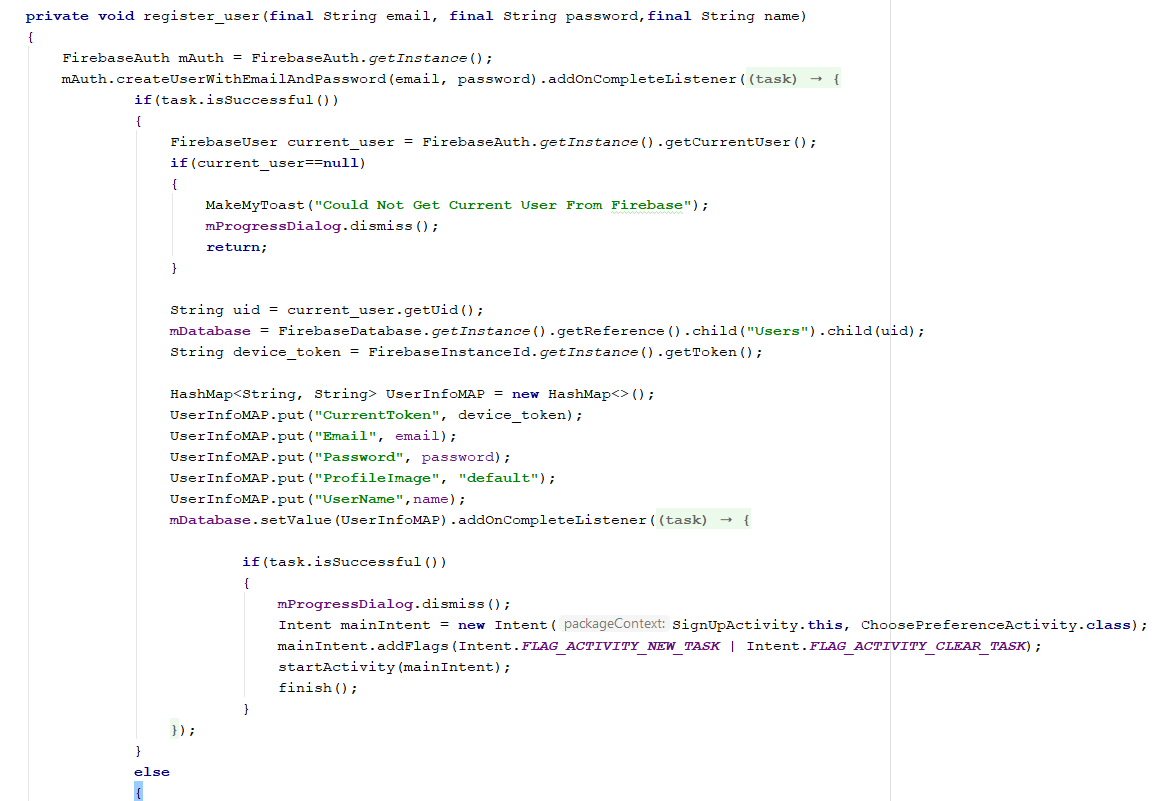
**IMPLEMENTATION**

**Table4Me ‘User App’ :**

We will look at important code snippets from each activity/fragment in the order of the application flow.

