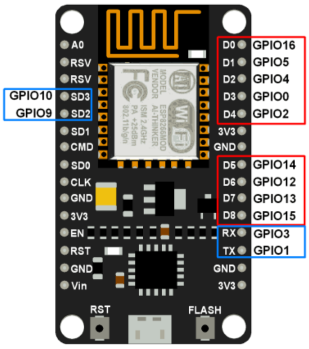
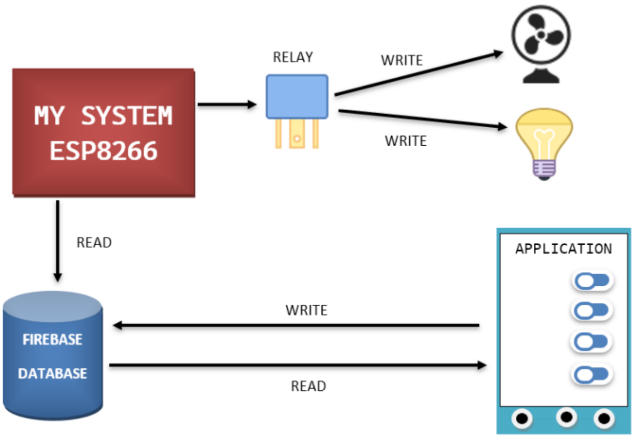
Home Automation Explained

A Complete Guide to Home Automation using ESP8266 Module with Basic Android Application





SHUBHAM TIWARI

(Author)

FIREBASE & (NODEMCU)ESP8266 MODULE COMMUNICATION

In this topic, we will learn about how to make commutation between your **(NodeMCU) ESP8266-12E** Module and **Firebase Realtime Database.**

We'll also learn about updating, fetching, removing values from Realtime database.

Realtime Database?

As per my point of view, A database which is capable of updating values of its corresponding fields very fast on a user response without need to refresh the page is known as Realtime database.

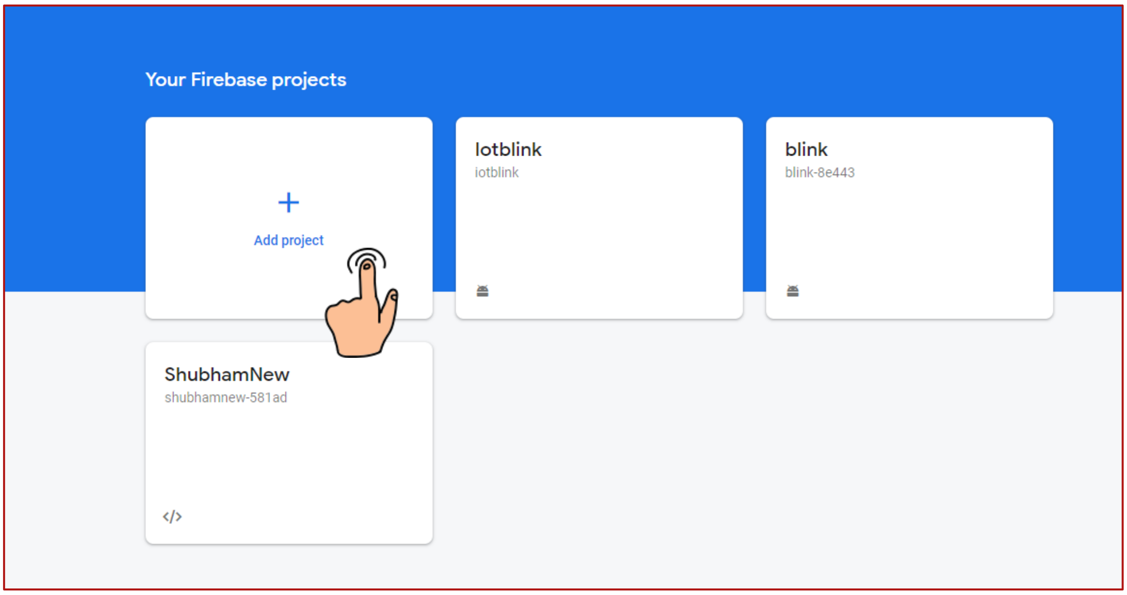
Video Link : See Demo

Why Firebase?

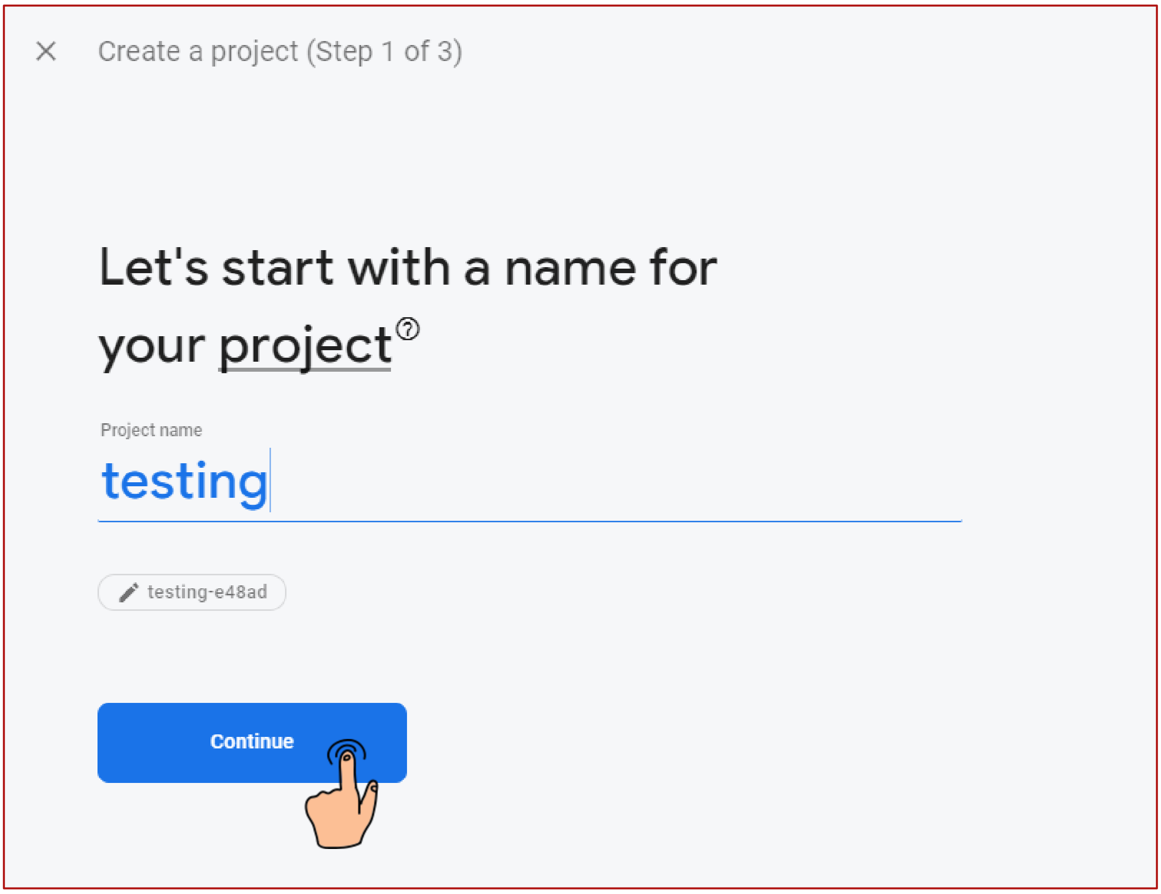
Firebase provide facility of Realtime & cloud databases and it has very user-friendly interface and easy to use after some learning that's why we use firebase which is a trusted service offered by Google.

How to Create Firebase Realtime Database ?

1. Go to https://console.firebase.google.com/
2. Click on **Add project**



3. Enter **Project Name** and click on **continue** & again click on **continue**.



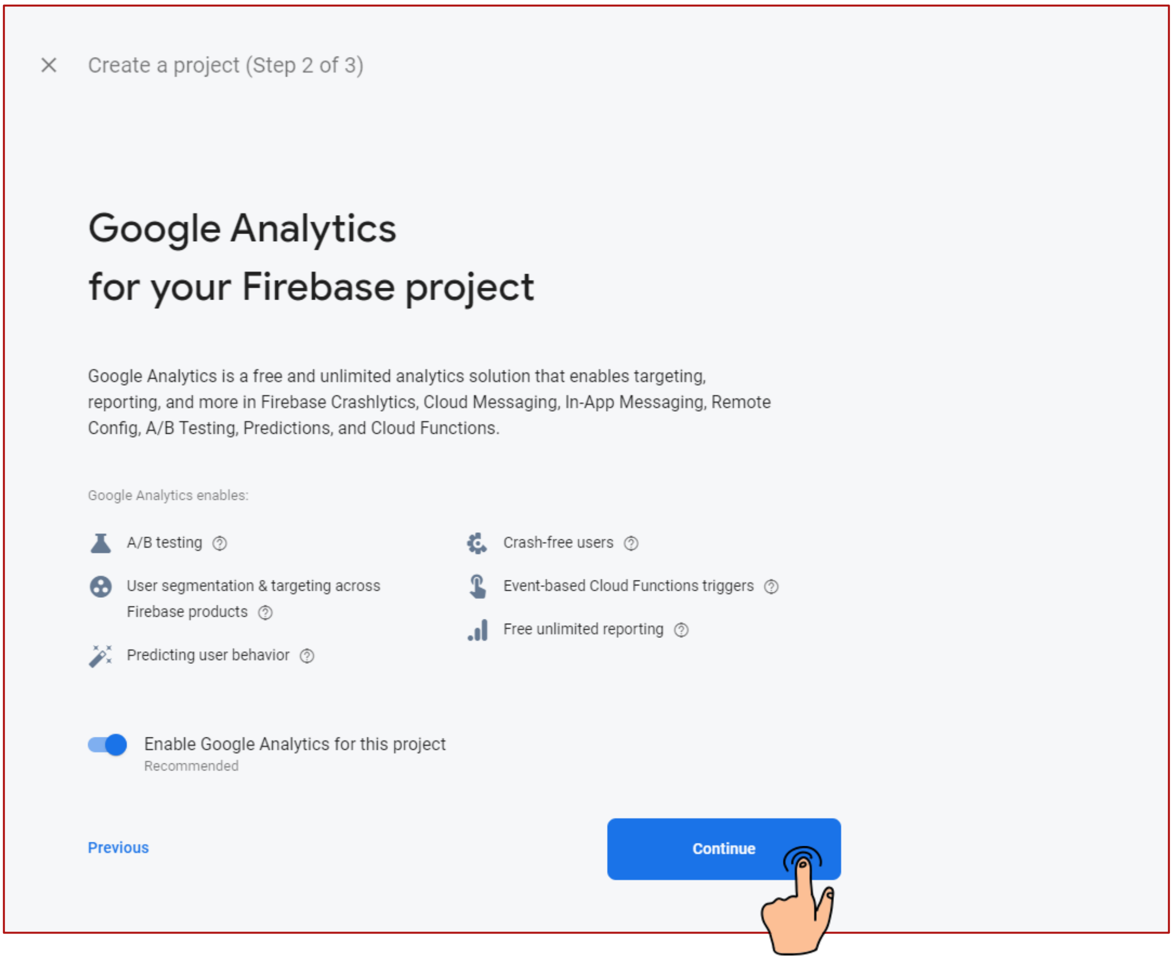
Create a project (Step 1 of 3)

Let's start with a name for your project

Project name

testing

testing-e48ad



Create a project (Step 2 of 3)

Google Analytics for your Firebase project

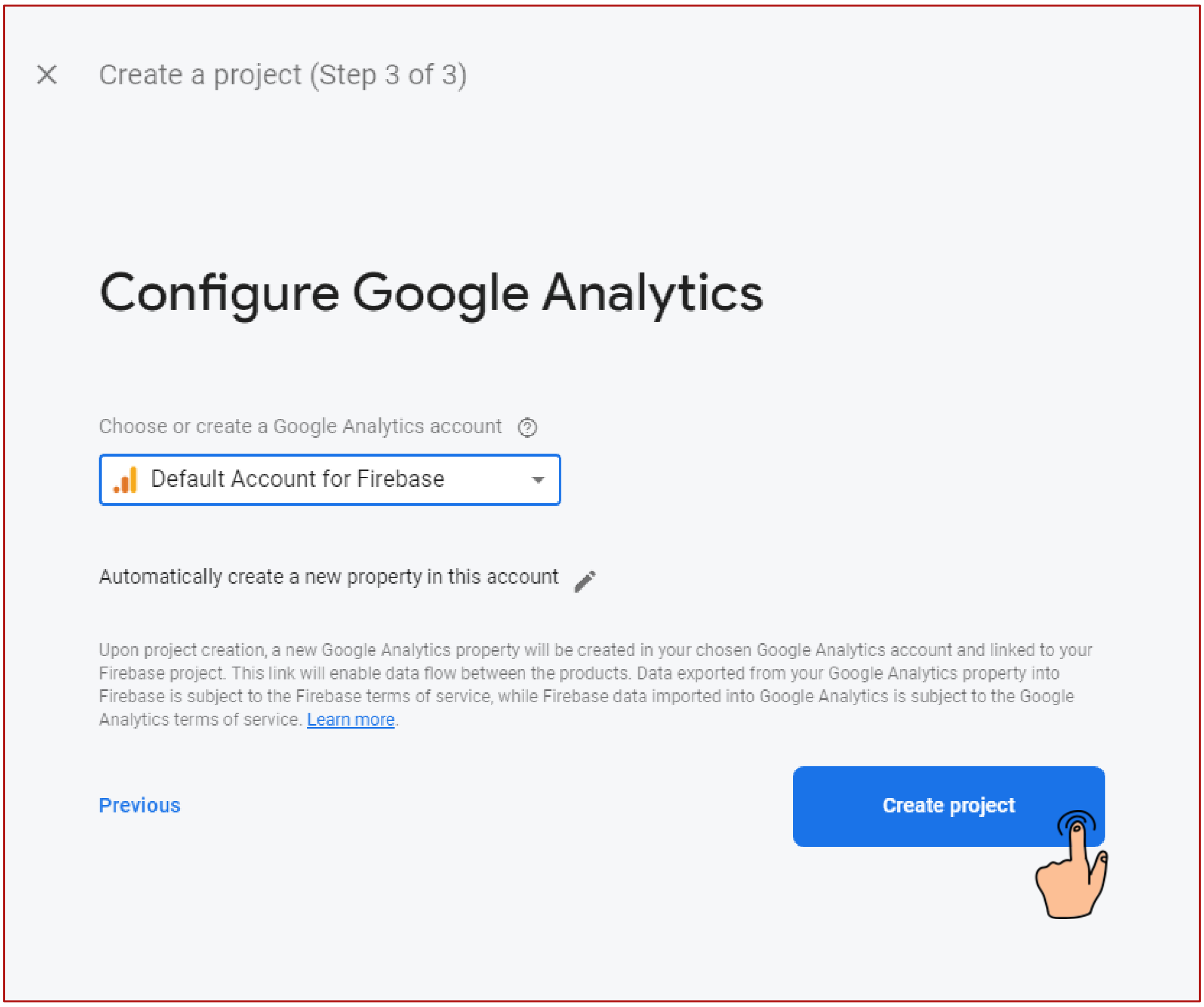
Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, Predictions, and Cloud Functions.

