



Foodie

Foodie
Final Project

Rahul Kadam and Vinutha Karanth

Table of Contents

Basic Features

1. Splash Screen	2
2. Navigation Drawer	3
3. Recycler View.....	4
4. View Pager	7
5. Toolbar, Menus, Floating Action Button, and Coordinator Layout	10
6. Multiple Fragments and Activities	14
7. User/App State.....	21
8. Multiple Devices/Screens & Orientation Changes using Constraint Layout.....	23
9. Host your data in the Cloud	26

Advanced Features

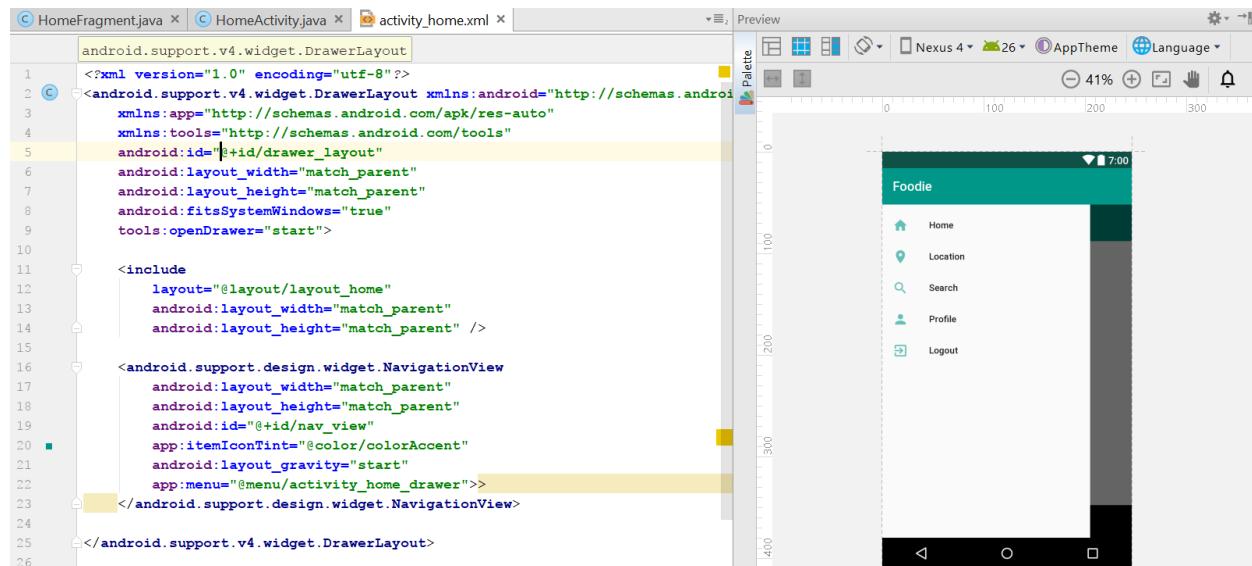
10. Media	30
10.1 Camera.....	30
10.2 Gallery.....	33
11. Location.....	36
11.1 Map	36
11.2 GPS	38
12. Advanced Animation - Flipbook Style	39
13. Data	41
13.1 Real API Data and Retrofit Library	41
14. Firebase Security	42
15. System - Service	43
16. Misc.....	47
16.1 Shortcuts	47
16.2 Multi-Window	49

1. Splash Screen



```
⑥ SplashActivity.java x
  ↗ SplashActivity
1 package com.connect.foodies.foodies.home;
2
3 import ...
4
5
6 public class SplashActivity extends AppCompatActivity{
7
8     private final int SPLASH_DISPLAY_LENGTH = 3000;
9
10    public void onCreate(Bundle icicle) {
11        super.onCreate(icicle);
12        setContentView(R.layout.activity_splash_screen);
13
14        /* New Handler to start the Menu-Activity
15         * and close this Splash-Screen after some seconds.*/
16        new Handler().postDelayed(() -> {
17            /* Create an Intent that will start the Menu-Activity. */
18            Intent mainIntent = new Intent(SplashActivity.this,HomeActivity.class);
19            SplashActivity.this.startActivity(mainIntent);
20            SplashActivity.this.finish();
21        }, SPLASH_DISPLAY_LENGTH);
22    }
23}
```

2. Navigation Drawer



```

107     }
108     });
109
110     DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
111     ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
112         this, drawer, toolbar, "Open navigation drawer", "Close navigation drawer");
113     drawer.setDrawerListener(toggle);
114     toggle.syncState();

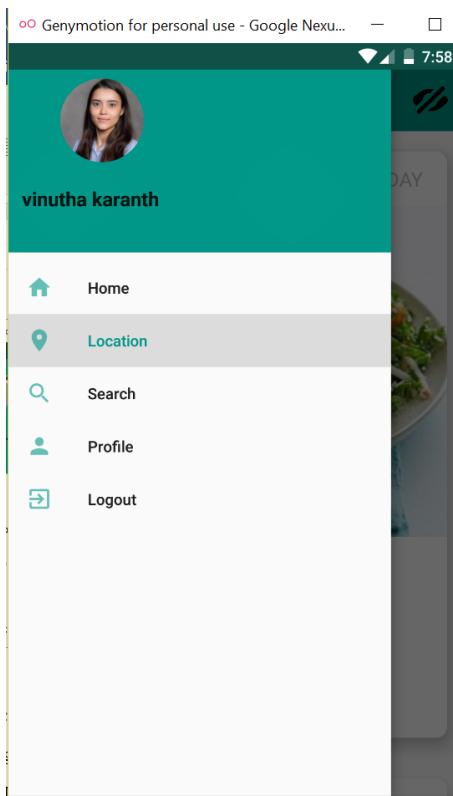
```



```

238     public boolean onNavigationItemSelected(MenuItem item) {
239         // Handle navigation view item clicks here.
240         int id = item.getItemId();
241         if (id == R.id.nav_home) {
242             intent = new Intent(HomeActivity.this, HomeActivity.class);
243             startActivity(intent);
244
245         } else if (id == R.id.nav_location) {
246             intent = new Intent(HomeActivity.this, MapsActivity.class);
247             startActivity(intent);
248
249         } else if (id == R.id.nav_search) {
250             intent = new Intent(HomeActivity.this, SearchActivity.class);
251             startActivity(intent);
252
253         } else if (id == R.id.nav_profile) {
254             intent = new Intent(HomeActivity.this, ProfileActivity.class);
255             startActivity(intent);
256
257         } else if (id == R.id.nav_logout) {
258             Log.d(TAG, "onClick: attempting to sign out.");
259             mAuth.signOut();
260             finish();
261         }
262         DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
263         drawer.closeDrawer(GravityCompat.START);
264         return true;
265     }

```



3. Recycler View

Recycler View is used in Home Fragment, MainFeedListAdapter file.

```
HomeFragment.java
fragment_home.xml
```

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <android.support.v7.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/recyclerview"
        app:layout_behavior="android.support.design.widget.AppBarLayout$Scro
```

Foodie



```
① HomeFragment.java x fragment_home.xml x
  HomeFragment|onCreateView()
52
53     @Override
54     public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
55         View view = inflater.inflate(R.layout.fragment_home, container, false);
56         mRecyclerView = (RecyclerView) view.findViewById(R.id.recyclerview);
57         mFollowing = new ArrayList<>();
58         mPhotos = new ArrayList<>();

② HomeFragment.java x fragment_home.xml x
  HomeFragment|displayPhotos()
141     private void displayPhotos(){
142         mPaginatedPhotos = new ArrayList<>();
143         if(mPhotos != null){
144             try{
145                 Collections.sort(mPhotos, (Comparator) (o1, o2) -> {
146                     return o2.getDate_created().compareTo(o1.getDate_created());
147                 });
148                 int iterations = mPhotos.size();
149
150                 if(iterations > 10){
151                     iterations = 10;
152                 }
153                 mResults = 10;
154                 for(int i = 0; i < iterations; i++){
155                     mPaginatedPhotos.add(mPhotos.get(i));
156                 }
157                 mAdapter = new MainfeedListAdapter(getActivity(), mPaginatedPhotos);
158                 mRecyclerView.setAdapter(mAdapter);
159                 mRecyclerView.setNestedScrollingEnabled(false);
160                 mRecyclerView.setHasFixedSize(true);
161
162                 LinearLayoutManager layoutManager = new LinearLayoutManager(getContext());
163                 mRecyclerView.setLayoutManager(layoutManager);
164
165                 //Animations
166                 AlphaInAnimationAdapter alphaAdapter = new AlphaInAnimationAdapter(mAdapter);
167                 ScaleInAnimationAdapter scaleAdapter = new ScaleInAnimationAdapter(alphaAdapter);
168                 scaleAdapter.setDuration(1000);
169                 mRecyclerView.setAdapter(scaleAdapter);
170
171
172 }
```

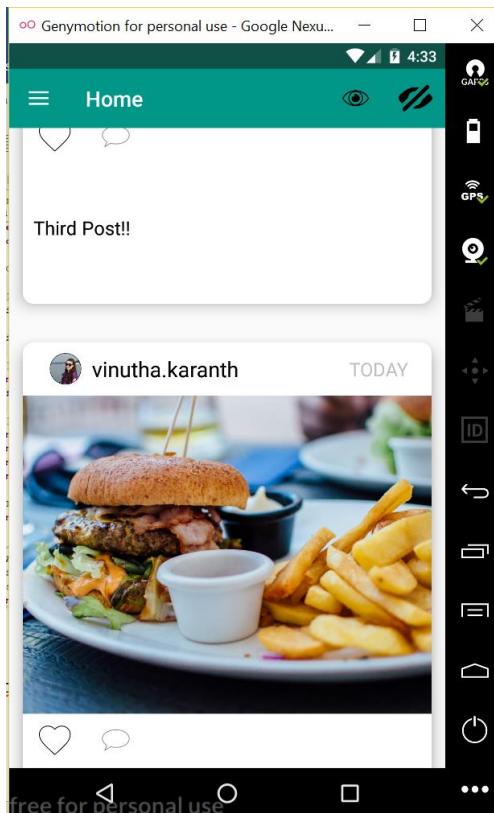
MainFeedListAdapter

```
① HomeFragment.java x MainfeedListAdapter.java x fragment_home.xml x
  MainfeedListAdapter|MyViewHolder
43
44     public class MainfeedListAdapter extends RecyclerView.Adapter<MainfeedListAdapter.MyViewHolder> {
45
46         private static final String TAG = "MainfeedListAdapter";
47         OnLoadMoreItemsListener mOnLoadMoreItemsListener;
48         private Context mContext;
49         private DatabaseReference mReference;
50         private String currentUsername = "";
51         private List<Photo> mListPhoto;
```

Foodie

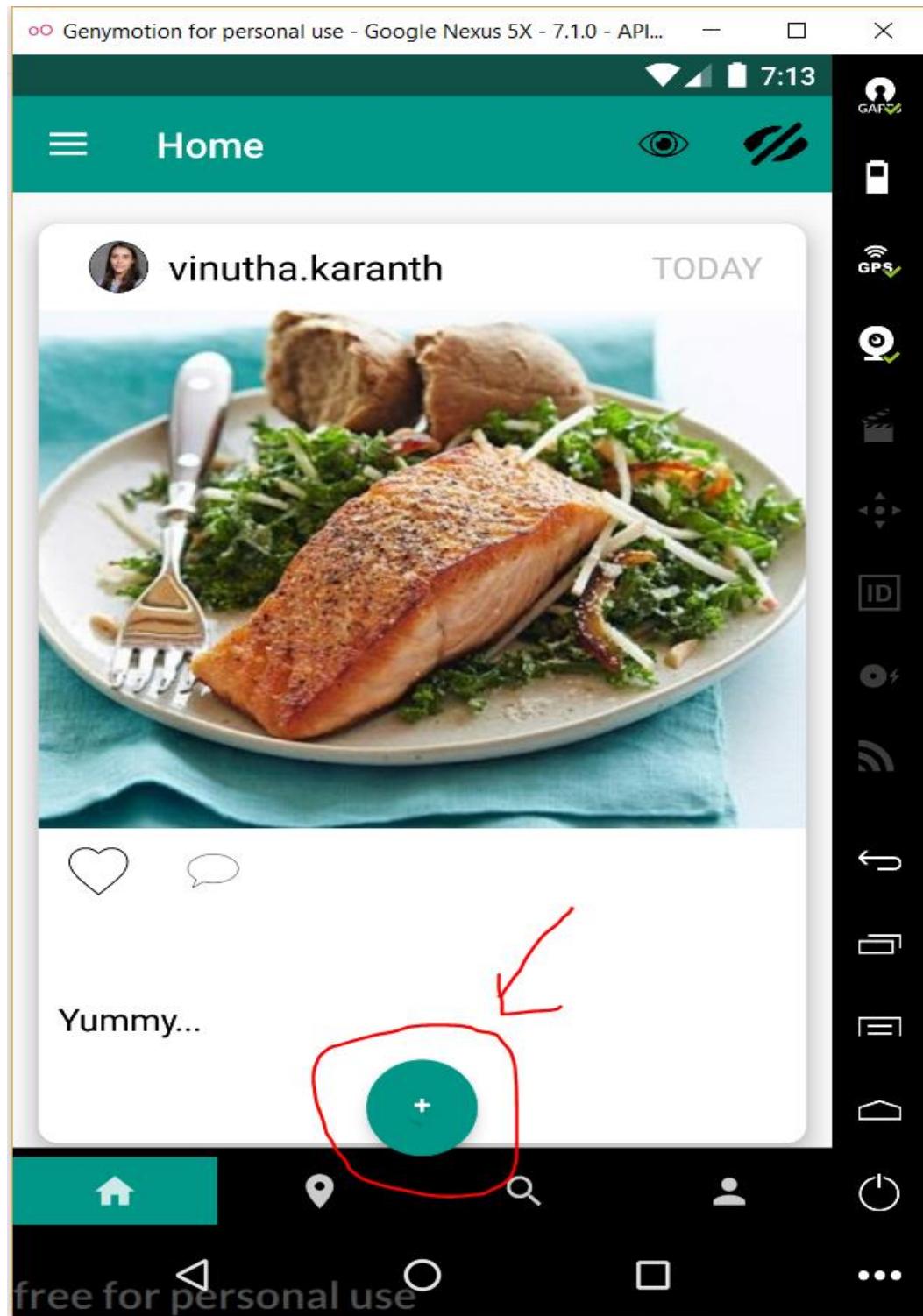


```
C HomeFragment.java x C MainfeedListAdapter.java x fragment_home.xml x
MainfeedListAdapter
438     static public class MyViewHolder extends RecyclerView.ViewHolder {
439         CircleImageView mprofileImage;
440         String likesString;
441         TextView username, timeDetail, caption, likes, comments;
442         ImageView image;
443         ImageView heartRed, heartWhite, comment;
444
445         UserAccountSettings settings = new UserAccountSettings();
446         User user = new User();
447         StringBuilder users;
448         boolean likeByCurrentUser;
449         Heart heart;
450         GestureDetector detector;
451         Photo photo;
452
453         public MyViewHolder(View itemView) {
454             super(itemView);
455
456             username = (TextView) itemView.findViewById(R.id.username);
457             image = (ImageView) itemView.findViewById(R.id.post_image);
458             heartRed = (ImageView) itemView.findViewById(R.id.image_heart_red);
```



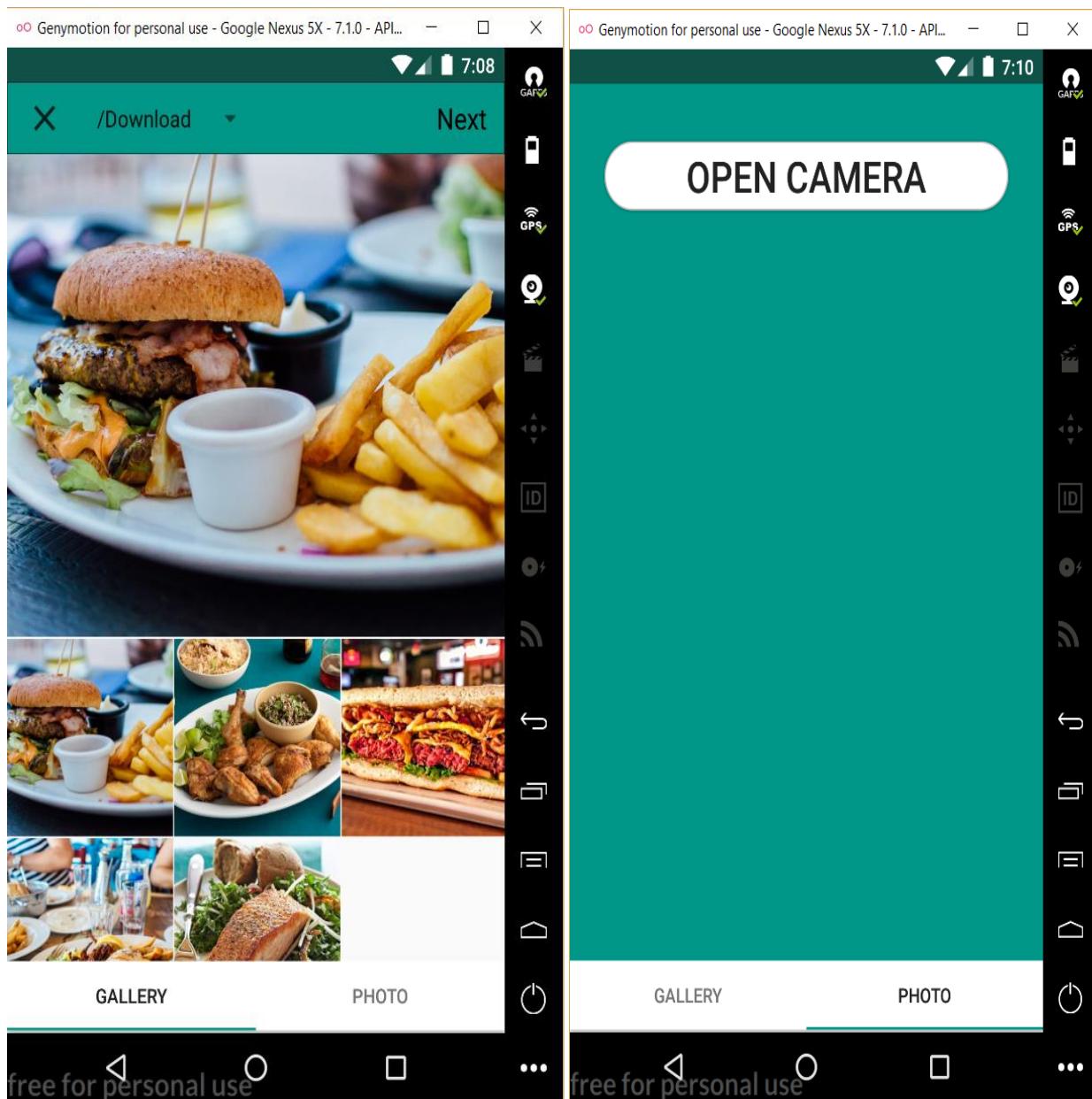
4. View Pager

View Pager is used we click green share floating button highlighted below:



This View Pager has two parts “Gallery” and “Photo” as show below:

Foodie



Code is shown on next page

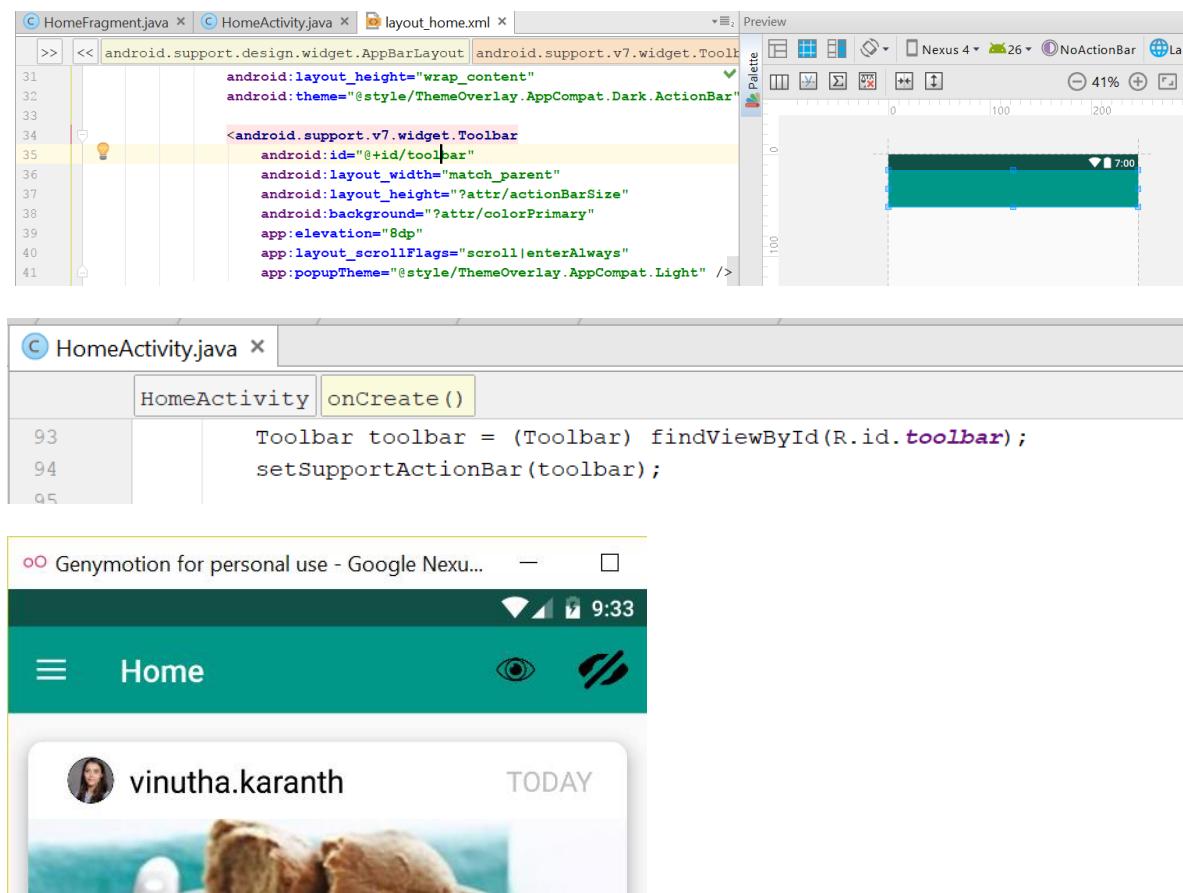
ShareActivity.java contains this View Pager:

