Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

GitHub Username: andermaco@gmail.com

Wouk

Description

Do you need to publish yourself offering for a job, maybe private lessons, homework, or a pet lost, but you don't want to use physical lampposts,bulletin boards, etc.
Wouk is an app that lets you to publish categorized advertisements (or just a bulletin/note) based on location in a very similar way of offering on lampposts, bulletin boards, etc.
You'll be able to publish an advertisement linked to an outdoor location.
As publisher, define your advertisement, bulletin, or note by a title, description and add an image if you need.

As viewer, you'll be able to get advertisements/notes based on your location, categorize adds and read about them, see the description, mark as favourite, chat with the publisher, or maybe share the advertisement with your friends.

E.g.:

Intended User

All people that need to publish an offer on a physical lampposts,bulletin boards, walls, about private lessons, pets, homework, etc.

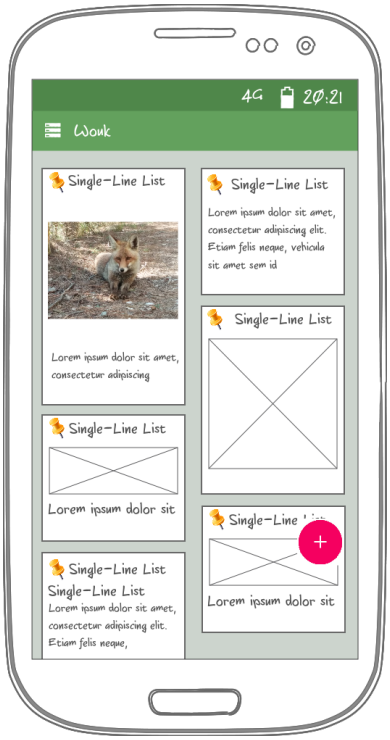
Features

List the main features of your app. For example:

- Saves information into the mobile
- Uses to Google Api Client for accessing to the Google Location Services API.
- Uses Google Cloud Messaging.
- Uses OAUTH2 validation capabilities.
- Connects to Google App Engine-Jersey based middleware to retrieve info from Google Cloud Datastore about closer advertisements and for saving advertisements.
- HTTP Google Cloud Messaging server to manage sender and response chat messages.
- Uses Google Cloud Storage to store images.
- Takes pictures
- Share advertisement with contacts.
- Send messages to publishers using GCM, and receive messages from them.

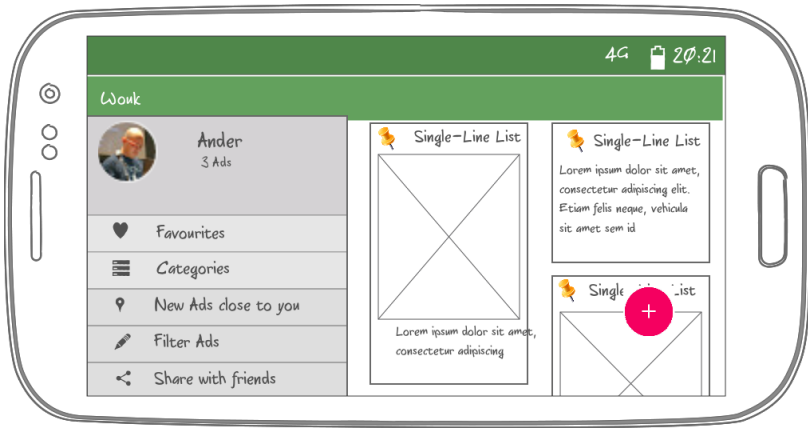
User Interface Mocks

Main Screen



Home frame, shows those advertisements close to the user (by close means a fixed distance)