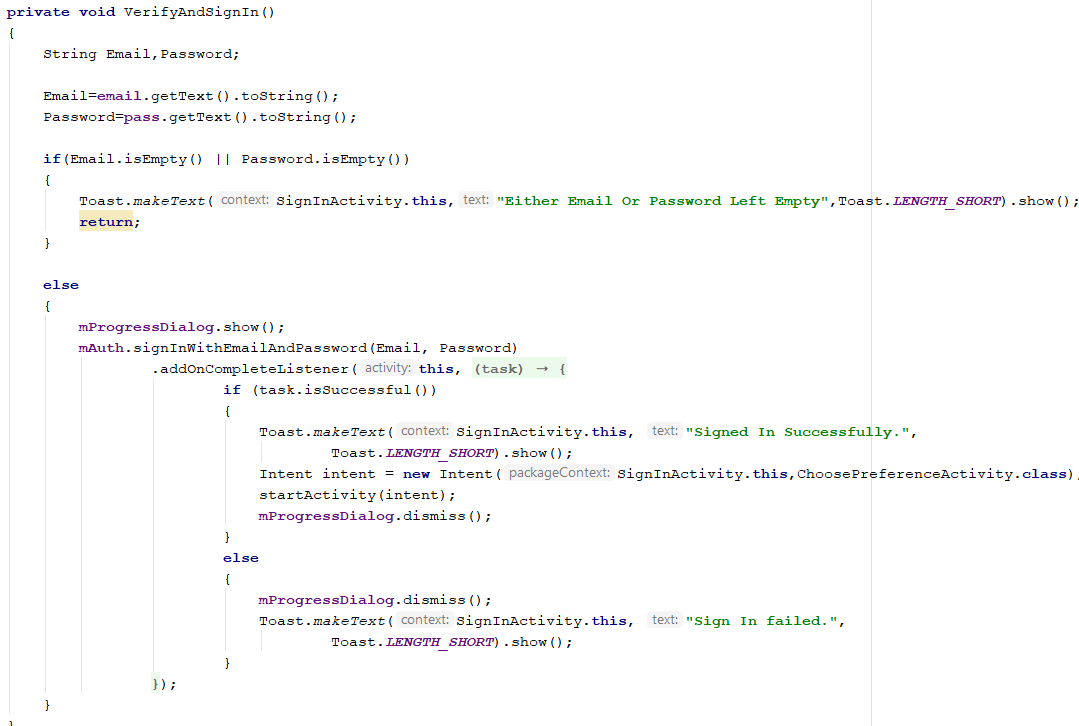
1. **Sign Up Activity**

****

This is the most important function in the signup activity. After verifying that the info provided by user in the required fields was good , that info is passed to this function to signup new user. Initially email and password are used to verify if a user already exists by the provided credentials and if this is Ok the signup starts.

After the Firebase Auth returns successful for user Signup the info is then added to Firebase Database to access whenever required.

1. **Sign In Activity**

****

This is a function provided by Firebase assistant in this function we use values that were taken as input from user , these values are checked for correct formatting and then passed to the Firebase Auth for SignIn.

1. **Preferences Activity**



This is the main activity of user app in this activity the user chooses that how to find restaurants i.e. Via location or Via cuisine type. In the onStart() function of this activity the Firebase Auth checks if the user is signed In or not and send them back if the user isn’t signed In.

1. **Choose By Location Activity**

**private void** FindNearestResturant()  
{  
  
 Log.*d*(**"mCheck->"**,**"Here 2"**);  
 **adapter** = **new** FirebaseRecyclerAdapter<myRestaurantsClass, EventsViewHolder>(myRestaurantsClass.**class**,  
 R.layout.***the\_resturant\_row***, EventsViewHolder.**class**, **mEventDataBase**)  
 {  
 */// Get the Restaurants Info from the Firebase DB and Populate them one by one in the XML via VIEWHOLDER class* @Override  
 **protected void** populateViewHolder(EventsViewHolder viewHolder, **final** myRestaurantsClass model, **int** position)  
 {  
 **mResturantProgress**.setVisibility(View.***GONE***);  
 **RestModel**.add(model);  
  
 Log.*d*(**"mCheck->"**,model.getResName());  
 **Elat** = model.getLat();  
 **Elon** = model.getLon();  
 **Dist** = **getDistance();**///////This function explained seperately  
 **if**(**Dist**>15)  
 {  
 viewHolder.SETVISIBILITY();  
 **return**;  
 }  
 *//////////////// Restaurant Data retrieved from Firebase DB and is Populated in the ViewHolder* String mdist = String.*valueOf*(**Dist**).substring(0, 4);  
 viewHolder.setTags(model.getResTags());  
 viewHolder.setName(model.getResName());  
 viewHolder.setDistance(**"Distance : "** + mdist + **" km"**);  
 **if** (**shortest** == **prev**)  
 {  
 **nearestText**.setText(**"Nearest Resturant : "** + model.getResName());  
 **shortest** = **Dist**;  
 }  
 **else if** (**Dist** < **shortest**) *// New Distance Less Than Previous Distance* {  
 **nearestText**.setText(**"Nearest Resturant : "** + model.getResName());  
 **shortest** = **Dist**;  
 }  
 **else if** (**Dist** > **shortest**) *// New Distance Greater Than Previous (No Change)* {  
 **prev** = **Dist**;  
 }  
  
 viewHolder.setResImage(model.getResImage(),ChooseByLocationActivity.**this**);  
 }

