

Josue Garza
Prueba practica
12/12/2022

Prueba Practica RedOm8 Flutter

Realizada por Josue Israel Esquivel Garza

12/12/2022

Contents

Prueba Practica RedOm8 Flutter	1
Aplicación móvil: DevJob	2
Disclaimer:	2
Prerrequisitos:.....	2
Main:	3
Login:.....	5
Sign Up:	7
Forget Password	9
Muestreo de productos:	10
Subida de productos:	13
Repositorio Github.....	15
Retos del proyecto, investigación y soluciones	15
Conexión con firebase y corrección de errores en archivos build.grandle:.....	15
Manejo de usuarios en flutter	15
Crud de objetos.....	15
Retos por cumplir:.....	16

Josue Garza
Prueba practica
12/12/2022

Aplicación móvil: DevJob

Esta aplicación fue creada con las siguientes tecnologías:

Flutter (versión 3.3.0), Dart, Firebase.

Se utilizaron las siguientes dependencias

```
cupertino_icons: ^1.0.2
cached_network_image: ^3.2.2
image_picker: ^0.8.6
image_cropper: ^3.0.1
fluttericon: ^2.0.0
url_launcher: ^6.1.6
curved_navigation_bar: ^1.0.3
uuid: ^3.0.6
flutterstoast: ^8.1.1
firebase_auth: ^4.1.1
firebase_core: ^2.1.1
cloud_firestore: ^4.0.4
firebase_storage: ^11.0.4
```

La aplicación se desarrolló en Android Studio, asegúrese de instalar todo lo necesario para su correcto funcionamiento.

En este manual de usuario se dará por obviado que se saben utilizar las tecnologías anteriores, por lo tanto, si desconoces alguna de estas tecnologías te recomiendo aprender primero y después venir a este manual.

Disclaimer:

Esta aplicación fue realizada con ayuda de tutoriales y sitios web que brindan apoyo a los programadores en situaciones específicas (Udemy y stackoverflow), se realizó un trabajo de investigación ante problemas específicos y estos sitios fueron de apoyo para el proyecto.

Prerrequisitos:

- Celular Android con una versión superior a 7.0
- Conexión a internet.

Josue Garza
Prueba practica
12/12/2022

Main:

Main.dart

Acá es donde se controla la app, iniciamos la app según el estado de la app te enviara a la pantalla principal o la pantalla de error.

```
class MyApp extends StatelessWidget {  
  final Future<FirebaseApp> _initialization = Firebase.initializeApp();  
  
  // This widget is the root of your application.  
  @override  
  Widget build(BuildContext context) {  
    return FutureBuilder(  
      future: _initialization,  
      builder: (context, snapshot) {  
        if (snapshot.connectionState == ConnectionState.waiting) {  
          return const MaterialApp(  
            home: Scaffold(  
              body: Center(  
                child: Text(  
                  "App its being initialized",  
                  style: TextStyle(  
                    color: Colors.cyan,  
                    fontSize: 40,  
                    fontWeight: FontWeight.bold,  
                    fontFamily: 'Signatra'  
                  ),  
                ),  
              ),  
            ),  
          );  
        } else if (snapshot.hasError) {  
          return const MaterialApp(  
            home: Scaffold(  
              body: Center(  
                child: Text(  
                  "The app its being NOT initialized - ERROR 404",  
                  style: TextStyle(  
                    color: Colors.cyan,  
                    fontSize: 40,  
                    fontWeight: FontWeight.bold,  
                  ),  
                ),  
              ),  
            ),  
          );  
        }  
      },  
    );  
    return MaterialApp(  
      debugShowCheckedModeBanner: false,  
      title: "Dev Job",  
      theme: ThemeData(  
        scaffoldBackgroundColor: Colors.black,  
        primarySwatch: Colors.blue,  
      ),  
      home: UserState(),  
    );  
  }  
}
```

Josue Garza
Prueba practica
12/12/2022

```
        );  
      });  
    }  
  }
```

En el archivo user_state.dart se realizan estas validaciones, considerando los siguientes errores:

