TramiEPET

Alumno: Castro Tomás

Profesor: Exequiel Wiedermann

ÍNDICE

PROGRAMACIÓN CON DART

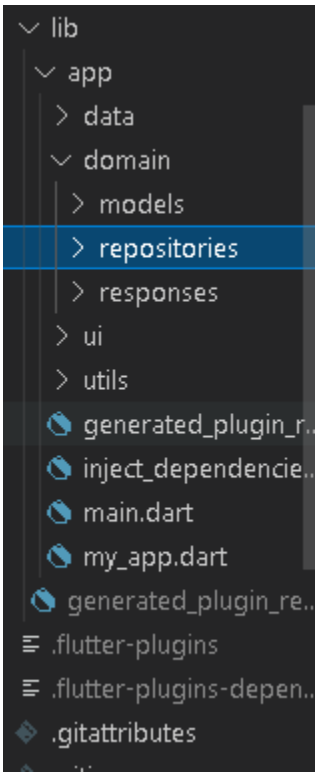
Bueno, todo lo empecé a programar como una aplicación clásica que se hace con flutter, empecé definiendo las carpetas de trabajo dentro del proyecto, utilicé un tipo de arquitectura limpio, dividido en **data, domain, ui y utils.**

Data: Es en donde se van a implementar los repositorios, todo el código de implementación.

Domain: Se definen las estructuras de los repositorios

Ui: Se definen todas las páginas e implementación, es la interfaz de usuario

Utils: Código extra de gran ayuda para mis implementaciones.



Estructuras de los repositorios

Autenticación

lib > app > domain > repositories > authentication_repository.dart > ...

```dart
import 'package:firebase_auth/firebase_auth.dart';
import 'package:tramipet/app/domain/responses/sign_in_response.dart';

abstract class AuthenticationRepository {
  Future<User?> get user;
  Future<void> signOut();
  Future<SignInResponse> signInWithGoogle();
}

```

(Se encunetra el inicio de sesión con google, el logout y el user de firebase auth)

lib > app > domain > repositories > preferences_repository.dart

```dart
abstract class PreferencesRepository {
  bool get isDarkModeAlias;
  Future<void> darkMode(bool enabled);
}

```

(Repositorio de preferencias del modo oscuro)

Implementaciones de los repositorios

```dart
class AuthenticationRepositoryImpl implements AuthenticationRepository {
  final FirebaseAuth _auth;
  final GoogleSignIn _googleSignIn;
  User? _user;

  final Completer<void> _completer = Completer();

  AuthenticationRepositoryImpl({
    required FirebaseAuth firebaseAuth,
    required GoogleSignIn googleSignIn,
  })  : _auth = firebaseAuth,
        _googleSignIn = googleSignIn {
    _init();
  }

  @override
  Future<User?> get user async {
    await _completer.future;
    return _user;
  }

  void _init() async {
    _auth.authStateChanges().listen(
      (User? user) {
        if (!_completer.isCompleted) {
          _completer.complete();
        }
        _user = user;
      },
    );
  }
```

```dart
@override
Future<void> signOut() async {
  final data = _user?.providerData ?? [];
  String providerId = 'firebase';
  for (final provider in data) {
    if (provider.providerId != 'firebase') {
      providerId = provider.providerId;
      break;
    }
  }
  if (providerId == 'google.com') {
    _googleSignIn.signOut();
  }
  return _auth.signOut();
}

@override
Future<SignInResponse> signInWithGoogle() async {
  CollectionReference users = FirebaseFirestore.instance.collection('users');
  try {
    final account = await _googleSignIn.signIn();
    if (account == null) {
      return SignInResponse(
        error: SignInError.cancelled,
        user: null,
        providerId: null,
      );
    }

    final googleAuth = await account.authentication;

    final OAuthCredential oAuthCredential = GoogleAuthProvider.credential(
```

```dart
    final googleAuth = await account.authentication;

    final OAuthCredential oAuthCredential = GoogleAuthProvider.credential(
      idToken: googleAuth.idToken,
      accessToken: googleAuth.accessToken,
    );

    final userCredential = await _auth.signInWithCredential(oAuthCredential);

    final User user = userCredential.user!;

    var userData = {
      'Nombre': account.displayName,
      'provider': 'google',
      'photoUrl': account.photoUrl,
      'email': account.email,
    };

    users.doc(user.uid).get().then((doc) {
      if (doc.exists) {
        doc.reference.update(userData);
      } else {
        // Nuevo usuario :D
        users.doc(user.uid).set(userData);
      }
    });

    return SignInResponse(
        error: null,
        user: userCredential.user,
        providerId: userCredential.credential?.providerId);
  } on FirebaseAuthException catch (e) {
```

```
      return SignInResponse(
          error: null,
          user: userCredential.user,
          providerId: userCredential.credential?.providerId);
    } on FirebaseAuthException catch (e) {
      e.credential?.providerId;
      return getSignInError(e);
    }
  }
}
```

