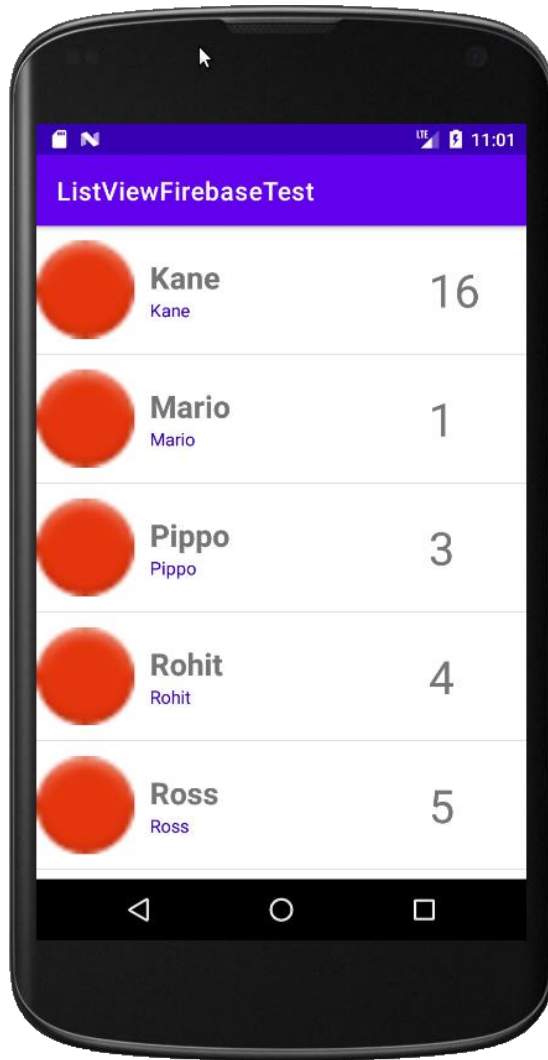




Firebase - Lab

ANDROID





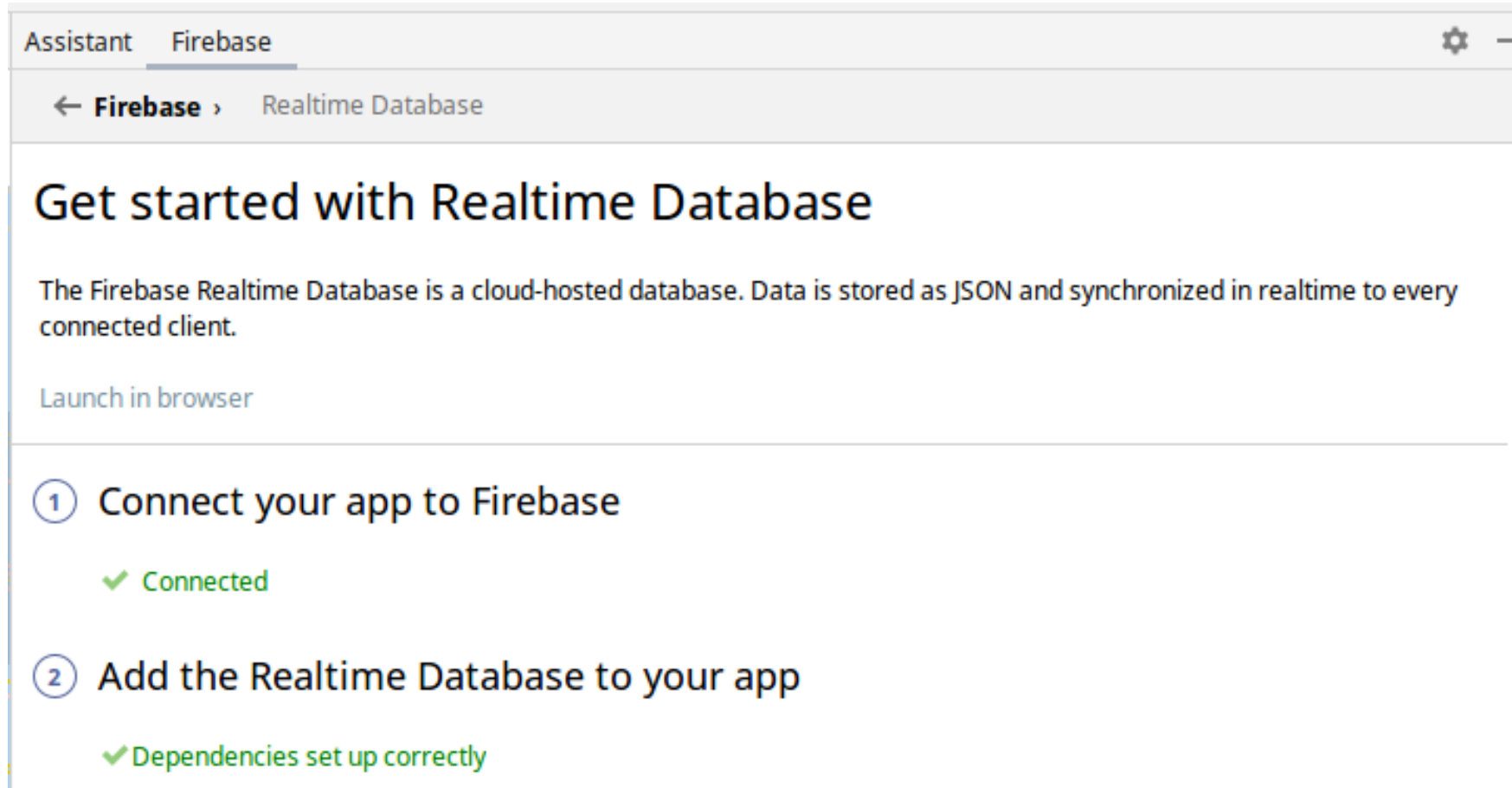
**Firestore: Save and read data**



**ANDROID**  
developer lab



# Connect your app to Firebase



# MainActivity read database

```
class MainActivity : AppCompatActivity() {
    private val TAG = "MainActivity"
    private var mUserReference: DatabaseReference? = FirebaseDatabase.getInstance().getReference("users")
    private val mUsers: MutableList<User> = ArrayList()
    private val mAdapter: MyAdapter = MyAdapter(this, mUsers)
    lateinit private var mUsersChildListener: ChildEventListener

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        list_view.adapter = mAdapter
    }

    override fun onStart() {
        super.onStart()

        val usersChildListener = getUsersChildEventListener()
        mUserReference!!.addChildEventListener(usersChildListener)
        mUsersChildListener = usersChildListener
    }

    override fun onStop() {
        super.onStop()

        if (mUsersChildListener != null) {
            mUserReference!!.removeEventListener(mUsersChildListener)
        }
    }
}
```

```
fun getUsersChildEventListener(): ChildEventListener {
    val childEventListener = object : ChildEventListener {
        override fun onChildAdded(...) {

        }
        override fun onChildChanged(...) {

        }
        override fun onChildRemoved(...) {

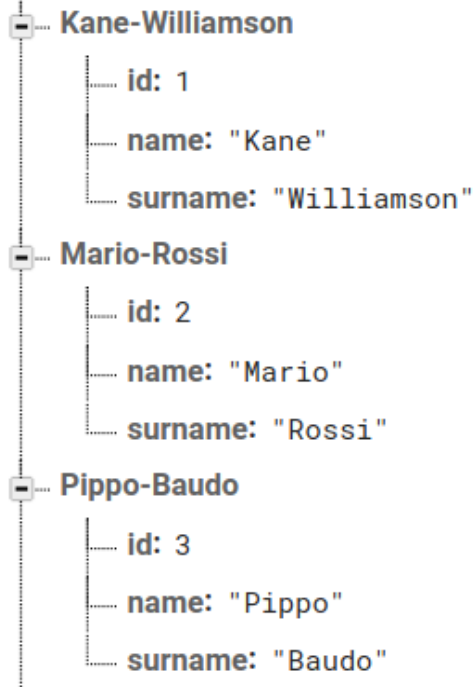
        }
        override fun onChildMoved(...) {

        }
        override fun onCancelled(...) {

        }
    }
    return childEventListener
}
```

# MainActivity read database

users



```
fun getUsersChildEventListener(): ChildEventListener{
    val childEventListener = object : ChildEventListener {
        override fun onChildAdded(dataSnapshot: DataSnapshot, previousChildName: String?) {