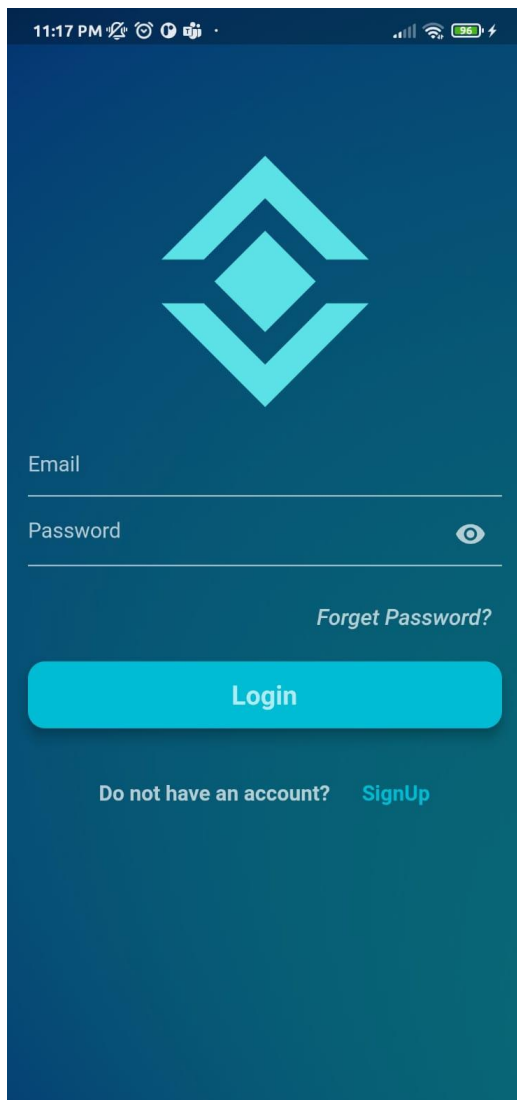
Logeado, no logeado, error en la base de datos, o en proceso de ingreso.

```
if (userSnapshot.data == null) {  
  print('user is not logged in yet');  
  return Login();  
}  
  
else if (userSnapshot.hasData) {  
  print('user is already logged in yet');  
  return JobScreen();  
}  
  
else if (userSnapshot.hasError) {  
  return const Scaffold(  
    body: Center(  
      child: Text('Error, try again latter'),  
    ),  
  );  
}  
  
else if (userSnapshot.connectionState == ConnectionState.waiting) {  
  return const Scaffold(  
    body: Center(  
      child: CircularProgressIndicator(),  
    ),  
  );  
}
```

Josue Garza
Prueba practica
12/12/2022

Login:

Login.dart



Para la creación de esta pantalla primero nos aseguramos de que existiera una conexión a firebase, y validamos si ya existía un usuario o no.

El corazón de esta pantalla lo encontramos en la siguiente función, aquí valida si existe un usuario según los controladores brindados, haciendo uso de “signInWithEmailAndPassword”, si existe el usuario le brinda entrada a las demás pantallas, en caso de que no se pongan las credenciales correctas no le brinda acceso.

```
void _submitFormOnLogin() async {  
  final isValid = _loginFormKey.currentState!.validate();  
  if (isValid) {  
    setState(() {  
      _isLoading = true;  
    });  
  }  
}
```

