Capstone – Vehicle expenses tracking

Capstone – Vehicle expenses tracking

GitHub Username: kalxasath
GitHub Profile: https://github.com/kalxasath

Capstone – Vehicle expenses tracking

Description

Simple and effective application to track vehicles consumption and expenses. The application supports more than one vehicle. For each vehicle, the application will show the total running costs.

For each vehicle the users can choose a representative photo from gallery, a name, the make, model, plate no, initial mileage, the distance unit (Kilometers, Miles, Hours), tank volume, the volume unit (Liters, Gallons), notes.

Each consumption or expense refers to a particular vehicle, and the stored data are the type, subtype, date, odometer, fuel quantity (if the type is refueling), amount, notes.
The types and subtypes are: (Refuel)[Full], (Bill)[Parking, Toll, Insurance], (Service)[Basic service, Damage].

For each vehicle in the dashboard screen the user can see the vehicle's total running costs divided into the 3 type categories (Refuel, Expenses, and Service).

Also, there is a Youtube search option where the users without leaving the app can direct search related videos according to his search criteria.

In the phones home screen, the user can add a widget for a particular vehicle to have a look of the vehicle's total running costs without to enter in the app. The user can add as many widgets as he likes. Each widget displays the data from the selected vehicle on widget creation.

Intended User

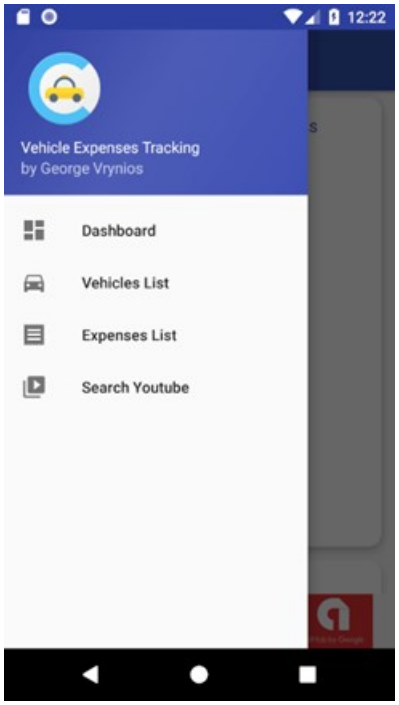
The app is intended for vehicle owners, giving him the ability to track the vehicle's consumption and expenses and provide him with information about the vehicle's total running costs.

Features

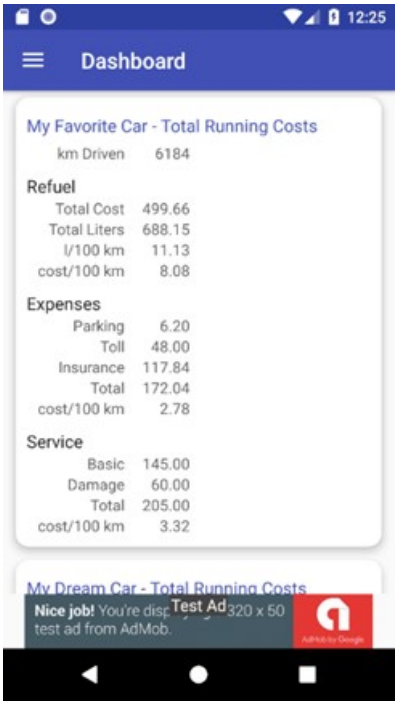
- Saves information about users vehicles
- Display a list of all of his vehicles
- Track consumption and expenses
- Display a history list of the consumptions and expenses
- Display for each vehicle analytical total running costs
- In-app search YouTube for related videos according to his search criteria
- Home Screen Widget
- At the end of the dashboard total running cost information there will be an AdMob horizontal banner.