            val newUser = dataSnapshot.getValue(User::class.java)
            mUsers.add(newUser!!)
            mAdapter.notifyDataSetChanged()
        }
        override fun onChildChanged(dataSnapshot: DataSnapshot, previousChildName: String?) {
            val newUser = dataSnapshot.getValue(User::class.java)
            val userKey = dataSnapshot.key
            mUsers.find { e -> e.toString().equals(userKey) }?.set(newUser!!)
            mAdapter.notifyDataSetChanged()
        }
        override fun onChildRemoved(dataSnapshot: DataSnapshot) {
            val userKey = dataSnapshot.key
            var fu = mUsers.find { e -> e.toString().equals(userKey) }
            mUsers.remove(fu)
            mAdapter.notifyDataSetChanged()
        }
        override fun onChildMoved(dataSnapshot: DataSnapshot, previousChildName: String?) {
            // ...
        }
        override fun onCancelled(databaseError: DatabaseError) {
            Log.w(TAG, "postComments:onCancelled", databaseError.toException())
            Toast.makeText(this@MainActivity, "Failed to load comments.", Toast.LENGTH_SHORT).show()
        }
    }
    return childEventListener
}
```

# MainActivity read database

```
class User(var name: String, var surname: String) {  
    var id: Int = -1  
  
    constructor(): this("", "")  
    constructor(name: String, surname: String, id: Int): this(name, surname){  
        this.id = id  
    }  
  
    override fun toString(): String{  
        return "$name-$surname"  
    }  
  
    fun set(user2: User){  
        id = user2.id  
        name = user2.name  
        surname = user2.surname  
    }  
}
```

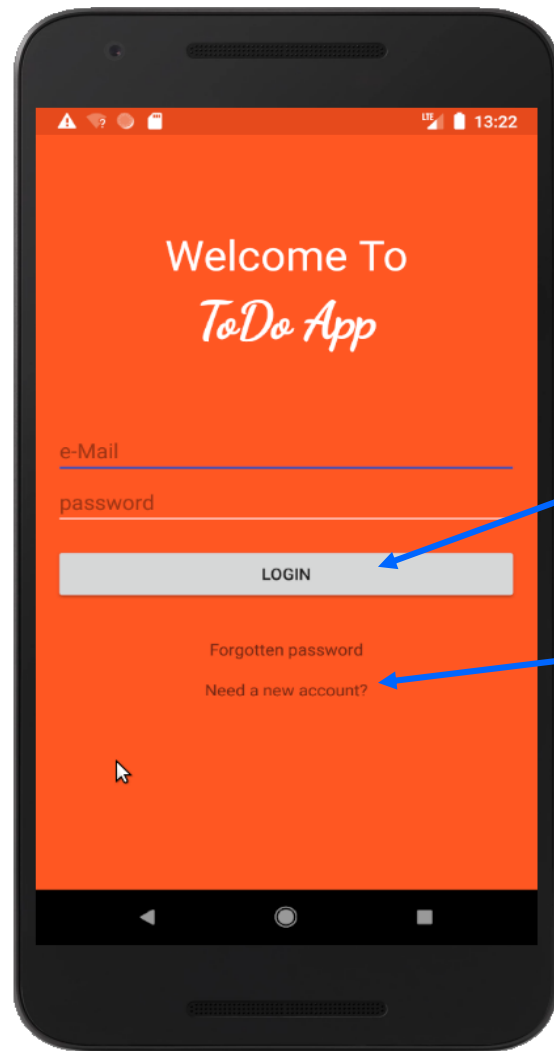
## Authentication



**ANDROID**  
developer lab

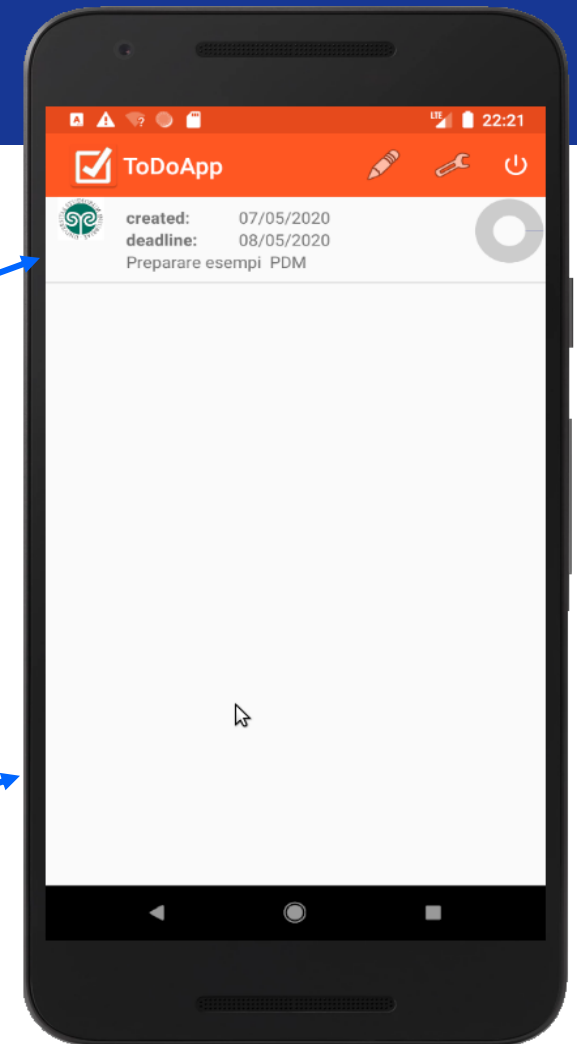
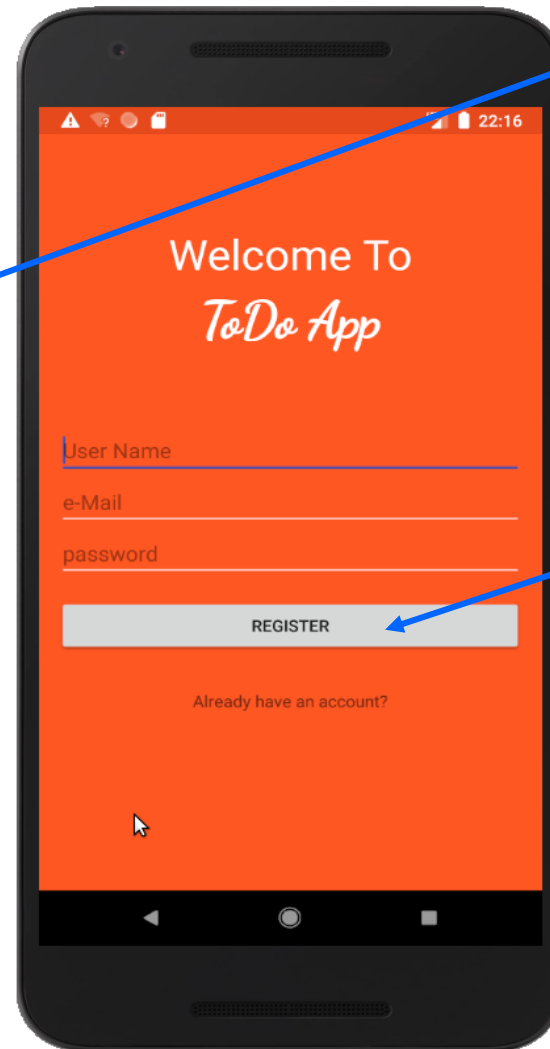


