
GitHub Username: parcenta

CustomerConnect

Description

This tool allows to register customers and/or prospects along with their contact information, map location, photo, etc. The tool also allows to register visits for their registered customers in order to keep track of their work with them.

Intended User

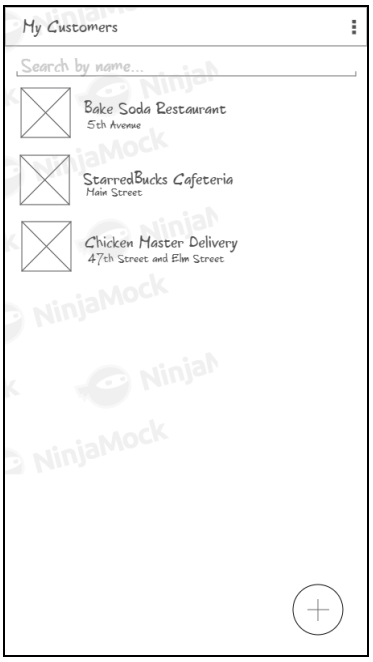
This app is targeted for those house-to-house sellers or for any user that needs to keep track of their daily work for old and new customers.

Features

- Keep saved the information about your customers and/or prospects. This tools allow to save:
    - **Contact Information:** Name, Phone Number, Email and Phone number.
    - **Address Information:** Street and Location in map (GPS).
    - **Photo:** Can attach or take one picture.
- Register visits for your Customers and keep track of your daily work.

User Interface Mocks

Customer List Panel (Non-Tablet)



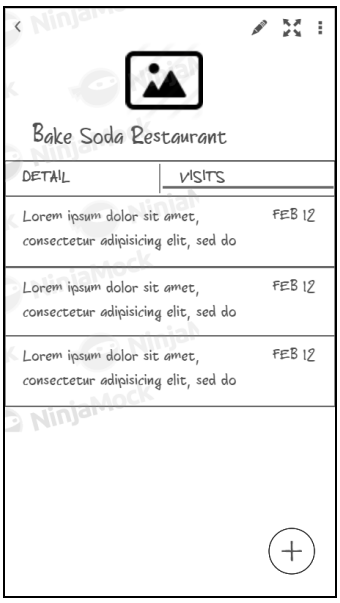
The main screen displays all the customers created by the user. The list shows the Name, image and Street Address of each customer.
The user can search by customer´s name and create new customers by pressing the “+” button.

Customer´s Detail panel: Detail’s Tab (Non-Tablet)

This panel will have 2 tabs: Detail and Visits.

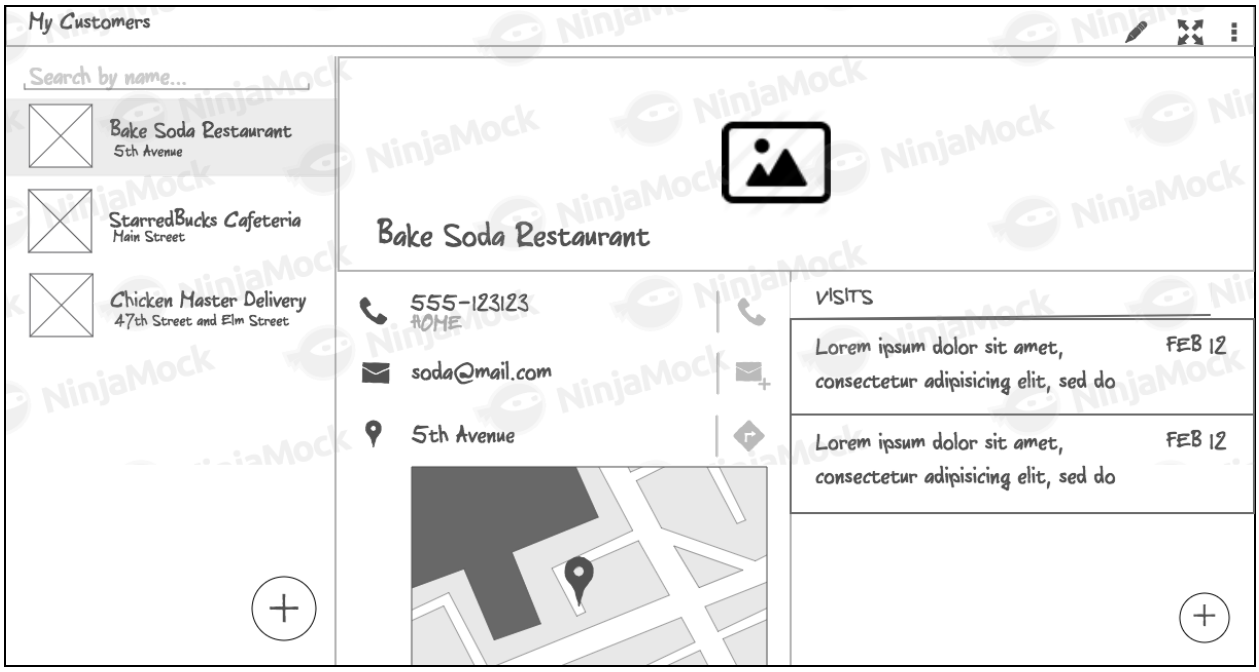
The Detail’s Tab will displays the saved information about the customer. Information like phone number, email and location will be displayed in this panel.
Also the user can make a call, send an email and navigate to the customer’s location. This can be done by pressing the respective action buttons in the right side of the each value.