Google Analytics enables:

- A/B testing
- User segmentation & targeting across Firebase products
- Predicting user behavior
- Crash-free users
- Event-based Cloud Functions triggers
- Free unlimited reporting

Enable Google Analytics for this project
Recommended

4. Select **Default Account for firebase** and click on **create project**.



Create a project (Step 3 of 3)

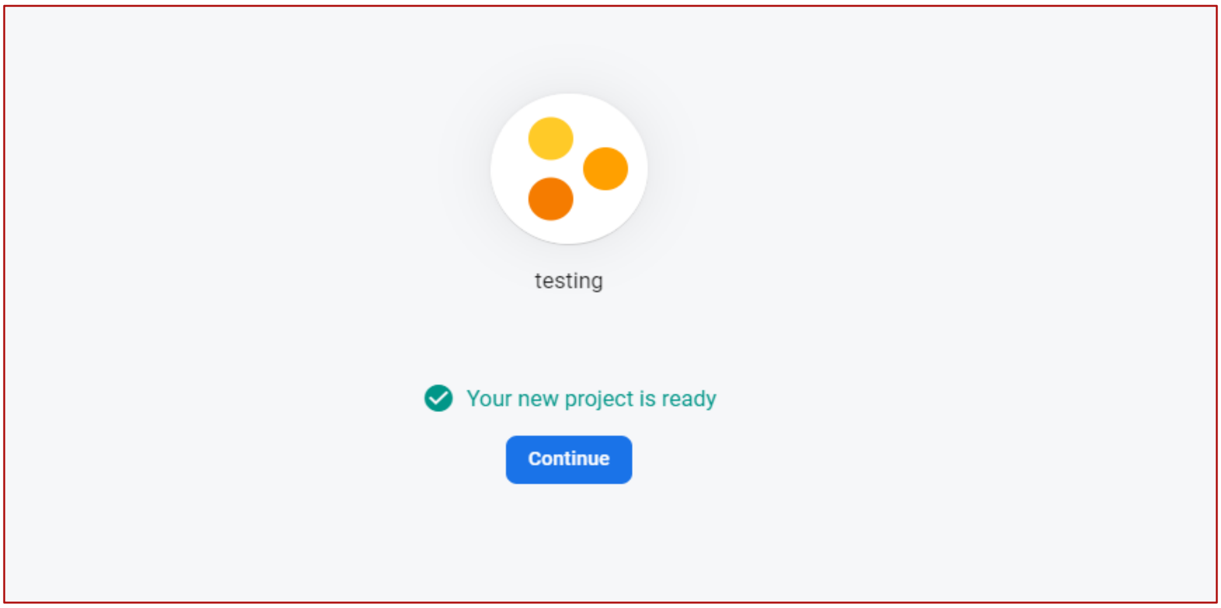
Configure Google Analytics

Choose or create a Google Analytics account

Default Account for Firebase

Automatically create a new property in this account

Upon project creation, a new Google Analytics property will be created in your chosen Google Analytics account and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. Learn more.

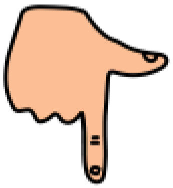


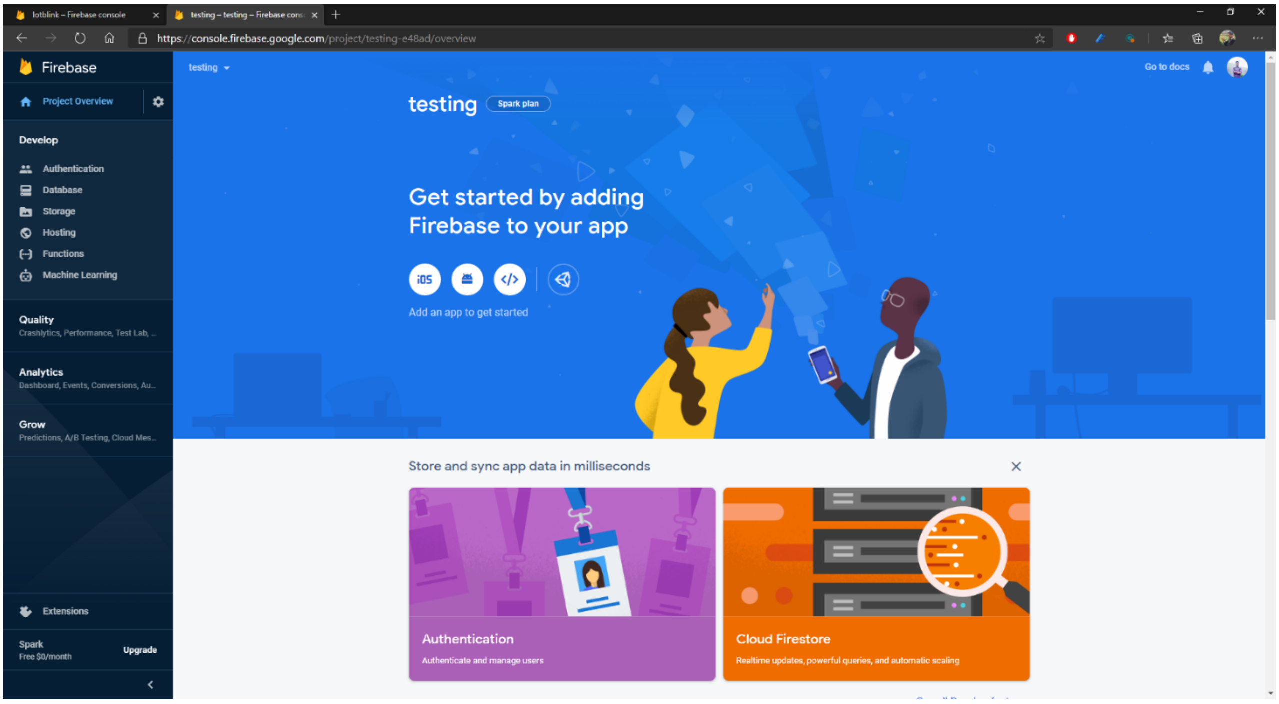
testing

Your new project is ready

Wait until your project is created!

Now we'll get Screen Like this!





lotblink – Firebase console
testing – testing – Firebase cons
https://console.firebase.google.com/project/testing-e48ad/overview
Firebase
Project Overview
Develop
Authentication
Database
Storage
Hosting
Functions
Machine Learning
Quality
Crashlytics, Performance, Test Lab, ...
Analytics
Dashboard, Events, Conversions, Au...
Grow
Predictions, A/B Testing, Cloud Mes...
Extensions
Spark
Free $0/month
Upgrade
testing
Go to docs
iOS

testing Spark plan

Get started by adding Firebase to your app

Add an app to get started

Store and sync app data in milliseconds

Authentication

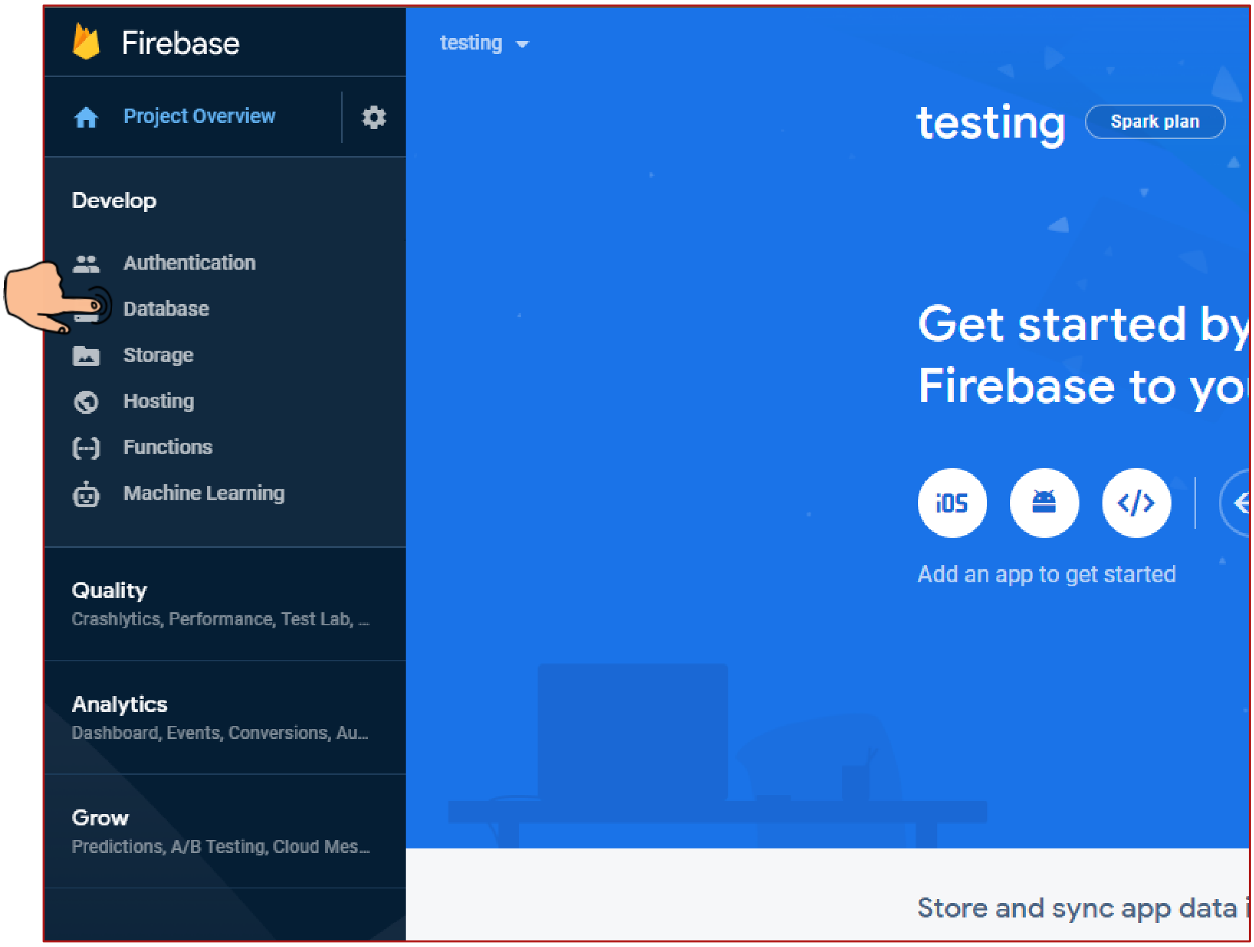
Authenticate and manage users

Cloud Firestore

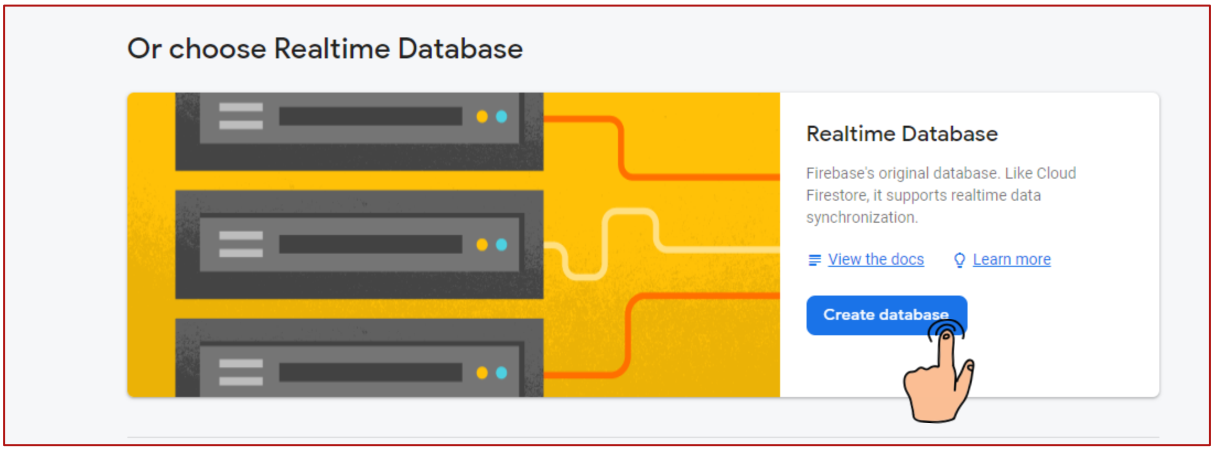
Realtime updates, powerful queries, and automatic scaling

Now we'll create Realtime Database Schema

1. Click on Database.



2. Scroll Down and find Realtime Database and click on create Database button

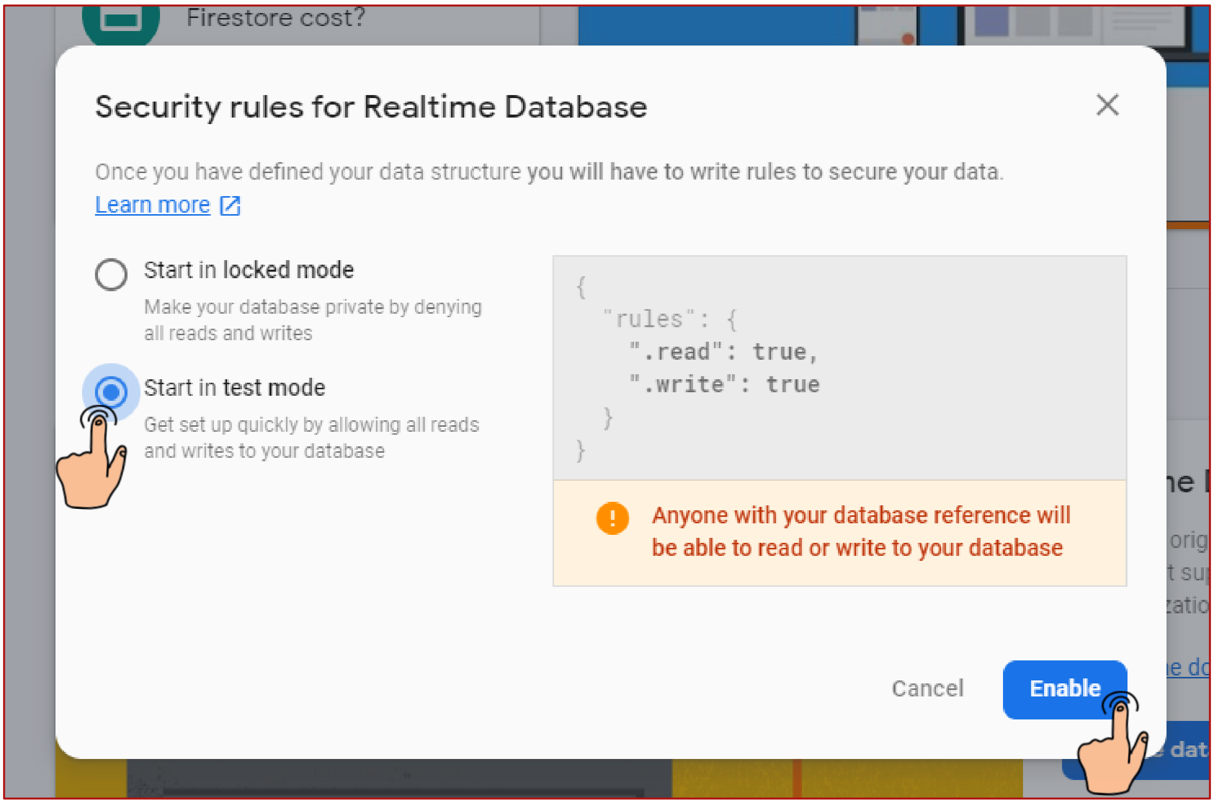


Or choose Realtime Database

Realtime Database

Firebase's original database. Like Cloud Firestore, it supports realtime data synchronization.

