PROGRAMAÇÃO II

AULA 5

Prof. Elton Masaharu Sato

CONVERSA INICIAL

Nesta etapa, veremos o que é e como integrar um aplicativo Flutter com Firebase, uma ferramenta online de armazenamento de dados, além de diversas funções de servidor prontas para desenvolvimento.

Flutter possui uma integração com Firebase chamada de *Flutterfire*, uma biblioteca de funções prontas que facilitam o desenvolvimento de aplicativos Flutter com Firebase.

O Dartpad não possui as ferramentas necessárias para integrar todas as funções do Firebase, portanto para esta e a próxima etapa, será necessário o VSCode instalado para desenvolver os aplicativos.

TEMA 1 – O QUE É FIREBASE E FLUTTERFIRE

Figura 1 – Firebase

Fonte: Firebase, [S.d.].

Firebase é uma plataforma para a criação de aplicativos móveis e *web* que foi adquirida pela Google.

O Firebase tecnicamente começou em 2011 como uma API de integração de funcionalidades de chat para sites e aplicações móveis, e naquela época se chamava *Envolve* e foi criado por James Tamplin e Andrew Lee, porém não foi preciso muito tempo para os desenvolvedores descobrirem que sua ferramenta podia fazer muito mais do que simplesmente passar mensagens de chat. Desenvolvedores usavam o Envolve para passar dados de aplicação como estado de jogos para sincronização, por exemplo, jogos de luta que precisam de dados precisos e rápidos em tempo real.

Em 2012, Firebase foi criado como um projeto separado do Envolve, mas focado em suas funções além de chat. O seu primeiro produto era uma base de dados de sincronização em tempo real, permitindo que diversos aplicativos pudessem estar conectados a uma mesma base independente da plataforma em que o aplicativo se encontrava: Android, web, ou iOS.

Em 2014, Firebase focou em maior utilidade móvel com o Firebase Hosting e Firebase Authentication, sendo esse segundo uma ferramenta extremamente útil para desenvolvedores móveis. Nesse ano, a Google adquiriu a Firebase e a integrou com sua ferramenta de web Divshot.

Em 2016, houve grandes integrações com serviços da Google, a Google Cloud Platform, Google Cloud Messaging, Admob, e Google Ads para monetização de aplicativos para pequenos desenvolvedores. Também nesse mesmo ano, a Google adquiriu outra desenvolvedora para plataformas móveis para integrar com o Firebase.

Em 2017, a Google adquiriu a Fabric para adicionar ferramentas de análises do Twitter para os serviços da Firebase. E nesse ano foi lançado o Cloud Firestore, uma base de dados que seria um sucessor ao Firebase original de 2012.

O FlutterFire é toda a integração do Firebase com o Flutter. A Google tira grande proveito da integração desses dois softwares, e portanto o Flutter possui acesso às seguintes funções:

- *Analytics;*
- *Authetication;*
- *Cloud Firestore;*
- *Cloud Functions;*
- *Cloud Messaging;*
- *Cloud Storage;*
- *Core;*

- *Crashlytics;*
- *Dynamic Links;*
- *In-App Messaging;*
- *Installations;*
- *Performance Monitoring;*
- *Realtime Database;*
- *Remote Config.*

Saiba mais

Para uma lista completa e atualizada, acesse o link a seguir:

FIREBASE. FlutterFire. Disponível em: <https://firebase.flutter.dev/>. Acesso em: 25 jul. 2022.

