

[Description](#)

[Intended User](#)

[Features](#)

[User Interface Mocks](#)

[App Intro](#)

[App Normal Start](#)

[App Settings](#)

[App Settings Base Prices](#)

[App Resume](#)

[App Choose a Restaurant](#)

[App Build a Shopping List](#)

[App Add New Item](#)

[App Click Image](#)

[App Total](#)

[App Widget](#)

[Key Considerations](#)

[How will your app handle data persistence?](#)

[Describe any edge or corner cases in the UX.](#)

[Describe any libraries you'll be using and share your reasoning for including them.](#)

[Describe how you will implement Google Play Services or other external services.](#)

[Other Code Requirements](#)

[Next Steps: Required Tasks](#)

[Task 1: Project Setup](#)

[Task 2: Implement UI for Each Activity and Fragment](#)

[Task 3: Store Regions Online](#)

[Task 4: Make it look better](#)

[Task 4: Beta Testing and Release](#)

GitHub Username: <https://github.com/toast777/>

Meetup Foodie

Description

Do you have trouble planning your next meetup? Do you have a budget for food for snacks or dinner? If you are an organizer of meetup meet your next app. This app will allow you plan your food for next meetup and get an estimated cost. Once you get your plan it even outputs the shopping list to a homescreen widget for easy reference when you go to order the food.

Intended User

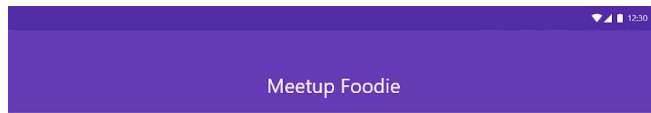
Meetup Organizers that have a food budget for catering/delivery/pickup.

Features

- Calculates food costs
- Creates Shopping Lists
- Customization of food prices for your specific region.

User Interface Mocks

App Intro



Welcome to Meetup Foodie

Please select a region to get started

Drop-Down

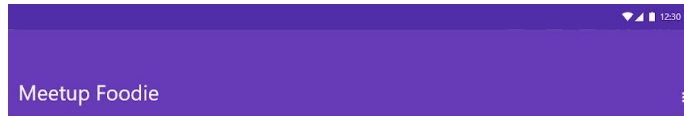
Submit

It's ok to select a location other than your own.
The app will let you customize your food items
and prices.



When App is first installed This is first screen presented to user, does not run every time

App Normal Start



Start Shopping

Would you like to?

Resume

(Bring up last saved list) -
in tooltip

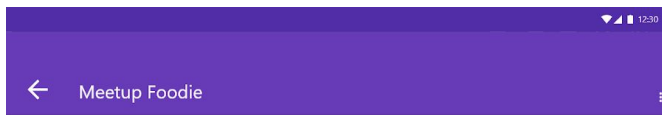
New List

Load a saved list



If Region is set this will be first Screen. This allows you to resume a saved list or start a new one.

App Settings



Change Region

Drop Down

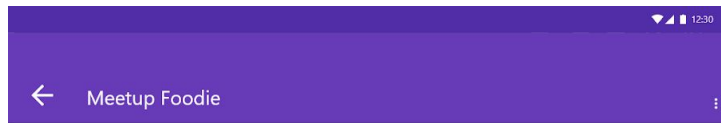
Customize Base Prices

- Papa Johns
- Pizza Hut
- Chik-fil-A
- Chipolte
- BBQ



This is accessed from 3 dot menu, this is first screen of settings. If only need to change region the user can select drop down and press up or back arrow to return to last screen. If they select a restaurant then the Next settings screen will be displayed.

App Settings Base Prices



Papa Johns

Medium Thin Crust Price

Medium Reg Crust Price

Large Thin Crust Price

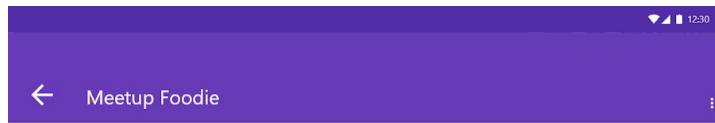
Large Ingredients Price

Medium Ingredients Price



If Restaurant is selected in Main Settings then this screen will come up to let you customize the base pricing. When inputting custom prices the data will be validated to be proper currency format. If not in correct format the data will not be saved and will provide guidance to user to change it.