(Implementación del repositorio autenticación en general)

```
const darkModeKey = 'dark-mode';

class PreferencesRepositoryImpl implements PreferencesRepository {
  final SharedPreferences _preferences;

  PreferencesRepositoryImpl(this._preferences);

  @override
  bool get isDarkModeAlias => _preferences.getBool(darkModeKey) ?? false;

  @override
  Future<void> darkMode(bool enabled) {
    return _preferences.setBool(darkModeKey, enabled);
  }
}
```

(Key del modo oscuro y los gets)

UI

La UI se divide en: **global controllers,global widgets,icons,pages,routes e utils.**

Global controllers: Son los controladores que se usan globalmente en la aplicación, como el de sesión activa del usuario y que se guarden todas sus preferencias predeterminadamente y el del modo oscuro, que también el usuario decide en que modo dejarlo, si en oscuro o blanco predeterminadamente.

```
class SessionController extends SimpleNotifier {
  User? _user;
  User? get user => _user;

  final AuthenticationRepository _auth = Get.i.find();

  void setUser(User user) {
    _user = user;
    notify();
  }

  Future<void> signOut() async {
    await _auth.signOut();
    _user = null;
  }
}

final sessionProvider = SimpleProvider(
  (ref) => SessionController(),
  autoDispose: false,
); // SimpleProvider
```

```
const Map<int, Color> swatch = {
  50: Color.fromRGBO(136, 14, 79, .1),
  100: Color.fromRGBO(136, 14, 79, .2),
  200: Color.fromRGBO(136, 14, 79, .3),
  300: Color.fromRGBO(136, 14, 79, .4),
  400: Color.fromRGBO(136, 14, 79, .5),
  500: Color.fromRGBO(136, 14, 79, .6),
  600: Color.fromRGBO(136, 14, 79, .7),
  700: Color.fromRGBO(136, 14, 79, .8),
  800: Color.fromRGBO(136, 14, 79, .9),
  900: Color.fromRGBO(136, 14, 79, 1),
};

const colorPrimarioDark = Color(0xff00bcd4);
const colorPrimarioLight = Color(0xff2962ff);

class ThemeController extends SimpleNotifier {
  late ThemeMode _mode;

  ThemeController() {
    _mode = _preferences.isDarkModeAlias ? ThemeMode.dark : ThemeMode.light;
  }

  final PreferencesRepository _preferences = Get.i.find();

  ThemeMode get mode => _mode;
  bool get isDark => _mode == ThemeMode.dark;

  TextTheme get _textTheme {
    return GoogleFonts.nunitoSansTextTheme();
  }
```

```dart
ThemeData get lightTheme {
  return ThemeData.light().copyWith(
    appBarTheme: const AppBarTheme(
      backgroundColor: colorPrimarioLight,
    ),
    textTheme: _textTheme,
    primaryColorLight: colorPrimarioLight,
    colorScheme: ColorScheme.fromSwatch(
        brightness: Brightness.light,
        primarySwatch: MaterialColor(colorPrimarioLight.value, swatch)), // ColorScheme.fromSwatch
    inputDecorationTheme: InputDecorationTheme(
      focusedBorder: OutlineInputBorder(
        borderSide: BorderSide(
          color: colorPrimarioLight.withOpacity(0.5),
        ), // BorderSide
      ), // OutlineInputBorder
      enabledBorder: const OutlineInputBorder(
        borderSide: BorderSide(
          color: Colors.black12,
        ), // BorderSide
      ), // OutlineInputBorder
    ), // InputDecorationTheme
  );
}

ThemeData get darkTheme {
  return ThemeData.dark().copyWith(
    appBarTheme: const AppBarTheme(
      backgroundColor: colorPrimarioDark,
    ),
    textTheme: _textTheme.merge(ThemeData.dark().textTheme).apply(
```

```dart
ThemeData get darkTheme {
  return ThemeData.dark().copyWith(
    appBarTheme: const AppBarTheme(
      backgroundColor: colorPrimarioDark,
    ),
    textTheme: _textTheme.merge(ThemeData.dark().textTheme).apply(
          fontFamily: _textTheme.bodyText1!.fontFamily,
        ),
    scaffoldBackgroundColor: const Color(0xff1b1b1b),
    primaryColorDark: colorPrimarioDark,
    textSelectionTheme: const TextSelectionThemeData(
      cursorColor: colorPrimarioDark,
    ),
    colorScheme: ColorScheme.fromSwatch(
        brightness: Brightness.dark,
        primarySwatch: MaterialColor(colorPrimarioDark.value, swatch)), // ColorScheme.fromSwatch
    inputDecorationTheme: const InputDecorationTheme(
      focusedBorder: OutlineInputBorder(
        borderSide: BorderSide(
          color: colorPrimarioDark,
        ), // BorderSide
      ), // OutlineInputBorder
      enabledBorder: OutlineInputBorder(
        borderSide: BorderSide(
          color: Colors.white54,
        ), // BorderSide
      ), // OutlineInputBorder
    ), // InputDecorationTheme
  );
}
```

```
  void toggle() {
    if (_mode == ThemeMode.light) {
      _mode = ThemeMode.dark;
      _preferences.darkMode(true);
      SystemChrome.setSystemUIOverlayStyle(SystemUiOverlayStyle.light);
    } else {
      _mode = ThemeMode.light;
      _preferences.darkMode(false);
      SystemChrome.setSystemUIOverlayStyle(SystemUiOverlayStyle.dark);
    }
    notify();
  }
}

final themeProvider =
    SimpleProvider((ref) => ThemeController(), autoDispose: false);
```

Global Widgets: Se encuentran los widgets globales, como el de alerta de usuario, cuando existen campos mal, de validación y donde también van clases reutilizables de código, POO.

