PORTFEL

Description

Track stock market using Yahoo finance data and read market news from Yahoo and Reuters. Choose a stock and follow it using an widget on a home screen. Create a stock portfolio and track its performance. Get personalized news and alerts.

Intended User

A financial investor who is interested in stock market.

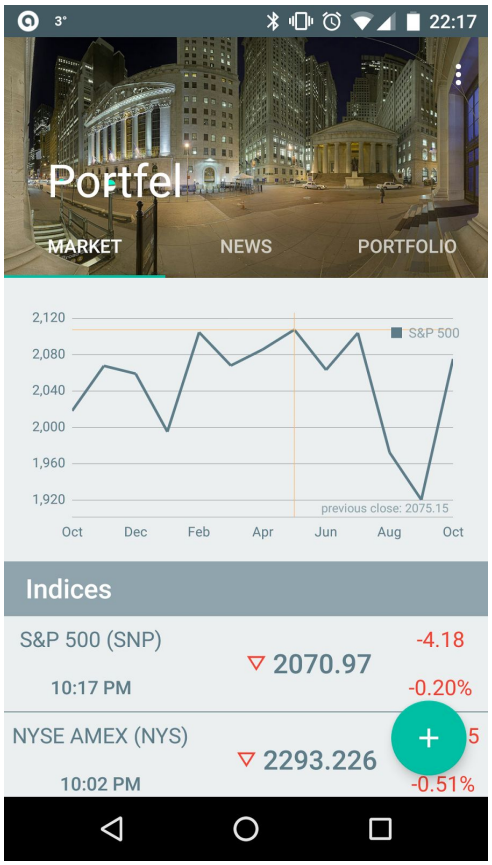
Features

Main features of the app:

- Track list of indices and stocks.
- Get detail information about chosen index or stock (including candlestick chart for a stock and news related to this symbol).
- Choose a stock and follow it on a home screen widget.
- Read market news.
- Track performance of your portfolio.
- Get personalized news and alerts.
- Use app when offline.

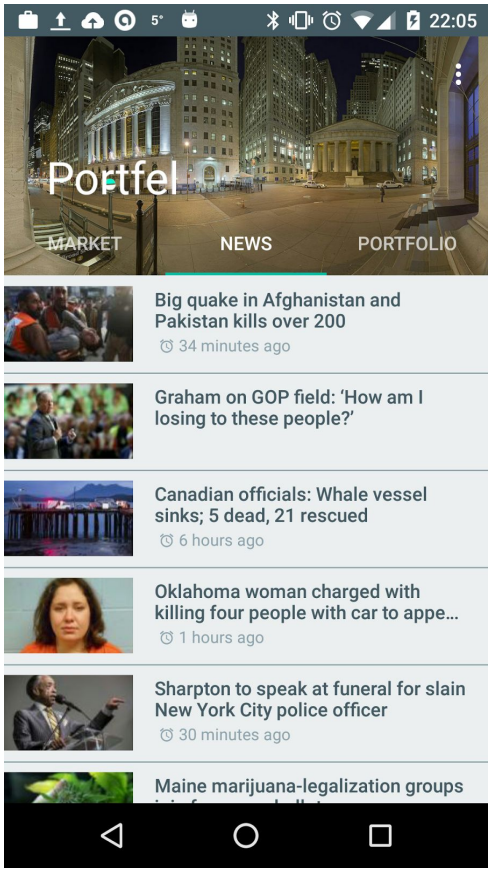
User Interface Mocks

Screen 1. Front screen - Market tab



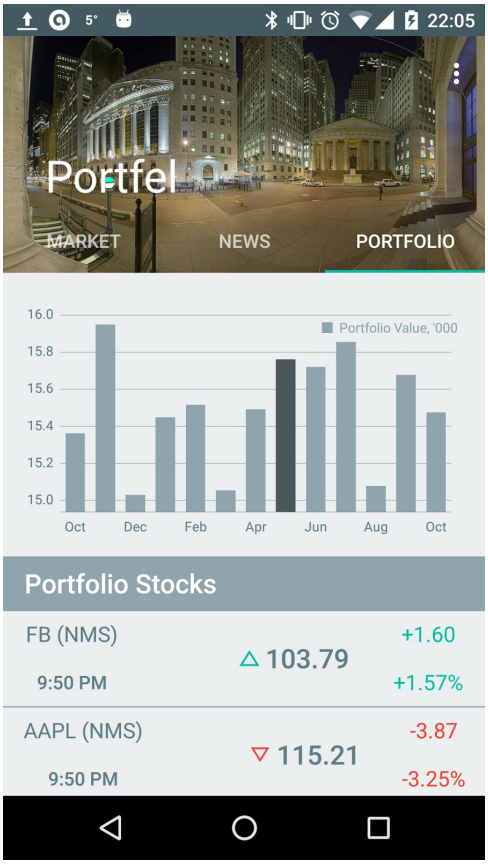
- The first tab - Market. Here a user can track indices and stocks.
- S&P 500 has a chart and other indices have only value and changes.
- The list of stocks is independent from user's portfolio - a user can include an arbitrary stock in this list.
- This screen has FAB button to add a stock to the list.
- A user can drill-down to details about index or stock.
- Screen has an overflow menu with Settings action.
- A user can update information with a swipe down.
- Toolbar is scrollable (not collapsible) with parallax effect.
- A user can also delete a symbol using long click and an action on snackbar.
- Possible additional features: choice of index for chart, interactive chart with drill-down to full-screen chart, etc.