3. Choose **start in test mode** and click on **Enable**.



Security rules for Realtime Database

Once you have defined your data structure you will have to write rules to secure your data.
Learn more

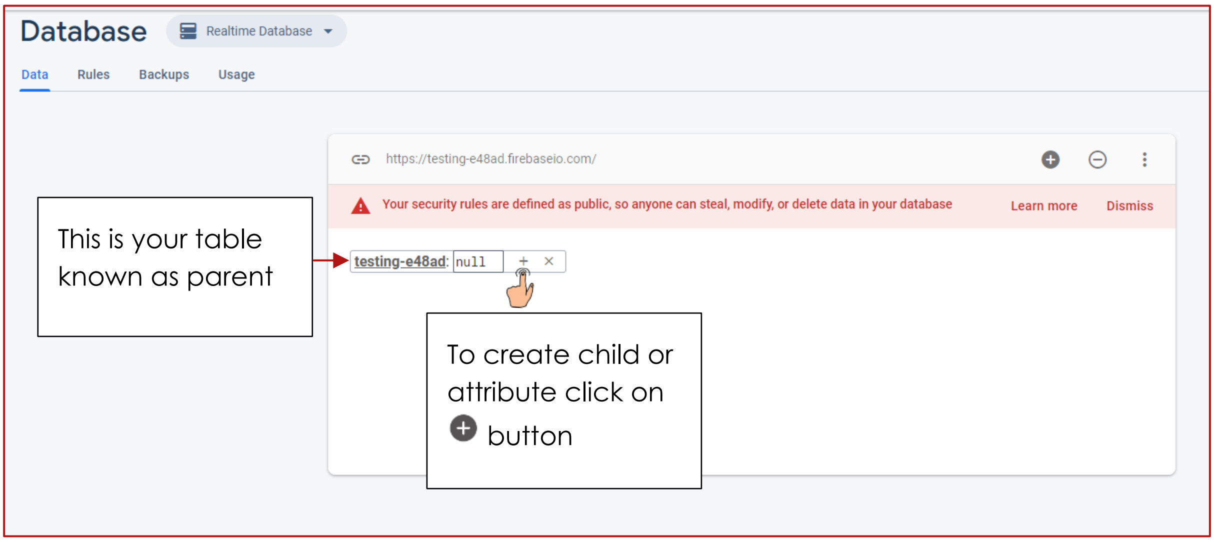
○ Start in locked mode
Make your database private by denying all reads and writes

● Start in test mode
Get set up quickly by allowing all reads and writes to your database

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

Anyone with your database reference will be able to read or write to your database

4. You'll have to create Schema for your database in the form of **tree like structure** or I can say in the form of parent-child schema.
For example, your **table name is said to be parent** comes at top of the schema and **attributes corresponding to your table is known as child** of the parent.
See this Picture Demonstration



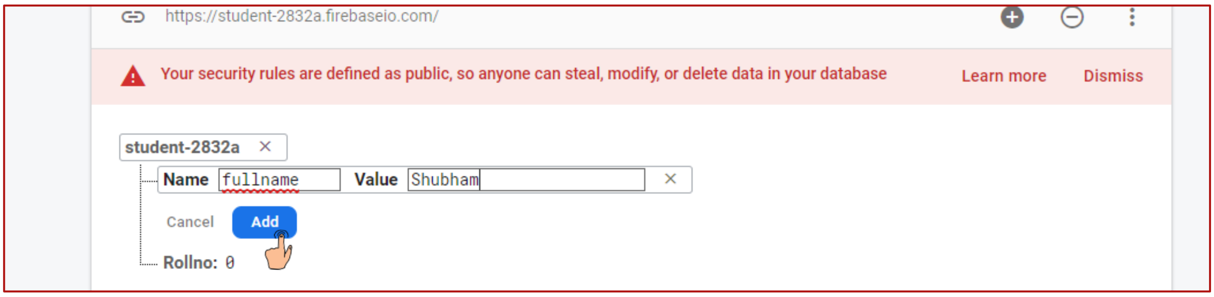
Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

testing-e48ad: null

This is your table known as parent

To create child or attribute click on ⊕ button

For example: Here is the example of student table/Database.



https://student-2832a.firebaseio.com/

Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

student-2832a

Name fullname Value Shubham

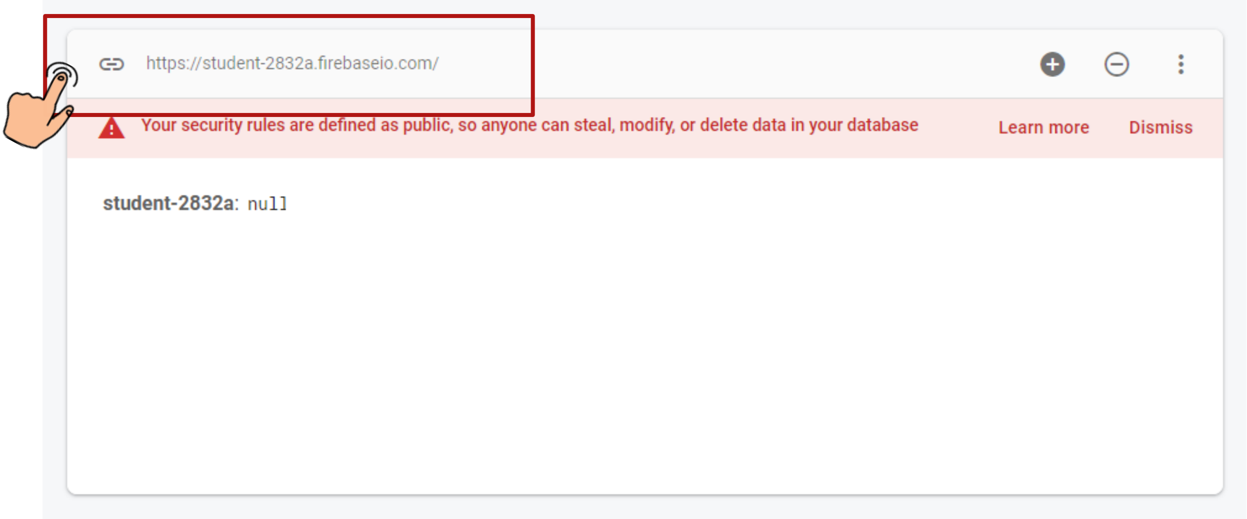
Rollno: 0

Things to Remember:

In order to perform linking you should know some basic Keywords, settings and procedure.

Here you need these three things:

- **Firebase Host**: Hostname for your database shown in the picture.

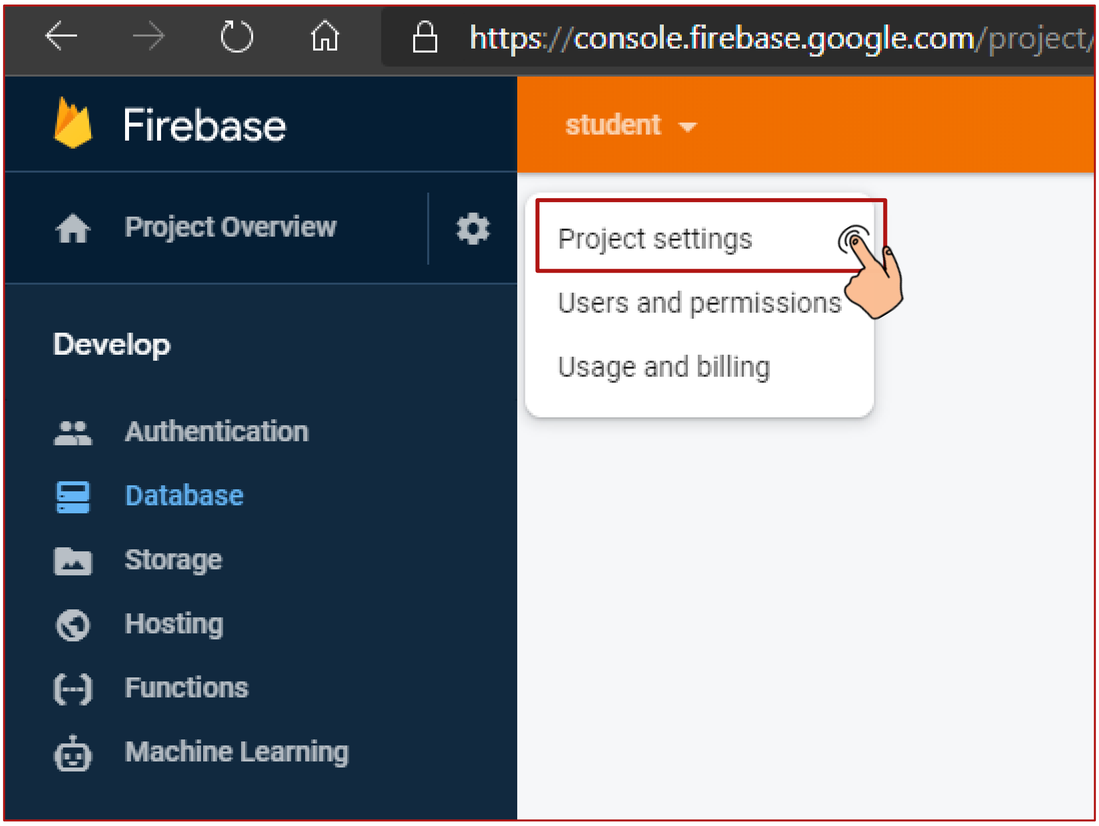


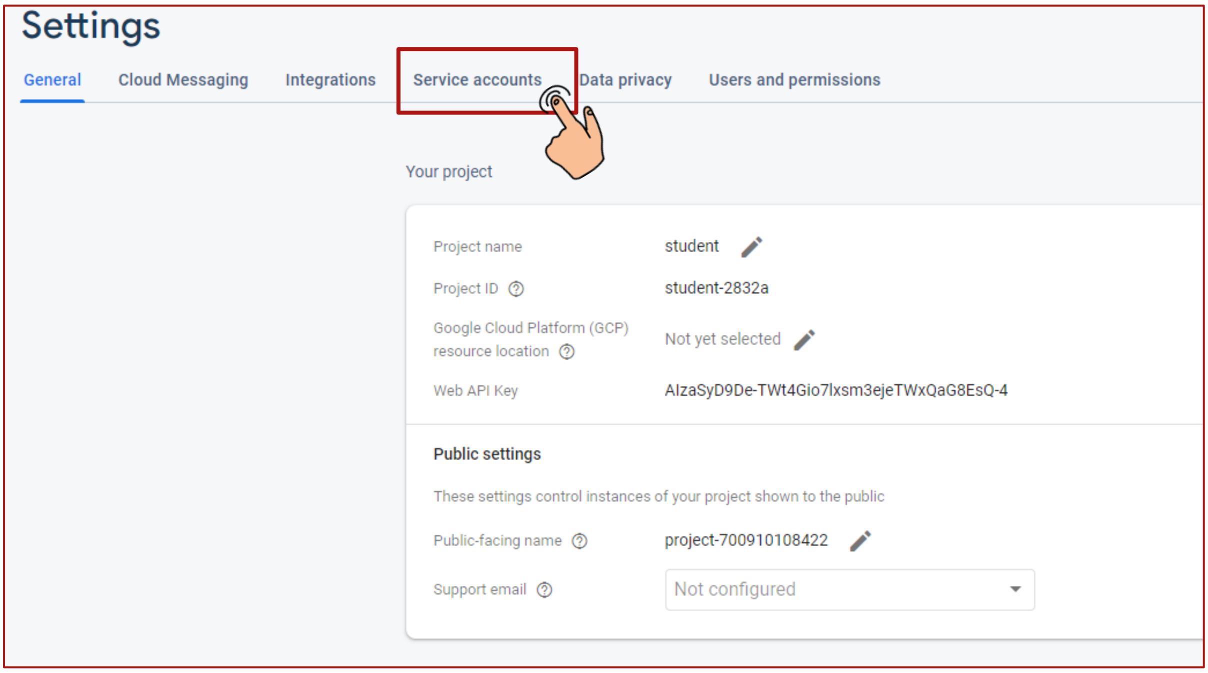
https://student-2832a.firebaseio.com/

Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

student-2832a: null

- **Firebase Auth** : You'll need some sort of token or secrete code for authorization purpose and this code is used by your software to communicate with your corresponding database. **Please refer to this image**
Click on Project Setting >> Service Account Tab >> Database Secret.





Your project

Project name student

Project ID student-2832a

Google Cloud Platform (GCP) resource location Not yet selected

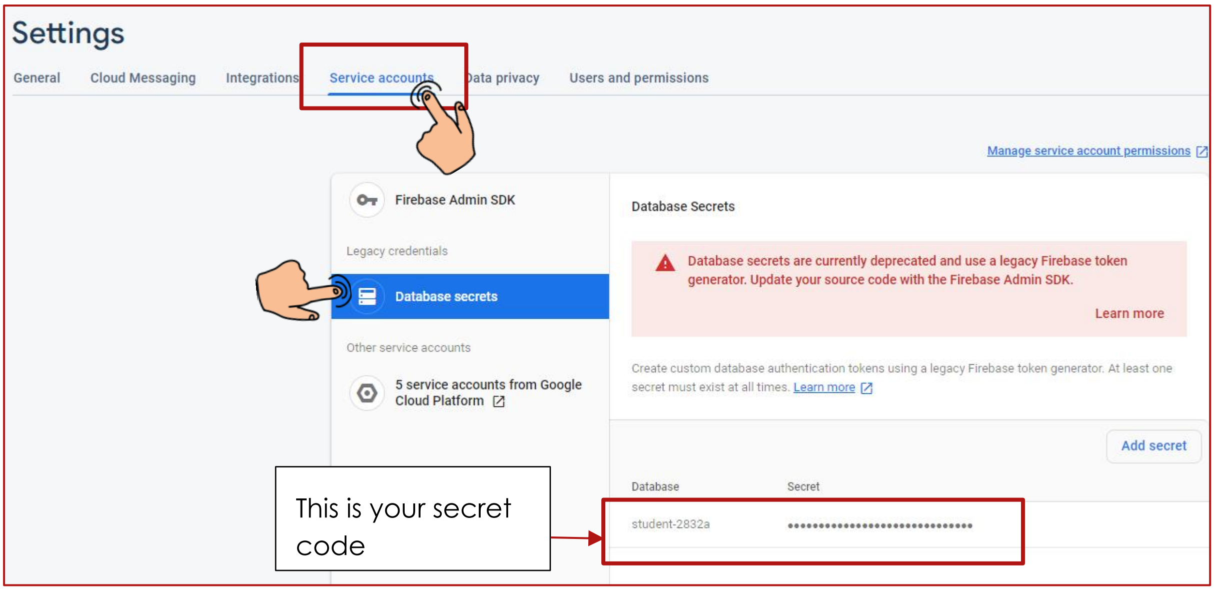
Web API Key AIzaSyD9De-TWt4Gio7lxsm3ejeTWxQaG8EsQ-4

Public settings

These settings control instances of your project shown to the public

Public-facing name project-700910108422

Support email Not configured



Settings

General Cloud Messaging Integrations Service accounts Data privacy Users and permissions

Manage service account permissions

Firebase Admin SDK

Legacy credentials

Database secrets

Other service accounts

5 service accounts from Google Cloud Platform

Database Secrets

Database secrets are currently deprecated and use a legacy Firebase token generator. Update your source code with the Firebase Admin SDK.

Learn more

Create custom database authentication tokens using a legacy Firebase token generator. At least one secret must exist at all times. Learn more