Customer´s Detail panel: Visit's Tab (Non-Tablet)

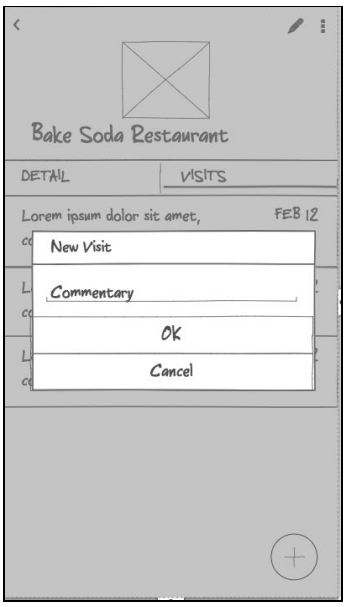


The Visit's Tab will displays all the registered visits of the detemined customer.
The user can register new visits by pressing the floating button in the bottom-right corner.

Tablet Layout



New Visit Dialog



In order to register a new visit for the customer, a user can input a commentary and then save it.

Create/Edit Customers Panel

This panel will be used for creating and updating the customers. The user can input the phone number, email, street address, map location and take/attach one photo. The user also can get the current location.
To save the changes, the user must press the floating button in the right-bottom corner.

Widget

A widget will be available to display the visits registered in the current day.

Key Considerations

How will your app handle data persistence?

All the data will be saved in a local SQLLite Database. There will be 2 tables: Customer and Visits. A single content provider will provide the communication to both tables.
The customer's images will be saved in the local application's directory.

Describe any edge or corner cases in the UX.

- If the user needs to edit information of a registered user, he needs to go to the customer´s detail panel and pressed the Edit button in the toolbar.
- If the user needs to delete a registered customer (along with its registered visits), he needs to go the customer´s detail panel and press the Delete Button (in the collapsed menu). A confirmation message will be displayed to ask confirmation about the delete action.

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

- Picasso library will be used to display the customer's photos.
- Timber library will be used for Logging.
- RxJava and RxBinding libraries to manage the "textChanged" behaviour in the Search input in the customer´s list. The "RxTextView.textChanges(view)...debounce(milliseconds)" is perfect to handle the text changing behaviour.
  (*https://github.com/codepath/android_guides/wiki/RxJava-and-RxBinding*)

Describe how you will implement Google Play Services.

- Google Play's Location services will be used to get current location. (Fused location)
- Google Play´s Map services will be used to display map location

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

Target and Min SDK Setup

- Specify the Target SDK to API 26 and Min SDK to 21.

Configuration of Libraries

- Must setup all the libraries (specified before). Must corroborate that they work correctly with the buildTools, target and min SDK versions specified.

Setting up needed permissions in Manifest

- Must specify three permissions that must be granted by the user:
    - ACCESS_FINE_LOCATION: to access current location with a high accuracy. Must be asked when user tries to get current location.
    - CAMERA: to access the device´s camera in the case the user wants to take a photo for the customer. Must be asked when user tries to take a photo.
    - WRITE_EXTERNAL_STORAGE: to save/read customer´s photos. Must be asked when user take/attach an image.

Authorize the use of Google Maps in the app.

- Must generate Google API_KEY from the Google Developer site in order to visualize the map in the application.

Task 2: Setup of Database and Providers.