Capstone – Vehicle expenses tracking

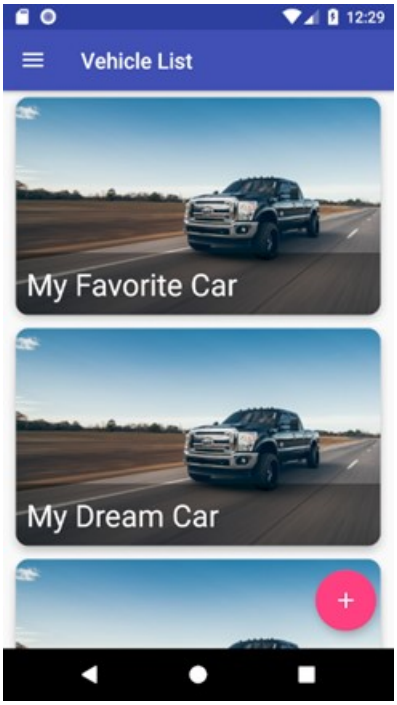
User Interface Mocks



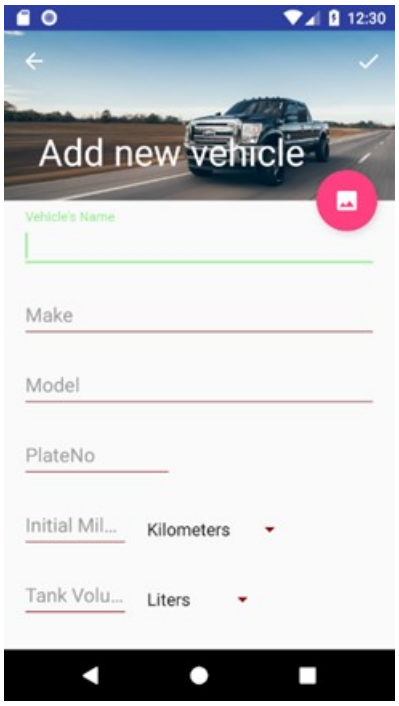
Application Menu Drawer



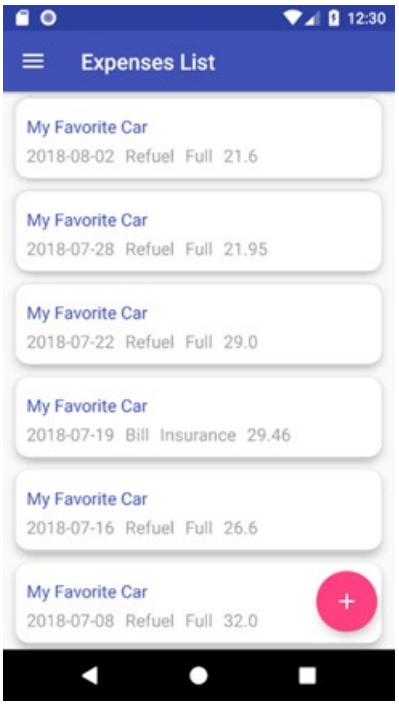
Dashboard



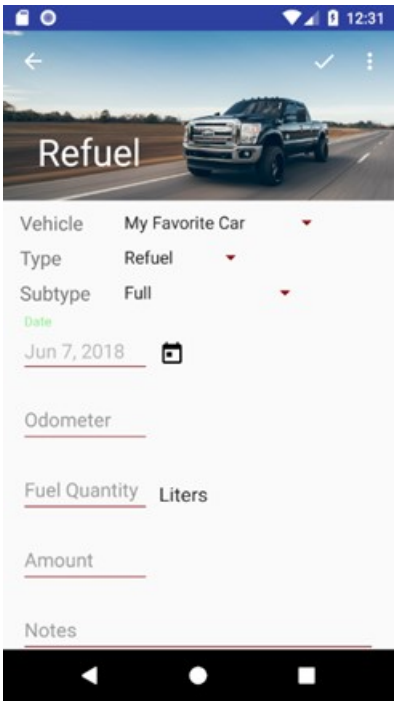
Vehicle List



Vehicle Entity

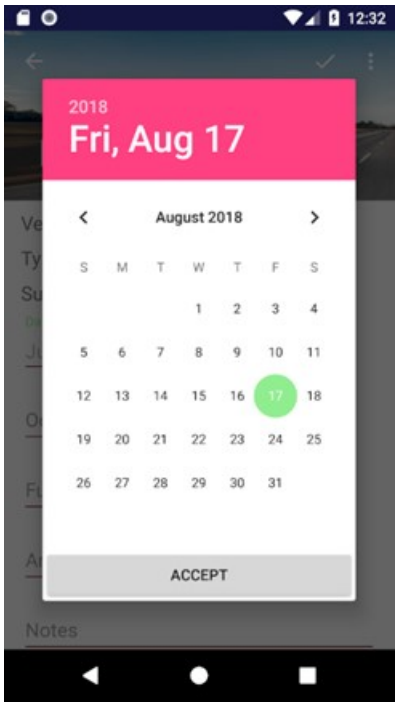


Expenses List

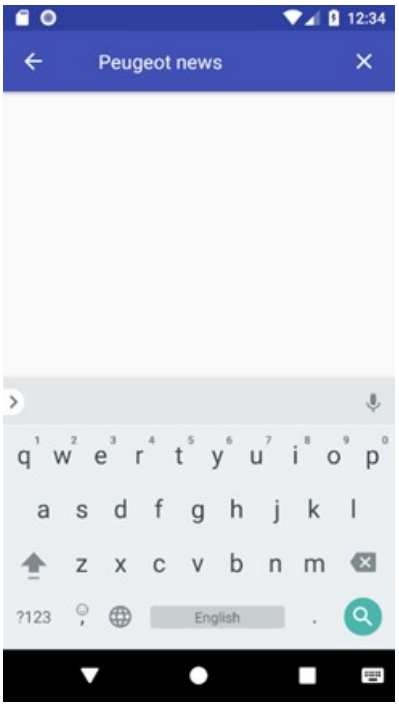


Expense Entity

Capstone – Vehicle expenses tracking

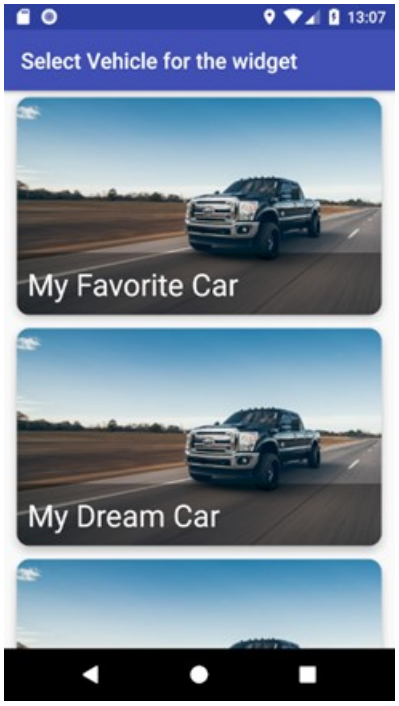


Date picker

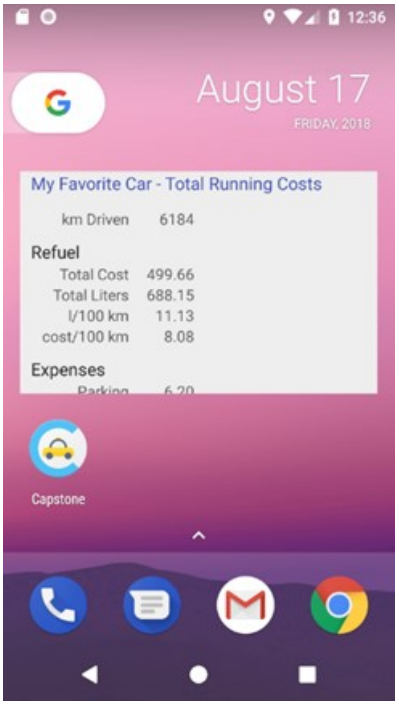


Search YouTube

Search YouTube Results



Home Screen Widget configuration screen



Home Screen Widget

Capstone – Vehicle expenses tracking

Key Considerations

How will your app handle data persistence?

1. The application stores & retrieve all data entities via a Content Provider in SQLite

2. The application uses complex SQL queries using the “CapstoneDBHelper” extended from “SQLiteOpenHelper”

3. The application uses Shared Preferences to store specific information for each home screen widget, the vehicle ID

Describe any edge or corner cases in the UX.

- The back button on the title bar of the entities screens navigates back to the parent screen, i.e., Vehicles List or Expenses List

- Moving back from the four main activities, i.e., Dashboard, Vehicles List, Expenses List or YouTube search returns back to the home screen.

- If the user taps on an item in the Vehicles List or Expenses List, then the corresponding entity screen opens to edit or delete the entity.

- If the user taps on an item in the YouTube search results, then the app tries to play the selected video via the installed YouTube app, if such an app is not installed in the phone, then the app tries to play the video via the web. If the app fails to start the YouTube app or the web, then it shows a toast that informs the user with the message ‘An associated YouTube player couldn't be found!’

- If the user taps on the home screen widget or the dashboard list, nothing happens.

- If the user selects the capstone widget to put it in the home screen, then a configuration widget screen with the list of all the vehicles is displayed. If the user taps on a vehicle, the widget is displayed on the home screen displaying the vehicle’s total running costs. Otherwise, the screen closes, and no widget is displayed on the home screen.