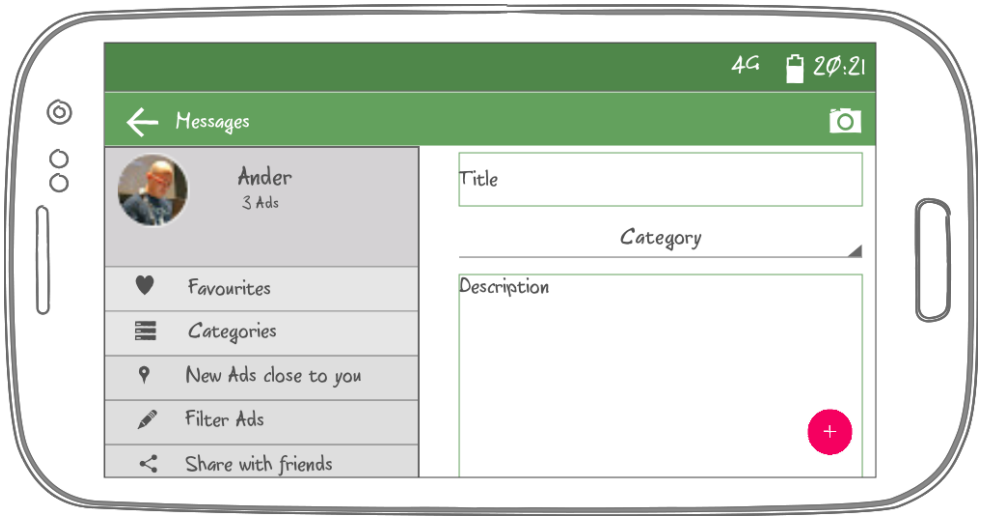
Horizontal orientation for tablet.



Add Screen

Lets to add a new Advertisement based on a title, category, description, and optionally a photo

Horizontal orientation for tablet



.

Detail Screen

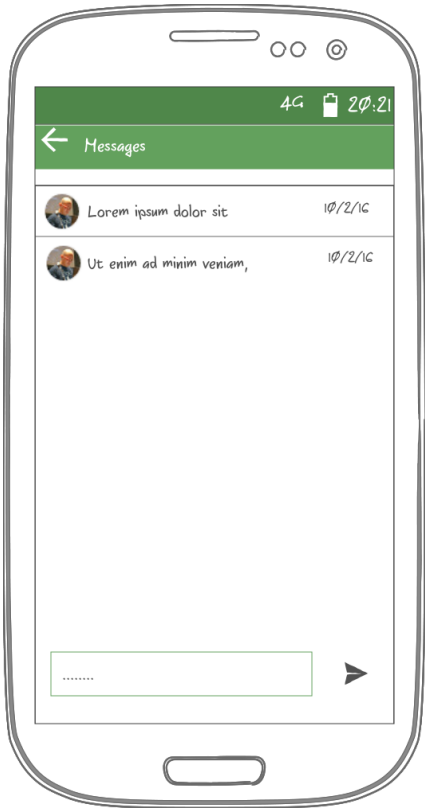
Advertisements Details, using parallax view mode, includes:

1. Full image view, centercrop
2. Title
3. Description
4. FAB button to chat with publisher
5. Share button
6. favorites button.

FAB button opens Message Screen (see next).

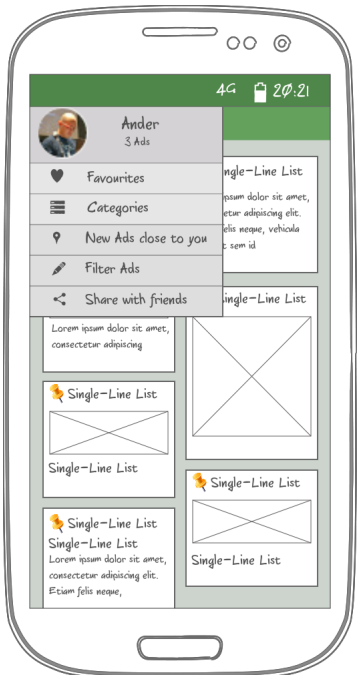
Horizontal orientation for tablet.

Messages Screen



Lets user to send messages to publishers, and view later.