@Override  
 **public** EventsViewHolder onCreateViewHolder(**final** ViewGroup parent, **int** viewType)  
 {   
   
 *///////////////////////////////// On Click Listner for each restaurant Card//////////////////////////////////////////* EventsViewHolder viewHolder = **super**.onCreateViewHolder(parent, viewType);  
 viewHolder.setOnClickListener(**new** EventsViewHolder.ClickListener()  
 {  
 @Override  
 **public void** onItemClick(View view, **int** position)  
 {  
 Bundle bundle = **new** Bundle();  
 Intent intent = **new** Intent(ChooseByLocationActivity.**this**, ReservationActivity.**class**);  
 bundle.putString(**"From"**,**"TableChooser"**);  
 bundle.putString(**"Restaurant\_Name"**, **RestModel**.get(position).getResName());  
 bundle.putString(**"RestaurantID"**, **RestModel**.get(position).getOwnerID());  
 intent.putExtras(bundle);  
 startActivity(intent);  
 }  
  
 @Override  
 **public void** onItemLongClick(View view, **int** position)  
 {  
 }  
 });  
 **return** viewHolder;  
 }  
 };  
  
  
 **ResturantRecycler**.setAdapter(**adapter**);  
}

This is the most important function in the Find restaurants by location activity in this function the Firebase UI library is used the Database Reference is pointed to read from the VendorRestaurant node of Database and from there it receives the info of restaurants and stores them in the Viewholder class and after all the data is retrieved and populated the **FirebaseRecycler** is notified of changes.

*// Method To Calculate Distance in KM***protected float** getDistance()  
{  
 **float**[] res = **new float**[1];

//User Coordinates // Restaurants Coordinates  
 Location.*distanceBetween*(**latitude**, **longitude**, **Elat**, **Elon**, res);  
 *// This function takes into parameter the users current coordinates  
 // and the coordinates of the restaurant and then returns the distance in KM's* **float** dis = res[0];  
 dis = dis / 1000;  
 **return** dis;  
}

In this method the users coordinates are stored in the (latitude),(longitude) variables and the Restaurant Coordinates are stored in (Elat)(Elon) variables the distance b/w them is calculated and stored in the (res) variable.

1. **Choose By Cuisine Type Activity**

*//////////////////////////////////////////////////////// LOAD Restauurants /////////////////////////////////////////////////////////////////***private void** FindResturant()  
{  
 **final** ProgressDialog progressDialog = **new** ProgressDialog(ShowRestaurantsByCuisineActivity.**this**);  
 progressDialog.setTitle(**"Please Wait"**);  
 progressDialog.setCanceledOnTouchOutside(**true**);  
 progressDialog.show();  
  
 **adapter** = **new** FirebaseRecyclerAdapter<myRestaurantsClass, ShowRestaurantsByCuisineActivity.EventsViewHolder>(myRestaurantsClass.**class**,  
 R.layout.***the\_resturant\_row***, ShowRestaurantsByCuisineActivity.EventsViewHolder.**class**, **mEventDataBase**)  
 {  
 @Override  
 **protected void** populateViewHolder(EventsViewHolder viewHolder, myRestaurantsClass model, **int** position)  
 {  
 **if**(model!=**null**)  
 {   
 *///////////// Restauarnt retrieved from firebase here* **RestModel**.add(model);  
 viewHolder.setType(model.getResType());  
 viewHolder.setName(model.getResName());  
 viewHolder.setTags(model.getResTags());  
 viewHolder.setResImage(model.getResImage(),ShowRestaurantsByCuisineActivity.**this**);  
   
 *////////////// if the retrieved restaurant has the same type as the type selected in Choose by cusine activity then OK   
 /// otherwise hide that restaurant* **if**(model.getResType().equals(**Cuisine**))  
 {  
 **Count**++;  
 **restCount**.setText(String.*valueOf*(**Count**));  
 progressDialog.dismiss();  
 }  
  