# Design the App: Kotlin version



LoginActivity

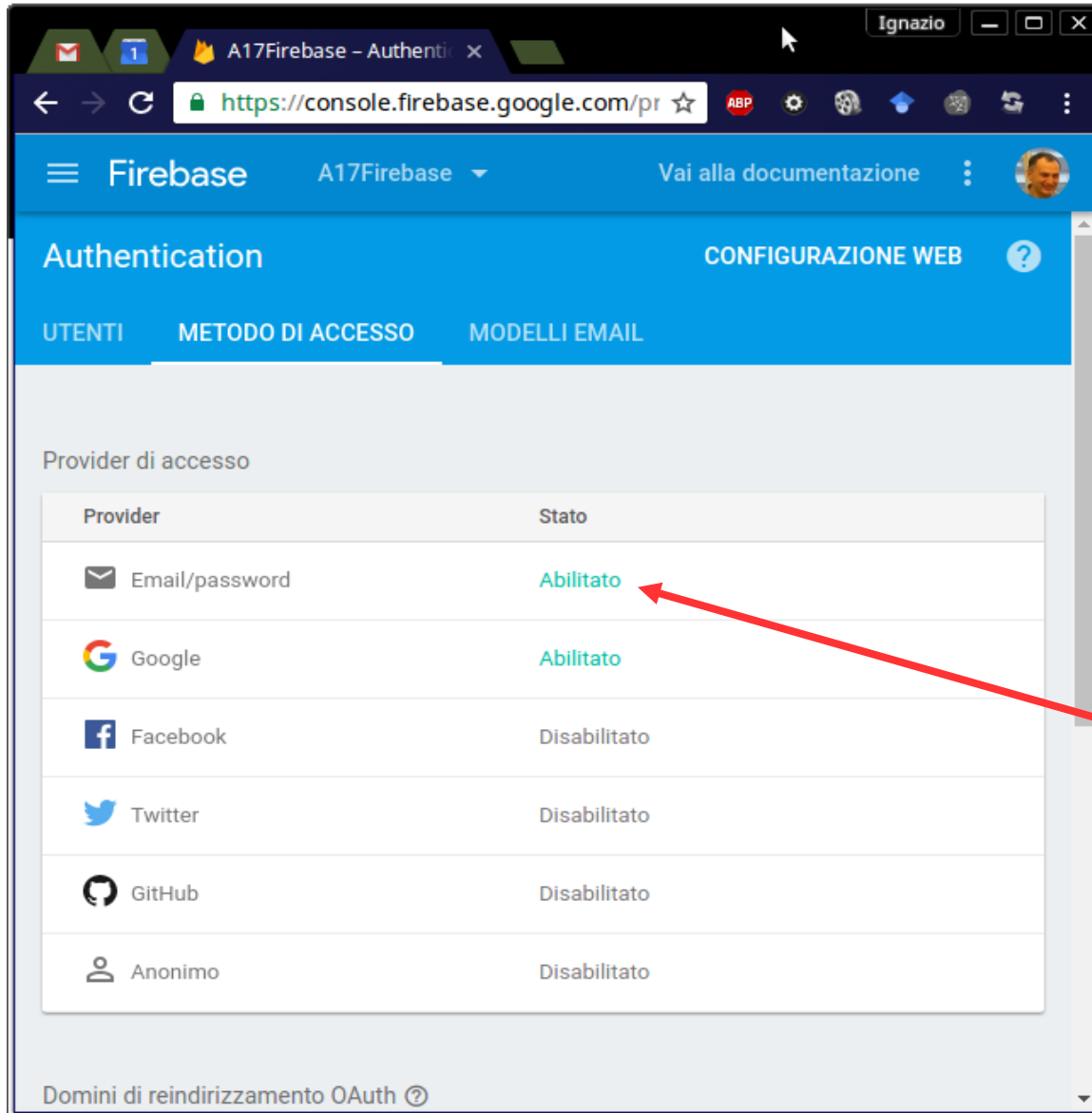
SignUpActivity



MainActivity

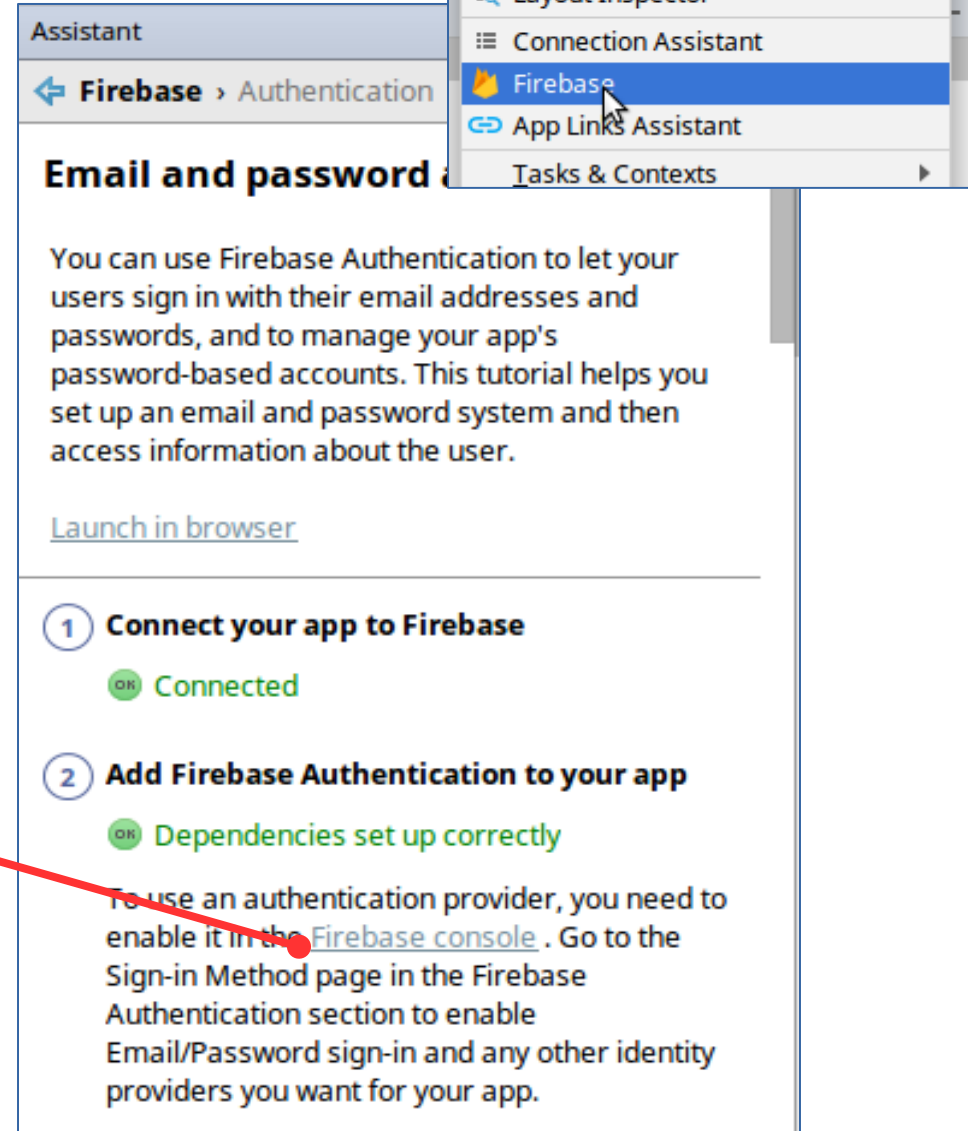


# Enable Authentication provider



The screenshot shows the Firebase Authentication console for project 'A17Firebase'. The 'Provider di accesso' section is active, displaying a table of authentication providers. The 'Email/password' provider is highlighted with a red arrow pointing to the word 'Abilitato'.

| Provider       | Stato        |
|----------------|--------------|
| Email/password | Abilitato    |
| Google         | Abilitato    |
| Facebook       | Disabilitato |
| Twitter        | Disabilitato |
| GitHub         | Disabilitato |
| Anonimo        | Disabilitato |



The screenshot shows the Android Studio interface. The 'Tools' menu is open, and 'Firebase' is selected. The 'Assistant' pane on the right displays the 'Email and password' tutorial.

**Tools** VCS Window Help

- AVD Manager
- SDK Manager
- Layout Inspector
- Connection Assistant
- Firebase**
- App Links Assistant
- Tasks & Contexts

**Assistant**

← **Firebase** > Authentication

### Email and password

You can use Firebase Authentication to let your users sign in with their email addresses and passwords, and to manage your app's password-based accounts. This tutorial helps you set up an email and password system and then access information about the user.

[Launch in browser](#)

- 1 Connect your app to Firebase**  
OK Connected
- 2 Add Firebase Authentication to your app**  
OK Dependencies set up correctly

To use an authentication provider, you need to enable it in the [Firebase console](#). Go to the Sign-in Method page in the Firebase Authentication section to enable Email/Password sign-in and any other identity providers you want for your app.