```

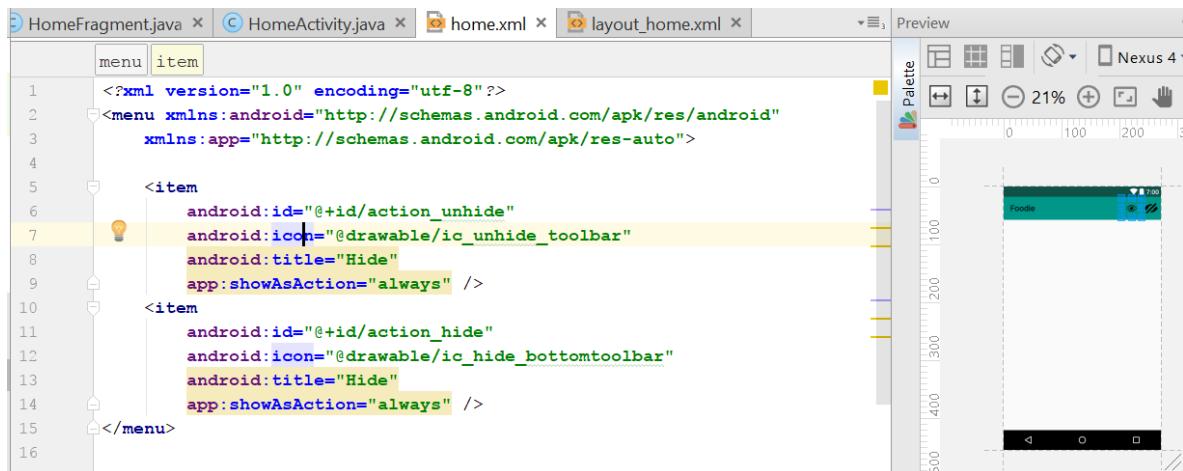
ShareActivity.java x ShareActivity
1 package com.connect.foodies.foodies.share;
2
3 import ...
28
29 public class ShareActivity extends AppCompatActivity {
30     private static final String TAG = "ShareActivity";
31
32     //constants
33     private static final int ACTIVITY_NUM = 2;
34     private static final int VERIFY_PERMISSIONS_REQUEST = 1;
35
36     private ViewPager mViewPager;
37     private TextView mTextView;
38     private TextView mTextView3;
39     private Context mContext = ShareActivity.this;
40
41     @Override
42     protected void onCreate(@Nullable Bundle savedInstanceState) {
43         super.onCreate(savedInstanceState);
44         setContentView(R.layout.activity_share);
45         Log.d(TAG, "onCreate: started.");
46
47         mTextView = (TextView) findViewById(R.id.textView);
48         mTextView3 = (TextView) findViewById(R.id.textView3);
49         if (checkPermissionsArray(Permissions.PERMISSIONS)) {
50             setupViewPager();
51         } else {
52             verifyPermissions(Permissions.PERMISSIONS);
53             mTextView.setText("Checking For Permissions...");
54             mTextView3.setText("Once Permissions Granted. Please Go Back and then Share");
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
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
7010
7011
7012
7013
7014
7015
7016
7017
7018
7019
7020
7021
7022
7023
7024
7025
7026
7027
7028
7029
70210
70211
70212
70213
70214
70215
70216
70217
70218
70219
70220
70221
70222
70223
70224
70225
70226
70227
70228
70229
70230
70231
70232
70233
70234
70235
70236
70237
70238
70239
702310
702311
702312
702313
702314
702315
702316
702317
702318
702319
702320
702321
702322
702323
702324
702325
702326
702327
702328
702329
702330
702331
702332
702333
702334
702335
702336
702337
702338
702339
702340
702341
702342
702343
702344
702345
702346
702347
702348
702349
702350
702351
702352
702353
702354
702355
702356
702357
702358
702359
702360
702361
702362
702363
702364
702365
702366
702367
702368
702369
702370
702371
702372
702373
702374
702375
702376
702377
702378
702379
702380
702381
702382
702383
702384
702385
702386
702387
702388
702389
7023810
7023811
7023812
7023813
7023814
7023815
7023816
7023817
7023818
7023819
70238110
70238111
70238112
70238113
70238114
70238115
70238116
70238117
70238118
70238119
702381110
702381111
702381112
702381113
702381114
702381115
702381116
702381117
702381118
702381119
7023811110
7023811111
7023811112
7023811113
7023811114
7023811115
7023811116
7023811117
7023811118
7023811119
70238111110
70238111111
70238111112
70238111113
70238111114
70238111115
70238111116
70238111117
70238111118
70238111119
702381111110
702381111111
702381111112
702381111113
702381111114
702381111115
702381111116
702381111117
702381111118
702381111119
7023811111110
7023811111111
7023811111112
7023811111113
7023811111114
7023811111115
7023811111116
7023811111117
7023811111118
7023811111119
70238111111110
70238111111111
70238111111112
70238111111113
70238111111114
70238111111115
70238111111116
70238111111117
70238111111118
70238111111119
702381111111110
702381111111111
702381111111112
702381111111113
702381111111114
702381111111115
702381111111116
702381111111117
702381111111118
702381111111119
7023811111111110
7023811111111111
7023811111111112
7023811111111113
7023811111111114
7023811111111115
7023811111111116
7023811111111117
7023811111111118
7023811111111119
70238111111111110
70238111111111111
70238111111111112
70238111111111113
70238111111111114
70238111111111115
70238111111111116
70238111111111117
70238111111111118
70238111111111119
702381111111111110
702381111111111111
702381111111111112
702381111111111113
702381111111111114
702381111111111115
702381111111111116
702381111111111117
702381111111111118
702381111111111119
7023811111111111110
7023811111111111111
7023811111111111112
7023811111111111113
7023811111111111114
7023811111111111115
7023811111111111116
7023811111111111117
7023811111111111118
7023811111111111119
70238111111111111110
70238111111111111111
70238111111111111112
70238111111111111113
70238111111111111114
70238111111111111115
70238111111111111116
70238111111111111117
70238111111111111118
70238111111111111119
702381111111111111110
702381111111111111111
702381111111111111112
702381111111111111113
702381111111111111114
702381111111111111115
702381111111111111116
702381111111111111117
702381111111111111118
702381111111111111119
7023811111111111111110
7023811111111111111111
7023811111111111111112
7023811111111111111113
7023811111111111111114
7023811111111111111115
7023811111111111111116
7023811111111111111117
7023811111111111111118
7023811111111111111119
70238111111111111111110
70238111111111111111111
70238111111111111111112
70238111111111111111113
70238111111111111111114
70238111111111111111115
70238111111111111111116
70238111111111111111117
70238111111111111111118
70238111111111111111119
702381111111111111111110
702381111111111111111111
702381111111111111111112
702381111111111111111113
702381111111111111111114
702381111111111111111115
702381111111111111111116
702381111111111111111117
702381111111111111111118
702381111111111111111119
7023811111111111111111110
7023811111111111111111111
7023811111111111111111112
7023811111111111111111113
7023811111111111111111114
7023811111111111111111115
7023811111111111111111116
7023811111111111111111117
7023811111111111111111118
7023811111111111111111119
70238111111111111111111110
70238111111111111111111111
70238111111111111111111112
70238111111111111111111113
70238111111111111111111114
70238111111111111111111115
70238111111111111111111116
70238111111111111111111117
70238111111111111111111118
70238111111111111111111119
702381111111111111111111110
702381111111111111111111111
702381111111111111111111112
702381111111111111111111113
702381111111111111111111114
702381111111111111111111115
702381111111111111111111116
702381111111111111111111117
702381111111111111111111118
702381111111111111111111119
7023811111111111111111111110
7023811111111111111111111111
7023811111111111111111111112
7023811111111111111111111113
7023811111111111111111111114
7023811111111111111111111115
7023811111111111111111111116
7023811111111111111111111117
7023811111111111111111111118
7023811111111111111111111119
70238111111111111111111111110
70238111111111111111111111111
70238111111111111111111111112
70238111111111111111111111113
70238111111111111111111111114
70238111111111111111111111115
70238111111111111111111111116
70238111111111111111111111117
70238111111111111111111111118
70238111111111111111111111119
702381111111111111111111111110
702381111111111111111111111111
702381111111111111111111111112
702381111111111111111111111113
702381111111111111111111111114
702381111111111111111111111115
702381111111111111111111111116
702381111111111111111111111117
702381111111111111111111111118
702381111111111111111111111119
7023811111111111111111111111110
7023811111111111111111111111111
7023811111111111111111111111112
7023811111111111111111111111113
7023811111111111111111111111114
7023811111111111111111111111115
7023811111111111111111111111116
7023811111111111111111111111117
7023811111111111111111111111118
7023811111111111111111111111119
70238111111111111111111111111110
70238111111111111111111111111111
70238111111111111111111111111112
70238111111111111111111111111113
70238111111111111111111111111114
70238111111111111111111111111115
70238111111111111111111111111116
70238111111111111111111111111117
70238111111111111111111111111118
70238111111111111111111111111119
702381111111111111111111111111110
7023811111111111111111111111111111
7023811111111111111111111111111112
7023811111111111111111111111111113
7023811111111111111111111111111114
7023811111111111111111111111111115
7023811111111111111111111111111116
7023811111111111111111111111111117
7023811111111111111111111111111118
7023811111111111111111111111111119
70238111111111111111111111111111110
70238111111111111111111111111111111
70238111111111111111111111111111112
70238111111111111111111111111111113
70238111111111111111111111111111114
70238111111111111111111111111111115
70238111111111111111111111111111116
70238111111111111111111111111111117
70238111111111111111111111111111118
70238111111111111111111111111111119
702381111111111111111111111111111110
702381111111111111111111111111111111
702381111111111111111111111111111112
702381111111111111111111111111111113
702381111111111111111111111111111114
702381111111111111111111111111111115
702381111111111111111111111111111116
702381111111111111111111111111111117
702381111111111111111111111111111118
702381111111111111111111111111111119
7023811111111111111111111111111111110
7023811111111111111111111111111111111
7023811111111111111111111111111111112
7023811111111111111111111111111111113
7023811111111111111111111111111111114
7023811111111111111111111111111111115
7023811111111111111111111111111111116
7023811111111111111111111111111111117
7023811111111111111111111111111111118
7023811111111111111111111111111111119
70238111111111111111111111111111111110
70238111111111111111111111111111111111
70238111111111111111111111111111111112
70238111111111111111111111111111111113
70238111111111111111111111111111111114
70238111111111111111111111111111111115
70238111111111111111111111111111111116
70238111111111111111111111111111111117
70238111111111111111111111111111111118
70238111111111111111111111111111111119
702381111111111111111111111111111111110
702381111111111111111111111111111111111
702381111111111111111111111111111111112
702381111111111111111111111111111111113
702381111111111111111111111111111111114
702381111111111111111111111111111111115
702381111111111111111111111111111111116
702381111111111111111111111111111111117
702381111111111111111111111111111111118
702381111111111111111111111111111111119
702381111111111111111
```

5. Toolbar, Menus, Floating Action Button, and Coordinator Layout

A. Tools



B. Menus



```

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.home, menu);
    return true;
}