Icons: Esto lo hice con una página donde te genera un icono de flutter, en este caso yo lo elegí para el botón de Google.

```
import 'package:flutter/widgets.dart';

class GoogleIcons {
  GoogleIcons._();

  static const _kFontFam = 'GoogleIcons';
  static const String? _kFontPkg = null;

  static const IconData icons8_logo_de_google =
      IconData(0xe800, fontFamily: _kFontFam, fontPackage: _kFontPkg);
}
```

Pages: Toda la interfaz del usuario y UX va acá, en las siguientes fotos mostraré el código, se divide en: **home y login.**

```dart
final homeProvider = SimpleProvider(
  (_) => HomeController(),
); // SimpleProvider

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

  @override
  Widget build(BuildContext context) {
    return ProviderListener<HomeController>(
      provider: homeProvider,
      builder: (_, controller) {
        return Scaffold(
          bottomNavigationBar: HomeTabBar(),
          appBar: AppBar(
            title: const Center(child: Text('TramiEPET')),
          ), // AppBar
          body: SafeArea(
            child: TabBarView(
              controller: controller.tabController,
              children: const [
                HomeTab(),
                ProfileTab(),
              ],
            ), // TabBarView
          ), // SafeArea
        ); // Scaffold
      },
```

Home Page

```dart
class HomeTab extends StatefulWidget {
  const HomeTab({Key? key}) : super(key: key);

  @override
  State<HomeTab> createState() => _HomeTabState();
}

class _HomeTabState extends State<HomeTab> {
  TextEditingController nombreSolicitudesForm = TextEditingController();
  TextEditingController apellidoSolicitudesForm = TextEditingController();
  TextEditingController cursoSolicitudesForm = TextEditingController();
  TextEditingController dniSolicitudesForm = TextEditingController();
  TextEditingController materiaSolicitudesForm = TextEditingController();

  final firebase = FirebaseFirestore.instance;

  enviarRendirMaterias() async {
    try {
      await firebase.collection("SolicitudesMaterias").doc().set({
        "nombre": nombreSolicitudesForm.text,
        "apellido": apellidoSolicitudesForm.text,
        "curso": cursoSolicitudesForm.text,
        "dni": dniSolicitudesForm.text,
        "materia": materiaSolicitudesForm.text,
      });
    } catch (e) {
      print(e);
```

```dart
final _formCertKey = GlobalKey<FormState>();

@override
Widget build(BuildContext context) {
  return SizedBox(
    child: Column(
      children: [
        const SizedBox(height: 20),
        const Text(
          "¡Bienvenido a TramiEPET!",
          style: TextStyle(fontSize: 20),
        ), // Text
        Image.asset('assets/tramipet.png', height: 330, width: 330),
        const Text(
            'Cualquier duda comunicarse con los directivos del colegio',
            style: TextStyle(fontSize: 13)), // Text
        const SizedBox(height: 70),
        LabelButton(
          label: 'Solicitud para rendir materias',
          value: '',
          onPressed: () {
            showDialog(
                context: context,
                builder: (context) => AlertDialog(
                      title: const Text('Rendir materias'),
                      content: Form(
                        key: _formCertKey,
                        child: SingleChildScrollView(
                          child: Column(
                            children: [
```

```
                    TextFormField(
                      controller: nombreSolicitudesForm,
                      onChanged: (value) {},
                      decoration: const InputDecoration(
                        hintText: "Ingrese su nombre ",
                        icon: Icon(
                          Icons.accessibility,
                          color: Colors.blue,
                        ), // Icon
                      ), // InputDecoration
                    ), // TextFormField
                    const SizedBox(height: 10),
                    TextFormField(
                      controller: apellidoSolicitudesForm,
                      onChanged: (value) {},
                      decoration: const InputDecoration(
                        hintText: "Ingrese su apellido",
                        icon: Icon(
                          Icons.accessibility,
                          color: Colors.blue,
                        ), // Icon
                      ), // InputDecoration
                    ), // TextFormField
                    const SizedBox(height: 10),
                    TextFormField(
                      controller: cursoSolicitudesForm,
                      onChanged: (value) {},
                      decoration: const InputDecoration(
                        hintText: "Ingrese su curso",
                        icon: Icon(
                          Icons.border_color,
                          color: Colors.blue,
                      ), // InputDecoration
                    ), // TextFormField
                    const SizedBox(height: 10),
                    TextButton(
                      onPressed: () {
                        salir();
                        nombreSolicitudesForm.clear();
                        apellidoSolicitudesForm.clear();
                        cursoSolicitudesForm.clear();
                        dniSolicitudesForm.clear();
                        materiaSolicitudesForm.clear();
                        router.pushReplacementNamed(Routes.HOME);
                      },
                      child: const Text('Salir'),
                    ), // TextButton
                    TextButton(
                      onPressed: () {
                        enviarRendirMaterias();
                        nombreSolicitudesForm.clear();
                        apellidoSolicitudesForm.clear();
                        cursoSolicitudesForm.clear();
                        dniSolicitudesForm.clear();
                        materiaSolicitudesForm.clear();
                        router.pushReplacementNamed(Routes.HOME);
                      },
                      child: const Text('Enviar'),
                    ), // TextButton
                  ],
                ), // Column
              ), // SingleChildScrollView
            ), // Form
          )); // AlertDialog
```

```dart
class LabelButton extends StatelessWidget {
  final String label, value;
  final VoidCallback? onPressed;
  const LabelButton(
      {Key? key, required this.label, required this.value, this.onPressed})
      : super(key: key);

  @override
  Widget build(BuildContext context) => ListTile(
        onTap: onPressed,
        contentPadding: const EdgeInsets.symmetric(vertical: 0, horizontal: 20),
        leading: Text(
          label,
          style: const TextStyle(fontWeight: FontWeight.w500, fontSize: 18),
        ), // Text
        trailing: Row(
          mainAxisSize: MainAxisSize.min,
          children: [
            Text(value, style: const TextStyle(fontWeight: FontWeight.w300)),
            const SizedBox(width: 5),
            const Icon(
              Icons.chevron_right_rounded,
              size: 22,
            ) // Icon
          ],
        ), // Row
      ); // ListTile
}
```

