

UNIVERSIDAD CENTRAL DEL ECUADOR

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

SISTEMAS DE LA INFORMACIÓN



DISPOSITIVOS MÓVILES

GRUPO 09 Trabajo Grupal: Aplicación Móvil

Alex Morales

Fausto Guamán

Fernando Novillo

Kenneth Taipe

2025-2025

Introducción

El presente informe describe el desarrollo de una aplicación móvil Android implementada en el entorno de desarrollo Android Studio, utilizando el lenguaje de programación Kotlin y la integración de servicios en la nube mediante Firebase. La aplicación tiene como objetivo principal la gestión de usuarios a través de un sistema de autenticación basado en correo electrónico y contraseña, complementado con el uso de recursos propios del dispositivo móvil, como el sensor de la cámara y el almacenamiento interno.

Además del proceso de autenticación, la aplicación incorpora funcionalidades de personalización del perfil del usuario y la gestión de contenido multimedia, permitiendo la captura y almacenamiento de imágenes, así como la creación de una galería de fotografías persistente. El proyecto busca aplicar los conocimientos adquiridos en la asignatura de Dispositivos Móviles, integrando interfaces gráficas modernas, lógica de negocio y servicios externos dentro de una solución funcional.

Objetivos

Objetivo General

- Desarrollar una aplicación móvil Android que implemente autenticación de usuarios mediante Firebase y el uso del sensor de la cámara del dispositivo, aplicando buenas prácticas de desarrollo móvil.

Objetivos Específicos

- Diseñar una interfaz gráfica clara, intuitiva y funcional para el registro e inicio de sesión de usuarios.
- Implementar la autenticación de usuarios utilizando correo electrónico y contraseña.
- Integrar Firebase Authentication para la validación segura de credenciales.

- Permitir la captura de imágenes mediante la cámara del dispositivo y la selección desde la galería.
- Implementar una galería de fotografías persistente para usuarios autenticados.
- Gestionar el almacenamiento local de imágenes y datos del perfil del usuario.

Alcance del Proyecto

La aplicación desarrollada se enfoca en la implementación de funcionalidades básicas de autenticación y gestión multimedia. El alcance del proyecto incluye:

- Registro e inicio de sesión de usuarios mediante Firebase Authentication.
- Validación de credenciales y control de sesión activa.
- Visualización de información del usuario autenticado.
- Captura de fotografías utilizando el sensor de la cámara del dispositivo.
- Selección de imágenes desde la galería del teléfono.
- Almacenamiento local de imágenes y datos del perfil.
- Creación y visualización de una galería de imágenes persistente.

Herramientas y Tecnologías Utilizadas

- Android Studio: Entorno de desarrollo integrado para aplicaciones Android.
- Kotlin: Lenguaje de programación principal del proyecto.
- Jetpack Compose: Construcción de interfaces gráficas modernas.
- Firebase Authentication: Gestión de usuarios y autenticación.
- SharedPreferences: Persistencia de datos locales del perfil.
- Sensor de Cámara: Captura de imágenes en tiempo real.
- Almacenamiento Interno: Gestión de archivos multimedia.

Estructura General del Proyecto

La aplicación se encuentra organizada de forma modular, separando la lógica de negocio de la interfaz gráfica. Los principales componentes del sistema son:

- MainActivity: Controla la navegación entre pantallas.
- LoginScreen y RegisterScreen: Interfaces de autenticación.
- LoginViewModel: Gestión de la lógica de autenticación.
- ProfileViewScreen: Visualización del perfil del usuario.
- GalleryCaptureScreen: Captura de imágenes.
- GalleryViewScreen: Visualización de la galería de fotos.