# FirebaseAuth and AuthStateListener

MainActivity.java x

MainActivity onCreate()

```
15
16 public class MainActivity extends AppCompatActivity {
17
18     private static final String TAG = "MainActivity";
19
20     private FirebaseAnalytics mFirebaseAnalytics;
21     private FirebaseAuth mAuth;
22     private FirebaseAuth.AuthStateListener mAuthListener;
23
24     @Override
25     protected void onCreate(Bundle savedInstanceState) {
26         super.onCreate(savedInstanceState);
27         setContentView(R.layout.activity_main);
28
29         // initialize the FirebaseAuth instance and
30         // the AuthStateListener method so you can track whenever the user
31         // signs in or out.
32         mAuth = FirebaseAuth.getInstance();
33         mAuthListener = new FirebaseAuth.AuthStateListener() {
34             @Override
35             public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
36                 FirebaseUser user = firebaseAuth.getCurrentUser();
37                 if (user != null) {
38                     // User is signed in
39                     Log.d(TAG, "onAuthStateChanged:signed_in:" + user.getId());
40                 } else {
41                     // User is signed out
42                     Log.d(TAG, "onAuthStateChanged:signed_out");
43                 }
44                 // ...
45             }
46         };
47
48         // Obtain the FirebaseAnalytics instance
```

Assistant

Firebase > Authentication

3 Listen for auth state

Declare the `FirebaseAuth` and `AuthStateListener` objects.

private FirebaseAuth mAuth;

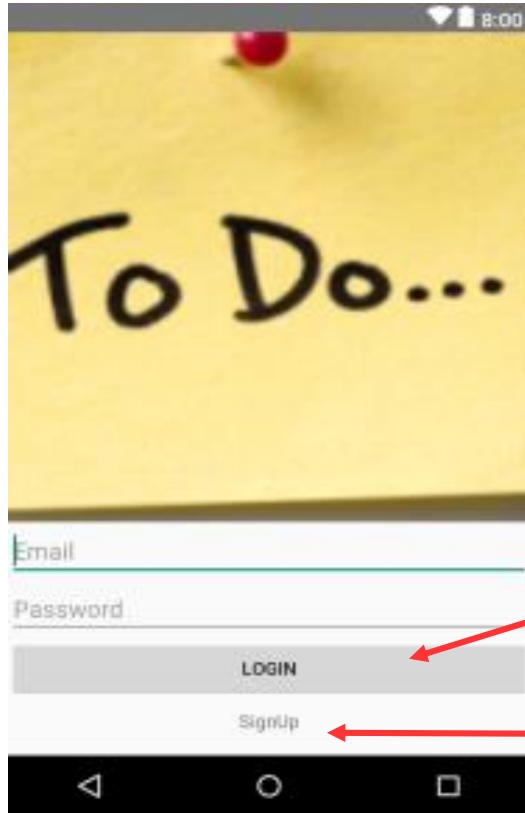
private FirebaseAuth.AuthStateListener mAuthListener;

In the `onCreate()` method, initialize the `FirebaseAuth` instance and the `AuthStateListener` method so you can track whenever the user signs in or out.

mAuth = FirebaseAuth.getInstance();