App Resume



Shopping List

Papa Johns

Large Thin Crust Cheese Pizza - \$10

Large Reg. Crust Peperoni - \$12

Grocery:

Bag of Ice: \$2

Package of Individual Chips: \$8

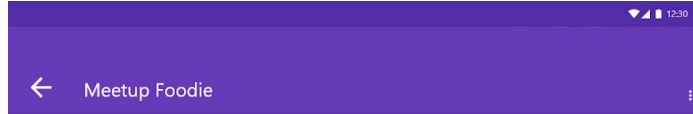
Total: \$32

Add Shopping List to Widget



On Normal Start of App this screen will be shown if Resume or Load is selected. The resume is last list and load will allow you to select older lists. This allows you to add the shopping list to widget if not already displayed. The up arrow will allow you to edit the list.

App Choose a Restaurant



What Restaurant Sounds Good?

Papa Johns
Pizza Hut
Chik-fil-A
Chipolte
BBQ

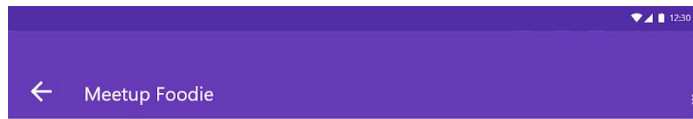
Customize Base Prices

Papa Johns
Pizza Hut
Chik-fil-A
Chipolte
BBQ

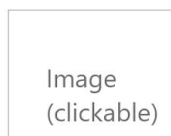


Since most meetups only one organizer picks of the food, you select the restaurant for pickup or delivery. Not many meetups have buffets from multiple restaurants. I have also include shortcuts to adjust base prices but may drop this when app is done depending on space constraints.

App Build a Shopping List



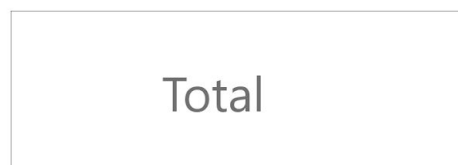
Papa Johns: Please Select your items



Quantity: (DropDown)

Item Total: \$200

+ Add Item



On App Start this screen will be selected if New list button is pressed. I have included an example item that is already added. The user can customize the item added by selecting image. Images will include Content Descriptions. The new item is added by clicking the Add Item text.

App Add New Item

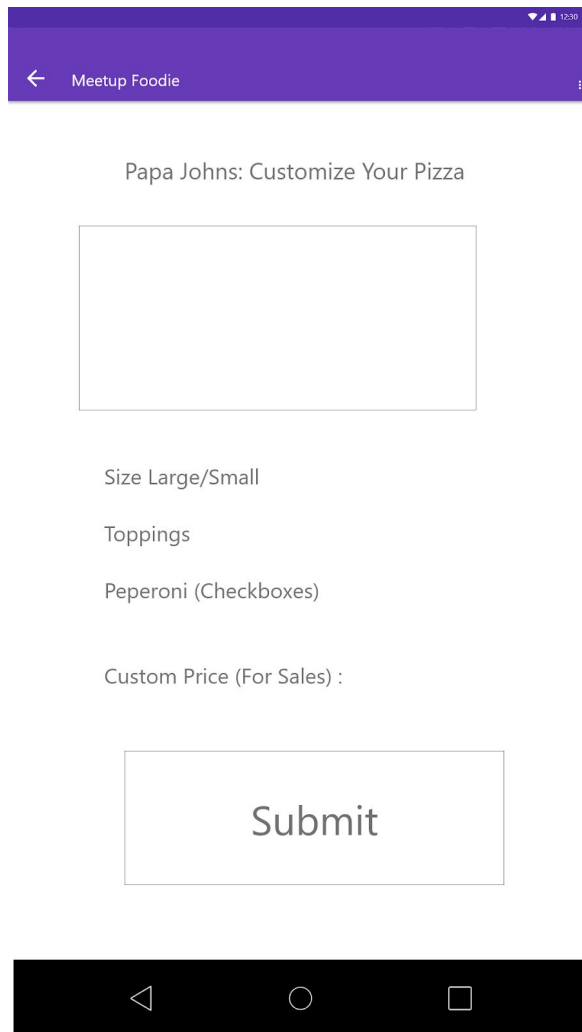
Meetup Foodie

Papa Johns: Select your Pizza or Side

Submit

If Add new item button is pressed this screen will appear. This will have image of various food items that be highlighted when selected. Once selected the user pressed submit to add the food to the shopping list screen and return to that screen. Images will include Content Descriptions.