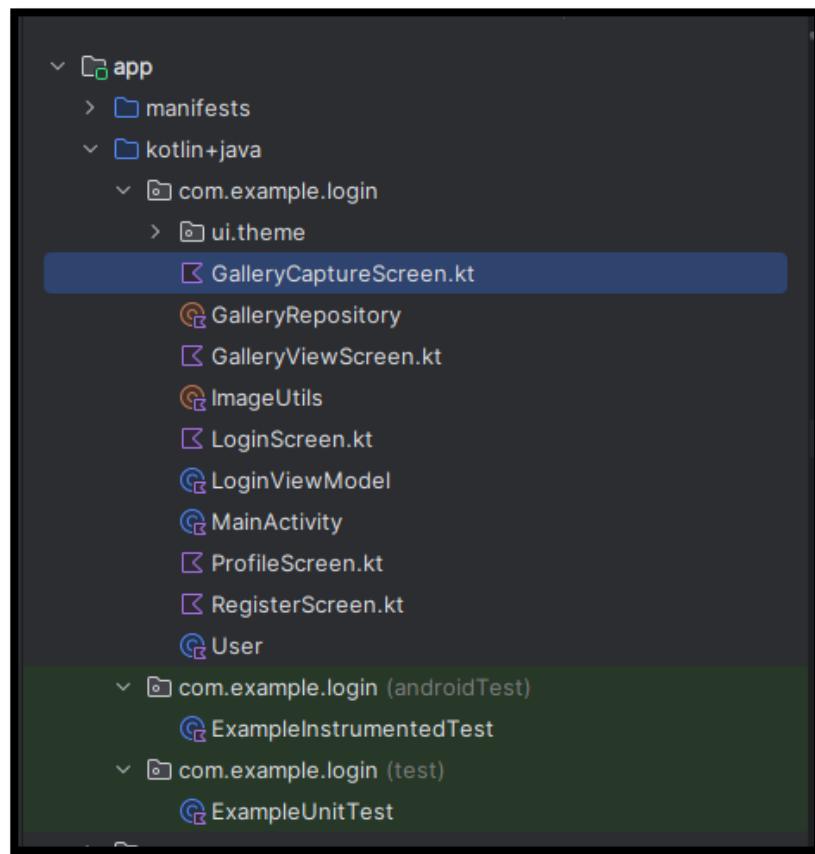


Ilustración 1: Estructura del proyecto

GalleryCaptureScreen

El código implementa una pantalla en Jetpack Compose que gestiona la captura de imágenes mediante la cámara del dispositivo, guarda las fotografías de forma local en una lista y controla el flujo de finalización de una galería, habilitando la opción de continuar únicamente cuando existen imágenes registradas.

```

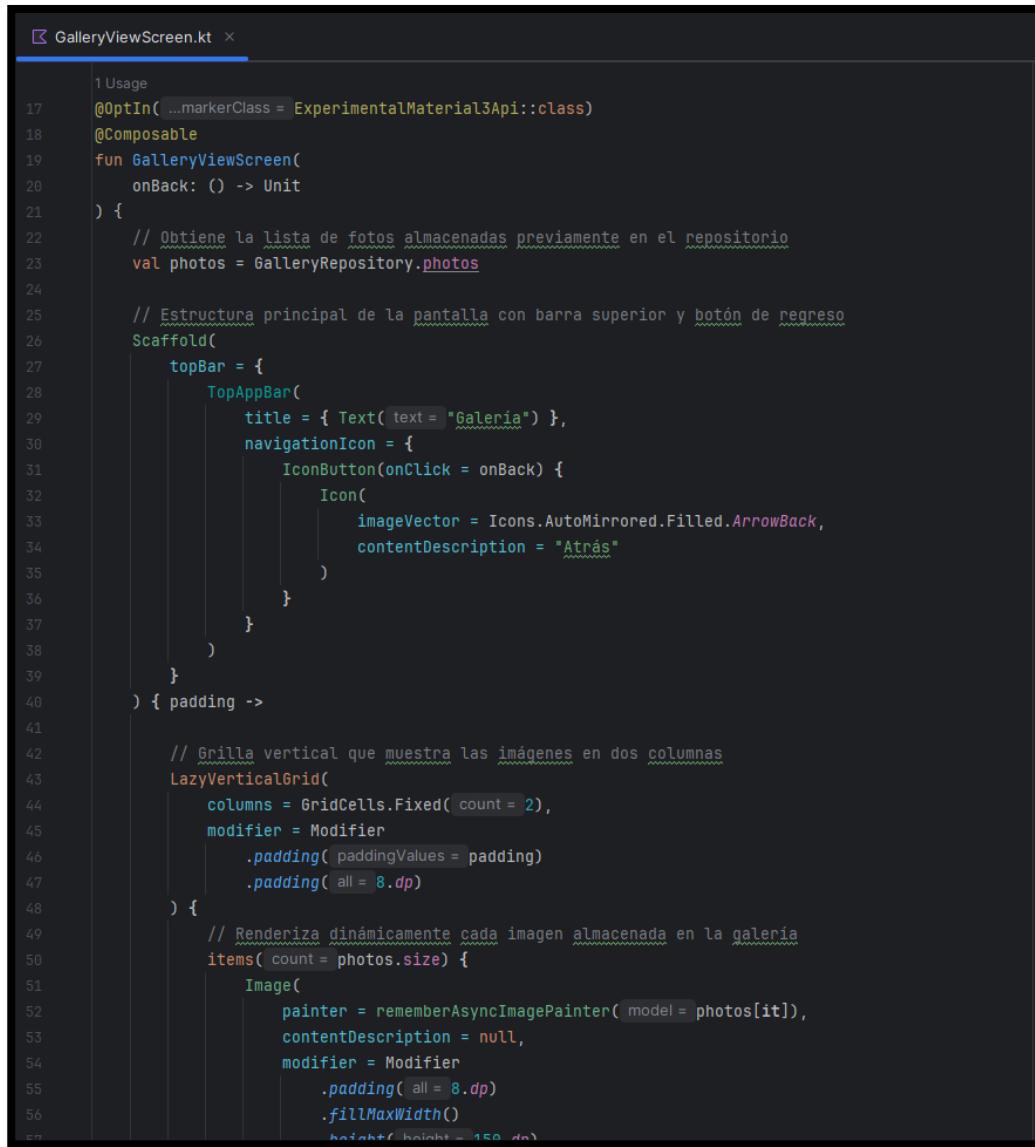
1 Usage
19 @OptIn( ...markerClass = ExperimentalMaterial3Api::class)
20 @Composable
21 fun GalleryCaptureScreen(
22     onFinish: () -> Unit
23 ) {
24     // Obtiene el contexto de la aplicación y crea una lista reactiva para almacenar las fotos capturadas
25     val context = LocalContext.current
26     val photos = remember { mutableListOf<Uri>() }
27
28     // Configura el lanzador de la cámara para capturar una imagen y guardarla localmente
29     val cameraLauncher = rememberLauncherForActivityResult(
30         contract = ActivityResultContracts.TakePicturePreview()
31     ) { bitmap ->
32         bitmap?.let {
33             val uri = ImageUtils.saveBitmap(context, bitmap = it)
34             photos.add(uri)
35         }
36     }
37
38     // Estructura principal de la pantalla con barra superior
39     Scaffold(
40         topBar = {
41             TopAppBar(
42                 title = { Text( text = "Crear Galería" ) }
43             )
44         }
45     ) { padding ->
46
47         // Contenedor principal que organiza los elementos de la pantalla en columna
48         Column(
49             modifier = Modifier
50                 .fillMaxSize()
51                 .padding( paddingValues = padding )
52                 .padding( all = 24.dp ),
53                 horizontalAlignment = Alignment.CenterHorizontally
54         ) {
55
56             // Muestra el título y la cantidad de fotos tomadas
57             Text(
58                 text = "Captura tus recuerdos",
59                 style = MaterialTheme.typography.headlineSmall

```

Ilustración 2: *GalleryCaptureScreen*

GalleryViewScreen

El código define una pantalla en Jetpack Compose encargada de visualizar una galería de imágenes previamente almacenadas. Utiliza una grilla vertical de dos columnas para mostrar las fotografías de forma ordenada y uniforme, cargándolas dinámicamente desde un repositorio. Además, incorpora una barra superior con un botón de regreso, que permite al usuario volver a la pantalla anterior manteniendo una navegación clara dentro de la aplicación.



```

1 Usage
17 @OptIn( ...markerClass = ExperimentalMaterial3Api::class)
18 @Composable
19 fun GalleryViewScreen(
20     onBack: () -> Unit
21 ) {
22     // Obtiene la lista de fotos almacenadas previamente en el repositorio
23     val photos = GalleryRepository.photos
24
25     // Estructura principal de la pantalla con barra superior y botón de regreso
26     Scaffold(
27         topBar = {
28             TopAppBar(
29                 title = { Text( text = "Galería" ) },
30                 navigationIcon = {
31                     IconButton(onClick = onBack) {
32                         Icon(
33                             imageVector = Icons.AutoMirrored.Filled.ArrowBack,
34                             contentDescription = "Atrás"
35                         )
36                     }
37                 }
38             )
39         }
40     ) { padding ->
41
42         // Grilla vertical que muestra las imágenes en dos columnas
43         LazyVerticalGrid(
44             columns = GridCells.Fixed( count = 2 ),
45             modifier = Modifier
46                 .padding( paddingValues = padding )
47                 .padding( all = 8.dp )
48         ) {
49             // Renderiza dinámicamente cada imagen almacenada en la galería
50             items( count = photos.size ) {
51                 Image(
52                     painter = rememberAsyncImagePainter( model = photos[it] ),
53                     contentDescription = null,
54                     modifier = Modifier
55                         .padding( all = 8.dp )
56                         .fillMaxWidth()
57                         .height( height = 150.dp )
58                 )
59             }
60         }
61     }
62 }

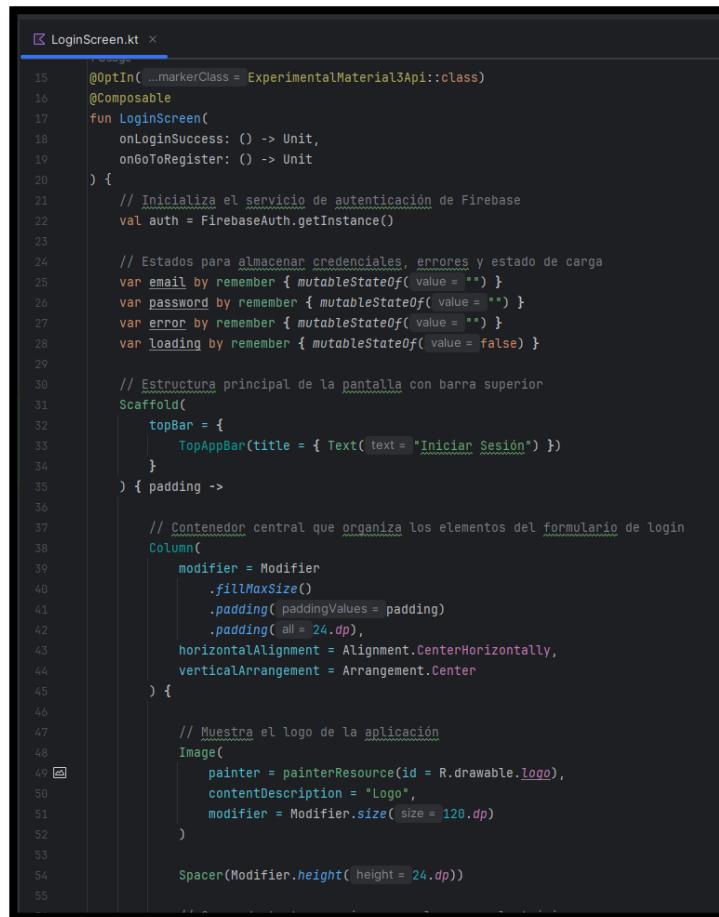
```

Ilustración 3:GalleryViewScreen

LoginScreen – Lógica y Funcionamiento

La pantalla LoginScreen está respaldada por una Activity o Fragment asociado a un ViewModel (LoginViewModel), el cual se encarga de gestionar la autenticación del usuario.

- Inicializa Firebase Authentication.
- Captura los datos ingresados por el usuario (correo y contraseña).
- Valida que los campos no estén vacíos.
- Escucha la respuesta del servicio de autenticación.
- Controla la navegación hacia la siguiente pantalla en caso de éxito.
- Maneja errores de autenticación mediante mensajes al usuario.