Home Tab

```dart
class HomeController extends SimpleNotifier {
  late TabController tabController;
  HomeController() {
    tabController = TabController(
      length: 2,
      vsync: NavigatorState(),
    ); // TabController
  }

  @override
  void dispose() {
    tabController.dispose();
    super.dispose();
  }
}
```

Controller de homeTab (tiene que ver con el gestor de estados) event y state

```dart
import 'package:tramipet/app/ui/routes/routes.dart';
import 'package:flutter_meedu/router.dart' as router;
import '../../../../utils/dark_mode_extension.dart';

class ProfileTab extends ConsumerWidget {
  const ProfileTab({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context, ref) {
    final sessionController = ref.watch(sessionProvider);
    final isDark = context.isDarkModeAlias;
    final user = sessionController.user!;

    final displayName = user.displayName ?? '';
    final letra = displayName.isNotEmpty ? displayName[0] : "";

    return ListView(
      children: [
        const SizedBox(height: 20),
        CircleAvatar(
          radius: 75,
          child: Text(
            letra,
            style: const TextStyle(fontSize: 65),
          ), // Text
          //  backgroundImage:
          //    user.photoURL != null ? NetworkImage(user.photoURL!) : null,      - Configuro cuando sepa manejar mejo
        ), // CircleAvatar
        const SizedBox(height: 10),
        Center(
            child: Text(
          displayName,
          style: const TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
        )), // Text // Center
        Center(child: Text(user.email ?? '')),
        const SizedBox(height: 50),
        const Center(
            child: Text("Información del usuario",
                style: TextStyle(fontWeight: FontWeight.w500))), // Text // Center
        const SizedBox(height: 20),
        LabelButton(label: "Nombre", value: displayName),
        LabelButton(label: "Email", value: user.email ?? ''),
        const LabelButton(label: "Solicitudes pedidas", value: ''),
        Padding(
          padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 30),
          child: Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: [
              const Text("Modo oscuro"),
              CupertinoSwitch(
                value: isDark,
                activeColor: isDark ? Colors.cyan : Colors.blue,
                onChanged: (_) {
                  themeProvider.read.toggle();
                },
              ), // CupertinoSwitch
            ],
          ), // Row
        ), // Padding
        const SizedBox(height: 50),
        LabelButton(
          label: "Salir",
          value: "",
          onPressed: () async {
```

```dart
    ), // Padding
    const SizedBox(height: 50),
    LabelButton(
      label: "Salir",
      value: "",
      onPressed: () async {
        ProgressDialog.show(context);
        await sessionProvider.read.signOut();
        router.pushNamedAndRemoveUntil(Routes.LOGIN);
      },
    ), // LabelButton
  ],
); // ListView
```