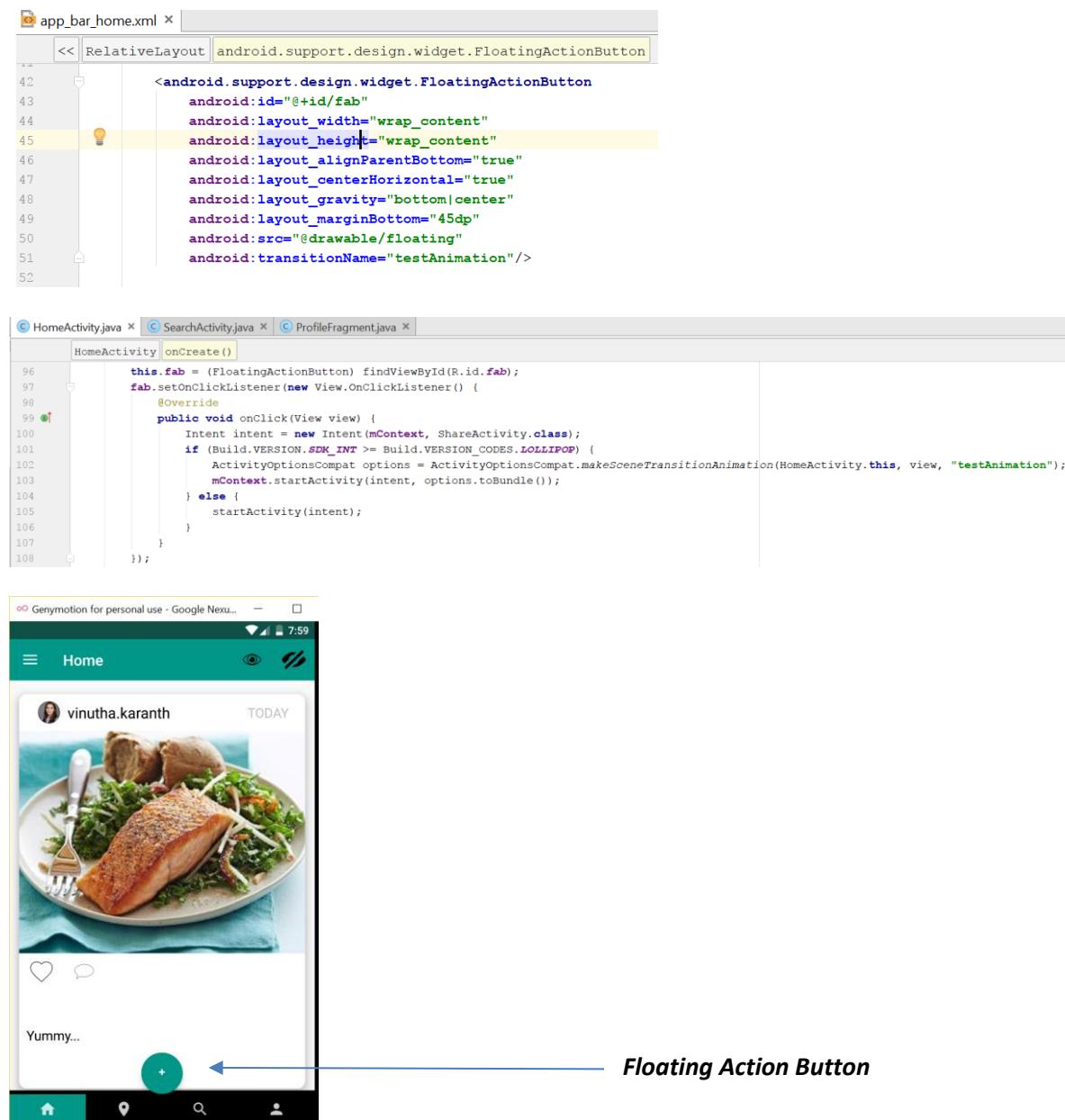
```

```

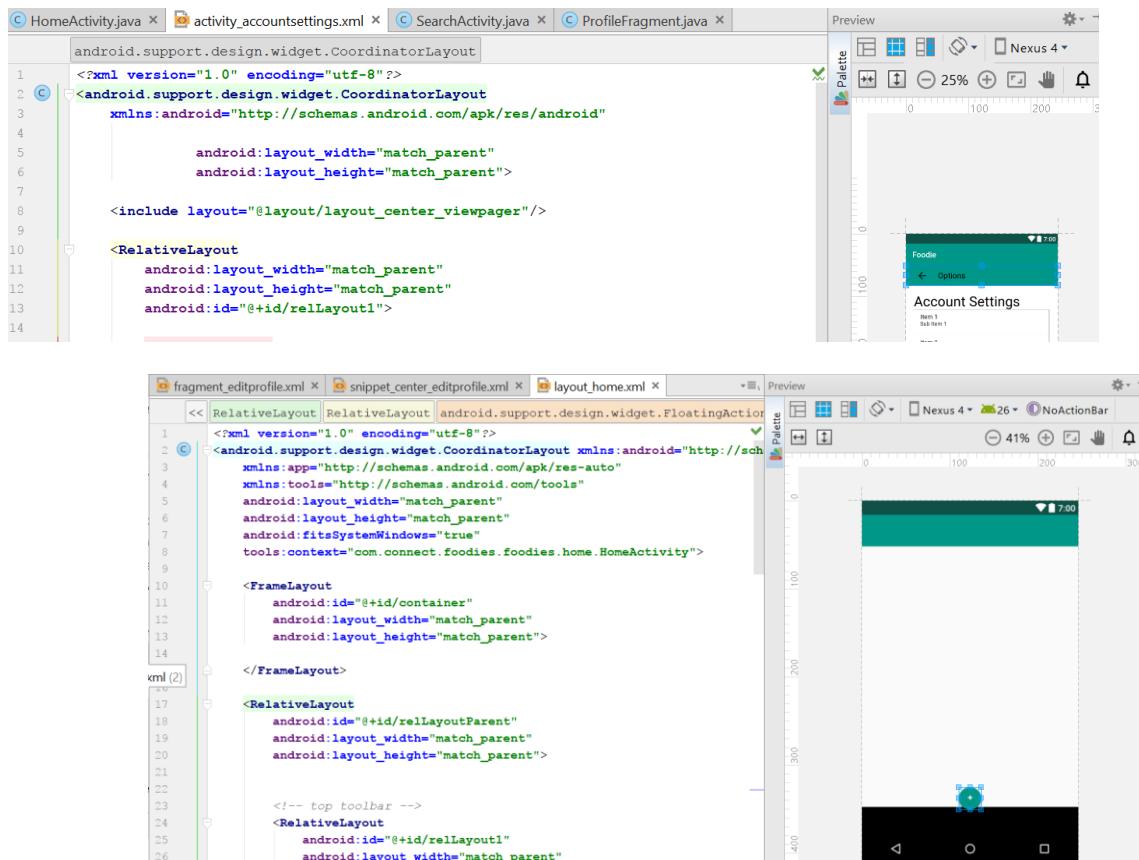
public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();
    //noinspection SimplifiableIfStatement
    if (id == R.id.action_hide) {
        bottomNavigationViewEx.setVisibility(View.GONE);
        fab.setVisibility(View.GONE);
        return true;
    }
    else if (id == R.id.action_unhide)
    {
        bottomNavigationViewEx.setVisibility(View.VISIBLE);
        fab.setVisibility(View.VISIBLE);
        return true;
    }
    return super.onOptionsItemSelected(item);
}

```

C. Floating action can be done in 3 activities – Home, profile and search



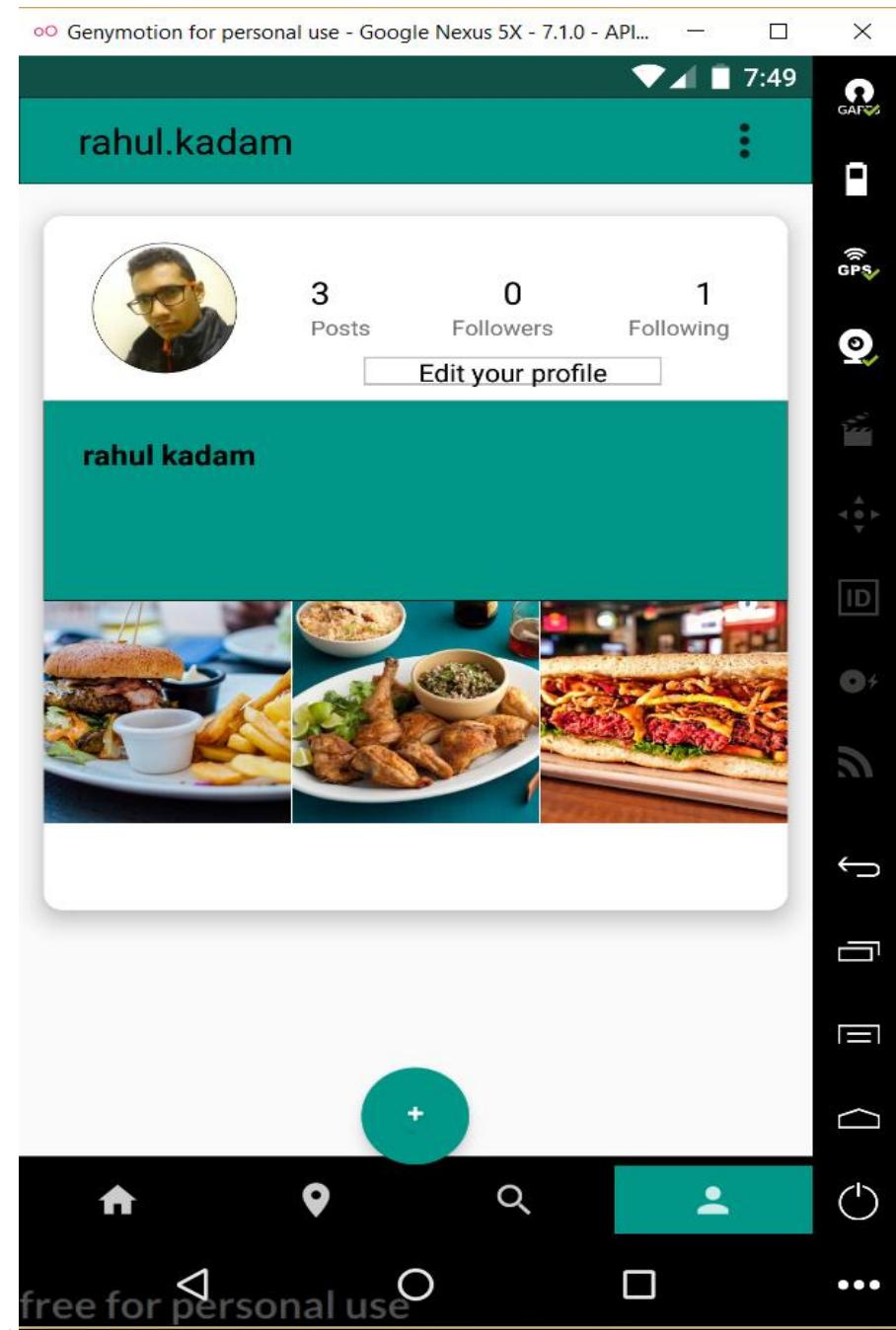
D. Coordinator layout is used in activity_accountsettings, activity_profile, activity_share, activity_bar_home, fragment_editprofile and layout_home. Floating action button and tool bar works perfectly without any disruption.



6. Multiple Fragments and Activities

One of way of using Fragments and Activities is for User Profile feature:

ProfileActivity.java first loads ProfileFragment.java

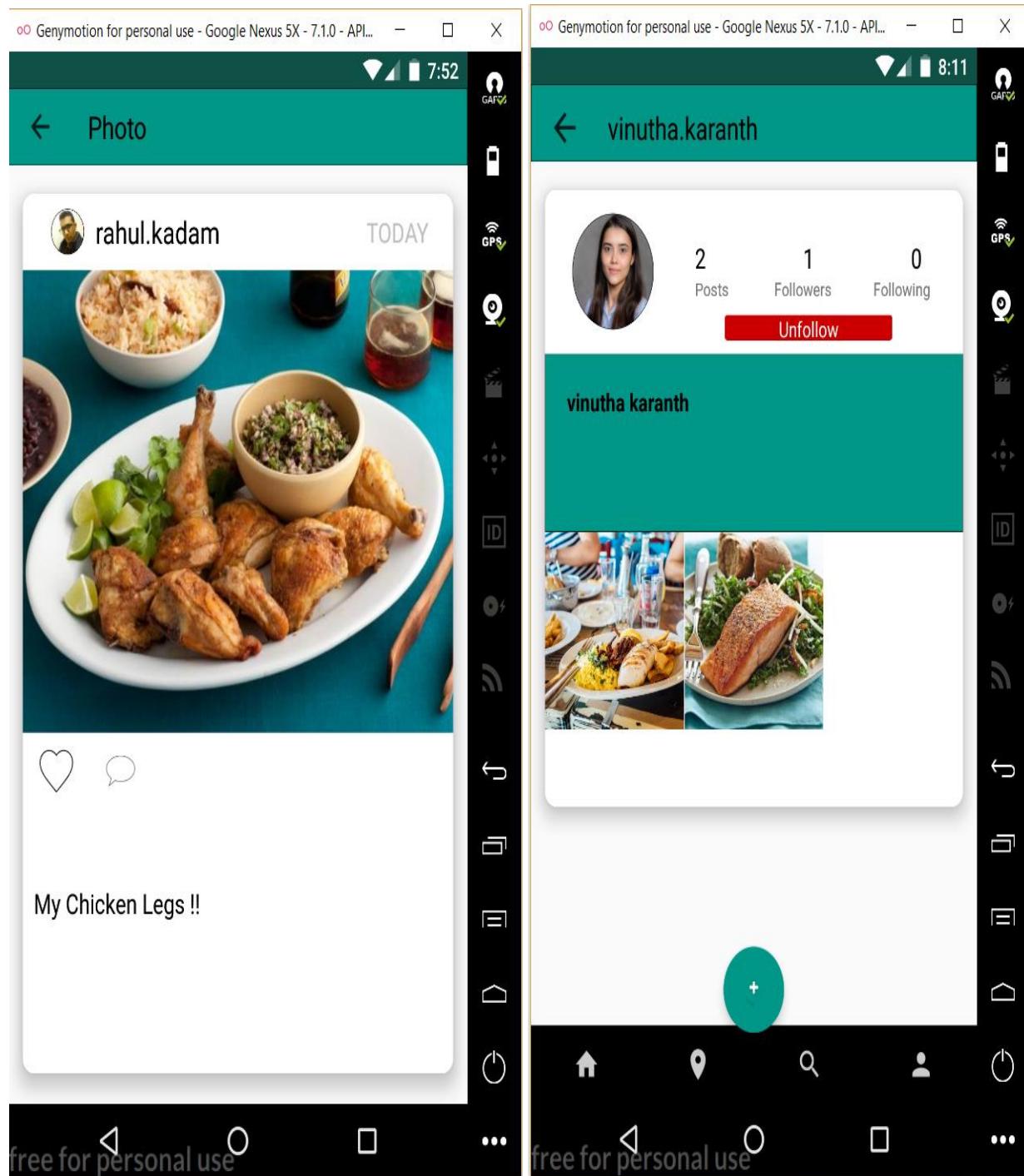


ProfileActivity.java acts as base activity and 3 fragments:

1. ProfileFragment.java (previous page)

2. ViewPostFragment.java

3. ViewProfileFragment.java



Foodie

Code: ProfileActivity.java

```
ProfileActivity
package com.connect.foodies.profile;

import ...

public class ProfileActivity extends AppCompatActivity implements
    ProfileFragment.OnGridImageSelectedListener ,
    ViewPostFragment.OnCommentThreadSelectedListener,
    ViewProfileFragment.OnGridImageSelectedListener {

    private static final String TAG = "ProfileActivity";
    private static final int ACTIVITY_NUM = 3;
    private static final int NUM_GRID_COLUMNS = 3;

    private Context mContext = ProfileActivity.this;
```

Code: ProfileFragment.java

```
ProfileFragment
package com.connect.foodies.profile;

import ...

public class ProfileFragment extends Fragment {

    private static final String TAG = "ProfileFragment";

    public interface OnGridImageSelectedListener {
        void onGridImageSelected(Photo photo, int activityNumber);
    }
    OnGridImageSelectedListener mOnGridImageSelectedListener;
```

Code: ViewPostFragment.java

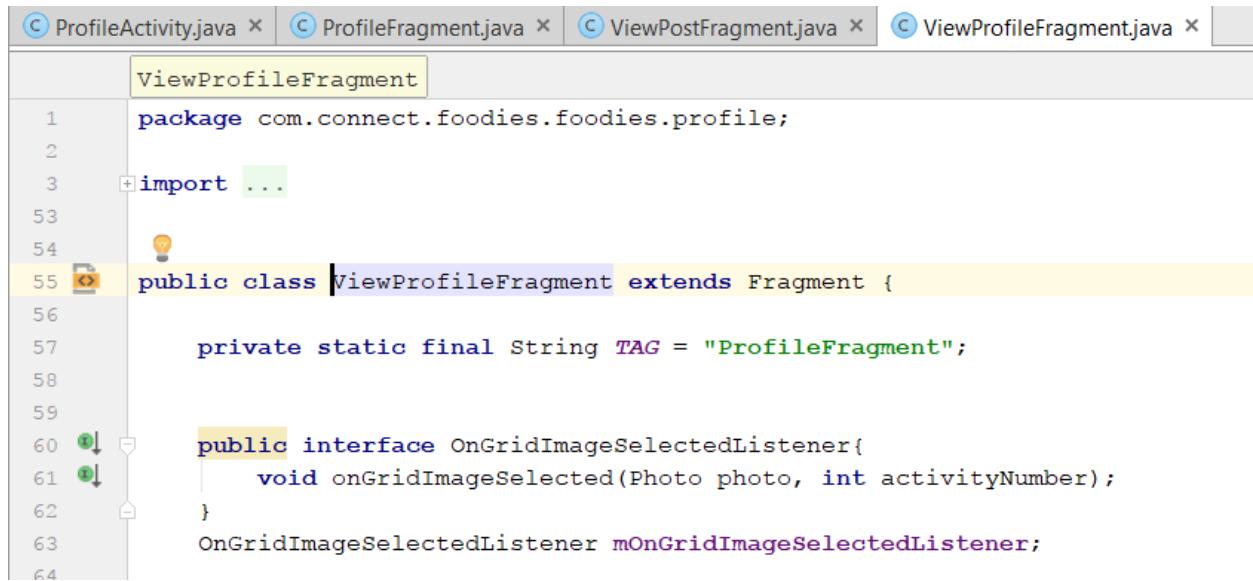
```
ViewPostFragment
package com.connect.foodies.share;

import ...

public class ViewPostFragment extends Fragment {
    private static final String TAG = "ViewPostFragment";

    public interface OnCommentThreadSelectedListener{
        void onCommentThreadSelectedListener(Photo photo);
    }
    OnCommentThreadSelectedListener mOnCommentThreadSelectedListener;
```

Code: ViewProfileFragment.java



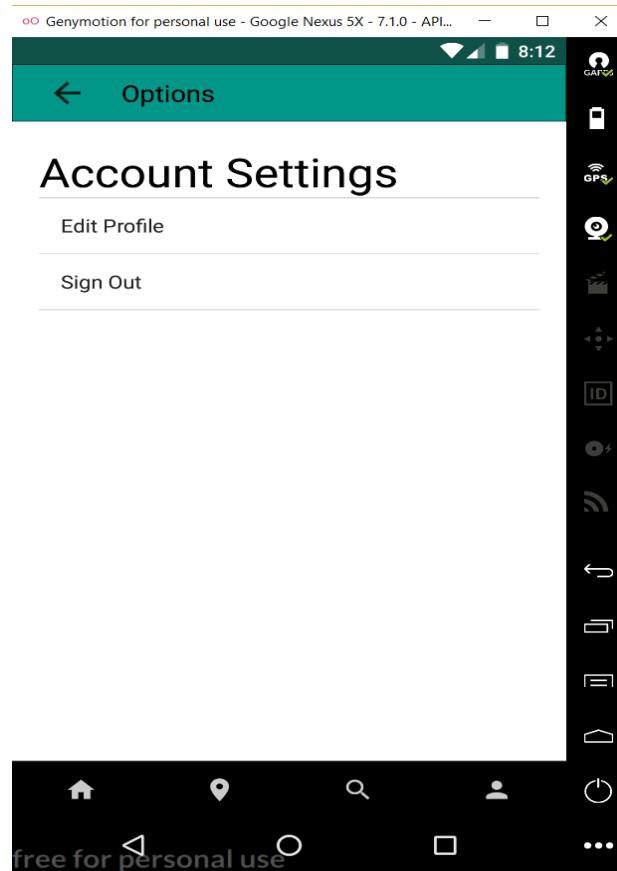
```