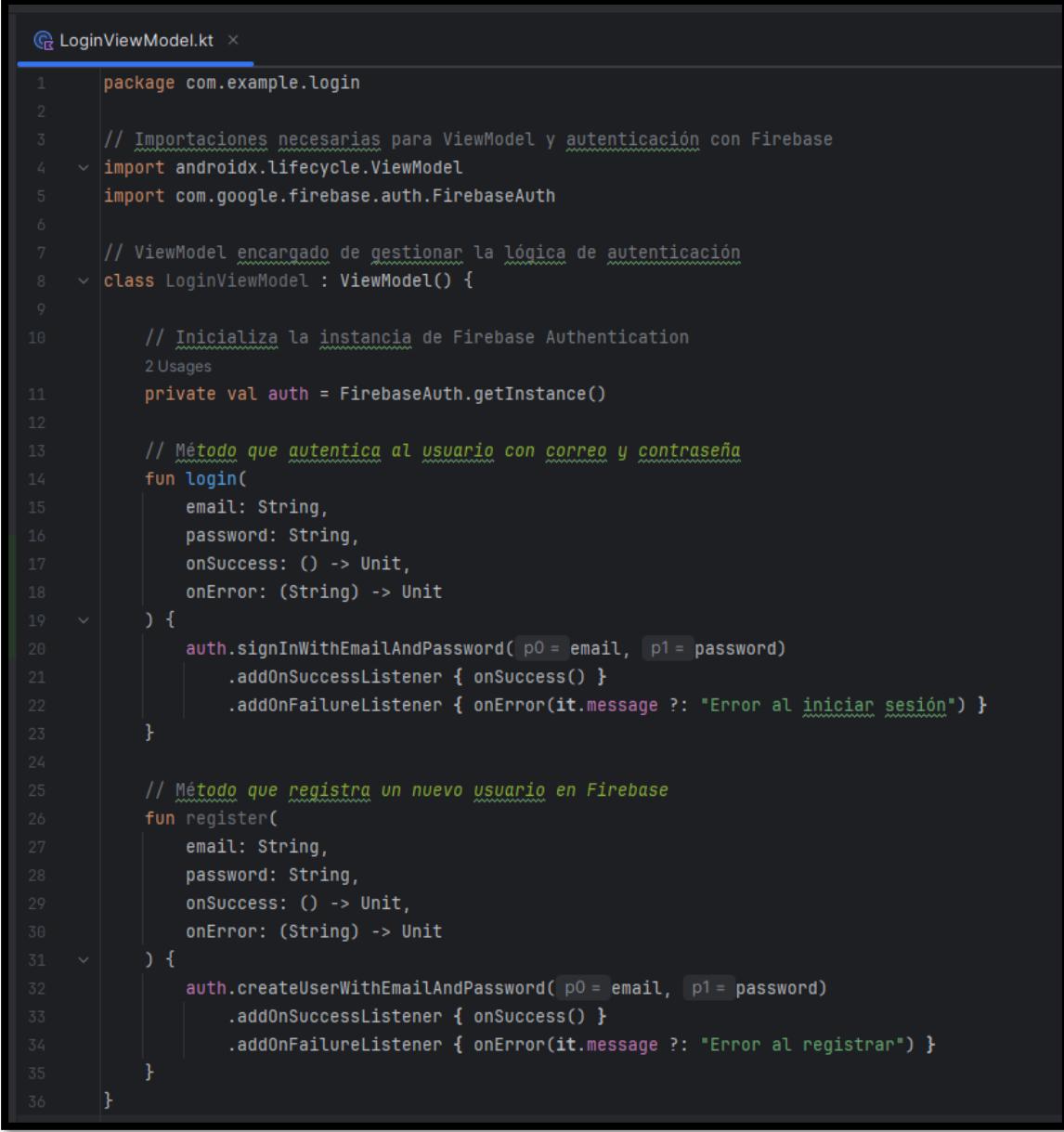


```

15  @OptIn( ...markerClass = ExperimentalMaterial3Api::class)
16  @Composable
17  fun LoginScreen(
18      onLoginSuccess: () -> Unit,
19      onGoToRegister: () -> Unit
20  ) {
21      // Inicializa el servicio de autenticación de Firebase
22      val auth = FirebaseAuth.getInstance()
23
24      // Estados para almacenar credenciales, errores y estado de carga
25      var email by remember { mutableStateOf( value = "" ) }
26      var password by remember { mutableStateOf( value = "" ) }
27      var error by remember { mutableStateOf( value = "" ) }
28      var loading by remember { mutableStateOf( value = false ) }
29
30      // Estructura principal de la pantalla con barra superior
31      Scaffold(
32          topBar = {
33              TopAppBar(title = { Text( text = "Iniciar Sesión" ) })
34          }
35      ) { padding ->
36
37          // Contenedor central que organiza los elementos del formulario de login
38          Column(
39              modifier = Modifier
40                  .fillMaxSize()
41                  .padding( paddingValues = padding )
42                  .padding( all = 24.dp ),
43                  horizontalAlignment = Alignment.CenterHorizontally,
44                  verticalArrangement = Arrangement.Center
45          ) {
46
47              // Muestra el logo de la aplicación
48              Image(
49                  painter = painterResource(id = R.drawable.logo),
50                  contentDescription = "Logo",
51                  modifier = Modifier.size( size = 120.dp )
52              )
53
54              Spacer(Modifier.height( height = 24.dp ))
55
56          }
57
58      }
59
60  }
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
317
318
319
319
320
321
322
323
324
325
325
326
327
327
328
329
329
330
331
332
333
333
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
141
```

LoginViewModel

El código define un ViewModel que centraliza la lógica de autenticación de la aplicación, utilizando Firebase Authentication para manejar el inicio de sesión y el registro de usuarios, y comunicando el resultado de estas operaciones mediante funciones de éxito y error.



```

1 package com.example.login
2
3 // Importaciones necesarias para ViewModel y autenticación con Firebase
4 import androidx.lifecycle.ViewModel
5 import com.google.firebase.auth.FirebaseAuth
6
7 // ViewModel encargado de gestionar la lógica de autenticación
8 class LoginViewModel : ViewModel() {
9
10    // Inicializa la instancia de Firebase Authentication
11    private val auth = FirebaseAuth.getInstance()
12
13    // Método que autentica al usuario con correo y contraseña
14    fun login(
15        email: String,
16        password: String,
17        onSuccess: () -> Unit,
18        onError: (String) -> Unit
19    ) {
20        auth.signInWithEmailAndPassword( p0 = email, p1 = password )
21            .addOnSuccessListener { onSuccess() }
22            .addOnFailureListener { onError(it.message ?: "Error al iniciar sesión") }
23    }
24
25    // Método que registra un nuevo usuario en Firebase
26    fun register(
27        email: String,
28        password: String,
29        onSuccess: () -> Unit,
30        onError: (String) -> Unit
31    ) {
32        auth.createUserWithEmailAndPassword( p0 = email, p1 = password )
33            .addOnSuccessListener { onSuccess() }
34            .addOnFailureListener { onError(it.message ?: "Error al registrar") }
35    }
36}

```

Ilustración 5:LoginViewModel

MainActivity

El código define la actividad principal de la aplicación, donde se configura el tema visual y se gestiona la navegación entre pantallas (login, registro, perfil, captura y visualización de galería) utilizando Jetpack Navigation Compose.



```

11 </> class MainActivity : ComponentActivity() {
12
13     override fun onCreate(savedInstanceState: Bundle?) {
14         super.onCreate(savedInstanceState)
15
16         // Define el contenido principal usando Jetpack Compose y aplica el tema
17         setContent {
18             LoginTheme {
19
20                 // Controlador de navegación para gestionar las pantallas
21                 val navController = rememberNavController()
22
23                 // Configura el grafo de navegación y la pantalla inicial
24                 NavHost(
25                     navController = navController,
26                     startDestination = "login"
27                 ) {
28
29                     // Ruta de inicio de sesión
30                     composable(route = "login") {
31                         LoginScreen(
32                             onLoginSuccess = {
33                                 navController.navigate(route = "welcome") {
34                                     popUpTo(route = "login") { inclusive = true }
35                             }
36                         },
37                         onGoToRegister = {
38                             navController.navigate(route = "register")
39                         }
40                     )
41                 }
42
43                 // Ruta de registro de usuario
44                 composable(route = "register") {
45                     RegisterScreen(
46                         onRegisterCompleted = {
47                             navController.navigate(route = "login") {
48                                 popUpTo(route = "register") { inclusive = true }
49                             }
50                         }
51                     )
52                 }
53             }
54         }
55     }
56 }

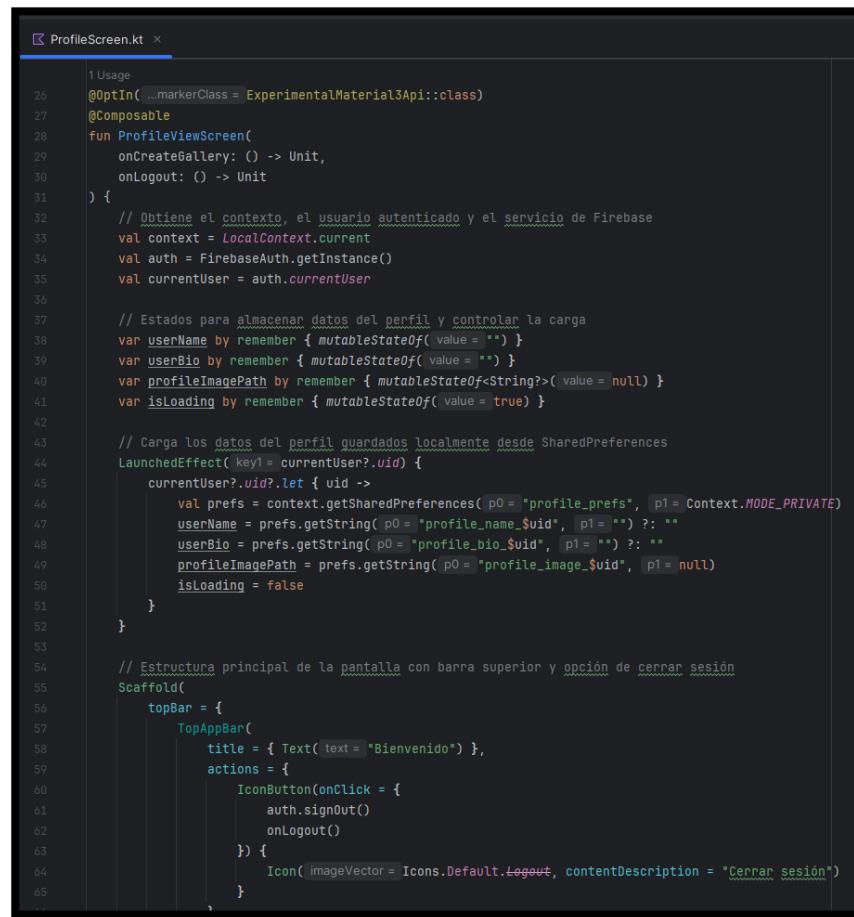
```

