GitHub Username: ahs.udacity@gmail.com

OffersGalore

Description

This App is for the owners of small stores, which will allow them to save their customer details and send offers to the added customers through SMS.

Intended User

This App is mainly for small shop or small grocery store owners, allowing them to create promotional offers through SMS.

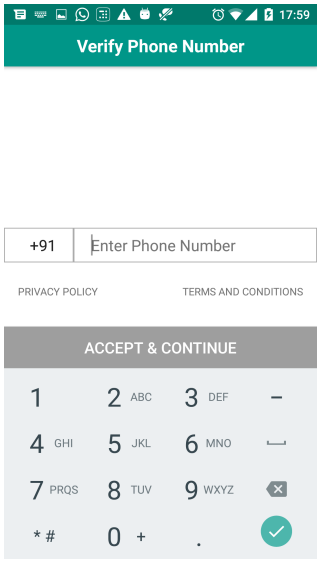
Features

List the main features of your app. For example:

- Login with mobile number, currently only for Indian numbers.
- OTP verification (sms is sent from merchants SIM who will be charged by the network provider)
- Allows Sellers to create or add customers details with mobile number(mandatory), interests ,name and email.
- Allow Sellers to edit created customers.
- Allows Sellers to create offers with the saved list of users or from the phonebook, with a scheduled date and offer text.
- Allow sellers to edit scheduled offers that has not been sent already
- Filter offers based on scheduled and sent
- Sends SMS to all the selected customers in the offer when the scheduled date is reached(sms is sent from merchants SIM who will be charged by the network provider).
- Incase of returning user, fetch customer and offer data from firebase on launch of landing page.

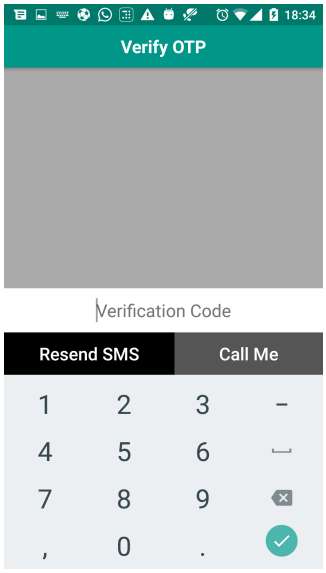
User Interface Mocks

Screen 1



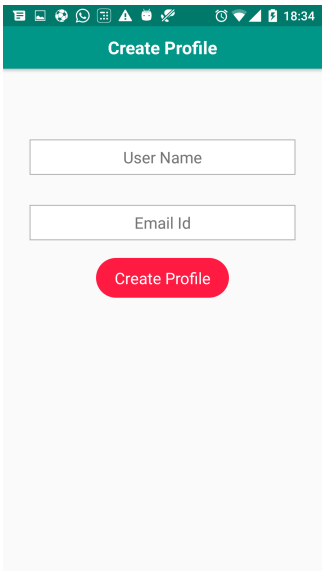
Login with Phone Number verification, 10 digit phone mobile number .

Screen 2



User will receive otp through sms and will have to get it verified here.

Screen 3

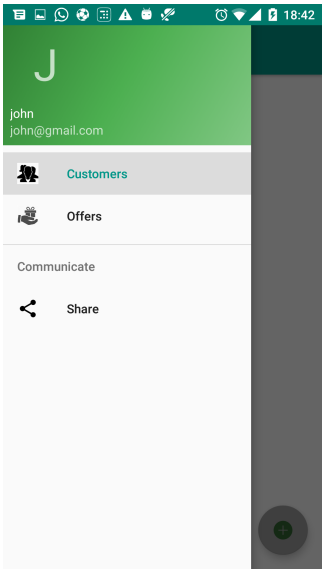


Create Profile with username and email id

Screen 4

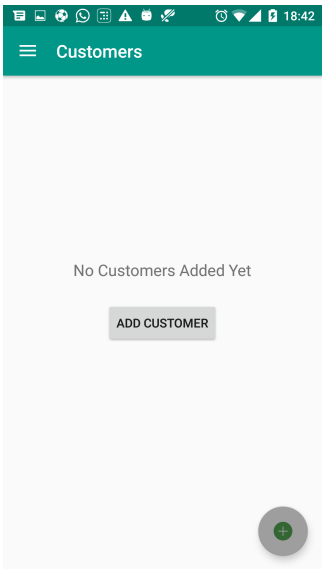
Select interests of merchant just for records.