Navigation Drawer Screen

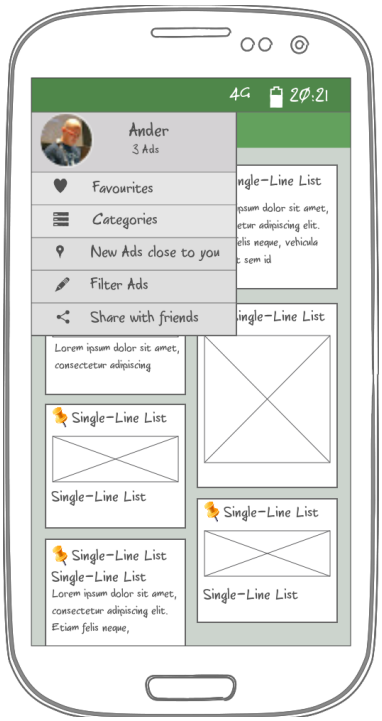


It shows:

1. Profile info
2. favorites
3. Categories
4. New adds close to your location
5. Filter Ads

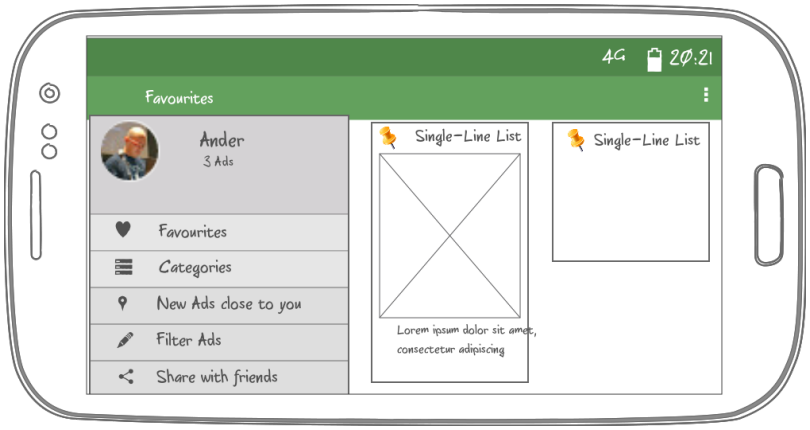
Add as many screens as you need to portray your app's UI flow.

favorites Screen



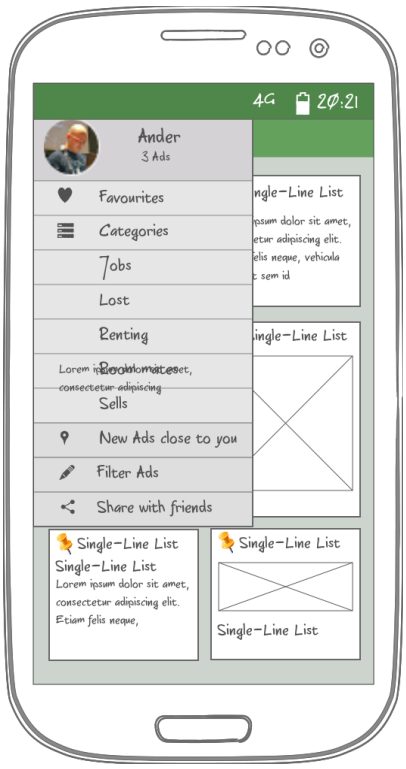
It shows those advertisements marked as favourite by user. User could select each one of

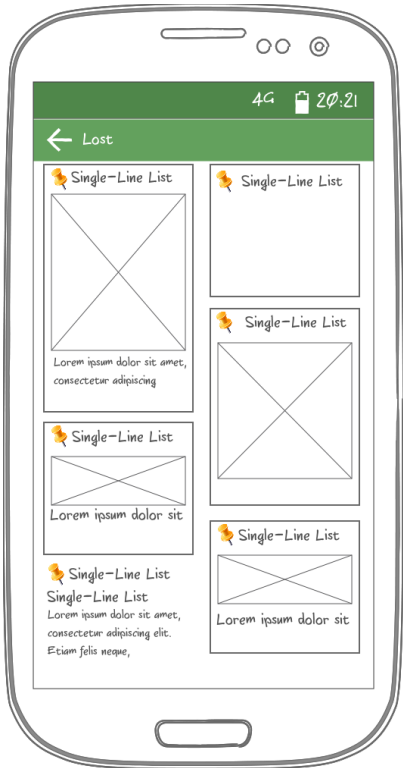
Horizontal orientation for tablet.



