**Name:** Yukta Bhatia

**RollNo:** 07

**Class:** D15A

**Experiment-6**

**Aim:**  To connect Flutter UI with Firebase Database.

Theory :

Prerequisites :

# To complete this tutorial, you will need:

* A Google account to use Firebase.

# Developing for iOS will require XCode.

* To download and install [Flutter](https://flutter.dev/docs/get-started/install).

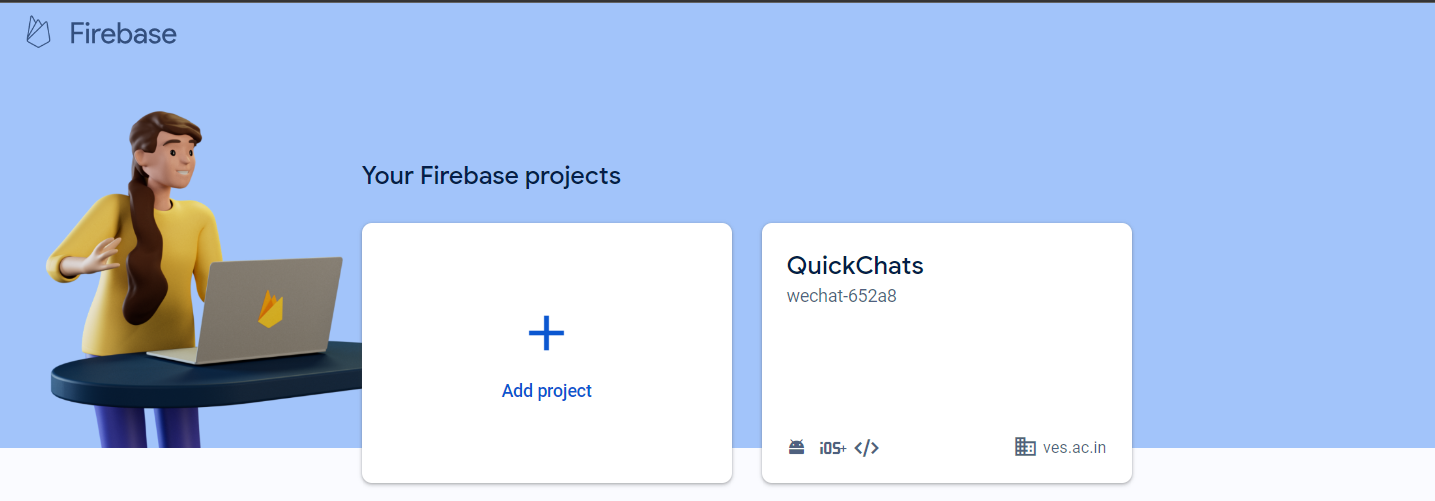
# To download and install [Android Studio](https://developer.android.com/studio) and [Visual Studio Code](https://code.visualstudio.com/).

* It is recommended to install plugins for your code editor:

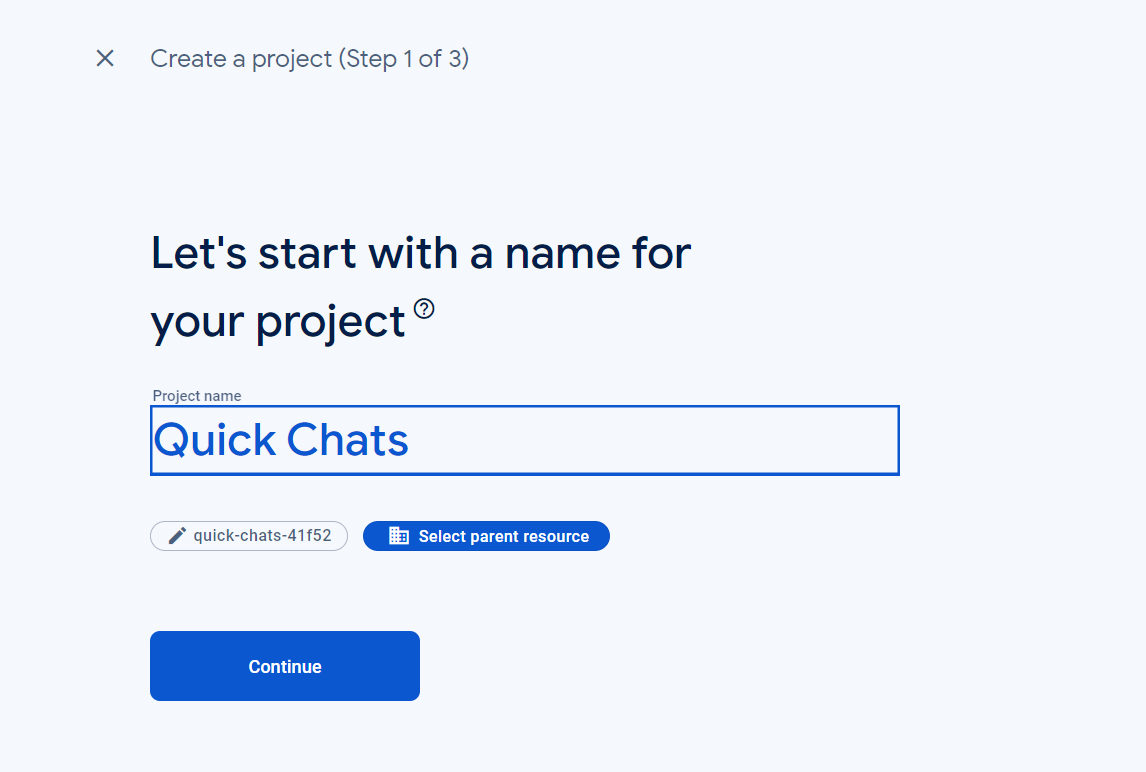
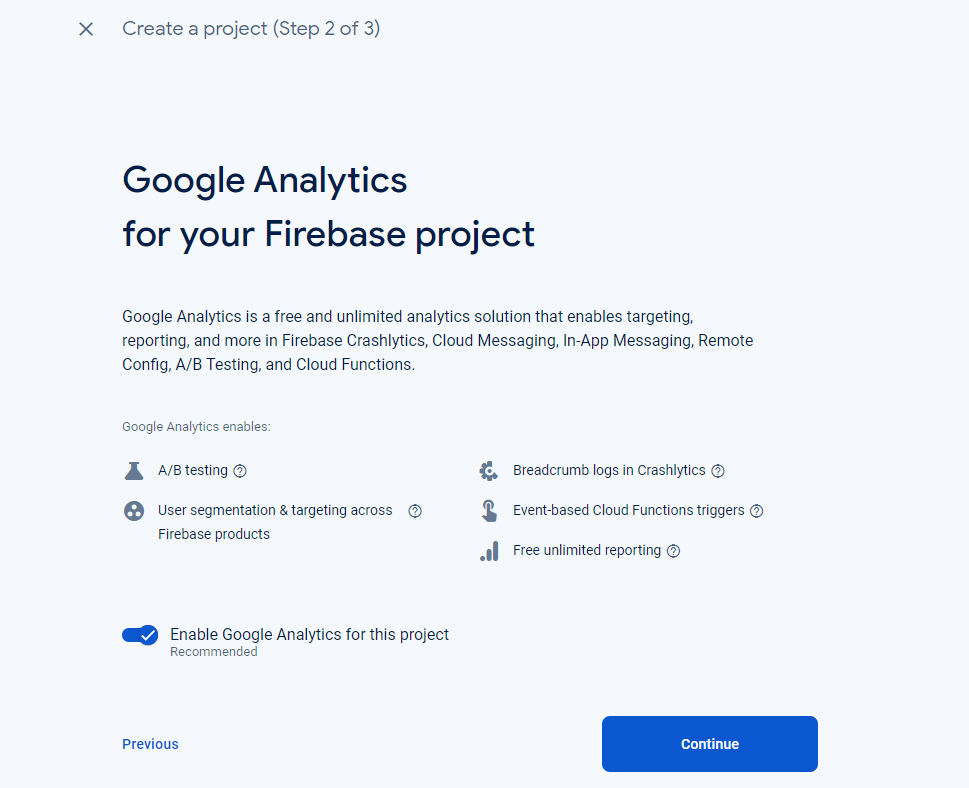
# [Flutter](https://plugins.jetbrains.com/plugin/9212-flutter) and [Dart](https://plugins.jetbrains.com/plugin/6351-dart) plugins installed for Android Studio.