Ilustración 6: MainActivity

ProfileScreen

Está diseñada para mostrar información asociada al usuario autenticado. A nivel lógico, esta pantalla:

- Recupera los datos del usuario desde Firebase.
- Muestra información relevante del perfil.
- Sirve como base para futuras funcionalidades como edición de datos o configuración de cuenta.



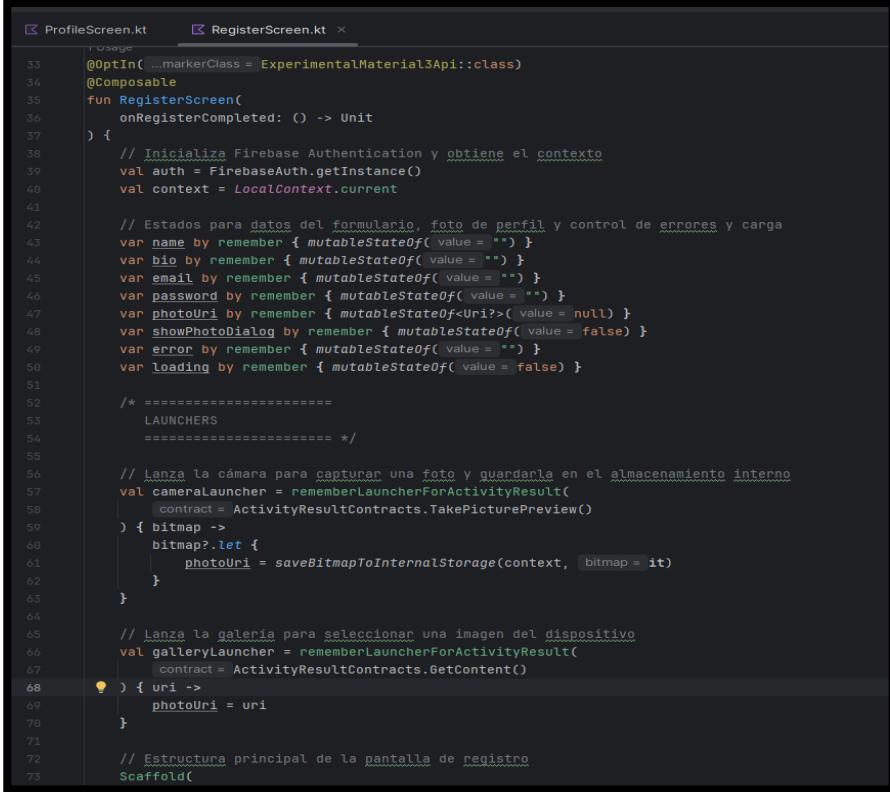
```

1 Usage
2 @OptIn( ...markerClass = ExperimentalMaterial3Api::class)
3 @Composable
4 fun ProfileViewScreen(
5     onCreateGallery: () -> Unit,
6     onLogout: () -> Unit
7 ) {
8     // Obtiene el contexto, el usuario autenticado y el servicio de Firebase
9     val context = LocalContext.current
10    val auth = FirebaseAuth.getInstance()
11    val currentUser = auth.currentUser
12
13    // Estados para almacenar datos del perfil y controlan la carga
14    var userName by remember { mutableStateOf(value = "") }
15    var userBio by remember { mutableStateOf(value = "") }
16    var profileImagePath by remember { mutableStateOf<String?>(value = null) }
17    var isLoading by remember { mutableStateOf(value = true) }
18
19    // Carga los datos del perfil guardados localmente desde SharedPreferences
20    LaunchedEffect(key1 = currentUser?.uid) {
21        currentUser?.uid?.let { uid ->
22            val prefs = context.getSharedPreferences(p0 = "profile_prefs", p1 = Context.MODE_PRIVATE)
23            userName = prefs.getString(p0 = "profile_name_$uid", p1 = "") ?: ""
24            userBio = prefs.getString(p0 = "profile_bio_$uid", p1 = "") ?: ""
25            profileImagePath = prefs.getString(p0 = "profile_image_$uid", p1 = null)
26            isLoading = false
27        }
28    }
29
30    // Estructura principal de la pantalla con barra superior y opción de cerrar sesión
31    Scaffold(
32        topBar = {
33            TopAppBar(
34                title = { Text(text = "Bienvenido") },
35                actions = {
36                    IconButton(onClick = {
37                        auth.signOut()
38                        onLogout()
39                    }) {
40                        Icon(imageVector = Icons.Default.Logout, contentDescription = "Cerrar sesión")
41                    }
42                }
43            )
44        }
45    )
46 }
47
48 
```

Ilustración 7:ProfileScreen

RegisterScreen

Este código implementa una pantalla de registro de usuarios en Android con Jetpack Compose, donde el usuario ingresa sus datos personales, selecciona o toma una foto de perfil mediante la cámara o la galería, y crea una cuenta utilizando Firebase Authentication. Además, la información del perfil y la imagen se guardan localmente en el dispositivo, mostrando validaciones, estados de carga y mensajes de error durante el proceso de registro.



```

1  // Usage
2
3  @OptIn( ...markerClass = ExperimentalMaterial3Api::class)
4  @Composable
5  fun RegisterScreen(
6      onRegisterCompleted: () -> Unit
7  ) {
8      // Inicializa Firebase Authentication y obtiene el contexto
9      val auth = FirebaseAuth.getInstance()
10     val context = LocalContext.current
11
12     // Estados para datos del formulario, foto de perfil y control de errores y carga
13     var name by remember { mutableStateOf("") }
14     var bio by remember { mutableStateOf("") }
15     var email by remember { mutableStateOf("") }
16     var password by remember { mutableStateOf("") }
17     var photoUri by remember { mutableStateOf<Uri?>() }
18     var showPhotoDialog by remember { mutableStateOf(false) }
19     var error by remember { mutableStateOf("") }
20     var loading by remember { mutableStateOf(false) }
21
22     /* =====
23      LAUNCHERS
24      ===== */
25
26     // Lanza la cámara para capturar una foto y guardarla en el almacenamiento interno
27     val cameraLauncher = rememberLauncherForActivityResult(
28         contract = ActivityResultContracts.TakePicturePreview()
29     ) { bitmap ->
30         bitmap?.let {
31             photoUri = saveBitmapToInternalStorage(context, bitmap = it)
32         }
33     }
34
35     // Lanza la galería para seleccionar una imagen del dispositivo
36     val galleryLauncher = rememberLauncherForActivityResult(
37         contract = ActivityResultContracts.GetContent()
38     ) { uri ->
39         photoUri = uri
40     }
41
42     // Estructura principal de la pantalla de registro
43     Scaffold(
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73

```

Ilustración 8: RegisterScreen

```
// Estructura principal de la pantalla de registro
Scaffold(
    topBar = { TopAppBar(title = { Text(text = "Registro") }) }
) { padding ->

    // Contenedor principal del formulario de registro
    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(paddingValues = padding)
            .padding(all = 24.dp),
        horizontalAlignment = Alignment.CenterHorizontally
    ) {

        // Área para seleccionar o capturar la foto de perfil
        Box(
            modifier = Modifier
                .size(size = 140.dp)
                .clip(CircleShape)
                .background(color = MaterialTheme.colorScheme.primaryContainer)
                .clickable { showPhotoDialog = true },
            contentAlignment = Alignment.Center
        ) {
            if (photoUri != null) {
                Image(
                    painter = rememberAsyncImagePainter(model = photoUri),
                    contentDescription = null,
                    modifier = Modifier.fillMaxSize(),
                    contentScale = ContentScale.Crop
                )
            } else {
                Icon(
                    imageVector = Icons.Default.Person,
                    contentDescription = null,
                    modifier = Modifier.size(size = 64.dp)
                )
            }
        }
    }
}
```

Ilustración 9: RegisterScreen

REGISTRO EN FIREBASE

La aplicación está correctamente conectada a Firebase mediante la integración del archivo google-services.json y la configuración del archivo build.gradle. En las actividades de Login y Registro se utilizan los métodos de autenticación provistos por Firebase:

- FirebaseAuth.getInstance().signInWithEmailAndPassword(...)
- FirebaseAuth.getInstance().createUserWithEmailAndPassword(...)