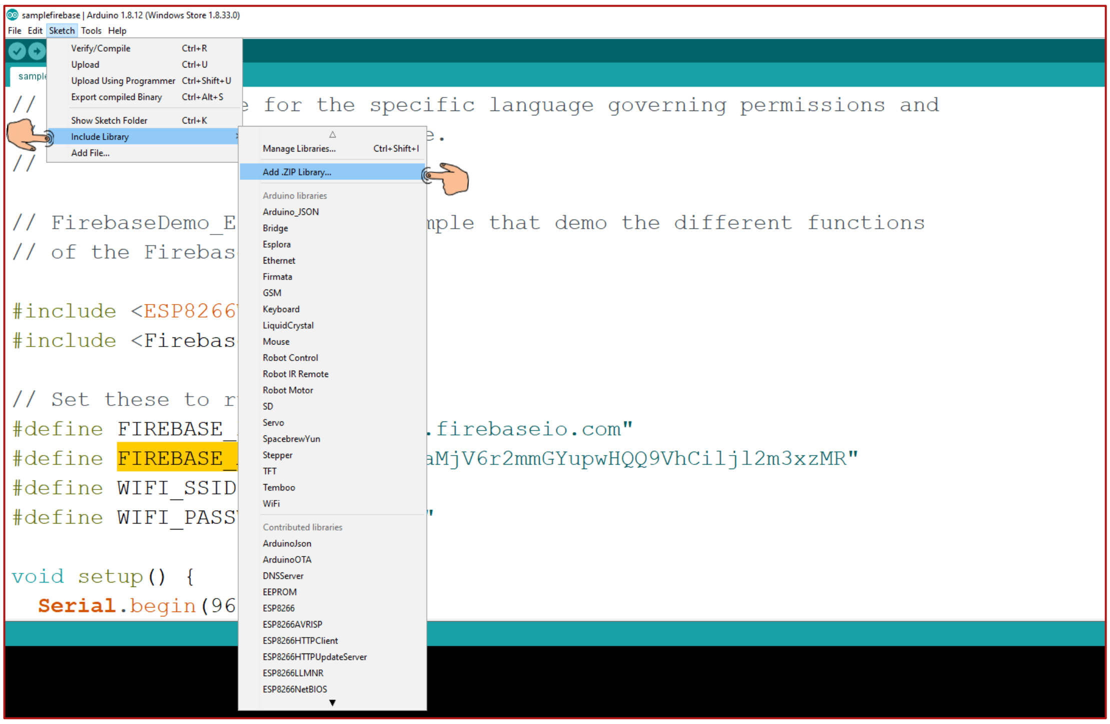
Add secret

Database	Secret
student-2832a	••••••••••••••••••••••••••••••

This is your secret code

Requirements

- **ArduinoFirebase Library** Offline Link Online Link

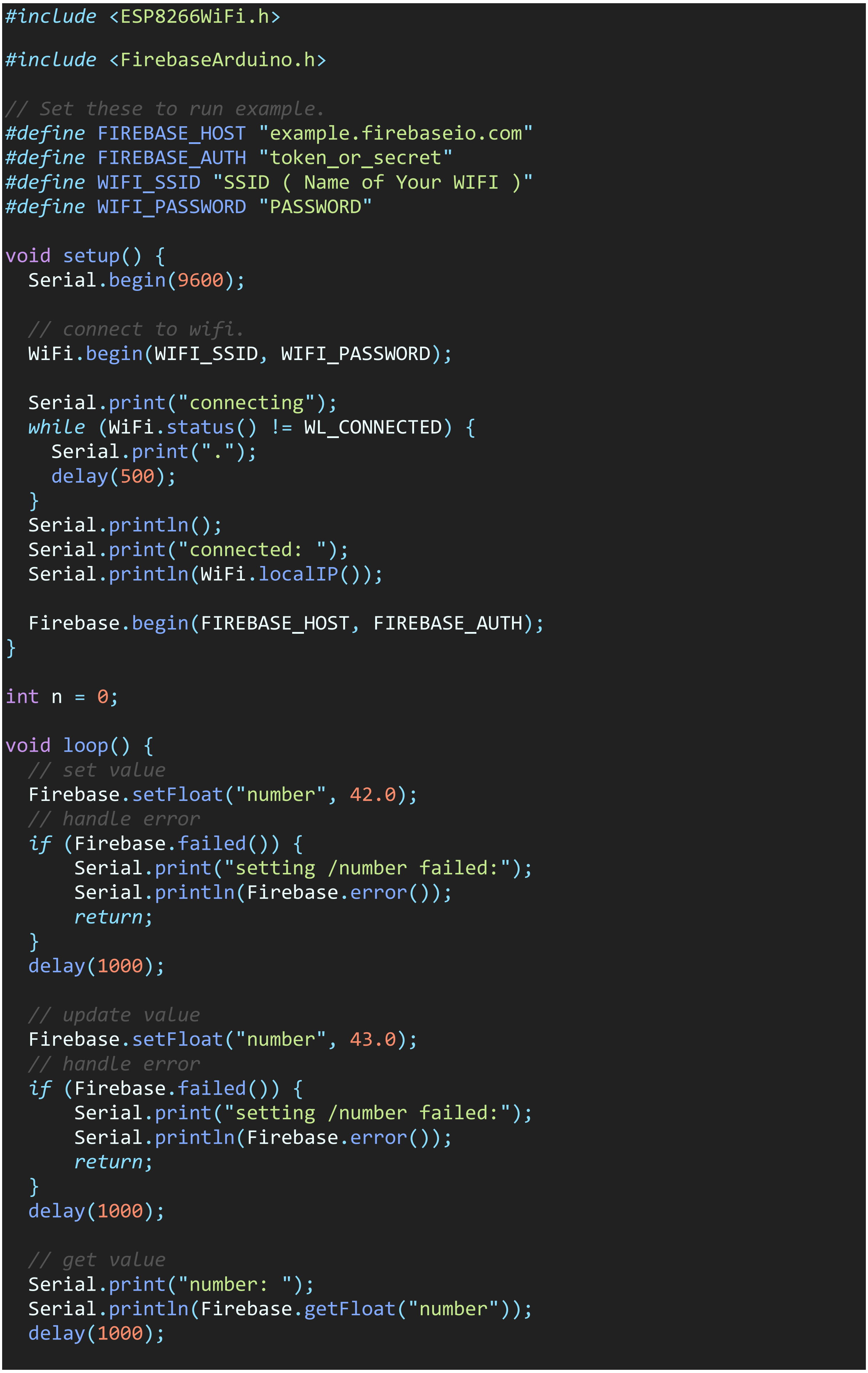


```
//                                   e for the specific language governing permissions and
//                                   e.
//

// FirebaseDemo_E                    mple that demo the different functions
// of the Firebas

#include <ESP8266
#include <Firebas

// Set these to r
#define FIREBASE_                    .firebaseio.com"
#define FIREBASE_                    aMjV6r2mmGYupwHQQ9VhCilj12m3xzMR"
#define WIFI_SSID
#define WIFI_PASS                    "

void setup() {
  Serial.begin(96
```

Click on **sketch** >> **include library** >> **Add Zip Library** >> select the Downloaded Zip File

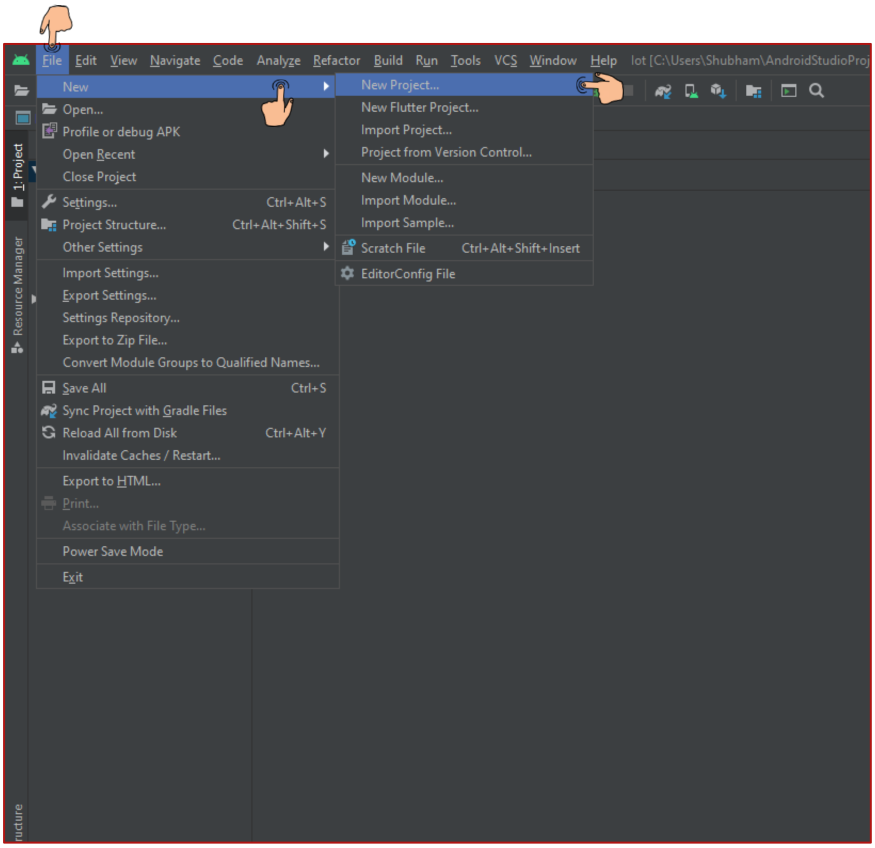
- **NodeMCU-ESP8266**
- **Firebase Database (Read Earlier)**

Now let's Understand Whole thing by a sample Code.

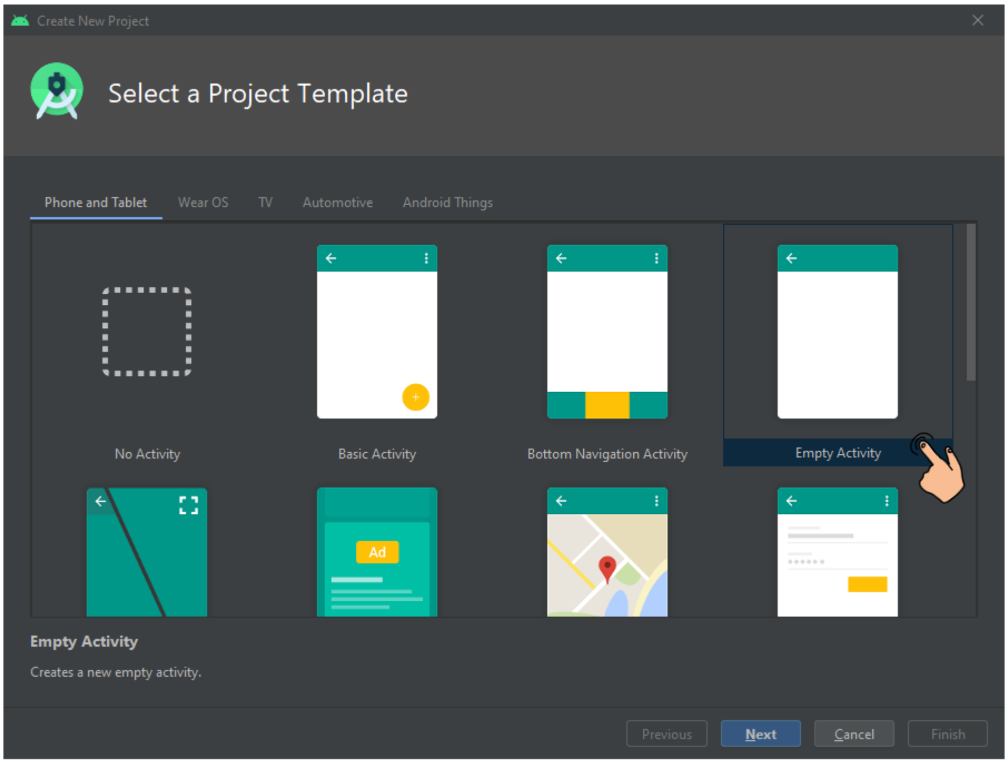


```
#include <ESP8266WiFi.h>

#include <FirebaseArduino.h>

// Set these to run example.
#define FIREBASE_HOST "example.firebaseio.com"
#define FIREBASE_AUTH "token_or_secret"
#define WIFI_SSID "SSID ( Name of Your WIFI )"
#define WIFI_PASSWORD "PASSWORD"

void setup() {
  Serial.begin(9600);

  // connect to wifi.
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

  Serial.print("connecting");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }
  Serial.println();
  Serial.print("connected: ");
  Serial.println(WiFi.localIP());

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
}

int n = 0;

void loop() {
  // set value
  Firebase.setFloat("number", 42.0);
  // handle error
  if (Firebase.failed()) {
      Serial.print("setting /number failed:");
      Serial.println(Firebase.error());
      return;
  }
  delay(1000);

  // update value
  Firebase.setFloat("number", 43.0);
  // handle error
  if (Firebase.failed()) {
      Serial.print("setting /number failed:");
      Serial.println(Firebase.error());
      return;
  }
  delay(1000);

  // get value
  Serial.print("number: ");
  Serial.println(Firebase.getFloat("number"));
  delay(1000);
  // remove value
  Firebase.remove("number");
  delay(1000);

  // set string value
  Firebase.setString("message", "hello world");
  // handle error
  if (Firebase.failed()) {
      Serial.print("setting /message failed:");
      Serial.println(Firebase.error());
      return;
  }
  delay(1000);

  // set bool value
  Firebase.setBool("truth", false);
  // handle error
  if (Firebase.failed()) {
      Serial.print("setting /truth failed:");
      Serial.println(Firebase.error());
      return;
  }
  delay(1000);

  // append a new value to /logs
  String name = Firebase.pushInt("logs", n++);
  // handle error
  if (Firebase.failed()) {
      Serial.print("pushing /logs failed:");
      Serial.println(Firebase.error());
      return;
  }
  Serial.print("pushed: /logs/");
  Serial.println(name);
  delay(1000);
}
```

1. Open Android Studio and Create a New Project **File >> New >> New Project**



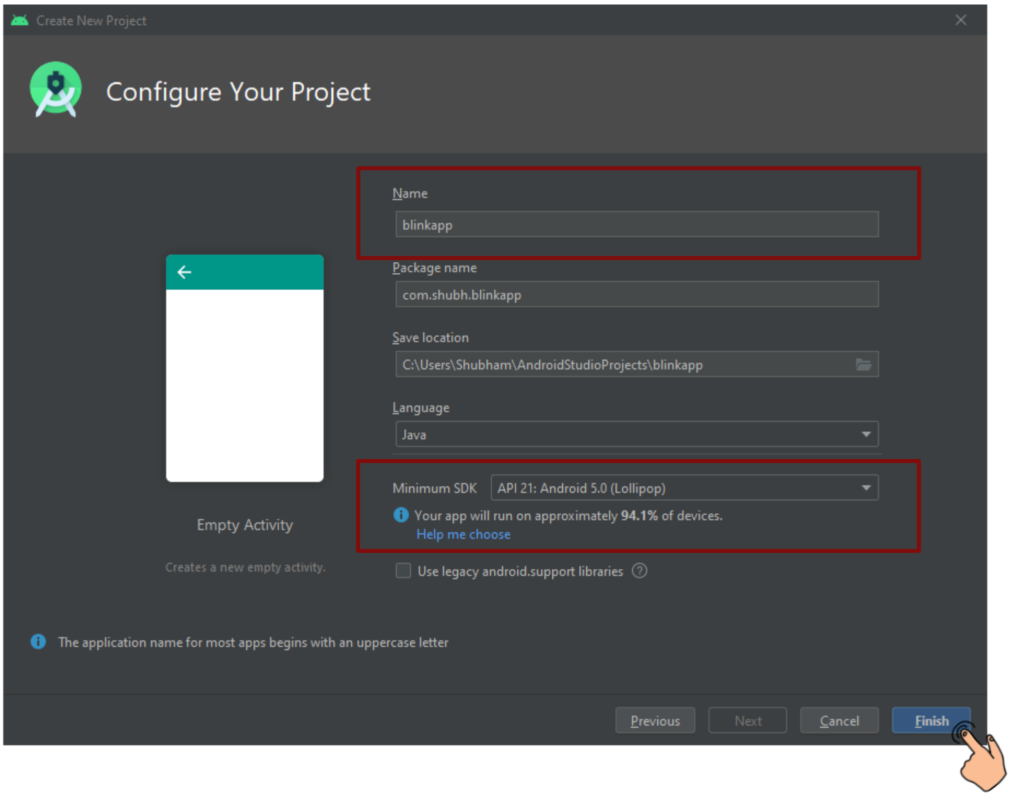
2. Select Empty Activity



Empty Activity

Creates a new empty activity.

3. Fill Project name & Select android Version above 5.0 to run across all the devices and click on finish.



Name: blinkapp

Package name: com.shubh.blinkapp

Save location: C:\Users\Shubham\AndroidStudioProjects\blinkapp

Language: Java

Minimum SDK: API 21: Android 5.0 (Lollipop)

Your app will run on approximately **94.1%** of devices.
Help me choose

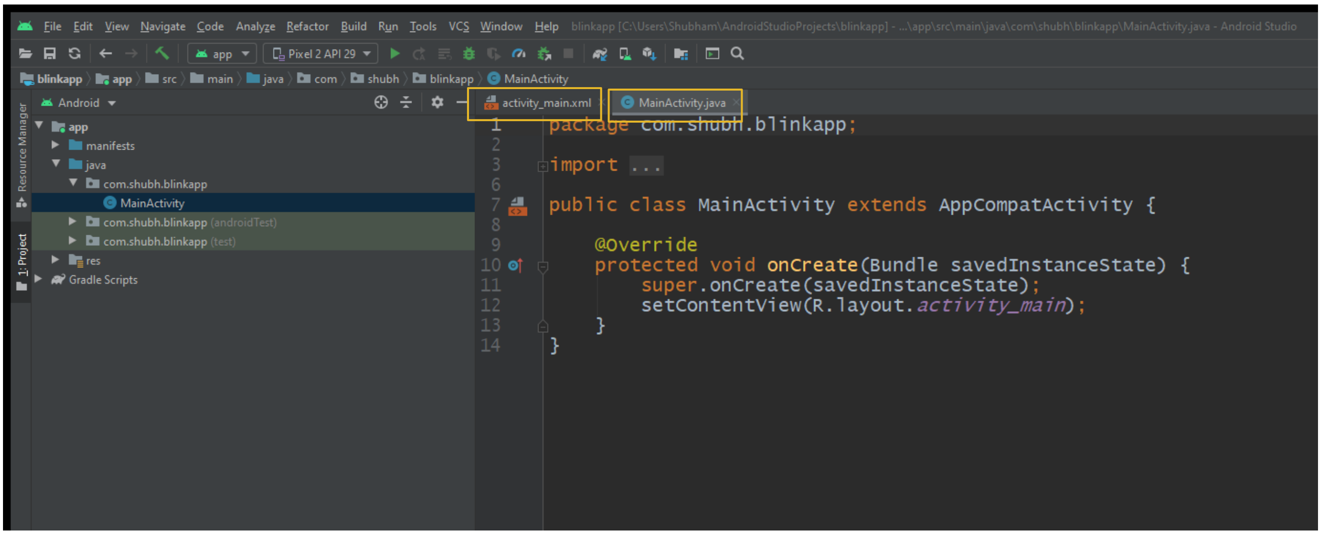
□ Use legacy android.support libraries

The application name for most apps begins with an uppercase letter

4. Wait, it will take some time to load your project when your project is loaded, you'll get the screen like this and there you find 2 files with the names
   i. Activity_main.xml
      - This File is responsible for Interface or Design of our Android App
   ii. MainActivity.java
      - This File is the backend for our Interface

Note :

In context of web Development, there also we need two languages **HTML** for Design & **PHP** for backend, In similar way here **xml** is used for designing and **java** is used for backend.