1 package com.connect.foodies.profile;
2
3 import ...
53
54
55 public class ViewProfileFragment extends Fragment {
56
57     private static final String TAG = "ProfileFragment";
58
59
60     public interface OnGridImageSelectedListener{
61         void onGridImageSelected(Photo photo, int activityNumber);
62     }
63
64     OnGridImageSelectedListener mOnGridImageSelectedListener;

```

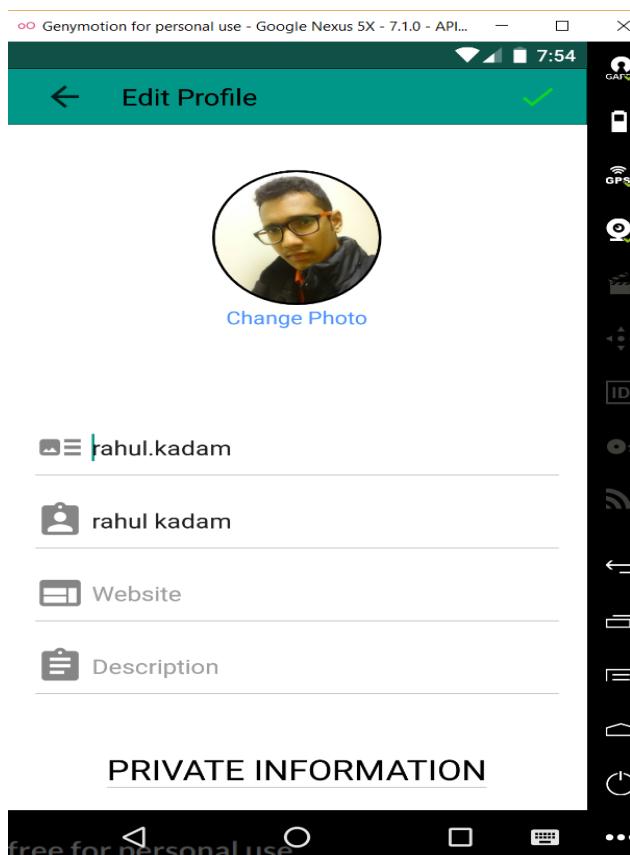
AccountSettingsActivity.java acts as base activity for: EditProfileFragement.java and SignOutFragment.java

AccountSettingsActivity.java:

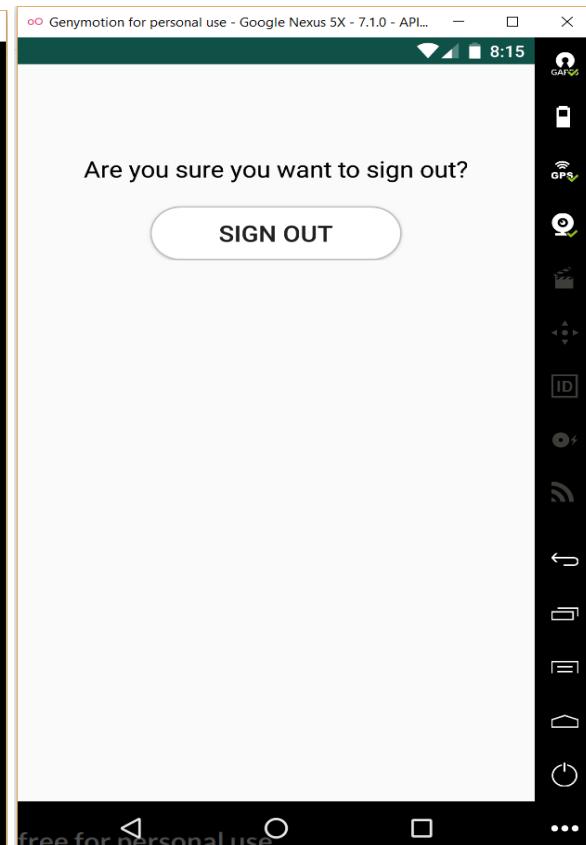


Foodie

EditProfileFragment.java



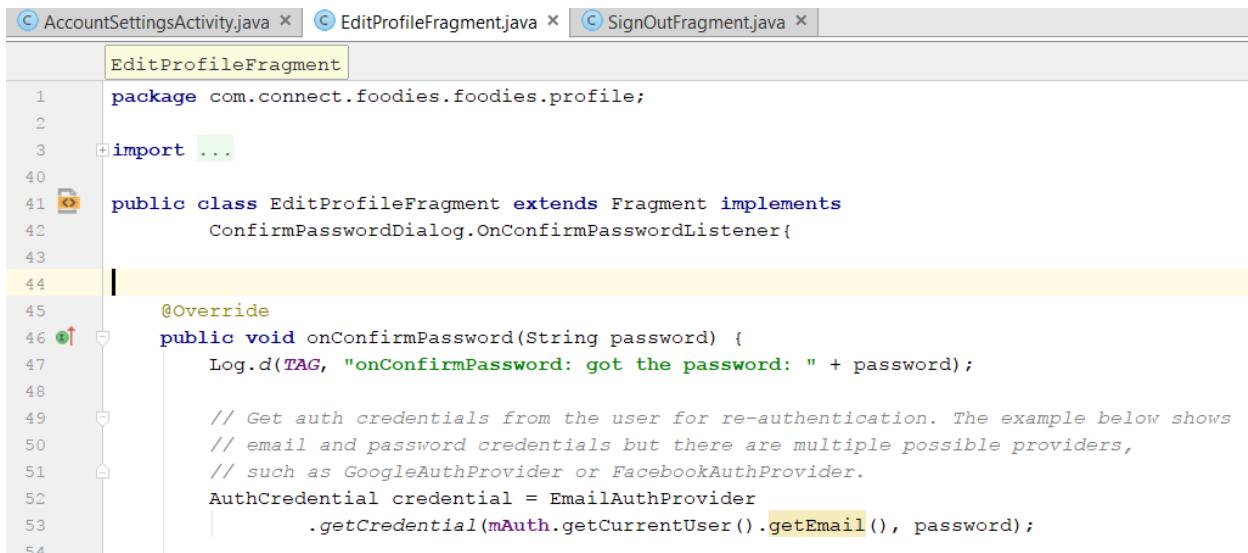
SignOutFragment.java



Code: AccountSettingsActivity.java

```
AccountSettingsActivity.java *  EditProfileFragment.java *  SignOutFragment.java *  
AccountSettingsActivity setupFragments()  
1 package com.connect.foodies.foodies.profile;  
2  
3 import ...  
27  
28 public class AccountSettingsActivity extends AppCompatActivity {  
29  
30     private static final String TAG = "AccountSettingsActivity";  
31     private static final int ACTIVITY_NUM = 3;  
32  
33     private Context mContext;  
34     public SectionsStatePagerAdapter pagerAdapter;  
35     private ViewPager mViewPager;  
36     private RelativeLayout mRelativeLayout;  
37
```

Code: EditProfileFragment.java



```

1 package com.connect.foodies.foodies.profile;
2
3 +import ...
4
5 public class EditProfileFragment extends Fragment implements
6     ConfirmPasswordDialog.OnConfirmPasswordListener{
7
8     @Override
9     public void onConfirmPassword(String password) {
10         Log.d(TAG, "onConfirmPassword: got the password: " + password);
11
12         // Get auth credentials from the user for re-authentication. The example below shows
13         // email and password credentials but there are multiple possible providers,
14         // such as GoogleAuthProvider or FacebookAuthProvider.
15         AuthCredential credential = EmailAuthProvider
16             .getCredential(mAuth.getCurrentUser().getEmail(), password);
17     }
18 }

```

Code: SignOutFragment.java



```

1 package com.connect.foodies.foodies.profile;
2
3 +import ...
4
5 public class SignOutFragment extends Fragment {
6
7     private static final String TAG = "SignOutFragment";
8
9     //firebase
10    private FirebaseAuth mAuth;
11    private FirebaseAuth.AuthStateListener mAuthListener;
12
13    private ProgressBar mProgressBar;
14    private TextView tvSignout, tvSigningOut;
15
16 }

```

There are other fragments used in this project as well, such as Gallery and Photo Fragments are explained in 9. Media Section.

Animations used:

Slide as Gravity Left when clicking on Comments and when clicking on Post

```
public void onCommentThreadSelected(Photo photo, String callingActivity) {
    Log.d(TAG, "onCommentThreadSelected: selected a comment thread");

    ViewCommentsFragment fragment = new ViewCommentsFragment();
    Bundle args = new Bundle();
    args.putParcelable("PHOTO", photo);
    args.putString("Home Activity", "Home Activity");
    fragment.setArguments(args);
    fragment.setEnterTransition(new Slide(Gravity.LEFT));
    FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();
    transaction.replace(R.id.container, fragment);
    transaction.addToBackStack("View Comments");
    transaction.commit();

}
```

Shared element Activity transition:

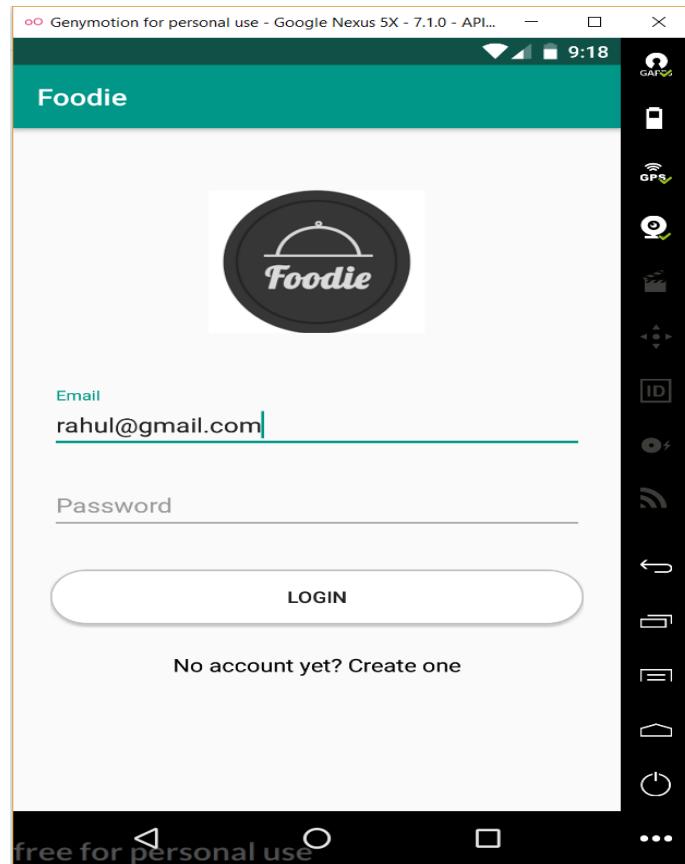
When Clicking on Floating Button Share Activity gets called.

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_gravity="bottom|center"
    android:layout_marginBottom="45dp"
    android:src="@drawable/floating"
    android:transitionName="testAnimation" />

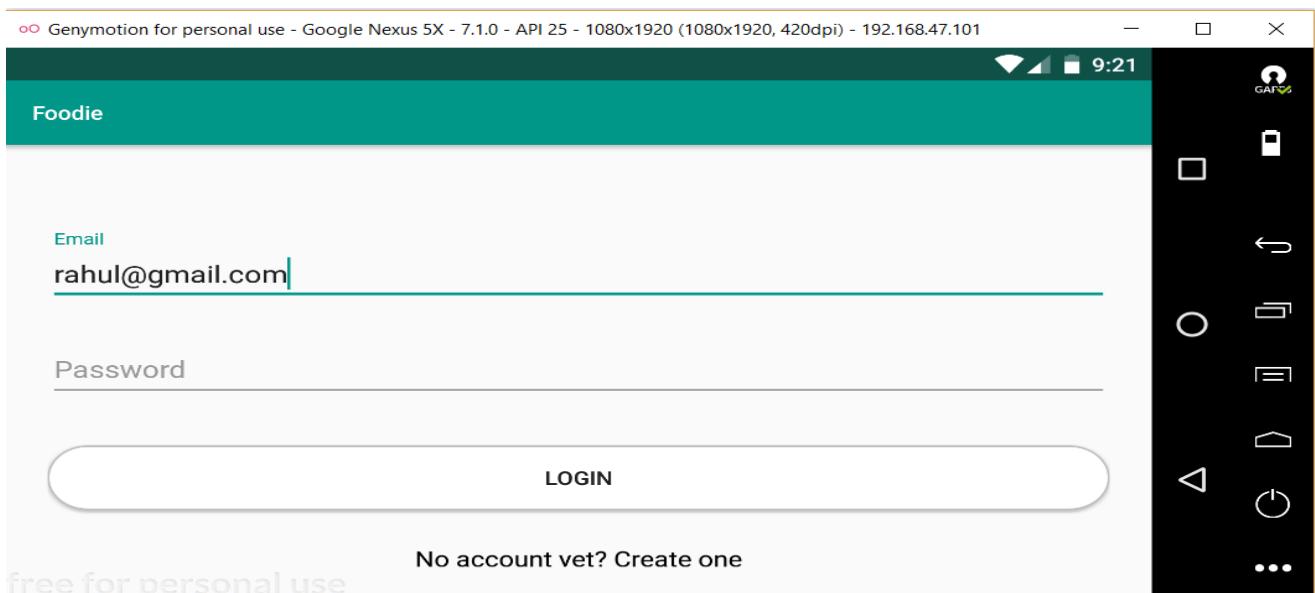
this.fab = (FloatingActionButton) findViewById(R.id.fab);
fab.setOnClickListener((view) ->
{
    Intent intent = new Intent(mContext, ShareActivity.class);
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
        ActivityOptionsCompat options = ActivityOptionsCompat.makeSceneTransitionAnimation(HomeActivity.this, view,
                "testAnimation");
        mContext.startActivity(intent, options.toBundle());
    } else {
        startActivity(intent);
    }
});
```

7. User/App State

Login Normal Mode:

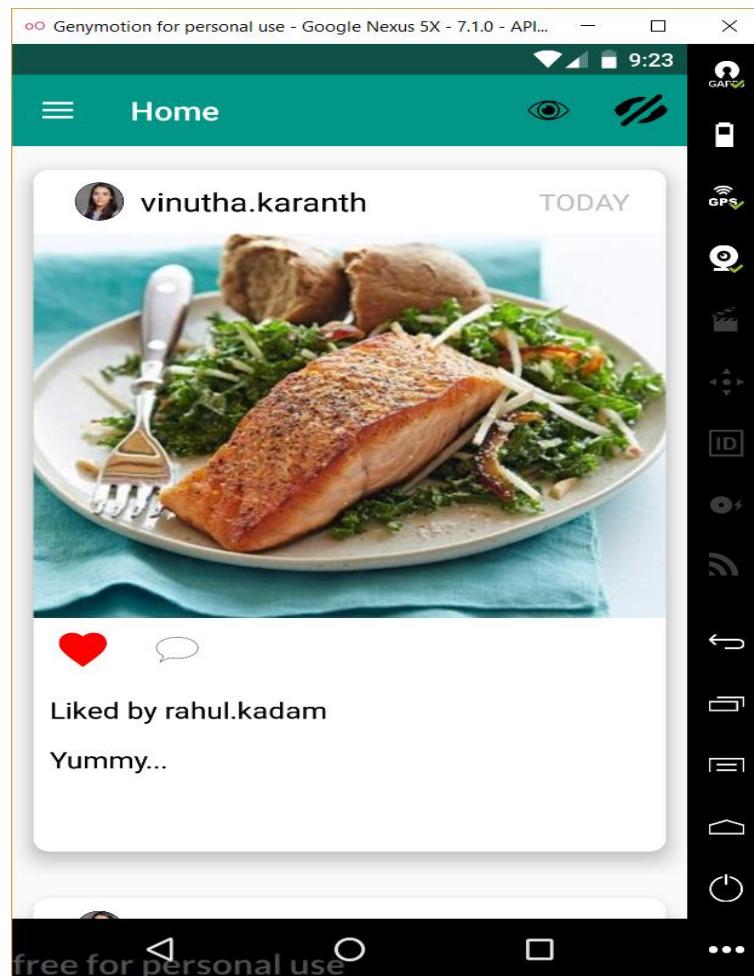


Login on Rotation:

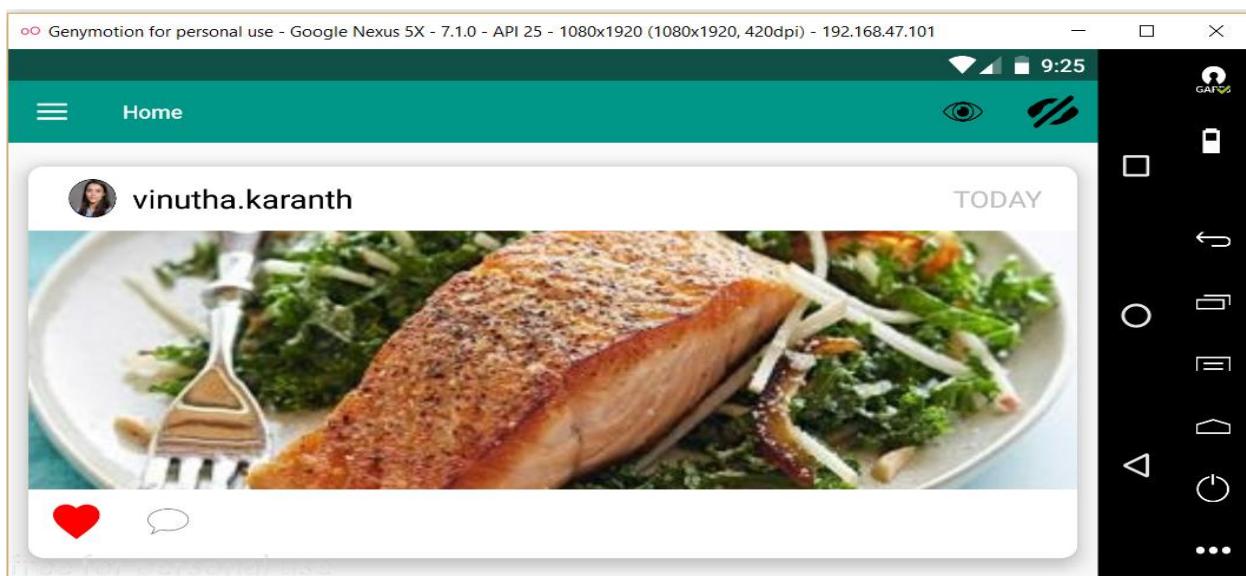


Foodie

Home Normal Mode:

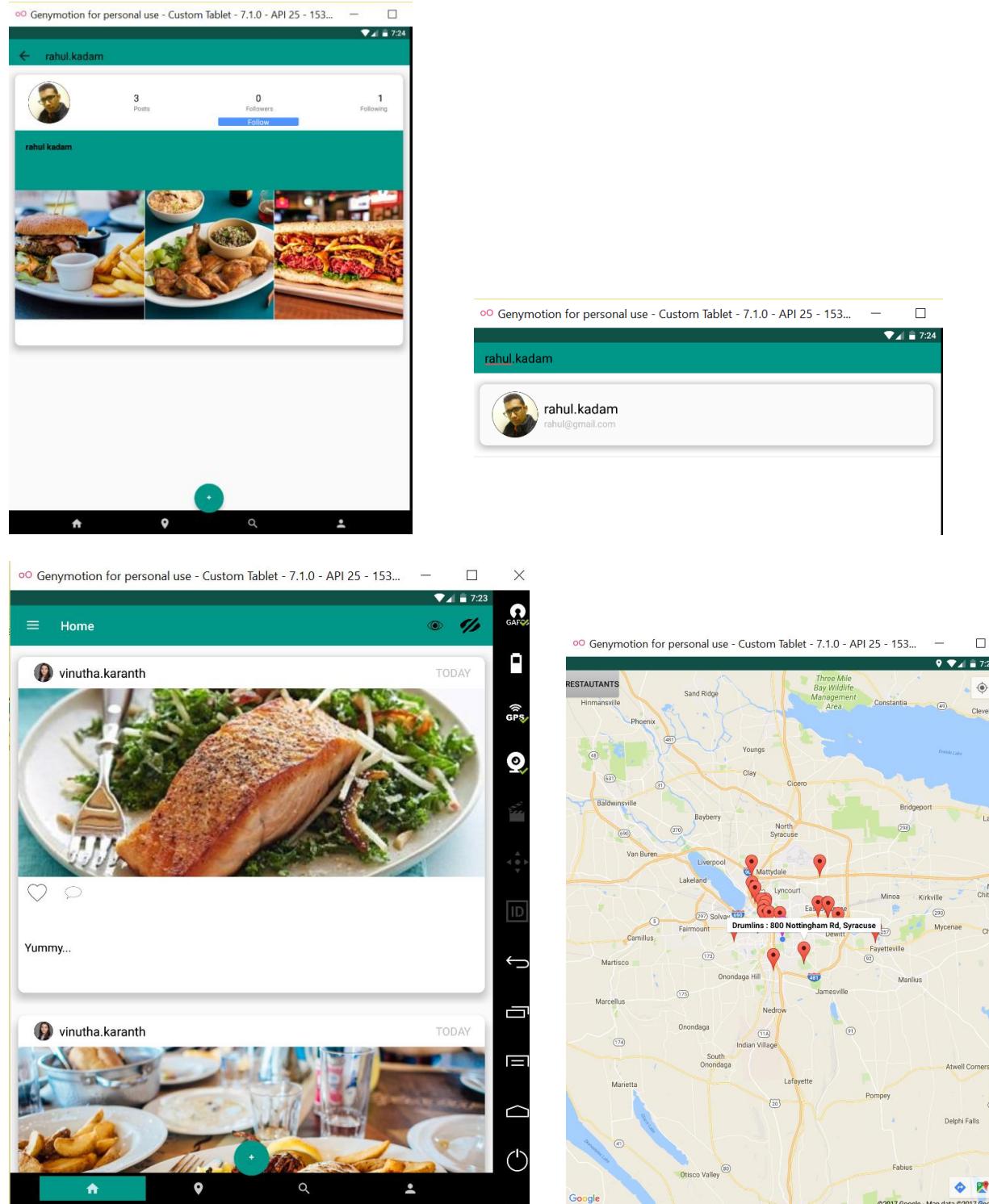


Home on Rotation:

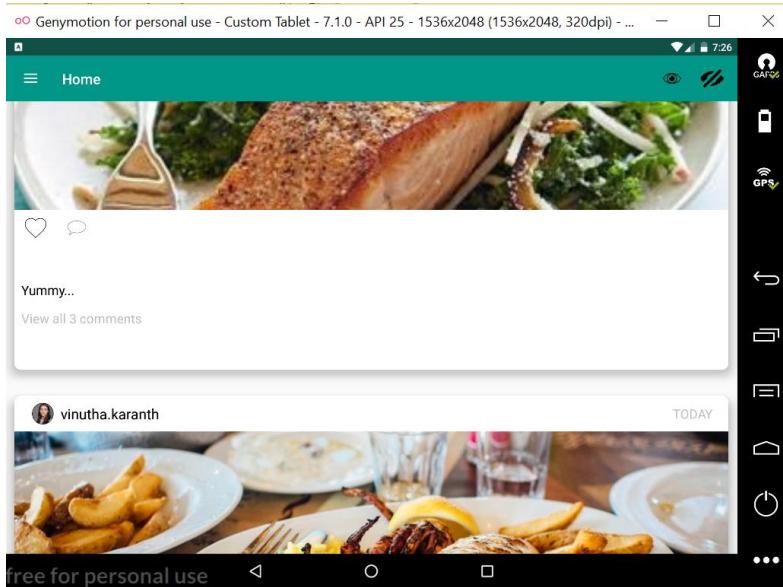


8. Multiple Devices/Screens & Orientation Changes using Constraint

App is tested on other devices like Custom Tablet 7.1.0 API 25 1536*2048 and all the screen are well organized and undisrupted.

