GitHub Username: bhagat19

Kaastkaar

Description

Kaastkaar aims to bridge the gap between the farmer and market(retailer) by bringing the market at the tap of a farmer's finger. A farmer can see retailer's posts based on his location while retailers can post their requirements to procure farm goods locally.

Kaastkaar's mission is to help small farmers find their profitable other half because kaastkaar at its core believes that a sower should reap maximum benefits of its harvest.

Intended User

Kaastkaar is intended to be used by local farmers and retailers who trade in perishable goods i.e. fruits and vegetables.

Features

- Connects local farmers with retailers
- Lets a retailer post his requirement
- Lets a farmer view posts based on his location

User Interface Mocks

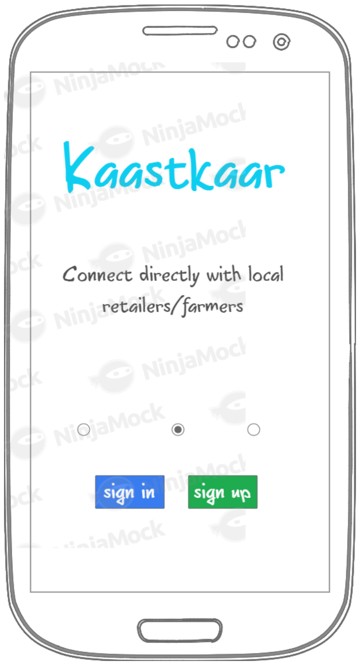
I have used ninja mock to create mock designs for Kasstkaar. link

Tablet Layout

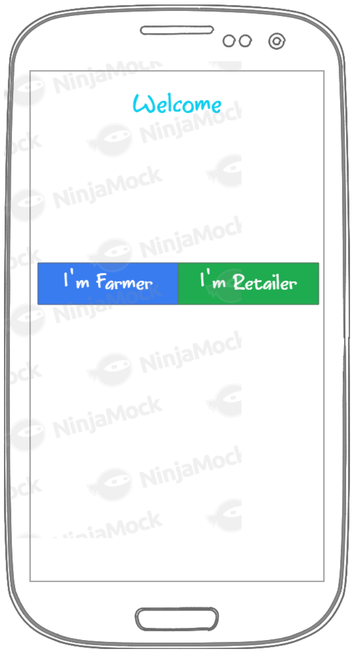
Tablet layout would be same as of mobile except screen 5 and 8.
A two pane layout would be built to show list of retailers in one pane and their details in other.

Mobile Layout

Please note that 12 screens have been added to showcase flow of mobile app, some screens may be added/modified/deleted in final version of app, though the basic functionalities would remain same.