Capstone – Vehicle expenses tracking

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

1. Google's Compatibility Library is used to ensure compatibility with older Android versions
2. Picasso is used for loading, caching and displaying the application images
3. Butterknife is used to eliminate any findViewById calls in adapters and activities
4. Google's play-services-ads is used to display the AdMob ads

Describe how you will implement Google Play Services or other external services.

1. AdMob banners appear at the end of the dashboard screen
2. YouTube search service is implemented via the 'https://www.googleapis.com/youtube/v3' API calls. The reviewer needs his API key to test the functionality.
3. YouTube videos playback goes through the installed YouTube app or the web service 'https://www.youtube.com/watch'

Google services were choose from the 'https://en.wikipedia.org/wiki/Category:Google_services'

Capstone – Vehicle expenses tracking

Next Steps: Required Tasks

Task 1: Project Setup

1. Create a clean project via the android studio

2. Add required google libraries for backward compatibility, design, recycleview, cardview, constraint layout

3. Add required google libraries for the AdMob ads

4. Add required libraries for Picasso and butterknife

5. For the four main activities, i.e., Dashboard, Vehicles List, Expenses List or YouTube search create a layout container that supports the 'DrawerLayout' in this layout include all the necessary components for the property use of the drawer, toolbar, floating action button, navigation view.
   The layout container should be a 'CoordinatorLayout' component, name it 'layout_container.' Later each activity inflates in this container its views.

6. For the two entity activities, i.e., Vehicle, Expenses create a layout container that supports the 'CoordinatorLayout'. This layout includes all the necessary components for the property use of the AppBarLayout, CollapsingToolbarLayout with ImageView, Toolbar.
   The layout container should be a 'NestedScrollView' component, name it 'layout_container.' Later each activity inflates in this container its views.

7. Helper functions for Network, Preferences, YouTube, Date, etc resides in the 'package' 'utilities'

Task 2: Database setup

1. SQLite is used to store and retrieve the application's data

2. The required classes resides in the 'package' 'data'

3. For the two entities, Vehicle and Expenses create the required Contract classes

4. To access and initialize the database create a class extended from 'SQLiteOpenHelper'

5. To handle the database's CRUD operations create a class extended from 'ContentProvider'

Capstone – Vehicle expenses tracking

6. For other database methods or functions create a class named DBQueries. For simple queries use the 'ContentProvider' created in step 5.
   For more complex queries use the 'SQLiteOpenHelper' created in step 4.

Task 3: Data Models

1. In the 'package' 'model' create required classes with the required fields, getters, setters for the Expense, Vehicle, VehiclesTotalRunningCosts, VideosListItem, YoutubeSearchInfo

Task 4: Initialize Menus, Arrays, Colors, Strings & other resources

1. The application uses the drawer menu for the four main activities. Create an XML menu file, name it 'drawer_main_menu' then add the descriptions for the four main activities, i.e., Dashboard, Vehicles List, Expenses List or YouTube search

2. In the 'package' 'navigation' create a class name it 'DrawerMenu' inside create a method name it 'navigate.' The method 'navigate' is responsible for handling the drawers selections and to start the associated activity.

3. For the two entities, the toolbar options menu is used. Create an XML menu file, name it 'menu_entity' then add:
   a. a menu item with title 'Done' and icon 'ic_done_white'
   b. a menu item with title 'Delete' and icon 'ic_delete_forever_black_24dp'. This menu option will never be showing an icon in the title bar. This menu option is displayed only if the entity is in edit mode.

4. For the YouTube search create an XML menu file, name it 'menu_search' then add:
   a. a menu item with title 'Search' and icon 'ic_menu_search'

5. In the 'res.values' folder create files for the colors, strings, styles, attrs, arrays, dimens.

6. In the arrays.xml create standard application data like:
   a. Arrays for the distance units one with full description and one with a short description, the supported units are Kilometers, Miles, Hours
   b. Arrays for the volume units one with full description and one with a short description, the supported units are Liters, Gallons
   c. Create an array for the expense type with the values: Refuel, Bill, Service
   d. Create an array for the subtype of expense Refuel with the value: Full
   e. Create an array for the subtype of expense Bill with the value: Parking, Toll, Insurance
   f. Create an array for the subtype of expense Service with the value: Basic Service, Damage

Capstone – Vehicle expenses tracking

Task 5: Launcher Activity

This activity is the primary activity that is called when the app is started. Its primary function is to prepare the intent for the next activity that will be called to display the screen, and it finishes immediately after starting the new activity.

With this launcher, we can quickly start the activity in that we are working in the development environment, and when we are done, we switch to the activity that the users see when the app starts in the production environment.