Pagina de login

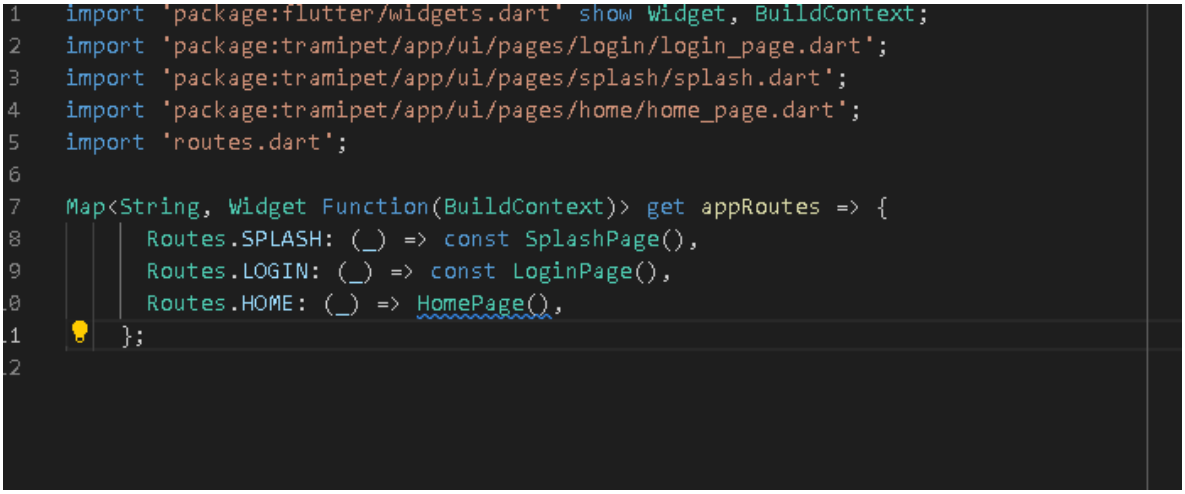
```dart
Widget build(BuildContext context) {
  return ProviderListener<LoginController>(
    provider: loginProvider,
    builder: (_, controller) {
      return Scaffold(
        body: SafeArea(
          child: GestureDetector(
            onTap: () => FocusScope.of(context).unfocus(),
            child: Contain
              color: Color
              width: doubl
              padding: con
              child: Form(
                key: contr
                child: Sin
                  child: C
                    const
                    SvgPic
                      'ass
                      widt
                    ), //
                    const Text("Agiliza tus trámites escolares"),
                    const SizedBox(height: 45),
                    const Text("Inicia sesión con",
                        style: TextStyle(fontSize: 35.0)), // Text
                    const SizedBox(height: 10),
                    const SocialButtons(),
                  ]), // Column
                ), // SingleChildScrollView
              ), // Form
```

```dart
Text Text(
  String data, {
  Key? key,
  TextStyle? style,
  StrutStyle? strutStyle,
  TextAlign? textAlign,
  TextDirection? textDirection,
  Locale? locale,
  bool? softWrap,
  TextOverflow? overflow,
  double? textScaleFactor,
  int? maxLines,
  String? semanticsLabel,
```

Controller del login

```dart
import 'package:flutter/widgets.dart' show FormState, GlobalKey;
import 'package:flutter_meedu/flutter_meedu.dart';
import 'package:tramipet/app/domain/repositories/authentication_repository.dart';
import 'package:tramipet/app/domain/responses/sign_in_response.dart';
import 'package:tramipet/app/ui/global_controllers/session_controller.dart';

class LoginController extends SimpleNotifier {
  final SessionController _sessionController;

  final _authenticationRepository = Get.i.find<AuthenticationRepository>();
  final GlobalKey<FormState> formKey = GlobalKey();

  LoginController(this._sessionController);

  Future<SignInResponse> signInWithGoogle() async {
    final response = await _authenticationRepository.signInWithGoogle();
    if (response.error == null) {
      _sessionController.setUser(response.user!);
    }
    return response;
  }
}
```

Routes: Acá se encuentran todas las rutas que voy a utilizar en mi aplicación, esto viene de una dependencia, que se llama flutter_meedu, que ya la voy a explicar mas adelante.



```dart
import 'package:flutter/widgets.dart' show Widget, BuildContext;
import 'package:tramipet/app/ui/pages/login/login_page.dart';
import 'package:tramipet/app/ui/pages/splash/splash.dart';
import 'package:tramipet/app/ui/pages/home/home_page.dart';
import 'routes.dart';

Map<String, Widget Function(BuildContext)> get appRoutes => {
      Routes.SPLASH: (_) => const SplashPage(),
      Routes.LOGIN: (_) => const LoginPage(),
      Routes.HOME: (_) => HomePage(),
    };
```

```dart
abstract class Routes {
  static const SPLASH = '/splash';
  static const LOGIN = '/login';
  static const REGISTER = '/register';
  static const HOME = '/home';
}
```

CÓDIGO GENERAL

Acá se encuentra todo el código de ejecución de mi app y donde se llaman a todas las dependencias, donde también inyecto todos los repositorios que implementé en mi proyecto.