Screen 5



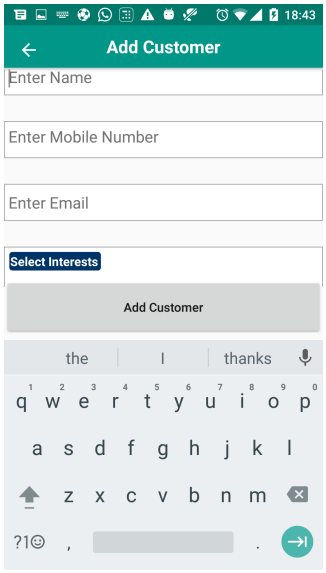
Landing page with navigation drawer displaying 2 main options Customers and Offers.

Screen 6



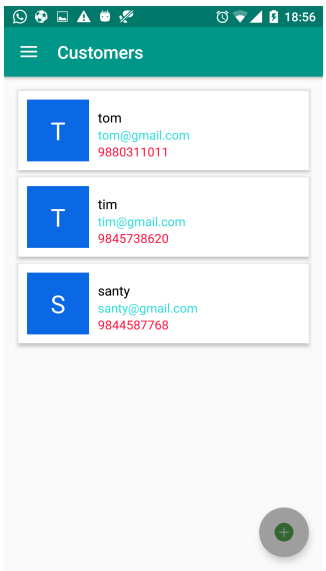
Page that displays created customers in case not created displays a button with option to add customer

Screen 7



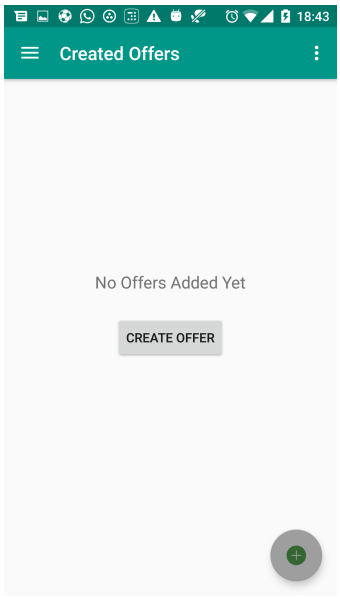
On click of the + button or the “Add Customer” button in the previous screen, it launches this page to add customer.
On Click of “Select Interest” button launches Screen 4 to register customer interest.

Screen 8



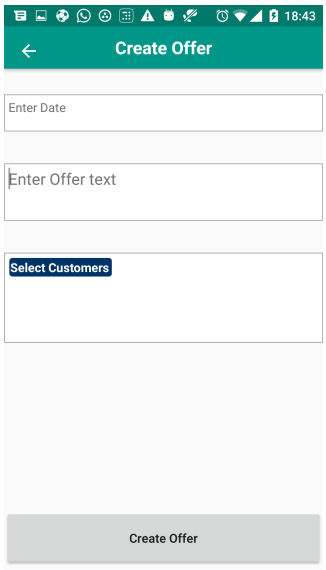
On creation of customer comes back to screen 6 where it displays the list of Customers created.

Screen 9



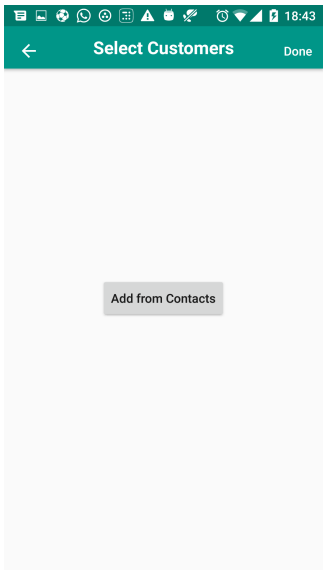
On click of the offers tab in navigation view on screen 5 it displays this screen, which lists the Created offers and incase no offers created displays a “create offer” button to create offer.