File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
blinkapp [C:\Users\Shubham\AndroidStudioProjects\blinkapp] - ...\app\src\main\java\com\shubh\blinkapp\MainActivity.java - Android Studio
app
Pixel 2 API 29
blinkapp
app
src
main
java
com
shubh
blinkapp
MainActivity
Android
app
manifests
java
com.shubh.blinkapp
MainActivity
com.shubh.blinkapp (androidTest)
com.shubh.blinkapp (test)
res
Gradle Scripts
Resource Manager
1: Project
activity_main.xml
MainActivity.java

```
package com.shubh.blinkapp;

import ...

public class MainActivity extends AppCompatActivity {

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

5. Now Design a simple interface from Activity_main.xml file by using drag and drop feature.

Task : Create & configure a firebase database named **home** with attribute **status** can have either **0** or **1** in its value, Create a button in Android that will update status attribute of led database when it is Pressed but first you have to check that previous value which is already present there in the status attribute of the led database and then update its complement value.

For example :

If 0 is already present in the status attribute of home database you have to update that value to 1, if user pressed that button and vice versa changes should take place.

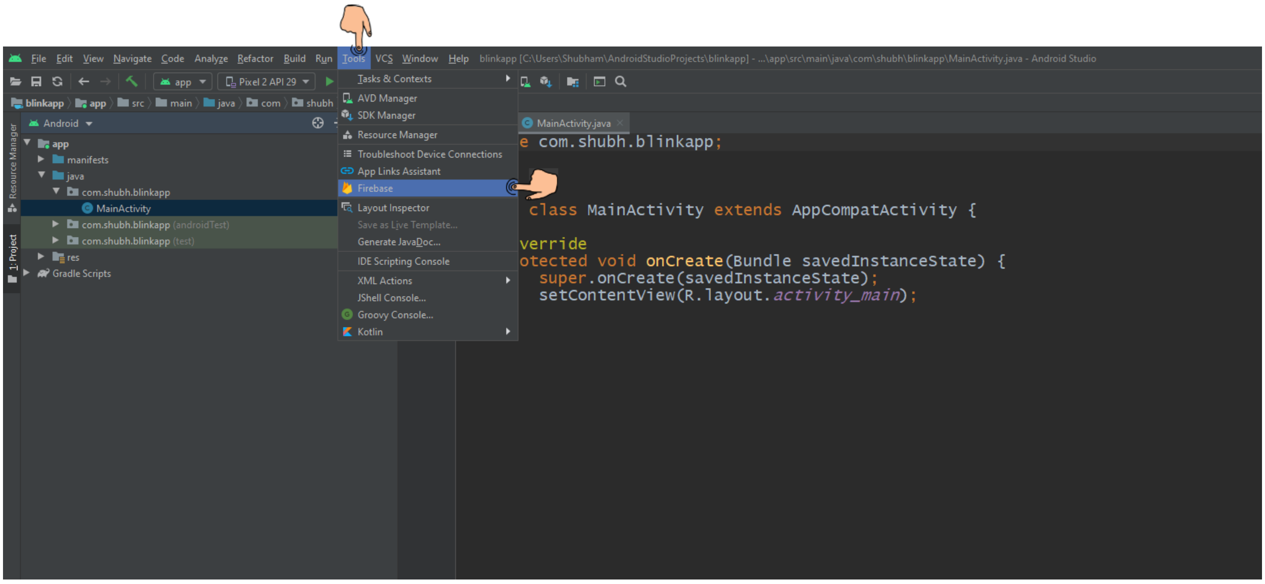
See Demo video

Solution : I have divided this problem into 3 Steps.

1. **Configure Firebase from Android Studio itself**
2. **Create sample Schema or Databases**
3. **Synchronisation of database with Android Application**

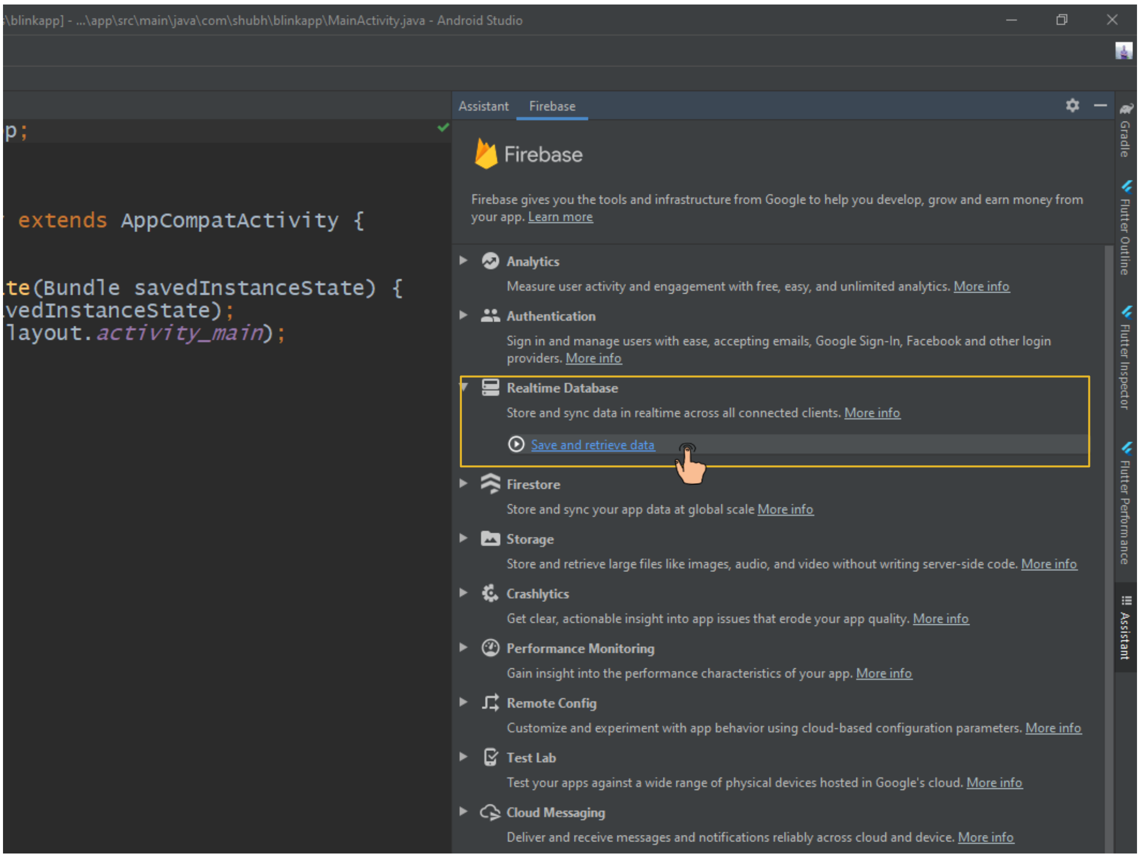
CONFIGURE FIREBASE FROM ANDROID STUDIO

1. Click on Tool >> **Firebase**



```
e com.shubh.blinkapp;

 class MainActivity extends AppCompatActivity {

verride
otected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
```

2. A new pane/widget should open.
   Find **Realtime Database** option >> Click on **Save and Retreive Data.**


\blinkapp] - ...\app\src\main\java\com\shubh\blinkapp\MainActivity.java - Android Studio
Assistant
Firebase
Gradle
Flutter Outline
Flutter Inspector
Flutter Performance
Assistant

```
p;

 extends AppCompatActivity {

te(Bundle savedInstanceState) {
vedInstanceState);
layout.activity_main);
```

Firebase

Firebase gives you the tools and infrastructure from Google to help you develop, grow and earn money from your app. Learn more

Analytics
Measure user activity and engagement with free, easy, and unlimited analytics. More info

Authentication
Sign in and manage users with ease, accepting emails, Google Sign-In, Facebook and other login providers. More info

Realtime Database
Store and sync data in realtime across all connected clients. More info

Save and retrieve data

Firestore
Store and sync your app data at global scale More info

Storage
Store and retrieve large files like images, audio, and video without writing server-side code. More info

Crashlytics
Get clear, actionable insight into app issues that erode your app quality. More info

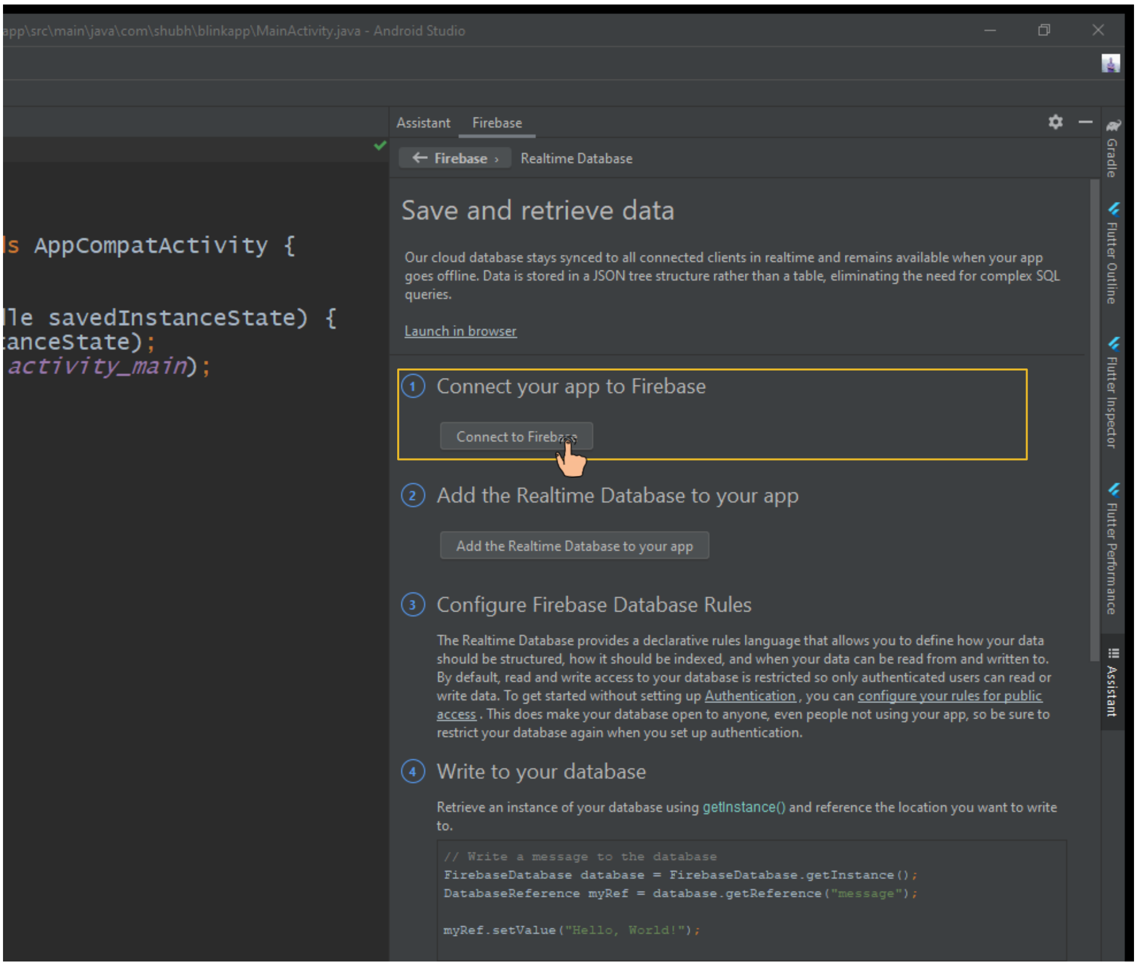
Performance Monitoring
Gain insight into the performance characteristics of your app. More info

Remote Config
Customize and experiment with app behavior using cloud-based configuration parameters. More info

Test Lab
Test your apps against a wide range of physical devices hosted in Google's cloud. More info

Cloud Messaging
Deliver and receive messages and notifications reliably across cloud and device. More info

3. Click on **connect to firebase.**



```
ls AppCompatActivity {

lle savedInstanceState) {
tanceState);
 activity_main);
```

Save and retrieve data

Our cloud database stays synced to all connected clients in realtime and remains available when your app goes offline. Data is stored in a JSON tree structure rather than a table, eliminating the need for complex SQL queries.

Launch in browser

1. Connect your app to Firebase

   Connect to Firebase

2. Add the Realtime Database to your app

   Add the Realtime Database to your app

3. Configure Firebase Database Rules

   The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to. By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up Authentication , you can configure your rules for public access . This does make your database open to anyone, even people not using your app, so be sure to restrict your database again when you set up authentication.

4. Write to your database

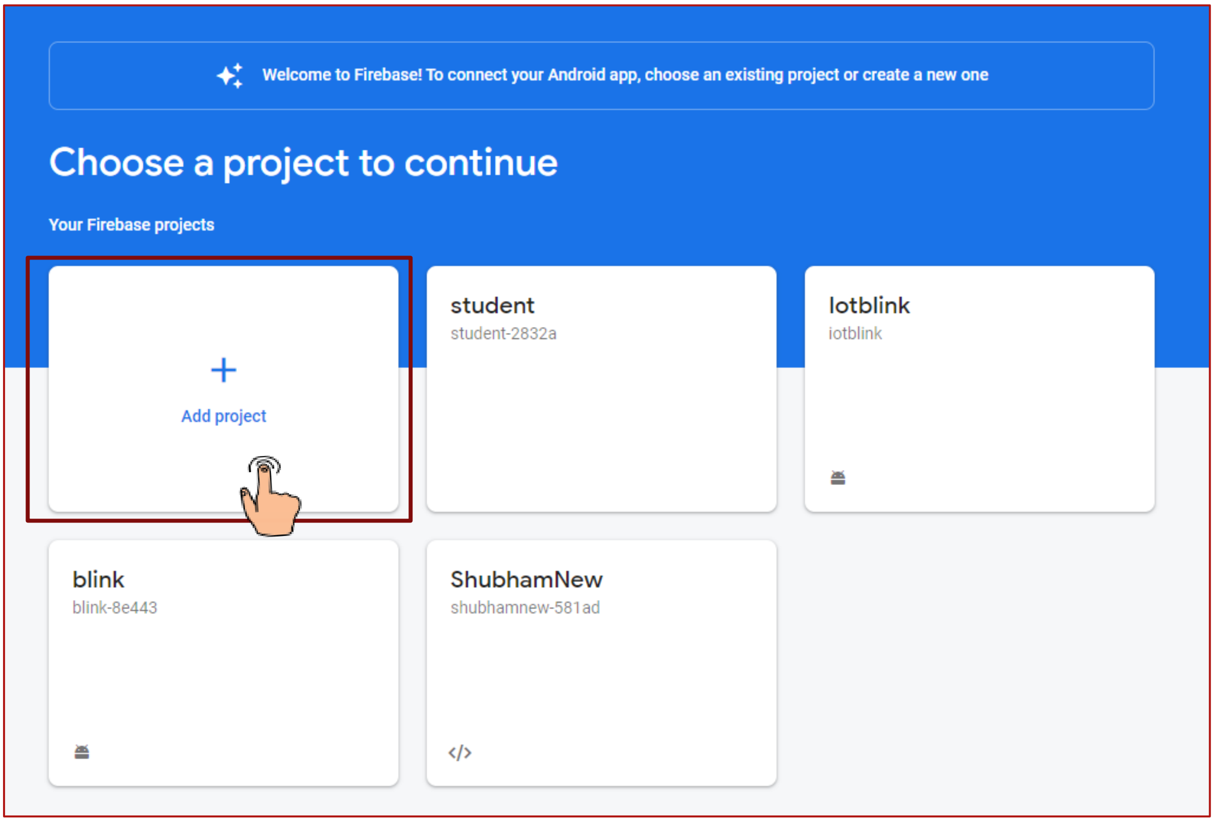
   Retrieve an instance of your database using getInstance() and reference the location you want to write to.