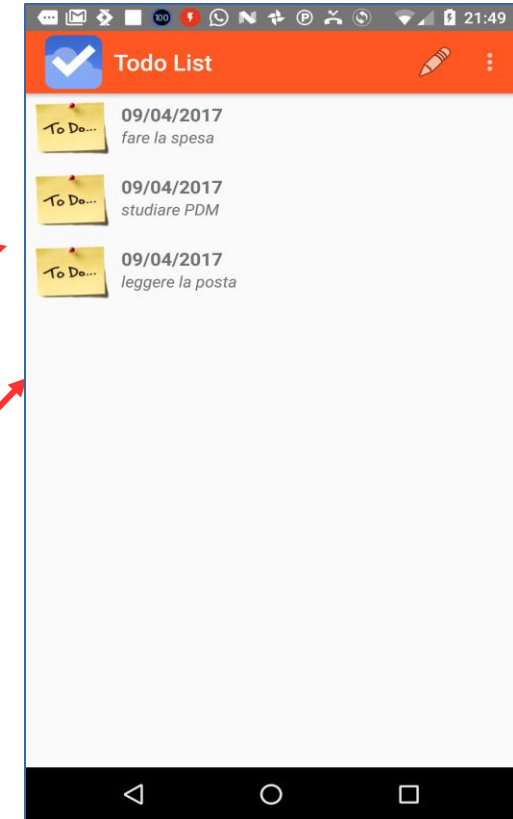
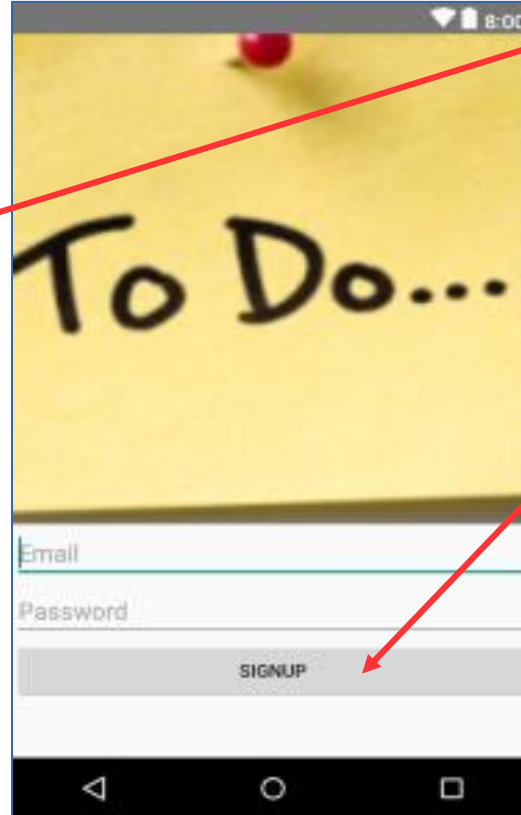
mAuthListener = new FirebaseAuth.AuthStateListener() {  
 @Override  
 public void onAuthStateChanged(@NonNull FirebaseAuth f  
 FirebaseUser user = firebaseAuth.getCurrentUser();  
 if (user != null) {  
 // User is signed in  
 Log.d(TAG, "onAuthStateChanged:signed\_in:" + u  
 } else {  
 // User is signed out  
 Log.d(TAG, "onAuthStateChanged:signed\_out");  
 }  
 // ...  
 }  
};

# Design the App: Java version



LoginActivity

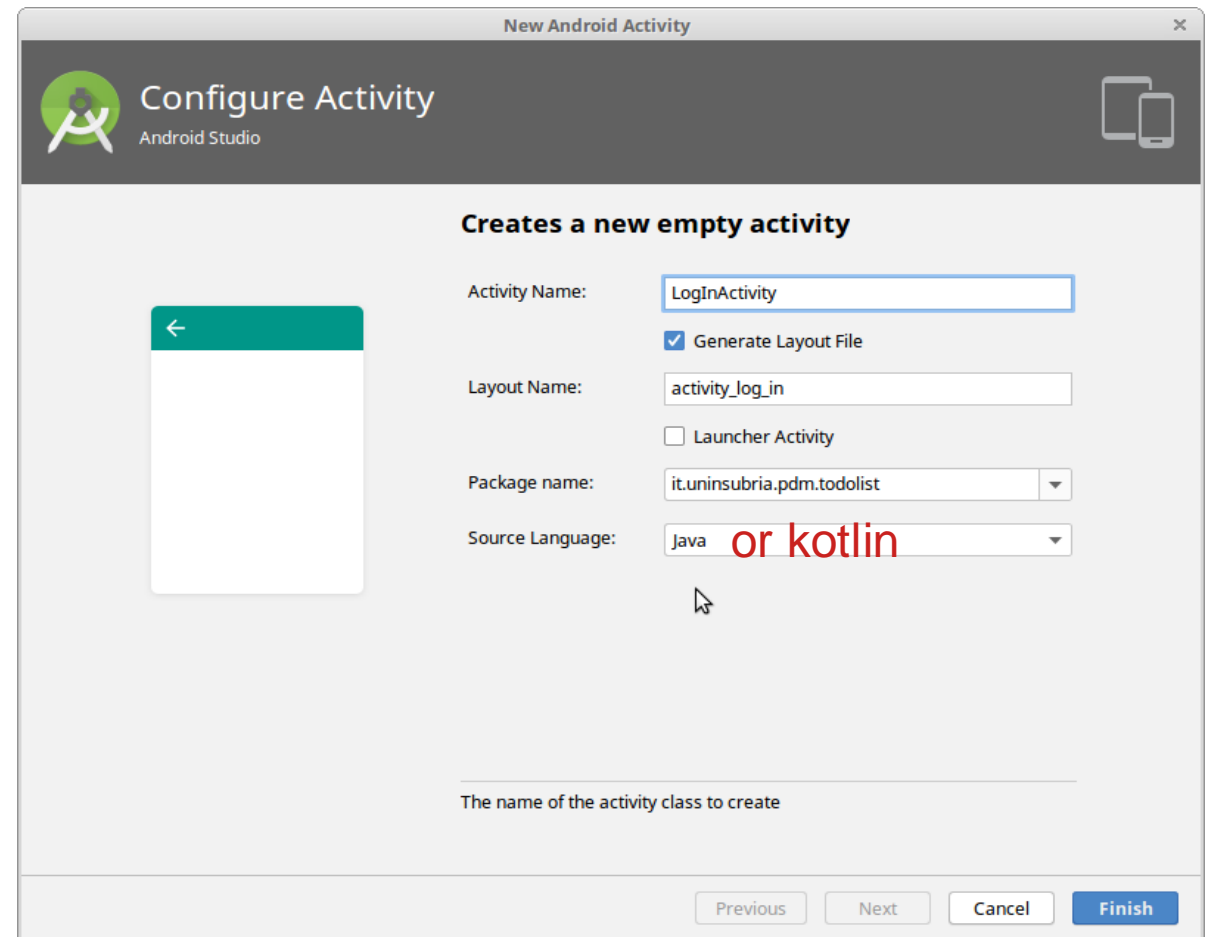
SignUpActivity



MainActivity

# Add two new Activities

- Create a new empty activity by selecting the **File -> New -> Activity -> Empty Activity** menu item and name it **LogInActivity**.
- Create another Empty Activity and name it **SignUpActivity**.