Inyección de repositorios usados: Acá es donde inyecté todos los repositorios, para que estén vinculados a la app, desde Firebase hasta Preferences

```dart
Future<void> injectDependencies() async {
  final preferences = await SharedPreferences.getInstance();
  Get.i.lazyPut<AuthenticationRepository>(
    () => AuthenticationRepositoryImpl(
      firebaseAuth: FirebaseAuth.instance,
      googleSignIn: GoogleSignIn(),
    ), // AuthenticationRepositoryImpl
  );

  Get.i.lazyPut<AccountRepository>(
    () => AccountRepositoryImpl(
      FirebaseAuth.instance,
    ),
  );

  Get.i.lazyPut<PreferencesRepository>(
    () => PreferencesRepositoryImpl(preferences),
  );
}
```

```dart
Run | Debug | Profile
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  await injectDependencies();
  runApp(
    MyApp(),
  );
}
```

```dart
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Consumer(builder: (_, ref, __) {
      final theme = ref.watch(themeProvider);
      return MaterialApp(
        key: router.appKey,
        title: 'Tramipet Componentes',
        navigatorKey: router.navigatorKey,
        debugShowCheckedModeBanner: false,
        navigatorObservers: [
          router.observer,
        ],
        initialRoute: Routes.LOGIN,
        darkTheme: theme.darkTheme,
        theme: theme.lightTheme,
        themeMode: theme.mode,
        routes: appRoutes,
      ); // MaterialApp
    }); // Consumer
  }
}
```

Dependencias

```yaml
environment:
  sdk: ">=2.13.0 <3.0.0"
  flutter: ">=2.0.0"

dependencies:
  flutter:
    sdk: flutter
  equatable: ^2.0.3
  firebase_core: ^1.7.0
  cloud_firestore: ^2.5.4
  firebase_auth: ^3.1.3
  flutter_meedu: "^5.1.0"
  google_fonts: ^2.1.0
  flutter_svg: ^0.23.0+1
  shared_preferences: ^2.0.8
  google_sign_in: ^5.2.0

  flutter_localizations:
    sdk: flutter

dependency_overrides:
  intl: ^0.17.0-nullsafety.2

  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.3

dev_dependencies:
  flutter_test:
    sdk: flutter
```

Equatable: nos permite comprar dos objetos y nos sirve con un gestor de estado

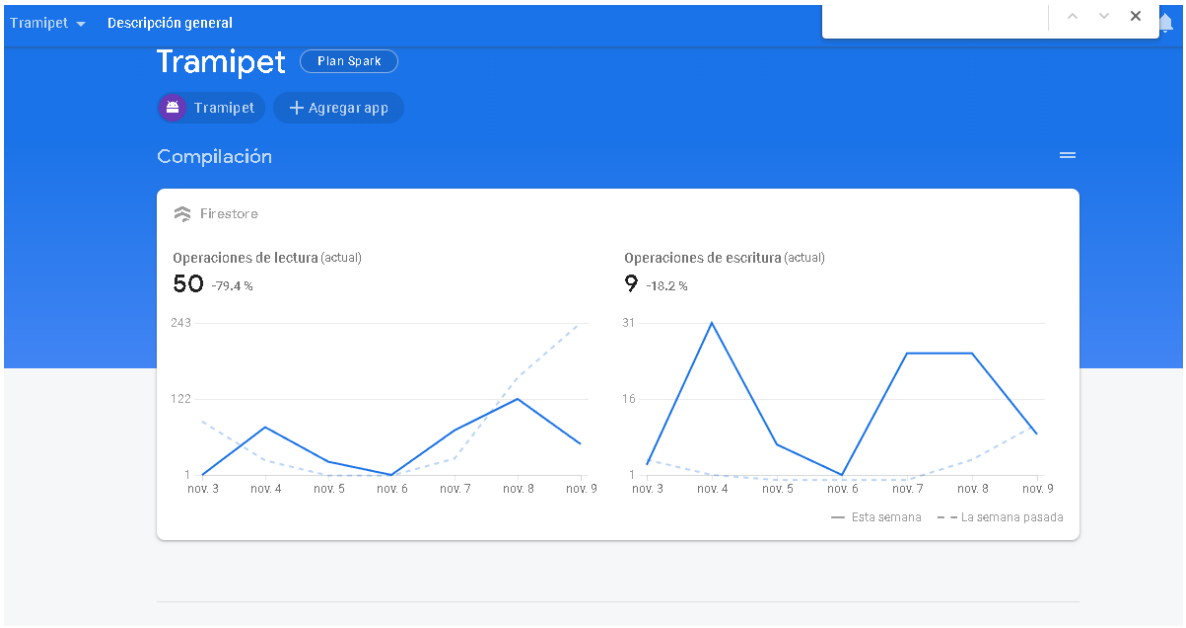
Flutter meedu: Es un gestor de estado que tiene lo mejor de bloc,provider y getX, está hecho por un programador de habla hispana, donde también se encuentra la inyección de dependencias y el manejo de rutas.