```
// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference("message");

myRef.setValue("Hello, World!");
```

4. A web browser will open, login with your Gmail account and go to firebase console.

Create **Firebase database with name home** with **status attribute** to it.

4.1. Click on Add Project

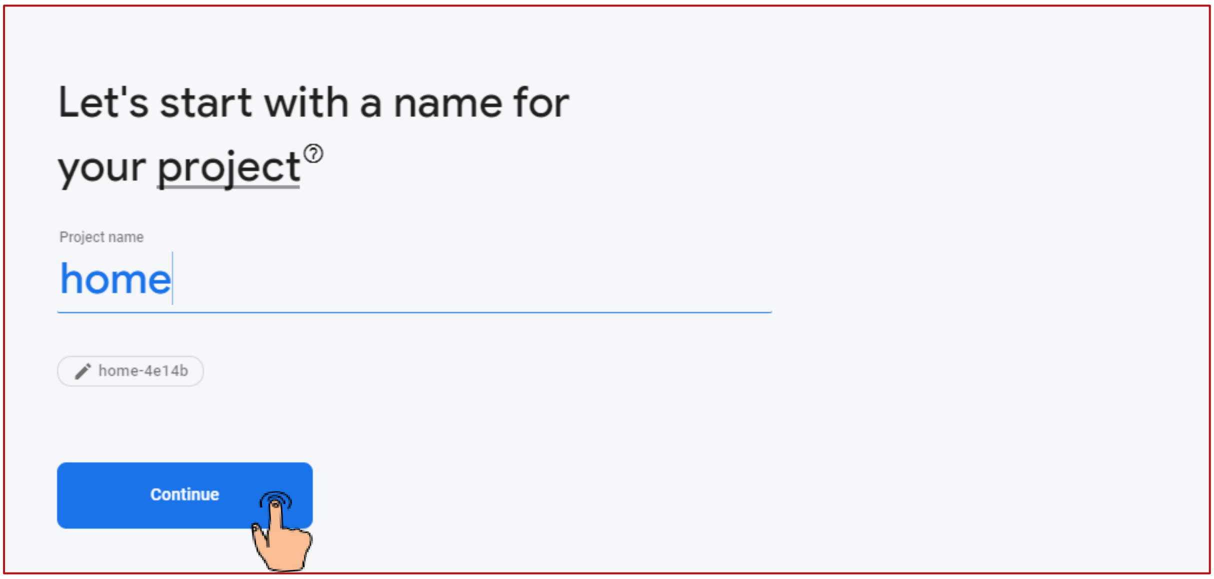


Welcome to Firebase! To connect your Android app, choose an existing project or create a new one

Choose a project to continue

Your Firebase projects

4.2. Enter Project Name home

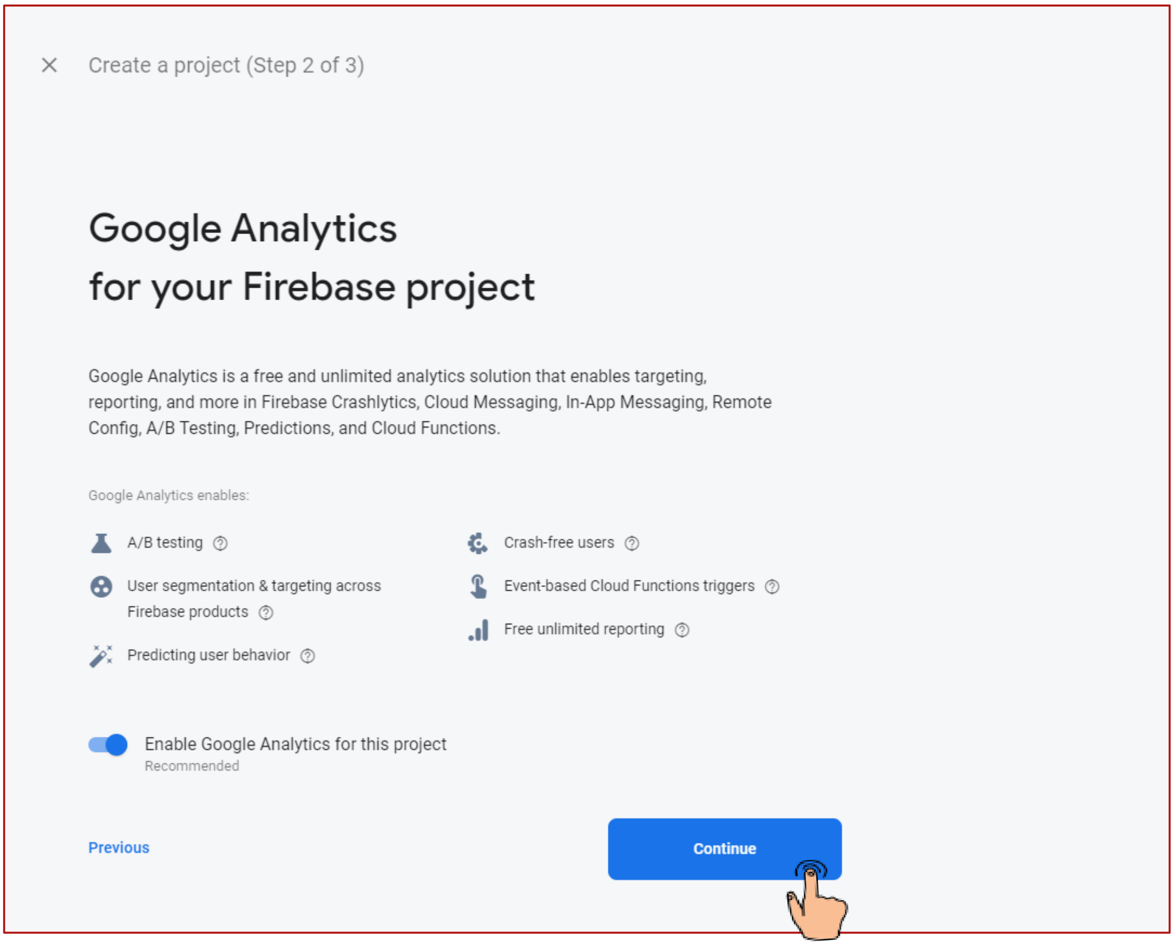


Let's start with a name for your project

Project name: home

home-4e14b

4.3. Click on continue and again continue and then click on create project (As shown in picture).



Create a project (Step 2 of 3)

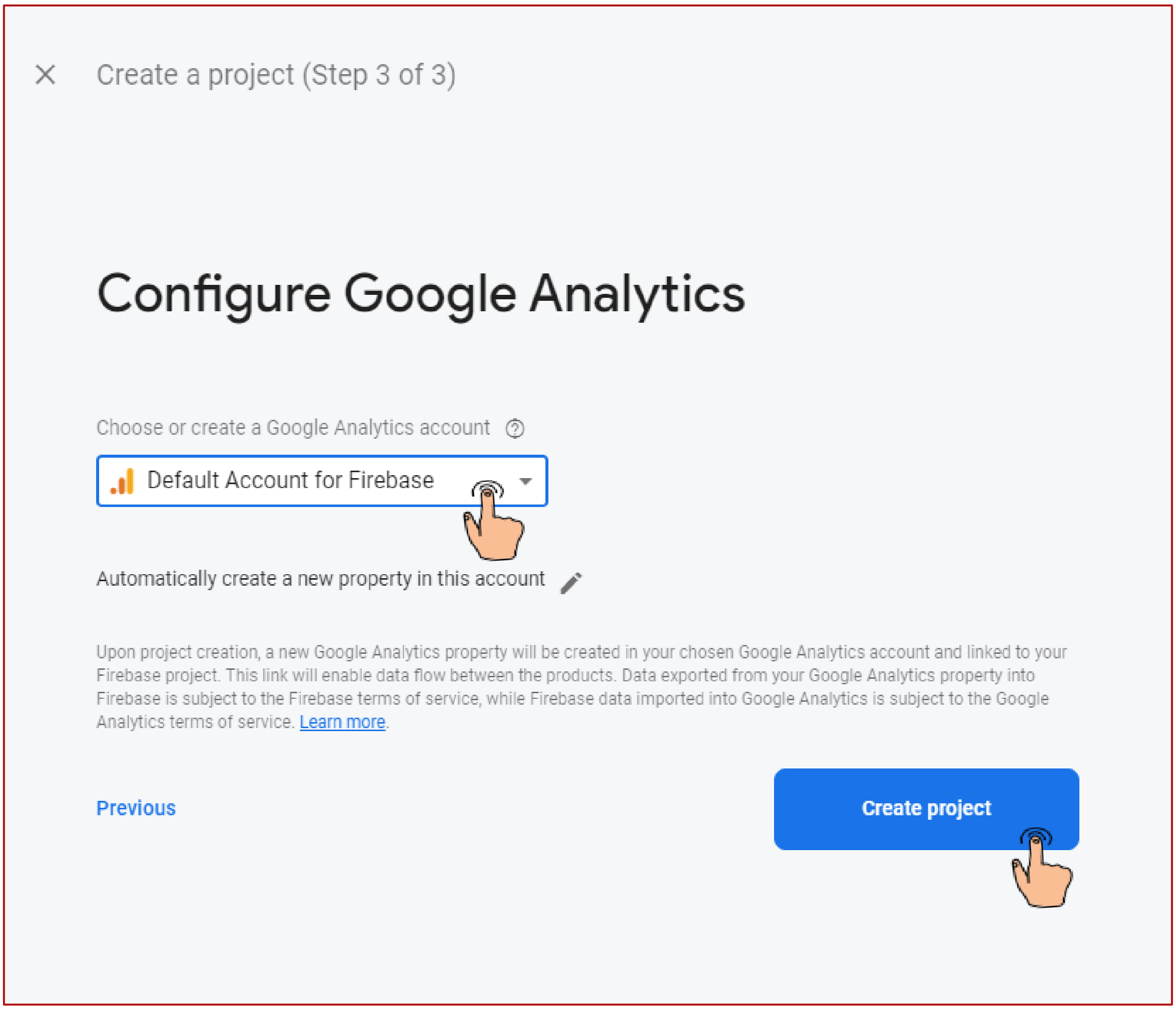
Google Analytics
for your Firebase project

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, Predictions, and Cloud Functions.

Google Analytics enables:

- A/B testing
- User segmentation & targeting across Firebase products
- Predicting user behavior
- Crash-free users
- Event-based Cloud Functions triggers
- Free unlimited reporting

Enable Google Analytics for this project
Recommended



Create a project (Step 3 of 3)

Configure Google Analytics

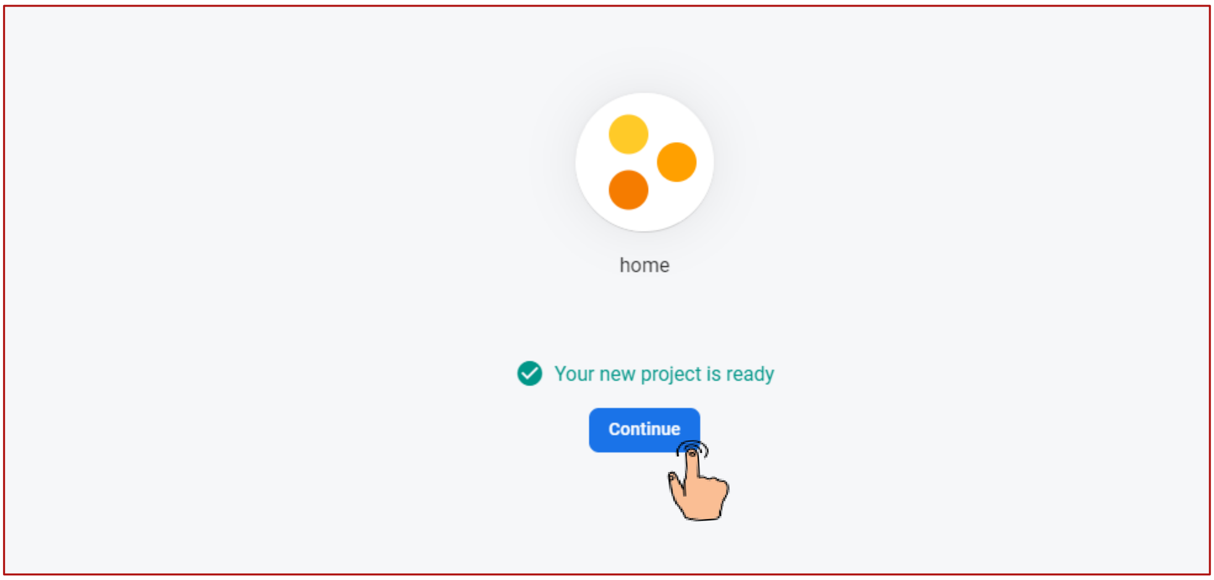
Choose or create a Google Analytics account

Default Account for Firebase

Automatically create a new property in this account

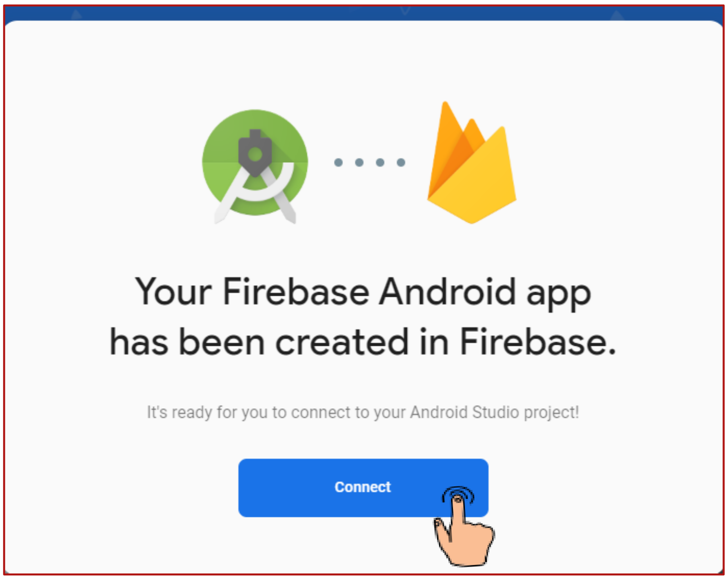
Upon project creation, a new Google Analytics property will be created in your chosen Google Analytics account and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. Learn more.

4.4. Now you have successfully created your project **click on continue**.



Your new project is ready

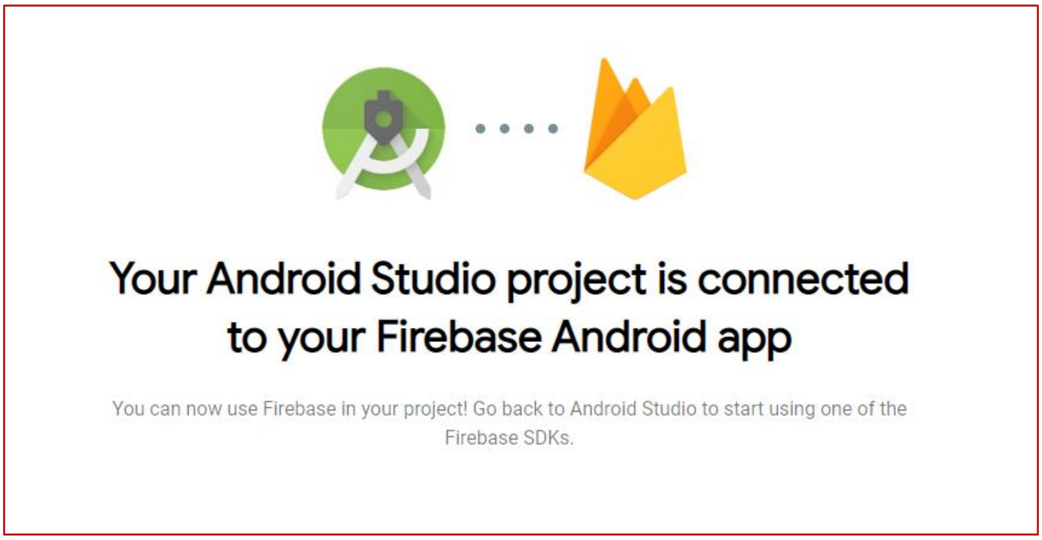
4.5. Now you'll get a message like this, **Click on Connect**.



Your Firebase Android app has been created in Firebase.

It's ready for you to connect to your Android Studio project!

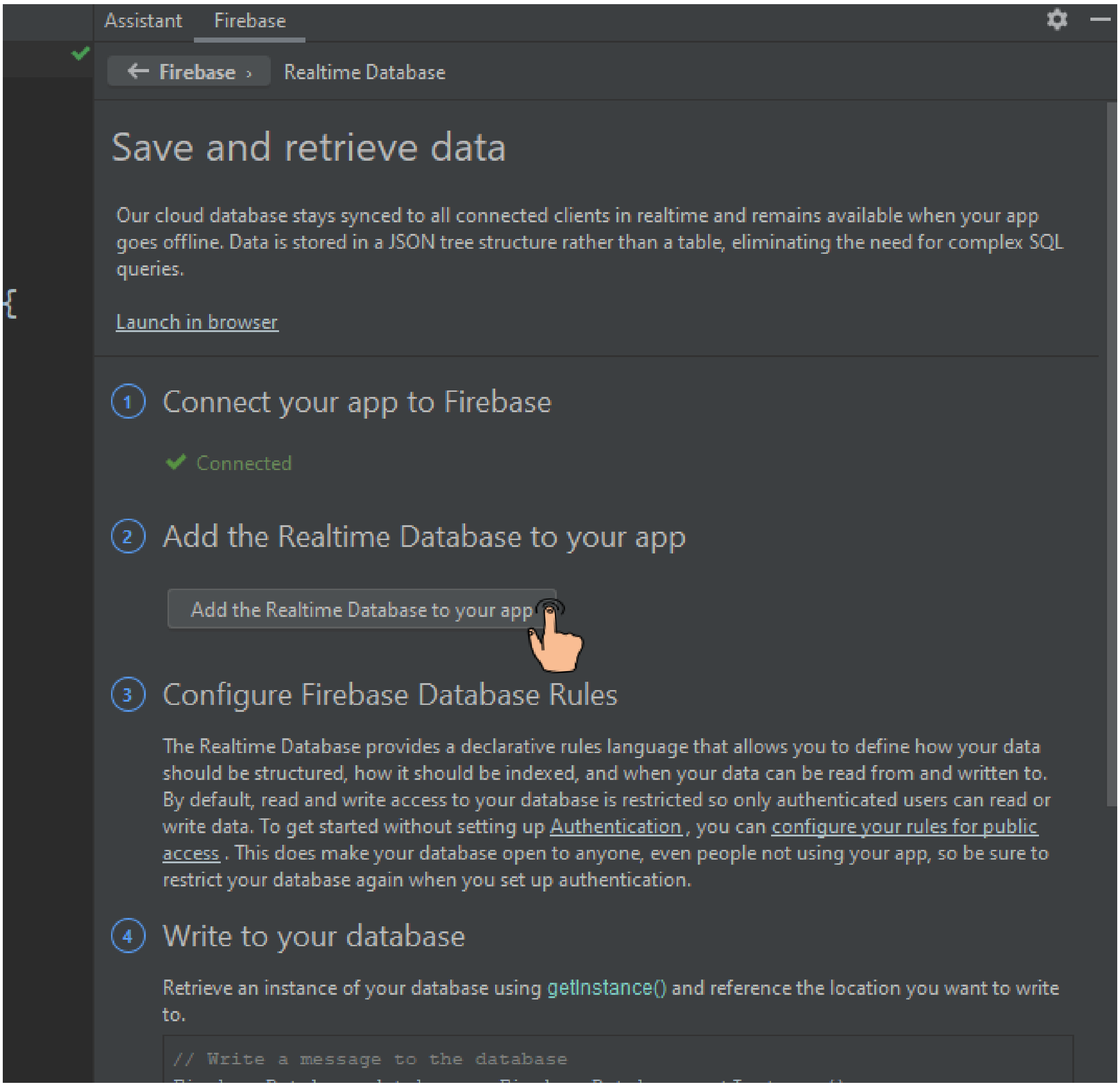
4.6. Now your android project is connected with database and you will be able to see the screen like this.



Your Android Studio project is connected to your Firebase Android app

You can now use Firebase in your project! Go back to Android Studio to start using one of the Firebase SDKs.

5. Now go to Android Studio and click on **Add the Realtime Database to your app**



Save and retrieve data

Our cloud database stays synced to all connected clients in realtime and remains available when your app goes offline. Data is stored in a JSON tree structure rather than a table, eliminating the need for complex SQL queries.

Launch in browser

1. Connect your app to Firebase

✓ Connected

2. Add the Realtime Database to your app

Add the Realtime Database to your app

3. Configure Firebase Database Rules

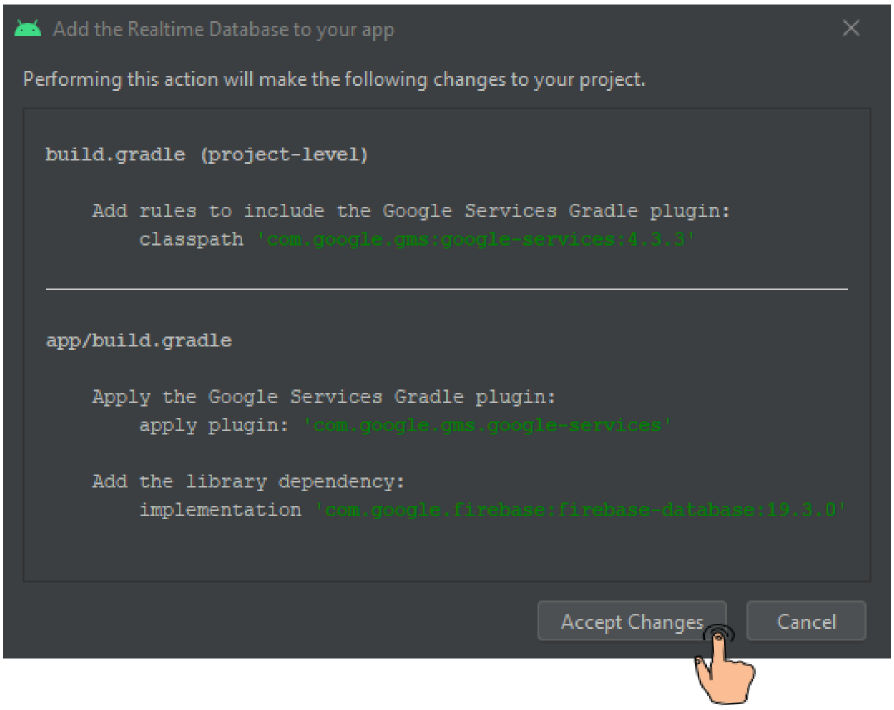
The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to. By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up Authentication , you can configure your rules for public access . This does make your database open to anyone, even people not using your app, so be sure to restrict your database again when you set up authentication.

4. Write to your database

Retrieve an instance of your database using getInstance() and reference the location you want to write to.