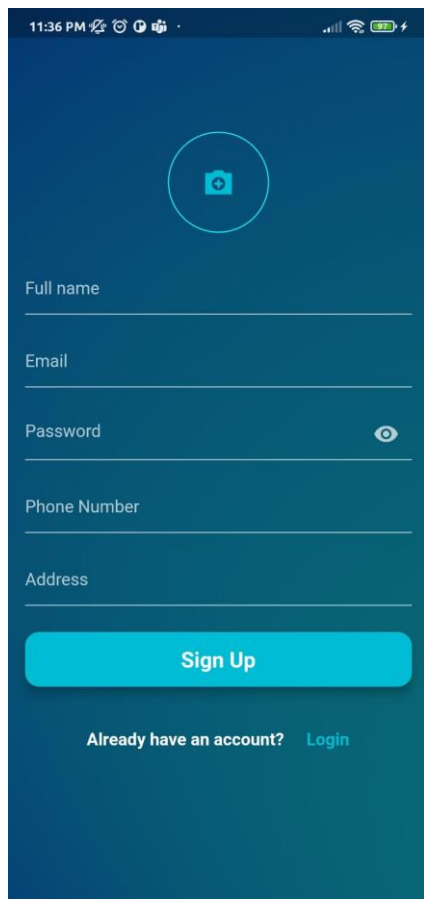
Josue Garza
Prueba practica
12/12/2022

```
    try {
      await _auth.signInWithEmailAndPassword(
        email: _emailTextController.text.trim().toLowerCase(),
        password: _passTextController.text.trim(),
      );
      Navigator.canPop(context) ? Navigator.pop(context) : null;
    } catch (error) {
      setState(() {
        _isLoading = false;
      });
      GlobalMethod.showErrorDialog(error: error.toString(), ctx: context);
      print('Error occurred $error');
    }
  }
  setState(() {
    _isLoading = false;
  });
}
```

Josue Garza
Prueba practica
12/12/2022

Sign Up:

En esta pantalla el usuario podrá registrarse en caso de no tener cuenta.



Para poder realizar esto creamos la siguiente función: `_submitFormOnSignUp()`, esta función se encarga de subir los datos solicitados y crear un usuario nuevo en la tabla users

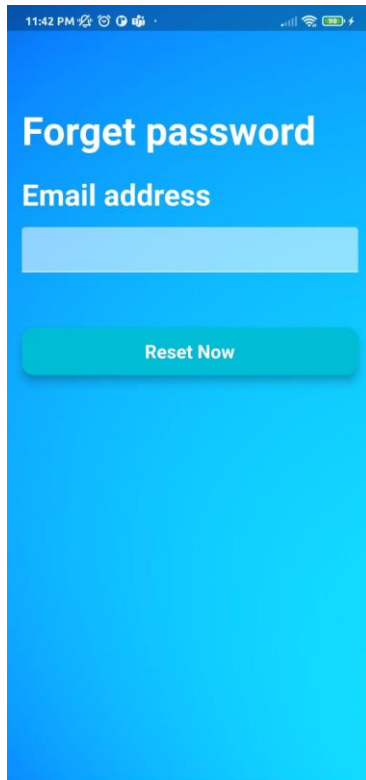
```
void _submitFormOnSignUp() async {  
  final isValid = _signUpFormKey.currentState!.validate();  
  if (isValid) {  
    if (imageFile == null) {  
      GlobalMethod.showErrorDialog(  
        error: 'Please pick an image', ctx: context);  
      return;  
    }  
    setState(() {  
      _isLoading = true;  
    });  
    try {  
      await _auth.createUserWithEmailAndPassword(  
        email: _emailTextController.text.trim().toLowerCase(),  
        password: _passTextController.text.trim(),  
      );  
      final User? user = _auth.currentUser;  
      final _uid = user!.uid;  
    }  
  }  
}
```

Josue Garza
Prueba practica
12/12/2022

```
        final ref = FirebaseStorage.instance
            .ref()
            .child('userImages')
            .child(_uid + '.jpg');
        await ref.putFile(imageFile!);
        imageUrl = await ref.getDownloadURL();
        FirebaseFirestore.instance.collection('users').doc(_uid).set({
            'id': _uid,
            'name': _fullNameController.text,
            'email': _emailTextController.text,
            'userImage': imageUrl,
            'phoneNumber': _phoneNumberController.text,
            'location': _locationController.text,
            'createdAt': Timestamp.now(),
        });
        Navigator.canPop(context) ? Navigator.pop(context) : null;
    } catch (error) {
        setState(() {
            _isLoading = false;
        });
        GlobalMethod.showErrorDialog(error: error.toString(), ctx: context);
    }
}
setState(() {
    _isLoading = false;
});
}
```


Josue Garza
Prueba practica
12/12/2022
Forget Password

En caso de que el usuario olvide su contraseña acá podrá restablecerla, se le enviara un correo electrónico de restablecimiento de contraseña y podrá cambiarla sin problema alguno.