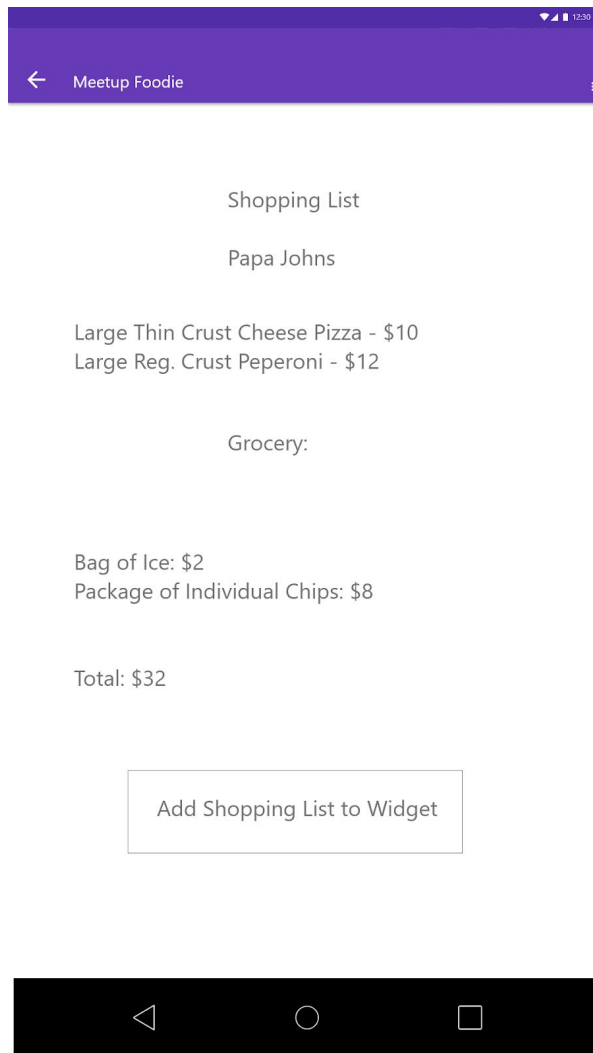
App Click Image



The screenshot shows a mobile application interface with a purple header bar. The header contains a back arrow, the text "Meetup Foodie", and a menu icon. Below the header, the title "Papa Johns: Customize Your Pizza" is displayed. The main content area includes a large empty rectangular box for a pizza image. Below this box are four text labels: "Size Large/Small", "Toppings", "Peperoni (Checkboxes)", and "Custom Price (For Sales) :". At the bottom of the form is a large rectangular button labeled "Submit". The entire app interface is shown within a black frame representing a mobile device, with standard Android navigation icons (back, home, recent apps) at the very bottom.

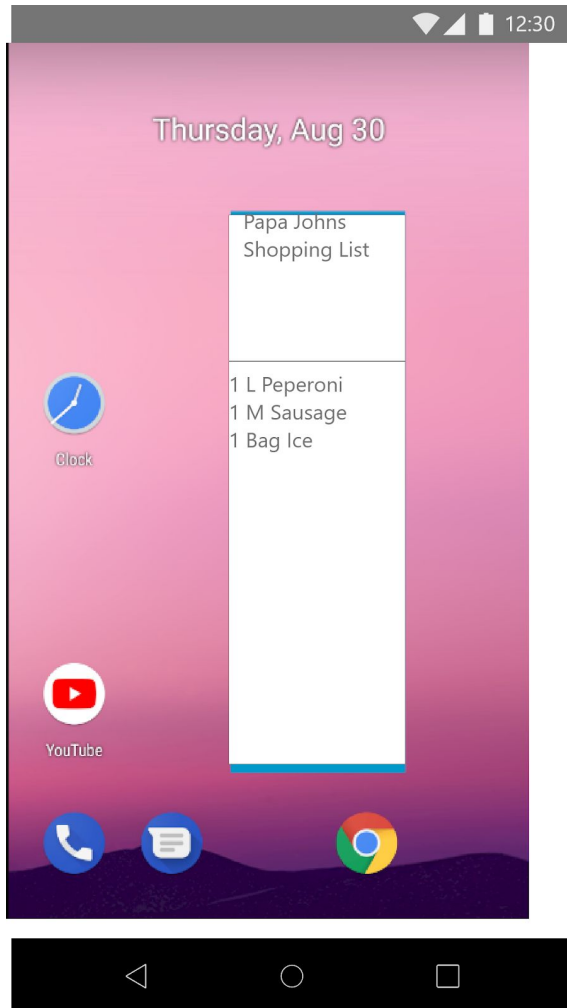
On Main Shopping List page once a new item is added the user can customize the selection. You can change options such as size of pizza or toppings on pizza. Also if there are sales on that type of pizza you enter a customized price for the pizza. This screen may differ depending on the type of food.

