class **MyLoginServices** with ConverUser implements **BaseAuth** {

var **firebase** = FirebaseAuth.instance; // UserModel? userModel;

//? \*\*\*\*\*\*\*\*\*\*\*< CREATE >\*\*\*\*\*\*\*\*\*\*\*

Future<UserModel?> **creatAnonimUser**(

{required String pasword, required String mail}) async {

try {

var olusanKisi = await firebase.createUserWithEmailAndPassword

(

email: mail, password: pasword);

userModel = **userConvert**(olusanKisi);

**return** userModel;

} on FirebaseAuthException catch (k)

**return** null; }

catch (e) { return null } }

Future<String?> **userKontrolVeKayit**(

{required String pasword, required String mail}) async {

try {

var olusanKisi = await firebase.createUserWithEmailAndPassword(

email: mail, password: pasword);

userModel = userConvert(olusanKisi);

**return** null;}

on FirebaseAuthException catch (k) { return null}

catch (e) { **return** null } }

////////////////////////////////////////////////////////

Future<void> **signOuth**() { return firebase.signOut(); }

//? \*\*\*\*\*\*\*\*\*\*\*< CHANCE >\*\*\*\*\*\*\*\*\*\*\*

Stream<UserModel> get **screeChance** => firebase

.authStateChanges() .map((event) => UserConvert().userConvert(event!));}

**class UserConvert** {

UserModel userConvert(User user) {

**return** **UserModel**(id: user.uid, mail: user.uid); }}

class **AuthWidgetBuilder** extends StatelessWidget {

const **AuthWidgetBuilder**({super.key, required this.onPageBuilder});

final Widget Function(BuildContext context, AsyncSnapshot<UserModel?> snapshot) **onPageBuilder**;

@override

Widget build(BuildContext context) {

//? SINIFLARIMIZA ULAŞACAĞIMIZ BAĞLANTI

final **authServices** = Provider.of<**MyLoginServices**>(context, listen: false);

return StreamBuilder<UserModel?>(

stream: authServices.screeChance,

builder: (context, AsyncSnapshot<UserModel?> snapshot) {

final **userData** = snapshot.data;

if (userData != null) {

return **MultiProvider**(

providers: [Provider.value(value: **userData**)],

child: onPageBuilder(context, snapshot),

); } else {

return **onPageBuilder**(context, snapshot); }

class **AuthWidget** extends StatelessWidget {

const AuthWidget({super.key, required this.snapshot});

final AsyncSnapshot<**UserModel**?> snapshot;

@override

Widget build(BuildContext context) {

if (snapshot.connectionState == ConnectionState.active) {

return snapshot.hasData

? HomeScreen(context: context)

: LoginScreen(context: context);

} else { **return**  const ErrorPage(); }}}

**class** ErrorPage extends StatelessWidget {

const ErrorPage({super.key});

Widget build(BuildContext context) {

return const **Scaffold**(

body: Center(

child: Text('HATA'),

),); }

**MAİN KLASÖRÜ**

Future<void> main() async {

WidgetsFlutterBinding.ensureInitialized();

**await Firebase.initializeApp();**

runApp(const MyApp());

}

class MyApp extends StatelessWidget {

const MyApp({super.key});

@override

Widget build(BuildContext context) {

return **MultiProvider**(

providers: [

ChangeNotifierProvider(create: (context) => **BtnTiklama**()),

Provider<MyLoginServices>(create: (context) => **MyLoginServices**()),

],

child: **AuthWidgetBuilder**(

onPageBuilder: (context, snapshot) => **MaterialApp**(

debugShowCheckedModeBanner: false,

home: **AuthWidget**(snapshot: snapshot),)));}}

**BUTONA TIKLAMA**

Future<void> **kisiKontrolVeolustur**() async {

String? errorMessage =

await Provider.of<**MyLoginServices**>(context, listen: false)

.**userKontrolVeKayit**( pasword: sifreController.text, mail: mailController.text);

setState(() {**errorMessage** = errorMessage;}) }

///////////////////

**butonKaydolTiklama**() async {

Provider.of<BtnTiklama>(context, listen: false).tiklamaState2();

*// ? TEXTFİLD KONTROLÜ*

if (sifreController.text == sifreTekrariController.text &&

sifreController.text.isNotEmpty &&

mailController.text.isNotEmpty)

{

*//! FİREBASE KONTROLÜ => O L U Ş T U R M A*

**await kisiKontrolVeolustur();**

*// KİŞİ OLUŞTURULUSA YANİ VERİ TABANINDA YOKSA*

*// ERROR MESAJI null GELECEK.*

*// VE NULL GELİRSE BAŞARILI GİRİŞ SNACBARI GELECCEK*

if (errorMessage == null) {

**snacBarSucces(context);**

} else {

**snacBarError(context);** // mail veri tabanında var uyarısı

Provider.of<BtnTiklama>(context, listen: false).tiklamaState2();

}

} else {

**snacBar(context);**

Provider.of<BtnTiklama>(context, listen: false).tiklamaState2();