 **else** {  
 viewHolder.setInVisibility();  
 progressDialog.dismiss();  
 }  
 }  
  
 }  
  
 *///////////////////////////////////////////////////////////////////////////////////////////////////////////////////* @Override  
 **public** ShowRestaurantsByCuisineActivity.EventsViewHolder onCreateViewHolder(**final** ViewGroup parent, **int** viewType)  
 {  
 ShowRestaurantsByCuisineActivity.EventsViewHolder viewHolder = **super**.onCreateViewHolder(parent, viewType);  
 *//////////////////////////////// ///////////Click Listner ///////////////////////////////////////////////////* viewHolder.setOnClickListener(**new** ChooseByLocationActivity.EventsViewHolder.ClickListener()  
 {  
 @Override  
 **public void** onItemClick(View view, **int** position)  
 {  
 Bundle bundle = **new** Bundle();  
 Intent intent = **new** Intent(ShowRestaurantsByCuisineActivity.**this**, ReservationActivity.**class**);  
 bundle.putString(**"From"**, **"Cuisine"**);  
 bundle.putString(**"Restaurant\_Name"**, **RestModel**.get(position).getResName());  
 bundle.putString(**"RestaurantID"**, **RestModel**.get(position).getOwnerID());  
 intent.putExtras(bundle);  
 startActivity(intent);  
 }  
  
 @Override  
 **public void** onItemLongClick(View view, **int** position) {  
 }  
 });  
 **return** viewHolder;  
 }  
  
 };  
  
 **ResturantRecycler**.setAdapter(**adapter**);  
}

This function retrieves restaurants from Firebase Database and shows them easily with the help of Firebase UI. Since the Firebase Database currently doesn’t allow filtering of search results i.e. Queries like in SQL therefore in the populateViewHolder() method the restaurant whose type is not the same as the type User requested for then the Card of that Restaurant is Received From Database But Not Shown to User.

1. **Reservation Activity**

**if**(getIntent().getExtras()!=**null**)  
{  
 Bundle extras = getIntent().getExtras();  
  
 **from** = getIntent().getStringExtra(**"From"**);

**if**(**from**.equals(**"TableChooser"**))  
 {  
 TableNum = extras.getString(**"TableNumber"**);  
 RestaurantID=extras.getString(**"RestaurantID"**);  
 mSeats=extras.getString(**"Seats"**);  
 Restaurant=extras.getString(**"Restaurant\_Name"**);

///////// In If Part the data is received from Table Chooser Activity////////  
 }  
  
 **else** {

//////// The bundle received by selecting Restaurant card//////////////////  
 **Restaurant**=getIntent().getStringExtra(**"Restaurant\_Name"**);  
 **RestaurantID**=getIntent().getStringExtra(**"RestaurantID"**);  
 **title**.setText(**Restaurant**+**" Reservation"**);  
 }  
  
}

At the start of activity a Bundle is received which contains Name and ID of the restaurant selected (From Firebase DB) and then used accordingly.

**private void** MakeReservation()  
{  
 String uid = **mAuth**.getCurrentUser().getUid();  
 *//////// Make a Node In Database For Reservations that will have subnode of the Restaurant that got the booking  
 /// and that will have another child node that will be the Reservation Maker (User)  
 /// 1 booking per USER per Restaurant* DatabaseReference mDatabase = FirebaseDatabase.*getInstance*().getReference().child(**"Reservations"**)  
 .child(**RestaurantID**)  
 .child(uid);  
  
 **final** ProgressDialog mProgressDialog = **new** ProgressDialog(ReservationActivity.**this**);  
 mProgressDialog.setTitle(**"Making Your Reservation"**);  
 mProgressDialog.setCancelable(**false**);  
 mProgressDialog.show();  
  
 HashMap<String, String> ReservationInfoMAP = **new** HashMap<>();  
 ReservationInfoMAP.put(**"ReserverEmail"**, **mAuth**.getCurrentUser().getEmail());  
 ReservationInfoMAP.put(**"RestaurantName"**, **Restaurant**);  
 Log.*d*(**"MCHECK->"**,**Restaurant**);  
 ReservationInfoMAP.put(**"Reservation\_Time"**, **theTime**);  
 ReservationInfoMAP.put(**"Table\_Number"**, **TableNum**);  
 ReservationInfoMAP.put(**"Reservation\_Seats"**, **mSeats**);  
 ReservationInfoMAP.put(**"ReserverID"**,uid);  
  
 mDatabase.setValue(ReservationInfoMAP).addOnCompleteListener(**new** OnCompleteListener<Void>()  
 {  
 @Override  
 **public void** onComplete(@NonNull Task<Void> task)  
 {  
  