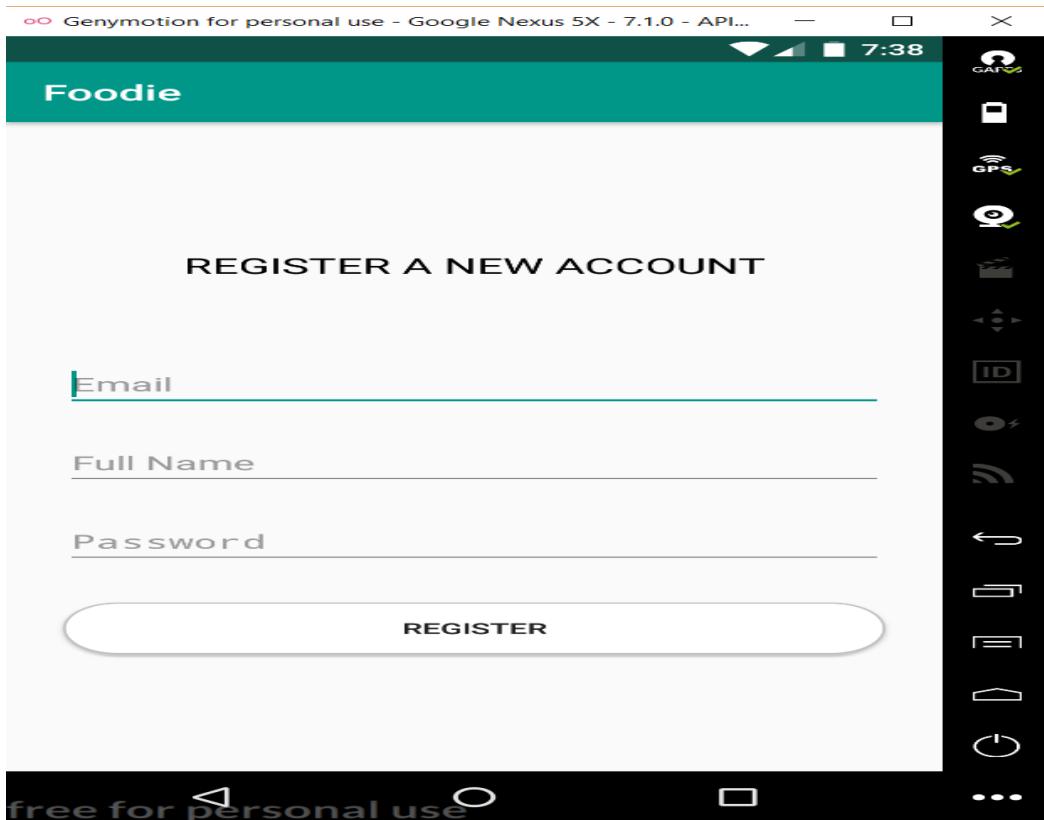


Screen Rotation in Tablet

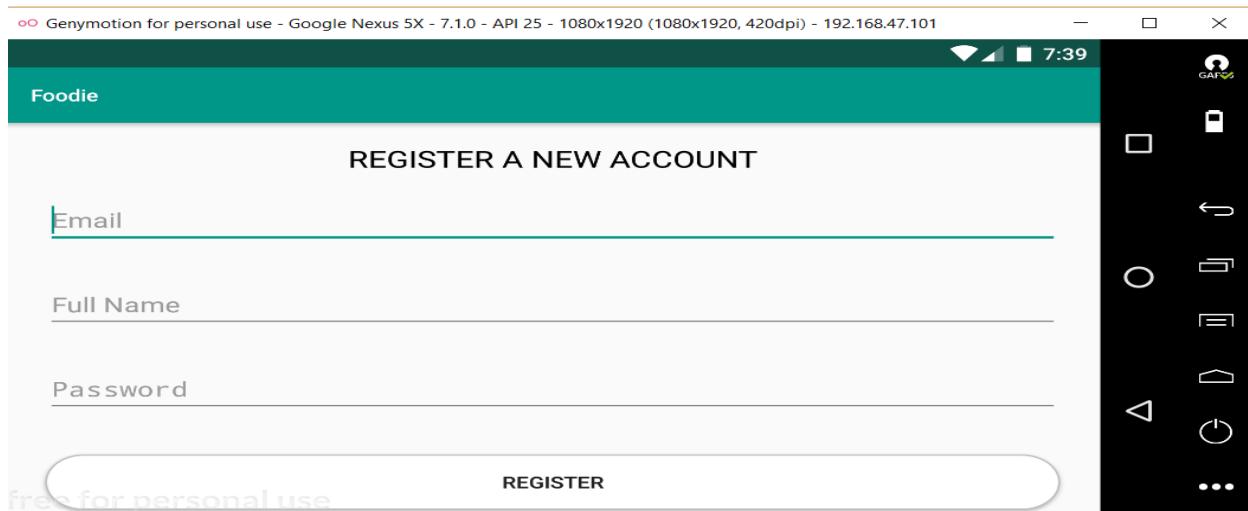


Orientation Changes using Constraint Layout:

Normal layout:



Landscape Layout:



Code: Normal layout:

```

<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

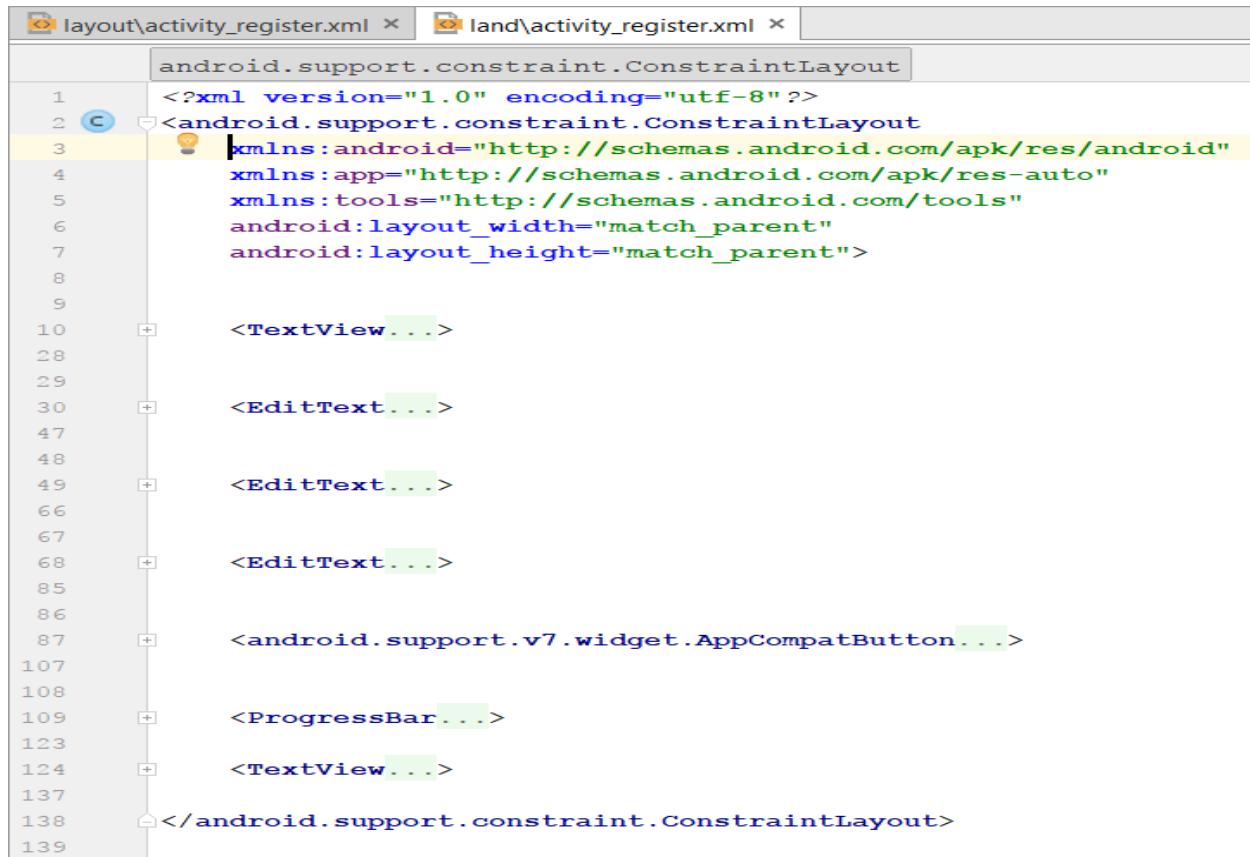
    <TextView...>
    <EditText...>
    <EditText...>
    <EditText...>
    <android.support.v7.widget.AppCompatButton...>
    <ProgressBar...>
    <TextView...>

</android.support.constraint.ConstraintLayout>

```

The code editor displays the XML layout for the 'activity_register.xml' file. It uses the 'ConstraintLayout' container and includes several 'EditText' and 'AppCompatButton' components. The XML uses standard Android namespaces and attributes like 'match_parent' for layout dimensions.

Code: Landscape Layout:



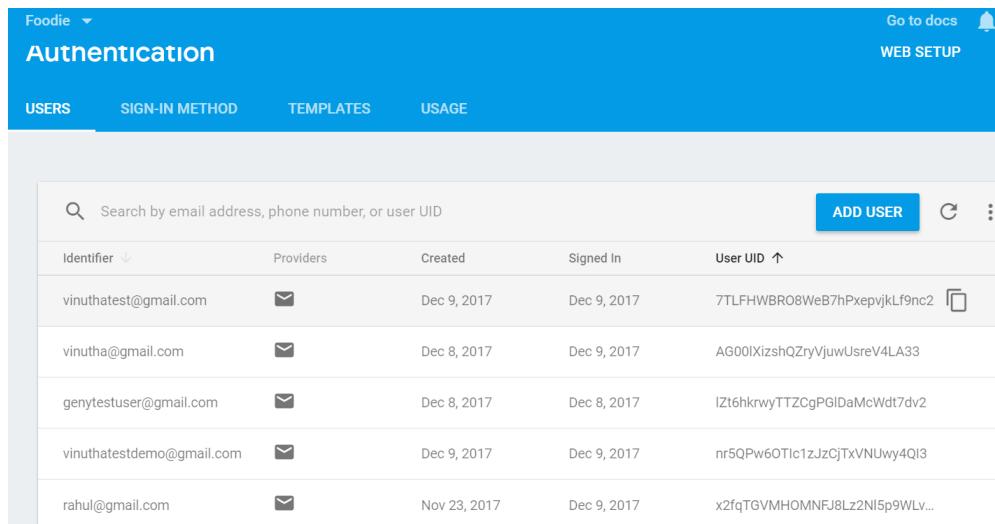
```

<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView...>
    <EditText...>
    <EditText...>
    <EditText...>
    <android.support.v7.widget.AppCompatButton...>
    <ProgressBar...>
    <TextView...>
</android.support.constraint.ConstraintLayout>

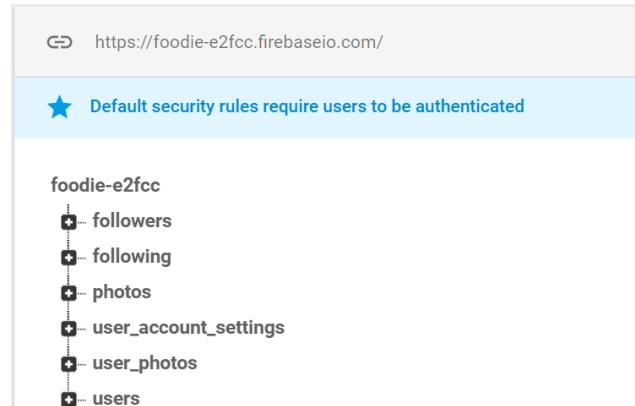
```

9. Host your data in the Cloud

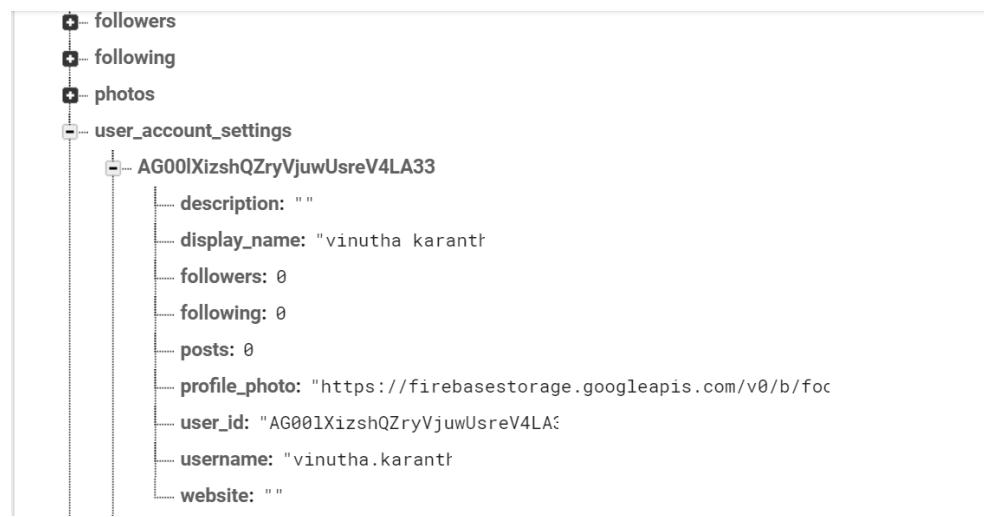


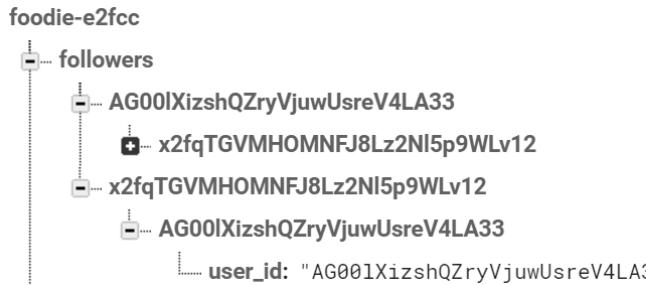
Identifier	Providers	Created	Signed In	User UID	
vinuthatest@gmail.com		Dec 9, 2017	Dec 9, 2017	7TLFHWBRO8WeB7hPxepyjkLf9nc2	
vinutha@gmail.com		Dec 8, 2017	Dec 9, 2017	AG00IXizshQZryVjuwUsreV4LA33	
genytestuser@gmail.com		Dec 8, 2017	Dec 8, 2017	IZt6hkrwyTTZCgPGIDaMcWdt7dv2	
vinuthatestdemo@gmail.com		Dec 9, 2017	Dec 9, 2017	nr5QPw6OTlc1zJzCjTxVNUwy4QI3	
rahul@gmail.com		Nov 23, 2017	Dec 9, 2017	x2fqTGVMHOMNfJ8Lz2NI5p9WLv...	

Nodes



UserAccount setting node





Storage

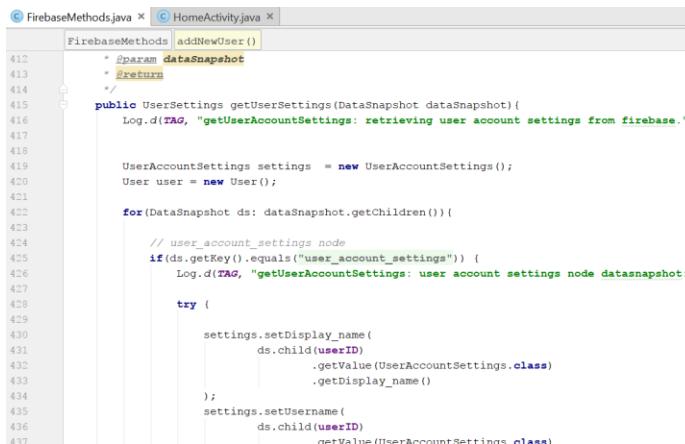
Name	Size	Type	Last modified
photos/	—	Folder	—
photo1	1.62 MB	image/jpeg	Dec 8, 2017
profile_photo	11.94 KB	image/jpeg	Dec 8, 2017

In the Java Code:

`FirebaseMethods.java` is the main Database file where all the database get and set activities are done. All the other activity or Fragment files calls the methods or function in the `FirebaseMethods.java` file by passing required arguments.

```

181
182     private void setProfilePhoto(String url){
183         Log.d(TAG, "setProfilePhoto: setting new profile image: " + url);
184
185         myRef.child(mContext.getString(R.string.dbname_user_account_settings))
186             .child(FirebaseAuth.getInstance().getCurrentUser().getUid())
187             .child(mContext.getString(R.string.profile_photo))
188             .setValue(url);
189     }
  
```



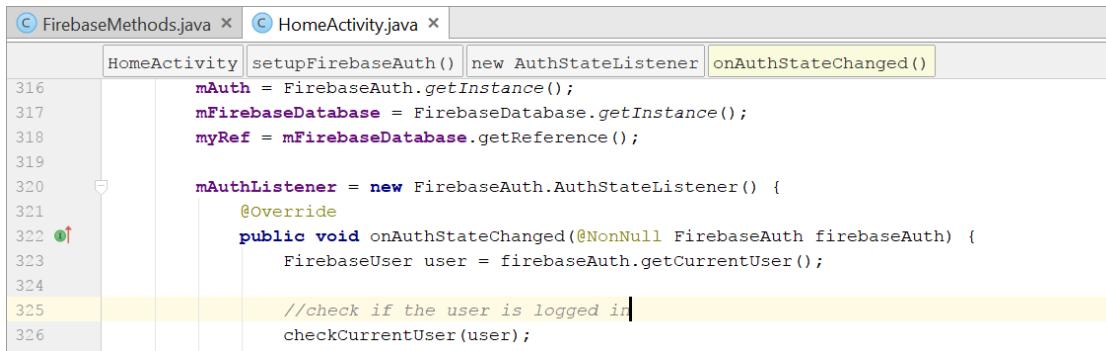
```

412     * @param dataSnapshot
413     * @return
414     */
415    public UserSettings getUserSettings(DataSnapshot dataSnapshot) {
416        Log.d(TAG, "getUserAccountSettings: retrieving user account settings from firebase.")
417
418        UserAccountSettings settings = new UserAccountSettings();
419        User user = new User();
420
421        for(DataSnapshot ds: dataSnapshot.getChildren()) {
422
423            // user_account_settings node
424            if(ds.getKey().equals("user_account_settings")) {
425                Log.d(TAG, "getUserAccountSettings: user account settings node dataSnapshot:");
426
427                try {
428
429                    settings.setDisplay_name(
430                        ds.child(userID)
431                            .getValue(UserAccountSettings.class)
432                                .getDisplay_name()
433                    );
434                    settings.setUsername(
435                        ds.child(userID)
436                            .getValue(UserAccountSettings.class)
437

```

Database is used in almost all the activities and fragments.

