GitHub Username: xenonzx

Omla

Description

Omla is an app that provides the currency exchange rates of banks operating in Egypt . Today Each bank available has its own currency exchange rate different than other banks since Egypt lifted up government Restrictions on currency and allowed Its currency to float.now the exchange rates are freely determined by the market. The purpose of Omla is to provide the user with banks currency exchange rates so that he can determine which bank to go to to get the best value for his money. The translation of the word “currency” in Arabic is the word “عملة” and it's pronounced “Omla” hance the name of the application. The app has free version(with ads) and paid version(ad-free).

Problem:
The user wants to know the current foreign currency exchange rate with regard to the Egyptian Pound (EGP) and which bank to go to so that he get the best price when exchanging from or to the Egyptian pound.

Proposed Solution:
Design an app that provides the user with banks currency exchange rates.The app should provide sorting option so the user could pick the highest sell rate or the highest buy rate according to his own need. The app should contain updated exchange rates from banks and location of the bank branches on map. The app also should provide a collection widget that can display rates on home screen.

Intended User

Tourism , Solid remittances(remittances of Egyptians working abroad) and Export revenues are among the sources of foreign currency. that makes the intended users for this application tourists, families of Egyptians working abroad and exporters.

Families of Egyptians working abroad are most likely to use the app on monthly basis and for a long duration of time (as long as their relative is working abroad).

Tourists are most likely going to use the app as they probably going to make the currency exchange once or few times along their tour.

Exporters are most likely to use the app frequently through their daily work but most likely be restricted with some banks that provide deals and Bank facilities for them.

Features

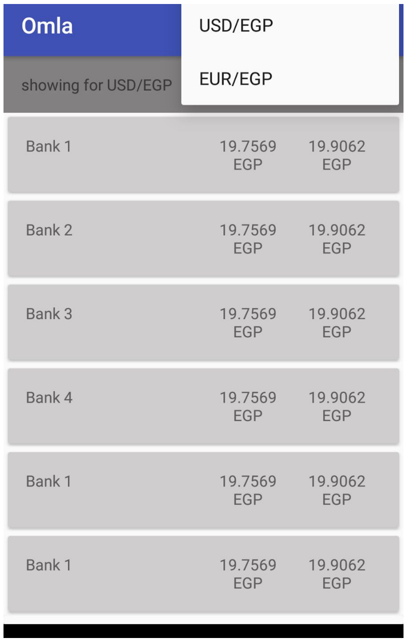
List the main features of your app. For example:

- Displays bank exchange rates for multiple currencies

- Sorting option for best sell and buy rates
- Home screen widget to display rates
- Map to show bank branches locations

User Interface Mocks

Screen 1

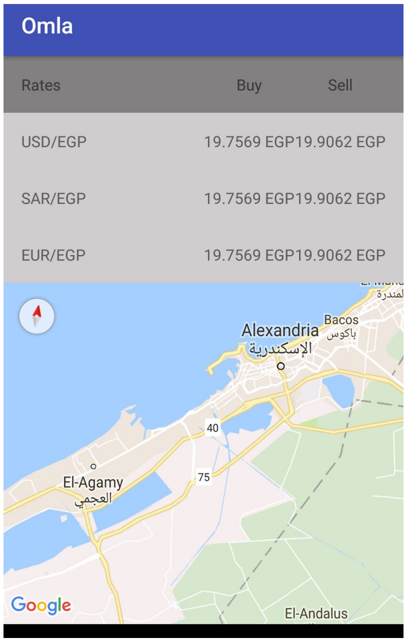


showing for USD/EGP

Bank 1	19.7569 EGP	19.9062 EGP
Bank 2	19.7569 EGP	19.9062 EGP
Bank 3	19.7569 EGP	19.9062 EGP
Bank 4	19.7569 EGP	19.9062 EGP
Bank 1	19.7569 EGP	19.9062 EGP
Bank 1	19.7569 EGP	19.9062 EGP

Main activity showing list of banks buy and sell rates for the USD -as an example- . the user can change the currency from options menu . each list item is clickable and leads to bank details screen.

