**GitHub Username**: hakeemtunde

# Kiipa

## Description

Kiipa is an app that help grocery store to keep and track stock record. The app will help store sales person or store manager to track stock item level, stock expiration, sales transaction, credit customers or stock issuance records.

## Intended User

Grocery traders, Store keepers

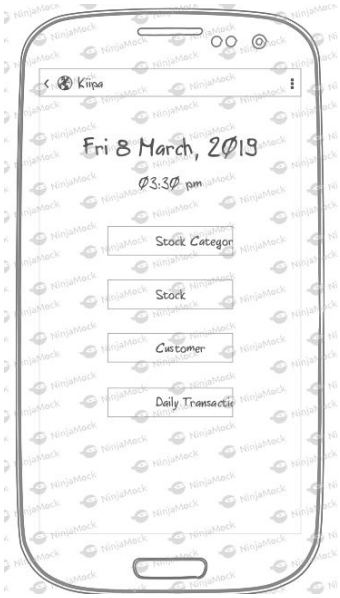
## Features

List the main features of your app. For example:

- Add/Update stock categories
- Add/Update stock items and stock measures
- Save credit customers detail
- Add/remove item to cart
- Make sales/checkout or issuance
- Keep transaction records
- Monitor and notify user on low stock level/expiration
- Display five (5) most recent transactions (home screen widget)
- Display Daily transaction summary

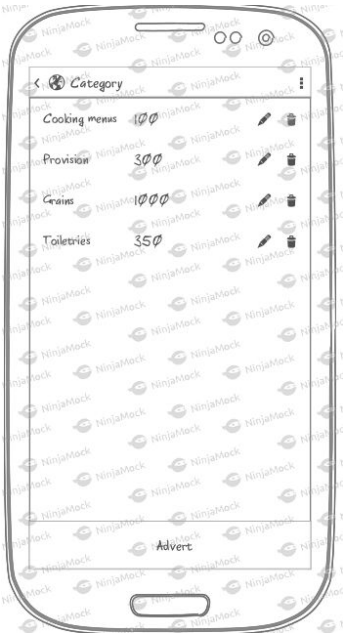
# User Interface Mocks

## Screen 1



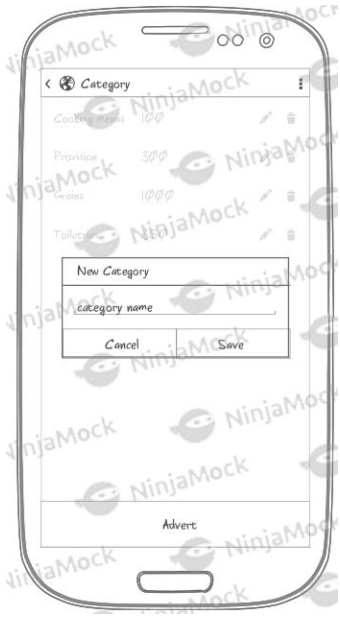

Application dashboard (Main UI)