## Authentication using Kotlin



**ANDROID**  
developer lab



# LoginActivity Layout

```
<?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"
    android:background="@color/colorPrimary"
    android:orientation="vertical"
    android:padding="16dp"
    tools:context=".LoginActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="64dp"
        android:gravity="center"
        android:text="Welcome To"
        android:textColor="#FFFFFF"
        android:textSize="36sp" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="64dp"
        android:fontFamily="@font/dancing_script_font"
        android:gravity="center"
        android:text="ToDo App"
        android:textColor="#FFFFFF"
        android:textSize="44sp"
        android:textStyle="bold"
        app:fontFamily="@font/dancing_script_font" />

    <EditText
        android:id="@+id/emailEditText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="e-Mail"
        android:inputType="textEmailAddress" />

    ...

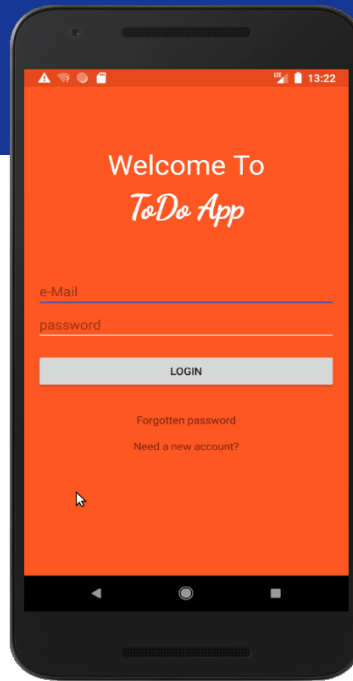
</LinearLayout>
```

```
<EditText
    android:id="@+id/passwordEditText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:ems="10"
    android:hint="password"
    android:inputType="textPassword" />

<Button
    android:id="@+id/loginButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Login" />

<TextView
    android:id="@+id/forgottenPasswordTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:layout_marginTop="32dp"
    android:text="Forgotten password" />

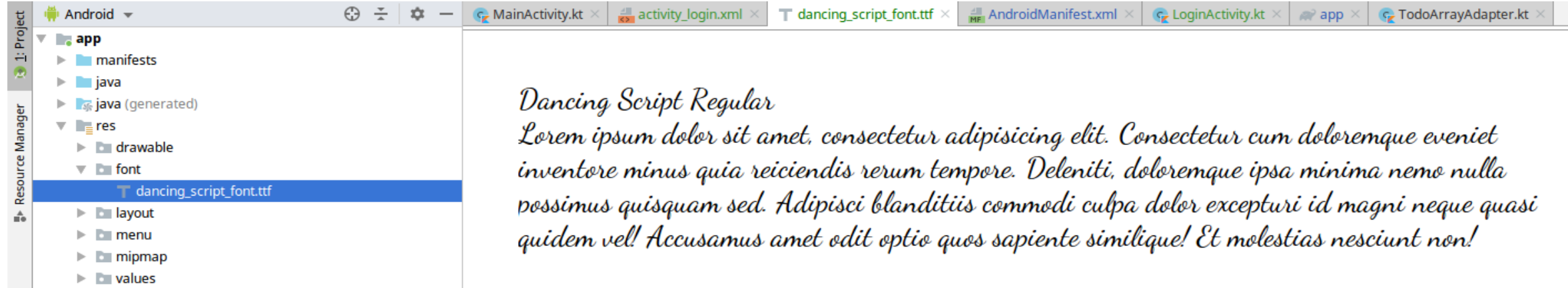
<TextView
    android:id="@+id/signUpTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:layout_marginTop="16dp"
    android:text="Need a new account?" />
```





# Font resources

- Create a folder **font** under **res** directory



- Download font from the WEB

<https://fonts.google.com/specimen/Dancing+Script>

- Now you can change font in layout using

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="ToDo App"
    android:textSize="36sp"
    android:fontFamily="@font/dancing_script_font"
    app:fontFamily="@font/dancing_script_font"
    android:gravity="center"
    android:layout_marginBottom="64dp"
/>
```

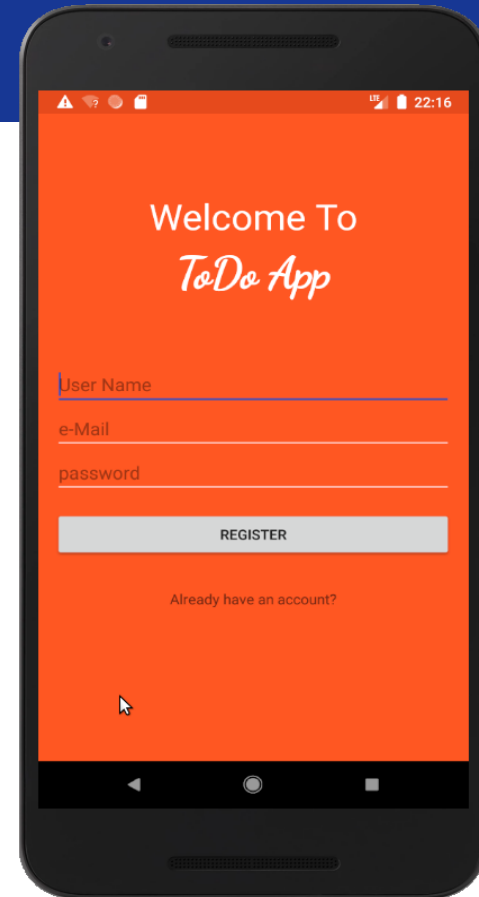
# SignUpActivity Layout

```
<?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"
    android:background="@color/colorPrimary"
    android:orientation="vertical"
    android:padding="16dp"
    tools:context=".LoginActivity">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="64dp"
        android:gravity="center"
        android:text="Welcome To"
        android:textColor="#FFFFFF"
        android:textSize="36sp" />
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="64dp"
        android:fontFamily="@font/dancing_script_font"
        android:gravity="center"
        android:text="ToDo App"
        android:textColor="#FFFFFF"
        android:textSize="44sp"
        android:textStyle="bold"
        app:fontFamily="@font/dancing_script_font" />
    <EditText
        android:id="@+id/nameEditText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:hint="User Name" />
```

...