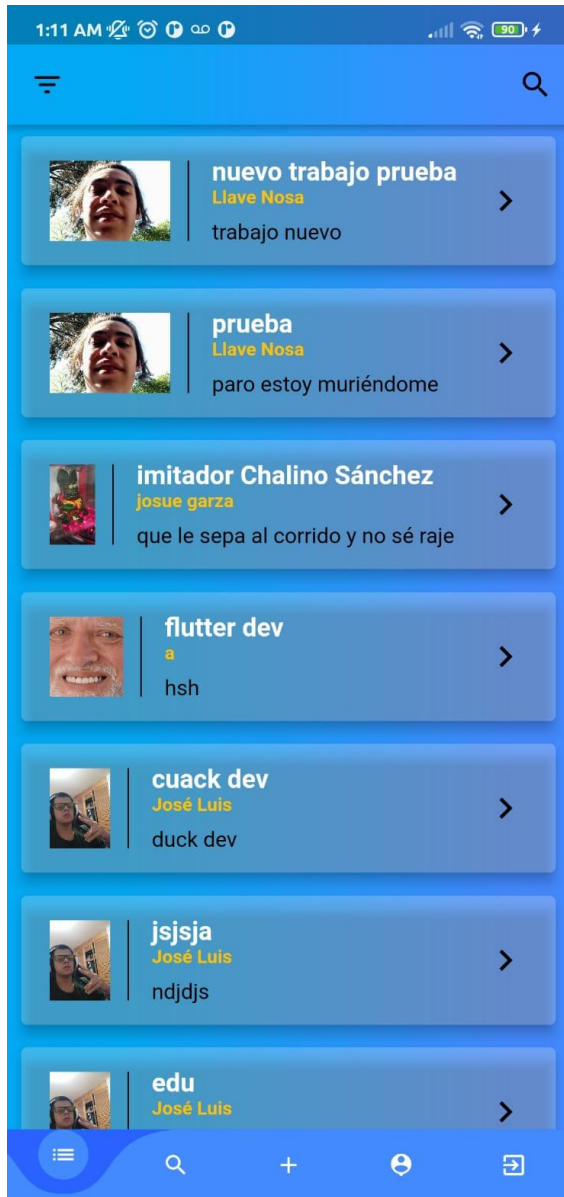
Para todo esto se creó la siguiente función `_forgetPassSubmitForm()`, una vez que se rellena el formulario se manda llamar esta función, se planea en un futuro mejorar esta pantalla y solicitar más datos para la validación del usuario, pero se planea implementar en un futuro.

```
void _forgetPassSubmitForm() async {  
  try {  
    await _auth.sendPasswordResetEmail(  
      email: _forgetPassTextController.text,  
    );  
    Navigator.pushReplacement(  
      context, MaterialPageRoute(builder: (_) => Login());  
    );  
  } catch (error) {  
    Fluttertoast.showToast(msg: error.toString());  
  }  
}
```

Josue Garza
Prueba practica
12/12/2022

Muestreo de productos:

Una vez ingresado el usuario puede acceder a la vista que muestra los productos, esta pantalla es una prueba, en el futuro se piensa cambiar la UI y se desea refinar lo que se muestra al usuario.