Screen 2. Front screen - News tab



- The second tab - News. This screen contains news feed that we fetch from our news provider API.
- Each news item contains: an image (if provided), a title of a news and it's date and time with a clock icon.
- We format date and time in a user friendly manner like so: 10 minutes ago, 2 hours ago etc.
- A user can click on news and get details within an app with the help of WebView;
- Screen has an overflow menu with Settings action.
- A user can update information with a swipe down.
- Possible additional features: more sources for news and chooser for type of news, possibility for a user to add new sources for RSS or Atom feed.

Screen 3. Front screen - Portfolio tab



- The third tab - Portfolio. Here a user tracks its portfolio performance.
- Screen contains yearly performance bar chart for value of the portfolio.
- Screen also contains list of stocks in the portfolio.
- Screen has an overflow menu with Settings action.
- A user can update information with a swipe down.
- Disclaimer: this is rather a prototype than a real portfolio, we don't really implement functionality of a real portfolio.

Screens 4-5. Detail screen - stock, index

- This screen contains detail information for a stock that a user can get if she clicks on market or portfolio tabs.
- The screen for an index is the same except we use linear chart, not candlestick chart.
- Screen contains: a) the same information about stock that is presented on market or portfolio tabs, b) a chart and c) news related to the company.
- Screen also contains a FAB button to share some information about the stock or index.
- We use the same news items that are presented on news tab.
- A user can click on news and get details within an app with the help of WebView;
- Screen has an overflow menu with Settings action.
- We add an ad banner to the bottom of the screen.
- Possible additional features: interactive chart with drill-down to full-screen chart, daily and monthly charts etc.

Screen 6. Widget on a home screen

The app has a widget that can be installed on a home screen. The information on a widget is the same as on an stock item of a market tab. When widget is clicked the market tab will be shown.

Screen 7. Settings

- A user can choose frequency for an update service and turn off notifications.
- Possible additional features: choose a symbol to be shown on a home-screen widget.

Key Considerations

Data persistence

We use a custom service (extends IntentService) to fetch data from API to a database. We build a content provider (we don't use 3d party libraries for this) and CursorLoader to supply info to UI. We use content provider for market data. We just fetch news from network using AsyncTask.

Describe any corner cases in the UX

- A user can add a stock to a list on market screen using FAB or delete it using long press and action on snackbar.
- A user can share information about stock or index from detail screen using FAB.
- On all 3 tabs a user can update information using swipe down.
- Our 3 tabs on the main screen use ViewPager so we can swipe from one tab to another.
- News and stocks items has appropriate detail screens as described above.
- A user can switch-off notifications and set frequency for update service.
- App includes content descriptions in all appropriate cases.
- App keeps all strings in a strings.xml file and enables RTL layout switching on all layouts.

Libraries

- **Image downloading.** We use picasso to download images for news feed. This library solves a lot of problems: networking, caching, handling of configuration changes etc.
- **Design.** We use material design in our app. When using libraries we are able to handle most of the tasks simply by writing xml: appcompat, design, recyclerview, cardview and necessary support libraries. So our activity extends AppCompatActivity and app theme extends Theme.AppCompat. We also use Toolbar and all related classes from design library (AppBarLayout, CoordinatorLayout, CollapsingToolbarLayout etc.) to make Toolbar scrollable and collapsible.
- **Parsing.** We use libraries to parse Yahoo Finance .csv files and news RSS feed: yahoofinance and earl.
- Chart. We use MPAndroidChart to plot charts in market, portfolio tabs and detail screen. We use line chart, bar chart and candlestick chart.

Google Play Services

- We use Admob and Analytics in our app. We show banner ad in detail screens. We track all user activity.

Notifications

- We notify a user about Reuter market news once per day. We set high priority for this notifications and add BigText style to it. A user can opt out from notifications using Settings.

Next Steps: Required Tasks

Task 1. Libraries setup

We add dependencies to build.gradle for libraries mentioned above.

Task 2. Data persistence

We create a contract, DB helper and a provider class. We don't create JSON parsers and networking classes to fetch data from API - we use libraries mentioned above.

Task 3-5. Service, Notifications and Settings

- We create an update service with default frequency - 12 hours. A user can change that in settings. We fetch data from Yahoo using yahoofinance library and store them in database.
- We also use service to make notifications. We store time for previous notification in prefs and check if elapsed time is mor than 24 hours. A user can opt-out of notifications in settings.

Task 6. Create UI

- Create main screen with 3 tabs and add ViewPager. Add Toolbar and make it collapsible using design library. Add FAB button and overflow menu. Add content description and RTL.
- Add methods for LoaderCallbacks on market and portfolio tabs fragments and for service call on news tab fragment.
- Construct class for RecyclerView adapter. Get images using Picasso.

Task 7. Widget

We create a widget using subclasses of AppWidgetProvider and IntentService (and register them in manifest). We fetch information about stock using yahoofinance library in onHandleIntent() and set remote views. We provide some information about widget in xml file.

Task 8. Analytics

We set up analytics using subclass of Application to create only one tracker and start this tracker in MainActivity. We set UserId and some flags in xml file.