 **if**(task.isSuccessful())  
 {  
 Intent mainIntent = **new** Intent(ReservationActivity.**this**, ChoosePreferenceActivity.**class**);  
 startActivity(mainIntent);  
 Toast.*makeText*(ReservationActivity.**this**,**"Your Booking Was Successful"**,Toast.***LENGTH\_SHORT***).show();  
 mProgressDialog.dismiss();  
 }  
 **else** {mProgressDialog.dismiss();}  
 }  
 });  
}

This is the last function called. In this function if all the required fields for booking are okay then the firebase database Write function is called to write this info in DB using a HashMap object.

Another interesting part to look here is the DatabaseReference object :

DatabaseReference mDatabase=FirebaseDatabase.*getInstance*().getReference()

.child(**"Reservations"**)

.child(**RestaurantID**)  
 .child(uid);

Here we are pointing our Object to the Reservations node in Firebase DB and then making a child node in the reservation node for each individual restaurant i.e. each restaurant has a unique Id and the reservations made will be separated for each restaurant.

The third child “uid” contains FirebaseAuth userID and this will make a child node for each users reservation against each restaurant

1. **Table Chooser Activity**

This activity is mostly hard coded and will be made generic if required in the future.

First Lets take a look at a row of tables in the XML .

XML

<**LinearLayout  
  
 android:gravity="center"  
 android:layout\_marginTop="60dp"  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**ImageView  
 android:onClick="setReservationTable"  
 android:id="@+id/Table4"  
 android:tag="Table4,3"  
 android:src="@drawable/table\_meeting\_3"  
 android:layout\_width="80dp"  
 android:layout\_height="70dp"** />  
  
 <**ImageView  
 android:onClick="setReservationTable"  
 android:id="@+id/Table5"  
 android:tag="Table5,2"  
 android:layout\_marginLeft="50dp"  
 android:src="@drawable/table\_dinner\_2"  
 android:layout\_width="80dp"  
 android:layout\_height="70dp"** />  
  
 <**ImageView  
 android:onClick="setReservationTable"  
 android:id="@+id/Table6"  
 android:tag="Table6,3"  
 android:layout\_marginLeft="50dp"  
 android:src="@drawable/table\_meeting\_3"  
 android:layout\_width="80dp"  
 android:layout\_height="70dp"** />  
  
</**LinearLayout**>

Two things to note here are the onClick property and the tag of table.

When the Image/Table is clicked then the value of TableInfo in Activity is set equal to the tag.

This procedure is followed for all the tables.

*// Makes The tables already Reserved Unavailable For Booking***private void** setReservedTables(String ResId)  
{  
 *///////////////////////////// Make Database Reference and Point it towards Required Node* DatabaseReference mRef = FirebaseDatabase.*getInstance*().getReference().child(**"Reservations"**).child(ResId);  
 ProgressDialog progressDialog = **new** ProgressDialog(TableChooserActivity.**this**);  
 progressDialog.setTitle(**"Please Wait"**);  
 progressDialog.setCanceledOnTouchOutside(**false**);  
 progressDialog.show();  
  
 mRef.addChildEventListener(**new** ChildEventListener()  
 {  
 @Override  
 **public void** onChildAdded(DataSnapshot dataSnapshot, String s)  
 {  
 Log.*d*(**"MCHECK->"**,**"Table Retrieving 2"**);  
 *// Run Loop Until You find nodes in specifed database reference* **for**(DataSnapshot snapshot : dataSnapshot.getChildren())  
 {  
 **TableNum** = (String)dataSnapshot.child(**"Table\_Number"**).getValue();  
 Log.*d*(**"MCHECK->"**,**"Table Retrieving 3"**);  
  