```
<EditText
    android:id="@+id/emailEditText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:hint="e-Mail"
    android:inputType="textEmailAddress" />
<EditText
    android:id="@+id/passwordEditText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:ems="10"
    android:hint="password"
    android:inputType="textPassword" />
<Button
    android:id="@+id/registerButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Register" />
<TextView
    android:id="@+id/loginTextView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:layout_marginTop="32dp"
    android:text="Already have an account?" />

</LinearLayout>
```





# LoginActivity - Kotlin

```
class LoginActivity : AppCompatActivity() {  
  
    private lateinit var auth: FirebaseAuth  
    private var TAG = "LoginActivity"  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_login)  
  
        auth = Firebase.auth  
        loginButton.setOnClickListener {  
            onLoginClick()  
        }  
        signUpTextView.setOnClickListener {  
            onSignUpClick()  
        }  
    }  
    private fun onSignUpClick() {  
        val intent = Intent(this, SignUpActivity::class.java)  
        startActivity(intent)  
        finish()  
    }  
    private fun onLoginClick() {  
        val email = emailEditText.text.toString().trim()  
        val password = passwordEditText.text.toString().trim()  
        if (email.isEmpty()) {  
            emailEditText.error = "Enter email"  
            return  
        }  
        if (password.isEmpty()) {  
            passwordEditText.error = "Enter password"  
            return  
        }  
        loginUser(email, password)  
    }  
}
```

```
private fun loginUser(email: String, password: String) {  
    auth.signInWithEmailAndPassword(email, password)  
        .addOnCompleteListener(this) { task ->  
        if (task.isSuccessful) {  
            // Sign in success, update UI with the signed-in user's information  
            Log.d(TAG, "signInWithEmail:success")  
            val intent = Intent(this, MainActivity::class.java)  
            startActivity(intent)  
            finish()  
        } else {  
            // If sign in fails, display a message to the user.  
            Log.w(TAG, "signInWithEmail:failure", task.exception)  
            val builder = AlertDialog.Builder(this)  
            with(builder)  
            {  
                setTitle("Authentication failed")  
                setMessage(task.exception?.message)  
                setPositiveButton("OK", null)  
                show()  
            }  
        }  
    }  
}
```

# SignUpActivity - Kotlin

```
class SignUpActivity : AppCompatActivity() {
```

```
    private lateinit var auth: FirebaseAuth
    private var TAG = "SignUpActivity"
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_register)
```

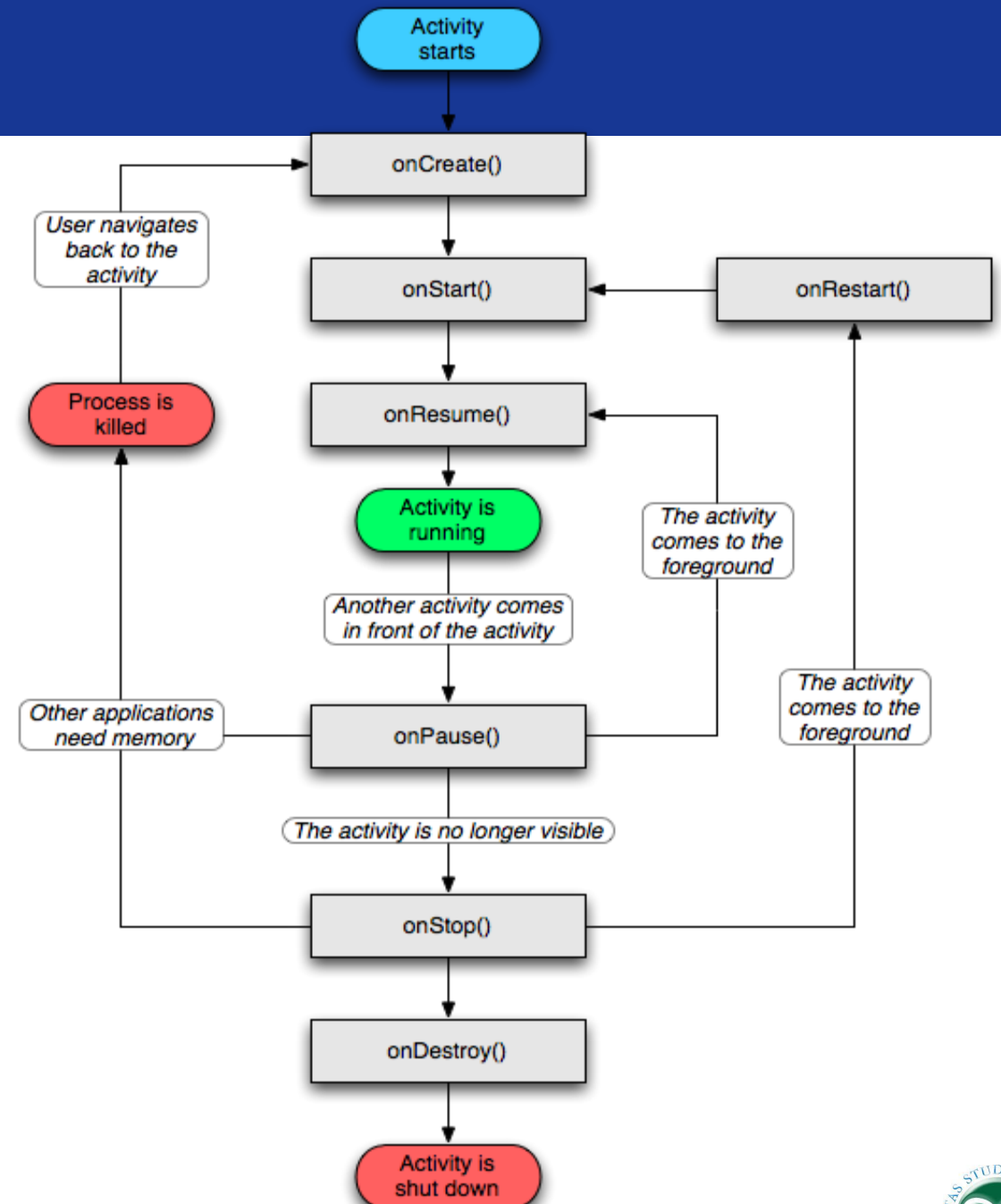
```
        auth = FirebaseAuth.getInstance()
        registerButton.setOnClickListener {
            onSignUpClick()
        }
```

```
    private fun onSignUpClick() {
        val email = emailEditText.text.toString().trim()
        val password = passwordEditText.text.toString().trim()
        val userName = nameEditText.text.toString().trim()
        if (userName.isEmpty()) {
            nameEditText.error = "Enter userName"
            return
        }
        if (email.isEmpty()) {
            emailEditText.error = "Enter email"
            return
        }
        if (password.isEmpty()) {
            passwordEditText.error = "Enter password"
            return
        }
        createUser(userName, email, password)
    }
```

```
    private fun createUser(userName: String, email: String, password: String) {
        auth.createUserWithEmailAndPassword(email, password)
            .addOnCompleteListener(this) { task ->
                if (task.isSuccessful) {
                    // Sign in success, update UI with the signed-in user's information
                    Log.d(TAG, "createUserWithEmail:success")
                    val currentUser = auth.currentUser
                    val uid = currentUser!!.uid
                    val userMap = HashMap<String, String>()
                    userMap["name"] = userName
                    val database = FirebaseDatabase.getInstance().getReference("Users").child(uid)
                    database.setValue(userMap).addOnCompleteListener { task ->
                        if (task.isSuccessful) {
                            val intent = Intent(applicationContext, MainActivity::class.java)
                            startActivity(intent)
                            finish()
                        }
                    }
                } else {
                    // If sign in fails, display a message to the user.
                    Log.w(TAG, "createUserWithEmail:failure", task.exception)
                    Toast.makeText(baseContext, "Authentication failed.",
                        Toast.LENGTH_SHORT).show()
                }
            }
    }
```

# MainActivity - Kotlin