Task 6: Vehicles List Activity

This activity will display a list of all vehicles stored in the database entity ‘Vehicle’. This activity will use the component ‘recyclerview’ to handle and display the list, for the list population the ‘LoaderManager’ with ‘AsyncTaskLoader’ will be used to query the database and fill an array list of type ‘Vehicle’. The ‘recyclerview’ will display the data from this list via an associated adapter.

In the right bottom corner a floating action button will be displayed with an icon of ‘ic_add_white_24dp’, if the user taps on this fab the ‘VehicleEntityActivity’ is called and an empty vehicle entity is displayed.

If the user taps on the list item of a vehicle, then the ‘VehicleEntityActivity’ is called and the screen is populated with the vehicle’s data.

The name of the activity is ‘VehicleListActivity’, in the OnCreate method the content view is set to ‘app_drawer’ then the ‘layout_container’ is inflated with the activities layout that is stored in the file ‘activity_vehicles_list’.

The associated ‘recyclerview’ adapter has the name ‘VehiclesListAdapter’ and is responsible for populating the ‘recyclerview’ with the data. For the Items, it will use the layout file ‘item_list_vehicle’.

Each vehicle item is based on the component ‘CardView’ which holds an ‘ImageView’ that fills the whole ‘CardView’ with a ‘TextView’ for the vehicle’s name. The ‘TextView’ is above the ‘ImageView’ and at the bottom of the ‘CardView’. The ‘CardView’ has a height of ‘200dp’

Capstone – Vehicle expenses tracking

Task 7: Vehicles Entity Activity

This activity will display the Vehicle's Entity fields for new entry, editing or deleting. The user can select an image from the gallery for the vehicle by tapping the fab with the gallery image.
In the toolbar options menu the action 'done' is always visible as an icon. The option 'delete' is available if the entity is in edit mode.

Actions by the user:

- Taps the 'done' menu option, The record will be validated for mistakes, when mistakes are found then the user will be informed to correct those mistakes when validation is passing then the data will be stored in the database via the 'ContentProvider' by using the entity contract 'VehiclesContract'.
  If the data are for the new record then a new record is inserted into the database, when the data are for an old record then the data are replacing the old data in the database.
  The activity will finish and the screen returns to the vehicle list activity.

- Taps the options menu and the taps on the 'delete' option, Via a popup or dialog the user is asked to confirm the deletion if the user answer with 'yes' then the record will be deleted plus all the assosiacted data from expenses for this vehicle if exists.
  If the deletion occurs then the activity will finish and the screen returns to the vehicle list activity, otherwise, the activity will stay in the screen.

- Taps the back button on the toolbar, the activity will finish without saving the data and the screen returns to the vehicle list activity.

The name of the activity is 'VehicleEntityActivity', in the OnCreate method the content view is set to 'app_coordinator_container' then the 'layout_container' is inflated with activities layout that is stored in the file 'activity_vehicles_entity'.

Task 8: Expenses List Activity

This activity will display a list of all expenses stored in the database entity 'Expense'. This activity will use the component 'recyclerview' to handle and display the list, for the list population the 'LoaderManager' with 'AsyncTaskLoader' will be used to query the database and fill an array list of type 'Expense'. The 'recyclerview' will display the data from this list via an associated adapter.

In the right bottom corner a floating action button will be displayed with an icon of 'ic_add_white_24dp', if the user taps on this fab the 'ExpenseEntityActivity' is called and an empty expense entity is displayed.

If the user taps on the list item of an expense, then the 'ExpenseEntityActivity' is called and the screen is populated with the expense data.

Capstone – Vehicle expenses tracking

The name of the activity is ‘ExpenseListActivity’, in the OnCreate method the content view is set to ‘app_drawer’ then the ‘layout_container’ is inflated with the activities layout that is stored in the file ‘activity_expenses_list’.

The associated ‘recyclerview’ adapter has the name ‘ExpensesListAdapter’ and is responsible for populating the ‘recyclerview’ with the data. For the Items, it will use the layout file ‘item_list_expense’.

Each expense item is based on the component ‘CardView’ which holds and ‘ConstraintLayout’ that with multiple ‘TextView’ for the expense’s fields. The ‘CardView’ and the ‘ConstraintLayout’ have a height of ‘wrap_content’

Task 9: Expenses Entity Activity

This activity will display the Expense’s Entity fields for new entry, editing or deleting. The Vehicle’s image is displayed in the ‘ImageView’ of the ‘CollapsingToolbarLayout’.
In the toolbar options menu the action ‘done’ is always visible as an icon. The option ‘delete’ is available if the entity is in edit mode.

