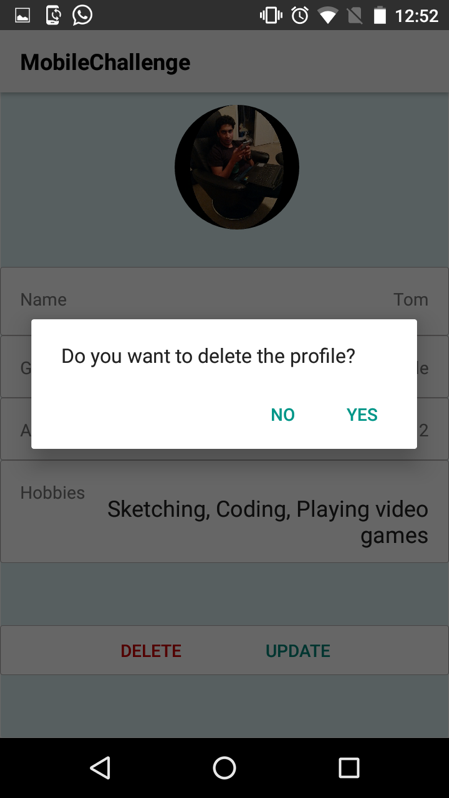


Fig 4. Profile Delete Confirmation

## **Technologies Used:**

Mobile Application – Android (Java)

Database – Google Firebase Realtime database, Firebase storage

Development IDE – Android Studio 3.2

3rd party Library – Picasso to load the images.

## **Database schema:**

**User Details**

**long \_id**;  
String **userName**;  
**long age**;  
String **gender**;  
String **userImage**;  
String **hobbies**;

{

"\_id" : 1,

"age" : 27,

"gender" : "Female",

"hobbies" : "Table Tennis, Swimming, Singing, Travelling, Reading",

"userImage" : "https://firebasestorage.googleapis.com/v0/b/mobilechallenge-59b2a.appspot.com/o/3\_photo.jpg?alt=media&token=5ff7a0cf-454d-47e1-ab48-14a769cea01a",

"userName" : "Swati Jha"