 **if**(**TableNum**.equals(**"Table1"**))  
 {  
 **t1**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 **t1**.setClickable(**false**);  
 }  
 **else if**(**TableNum**.equals(**"Table2"**))  
 {  
 **t2**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 **t2**.setClickable(**false**);  
 }  
 **else if**(**TableNum**.equals(**"Table3"**))  
 {  
 **t3**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 **t3**.setClickable(**false**);  
 }  
 **else if**(**TableNum**.equals(**"Table4"**))  
 {  
 **t4**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 **t4**.setClickable(**false**);  
 }  
 **else if**(**TableNum**.equals(**"Table5"**))  
 {  
 **t5**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 **t5**.setClickable(**false**);  
 }  
 **else if**(**TableNum**.equals(**"Table6"**))  
 {  
 **t6**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 **t6**.setClickable(**false**);  
 }  
 **else if**(**TableNum**.equals(**"Table7"**))  
 {  
 **t7**.setClickable(**false**);  
 **t7**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 }  
 **else if**(**TableNum**.equals(**"Table8"**))  
 {  
 **t8**.setClickable(**false**);  
 **t8**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 }  
 **else if**(**TableNum**.equals(**"Table9"**))  
 {  
 **t9**.setClickable(**false**);  
 **t9**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 }  
 **else if**(**TableNum**.equals(**"Table10"**))  
 {  
 **t10**.setClickable(**false**);  
 **t10**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 }  
 **else if**(**TableNum**.equals(**"Table12"**))  
 {  
 **t12**.setClickable(**false**);  
 **t12**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 }  
 **else if**(**TableNum**.equals(**"Table13"**))  
 {  
 **t13**.setClickable(**false**);  
 **t13**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***round\_edge\_btn***));  
 }  
 }  
 }  
  
 @Override  
 **public void** onChildChanged(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onChildRemoved(DataSnapshot dataSnapshot) {  
  
 }  
  
 @Override  
 **public void** onChildMoved(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError) {  
  
 }  
 });  
  
 progressDialog.dismiss();  
}

This function checks the tables that are already booked. It takes into parameter the restaurantID and then goes to the restaurantId node inside the “Reservations” node of FirebaseDB , there it checks the already reserved tables for that restaurant and then marks those table with a red circle from a custom XML in drawable.

**public void** setReservationTable(View view)  
{  
 **if**(!**selected**)  
 {  
 **selected**=**true**;  
 **linearLayout**.setBackgroundColor(getResources().getColor(R.color.***colorOffWhite***));  
 ImageView Table = (ImageView) view;  
 **SeatingInfo** = view.getTag().toString();  
  
 **int** Seperator=**SeatingInfo**.indexOf(**","**);  
 **a**=**SeatingInfo**.substring(0,Seperator);  
 **b**=**SeatingInfo**.substring(Seperator+1,**SeatingInfo**.length());  
 Toast.*makeText*(TableChooserActivity.**this**,**"Table Selected\nTable Number : "**+**a**+**"\nSeats : "**+**b**,  
 Toast.***LENGTH\_SHORT***).show();  
  
 Table.setBackgroundDrawable(getResources().getDrawable(R.drawable.***my\_reserved\_table***));  
  
 }  
  
 **else** {  
 Toast.*makeText*(TableChooserActivity.**this**,**"Please Deselect Table To Choose New Table"**,Toast.***LENGTH\_SHORT***).show();  
 }  
}

This is the function that was mentioned above. It is the onClick function of the tables/Imageview it selects the tables that are available with a green circle from custom XML.

**USER APP IMPLEMENTATION END**

ADMIN APP Implementation

1. NewVendorActivity:

*This is the admin signup activity*

*///////////////////////// THis method is used to Signup the Restaurant Vendors//////////////////////////////////***private void** register\_user(**final** String email, **final** String password)  
{  
 **mAuth**.createUserWithEmailAndPassword(email, password).addOnCompleteListener(**new** OnCompleteListener<AuthResult>()  
 {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task)  
 {  
  