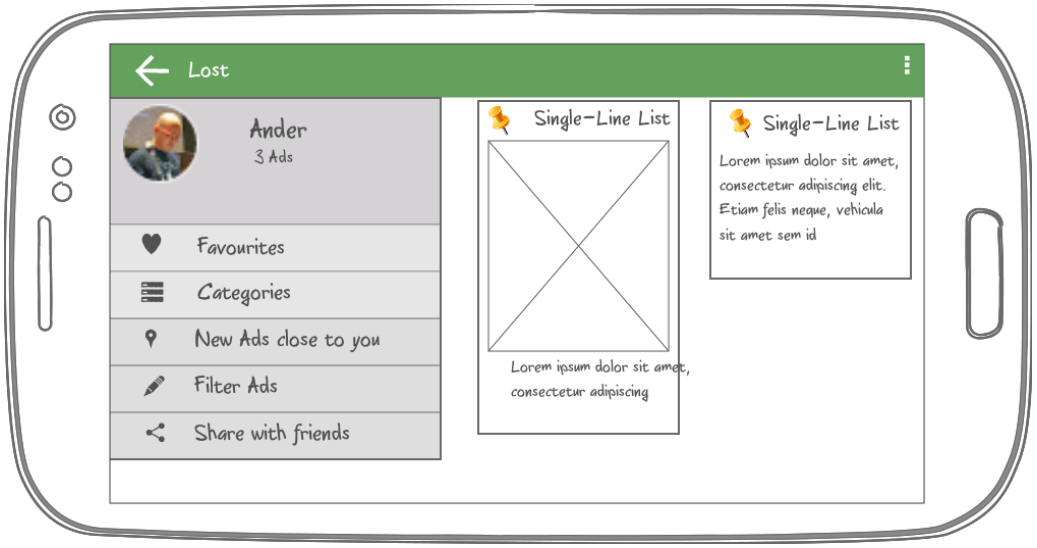
Categories Screen

Lets to view advertisements for an specific category, e.g.: category,e.g.:Lost

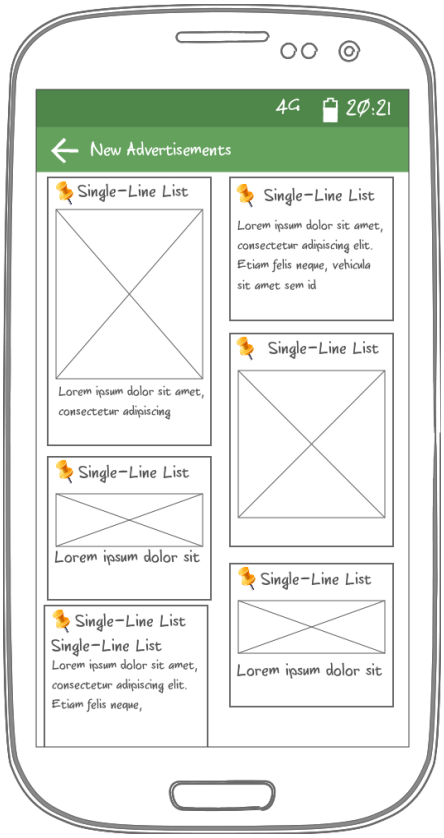




Horizontal orientation for tablet.