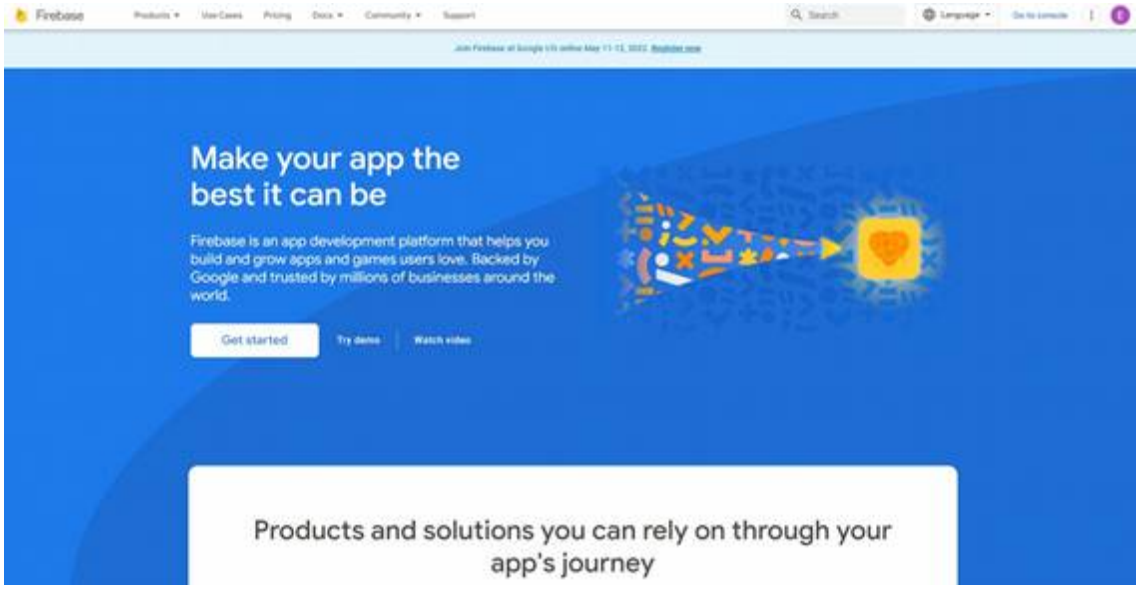
TEMA 2 – CRIANDO UMA FIREBASE

Primeiramente, devemos nos direcionar ao site do Firebase:

Saiba mais

FIREBASE. Disponível em: <https://firebase.google.com/>. Acesso em: 25 jul. 2022.

Figura 1 – *Homepage* do Firebase

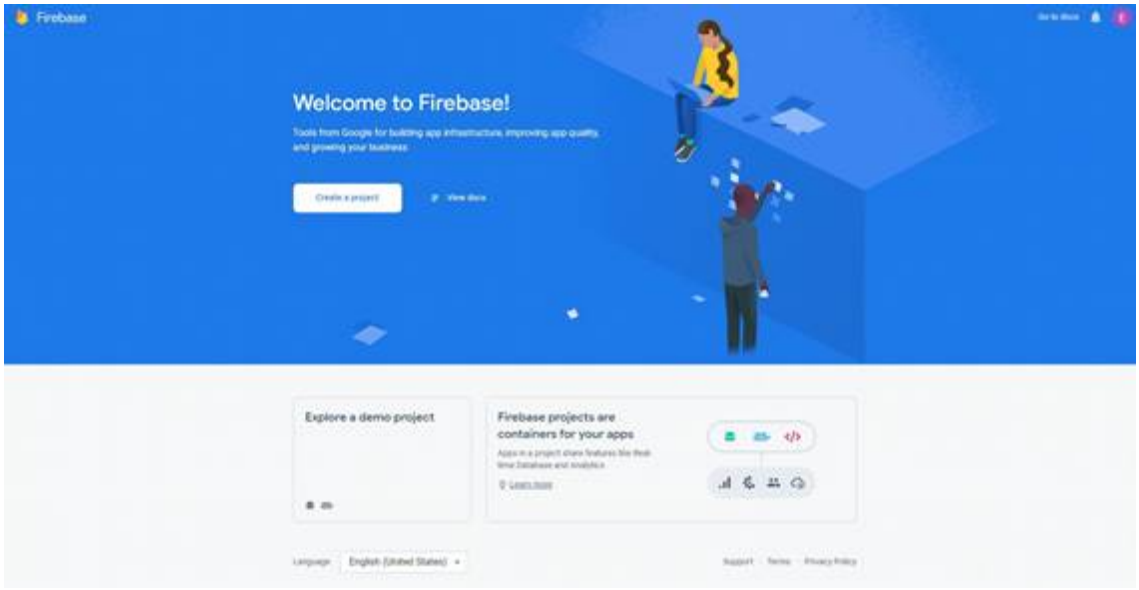


Fonte: Firebase, [S.d.].