El manejo de resultados se realiza con onCompleteListener, lo que permite notificar al usuario si la operación fue exitosa o falló.

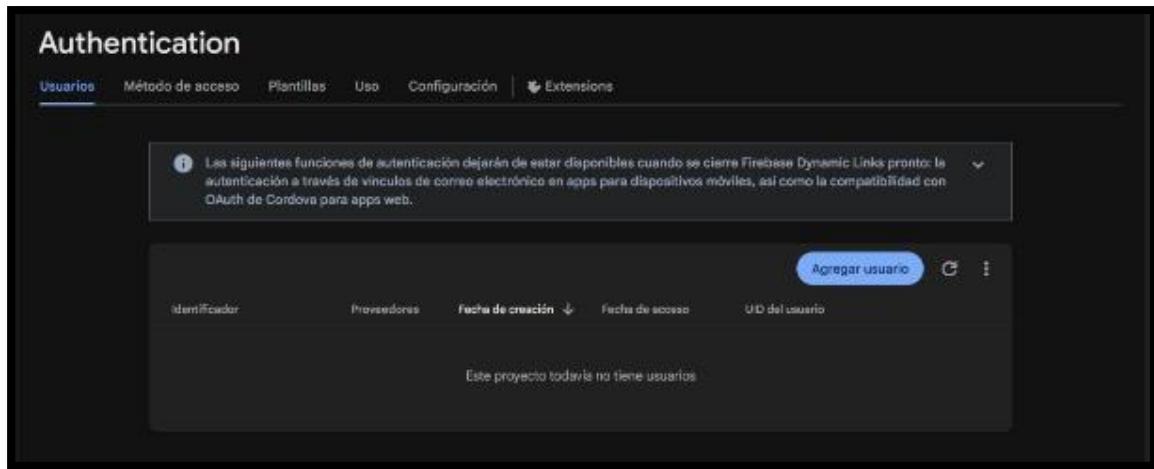


Ilustración 10: REGISTRO EN FIREBASE

DIAGRAMA DE FLUJO

El diagrama de flujo representa la lógica secuencial del sistema, mostrando las decisiones y procesos que se ejecutan desde que el usuario inicia la aplicación. Incluye la verificación de sesión activa, el proceso de autenticación, el registro de nuevos usuarios y el acceso a las funcionalidades principales, como la gestión de galerías y fotografías.

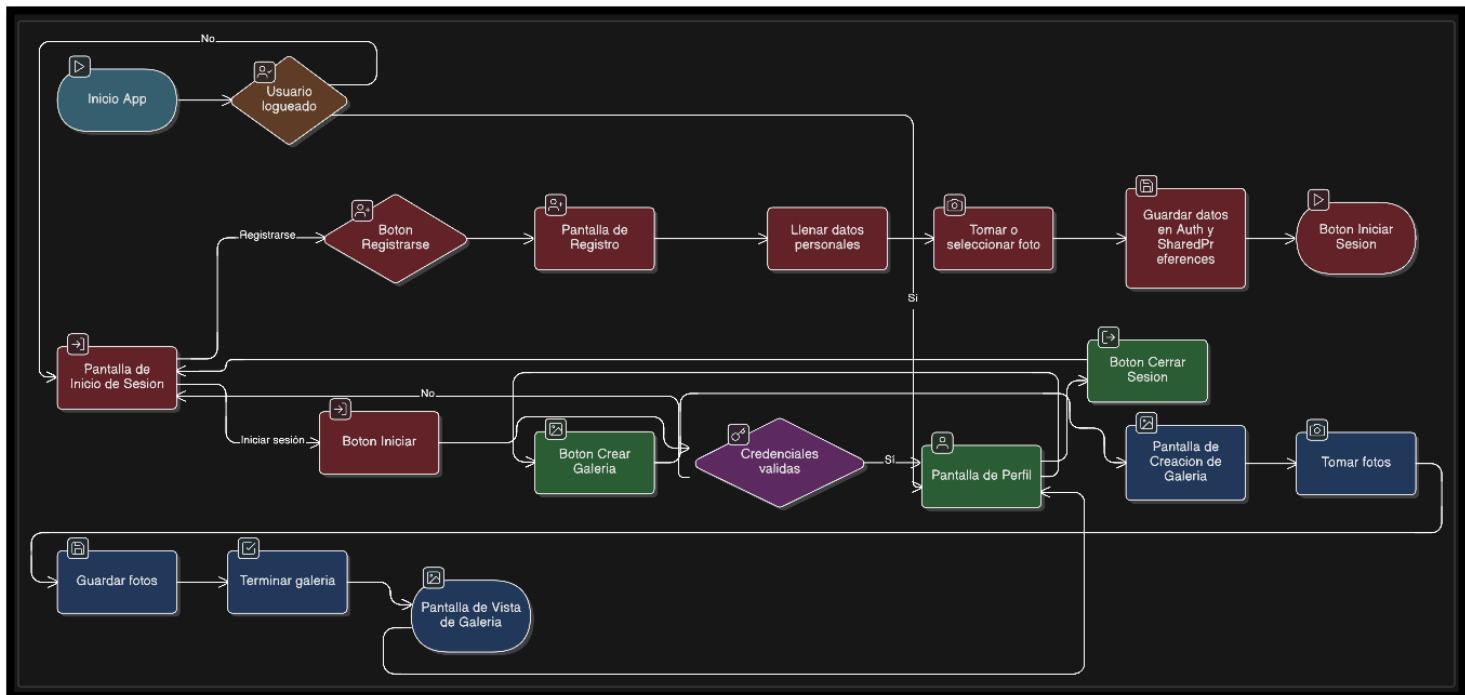


DIAGRAMA DE COMPONENTES

El diagrama de componentes muestra la arquitectura lógica del sistema, destacando la interacción entre la interfaz de usuario desarrollada con Jetpack Compose, el componente de navegación, Firebase Authentication para la gestión de usuarios, SharedPreferences para la persistencia local de datos, el uso de la cámara y galería del dispositivo, y el almacenamiento interno para la gestión de archivos multimedia.

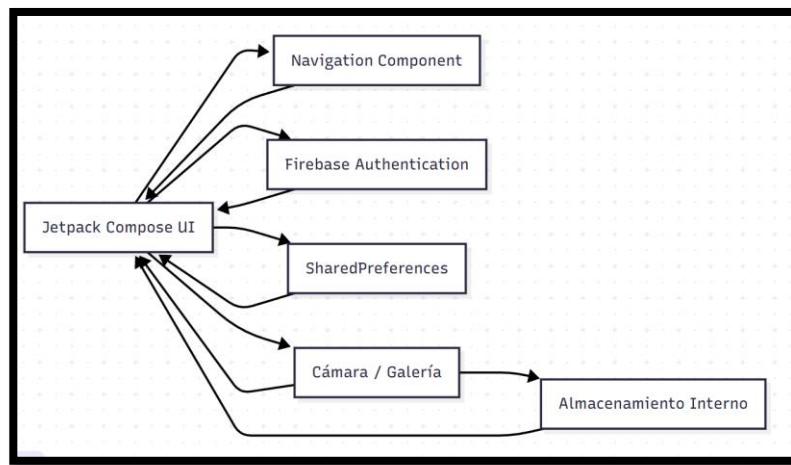


Ilustración 11: Diagrama de componentes

ARQUITECTURA

El diagrama de navegación funcional describe el recorrido del usuario entre las distintas pantallas de la aplicación. Este diagrama permite identificar las transiciones entre vistas, los eventos asociados a botones y las opciones disponibles según el estado del usuario, contribuyendo a una experiencia de uso clara y estructurada.