### Screen 2




Category List UI

### Screen 3

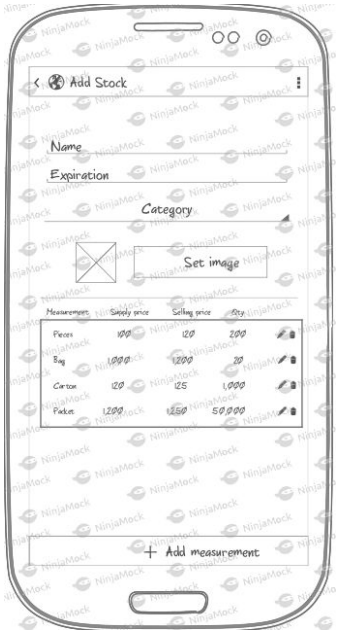



Dialog UI for adding new stock category

## Screen 4


Settings UI

## Screen 5

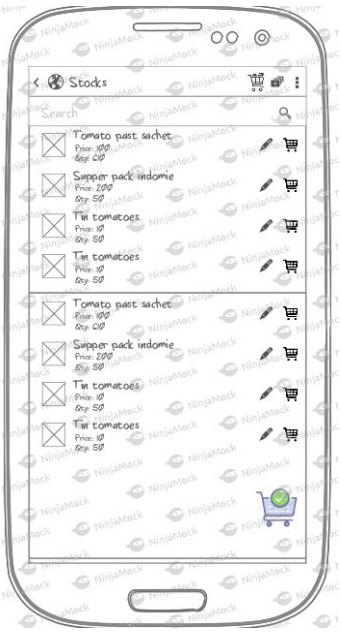



UI for adding new Stock item

## Screen 6

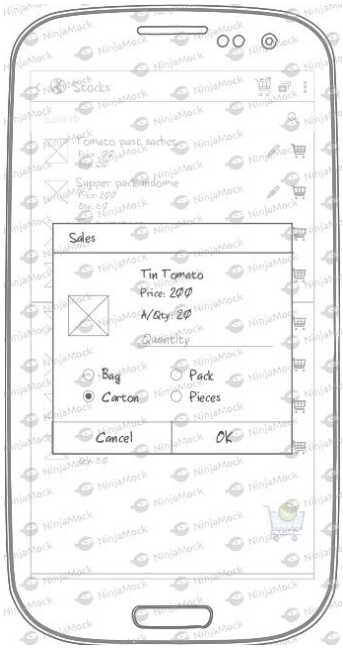

UI for adding stock measuring parameters (Pieces, Bag, Carton, Serchet, Packet etc) and quantities

## Screen 7



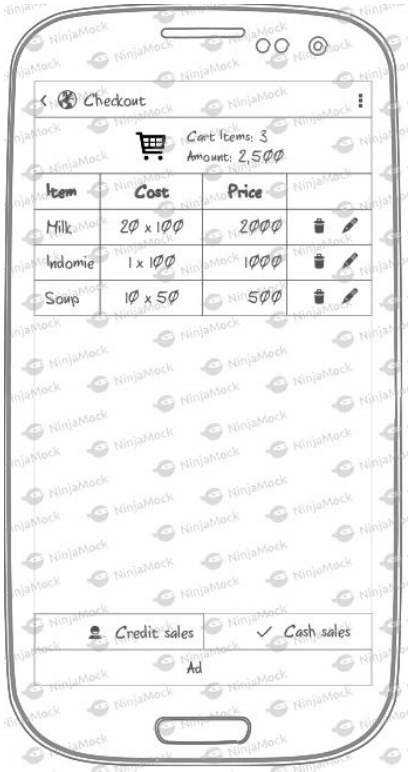

Stock item list UI (for stock selection)

## Screen 8




Stock parameter input & selection UI (on selecting a stock this UI is presented for the user to select measures and quantities )

## Screen 9

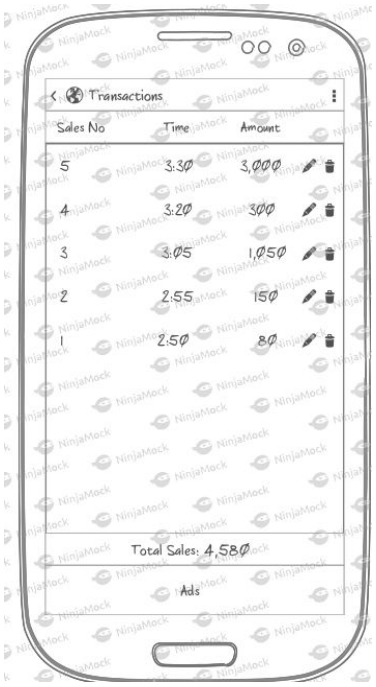



Checkout screen for selling stock(credit/paid customer)