App Total



This screen gives you the grand total of items selected. It has one button that will add items to widget to create a shopping list so you can quickly access from home screen. If this page is entered from the widget the up arrow will function to edit the list. Back arrow will function as normal.

App Widget



The widget will give you quick access to your shopping list that you created using the app. On click anywhere in the list part of widget you will be returned to total page of app for further details about list such as price.

Key Considerations

How will your app handle data persistence?

The list of food prices and base prices will be stored in a Firebase Realtime Database. The user's customizations and saved food lists will be stored in a room database and shared preferences.

Describe any edge or corner cases in the UX.

When the user clicks on the widget it will return to the saved list screen. I may only allow settings changes when on the intro pages. Changing the base prices may break things if allowed in other areas.

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

Library/Program Name	Reason for including	Version
Espresso	Used for UI testing	3.0.2
Room	Used for persistent app data store	1.1.1
LifeCycle	LiveData and Viewmodel for Room with ViewModel implementation	1.1.1
Firebase Analytics	To see how app is used in the wild	16.0.3
Firebase Realtime DB	Stores Global item constants that can change over time	16.0.1
Butterknife	Data Binding shorthand, makes code look cleaner	8.8.1
Gson	For converting Lists into or out of GSON/JSON formats	2.8.5
Android Studio	Main IDE for developing Android	3.1.4
Build.gradle	Main Build System for Android	Plugin / Gradle 3.1.4 / 4.4

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

I plan to use Firebase Realtime Database to have global constants such as base food types and prices. I will also use Firebase analytics to understand how my app is used and how to improve it.

Other Code Requirements

The Client requests the following code requirements that are not addressed in other sections.

1. App will be written in Java only.
2. All Strings will kept in Strings.xml and the layouts will use options such as end and start instead of right and left to support RTL languages.
3. App theme will extend AppCompatActivity
4. App will use standard android toolbar. It will display up arrow (where needed), title and menu (if needed).
5. Final Release will uploaded to github and tested on another computer to verify clean build.
6. App will use installRelease Gradle task to deploy.
7. App will be equipped with a signing configuration, and the keystore and passwords are included in the repository. Keystore will be referred to by a relative path.
8. App will only use dependencies that will be managed by build.gradle.

Next Steps: Required Tasks

UI tests will be implemented and errors will be resolved after every phase.

Task 1: Project Setup

1. Set up Room Database to save and retrieve Base Prices
2. Make sure Room is setup properly with Viewmodels so that database is not being queried when not necessary.
3. Set up Initial Region Screen to select and save Shared Preference
4. Set up menu to reset Region Preference and adjust Base Prices
5. Add both Settings Screens
6. Verify the database and shared preferences are working properly

Task 2: Implement UI for Each Activity and Fragment

1. Implement Rest. Selection Screen
2. Implement Item Selection Screen
3. Implement Item Detail Screen to modify item - (pizza add ingredients, etc)
4. Implement Shopping List/ Total Screen
5. Implement Shopping List Widget

Task 3: Store Regions Online

1. Implement the Firebase Database so app can improve over time
2. Validate Data coming in from Firebase since remote source may give bad data.
3. Retrieve Base Regions and Prices from Firebase
4. Implement a syncing schedule for Firebase to limit network calls
5. Verify Firebase Calls are Async and not hanging up main thread.
6. Verify App works offline.

Task 4: Make it look better

1. Make the UI use material Design Concepts for consistent look.
2. Select colors that match theme of product
3. Possible Theming for Different Regions

Task 4: Beta Testing and Release

1. Connect and use Firebase Analytics, make sure only one instance is used.
2. Remove Unused services and libraries (from Build.Gradle)
3. Invite users to beta test app
4. If app is looking good and no glaring errors release to app store