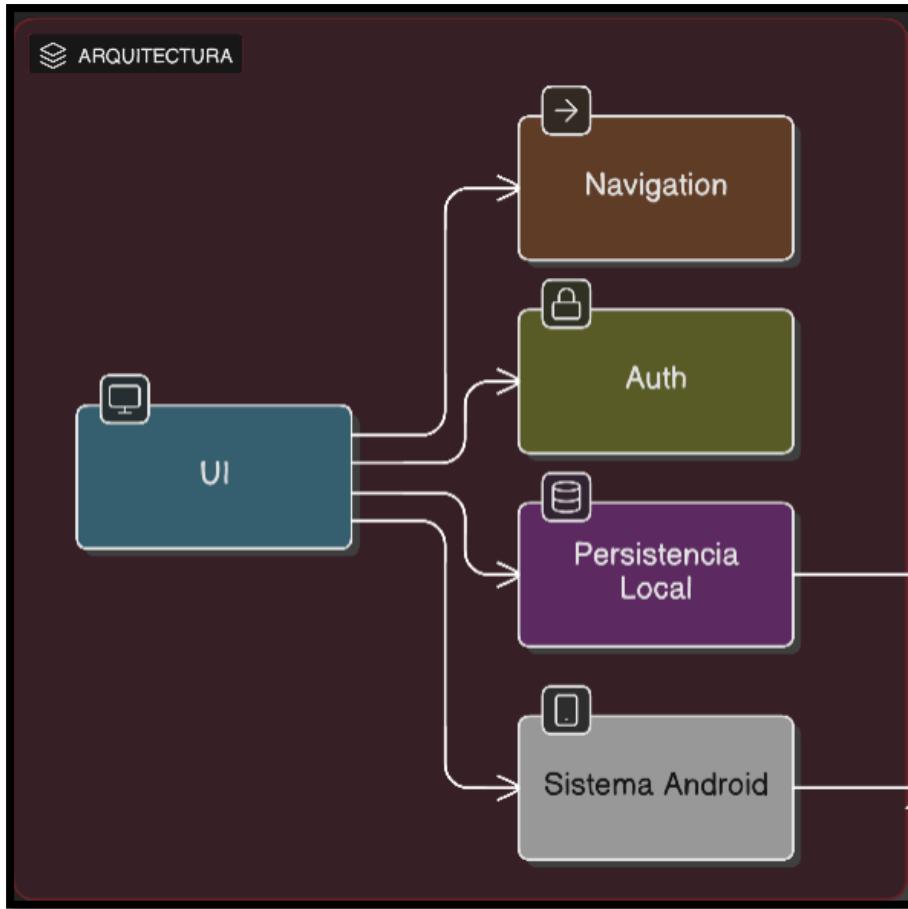


Ilustración 12: Arquitectura

ARQUITECTURA DE LA ESTRUCTURA DEL PROYECTO

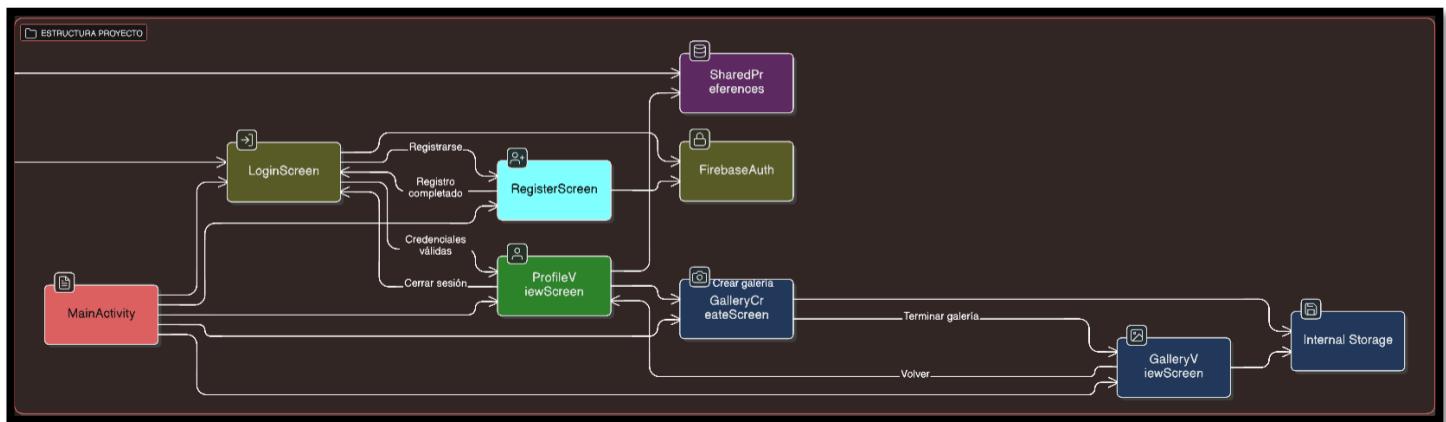


Ilustración 13: Estructura del proyecto

ARQUITECTURA DE GALERIA

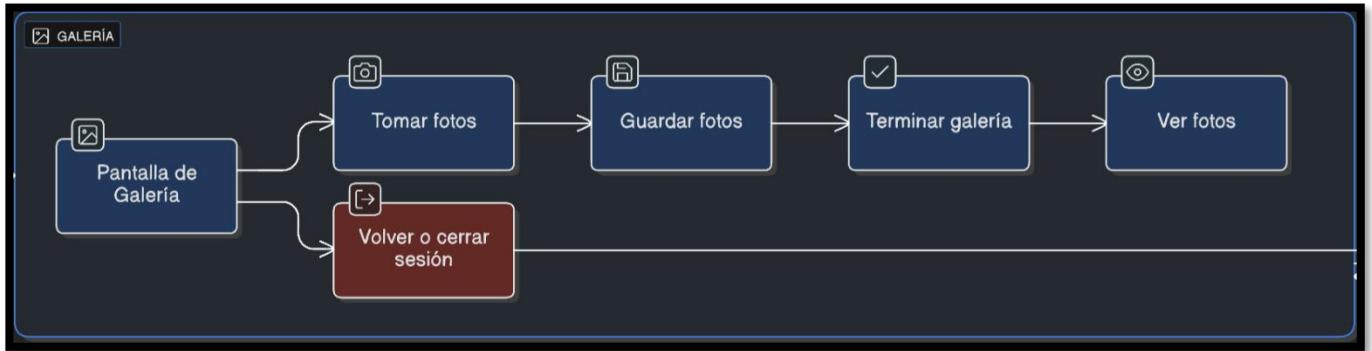


Ilustración 14: Arquitectura de Galeria

LOGIN

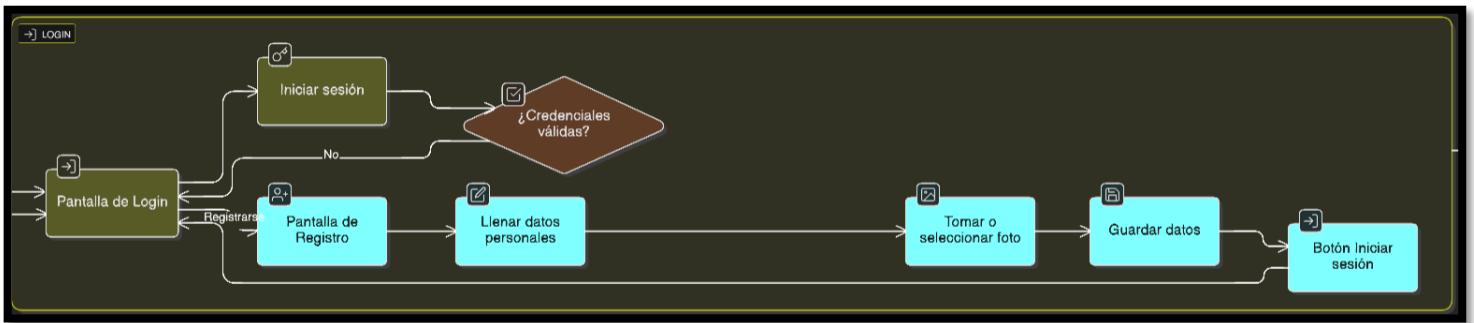
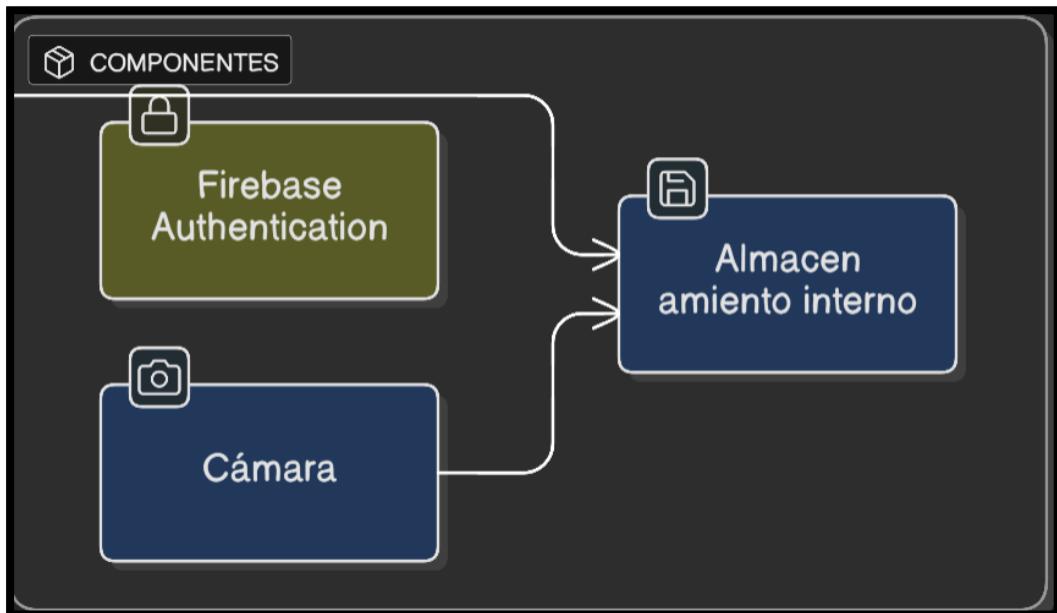