Screen 10



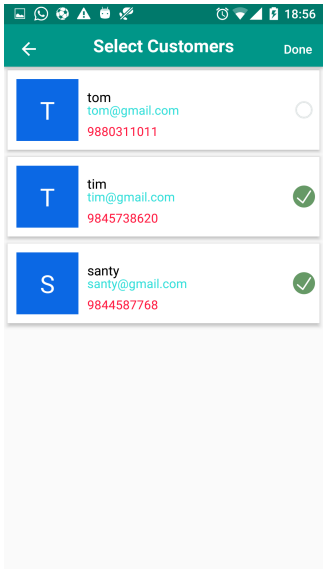
On click of create offer from previous screen it displays this screen where you can enter the date,
as to when the offer needs to be sent , along with the offer text and the customers.

Screen 11



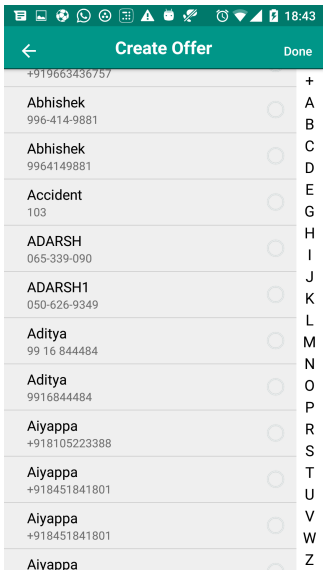
On click of “Select Customer” button from previous screen it launches the list of customers added if not added then it displays an add from Contacts button to add from phonebook as shown in this screen.

Screen 12



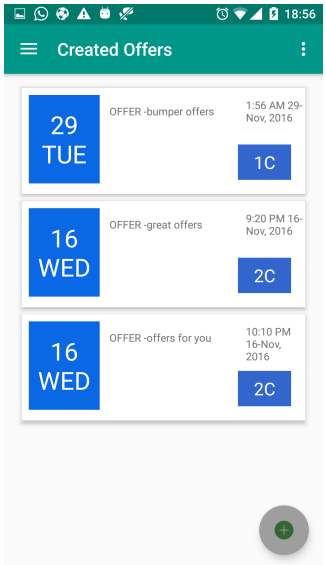
On click of “Select Customer” on screen 10, it navigates to this screen where it displays the list of customers to choose from for the offer.

Sreen 13



On click of “Add from Contacts” on screen 11 it launches this screen with an option to select customers from phonebook.

Screen 14



On click of create offer button in screen 10, it creates offer and navigates to this screen where list of offers created are displayed.
A pending intent is created to send the SMS at the scheduled date to the numbers of all the selected customers for the created offer.

Key Considerations

How will your app handle data persistence?

The app uses firebase for remote persistence and realm database for local persistence.

Describe any corner cases in the UX.

The app has a navigation drawer to navigate to 2 of the major features provided like offer and customer.
It also has detailed view for tablets required for editing and saving created customer details and created offer (scheduled offers only) details.

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

I Will be using realm library to persist data as it makes implementation easier while using firebase since both are nosql databases, also integration is faster and conversion of json to object is already available as firebase libs.

Also Picasso for images and Butterknife for databinding.

Describe how you will implement Google Play Services.

I'll be using firebase which is a part of google play services to store data remotely, which will be useful if the user uninstalls the app and reinstalls it with the same number, in which case it shall fetch the data(customer and offer related) already created remotely from firebase.