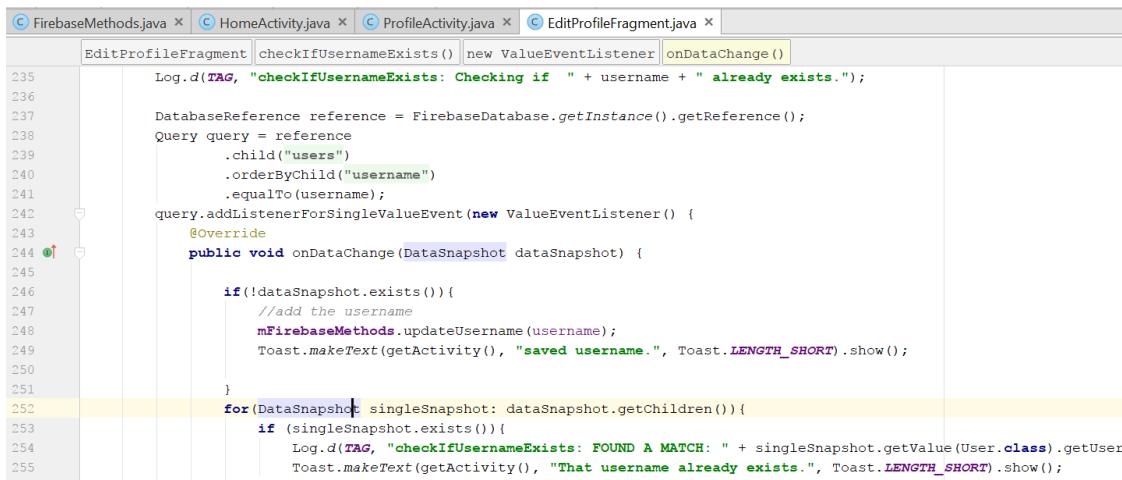
Example HomeActivity:



```

316     mAuth = FirebaseAuth.getInstance();
317     mFirebaseDatabase = FirebaseDatabase.getInstance();
318     myRef = mFirebaseDatabase.getReference();
319
320     mAuthListener = new FirebaseAuth.AuthStateListener() {
321         @Override
322         public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
323             FirebaseUser user = firebaseAuth.getCurrentUser();
324
325             //check if the user is logged in
326             checkCurrentUser(user);

```



```

235     Log.d(TAG, "checkIfUsernameExists: Checking if " + username + " already exists.");
236
237     DatabaseReference reference = FirebaseDatabase.getInstance().getReference();
238     Query query = reference
239         .child("users")
240         .orderByChild("username")
241         .equalTo(username);
242     query.addValueEventListener(new ValueEventListener() {
243         @Override
244         public void onDataChange(DataSnapshot dataSnapshot) {
245
246             if(!dataSnapshot.exists()){
247                 //add the username
248                 mFirebaseMethods.updateUsername(username);
249                 Toast.makeText(getActivity(), "saved username.", Toast.LENGTH_SHORT).show();
250             }
251         }
252         for(DataSnapshot singleSnapshot: dataSnapshot.getChildren()){
253             if (singleSnapshot.exists()){
254                 Log.d(TAG, "checkIfUsernameExists: FOUND A MATCH: " + singleSnapshot.getValue(User.class).getUsername());
255                 Toast.makeText(getActivity(), "That username already exists.", Toast.LENGTH_SHORT).show();

```

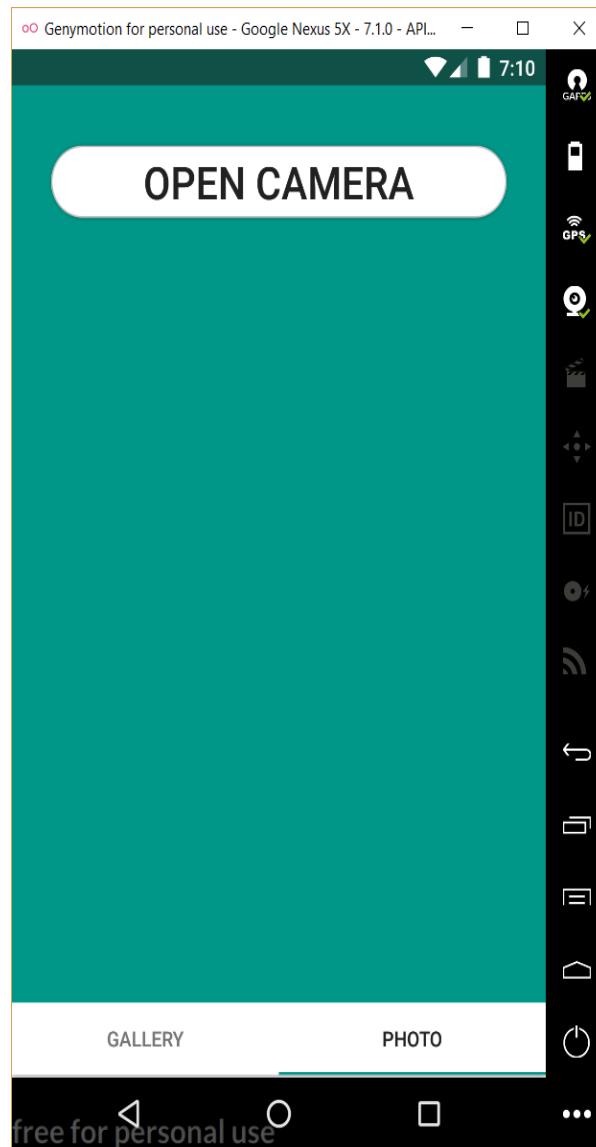
10. Media

Setting Up Permissions in AndroidManifest.xml

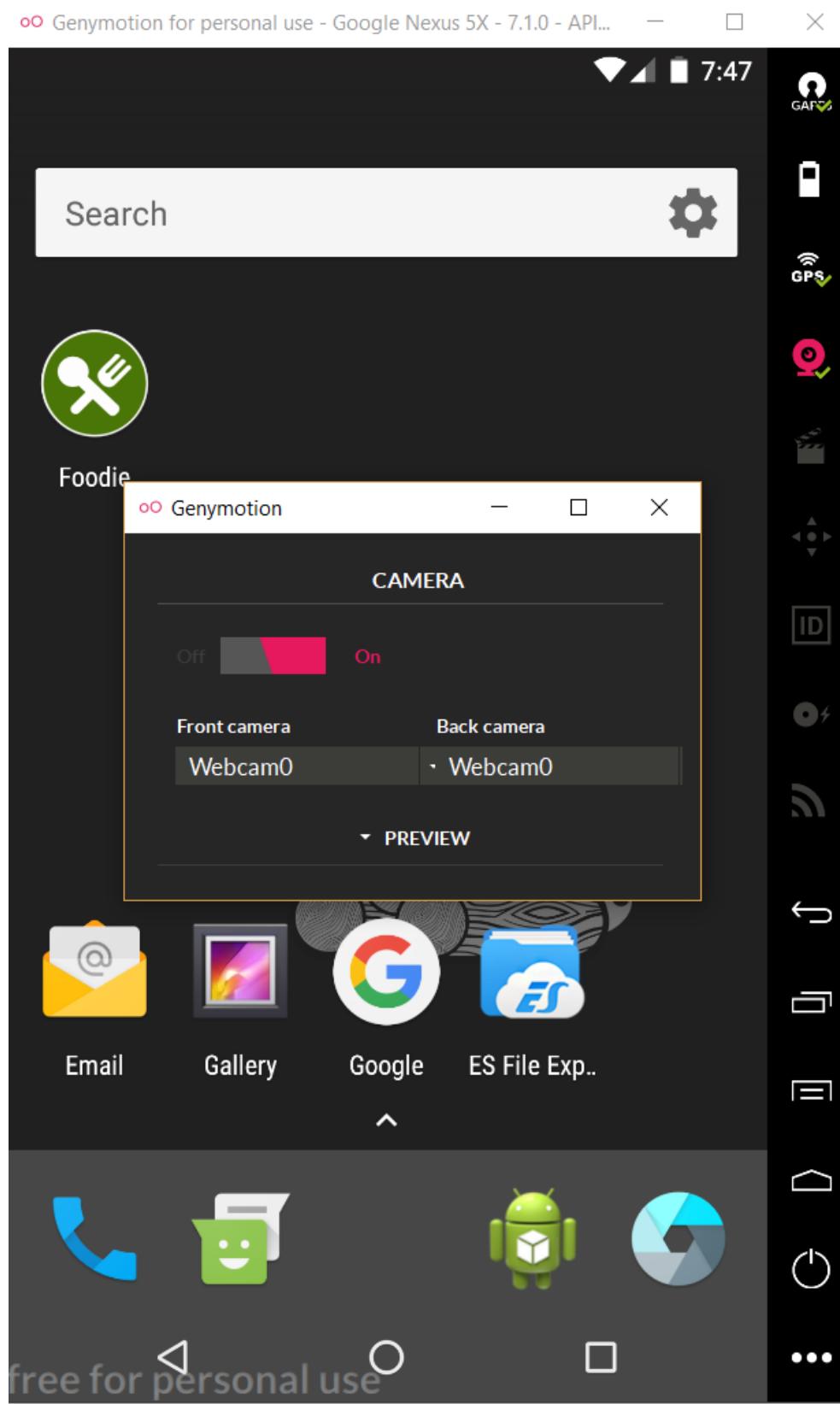
```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>
<uses-permission android:name="com.connect.foodies.mapretroapi.mymapsappsdirection.permission.MAPS_RECEIVE" />
```

10.1 Camera

Camera Fragment:



Adding Camera Settings in Geny Motion:



Code: Some essential functions in class PhotoFragment.java

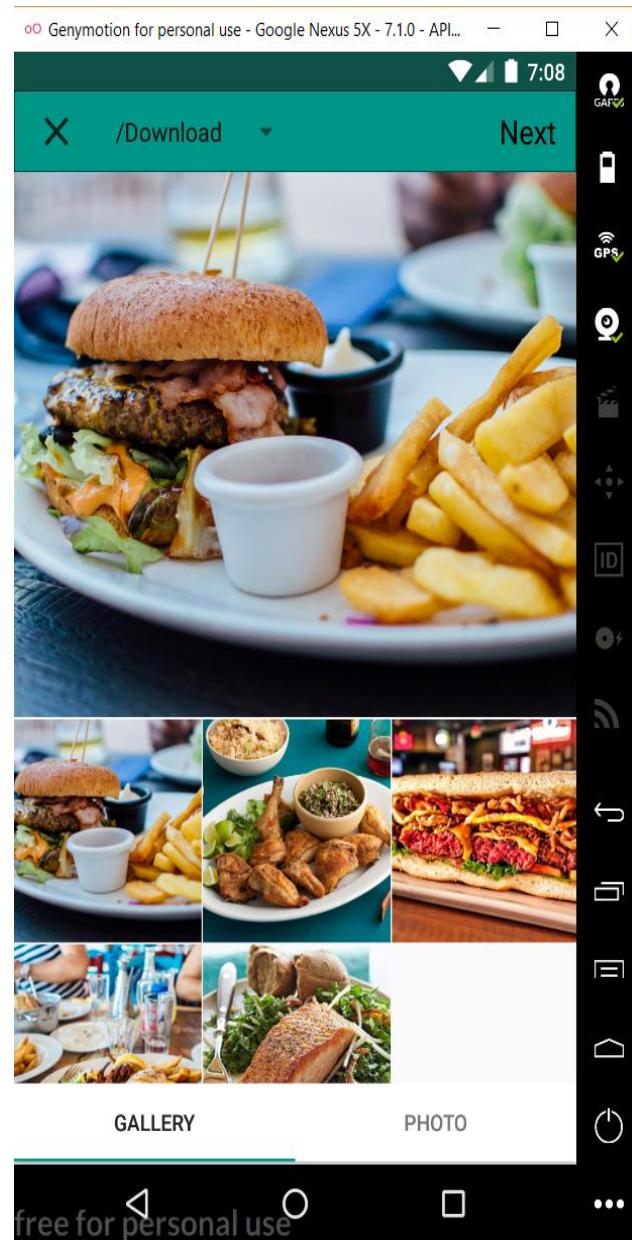
```

1 package com.connect.foodies.foodies.share;
2
3 import ...
4
5
6 public class PhotoFragment extends Fragment {
7     private static final String TAG = "PhotoFragment";
8     //constant
9     private static final int PHOTO_FRAGMENT_NUM = 1;
10    private static final int GALLERY_FRAGMENT_NUM = 2;
11    private static final int CAMERA_REQUEST_CODE = 5;
12
13    @Nullable
14    @Override
15    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container,
16                            @Nullable Bundle savedInstanceState) {
17        View view = inflater.inflate(R.layout.fragment_photo, container, false);
18        Log.d(TAG, "onCreateView: started.");
19        Button btnLaunchCamera = (Button) view.findViewById(R.id.btnLaunchCamera);
20        btnLaunchCamera.setOnClickListener((v) -> {
21            Log.d(TAG, "onClick: launching camera.");
22
23            if (((ShareActivity) getActivity()).getCurrentTabNumber() == PHOTO_FRAGMENT_NUM) {
24                if (((ShareActivity) getActivity()).checkPermissions(Permissions.CAMERA_PERMISSION[0])) {
25                    Log.d(TAG, "onClick: starting camera");
26                    Intent cameraIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
27                    startActivityForResult(cameraIntent, CAMERA_REQUEST_CODE);
28                } else {
29                    Intent intent = new Intent(getActivity(), ShareActivity.class);
30                    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
31                    startActivity(intent);
32                }
33            }
34        });
35        return view;
36    }
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62    public void onActivityResult(int requestCode, int resultCode, Intent data) {
63        super.onActivityResult(requestCode, resultCode, data);
64        if (requestCode == CAMERA_REQUEST_CODE) {
65            if (data != null) {
66                if (data.getExtras() != null) {
67                    Bitmap bitmap;
68                    bitmap = (Bitmap) data.getExtras().get("data");
69                    if (isRootTask()) {
70                        try {
71                            Log.d(TAG, "onActivityResult: received new bitmap from camera: " + bitmap);
72                            Intent intent = new Intent(getActivity(), NextActivity.class);
73                            intent.putExtra("selected_bitmap", bitmap);
74                            startActivity(intent);
75                        } catch (NullPointerException e) {
76                            Log.d(TAG, "onActivityResult: NullPointerException: " + e.getMessage());
77                        }
78                    } else {
79                        try {
80                            Log.d(TAG, "onActivityResult: received new bitmap from camera: " + bitmap);
81                            Intent intent = new Intent(getActivity(), AccountSettingsActivity.class);
82                            intent.putExtra("selected_bitmap", bitmap);
83                            intent.putExtra("return_to_fragment", "Edit Profile");
84                            startActivity(intent);
85                            getActivity().finish();
86                        } catch (NullPointerException e) {
87                            Log.d(TAG, "onActivityResult: NullPointerException: " + e.getMessage());
88                        }
89                    }
90                } else {
91                    Log.d(TAG, "onActivityResult: Camera Cancelled");
92                }
93            }
94        }
95    }

```

10.2 Gallery

Gallery Fragment



Code given on next page

Code: Some essential functions in class GalleryFragment.java



```

C) GalleryFragment.java x
    |   GalleryFragment onCreateView()
1 package com.connect.foodies.foodies.share;
2 import ...
3 public class GalleryFragment extends Fragment {
4     private static final String TAG = "GalleryFragment";
5     //constants
6     private static final int NUM_GRID_COLUMNS = 3;
7     //widgets
8     private GridView gridView;
9     private ImageView galleryImage;
10    private ProgressBar mProgressBar;
11    private Spinner directorySpinner;
12    //vars
13    private ArrayList<String> directories;
14    private String mAppend = "file:/";
15    private String mSelectedImage;
16    @Nullable
17    @Override
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43

C) GalleryFragment.java x
    |   GalleryFragment onCreateView()
44 public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container,
45                                     @Nullable Bundle savedInstanceState) {
46     View view = inflater.inflate(R.layout.fragment_gallery, container, false);
47     galleryImage = (ImageView) view.findViewById(R.id.galleryImageView);
48     gridView = (GridView) view.findViewById(R.id.gridView);
49     directorySpinner = (Spinner) view.findViewById(R.id.spinnerDirectory);
50     mProgressBar = (ProgressBar) view.findViewById(R.id.progressBar);
51     mProgressBar.setVisibility(View.GONE);
52     directories = new ArrayList<>();
53     Log.d(TAG, "onCreateView: started.");
54     ImageView shareClose = (ImageView) view.findViewById(R.id.ivCloseShare);
55     shareClose.setOnClickListener(v -> {
56         Log.d(TAG, "onClick: closing the gallery fragment.");
57         getActivity().finish();
58     });
59     TextView nextScreen = (TextView) view.findViewById(R.id.tvNext);
60     nextScreen.setOnClickListener(v -> {
61         Log.d(TAG, "onClick: navigating to the final share screen.");
62
63         if (isRootTask()) {
64             Intent intent = new Intent(getActivity(), NextActivity.class);
65             intent.putExtra("selected_image", mSelectedImage);
66             startActivity(intent);
67         } else {
68             Intent intent = new Intent(getActivity(), AccountSettingsActivity.class);
69             intent.putExtra("selected_image", mSelectedImage);
70             intent.putExtra("return_to_fragment", "Edit Profile");
71             startActivity(intent);
72             getActivity().finish();
73         }
74     });
75
76
77
78
79
80
81     init();
82
83     return view;
}