 **if**(task.isSuccessful())  
 {  
 FirebaseUser current\_user = FirebaseAuth.*getInstance*().getCurrentUser();  
 **if**(current\_user==**null**)  
 {  
 MakeMyToast(**"Could Not Get Current User From Firebase"**);  
 **mProgressDialog**.dismiss();  
 **return**;  
 }  
 String uid = current\_user.getUid();  
 **mDatabase** = FirebaseDatabase.*getInstance*().getReference().child(**"Vendors"**).child(uid);  
 String device\_token = FirebaseInstanceId.*getInstance*().getToken();  
 HashMap<String, String> UserInfoMAP = **new** HashMap<>();  
 UserInfoMAP.put(**"CurrentToken"**, device\_token);  
 UserInfoMAP.put(**"VendorID"**, uid);  
 UserInfoMAP.put(**"Email"**, email);  
 UserInfoMAP.put(**"Password"**, password);  
 UserInfoMAP.put(**"Approval"**, **"false"**);  
  
 **mDatabase**.setValue(UserInfoMAP).addOnCompleteListener(**new** OnCompleteListener<Void>()  
 {  
 @Override  
 **public void** onComplete(@NonNull Task<Void> task)  
 {  
 **if**(task.isSuccessful())  
 {  
 **mProgressDialog**.dismiss();  
 Intent mainIntent = **new** Intent(NewVendorActivity.**this**, AccountVaildityActivity.**class**);  
 mainIntent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK*** | Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 startActivity(mainIntent);  
 finish();  
 }  
 }  
 });  
 }  
 **else** {  
 **mProgressDialog**.dismiss();  
 MakeMyToast(**"SignUp Failed"**);  
 }  
 }  
 });  
}  
  
@Override  
**protected void** onStart()  
{  
 **super**.onStart();  
  
 **if**(**mAuth**.getCurrentUser()!=**null**)  
 {  
 Intent intent = **new** Intent(NewVendorActivity.**this**,AccountVaildityActivity.**class**);  
 startActivity(intent);  
 }  
}

In this activity a new vendor/restaurant owner can sign up for a new account.

Input is taken and verified afterwards passed into the FirebaseAuth signup function. When the new Vendor signs up okay , then their info is written in the Firebase DB.

In the onStart() function of this activity we see if the Vendor is already signed In and then re Route them to the next activity.

1. **Vendor SignIn Activity**

**private void** VerifyandSignin(String email,String password)  
{  
 **if**(email.isEmpty() || password.isEmpty())  
 {  
 Toast.*makeText*(AdminLoginActivity.**this**,**"Please Make Sure Email and Password are given"**,Toast.***LENGTH\_SHORT***).show();  
 **return**;  
 }  
  
 **if**(password.length()<6)  
 {  
 Toast.*makeText*(**this**,**"Password To Short"**,Toast.***LENGTH\_SHORT***).show();  
 **return**;  
 }  
 **final** ProgressDialog progressDialog = **new** ProgressDialog(AdminLoginActivity.**this**);  
 progressDialog.setTitle(**"Signing In"**);  
 progressDialog.setMessage(**"Please Wait"**);  
 progressDialog.setCanceledOnTouchOutside(**false**);  
 progressDialog.show();  
  
 **mAuth**.signInWithEmailAndPassword(email, password)  
 .addOnCompleteListener(**this**, **new** OnCompleteListener<AuthResult>()  
 {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task)  
 {  
 **if** (task.isSuccessful())  
 {  
 Intent intent = **new** Intent(AdminLoginActivity.**this**,AccountVaildityActivity.**class**);  
 startActivity(intent);  
 }  
 **else** {  
 progressDialog.dismiss();  
 Toast.*makeText*(AdminLoginActivity.**this**,**"Login Failed\nEither password or email incorrect"**,Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
}

in this function the input is taken , verified and then passed to the FIreaseAuth Sign In function. On success the vendor is directed to the next activity.

1. Account Validity Activity

**if**(mAuth.getCurrentUser()==**null**)  
{  
 Intent intent = **new** Intent(AccountVaildityActivity.**this**,NewVendorActivity.**class**);  
 startActivity(intent);  
 finish();  
}  
  
**else**{  
 **final** String uid = mAuth.getCurrentUser().getUid();  
 DatabaseReference mRef = FirebaseDatabase.*getInstance*().getReference().child(**"Vendors"**).child(uid);  
 mRef.addValueEventListener(**new** ValueEventListener()  
 {  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot)  
 {  
 **for**(DataSnapshot snapshot : dataSnapshot.getChildren())  
 {  
 String UID =(String) dataSnapshot.child(**"VendorID"**).getValue();  
 String Approval = (String)dataSnapshot.child(**"Approval"**).getValue();  
 Log.*d*(**"Validity->"**,**"UID>"**+UID);  
 Log.*d*(**"Validity->"**,**"UID>"**+Approval);  
  