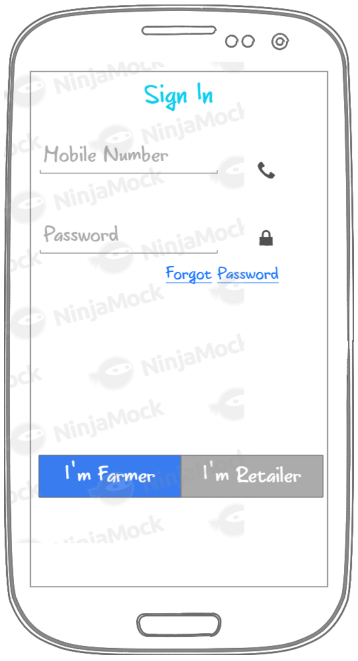


Screen 1
3 Sliding screens giving info about app



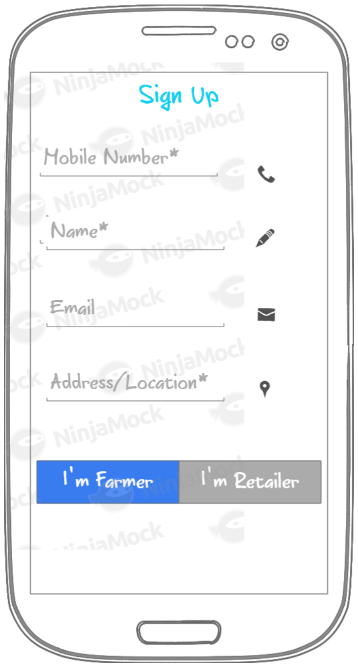
Screen 2
Welcome screen to user

Screen 3

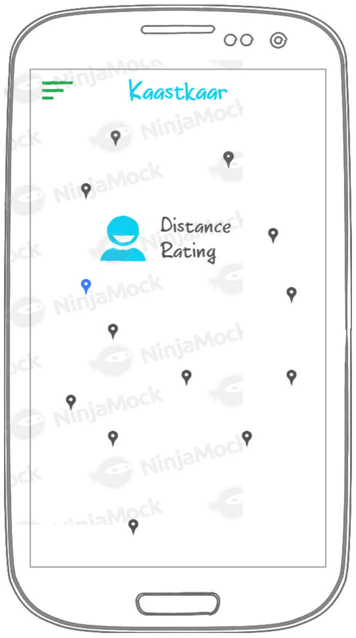


Sign in page of Farmer

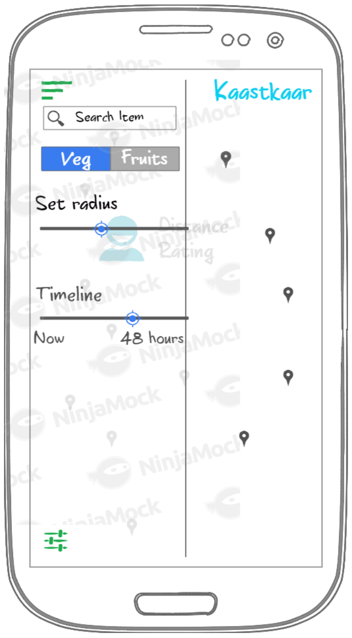
Screen 4



Sign up Page of Farmer

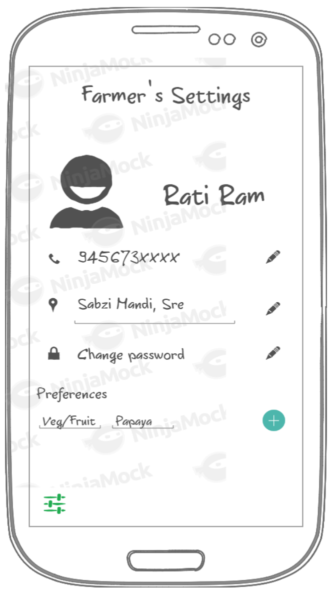


Farmer's view based on his location
Screen 5



A dialog box to filter results
Screen 6

Screen 7



Rati Ram

945673xxxx

Sabzi Mandi, Sre

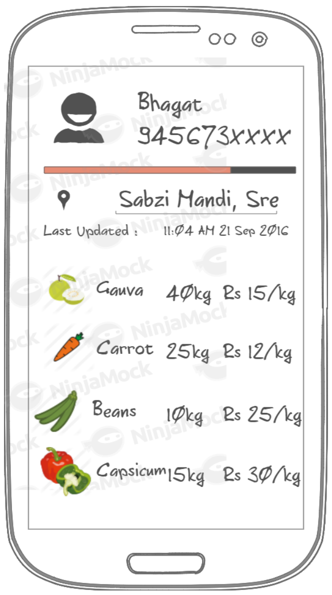
Change password

Preferences

Veg/Fruit Papaya

Farmer's setting Page

Screen 8



Bhagat
945673xxxx

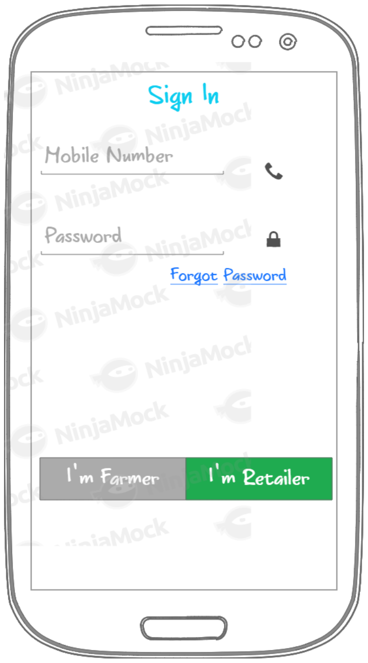
Sabzi Mandi, Sre

Last Updated : 11:04 AM 21 Sep 2016

Gauva	40kg	Rs 15/kg
Carrot	25kg	Rs 12/kg
Beans	10kg	Rs 25/kg
Capsicum	15kg	Rs 30/kg

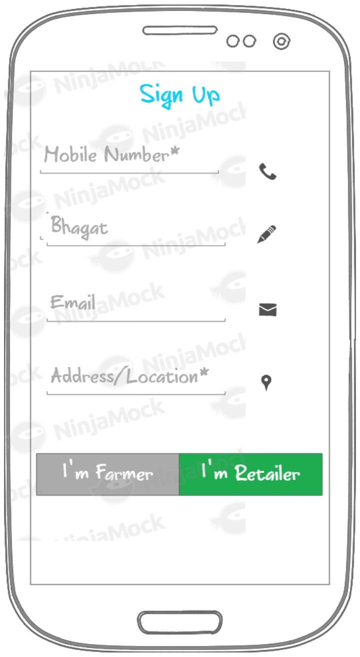
Retailer's detail view to farmer from Screen 5