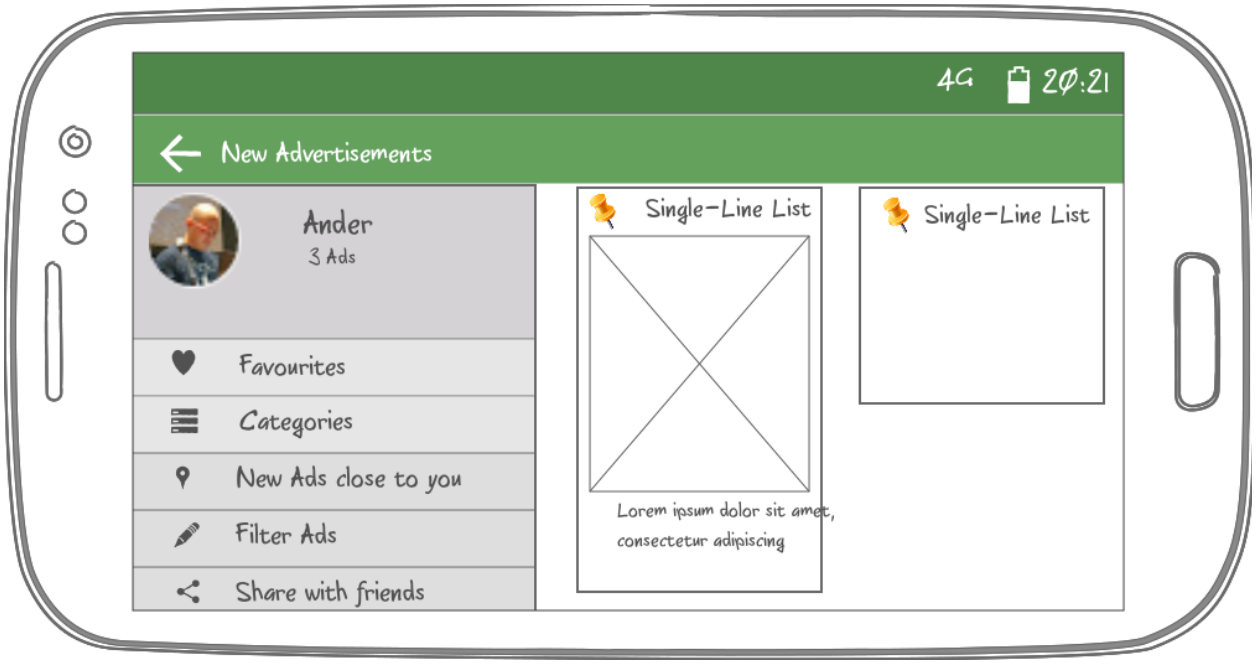


Screen New Advertisements



Shows latests ads

Horizontal orientation for tablet.



Key Considerations

Data persistence

From mobile point of view, the app will use the following approaches:

1. SQLite: Its own content provider for storing chat messages on a SQLite database.
2. Shared Preferences: For storing internal app data.

From Backend point of view:

1. Google Datastore to handle Advertisements and users persistence.
2. Google Storage to handle image persistence.
3. A web service deployed on Google App Engine will be developed using Jersey to implement RESTFul services.

Corner cases in the UX.

- For mobiles, all Up navigation button and back button will return to the parent activity in vertical orientation.
- For tables, all Up navigation button at right fragment, return to the parent fragment..

Libraries used

1. Volley, included as a module for implementing http* connections.
2. com.github.afollestad.material-dialogs: For implementing customized material dialogs.
3. de.hdodenhof:circleimageview: Circle imageview used at Drawer profile to draw navigation drawer profile image view.
4. Support libraries:
   com.android.support:appcompat-v7
   com.android.support:support-v4
   com.android.support:design
   com.android.support:cardview-v7
   com.android.support:recyclerview-v7
5. com.google.code.gson: Gson lib for managing communication with GAE middleware.
6. Play services libraries:
   com.google.android.gms:play-services: Google Play services library to retrieve user's profile image at drawer navigation.
   com.google.android.gms:play-services-auth Auth play services library for implementing oauth2 validation

Google.android.gms:play-services-gcm Google Cloud Messaging Play Services library for chat feature**.**

7. Picasso and handler to download images from https connections
   com.squareup.okhttp:okhttp
   com.squareup.okhttp:okhttp-urlconnection
   com.squareup.picasso:picasso
8. Com.google.apis:google-api-services-storage:v1-rev66 Cloud Storage JSON API for implementing Google storaging capabilities
9. com.google.http-client:google-http-client-android: For getting GoogleAuthorizationCodeFlow at Google Maps activity (LocateAdsMapsActivity)
10. com.google.oauth-client:google-oauth-client-jetty: For implementing LocalServerReceiver at Google Maps activity (LocateAdsMapsActivity)
11. Com.google.oauth-client:google-oauth-client-java6: for implementing GooglePromptReceiver at Google Maps activity (LocateAdsMapsActivity)
12. com.github.darsh2:MultipleImageSelect:For implementing a grid wherever select multiple image.
13. fr.avianey.com.viewpagerindicator:library: For implementing a viewpager for images.
14. com.squareup:otto Otto event bus for communication different parts of your application
15. com.oguzdev:CircularFloatingActionMenu: For customizing a floating action button with sub-buttons

Required Tasks

This is the section describes the whole required tasks. The most of them depends one on each other, so the order they have been shown is not so relevant.