 **if**(UID!=**null** && Approval!=**null**)  
 {  
 **if**(UID.equals(uid)&&Approval.equals(**"true"**))  
 {  
 **ApprovalProgress**.setVisibility(View.***INVISIBLE***);  
 Intent intent = **new** Intent(AccountVaildityActivity.**this**,MainActivity.**class**);  
 startActivity(intent);  
 }  
  
 **else if**(UID.equals(uid)&&Approval.equals(**"false"**))  
 {  
 **ApprovalProgress**.setVisibility(View.***INVISIBLE***);  
 TextView mtext = findViewById(R.id.***StatusText***);  
 mtext.setVisibility(View.***VISIBLE***);  
 }  
 }  
 }  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError)  
 {  
  
 }  
 });  
}

This is a very important part of the admin app as the Signup and SignIn activities are routed here . In this activity the validity of the vendors account is checked.

At time of signup a child node named “Validity” is created in the Vendors node of DB and is set to ‘false’. The Vendor can only use account when this value is set to ‘true’ by the ADMIN.

This is exactly what we are checking in this activity b reading from our Firebase DB.

OnStart function checks if the Vendor is Signed In or Not otherwise they return them to SignIn activity because if this is not checked the FirebaseAuth.getCurrentUser().getUID() will return a null value and hence cause us to point our Database Reference to a null node causing app to crash badly.

1. Check Reservations Activity

FirebaseAuth mAuth = FirebaseAuth.*getInstance*();*// Make Auth Reference*String UID = mAuth.getCurrentUser().getUid();*// Get Firebase User ID  
///////////////////////////// Make Database Reference and Point it towards Required Node*DatabaseReference mRef = FirebaseDatabase.*getInstance*().getReference().child(**"Reservations"**).child(UID);  
  
mRef.addChildEventListener(**new** ChildEventListener() {  
 @Override  
 **public void** onChildAdded(DataSnapshot dataSnapshot, String s) {  
 *// Run Loop Until You find nodes in specifed database reference* **for**(DataSnapshot snapshot : dataSnapshot.getChildren())  
 {  
 **TableNum** = (String)dataSnapshot.child(**"Table\_Number"**).getValue();  
 **Seats** = (String)dataSnapshot.child(**"Reservation\_Seats"**).getValue();  
 **ReservationTime** = (String)dataSnapshot.child(**"Reservation\_Time"**).getValue();  
 **ReserverID** = (String)dataSnapshot.child(**"ReserverID"**).getValue();  
  
 ReservationInfo mInfo = **new** ReservationInfo();  
  
 **if**(**TableNum**.equals(**"Table1"**))  
 {  
 **t1**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table2"**))  
 {  
 **t2**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table3"**))  
 {  
 **t3**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table4"**))  
 {  
 **t4**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table5"**))  
 {  
 **t5**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table6"**))  
 {  
 **t6**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table7"**))  
 {  
 **t7**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table8"**))  
 {  
 **t8**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table9"**))  
 {  
 **t9**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table10"**))  
 {  
 **t10**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table12"**))  
 {  
 **t12**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 **else if**(**TableNum**.equals(**"Table13"**))  
 {  
 **t13**.setBackgroundDrawable(getResources().getDrawable(R.drawable.***edit\_text\_capsule***));  
 }  
 }  
 }  
  
 @Override  
 **public void** onChildChanged(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onChildRemoved(DataSnapshot dataSnapshot) {  
  
 }  
  
 @Override  
 **public void** onChildMoved(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError) {  
  
 }  
});

This is the most important activity, in this activity the Vendor is shown the reservations that were made From Users App for their restaurant.

The Database Reference object is pointed towards the same “Reservations” node where the USER APP was writing at time of making reservations and Reading from Db to see the already reserved tables.

we read from our DB and set the Reserved Table variable equal to the value of the Table NUMBERS Reserved by users then ……In the IF ELSE conditions we mark a blue circle for those tables .