## Screen 10

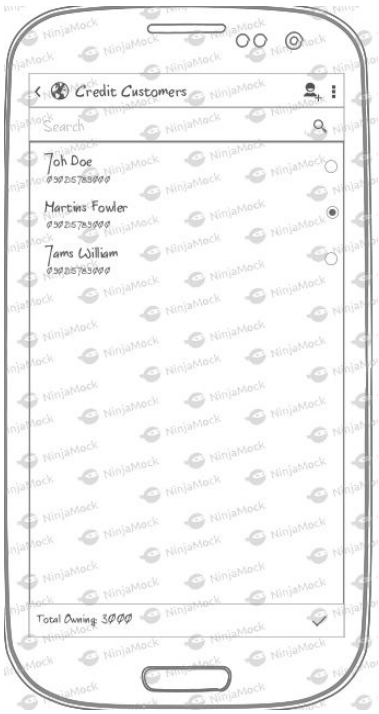

Checkout screen for store keeper(stock issue screen)

## Screen 11




Daily Transaction list

## Screen 12




Customer list (UI that list credit customers)

## Screen 13


App home screen widget

## Key Considerations

- App will be written solely in java programing language
- App will utilize stable release versions of all libraries, Gradle, and Android Studio.

| | |
|---|---|
| Gradle | 4.10.1 |
| Android Studio | 3.3 |
| Android Support Library | AndroidX |

- App keeps all strings in a `strings.xml` file and enables RTL layout switching on all layouts.

**How will your app handle data persistence?**

Application will use Sqlite/Room for storing data

**Describe any edge or corner cases in the UX.**

- Validation of input values and show a nice error state of the view fields
- Checkout button prompt a notification when the cart is empty
- Loader is show when performing long operation

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

- ButterKnife for view binding convenience
- Room for easy implementation of database operation
- Glide for loading stock image

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

Analytics and Admob(Banner ad)

## Next Steps: Required Tasks

## Task 1: Project Setup

Create and do project configuration

Subtasks

- Create android project
- Configure libraries

## Task 2: Implement UI for Each Activity and Fragment

Build all necessary User interface
Subtasks

- Build UI for MainActivity
- Build UI for StockCategoryActivity (Listing, Adding, editing)
- Build UI for SockItemActivity (Listing, Adding, editing)
- Build UI for CustomerActivity (Listing, Adding, editing)
- Build UI for SalesCheckoutActivity
- Build UI for DailySalesTransactionActivity (Listing, editing)
- Build UI for app Settings preference

## Task 3: Database implementation

Implement persistence logic

Subtasks

- Create database
- Create tables (category, measure, stock, customer,transactionlog)
- Create models for the tables
- Implement db operations using Room
- Write test

## Task 4: business logic implementation

Implement business logic for performing db operations (insert/update/delete)

Subtask

- Create class/logic for fetching, adding, update and delete stock category
- Create class/logic for fetching, adding, update and delete stock item
- Create class/logic for fetching, adding, update and delete customer
- Create class/logic for fetching transactions
- Implement application preference settings logic
- Implement cart logic for sales
- Implement checkout logic for sales
- Write test

## Task 5: Wire UI to business logics

Connecting the UI actions to the business logic and make necessary input data validations

Subtasks:

- Wire StockCategoryActivity to it business logic
- Wire SockItemActivity to it business logic
- Wire CustomerActivity to it business logic
- Wire SalesCheckoutActivity to it business logic
- Wire DailySalesTransactionActivity to it business logic
- Write test

## Task 6: AppBar implementation

Implement AppBar for the application
Subtasks:

- Create AppBar for the app
- Add Icons for Cart
- Implement FAB for checkout action
- Implement logic for adding item to cart
- Implement logic for making sales transaction i.e checkout

## Task 7: App home screen widget implementation

Implement home screen widget for listing five(5) most recent sales/issued stock
Subtasks:

- Create StockWidget xml file
- Implement the StockWidget service

## Task 8: Google Admob implementation

Implement Google Admob
Subtasks:

- Add play store dependencies
- Do adMob configurations
- Give network permission
- Implement Banner ad

## Task 9: Implement JobScheduler service

Implement service for monitoring stock level and expiration notification
Subtasks:

- Implement JobScheduler for monitoring stock level & expiration
- Implement notification logics

---

**Submission Instructions**

- After you've completed all the sections, download this document as a PDF [ File → Download as PDF ]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"