Next Steps: Required Tasks

Task 1: Project Setup

- Configure firebase with android project following instructions on the link - https://firebase.google.com/docs/android/setup
- Add libs Butter Knife, realm and Picasso.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for splash screen ShopOnSplashActivity
- Build UI for login screens ShopOnMsisdnActivity, SmsOtpVerify, ShopOnProfileCreation,
- Build UI for interests screen ShopOnCategoryActivity.
- Build UI for landing page ShopOnActivity with NavigationView(displaying two tabs offers and customers).
- Build UI for customer screens CustomerFragment, CustomerActivity, CustomerDetailFragment, CustomerDetailActivity.
- Build UI for offer screens OfferFragment, OfferActivity, OfferDetailFragment, OfferDetailActivity.
- Handle UI for tablets by adding detail screen(CustomerDetailFragment, OfferDetailFragment) and created list(CustomerFragment,OfferFragment) in one screen.
  .

Task 3: Design Remote and Local Database

- On firebase, design 3 different dictionaries in the database, one for merchant, 2nd for merchants customers and 3rd for merchants offers.
- Map each of the dictionaries in the firebase to objects. Have to create 2 pojos for all 3 dictionaries because firebase cannot use pojo that extends RealmObject.
- Create utils functions for creating retrieving and updating data in firebase.
- Create util functions for converting firebase pojo objects to realm pojo objects and vice versa.

Task 4: Implement Login Flow

- On Number verification screen generate and send sms through subscribed SIM
- Store number and OTP in shared pref.
- Store login state(msisdn_state,otp_state,profile_state,category_state,login_complete) in pref on each of the login screen so that incase user closes app before completion he can revisit the state where he had left off.
- On otp verification screen check if number already created in firebase, if created then proceed to landing page(ShopOnActivity) else proceed to create profile.
- On create profile screen process profile name and email id and proceed to interest screen,
- Select interest and save all the details to firebase.

Task 5: Adding Customers and Offers

- On Created Customers screen list all the created customers.

- FAB button to launch Add Customer, in Add Customer screen create customer with name, email, interests and number.
- On Created Offer screen list all the created offers.
- FAB button to launch Create Offer, in Create Offer Screen create offers with selected numbers of created customers, offer text and schedule date.
- Edit options for created offers that has not been sent yet.
- Edit option for created customers
- Filter option based on scheduled and sent Offers in Created Offer Screen.

Task 6: Sync LocalDB with RemoteDB

- Sync Local Offer table with remote offer dictionary
- Sync Local Customer table with remote customer dictionary
- Initiate sync on launch of landing page.

Task 7: Handle Error Cases

- Check Internet connectivity before proceeding with OTP verification.
- Field validations for creating profile and in edit offer for and edit customer, create offer and create customer page.
- Add permissions to access features programmatically for marsh mellow onwards to selectively grant permission through user input to use phone features like sms, phone state and to access contacts.

Task 8: Send SMS for created Offers

- On creation of offers create a pending intent to send sms and update the status locally and remotely.
- Send SMS through merchants registered SIM(wil be charged by network provider , atleast for this version).

Task 9: App Widget to display Scheduled or sent offers for the current day

- Create App widget.
- Display the list of scheduled or sent offers for the current day.

Task 10: Setup Admob banner in create offer screen.

- Follow instructions in firebase setup
- Generate app id and ad unit id.
- Add code in Create Offer Screen to display Ad Banner.

Task 11: Configure InstallRelease command in build.gradle

- Generate .jks file and store credentials and file in app
- Confiure path to jks file and key value in gradle.properties file.
- Configure gradle installRelease build

.

Task 12: Create wrapper for realm using content provider

- Create content provider.
- Override content provider methods to perform realm specific operations.
- Use content provider to query, update, insert, data and use loader in list views

.

Task 13: Replace realm with mysqlite and task12 with content provider for mysqlite

- Reason for replacing: realm does not support auto refresh for non looper threads, content provider with loader does not run on a looper thread hence resulting in inconsistent read operations, that follow subsequent writes.
- Implement content provider for mysqlite table.
- Replace all refrerences of realm invocation with mysqlite operations through content provider.