No site, como aparece a Figura 1, devemos então fazer um *login* para poder acessar o console. Clicando tanto no canto superior direito, quanto em *Get Started*, começará o processo de criação de uma conta.

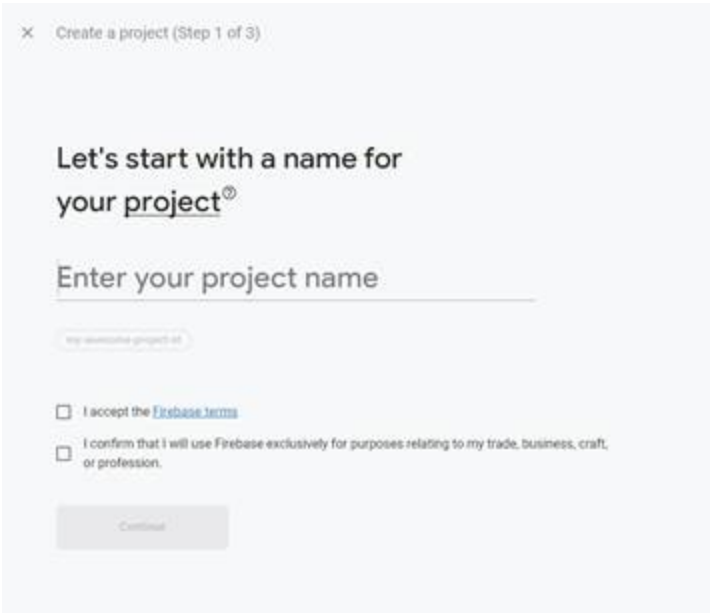
Ao finalizar esse processo, o site o direcionará para a seleção de projetos do Firebase, como mostra a Figura 2.

Figura 2 – Seleção de projetos do Firebase



Fonte: Firebase, [S.d.].

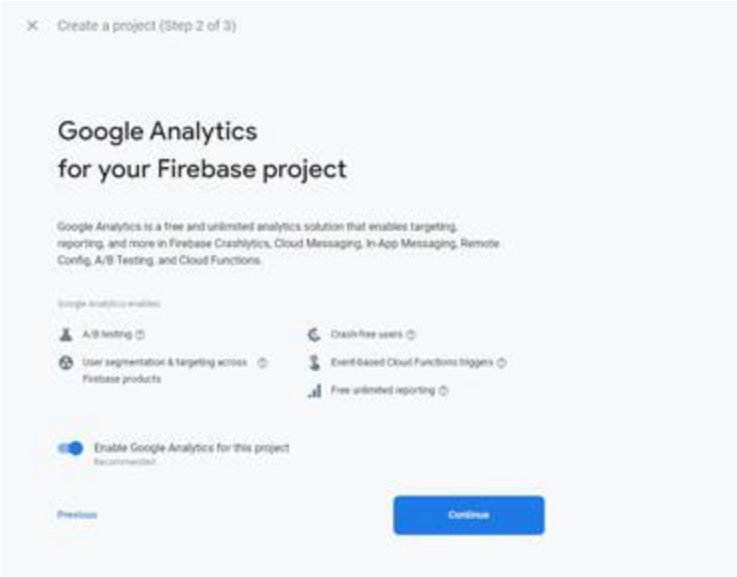
Figura 3 – Primeira etapa na criação do projeto



Fonte: Firebase, [S.d.].

A Figura 3 apresenta a tela onde se colocará o nome do projeto. Devem-se aceitar os termos de uso do Google.