Task 1: Project Setup

- Google Cloud platform setup: Google App Engine, Google Data Storage, Google Storage, Google Cloud Message credentials.
  - Define Data Storage model's schema
  - Define Storage properties.
  - Create Jersey RESTFul services

- Android Studio:
  - Create project
  - Configure gradle properties, version, variants, libraries, etc.
  - Zip align the apk at app gradle file.
  - Enable multidex support

Task 2: Register app with Google Cloud Console

Register the app with Google Cloud Console.
In order to access to user's validation and picker accounts based on OAUTH2 validation, register the app with the Google Cloud Console, provide app's package name and SHA1 fingerprint, next:

Task 3: Implement MainActivity app procedures

- Checks device connectivity
- Gets User's validation with OAUTH2.
- Initialize Google Api Client with access to the Google Location Services API
- Check User location settings providing resolution required if not.
- Checks location capabilities gps and network provider capabilities
- Configure app for using Google Cloud Messaging:
    - Check whether Google play services is available on device providing resolution required if not
    - Init Google Cloud Messages broadcast receiver
- Load geo located advertisements
- Init Drawer menu and profile settings. Drawer menu manage access to the whole app functionality

Task 4 : Implement SQLite data skeleton and providers

SQLite: SQLite related classes: Defines chat capabilities using SQLite as storage as internal storage, contract, schema,...

- Dao
    - ChatDataSource: Defines Chat data source
- Model
    - Chat: Defines Chat model
- BulletinContract: Defines database contract
- BulletinProvider: Defines database content provider.
- SQLiteHelper: Defines SQLite helper methods.

Task 5: Implement UI for Each Activity and Fragment

Activities:

- AdvertisementGridActivityFragment: A placeholder fragment containing a grid of geolocated advertisments.
    - Get User profile data
    - Build recycler grid view and adapter for showing advertisements.

- CreateAdvertisementActivity: Lets to create a geo located advertisement.
    - Build interface to create advertisements.
    - *Instantiate a ViewPager and a PagerAdapter.*
    - *Instantiate a d*ot Page indicator for images
    - Build Circular Floating Action Button, a customizable floating action button with sub-menus for:
        - Settings:
        - Action done: finish Create Advertisement activity
        - Location: Let's specify another location for advertisement instead current location
        - Add photos to advertisement
        - Delete photos from advertisement
    - Build Image pickup from camera or gallery.
    - Implement storage capabilities against Google Storage and Google Datastore

- EditAdvertisementActivity: Lets to edit a geo located advertisement.
    - Get advertisement data to be modified.
    - *Instantiate a ViewPager and a PagerAdapter.*
    - *Instantiate a d*ot Page indicator for images
    - Build Circular Floating Action Button, a customizable floating action button with sub-menus for:
        - Settings: Opens submenu actions
        - Action done: finish Create Advertisement activity
        - Location: Let's specify another location for advertisement instead current location
        - Add photos to advertisement
        - Delete photos from advertisement
        - Delete advertisement
    - Build Image pickup from camera or gallery.
    - Implement storage capabilities against Google Storage and Google Datastore
    -

- LocateAdsMapsActivity: Lets to specify (by using markers) a different advertisement location than current.
    - *Authorization code flows*

        - *Add Snackbar to show instructions*
        - Build onMapReady functionalities, current advertisement location, etc.
- ShowAdvertisementActivity: Shows advertisement content.
    - *Get sharedpreferences bookmark status*
    - *Int images ViewPager and a PagerAdapter*
    - *Init Viewpager dot indicator*
    - *Init CircularFloatingActionMenu* , a customizable floating action button with sub-menus for:
        - Settings: Opens submenu actions
        - Chat: Lets chat with publisher
        - Share: Lets share a description of the advertisement with your contacts
        - Bookmark the advertisements.
    - *Retrieve Advertisement data*
- *ShowImagesActivity: Lets to show advertisements images on fullscreen mode*
- NoConnectivity: Activity shown whenever user lose network connectivity.