```

Foodie



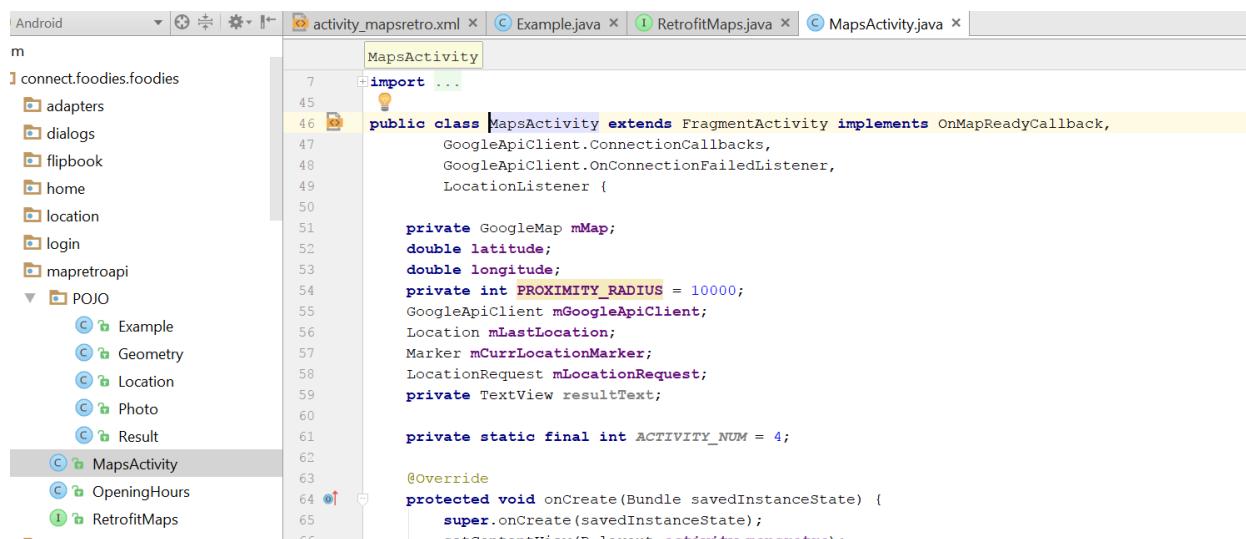
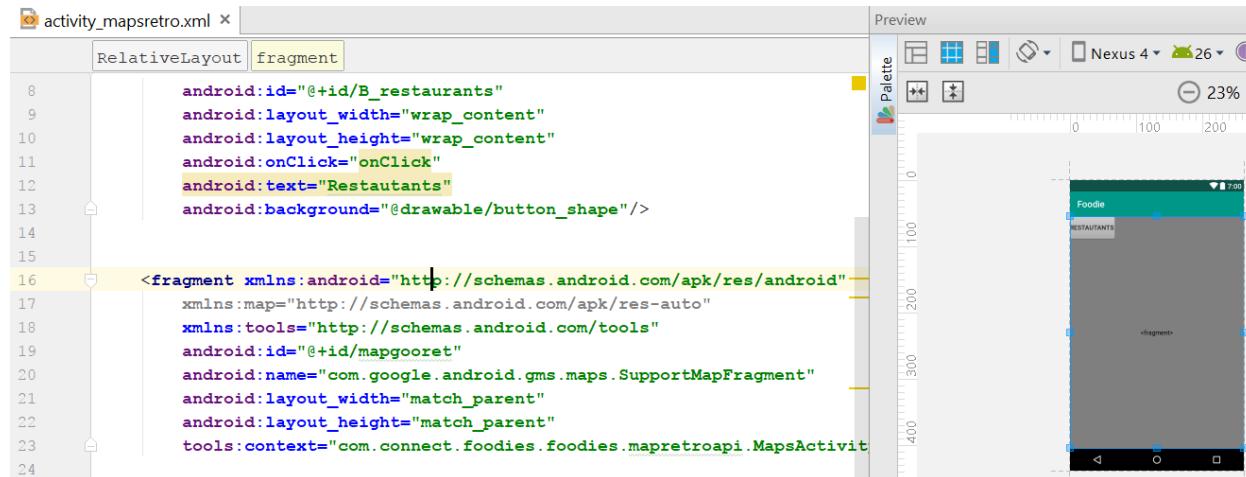
```
© GalleryFragment.java x
    GalleryFragment init() new OnItemSelectedListener
91     private void init() {
92         FilePaths filePaths = new FilePaths();
93
94         //check for other folders inside "/storage/emulated/0/pictures"
95         if (FileSearch.getDirectoryPaths(filePaths.PICTURES) != null) {
96             directories = FileSearch.getDirectoryPaths(filePaths.PICTURES);
97         }
98         directories.add(filePaths.DOWNLOAD);
99         directories.add(filePaths.CAMERA);
100        ArrayList<String> directoryNames = new ArrayList<>();
101        for (int i = 0; i < directories.size(); i++) {
102            Log.d(TAG, "init: directory: " + directories.get(i));
103            int index = directories.get(i).lastIndexOf("/");
104            String string = directories.get(i).substring(index);
105            directoryNames.add(string);
106        }
107        ArrayAdapter<String> adapter = new ArrayAdapter<~>(getActivity(),
108            android.R.layout.simple_spinner_item, directoryNames);
109        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
110        directorySpinner.setAdapter(adapter);
111        directorySpinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
112            @Override
113            public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
114                Log.d(TAG, "onItemClick: selected: " + directories.get(position));
115
116                //setup our image grid for the directory chosen
117                setupGridView(directories.get(position));
118            }
119            @Override
120            public void onNothingSelected(AdapterView<?> parent) {
121            }
122        });
123    }
124}
```

```
© GalleryFragment.java x
    GalleryFragment setupGridView()
125    private void setupGridView(String selectedDirectory) {
126        Log.d(TAG, "setupGridView: directory chosen: " + selectedDirectory);
127        final ArrayList<String> imgURLs = FileSearch.getFilePaths(selectedDirectory);
128        //set the grid column width
129        int gridWidth = getResources().getDisplayMetrics().widthPixels;
130        int imageWidth = gridWidth / NUM_GRID_COLUMNS;
131        gridView.setColumnWidth(imageWidth);
132        if (imgURLs != null) {
133            if (imgURLs.size() != 0) {
134                //use the grid adapter to adapter the images to gridview
135                GridImageAdapter adapter = new GridImageAdapter(getActivity(), R.layout.layout_grid_imageview, mAppend, imgURLs);
136                gridView.setAdapter(adapter);
137                //set the first image to be displayed when the activity fragment view is inflated
138                try {
139                    setImage(imgURLs.get(0), galleryImage, mAppend);
140                    mSelectedImage = imgURLs.get(0);
141                } catch (ArrayIndexOutOfBoundsException e) {
142                    Log.e(TAG, "setupGridView: ArrayIndexOutOfBoundsException: " + e.getMessage());
143                }
144            }
145            gridView.setOnItemClickListener((parent, view, position, id) -> {
146                Log.d(TAG, "onItemClick: selected an image: " + imgURLs.get(position));
147                setImage(imgURLs.get(position), galleryImage, mAppend);
148                mSelectedImage = imgURLs.get(position);
149            });
150        } else {
151            Log.d(TAG, "setupGridView: No Images In Download Folder");
152        }
153    } else {
154        Log.d(TAG, "setupGridView: No Images In Download Folder");
155    }
156}
157}
```

11. Location

11.1 Map

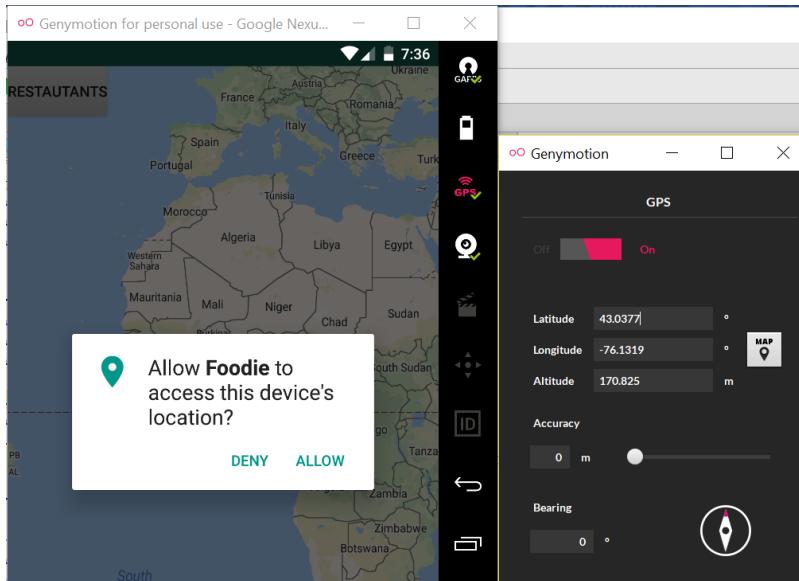
Longitude and Latitude is received from the retrofit URL in json format. All the acquired data will be set on the map with the markers.



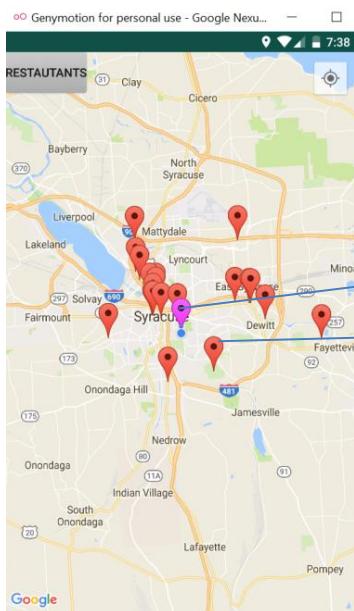
Foodie



```
C MapsActivity.java x
MapsActivity
138
139     }
140
141     private void build_retrofit_and_get_response(String type) {
142
143         String url = "https://maps.googleapis.com/maps/";
144
145         Retrofit retrofit = new Retrofit.Builder()
146             .baseUrl(url)
147             .addConverterFactory(GsonConverterFactory.create())
148             .build();
149
150         RetrofitMaps service = retrofit.create(RetrofitMaps.class);
151
152         Call<Example> call = service.getNearbyPlaces(type, latitude + "," + longitude, PROXIMITY_RAD
153
154         call.enqueue(new Callback<Example>() {
155             @Override
156             public void onResponse(Response<Example> response, Retrofit retrofit) {
157
158                 try {
159                     mMap.clear();
160                     // This loop will go through all the results and add marker on each location.
161                     for (int i = 0; i < response.body().getResults().size(); i++) {
162                         Double lat = response.body().getResults().get(i).getGeometry().getLocation()
163                         Double lng = response.body().getResults().get(i).getGeometry().getLocation()
164                         String placeName = response.body().getResults().get(i).getName();
165
166                     }
167                 } catch (Exception e) {
168                     e.printStackTrace();
169                 }
170             }
171         });
172     }
173 }
```



On clicking Allow; Go back and select Location tab again to get current location and click on Restaurant button to view all the nearby Restaurants.



Current Location will be shown in
HUE_MAGENTA
All the Nearby Restaurants will be shown in
HUE_RED

11.2 GPS

For Current Location we enable GPS location. And the current location will be shown in HUE_MAGENTA

```

<?xml version="1.0" encoding="utf-8"?
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.metro" >
<application>
    <activity android:name=".MainActivity" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
</manifest>

```

```

    /**
     * @Override
     public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        // mMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);

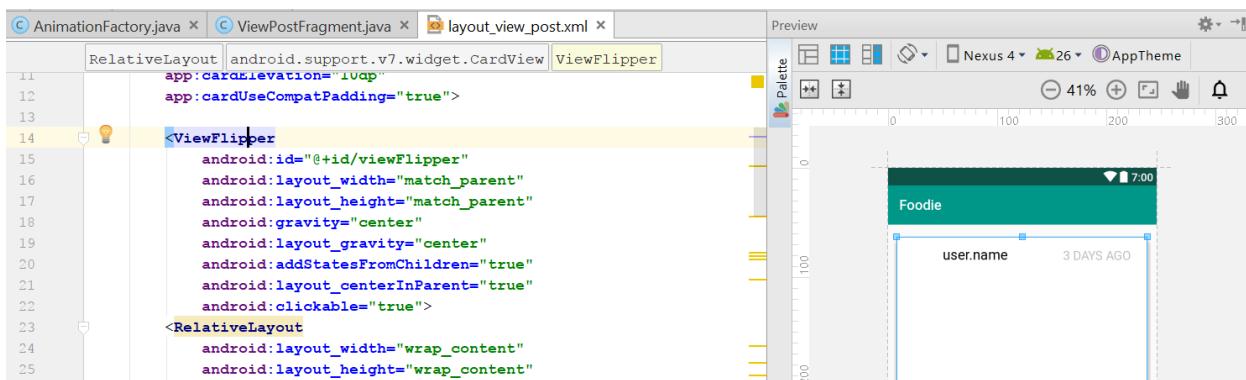
        //Initialize Google Play Services
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (ContextCompat.checkSelfPermission(this,
                    Manifest.permission.ACCESS_FINE_LOCATION)
                    == PackageManager.PERMISSION_GRANTED) {
                buildGoogleApiClient();
                mMap.setMyLocationEnabled(true);
            }
        }
    }

```

12. Advanced Animation - Flipbook Style



```
(C) AnimationFactory.java x (C) FlipAnimation.java x
FlipAnimation
37  */
38 public class FlipAnimation extends Animation {
39     private final float mFromDegrees;
40     private final float mToDegrees;
41     private final float mCenterX;
42     private final float mCenterY;
43     private Camera mCamera;
44
45     private final ScaleUpDownEnum scaleType;
46
47     /**
48      * How much to scale up/down. The default scale of 75% of full size seems optimal based on testing
49      */
50     public static final float SCALE_DEFAULT = 0.75f;
51
52     private float scale;
```



```
(C) AnimationFactory.java x (C) ViewPostFragment.java x
ViewPostFragment onCreateView()
117
118     mComment = (ImageView) view.findViewById(R.id.speech_bubble);
119     mComments = (TextView) view.findViewById(R.id.image_comments_link);
120     mViewAnimator = (ViewAnimator) view.findViewById(R.id.viewFlipper);
121
```

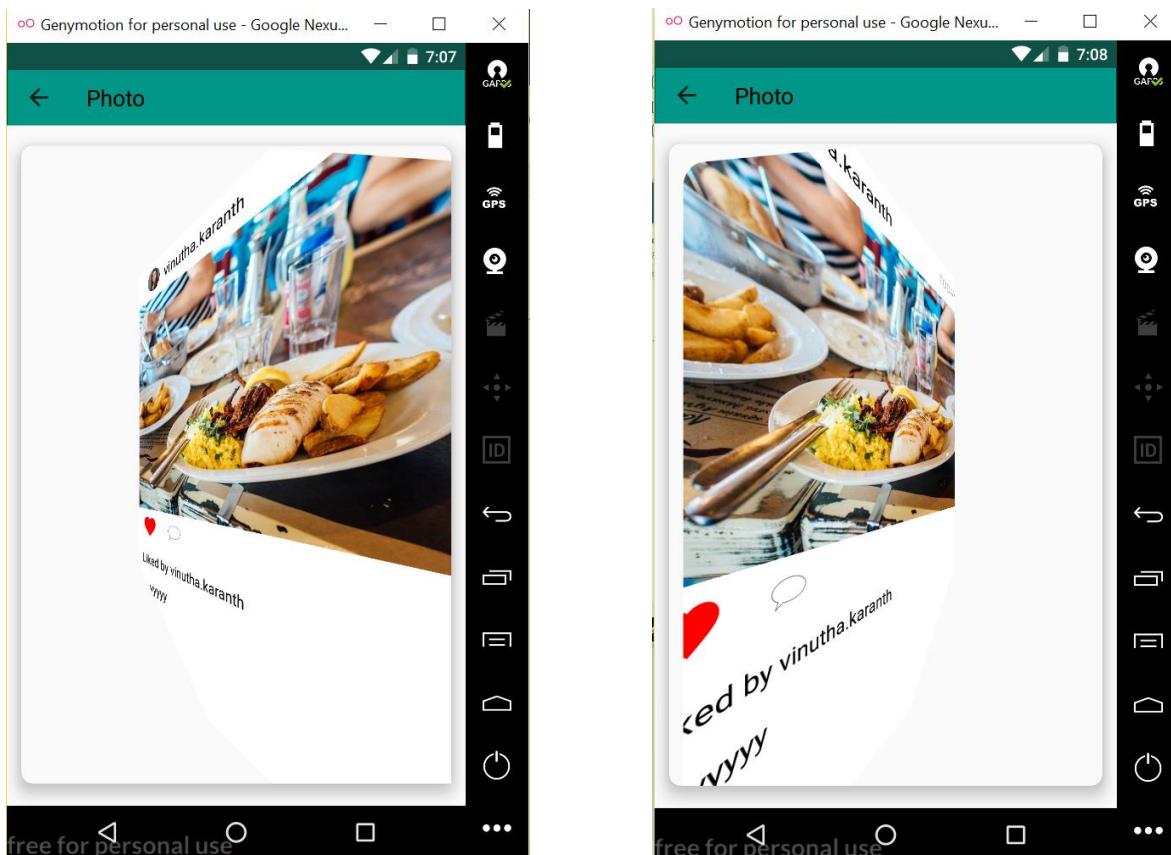


```

C AnimationFactory.java x C ViewPostFragment.java x
ViewPostFragment onCreateView()
470     mComment.setOnTouchListener((v) -> {
471         Log.d(TAG, "onClick: navigating back");
472         mOnCommentThreadSelectedListener.onCommentThreadSelectedListener(mPhoto);
473     });
474
475     mPostImage.setOnClickListener(new View.OnClickListener() {
476         @Override
477         public void onClick(View view) {
478             Log.d(TAG, "onClick: Image clicked");
479             AnimationFactory.flipTransition(mViewAnimator, AnimationFactory.FlipDirection.LEFT_RIGHT);
480         }
481     });
482 }
483
484 }
485
486 });

```

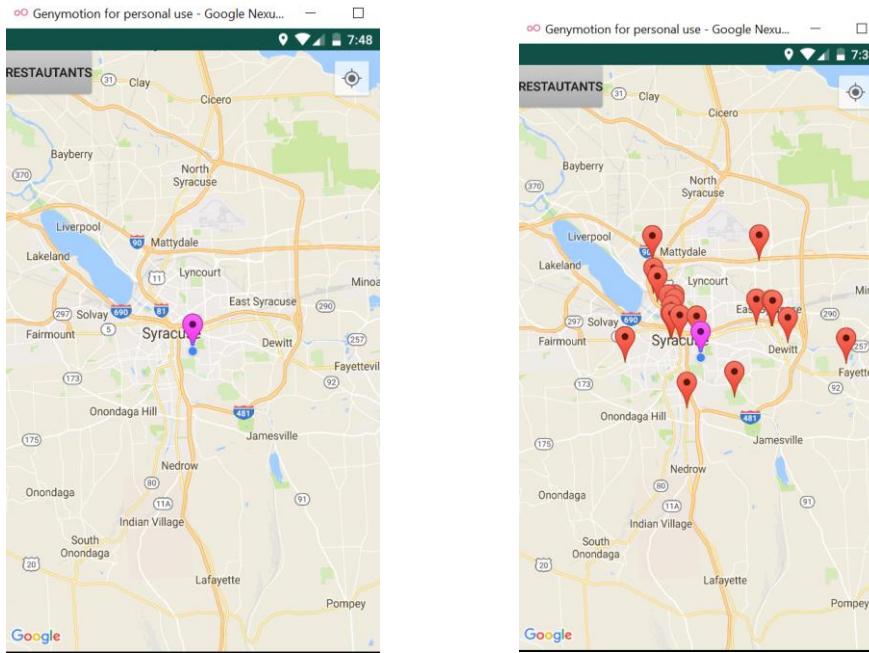
Animation in APP for ViewPostFragment



13. Data

13.1 Real API Data and Retrofit Library

App uses Google Maps API using Retrofit. We have used Retrofit to capture data from URL. Uses POJO classes to retrieve data from the source.