Figura 4 – Segunda etapa na criação do projeto



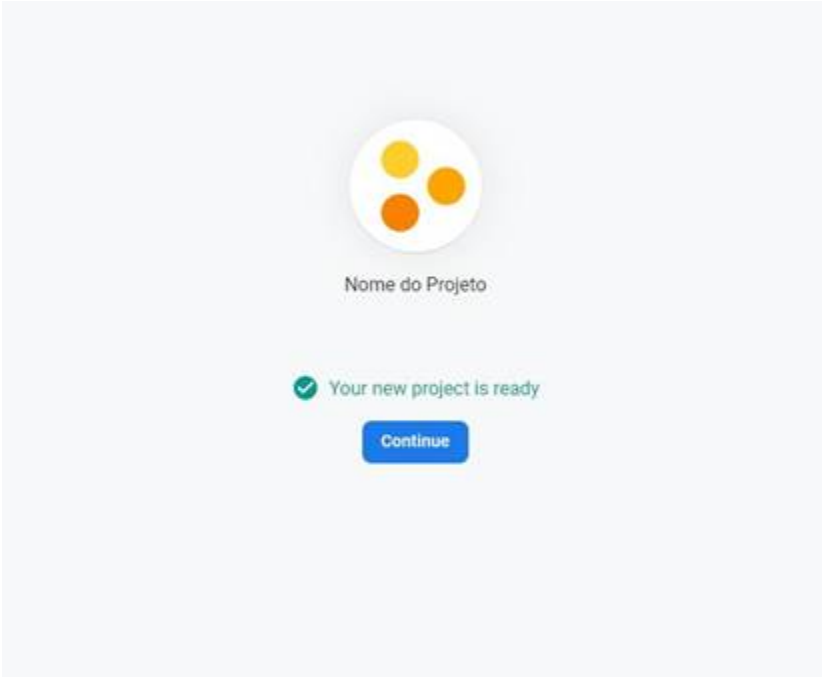
Fonte: Firebase, [S.d.].

Figura 5 – Terceira etapa na criação do projeto

Fonte: Firebase, [S.d.].

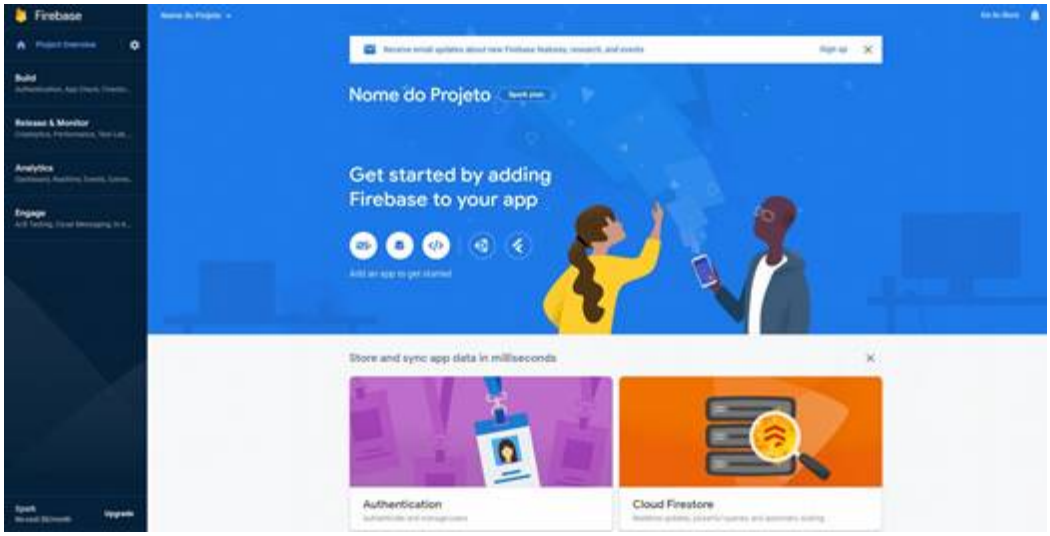
As Figura 4 e 5 apresentam as etapas relacionadas ao Google Analytics, as quais podem ser puladas ao deselecionar a opção *Enable Google Analytics for this Project* (do inglês, habilitar Google Analytics para este projeto) na Figura 4.