Fragment related classes:

- AdvertisementGridActivityFragment: *A placeholder fragment containing a grid of Advertisements.*
- ChatActivityFragment: *A placeholder fragment containing the chat.*
- CreateAdvertisementActivityFragment: *A placeholder fragment containing the Advertisement creation.*
- EditAdvertisementActivityFragment: *A placeholder fragment containing the Advertisement edition*
- ShowAdvertisementActivityFragment: *A placeholder fragment containing the Advertisement showing mode.*
- ShowImagesActivityFragment: *A placeholder fragment containing the Advertisement Image showing mode*

Task 6: Implement AsyncTasks

Define and implement async tasks

- GetUserProfileTask
    - *Get user account profile*
- RequestUserAdvertisementsTask:: *Load geolocated advertisements from remote server asynchronously*
- StorageAsyncTask: Upload images to Google Storage asynchronously

TAsk 7: Define Interfaces

Interfaces:

- downloadTaskDone: Interface used to send data once data has been loaded or throw an exception in case of error.
- onDataSendToActivity: Interface to send User's Advertisements from RequestUserAdvertisementsTask
- onStorageCallback: Interface that defines callback methods for onStoragePostActions and onStoragePostActionsFailure
- customDialogFragmentInterface: Interface that defines callback methods for onAction, onAction(String targetMethod, and onCancel functionalities
- onAdvertisementClickedListener: Interface that lets communicate Advertisement's recycler view with MainActivity.
- onAdvertisementHolderClickListener: Interface that lets communicate Advertisement holder with AdvertisementRecyclerViewAdapter

Task 8: Define and implement Google Cloud Storage classes

Google Datastore and Storage related classes.

- CloudStorage: Defines the whole functionalities to be executed by StorageAsyncTask with Google Cloud Storage: uploadFile, downloadFile, deleteFile, createBucket, deleteBucket, listBuckets, listBucketObjects, getStorate, etc.
- Storage Models: Google Datastore request and response data models
    - AdModelRequest: Advertisement model request class
    - AdModelResponse: Advertisement model response class.

Task 9: Define and implement Google Cloud Messaging classes

GCM related classes.

- MyGcmListenerService: Service that manage messages reception, notification creation, etc.
- MyInstanceIDListenerService: *Handle the creation, rotation, and updating of registration tokens*
- QuickstartPreferences: GCM constants definition.
- RegistrationIntentService: Manage GCM registration, subscription, also internal server registration processes.

Task 10: Views related classes

Views related classes.

- Adapter
  - Activity
    - AdvertisementRecyclerViewAdapter: Recyclerview adapter to manage view holder on grid.
  - Drawer:
    - ExpandableListAdapter: Expandable submenu for drawer menu.
  - FullScreenSlidePagerAdapter: Adapter for images full screen mode
  - MediaStoreDialogAdapter: Adapter to manage image's picking process from gallery or camera.
  - RecyclerViewChatAdapter: Adapter to manage view holders on chat.
  - ScreenSlidePagerAdapter: Adapter to manage viewholder for images from Google storage.
- Holder
  - AdvertisementHolder:Defines Advertisement holder properties
  - ChatHolder: Defines Chat holder properties.
- Model
  - Drawer
    - ExpandedMenuModel: Defines expanded menu model
    - Message: Defines Chat message parcelable object.

Task 11 Util classes

Util classes.

- Util: Class with common methods.
- Config
  - ConfigProperties: Util class that manages config.properties local resoures treatement.
- eventBus
  - BusProvider: Defines Otto event bus singleton pattern.
- Gae
  - CustomJsonObjectRequest: Lets getting json request without returned object
  - GAEUtils: Defines GET and POST methods to be used with GAE
- Image
  - ImageUtils: Manage Scale down the picture to image view's container
  - PicassoTrustAll: Lets to enable https image downloads from Google Storage
- PlayServices:

    - PlayServicesUtil: Checks whether Google Play services are available on the device
- Constants: Constant definition container for the whole application.
- CustomDialogFragment: Lets to define customizable dialog fragments and callback methods depending of the action selected.
- CustomProgressDialog: Lets to define customizable progress dialogs.