Actions by the user:

- Taps the ‘done’ menu option, The record will be validated for mistakes, when mistakes are found then the user will be informed to correct those mistakes, when validation is passing then the data will be stored in the database via the ‘ContentProvider’ by using the entity contract ‘ExpensesContract’.
  If the data are for the new record then a new record is inserted into the database, when the data are for an old record then the data are replacing the old data in the database.
  The activity will finish and the screen returns to the expense list activity.

- Taps the options menu and the taps on the ‘delete’ option, Via a popup or dialog the user is asked to confirm the deletion if the user answer with ‘yes’ then the record will be deleted.
  If the deletion occurs then the activity will finish and the screen returns to the expense list activity. Otherwise, the activity will stay in the screen.

- Taps the back button on the toolbar, the activity will finish without saving the data and the screen returns to the expense list activity.

- The title of the expense type is displayed in the Toolbars title of the ‘CollapsingToolbarLayout’.

The name of the activity is ‘ExpenseEntityActivity’, in the OnCreate method the content view is set to ‘app_coordinator_container’ then the ‘layout_container’ is inflated with activities layout that is stored in the file ‘activity_expenses_entity’.

Capstone – Vehicle expenses tracking

Task 10: Search YouTube Activity

This activity is based on the ‘app_drawer’ layout, then the ‘layout_container’ is inflated with the activities layout that is stored in the file ‘activity_search_youtube’.

The toolbar options menu is populated with the menu ‘menu_search’ with the option ‘Search’. When the user taps on the option ‘Search’ icon a SearchView is displayed in the toolbar.

After tapping the search button on the keyboard the search begins by retrieving only the first 10 videos from the YouTube search service.

The results are loaded via the ‘LoaderManager’ and ‘AsyncTaskLoader’. In the YoutubeUtils are all the necessary implementations to query and parse YouTube results in an Array List of type ‘VideosListItem’. Then the results are displayed via a ‘recyclerview’ it will display the data from this list via an associated adapter.

The associated ‘recyclerview’ adapter has the name ‘VideosListAdapter’ and is responsible for populating the ‘recyclerview’ with the data. For the Items, it will use the layout file ‘item_list_video’.

When the user taps on a video, the video is played via tha associated app or the web url for watching YouTube videos.

Task 11: Dashboard

This activity is based on the ‘app_drawer’ layout, then the ‘layout_container’ is inflated with the activities layout that is stored in the file ‘activity_dashboard’.
In this activity all entered expenses are calculated per vehicle to display the total running costs for each vehicle.

Again the data are read and calculated from the database via the ‘LoaderManager’ and ‘AsyncTaskLoader’. The results are stored in an array list of type ‘VehiclesTotalRunningCosts’. Then the results are displayed via a ‘recyclerview’ it will display the data from this list via an associated adapter.

The associated ‘recyclerview’ adapter has the name ‘DashboardListAdapter’ and is responsible for populating the ‘recyclerview’ with the data. For the Items, it will use the layout file ‘item_list_dashboard’.

Capstone – Vehicle expenses tracking

Task 12: Home Screen Widget

When the user selects to add the capstone widget on the home screen, then a list of all available vehicles is displayed on the screen. The population and the functionality are the same as the VehiclesListActivity except for what happens when the user taps on a vehicle item.

When the user taps on a vehicle item then the vehicleId is stored in the shared preferences with ‘widgetid’ as the key. Then a ‘sendRefreshBroadcast’ is sent to the widget provider, and returns with RESULT_OK and finish its cycle, the widget will not be displayed on the home screen.

If the user presses the back key then the configure activity returns RESULT_CANCELED and finish its cycle, the widget will not be displayed on the home screen.

The widget provider will send a broadcast to the AppWidgetManager to update the widget content. The AppWidgetManager will call the method onUpdate from widget provider and in this method a service will be started to read the data from the database and to populate the remote views with the data. The displayed data are only for one vehicle and are the same data that are displayed in the dashboard for this particular vehicle.

In the home screen the user can add as many widgets as he likes, for each widget he can select a particular vehicle.

Credits

- Udacity and Google for the lectures and the advantageous examples
- many many thanks to the reviewers and the mentors for their excellent work and suggestions
- George Vrynios for the implementing of the app, starting at 23-Jun-2018
- zularizal for the app Icons (https://github.com/zularizal)
- Unsplash, Jonathan Daniels, for the included images