Figura 6 – Projeto no Firebase criado com sucesso



Fonte: Firebase, [S.d.].

Figura 7 – Tela principal do Firebase



Fonte: Firebase, [S.d.].

A Figura 6 apresenta a tela de sucesso de criação de um projeto no Firebase, que é então direcionada para tela principal do projeto, assim como apresentada pela Figura 7.

Na tela da Figura 7 podemos ver as opções possíveis:

Build:

- Autenticação;
- *App Check;*
- *Firestore Database;*
- *Realtime Database;*
- Extensões;
- Armazenagem;
- *Hosting;*
- Funções;
- Aprendizado de máquina.

Release & Monitor:

- *Crashlytics;*
- *Performance;*
- Laborátorio de testes;
- Distribuição de App.

Analytics:

- *Dashboard;*
- *Realtime;*
- Eventos;
- Conversões;
- Audiência;
- Definições customizadas;
- Últimos lançamentos;
- Visualização de *Debug*.

Engage:

- *A/B Testing;*
- *Cloud Messaging;*
- *In-App Messaging;*
- Configuração remota;
- *Links* dinâmicos;
- *AdMob*.

TEMA 3 – *SETUP* DO FLUTTERFIRE

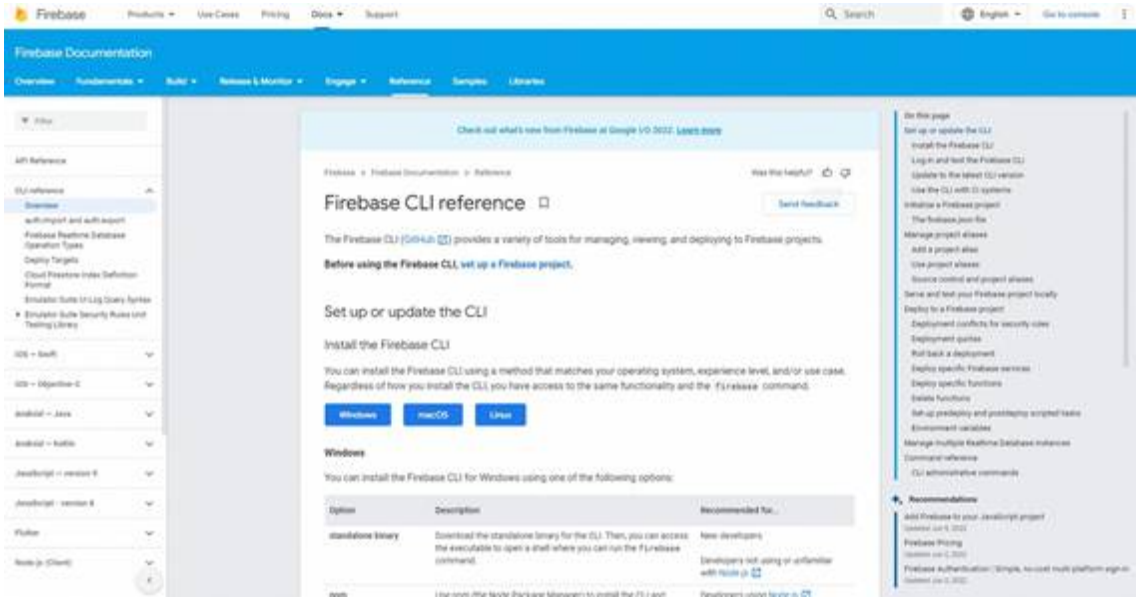
Após criar um projeto no Firebase, devemos configurar o nosso sistema para podermos utilizar as suas funcionalidades e conectar-se a ele.

Saiba mais

O primeiro passo é acessar o site do Firebase e instalar o Firebase CLI(*Command Line Interface*, ou interface de linha de comando em português) pelo link a seguir:

MATERIAL de consulta sobre a CLI do Firebase. Firebase, [S.d.]. Disponível em: <https://firebase.google.com/docs/cli#install-cli-%20windows>. Acesso em: 25 jul. 2022.