```
// Write a message to the database
```

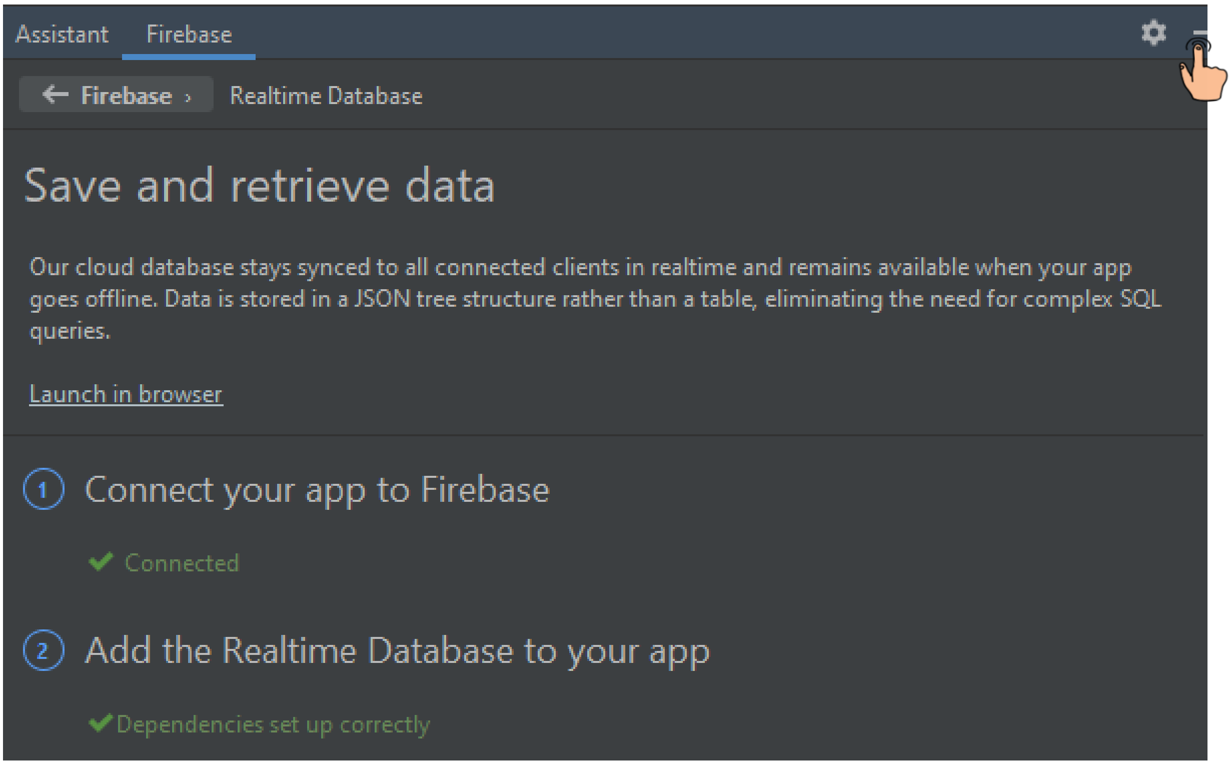
6. **Click on Accept Changes**



Performing this action will make the following changes to your project.

```
build.gradle (project-level)

    Add rules to include the Google Services Gradle plugin:
        classpath 'com.google.gms:google-services:4.3.3'

app/build.gradle

    Apply the Google Services Gradle plugin:
        apply plugin: 'com.google.gms.google-services'