Android RetrofitMaps.java MapsActivity.java

```

RetrofitMaps
package com.connect.foodies.foodies.mapretroapi;

import ...

/**
 * Created by vinut on 11/27/2017.
 */

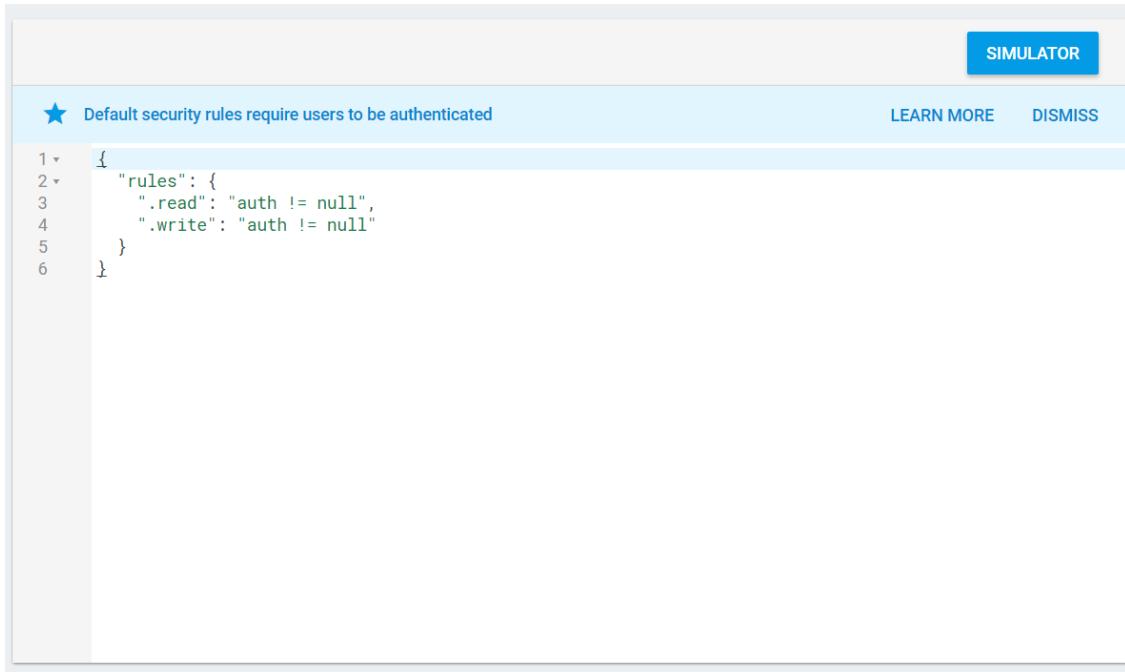
public interface RetrofitMaps {
    /**
     * Retrofit get annotation with our URL
     * And our method that will return us details of student.
     */
    @GET("api/place/nearbysearch/json?sensor=true&key=AIzaSyBh9yUPuiWsg4Mc0mwQroY96h6NdShmRB8")
    Call<Example> getNearbyPlaces(@Query("type") String type,
                                    @Query("location") String location,
                                    @Query("radius") int radius);
}

```



```
1 package com.connect.foodies.foodies.mapretroapi.POJO;
2
3 import ...
4
5 /**
6  * Created by vinut on 11/27/2017.
7  */
8
9
10 public class Example {
11
12     @SerializedName("html_attributions")
13     @Expose
14     private List<Object> htmlAttributions = new ArrayList<~>();
15
16     @SerializedName("next_page_token")
17     @Expose
18     private String nextPageToken;
19
20     @SerializedName("results")
21 }
```

14. Firebase Security



Default security rules require users to be authenticated

SIMULATOR

```
1 *
2 *   "rules": {
3 *     ".read": "auth != null",
4 *     ".write": "auth != null"
5 *   }
6 }
```

LEARN MORE DISMISS

15. System - Service

We have implemented Firebase Push Notifications as part of Service.

1. Creating a Push Notification in Firebase:



Notifications > Compose message

Message text
My First Push Demo

Message label (optional) ⓘ
Enter message nickname

Delivery date ⓘ
Send Now ▾

Target

User segment Topic Single device

Target user if...

App AND com.connect.foodies.foodies

Cannot add additional statements. All apps have been selected.

Conversion events ⓘ

Advanced options

All fields optional

Title ⓘ
My First Push Demo Title

Notifications > Compose message

Cannot add additional statements. All apps have been selected.

Conversion events ⓘ

Advanced options

All fields optional

Title ⓘ
My First Push Demo Title

Android Notification Channel ⓘ

Custom data ⓘ

Key	Value
-----	-------

Priority ⓘ Sound

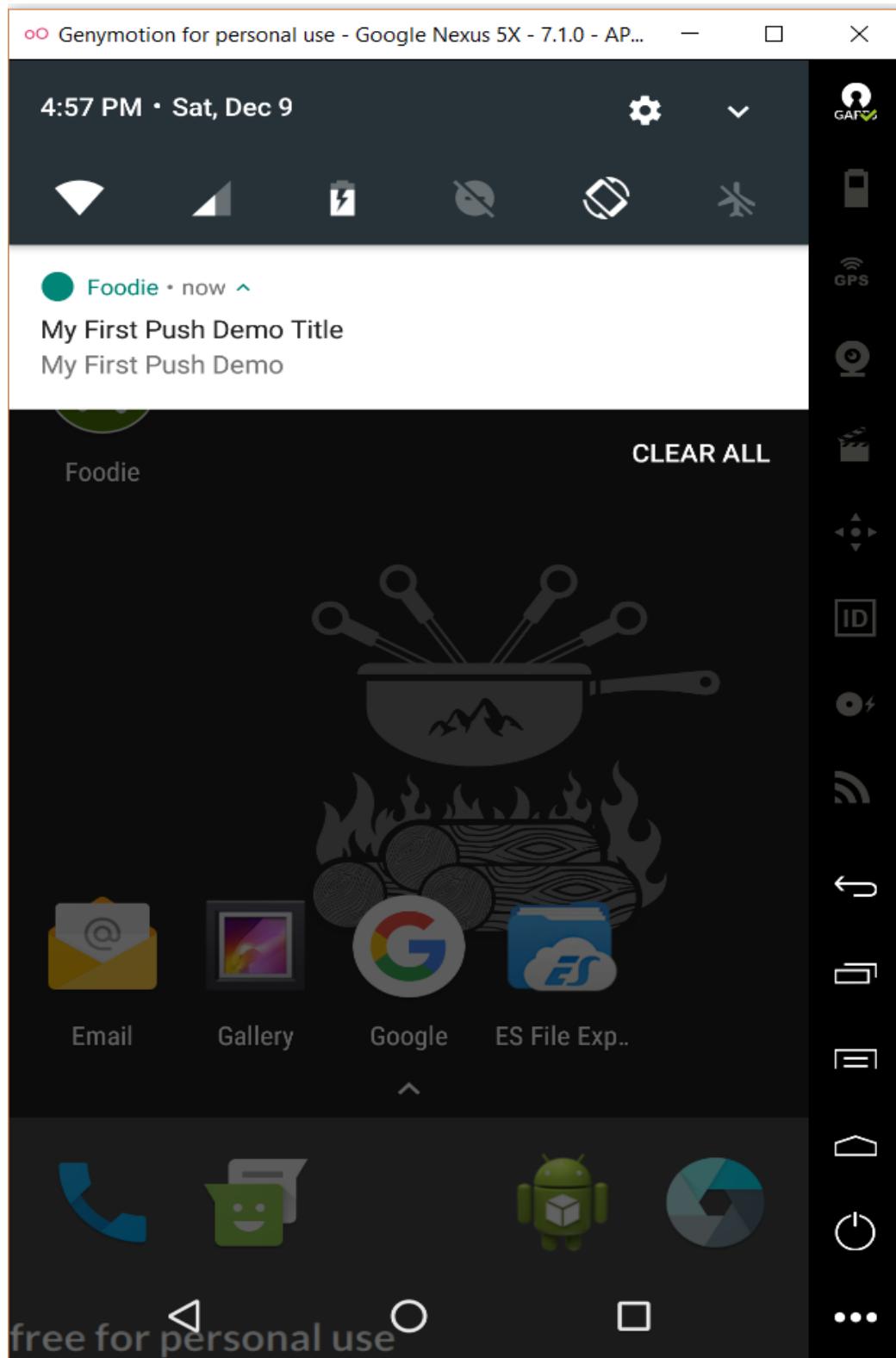
High ▾	Enabled ▾
--------	-----------

Expires ⓘ

4 ▾	Weeks ▾
-----	---------

SAVE AS DRAFT **SEND MESSAGE**

2. App Screen:



3. AndroidManifest.xml information added



```
AndroidManifest.xml x

81      android:name=".share.ShareActivity"
82      android:label="Share"
83      android:theme="@style/AppTheme.NoActionBar"/>
84  <activity
85      android:name=".share.NextActivity"
86      android:label="Next"
87      android:theme="@style/AppTheme.NoActionBar"/>
88
89
90  <activity android:name=".mapretroapi.MapsActivity"
91      android:configChanges="orientation|screenSize"
92      android:theme="@style/AppTheme.NoActionBar"
93      android:label="MapsgooRet"/>
94
95  <service
96      android:name=".utils.FirebaseMessagingService">
97      <intent-filter>
98          <action android:name="com.google.firebase.MESSAGING_EVENT" />
99      </intent-filter>
100     </service>
101
102    <meta-data
103        android:name="com.google.firebaseio.messaging.default_notification_color"
104        android:resource="@color/colorAccent" />
105
106
107    </application>
108
109 </manifest>
```

2. FirebaseMessagingService.java – File used for push Notifications.



```
C FirebaseMessagingService.java x
1 package com.connect.foodies.foodies.utils;
2 import ...
14 public class FirebaseMessagingService extends com.google.firebase.messaging.FirebaseMessagingService {
15     private static final String TAG = "FirebaseMessagingService";
16     public FirebaseMessagingService() {
17     }
18     @Override
19     public void onMessageReceived(RemoteMessage remoteMessage) {
20         String title = remoteMessage.getNotification().getTitle();
21         String message = remoteMessage.getNotification().getBody();
22         Log.d(TAG, "onMessageReceived: Message Received: \n" +
23             "Title: " + title + "\n" +
24             "Message: " + message);
25
26         sendNotification(title, message);
27     }
28     @Override
29     public void onDeletedMessages() {
30     }
31     private void sendNotification(String title, String messageBody) {
32         Intent intent = new Intent(this, HomeActivity.class);
33         intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
34         PendingIntent pendingIntent = PendingIntent.getActivity(this, 0 /* Request code */, intent,
35             PendingIntent.FLAG_ONE_SHOT);
36         Uri defaultSoundUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
37         NotificationCompat.Builder notificationBuilder = new NotificationCompat.Builder(this)
38             .setSmallIcon(R.mipmap.ic_launcher)
39             .setContentTitle(title)
40             .setContentText(messageBody)
41             .setAutoCancel(true)
42             .setSound(defaultSoundUri)
43             .setContentIntent(pendingIntent);
44         NotificationManager notificationManager =
45             (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
46         notificationManager.notify(0 /* ID of notification */, notificationBuilder.build());
47     }
48 }
```

16. Misc.

16.1 Shortcuts

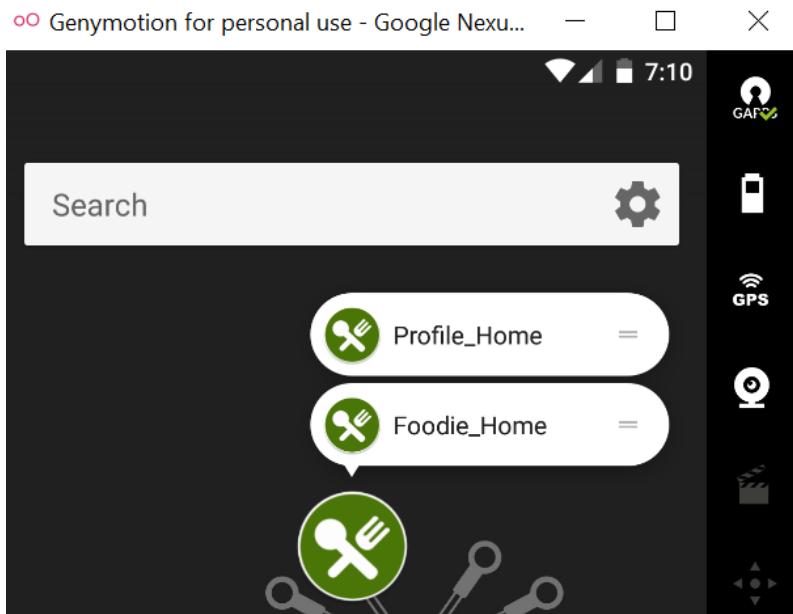
AndroidManifest.xml

```
46     <meta-data
47         android:name="android.app.shortcuts"
48         android:resource="@xml/shortcuts" />
49     </activity>
```

shortcuts.xml

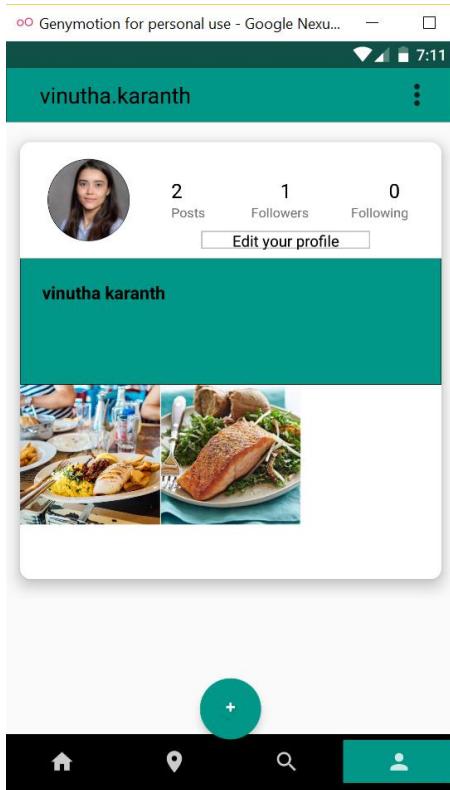
```
1  <?xml version="1.0" encoding="utf-8"?>
2  <shortcuts xmlns:android="http://schemas.android.com/apk/res/android">
3      <shortcut
4          android:shortcutId="Foodie_Home"
5          android:enabled="true"
6          android:icon="@mipmap/ic_launcher"
7          android:shortcutShortLabel="Foodie_Home"
8          android:shortcutLongLabel="Foodie_Home">
9              <intent
10                 android:action="android.intent.action.VIEW"
11                 android:targetPackage="com.connect.foodies.foodies"
12                 android:targetClass="com.connect.foodies.home.HomeActivity" ></intent>
13             <categories android:name="android.shortcut.conversation"/>
14         </shortcut>
15         <shortcut
16             android:shortcutId="Profile_Home"
17             android:enabled="true"
18             android:icon="@mipmap/ic_launcher"
19             android:shortcutShortLabel="Profile_Home"
20             android:shortcutLongLabel="Profile_Home">
21                 <intent
22                     android:action="android.intent.action.VIEW"
23                     android:targetPackage="com.connect.foodies.foodies"
24                     android:targetClass="com.connect.foodies.profile.ProfileActivity" ></intent>
25                 <categories android:name="android.shortcut.conversation"/>
26             </shortcut>
27         </shortcuts>
```

Shortcut display on long pressing on app icon

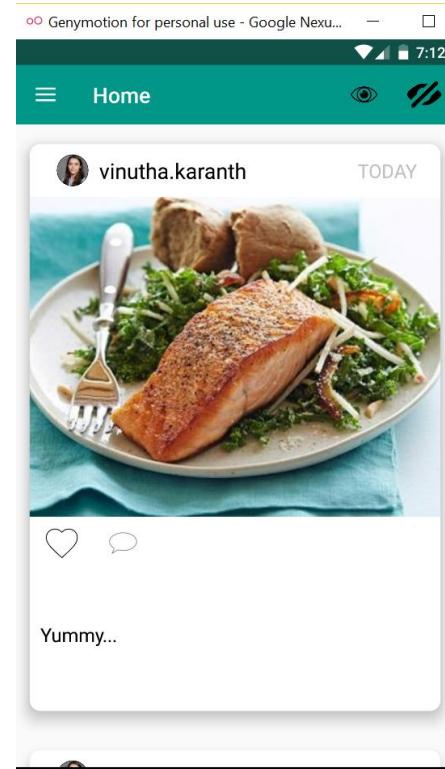


If User is Logged in:

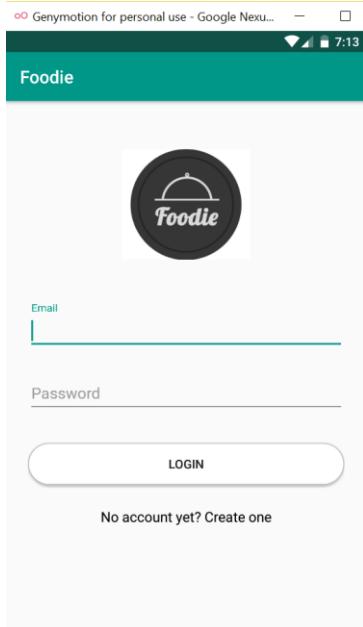
On clicking Profile_Home:



On clicking Foodie_Home:



When User is Logged Out both short cut takes user to Login Screen, App will not have any data backstacked:



16.2 Multi-Window

```
16
17
18 <application
19     android:allowBackup="true"
20     android:icon="@mipmap/ic_launcher"
21     android:label="Foodie"
22     android:supportsRtl="true"
23     android:resizeableActivity="true" <-- This line is highlighted
24     android:theme="@style/AppTheme"
25     tools:replace="android:allowBackup">
26 </application>
```

```
<activity
    android:name=".home.HomeActivity"
    android:label="Home"
    android:theme="@style/AppTheme.NoActionBar">
<layout android:defaultHeight="500dp"
        android:defaultWidth="600dp"
        android:gravity="top|end"
        android:minHeight="450dp"
        android:minWidth="300dp" />

</activity>
```

MultiWindow Demo in App (Both the apps works simultaneously):