Figura 8 – Página com instruções do Firebase CLI



Fonte: Material..., [S.d.].

Na Figura 8, podemos clicar no canto superior direito onde está escrito *English* (inglês) para alterar o idioma para o português brasileiro. Feito isso, fica mais simples para os alunos que não são fluentes na língua.

Saiba mais

Nesta etapa, utilizaremos a instalação via o npm (Gerenciador de pacotes do Node.js, que fará a instalação correta para nós). Devemos então acessar o site do Node.js usando o nvm-windows no *link* a seguir:

NODE. Disponível em: <https://nodejs.org/en/>. Acesso em: 25 jul. 2022.

E selecionaremos a versão da esquerda, a recomendada, assim como aparece na Figura 9. Caso o aluno tenha interesse em saber em qual versão utilizaremos neste estudo, é a versão 16.15.1 LTS.

Figura 9 – Foto da tela do site

Fonte: Nodejs, [S.d.].

Assim que baixado o programa, execute o instalador para então prosseguirmos com o VSCode. Dentro do VSCode, digite “firebase login” para logar na sua conta vinculada ao Firebase e “dart pub global activate flutterfire_cli” para ativar o Flutterfire.

Saiba mais

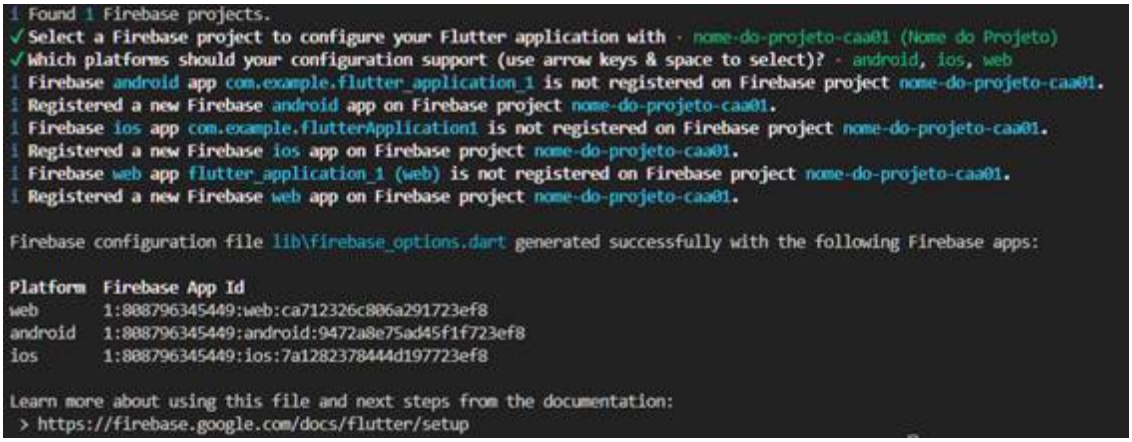
O programa assume que o usuário está tentando instalar na pasta C:/, portanto, lembre-se de mover o instalador para a pasta C:/ antes de executá-lo.

Saiba mais

Caso tenha problemas com o comando “npm”, verifique se o local onde ele foi instalado está na lista de variáveis do ambiente do seu Windows. Digitando variáveis de ambiente no menu inicial do Windows, escolhendo o programa de mesmo nome, clique em variáveis de

ambiente, caminho (*path*), e editar. Na nova janela é possível adicionar novos caminhos, por exemplo, “C:/Arquivos de Programas/nodejs”.

Figura 10 – Foto da tela do VSCode executando o Flutterfire Configure



Fonte: Firebase, [S.d.].

Após executarmos os códigos mostrados, executaremos o código *flutterfire configure*, que pedirá o seguinte:

- Selecione o projeto firebase que deseja configurar;
- Selecionar para quais plataformas o desenvolvedor deseja criar os aplicativos.

A próxima etapa agora é inicializar o Firebase no app, utilizando o comando “flutter pub add firebase”.

Para finalizar, só precisamos agora inicializar essas alterações no código. Importaremos os pacotes do Firebase_core e Firebase_options, além de alterarmos a nossa main para inicializar o Firebase de acordo com a Figura 11.