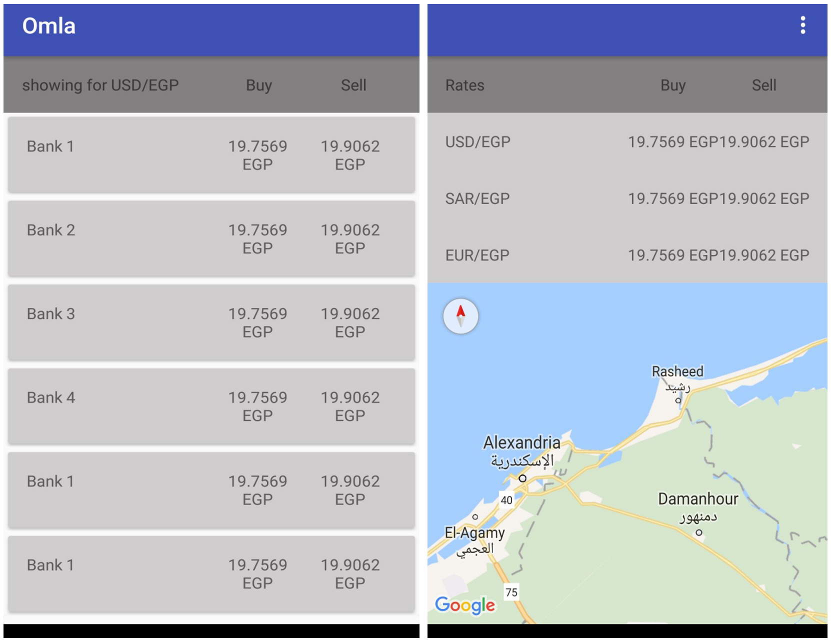
Screen 2



Rates	Buy	Sell
USD/EGP	19.7569 EGP	19.9062 EGP
SAR/EGP	19.7569 EGP	19.9062 EGP
EUR/EGP	19.7569 EGP	19.9062 EGP

Bank details activity showing all the currency rates of the bank and a map to show near by bank branches.

Screen 3



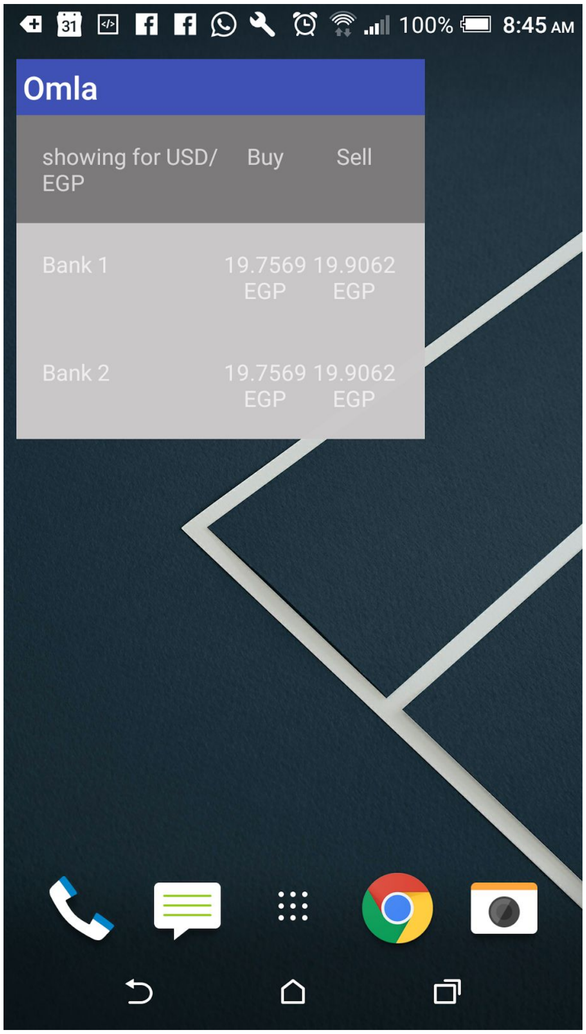
Omla

showing for USD/EGP	Buy	Sell
Bank 1	19.7569 EGP	19.9062 EGP
Bank 2	19.7569 EGP	19.9062 EGP
Bank 3	19.7569 EGP	19.9062 EGP
Bank 4	19.7569 EGP	19.9062 EGP
Bank 1	19.7569 EGP	19.9062 EGP
Bank 1	19.7569 EGP	19.9062 EGP