}

## **Project Structure:**

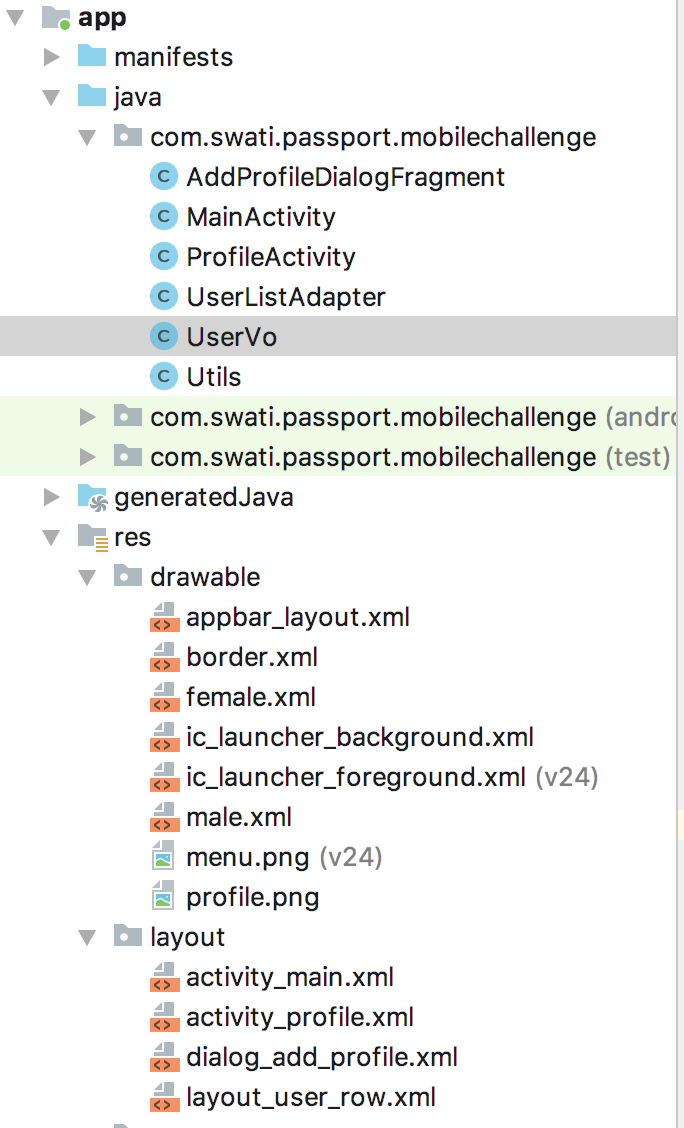
****

Fig 5. Code Structure

## **Classes:**

1. **MainActivity**

This class is the activity launcher class. This class inflates activity\_main layout. It contains the Spinner to display the sort list. It contains the checkboxes to filter the list based on gender.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*//Initializing views***btnAddProfile** = findViewById(R.id.***btnAddProfile***);  
**btnAddProfile**.setOnClickListener(**new** OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 showDialog();  
 }  
});  
**cbFemale** = findViewById(R.id.***checkBoxF***);  
**cbMale** = findViewById(R.id.***checkBoxM***);  
**cbFemale**.setOnClickListener(**this**);  
**cbMale**.setOnClickListener(**this**);  
**userList** = **new** ArrayList<UserVo>();  
**mRecyclerView** = (RecyclerView) findViewById(R.id.***userRecyclerView***);  
**mLayoutManager** = **new** LinearLayoutManager(**this**);  
**mRecyclerView**.setLayoutManager(**mLayoutManager**);  
  
*//Spinner for selecting sort parameter*Spinner spinner = (Spinner) findViewById(R.id.***spinner***);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

It also has firebase real-time database reference and listener. It fetches the list of user profiles on start of the activity and removes the listener on stop of the activity.

ValueEventListener userListener = **new** ValueEventListener() {  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
 **userList** = **new** ArrayList<UserVo>();  
 **for** (DataSnapshot noteDataSnapshot : dataSnapshot.getChildren()) {  
 UserVo user = noteDataSnapshot.getValue(UserVo.**class**);  
 **if**(!**filters**.get(*GENDER\_KEY*).equalsIgnoreCase(*GENDER\_ALL*)){  
 **if**(**filters**.get(*GENDER\_KEY*).equalsIgnoreCase(*GENDER\_F*) && user.getGender().equalsIgnoreCase(*GENDER\_F*)){  
 **userList**.add(user);  
 } **else if**(**filters**.get(*GENDER\_KEY*).equalsIgnoreCase(*GENDER\_M*) && user.getGender().equalsIgnoreCase(*GENDER\_M*)){  
 **userList**.add(user);  
 }  
 }**else**{  
 **userList**.add(user);  
 }  
 }  
 sortUserList();  
  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError) {  
 Log.*d*(**"getUserList:onCancelled"**, databaseError.toException().toString());  
  
 }  
  
};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

It also has a floating button, on click of which the DialogFragment to create the profile is launched.

**void** showDialog() {  
 FragmentTransaction ft = getFragmentManager().beginTransaction();  
 Fragment prev = getFragmentManager().findFragmentByTag(**"dialog"**);  
 **if** (prev != **null**) {  
 ft.remove(prev);  
 }  
 ft.addToBackStack(**null**);  
  
 FragmentManager fm = getSupportFragmentManager();  
 AddProfileDialogFragment addProfileDialogFragment = AddProfileDialogFragment.*newInstance*();  
 addProfileDialogFragment.show(fm, **"dialog"**);  
  
  
}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

It also has a interface implementation to save the user profile. Since firebase is ordered, so while creating new user in the database, the user id of the last user is fetched and incremented by 1 to create new user id.