Figura 11 – Foto da tela do VSCode com as alterações no código para receber o Firebase

```dart
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'firebase_options.dart';

Run | Debug | Profile
Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );
  runApp(const MyApp());
}
```

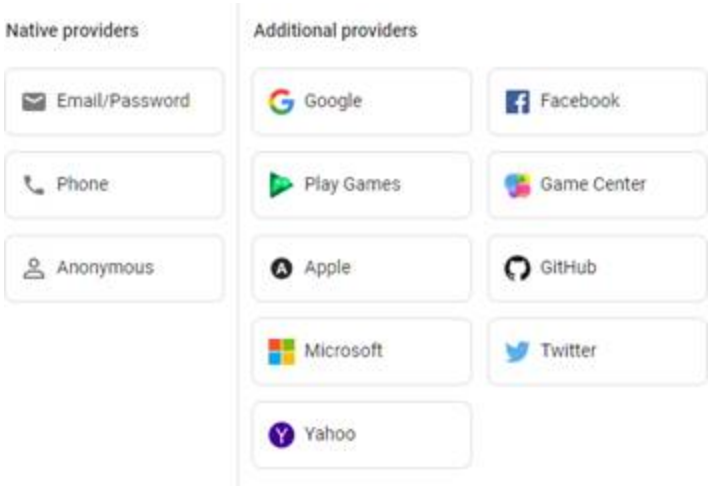
Fonte: Firebase, [S.d.].

TEMA 4 – AUTENTICAÇÃO FIREBASE

Autenticação é provavelmente uma das funções mais utilizadas do Firebase. A maioria dos aplicativos precisam saber da identidade do usuário para poder carregar informações personalizadas para o usuário. Conhecer a identidade do usuário permite um aplicativo salvar dados do usuário na nuvem de forma segura e permitir que o usuário tenha sua experiência personalizada por todos os dispositivos que o usuário possuir.

Autenticação pelo Firebase possui vários serviços integrados, além de ser possível de se autenticar por diversas formas, desde e-mail e número de telefone até identificadores famosos como o do Google, Facebook, e Twitter.

Figura 12 – Tela principal do Firebase



Fonte: Firebase, [S.d.].

A Figura 12 apresenta as opções de autenticação disponíveis no menu de autenticação do Firebase. Neste estudo, utilizaremos o login por e-mail e senha.

Primeiramente, devemos em nosso Vscode, adicionar as dependências que utilizaremos. Para isso, abriremos um novo terminal clicando na aba “Terminal” e depois em “Novo Terminal”/”New Terminal”, o atalho padrão é “Ctrl + Shift + ‘”. E então digitaremos “flutter pub add firebase_auth”, “flutter pub add provider”, e “flutter pub add flutterfire_ui”. O que este código faz é colocar nas dependências do nosso arquivo pubspec.yaml as versões mais recentes do “firebase_auth” e “provider”.

Exemplo:

```
import 'package:flutter/material.dart';
 import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_core/firebase_core.dart';
import 'firebase_options.dart';
import 'package:flutterfire_ui/auth.dart';



Future<void> main() async {

WidgetsFlutterBinding.ensureInitialized();
await Firebase.initializeApp(
options: DefaultFirebaseOptions.currentPlatform,
);
runApp(const MyApp());
}

class MyApp extends StatelessWidget { const
 MyApp({Key? key}) : super(key: key);

 // This widget is the root of your application. @override
Widget build(BuildContext context) { return
 MaterialApp(
 home: AuthGate(),
 );
 }
 }

class AuthGate extends StatelessWidget { const
 AuthGate({Key? key}) : super(key: key);

 @override
Widget build(BuildContext context) { return
 StreamBuilder<User?>(
 stream: FirebaseAuth.instance.authStateChanges(),
 builder: (context, snapshot) {
 // User is not signed in if
 (!snapshot.hasData) { return
 RegisterScreen(
 providerConfigs: [EmailProviderConfiguration()],
 );
 }

    // Render your application if authenticated
    return MyStatefulWidget();
  },
  );
 }
 }
```

TEMA 5 – *REALTIME DATABASE*

Realtime Database é a base de dados original da Firebase. Ela providencia uma solução eficiente e de baixa latência para aplicativos móveis que requerem dados sincronizados entre aplicativos.