Building CustomerConnectDatabaseHelper

- Must create a database helper that will handle the creation of the database.
- The database need to contain 2 tables that extends from BaseColumns:

Customer	**Visit**
+ ID (PK - Integer) (Autoincrement) + CustomerName (Text) + CustomerPhoneNumber (Text) + CustomerPhoneType (Text) + CustomerEmail (Text) + CustomerAddressStreet (Text) + CustomerAddressLatitude (Real) + CustomerAddressLongitude (Real) + CustomerCity (Text) + CustomerCountry (Text) + CustomerPhotoPath (Text)	+ ID (PK) (Autoincrement) + CustomerID (Integer) + VisitCommentary (Text) + VisitDatetime (Datetime)

Building the CustomerConnectProvider

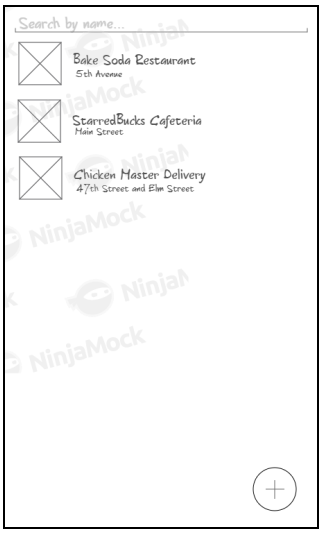
- Must build a single content provider that will handle the query, create, update and delete (CRUD actions) of the 2 database tables.
- Must declare it in the AndroidManifest.xml.

Task 3: Implement UI + Functionality for each Fragment

Creating the CustomerConnectMainActivity (Launcher Activity)

- Must set this activity as the Launcher activity in the manifest. **NOTE:** For now, just create the activity. Like we are going to apply the “MasterListFragment” style in our app, we need to build the fragments first and then implement those fragments in this activity accordingly.

Building CustomerListFragment (Fragment): UI + Functioning



- Must contain a recycler view that displays all the registered customers.
- Must contain a text input to specify the search criteria.
- Must contain a Floating Action Button (FAB) that will allow the creation of new customers.
- The Search input must make a search everytime the user change the text. For this, we can implement a “TextChanged” listener like the one implemented in the “RxJava + RxBinding”.
- If user presses on an customer (item in the recycler view), the details of that customer must be display.

Building CustomerDetailHeaderFragment (Fragment) UI + Functioning

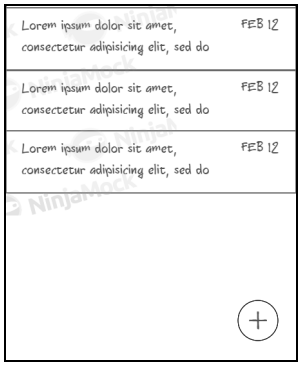
- Must contain an imageview to show the customer´s image.
- Must contain a Textview to display the customer´s name.
- Must setup a toolbar with actions of “Update”, “Zoom picture” and “Delete” (collapsed option).
- This fragment must received the “CustomerId” in order to load (via Loader) the respective information about the selected customer.

Building CustomerDetailInfoFragment (Fragment)

- Must display phone number, email and location.
- The user has the option to make a call, send an email and navigate to the customer's location. This actions will be handled with implicit intents.
- This fragment must received the "CustomerId" in order to load (via Loader) the respective information about the selected customer.

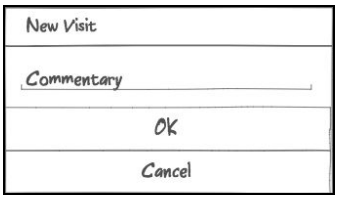
Building CustomerVisitsFragment (Fragment)

- Must contain a recycler view with the registered visits for the selected customer.
- Must contains a FAB button that will allow the user to register new visits.
- This fragment must received the "CustomerId" in order to load (via Loader) the respective information about the selected customer.



Building CustomerNewVisitDialog Fragment (DialogFragment)

- Must contain a Text Input for the commentary.
- Must contain a button for saving the visit (with its commentary).
- Must contain a button to return to the previous screen.
- This fragment must received the "CustomerId" in order to save correctly the respective visit to its customer.



Building CustomerEditActivity (Activity)

- Must contain a Text Input for Phone Number, Email and Street address.
- Must contain a Spinner for the Address Type. With 3 possible values: Home, Work, Other.
- Must contain a MapFragment which allows to get current location.
- Must contain an ImageView to display to customer´s image. If no image is selected, then show buttons that will allow to attach or take the photo. If an image is already selected, then an erase button must be display over the image.