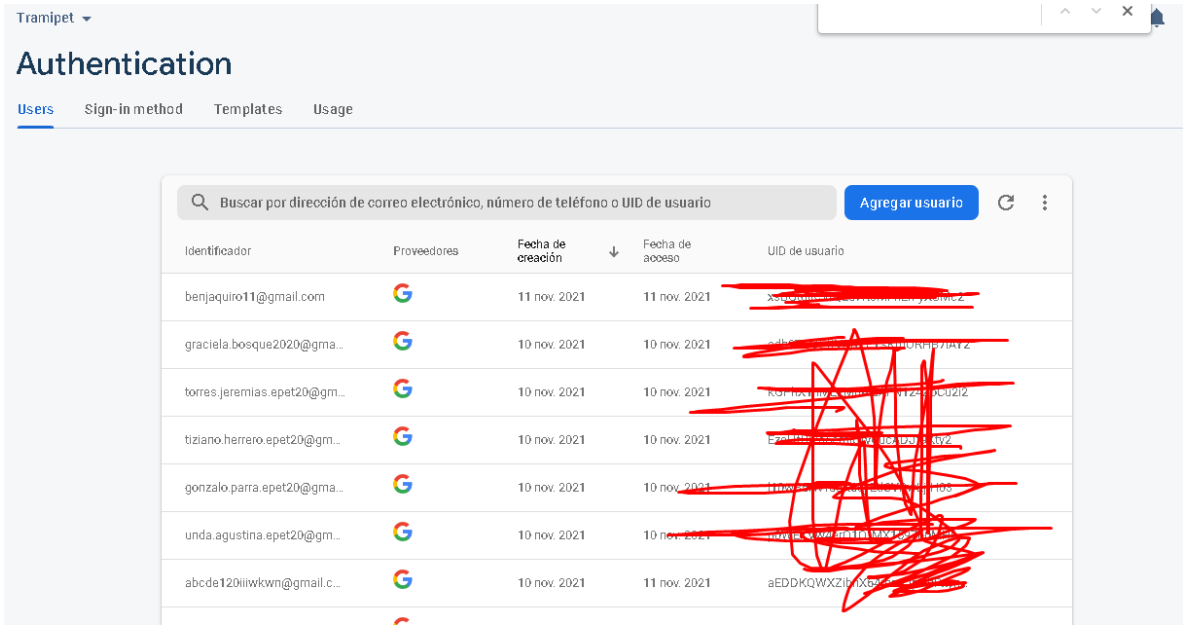
Flutter SVG: es un tipo de gráficos vectoriales, lo usé por su flexibilidad a la hora de ajustar las imágenes
Shared preferences: preferencias del usuario, lo que se queda en el celu.