A base de dados *Realtime Database* é uma base de dados hospedada em nuvem: os dados são armazenados como um JSON e sincronizados em tempo real em cada cliente conectado. Independentemente se cada aplicativo está rodando em um dispositivo Android, iOS, ou web, todos os clientes estarão conectados a uma única Realtime Database e automaticamente receberão atualizações com os dados mais recentes.

Suas principais características são as seguintes:

5.1 *REALTIME* (TEMPO REAL)

Em vez de requisições HTTP típicas, a *Realtime Database* usa sincronização de dados, ou seja, toda vez que um dado é alterado, todos os dispositivos conectados recebem uma atualização dentro de milisegundos. Isso providencia experiências colaborativas e imersivas sem precisar se preocupar com o código de *networking*.

5.2 OFFLINE

Aplicativos do Firebase continuam responsivos mesmo quando estiverem offline, pois o SDK mantém dados no dispositivo. Quando a conexão voltar, o cliente recebe as atualizações perdidas e se sincroniza com o servidor.

5.3 ACESSÍVEL

O banco de dados do Firebase pode ser acessado diretamente por um aplicativo móvel ou web sem a necessidade de rodar uma aplicação de servidor. Segurança e validação de dados são disponíveis pelas regras de segurança, que são executadas toda vez que dados são lidas ou escritas.

Criando um novo projeto flutter, iremos então abrir o terminal e digitar os seguintes comandos:

- npm install -g firebase-tools

- dart pub global activate flutterfire_cli
- flutter pub add flutterfire_ui
- flutterfire configure (e selecione o firebase desejado, caso não esteja logado, faço o login utilizando “flutterfire login”)
- flutter pub add firebase_database
- flutter pub add firebase_core

EXEMPLO:

```
import 'package:flutter/material.dart';
 import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_core/firebase_core.dart';
import 'firebase_options.dart';
import 'package:flutterfire_ui/auth.dart';



Future<void> main() async {

WidgetsFlutterBinding.ensureInitialized();
await Firebase.initializeApp(
options: DefaultFirebaseOptions.currentPlatform,
);
runApp(const MyApp());
}

class MyApp extends StatelessWidget { const
 MyApp({Key? key}) : super(key: key);

 // This widget is the root of your application. @override
Widget build(BuildContext context) { return
 MaterialApp(
 home: AuthGate(),
 );
 }
 }

class AuthGate extends StatelessWidget { const
 AuthGate({Key? key}) : super(key: key);

 @override
Widget build(BuildContext context) { return
 StreamBuilder<User?>(
 stream: FirebaseAuth.instance.authStateChanges(),
 builder: (context, snapshot) {
 // User is not signed in if
 (!snapshot.hasData) { return
 RegisterScreen(
 providerConfigs: [EmailProviderConfiguration()],
 );
 }

    // Render your application if authenticated
    return MyStatefulWidget();
  },
  );
 }
 }
```

FINALIZANDO

Nesta etapa aprendemos o que é o banco de dados da Firebase, criamos um projeto neste mesmo banco de dados e o configuramos, aprendendo a utilizar algumas de suas funções.

Vimos também a origem do Firebase e como ele cresceu e se tornou uma das maiores ferramentas de desenvolvimento em nuvem para dispositivos móveis.

REFERÊNCIAS

FIREBASE. Disponível em: <https://firebase.google.com/>. Acesso em: 25 jul. 2022.

MATERIAL de consulta sobre a CLI do **Firebase**. Firebase, [S.d.]. Disponível em: <https://firebase.google.com/docs/cli#install-cli-%20windows>. Acesso em: 25 jul. 2022.

NODE. Disponível em: <https://nodejs.org/en/>. Acesso em: 25 jul. 2022.