```
class MainActivity : AppCompatActivity() {  
    ...  
    private lateinit var auth: FirebaseAuth  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
        ...  
        // Initialize Firebase Auth  
        auth = FirebaseAuth  
    }  
  
    override fun onStart() {  
        super.onStart()  
        // Check if user is signed in (non-null)  
        val currentUser = auth.currentUser  
        if(currentUser == null) {  
            val intent = Intent(this, LoginActivity::class.java)  
            startActivity(intent)  
            finish()  
        }  
    }  
  
    override fun onOptionsItemSelected(item: MenuItem): Boolean {  
        if (id == R.id.action_settings) {  
            ...  
        } else if (id == R.id.action_new_item) {  
            ...  
        } else if (id == R.id.action_logout) {  
            auth.signOut()  
            finish()  
        }  
        return super.onOptionsItemSelected(item)  
    }  
}
```



# MainActivity - Kotlin

```
class MainActivity : AppCompatActivity() {  
    ...  
    private lateinit var auth: FirebaseAuth  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
        ...  
        // Initialize Firebase Auth  
        auth = FirebaseAuth.getInstance()  
    }  
  
    override fun onStart() {  
        super.onStart()  
        // Check if user is signed in (non-null)  
        val currentUser = auth.currentUser  
        if(currentUser == null) {  
            val intent = Intent(this, LoginActivity::class.java)  
            startActivity(intent)  
            finish()  
        }  
    }  
  
    override fun onOptionsItemSelected(item: MenuItem): Boolean {  
        if (id == R.id.action_settings) {  
            ...  
        } else if (id == R.id.action_new_item) {  
            ...  
        } else if (id == R.id.action_logout) {  
            auth.signOut()  
            finish()  
        }  
        return super.onOptionsItemSelected(item)  
    }  
}
```

## Authentication State Persistence

For an Android application, the default behavior is to persist a user's session even after the user closes the App.

## main\_activity\_menu.xml

```
<item  
    android:id="@+id/action_logout"  
    android:title="@string/action_logout"  
    android:icon="@android:drawable/ic_lock_power_off"  
    app:showAsAction="ifRoom" />
```



## Firestore: Save and read data

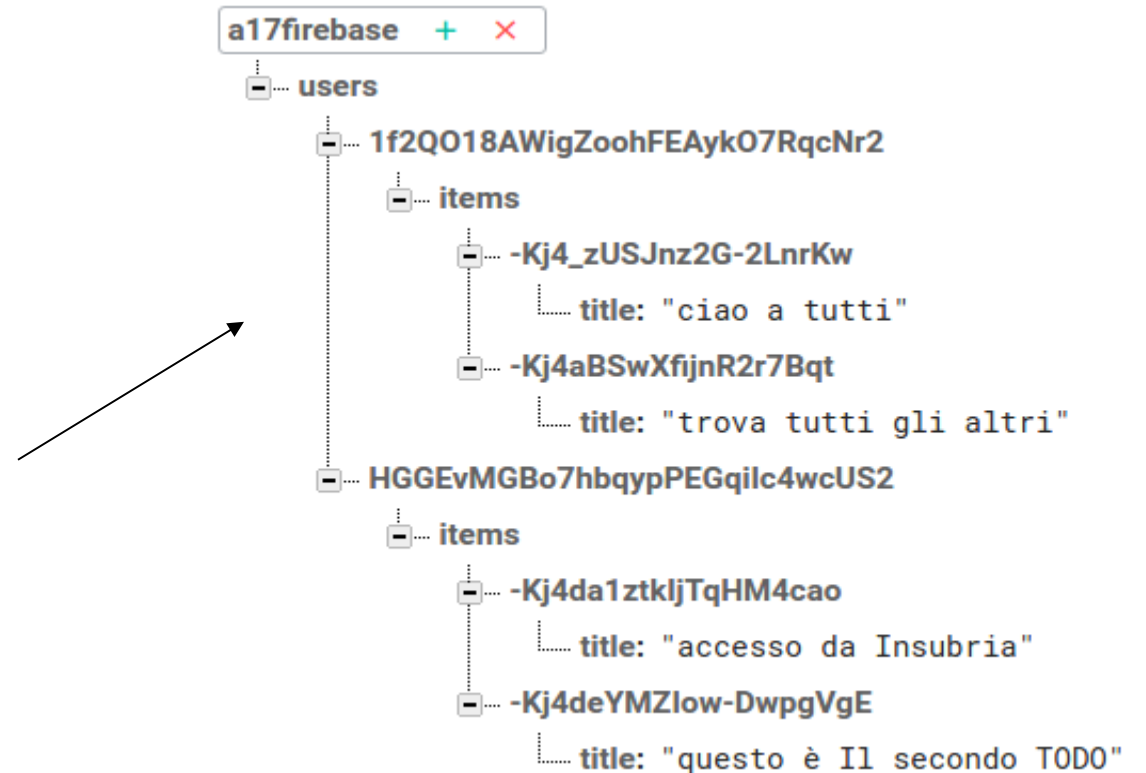


ANDROID  
developer lab



# MainActivity write database

```
private void addNewItem(String todo) {  
    if (todo.length() == 0) {  
        Toast.makeText(getApplicationContext(),  
            "Empty Todo string",  
            Toast.LENGTH_LONG).show();  
        return;  
    }  
    TodoItem newTodo = new TodoItem(todo);  
    // add to the DB and set the item idx  
    long idx = dbHelper.insertItem(newTodo);  
    newTodo.setId((int)idx);  
  
    todoItems.add(0, newTodo);  
  
    mDatabase.child("users").child(mUserId)  
        .child("items").push().setValue(newTodo);  
  
    mAdapter.notifyDataSetChanged();  
}
```



# MainActivity read database

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    // ...
    // populate the list.
    mDatabase.child("users").child(mUserId).child("items").addChildEventListener(new ChildEventListener() {
        @Override
        public void onChildAdded(DataSnapshot dataSnapshot, String s) {
            TodoItem item = new TodoItem((String) dataSnapshot.child("title").getValue());
            todoItems.add(item);
            mAdapter.notifyDataSetChanged();
        }
        @Override
        public void onChildChanged(DataSnapshot dataSnapshot, String s) { }

        @Override
        public void onChildRemoved(DataSnapshot dataSnapshot) {
            TodoItem item = new TodoItem((String) dataSnapshot.child("title").getValue());
            todoItems.remove(item);
            mAdapter.notifyDataSetChanged();
        }
        @Override
        public void onChildMoved(DataSnapshot dataSnapshot, String s) { }

        @Override
        public void onCancelled(DatabaseError databaseError) { }
    });
}
```