Josue Garza

Prueba practica

12/12/2022

La función encargada de mostrar la información solicitada en la base de datos es:

```
_showTaskCategoriesDialog({required Size size}) {
  showDialog(
    context: context,
    builder: (ctx) {
      return AlertDialog(
        backgroundColor: Colors.black54,
        title: const Text(
          'Product Category',
          textAlign: TextAlign.center,
          style: TextStyle(fontSize: 20, color: Colors.white),
        ),
        content: Container(
          width: size.width * 0.9,
          child: ListView.builder(
            shrinkWrap: true,
            itemCount: Persistent.jobCategoryList.length,
            itemBuilder: (ctx, index) {
              return InkWell(
                onTap: () {
                  setState(() {
                    jobCategoryFilter = Persistent.jobCategoryList[index];
                  });
                  Navigator.canPop(context) ? Navigator.pop(context) : null;
                  print(
                    'jobCategoryList[index],${Persistent.jobCategoryList[index]}');
                },
                child: Row(
                  children: [
                    const Icon(
                      Icons.arrow_right_alt_outlined,
                      color: Colors.grey,
                    ),
                    Padding(
                      padding: const EdgeInsets.all(8.0),
                      child: Text(
                        Persistent.jobCategoryList[index],
                        style: const TextStyle(
                          color: Colors.grey,
                          fontSize: 16,
                        ),
                      ),
                    ),
                  ],
                ),
              );
            },
          ),
        actions: [
          TextButton(
            onPressed: () {
              Navigator.canPop(context) ? Navigator.pop(context) : null;
            },
          ),
        ],
      );
    },
  );
}
```

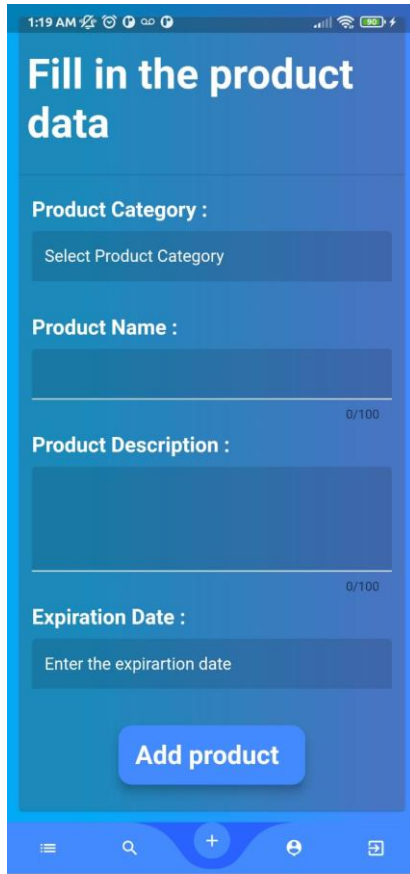
Josue Garza
Prueba practica
12/12/2022

```
        child: const Center(  
          child: Text(  
            'Close',  
            style: TextStyle(  
              color: Colors.white,  
              fontSize: 16,  
            ),  
          ),  
        ),  
      ),  
    ),  
  ),  
  TextButton(  
    onPressed: () {  
      setState(() {  
        jobCategoryFilter = null;  
      });  
      Navigator.canPop(context) ? Navigator.pop(context) : null;  
    },  
    child: const Text(  
      'Cancel filter',  
      style: TextStyle(color: Colors.white),  
    ),  
  ),  
],  
);  
},  
);  
}
```

Josue Garza
Prueba practica
12/12/2022

Subida de productos:

Para subir un producto deberás realizarlo mediante la siguiente pantalla:



La función que realiza este trabajo es la siguiente:

```
void _uploadTask() async {
  final jodId = const Uuid().v4();
  User? user = FirebaseAuth.instance.currentUser;
  final _uid = user!.uid;
  final isValid = _formKey.currentState!.validate();

  if (isValid) {
    if (_jobDeadlineDateController.text == 'Job DeadLine date' ||
        _jobCategoryController.text == 'Select Job Category') {
      GlobalMethod.showErrorDialog(
        error: 'Please pick everything', ctx: context);
      return;
    }
    setState(() {
      _isLoading = true;
    });
    try {
      await FirebaseFirestore.instance.collection('jobs').doc(jodId).set({
```