Screen 9



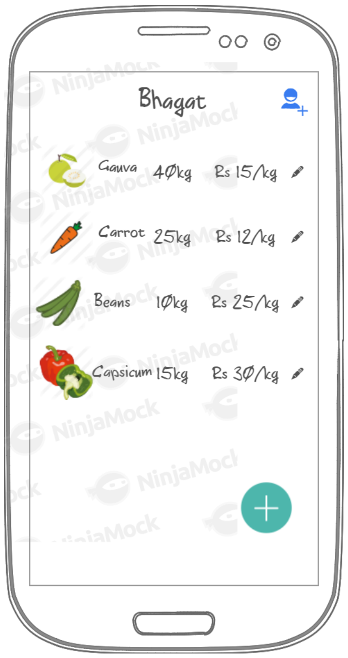
Retailer's sign in page

Screen 10



Retailer's sign up page

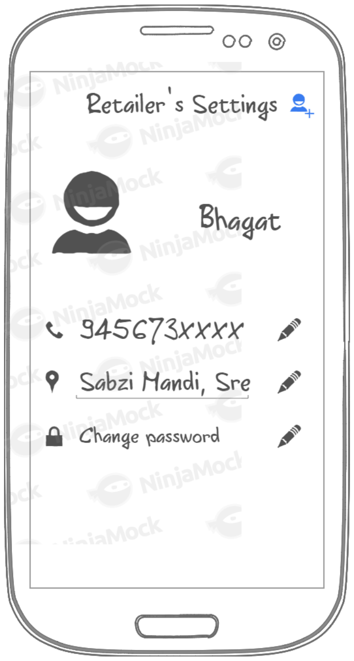
Screen 11



Gauva	40kg	Rs 15/kg
Carrot	25kg	Rs 12/kg
Beans	10kg	Rs 25/kg
Capsicum	15kg	Rs 30/kg

Retailers main page

Screen 12



Retailer's Settings page

Key Considerations

How will your app handle data persistence?

Users data including login credentials, their preferences would be handled by Firebase auth API and Firebase Relatime Database API respectively.

While a Content provider would be built to cache in and locate nearest mandi(a place to trade agricultural goods) to farmer and retailer.
Data for mandis would be taken from data.gov.in, which is an open data platform under digital India initiative.

Describe any corner cases in the UX.

The app implementation will try to consider any possible corner case, in order to avoid strange behaviours from the user point of view. For example, when using the app without any network available the local data will still be accessible.
And when no data could be retrieved for any reason, placeholder images will be used to point out this situation. In addition, the possibility of not having any available image for a certain vegetable or fruit will be taken in consideration, in order to avoid strange or broken layouts.

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

Picasso/UIL - Image loading and caching
Butterknife - Views Injection
OkHttp - Handling Networking calls
Schematic - Generating Content Providers

Describe how you will implement Google Play Services.

Places API - for showing retailer's position on Google maps.
Firebase Realtime Database - to manage and update realtime data across devices.
Firebase Crash Reports - to observe app behaviour and collect data to prevent crashes.
Android Design Support Libraries - Material design concepts would be leveraged to provide delightful user experience.

Next Steps: Required Tasks

Task 1: Project Setup

● Create a new empty Android Studio project for Kaastkaar app.
● Create a new GitHub repository for the project. Remember including .gitignore and README.md files.
● Configure used libraries (following in each case the pertinent instructions), dependencies and other aspects in the Gradle files.
● Configure the use of Open Government Data API: get an api key from their portal.
● Analyze thoroughly Open Data Government API documentation, in order to determine the calls that will be used to cover all the app features.
● Configure the use of the API key in the project in a way that allows not to include it in the repository.

Task 2: Implement UI for Each Activity and Fragment

● Build UI for Farmer's view screen (MainActivity).
● Build UI for Farmer's preferences screen.
● Build UI for Farmer's Settings Page.
● Build UI for Retailers detail page(Contact and requirement info detail page).
● Build UI for Retailer's main page(add items).
● Build UI for Retailer's settings page..
● Build UI for tablet layout of Farmers's main screen(MainActivity).

All the UI implementation will be made taking in consideration the use of Material Design guidelines, features and components (such as Floating Action Button, Floating Action Menu, Coordinator Layout, Collapsing Toolbar Layout…).

Task 3: Build data persistence support

Implement all the classes needed to handle data persistence: Content Provider, Cursor Adapters, Database classes, etc

Task 4: Implement Open Government Data API integration

● Implement classes needed for retrieving mandi data.
● An AsyncTask in order to handle the on demand requests for searching series by keywords from the app main screen or the top search box.

Task 5: Implement Google Play Services/ Firebase integration

● Build Google Places Api Integration - app will show markers on map to locate retailers.
● Build Firebase Realtime Database Integration
● Build Firebase Crash Reports Integration

Task 6: Accessibility and localization

● Ensure that the app offers a good enough experience to users with disabilities, through the correct use of content descriptions, consistent and coherent focus navigation, etc.
● Keep all the app strings in XML files, including localized versions for, at least, hindi and english (which will be the default language).
● Ensure that the app supports RTL layout usage.

Task 7: Implement Kaastkaar app widget

The widget must provide local mandi rates to farmers and retailers based upon their set of preferences.

Task 8: Configure app building

● Configure app signing, including the keystore and passwords in the repository.
● Ensure that app builds and deploys using the ‘installRelease’ Gradle task.