@Override  
**public void** onDialogSaveClick(DialogFragment dialog, **final** UserVo user, **final** ImageView mProfileImage) {  
 Query lastQuery = **mDatabase**.orderByKey().limitToLast(1);  
 lastQuery.addListenerForSingleValueEvent(**new** ValueEventListener() {  
 **long key** = 1;  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
 **for** (DataSnapshot noteDataSnapshot : dataSnapshot.getChildren()) {  
 UserVo userVo = noteDataSnapshot.getValue(UserVo.**class**);  
 **key** = userVo.get\_id() + 1;  
 }  
 user.set\_id(**key**);  
 addUserImage(mProfileImage, user);  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError) {  
 Log.*d*(**"getLastUser:onCancelled"**, databaseError.toException().toString());  
 }  
 });  
  
 dialog.dismiss();  
}

1. **AddProfileDialogFragment**

This is the DialogFragment to create new user profile. It has views to enter user details and save and cancel button to save and cancel profile creation.

**final** View v = inflater.inflate(R.layout.***dialog\_add\_profile***, container, **false**);  
**mProfileImage** = v.findViewById(R.id.***userImage***);  
**mName** = v.findViewById(R.id.***editTextName***);  
**mAge** = v.findViewById(R.id.***editTextAge***);  
**mHobbies** = v.findViewById(R.id.***editTextHobbies***);  
**mProfileImage** = v.findViewById(R.id.***userImage***);  
**mRadioGroup** = v.findViewById(R.id.***rGroupGender***);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

It also has function to access gallery and upload image for user profile.

*//Image upload button*ImageButton buttonLoadImage = v.findViewById(R.id.***imageButtonProfile***);  
buttonLoadImage.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View arg0) {  
 **try** {  
 **if** (ActivityCompat.*checkSelfPermission*(**mContext**, Manifest.permission.***READ\_EXTERNAL\_STORAGE***) != PackageManager.***PERMISSION\_GRANTED***) {  
 requestPermissions(**new** String[]{Manifest.permission.***READ\_EXTERNAL\_STORAGE***, Manifest.permission.***WRITE\_EXTERNAL\_STORAGE***}, *RESULT\_LOAD\_IMAGE*);  
 } **else** {  
 Intent i = **new** Intent(  
 Intent.***ACTION\_PICK***,  
 android.provider.MediaStore.Images.Media.***EXTERNAL\_CONTENT\_URI***);  
 startActivityForResult(i, *RESULT\_LOAD\_IMAGE*);  
 }  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
});

1. **UserListAdapter**

It is the adapter for recyclerview which contains the detail of a single row item in the list. The item has an onclick listener, which starts profile detail activity and passes the userid to it.

**public void** onBindViewHolder(MyViewHolder holder, **int** position) {  
 **final** UserVo user = **mDataset**.get(position);  
 **if**(user.getGender().equalsIgnoreCase(**"Female"**)){  
 holder.**itemView**.setBackgroundResource(R.drawable.***female***);  
 }**else**{  
 holder.**itemView**.setBackgroundResource(R.drawable.***male***);  
 }  
 holder.**userName**.setText(user.getUserName());  
 holder.**age**.setText(user.getAge()+**""**);  
 holder.**gender**.setText(user.getGender());  
 holder.**hobbies**.setText(user.getHobbies());  
 Picasso.*with*(holder.**userImage**.getContext()).load(user.getUserImage()).transform(**new** Utils.CircleTransform()).into(holder.**userImage**);  
 holder.**itemView**.setOnClickListener(**new** OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent intent = **new** Intent(**mContext**,ProfileActivity.**class**);  
 intent.putExtra(***USER\_ID***,String.*valueOf*(user.get\_id()));  
 **mContext**.startActivity(intent);  
 }  
 });  
}

1. **ProflileActivity**

It contains the view to display the profile details of the selected user. It gets the userId passed in the intent and fetches the user details from the firebase database using that userId. It also has a listener to listen to the changes in the database and updates the user details.