Rates	Buy	Sell
USD/EGP	19.7569 EGP	19.9062 EGP
SAR/EGP	19.7569 EGP	19.9062 EGP
EUR/EGP	19.7569 EGP	19.9062 EGP

Tablet version showing master and detail view.

Screen 4



Omla

showing for USD/ EGP	Buy	Sell
Bank 1	19.7569 EGP	19.9062 EGP
Bank 2	19.7569 EGP	19.9062 EGP

Widget on home screen showing list of banks with their corresponding rates for the preferred currency. By clicking on the header containing the app name the user is directed to main activity and by clicking on list item the user is directed to selected bank details activity.

Key Considerations

How will the app get banks exchange rates?

By calling the api api.curates.club that is provided from curates.club .

How will your app handle data persistence?

The app should keeps only the last updated version of bank exchange rates.
The app should keep the user selected currency option in shared preferences.

How will your app keep the data in sync?

Data sync jobs should be scheduled periodically by using JobDispacter. The jobs should execute when the network is available.

How will activities and widgets react if data changed?

The activities are registering Loaders to load data therefore once the data is changed/updated the content provider notify observers by calling getContext.getContentResolver().notifyChange(Uri uri,ContentObserver observer) also the content provider should send a data changed action that is registered by widgets ,once data changed action is received by the widget the widget should update its view.

Describe any corner cases in the UX.

The user pulls down the list of rates in main activity a swipe to refresh view shows progress and sync-immediately job is started.

The user opens the app for the first time and has no connection, the user is presented with a message to turn on his wifi and sync job should start when he is connected to the internet.

Nothing should happen if the user clicked on rates inside the bank detail activity as he is already watching all the details of the bank.

Describe any libraries you'll be using and share your reasoning for including them.

Butter knife for view binding.
Okhttp to handle network http requests.
Retrofit to mainy to use okhttp to make network requests and get type safe object from response.
Crashlytics to integrate crash reporting in the app.

Describe how you will implement Google Play Services.

Admob is used in the free version for monetization. The free version should show banner adds in main activity(the list of banks exchange rates) and detail activity (the activity showing all the detail of the bank).

The app should use google Maps to show the branches available for that bank.

The app should use location services to set the camera of map view to the users current location.

Next Steps: Required Tasks

Task 1: Project Setup

Initiate project

- Add libraries to gradle.
- Initializing git repo on github.

Task 2: Implement UI for Main Activity and Detail activity on mobile

Building main activity with dummy data.

- Populate main activity with dummy data.
- Build detail activity as a fragment activity showing single fragment with dummy data.
- Inflating menu and and adding actions on option items in main activity.
- Adding pull to refresh layout.
- Adding sorting option on dummy data.
- Handel accessibility by adding content description.
- Handling localization by designing Right-to-Left layout and showing which strings are translatable in xml.
- Keeping good formate by showing currency formats and dates formats.
- Handle empty results response.
- Handel no connection message.

Task 3: Sharedpreferences Utils

Building shared preferences utils to keep the user preferences.

- Keep prefered currency exchange rate.

Task 4: Maps integration

Using dummy data display bank branches.

- Google Maps integration.

Task 5: Tablet version

Determining the layout to be displayed given the width of the screen.

- Display master and detail layout on tablet version.

Task 6: Content provider

Build content provider to provide actual data.

Task 7: Data sync

Schedule data sync jobs to pull latest banks exchange rates.

Task 8: Paid and free versions

Create paid and free versions from the app giving each a different application id.

- Handling build variants.
- Integrating AdMob.

Task 9: Widget

Creating the collection widget.

- Create widget provider class.
- Create remote service factory to populate the widget.
- Create widget configuration xml.
- Create widget layout.