Ilustración 15: Login

ARQUITECTURA DE COMPONENTES



USUARIO LOGEADO

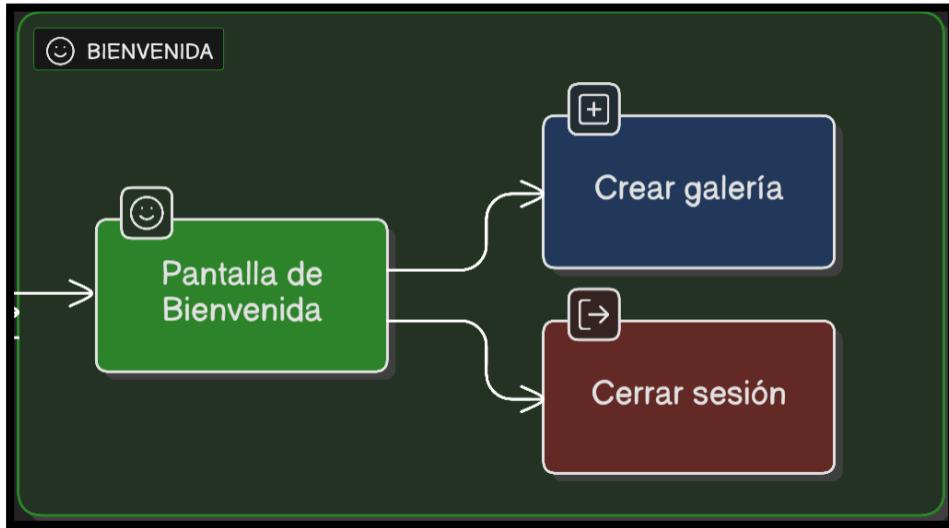
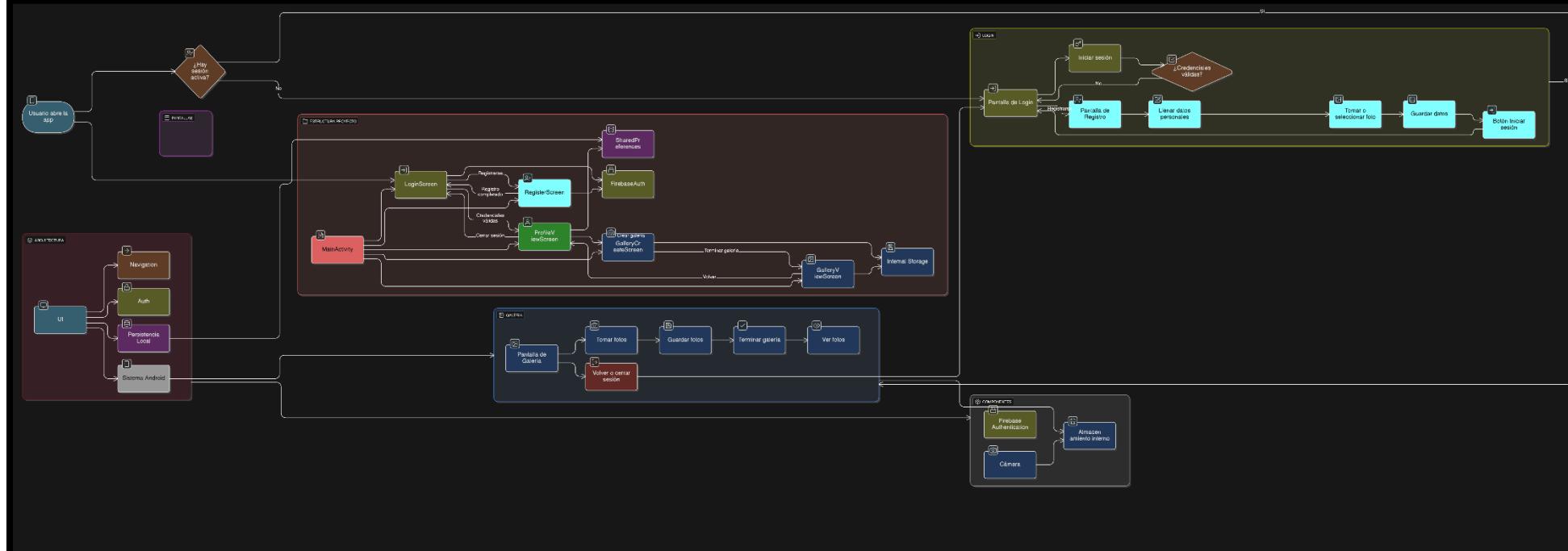


Ilustración 16: Usuario logeado

DIAGRAMA DE ARQUITECTURA COMPLETO



RESULTADOS OBTENIDOS

Durante el desarrollo de la aplicación móvil se obtuvieron resultados satisfactorios que evidencian el correcto funcionamiento de las funcionalidades planteadas al inicio del proyecto.

En primer lugar, se logró implementar de manera exitosa un sistema de autenticación de usuarios mediante Firebase Authentication, permitiendo el registro e inicio de sesión con correo electrónico y contraseña, garantizando un acceso seguro y controlado a la aplicación.

En segundo lugar, se integró correctamente el uso del sensor de la cámara del dispositivo, permitiendo al usuario capturar imágenes en tiempo real o seleccionarlas desde la galería del teléfono. Estas imágenes se utilizan tanto para la personalización del perfil como para la creación de una galería de fotos.

Un resultado relevante del proyecto es la implementación de una galería de imágenes persistente, en la cual las fotografías capturadas por el usuario se almacenan localmente y pueden ser visualizadas posteriormente sin perderse al cambiar de pantalla. Esto demuestra un manejo adecuado del almacenamiento interno y de la gestión de recursos multimedia.

Asimismo, se desarrolló una interfaz gráfica clara e intuitiva, facilitando la navegación entre pantallas como login, registro, perfil, captura de imágenes y visualización de la galería. El uso de Jetpack Compose permitió una construcción moderna y eficiente de las interfaces.

Finalmente, los diagramas de flujo, navegación y componentes permitieron validar visualmente la arquitectura del sistema y el flujo lógico de la aplicación, confirmando que el diseño propuesto fue implementado de manera correcta.

LOGINSCREEN – Interfaz de Autenticación

La pantalla LoginScreen es la primera interfaz con la que interactúa el usuario. Presenta un diseño simple y funcional, compuesto por:

- Un campo de texto para correo electrónico.

- Un campo de texto para contraseña.
- Un botón principal para iniciar sesión.

Función de los Componentes

Campo Correo Electrónico: Permite al usuario ingresar su email registrado.

Campo Contraseña: Permite ingresar la clave asociada a la cuenta.

Botón Iniciar Sesión: Ejecuta el proceso de autenticación en Firebase.