    Add the library dependency:
        implementation 'com.google.firebase:firebase-database:19.3.0'
```

7. It will load for few seconds and you will get a status like this, now close this panel and get back to firebase.



Save and retrieve data

Our cloud database stays synced to all connected clients in realtime and remains available when your app goes offline. Data is stored in a JSON tree structure rather than a table, eliminating the need for complex SQL queries.

Launch in browser

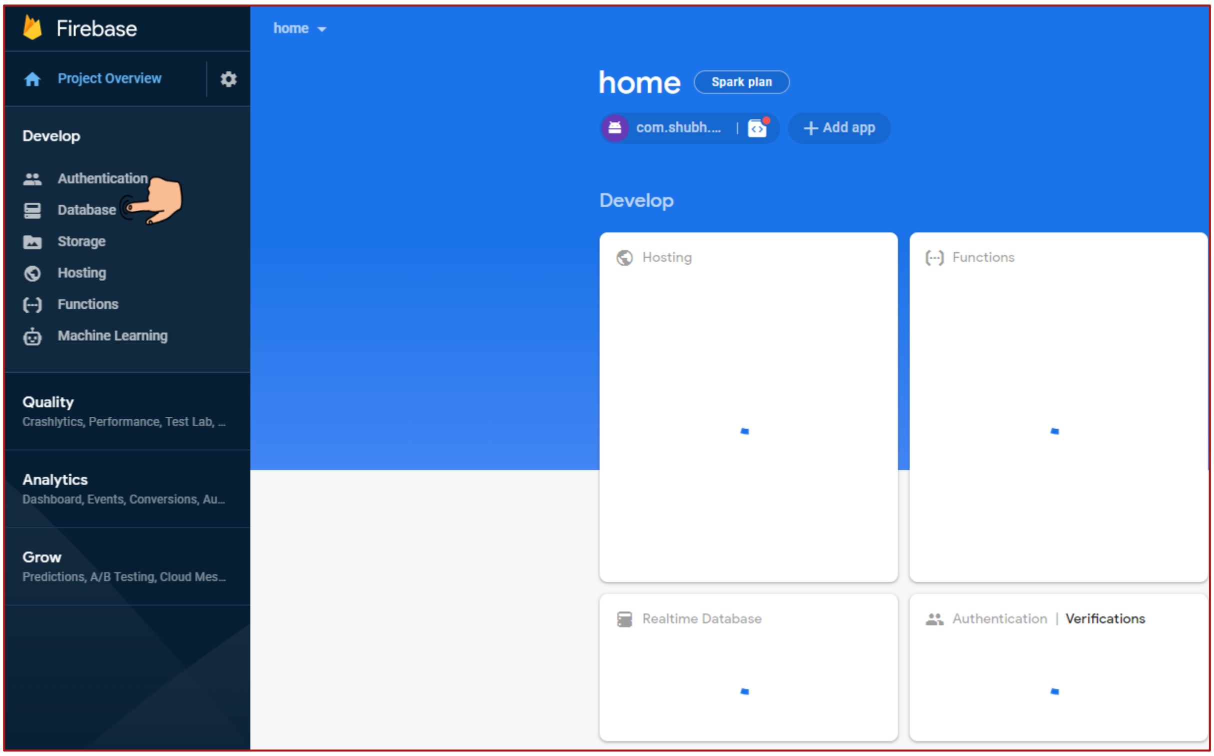
1 Connect your app to Firebase

✓ Connected

2 Add the Realtime Database to your app

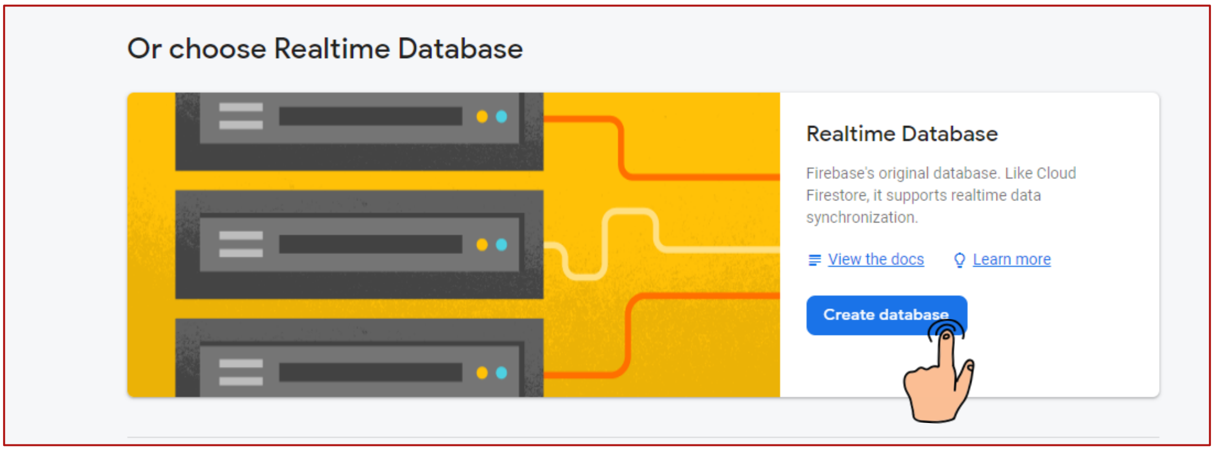
✓ Dependencies set up correctly

8. Now click on the database.



Firebase
Project Overview
Develop
Authentication
Database
Storage
Hosting
Functions
Machine Learning
Quality
Crashlytics, Performance, Test Lab, ...
Analytics
Dashboard, Events, Conversions, Au...
Grow
Predictions, A/B Testing, Cloud Mes...
home
home
Spark plan
com.shubh....
Add app
Develop
Hosting
Functions
Realtime Database
Authentication | Verifications

9. Scroll Down and find Realtime Database and click on **create Database** button

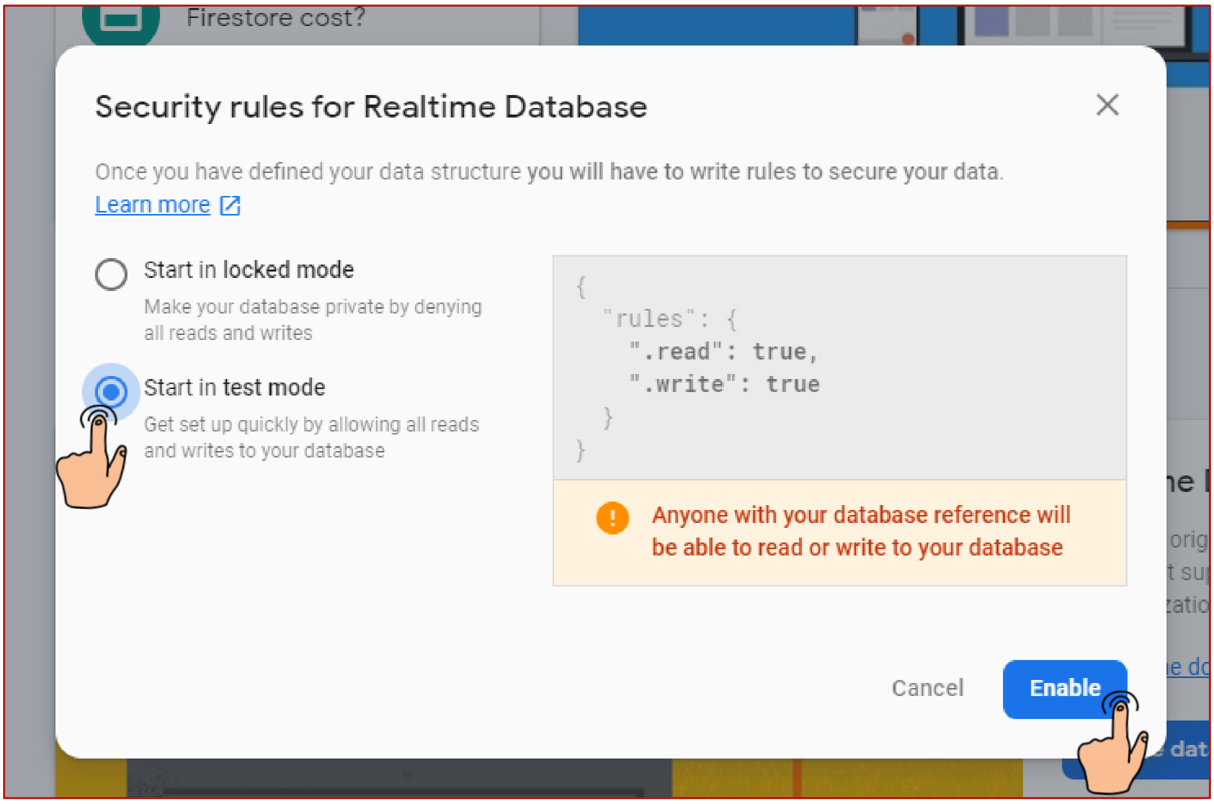


Or choose Realtime Database

Realtime Database

Firebase's original database. Like Cloud Firestore, it supports realtime data synchronization.

10. Click **start in test mode** and click on **Enable**.



Security rules for Realtime Database

Once you have defined your data structure you will have to write rules to secure your data. Learn more

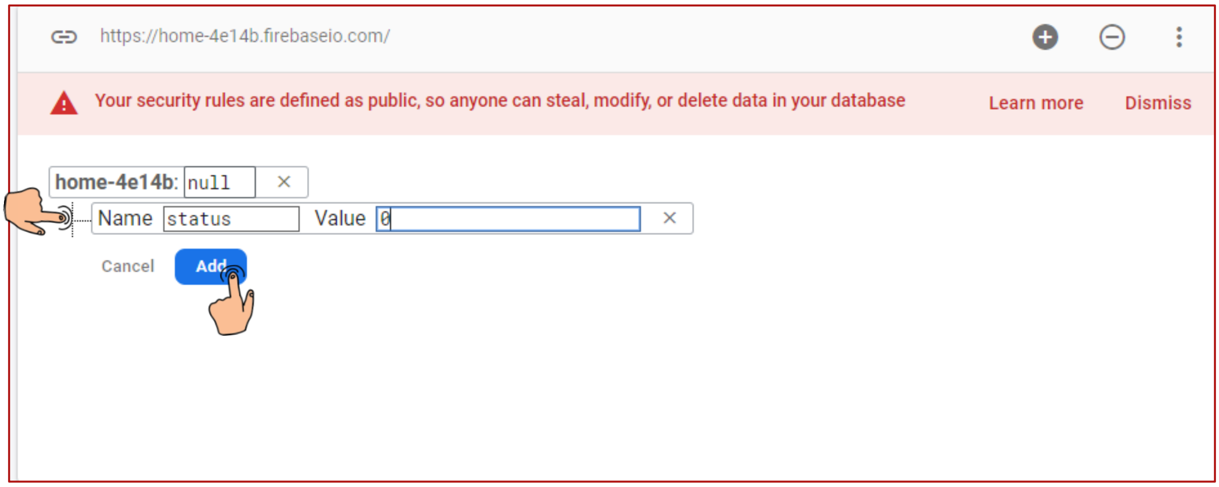
○ Start in locked mode
Make your database private by denying all reads and writes

● Start in test mode
Get set up quickly by allowing all reads and writes to your database

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

Anyone with your database reference will be able to read or write to your database

11. Now Create an attribute with name **status** with default value 0 and **click on Add Button** as shown below.

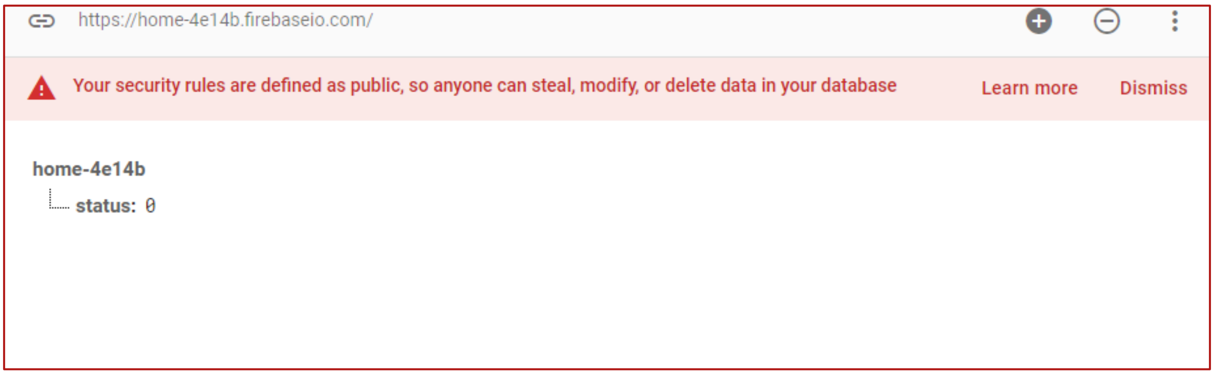


https://home-4e14b.firebaseio.com/

Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

home-4e14b: null

Name status Value 0

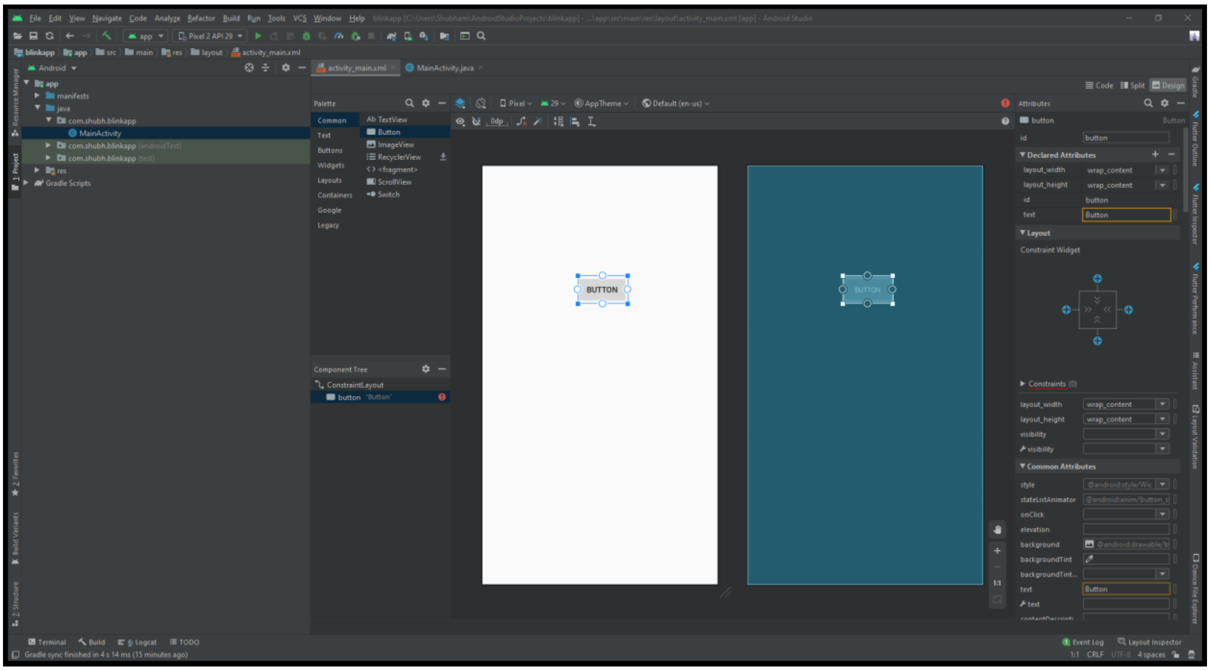


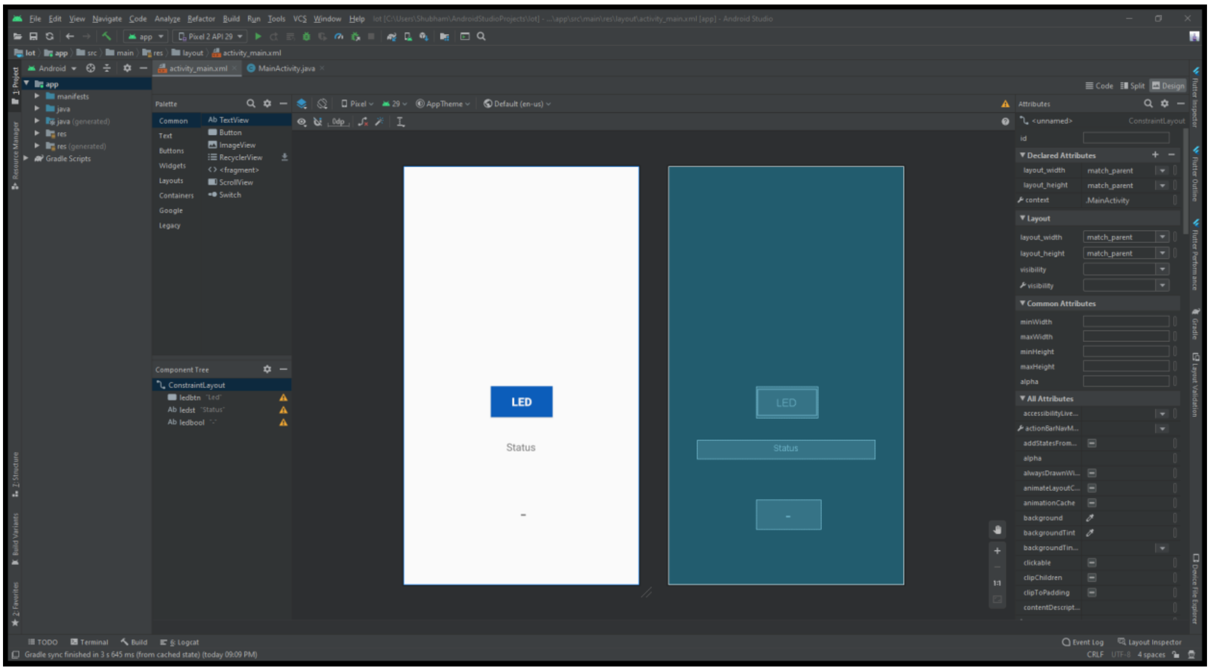
https://home-4e14b.firebaseio.com/

Your security rules are defined as public, so anyone can steal, modify, or delete data in your database

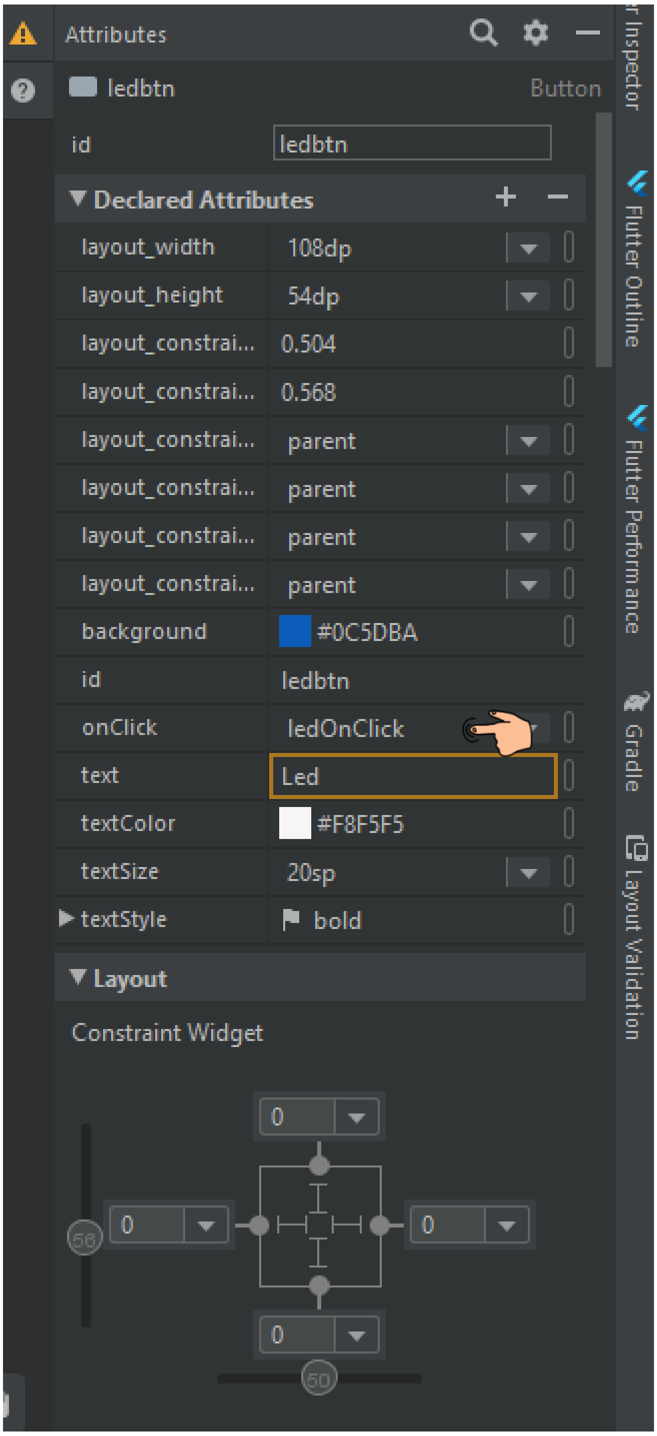
home-4e14b
status: 0

12. Now go back to android studio, we will create basic interface of our android app using drag and drop feature. Shown below.





13. Configure a onclick event on button led (Search **onclick** in attribute panel of button and give it a name) and define it on code.



ledbtn

id: ledbtn

Declared Attributes

layout_width	108dp
layout_height	54dp
layout_constrai...	0.504
layout_constrai...	0.568
layout_constrai...	parent
layout_constrai...	parent
layout_constrai...	parent
layout_constrai...	parent
background	🟦 #0C5DBA
id	ledbtn
onClick	ledOnClick
text	Led
textColor	⬜ #F8F5F5
textSize	20sp
textStyle	bold

Layout

Constraint Widget

Code of Activty_main.xml (Above Designing)

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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=".MainActivity">

    <Button
        android:id="@+id/ledbtn"
        android:layout_width="108dp"
        android:layout_height="54dp"
        android:background="#0C5DBA"
        android:onClick="ledOnClick"
        android:text="Led"
        android:textColor="#F8F5F5"
        android:textSize="20sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.504"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.568" />

    <TextView
        android:id="@+id/ledst"
        android:layout_width="311dp"
        android:layout_height="33dp"
        android:text="Status"
        android:textAlignment="center"
        android:textSize="18sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ledbtn"
        app:layout_constraintVertical_bias="0.154" />

    <TextView
        android:id="@+id/ledbool"
        android:layout_width="113dp"
        android:layout_height="51dp"
        android:text="-"
        android:textAlignment="center"
        android:textSize="35sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.513"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.857" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Code of MainActivity.java(Backend)

```
package com.shubh.iot;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.TextView;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class MainActivity extends AppCompatActivity {
    FirebaseDatabase database= FirebaseDatabase.getInstance();
    TextView obj;
    TextView obj1;

    DatabaseReference reff,reff1;
    int val;

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

    }
    public void ledOnClick(View V){
        obj1=(TextView)findViewById(R.id.ledst);
        reff1=database.getReference("led");
        if(val==0){
            reff1.setValue(1);
            obj1.setText("On Request Sent");
        }else{
            reff1.setValue(0);
            obj1.setText("Off Request Sent");
        }
        check_status();

    }
    public void check_status(){
        obj=(TextView)findViewById(R.id.ledbool);
        reff=FirebaseDatabase.getInstance().getReference();
        reff.child("led").addListenerForSingleValueEvent(new
ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                String value=dataSnapshot.getValue().toString();

                val=Integer.parseInt(value);

                if(value.equals("0")){
                    obj.setText("OFF");
                }else{
                    obj.setText("ON");
                }

            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError)
{
                obj.setText("Error");
            }
        });
    }
}
```