- Must contain a FAB for creating the customer or saving changes.
- Location must be search with FusedLocationProviderClient. If a valid location is returned, then search the city and country using the Geocode Google API webservice https://developers.google.com/maps/documentation/geocoding/. Must use an **AsyncTask** to consume this webservice and get the city/country of the current location. NOTE: We must specify a Google API_KEY to use this webservice.

Building CustomerDetailActivity UI (Activity)

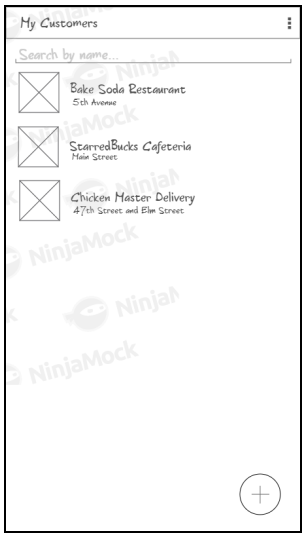
- Available for devices that are **NOT tablets**.
- Must contain the "CustomerDetailHeaderFragment" in the top.
- Must contain a viewpager with 2 tabs:
  Tab "Detail": Display the "CustomerDetailInfoFragment".
  Tab "Visits": Display the "CustomerVisitsFragment".

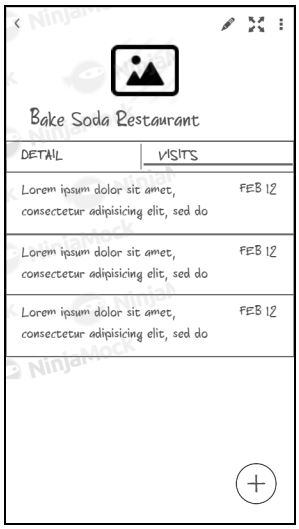
Task 4: Bring all Activities and Fragments together

Now we can proceed to join the fragments to their respective activities considering on a tablet and not-tablet layouts.

For NON-TABLETS:

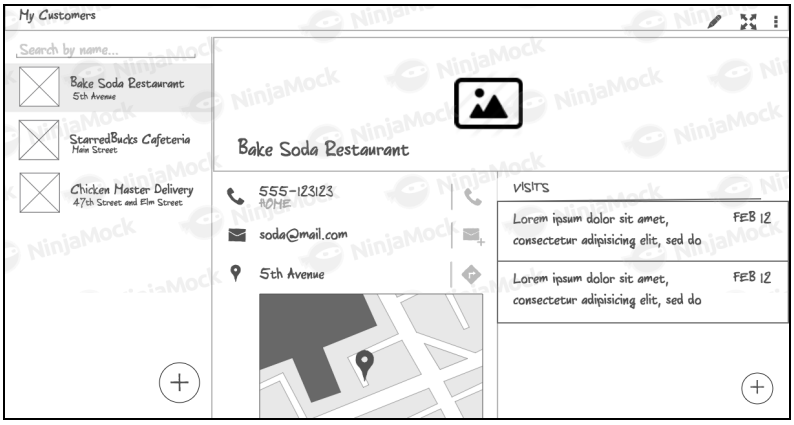
- **CustomerConnectMainActivity:** Must display the CustomerListFragment only.
- **CustomerDetailActivity:** must display CustomerDetailHeaderFragment + CustomerDetailInfoFragment + CustomerVisitsFragment fragments.





For TABLETS:

- **CustomerConnectMainActivity:** Must display CustomerListFragment, CustomerDetailHeaderFragment, CustomerDetailInfoFragment and CustomerVisitsFragment fragments.



- The first customer must be displayed as default. If a user makes a search, the previous customer will still be displayed (until a new customer is selected).
- The toolbar (of this activity) must be set by the “CustomerDetailHeaderFragment” to include the edit, zoom and delete (collapsed) actions.

Widget:

- Must display the last 10 visits registered in the current day.

Task 5: Unit Testing

Unit testing for the next methods:

- Email validation
- Phone validation (Can´t be alfanumeric value)

Task 5: Setup of Release version

Create release Key Store

- Must create the Key Store for the release version and save it in the root of the project´s directory.
- Execute assembleRelease and check that release version

Authorize release version for the use of Google Maps API.

- Must authorize the use of maps api for the release version. We must specify the release´s SHA1 fingerprint along with the package name in the Google Developer console.