**userID** = getIntent().getStringExtra(UserListAdapter.***USER\_ID***);  
**mDatabase** = FirebaseDatabase.*getInstance*().getReference().child(**"users"**);

**mUSerQuery** = **mDatabase**.orderByChild(**"\_id"**).equalTo(Integer.*parseInt*(**userID**));

**name** = findViewById(R.id.***textViewUName***);  
**age** = findViewById(R.id.***textViewUAge***);  
**gender** = findViewById(R.id.***textViewUGender***);  
**imageViewProfile** = findViewById(R.id.***imageViewProfile***);  
  
**hobbies** = findViewById(R.id.***editTextUHobbies***);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

It contains delete and update button as well. Clicking on the delete button, presents the user with a confirmation box and if user selects yes, then profile is deleted or else cancelled. It also has an update button which updates the user hobbies, if it has been changed and is not empty.

**case** R.id.***buttonDelete***:  
 **new** AlertDialog.Builder(v.getContext())  
 .setMessage(**"Do you want to delete the profile?"**)  
 .setCancelable(**false**)  
 .setPositiveButton(**"Yes"**, **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialog, **int** id) {  
 **mDatabase**.child(**userID**).removeValue();  
 Toast.*makeText*(\_this,**"Profile deleted."**,Toast.***LENGTH\_SHORT***).show();  
 finish();  
 }  
 })  
 .setNegativeButton(**"No"**, **null**)  
 .show();  
  
 **break**;  
 *// case to update profile.***case** R.id.***buttonUpdate***:  
 String userHobbies = **hobbies**.getText().toString();  
 **if**(TextUtils.*isEmpty*(userHobbies)){  
 Toast.*makeText*(**this**,**"Please enter hobbies."**,Toast.***LENGTH\_SHORT***).show();  
 } **else if**(!**mUser**.getHobbies().equals(userHobbies)){  
 **mUser**.setHobbies(userHobbies);  
 **mDatabase**.child(**userID**).setValue(**mUser**);  
 **hobbies**.setCursorVisible(**false**);  
 **hobbies**.setKeyListener(**null**);  
 Toast.*makeText*(**this**,**"Profile updated."**,Toast.***LENGTH\_SHORT***).show();  
 }  
 **break**;

1. **Utils**

This is a util class to create rounded images.

**public static** Bitmap getBitmap(Bitmap source) {  
 **int** size = Math.*min*(source.getWidth(), source.getHeight());  
  
 **int** x = (source.getWidth() - size) / 2;  
 **int** y = (source.getHeight() - size) / 2;  
  
 Bitmap squaredBitmap = Bitmap.*createBitmap*(source, x, y, size, size);  
 **if** (squaredBitmap != source) {  
 source.recycle();  
 }  
  
 Bitmap bitmap = Bitmap.*createBitmap*(size, size, source.getConfig());  
  
 Canvas canvas = **new** Canvas(bitmap);  
 Paint paint = **new** Paint();  
 BitmapShader shader = **new** BitmapShader(squaredBitmap,  
 BitmapShader.TileMode.***CLAMP***, BitmapShader.TileMode.***CLAMP***);  
 paint.setShader(shader);  
 paint.setAntiAlias(**true**);  
  
 **float** r = size / 2f;  
 canvas.drawCircle(r, r, r, paint);  
  
 squaredBitmap.recycle();  
 **return** bitmap;  
}

**The git repo for the project is:** [**https://github.com/swati4jha/PassportMobileChallengeAndroid/tree/82b156d00a7fe77f86ed101e0dc8bf9b25589620**](https://github.com/swati4jha/PassportMobileChallengeAndroid/tree/82b156d00a7fe77f86ed101e0dc8bf9b25589620)