Josue Garza
Prueba practica
12/12/2022

```
        'jobId': jobId,  
        'uploadedBy': _uid,  
        'email': user.email,  
        'jobTitle': _jobTitleController.text,  
        'jobDescription': _jobDescriptionController.text,  
        'deadlineDate': _jobDeadlineDateController.text,  
        'deadlineDateTimeStamp': deadlineDateTimeStamp,  
        'jobCategory': _jobCategoryController.text,  
        'jobComments': [],  
        'recruitment': true,  
        'createdAt': Timestamp.now(),  
        'name': name,  
        'userImage': userImage,  
        'location': location,  
        'applicants': 0,  
    });  
    await Fluttertoast.showToast(  
      msg: 'The task has been uploaded',  
      toastLength: Toast.LENGTH_LONG,  
      backgroundColor: Colors.grey,  
      fontSize: 18.0,  
    );  
    _jobTitleController.clear();  
    _jobDescriptionController.clear();  
    setState(() {  
      _jobCategoryController.text = 'Choose job category';  
      _jobDeadlineDateController.text = 'Choose job Deadline date';  
    });  
  } catch (error) {  
    {  
      setState(() {  
        _isLoading = false;  
      });  
      GlobalMethod.showErrorMessage(error: error.toString(), ctx: context);  
    }  
  } finally {  
    setState(() {  
      _isLoading = false;  
    });  
  }  
} else {  
  print('Its not valid');  
}
```

Josue Garza
Prueba practica
12/12/2022

Repositorio Github

<https://github.com/josuegarza42/om8Test>

Retos del proyecto, investigación y soluciones

Hubo varios retos en el proyecto, redactare a continuación los retos más relevantes a mi parecer.

Conexión con firebase y corrección de errores en archivos build.gradle:

Para poder comenzar con el proyecto debía de checar primero que nada que la app pudiera conectar con una base de datos, en este caso fue firebase, pues de nada me sirve una app que funcione únicamente en local o que no sea capaz de manipular datos.

Seguido de esto tuve un problema con las versiones del sdk mínimo que debe de soportar la aplicación, personalmente no me había enfrentado al problema anteriormente, pero un artículo en stackoverflow salvo el día, fue frustrante pues el error no era intuitivo y pase unos días averiguando que podría estar mal, aprendí a resolver problemas relacionados con versiones de controladores gracias a este proyecto.

Solución:

<https://stackoverflow.com/questions/57238933/gradle-failure-a-problem-occurred-evaluating-project-app>

Manejo de usuarios en flutter

El hacer un login podría parecer sencillo, pero cuando empiezas a implementar validaciones de seguridad te das cuenta de que el tema puede ser tan extenso como te lo propongas, realice el registro de usuarios mediante las pantallas afiliadas al login, en otros lenguajes esto es algo complicado, pero en flutter pude resolver esto con ciertas dependencias que hacen la tarea mas sencilla y con un gran resultado en la implementación. Se realizo con éxito las tareas de logear, registrar y modificar contraseñas con éxito.

Solución:

<https://www.udemy.com/user/muhammad-ali-zeb-3/> (Varios cursos de este autor)

<https://www.udemy.com/course/learn-flutter-3-firebase-build-freelancer-clone-app/learn/lecture/33660074#overview>

Crud de objetos

El proyecto claramente menciona que se desea unas pantallas en las cuales el usuario puede cargar productos al carrito, ver los productos etc., en mi caso particular me encontré escaso de tiempo cuando llegué a este punto, y decidí mostrar como se hace una inserción según el usuario registrado y como se ve esta inserción en la pantalla correspondiente, también como se elimina el producto de la lista generada.

Solución:

<https://www.udemy.com/user/muhammad-ali-zeb-3/> (Varios cursos de este autor)

Josue Garza
Prueba practica
12/12/2022

<https://www.udemy.com/course/learn-flutter-3-firebase-build-freelancer-clone-app/learn/lecture/33660208#overview>

Una vez que pude mostrar productos que yo creaba ahora si podía pasar al siguiente paso, el cual era la implementación de la API de FakeStore y esta misma poder brindar la implementación del modulo de pago con stripe. Desafortunadamente el tiempo me alcanzo y esto será implementado en una siguiente versión.

Retos por cumplir:

Implementación de fakestore y modulo de cobro con stripe.

Documentación Stripe https://docs.page/flutter-stripe/flutter_stripe

API de Fake Store: <https://fakestoreapi.com/docs>