Ilustración 13: Login Scree

PROFILE SCREEN – Interfaz de Perfil de Usuario

La Profile Screen muestra la información asociada a la cuenta del usuario. Su diseño está orientado a la organización clara de los datos personales.

Función de los Componentes

Datos del Usuario: Muestran información obtenida desde Firebase.

Estructura Visual Clara: Facilita la lectura y comprensión de la información.

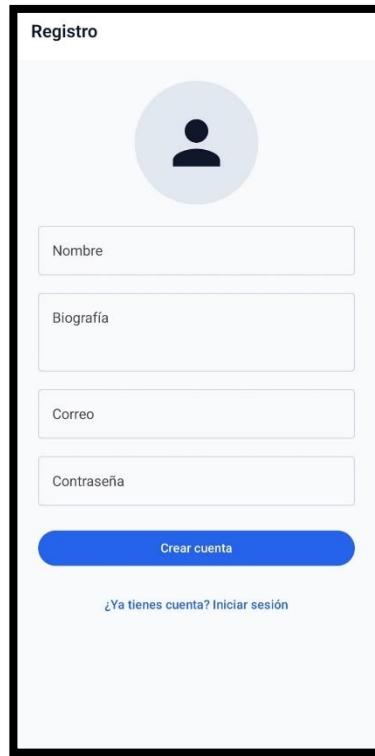


Ilustración 14: Perfil

Pantalla de Bienvenida y Perfil del Usuario

Una vez completado correctamente el proceso de registro e inicio de sesión, el sistema redirige al usuario a la pantalla de bienvenida, la cual confirma visualmente que la autenticación

se ha realizado de forma exitosa. En esta interfaz se muestran los datos registrados del usuario, permitiendo identificar de manera clara la información asociada a la cuenta.

En la parte superior de la pantalla se presenta un mensaje de bienvenida, acompañado de la imagen de perfil del usuario, la cual puede ser capturada mediante el sensor de la cámara del dispositivo o seleccionada desde la galería. Esta imagen contribuye a la personalización de la experiencia del usuario dentro de la aplicación.

Debajo de la fotografía se visualiza información relevante del perfil, como el nombre del usuario, la carrera que cursa, la institución educativa y el nivel académico, datos que fueron registrados previamente durante el proceso de autenticación o configuración del perfil.

Adicionalmente, la pantalla incluye un botón denominado “Crear galería”, cuya función es permitir al usuario gestionar contenido multimedia, sirviendo como punto de acceso para la creación o visualización de una galería de imágenes. Finalmente, se incorpora la opción “Cerrar sesión”, que permite finalizar de forma segura la sesión activa y regresar a la pantalla de inicio de sesión.

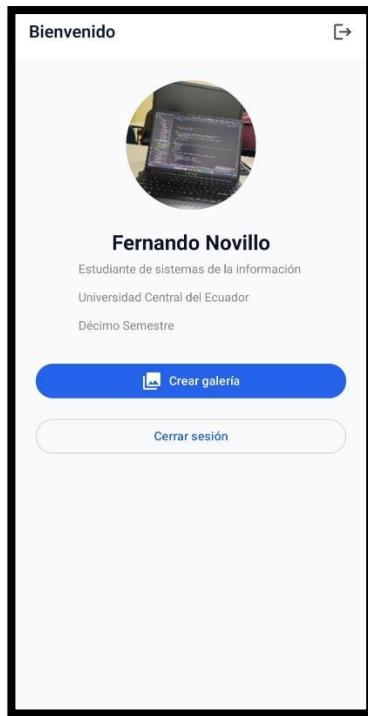


Ilustración 15: Pantalla de bienvenido

Como se puede observar se puede actualizar la foto de perfil en el momento que nosotros deseamos, también contamos con campos obligatorios

GESTIÓN DE GALERÍA DE IMÁGENES

Una vez que el usuario ha completado el proceso de registro e inicio de sesión, la aplicación habilita la funcionalidad de creación y gestión de una galería de fotografías. Desde la pantalla de perfil, el usuario puede acceder a la opción *Crear galería*, la cual permite capturar imágenes utilizando la cámara del dispositivo.

Cada fotografía capturada es almacenada en el almacenamiento interno del dispositivo, garantizando que las imágenes no se pierdan mientras la aplicación permanezca instalada. La aplicación mantiene un registro de las fotografías tomadas, permitiendo que estas puedan ser visualizadas posteriormente sin necesidad de volver a capturarlas.

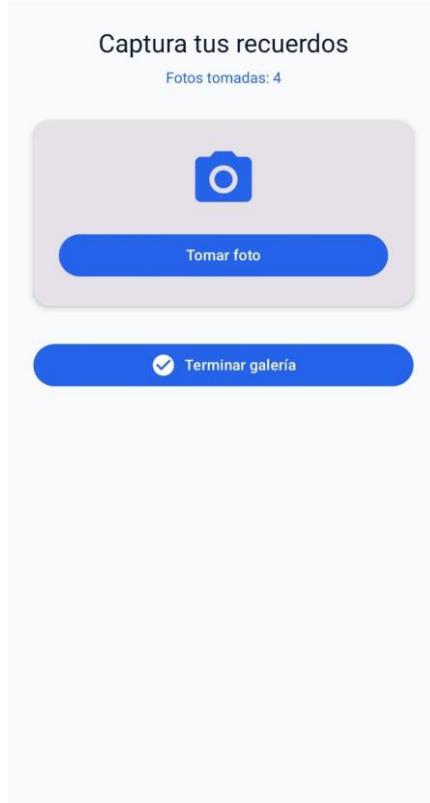
Crear Galería

Ilustración 16: Tomar fotos

La pantalla `GalleryCaptureScreen` gestiona la captura de imágenes y muestra la cantidad de fotos registradas. Posteriormente, la pantalla `GalleryViewScreen` presenta las imágenes en una grilla organizada, facilitando su visualización. Esta funcionalidad demuestra el manejo adecuado del sensor de cámara, la persistencia de datos y la gestión de contenido multimedia.

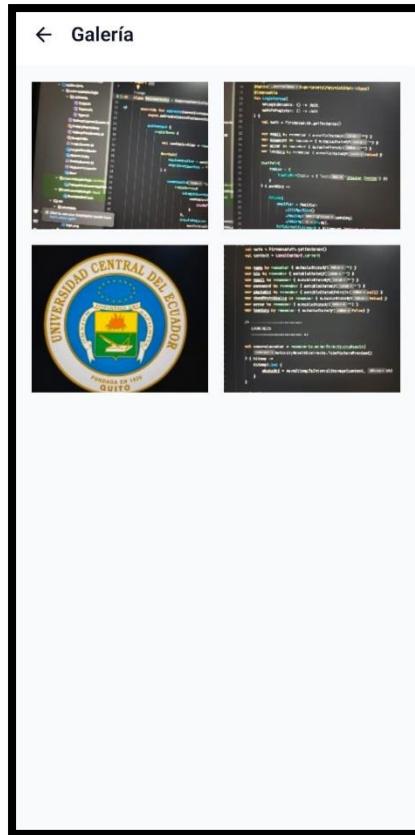


Ilustración 17: Galería de fotos

Conclusiones

El desarrollo de la aplicación móvil permitió aplicar de manera práctica los fundamentos del desarrollo Android, integrando componentes de interfaz gráfica, lógica de negocio y servicios en la nube dentro de un mismo proyecto funcional.

La implementación de Firebase Authentication demostró ser una solución eficiente y segura para la gestión de usuarios, facilitando el registro e inicio de sesión mediante correo electrónico y contraseña sin necesidad de implementar mecanismos de seguridad complejos desde cero.

La integración del sensor de la cámara del dispositivo evidenció la capacidad de Android para interactuar con el hardware del equipo, permitiendo enriquecer la experiencia del usuario mediante la captura y selección de imágenes para la personalización del perfil.

La estructura del proyecto, basada en la separación entre lógica y presentación, contribuye a la mantenibilidad del código y establece una base sólida para la escalabilidad de la aplicación en futuras versiones.

Recomendaciones

Integrar Firebase Storage para almacenar de forma persistente las imágenes capturadas o seleccionadas por los usuarios, garantizando disponibilidad y respaldo de la información.

Implementar mecanismos adicionales de seguridad, como verificación de correo electrónico, recuperación de contraseña y cierre de sesión, para fortalecer la protección de las cuentas de usuario.

Mejorar el diseño visual de la aplicación aplicando los lineamientos de Material Design, con el fin de ofrecer una interfaz más atractiva, accesible y consistente en distintos dispositivos.

Optimizar la gestión de permisos del sistema, mostrando mensajes claros al usuario sobre el uso de la cámara y la galería, y cumpliendo con las buenas prácticas de privacidad.