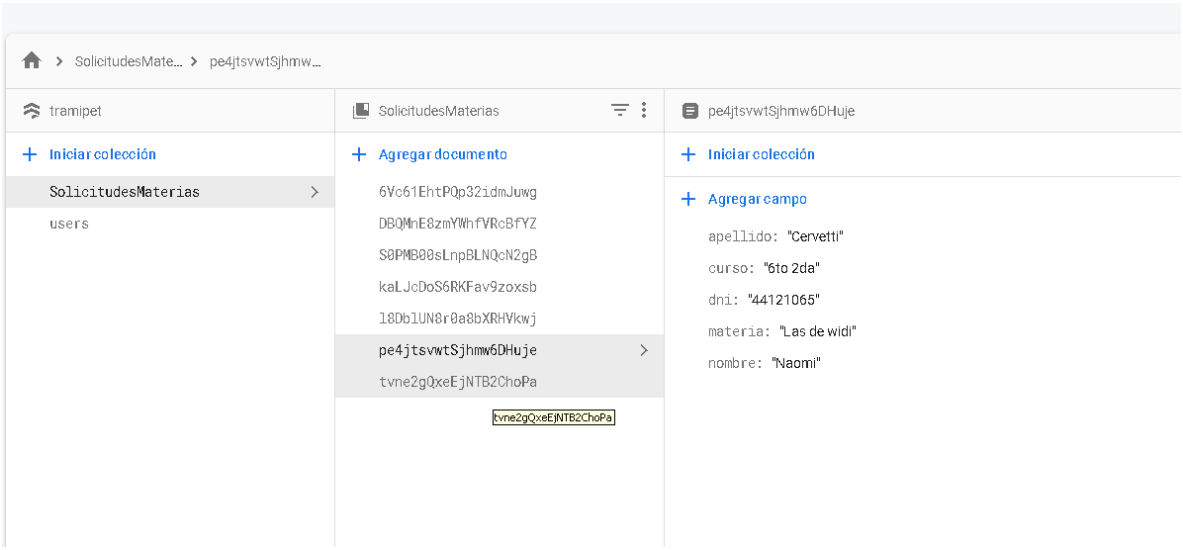
Use flutter 2 y sdk 2.13

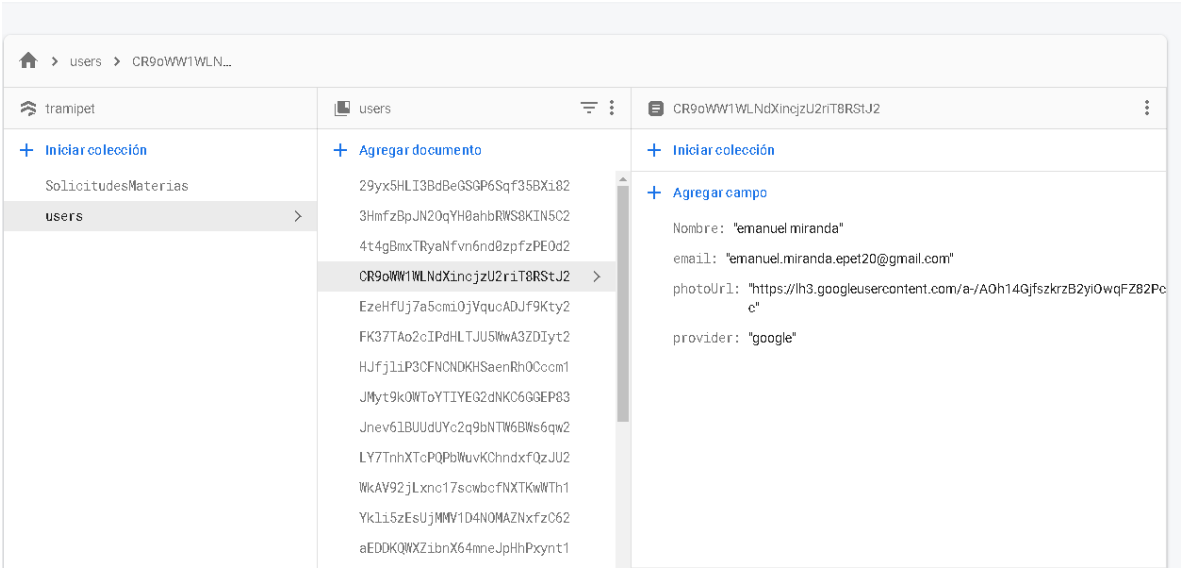
BASE DE DATOS

Usé una base de datos No SQL, esta es Firebase, cuenta con el inicio de sesión con google y el signInWithEmailAndPassword, que es para la aplicación de secretarios, y obviamente, también utiliza firestore, cloud.





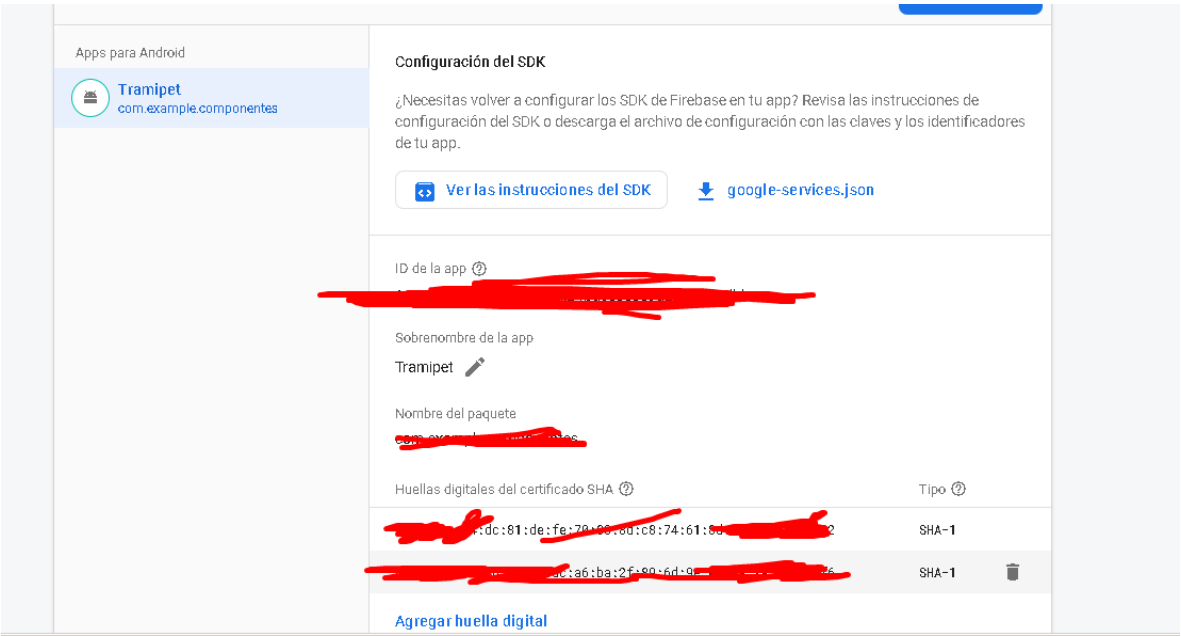




Cloud:

Colecciones: SolicitudesMaterias > vinculado con el formulario alertDialog de la homePage

Y users, que es de googleSignIn y signInwIthEmailAndPassword



Las clave SHA1 son importantes para la implementación de googleSignIn