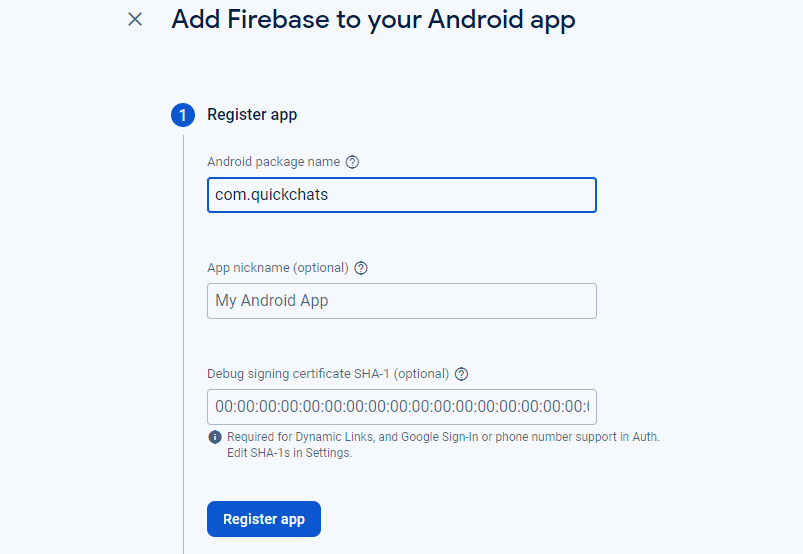
* + [Flutter](https://marketplace.visualstudio.com/items?itemName=Dart-Code.flutter) extension installed for Visual Studio Code.

# Creating a New Firebase Project :

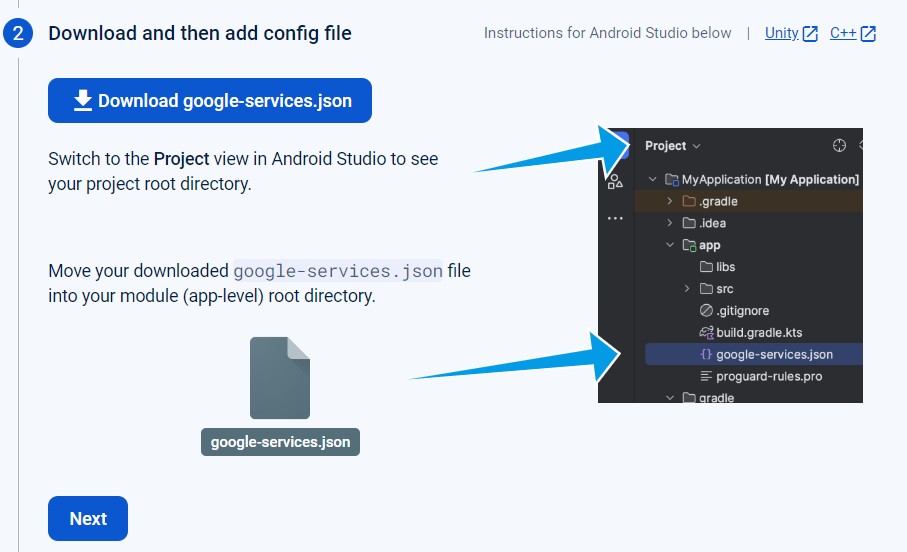


Adding Android support:



# Downloading the Config File:



Adding dependencies to the project:

dependencies:

  flutter:

    sdk: flutter

  # For using cupertino icons

  cupertino\_icons: ^1.0.2

  firebase\_core: ^2.2.0

  firebase\_auth: ^4.1.3

  google\_sign\_in: ^5.4.2

  cloud\_firestore: ^4.1.0

  cached\_network\_image: ^3.2.3

  image\_picker: ^0.8.6

  firebase\_storage: ^11.0.6

  emoji\_picker\_flutter: ^1.5.1

  firebase\_messaging: ^14.2.0

  http: ^0.13.5

  flutter\_notification\_channel: ^2.0.0

**Code:**

**login\_screen.dart**

import 'dart:developer';

import 'dart:io';

import 'package:firebase\_auth/firebase\_auth.dart';

import 'package:flutter/material.dart';

import 'package:google\_sign\_in/google\_sign\_in.dart';

import '../../api/apis.dart';

import '../../helper/dialogs.dart';

import '../../main.dart';

import '../home\_screen.dart';

//login screen -- implements google sign in or sign up feature for app

class LoginScreen extends StatefulWidget {

const LoginScreen({super.key});

@override

State<LoginScreen> createState() => \_LoginScreenState();

}

class \_LoginScreenState extends State<LoginScreen> {

bool \_isAnimate = false;

@override

void initState() {

super.initState();

//for auto triggering animation

Future.delayed(const Duration(milliseconds: 500), () {

setState(() => \_isAnimate = true);

});

}

\_handleGoogleBtnClick() {

//for showing progress bar

Dialogs.showProgressBar(context);

\_signInWithGoogle().then((user) async {

//for hiding progress bar

Navigator.pop(context);

if (user != null) {

log('\nUser: ${user.user}');

log('\nUserAdditionalInfo: ${user.additionalUserInfo}');

if ((await APIs.userExists())) {

Navigator.pushReplacement(

context, MaterialPageRoute(builder: (\_) => const HomeScreen()));

} else {

await APIs.createUser().then((value) {

Navigator.pushReplacement(

context, MaterialPageRoute(builder: (\_) => const HomeScreen()));

});

}

}

});

}

Future<UserCredential?> \_signInWithGoogle() async {

try {

await InternetAddress.lookup('google.com');

// Trigger the authentication flow

final GoogleSignInAccount? googleUser = await GoogleSignIn().signIn();

// Obtain the auth details from the request

final GoogleSignInAuthentication? googleAuth =

await googleUser?.authentication;

// Create a new credential

final credential = GoogleAuthProvider.credential(

accessToken: googleAuth?.accessToken,

idToken: googleAuth?.idToken,

);

// Once signed in, return the UserCredential

return await APIs.auth.signInWithCredential(credential);

} catch (e) {

log('\n\_signInWithGoogle: $e');

Dialogs.showSnackbar(context, 'Something Went Wrong (Check Internet!)');

return null;

}

}

@override

Widget build(BuildContext context) {

//initializing media query (for getting device screen size)

mq = MediaQuery.of(context).size;

return Scaffold(

//app bar

appBar: AppBar(

automaticallyImplyLeading: false,

title: const Text('Welcome to Quick Chats'),

),

//body

body: Stack(children: [

//app logo

AnimatedPositioned(

top: mq.height \* .15,

right: \_isAnimate ? mq.width \* .25 : -mq.width \* .5,

width: mq.width \* .5,

duration: const Duration(seconds: 1),

child: Image.asset('images/icon.png')),

//google login button

Positioned(

bottom: mq.height \* .15,

left: mq.width \* .05,

width: mq.width \* .9,

height: mq.height \* .06,

child: ElevatedButton.icon(

style: ElevatedButton.styleFrom(

backgroundColor: const Color.fromARGB(255, 223, 255, 187),

shape: const StadiumBorder(),

elevation: 1),

onPressed: () {

\_handleGoogleBtnClick();

},

//google icon

icon: Image.asset('images/google.png', height: mq.height \* .03),

//login with google label

label: RichText(

text: const TextSpan(

style: TextStyle(color: Colors.black, fontSize: 16),

children: [

TextSpan(text: 'Login with '),

TextSpan(

text: 'Google',

style: TextStyle(fontWeight: FontWeight.w500)),

]),

))),

]),

);

}

}

**Output:**

