How to Use this Template

i. Make a copy [File → Make a copy...]
ii. Rename this file: “**Capstone_Stage1**”
iii. Replace the text in green

Submission Instructions

i. After you’ve completed all the sections, download this document as a PDF [File → Download as PDF]
ii. Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
iii. Add this document to your repo. Make sure it’s named “**Capstone_Stage1.pdf**”

GitHub Username: theeheng

StoCount

Description

StoCount is a mobile application which helps user to keep record of their stock taking on a mobile phone.

Intended User

StoCount is perfect for independent shopkeeper, small restaurant or home user which interested to keep track of store inventory electronically on a portable device such as mobile phone or tablet.

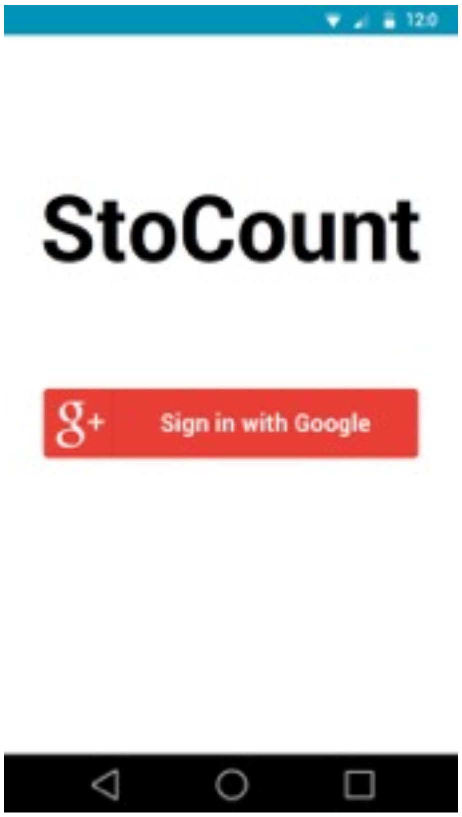
Features

- Keep a list of product use for stock taking
- Allow to do stock count with barcode scanning
- Able to do stock count offline
- Connected to Tesco (UK largest supermarket) API, provide wide range of household product information (include barcode info)
- Integrated with Google+ login

User Interface Mocks

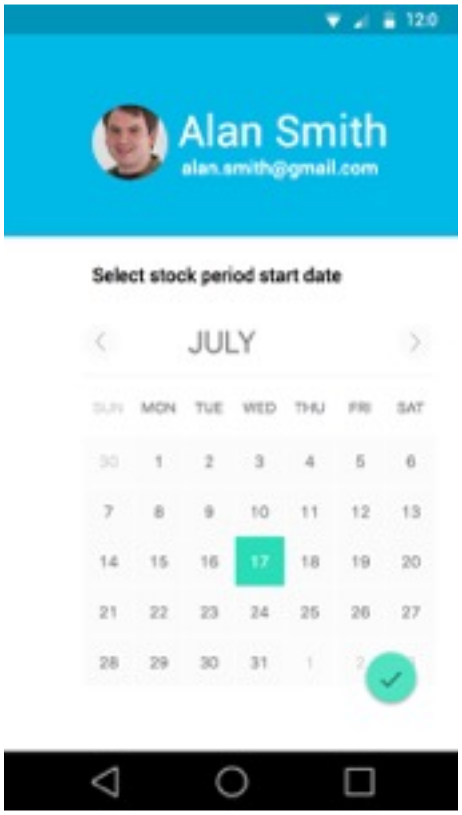
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1 : Login



When users first launch the application, they will be as to login using their google account. This is part of integrating the user identity using google play service.

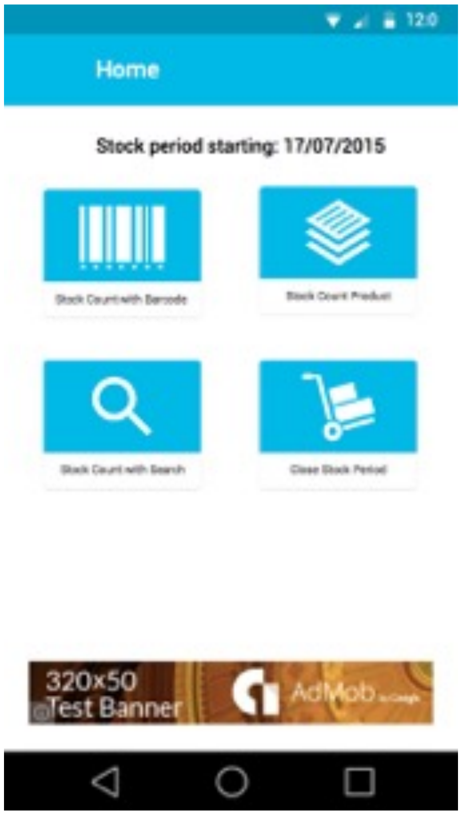
Screen 2 : Selecting Stock Period Start Date



SUN	MON	TUE	WED	THU	FRI	SAT
30	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	

Once the users had login, they are required to select the start date of the first stock period.

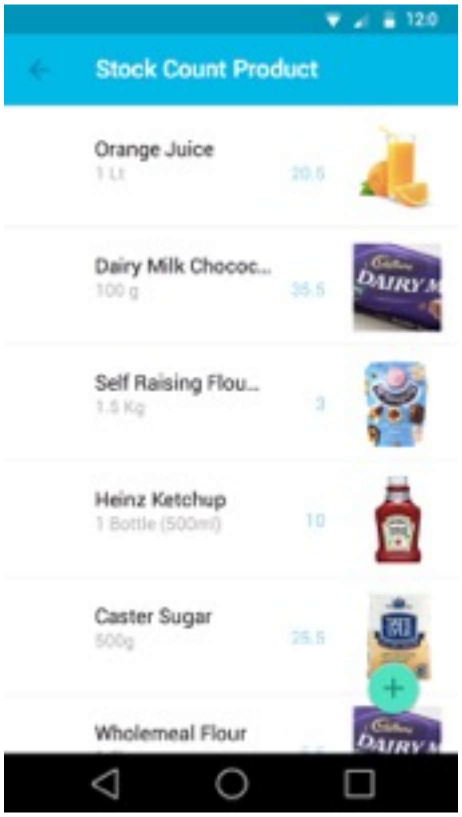
Screen 3 : Home



This is the Home Screen as well as the main activity which will be launch on app start (except on the first time). There are 4 options: Stock count using barcode scanning, Stock count using product name search, View stock count product and Close Stock Period.

To monetising the app, it is practical to show an ad banner on this activity because it is an activity that will achieve the highest view yet the ad banner will not obstruct the main content.

Screen 4 : Stock Count Product



Orange Juice
1 Lt 20.5

Dairy Milk Chococ...
100 g 35.5

Self Raising Flou...
1.5 Kg 3

Heinz Ketchup
1 Bottle (500ml) 10

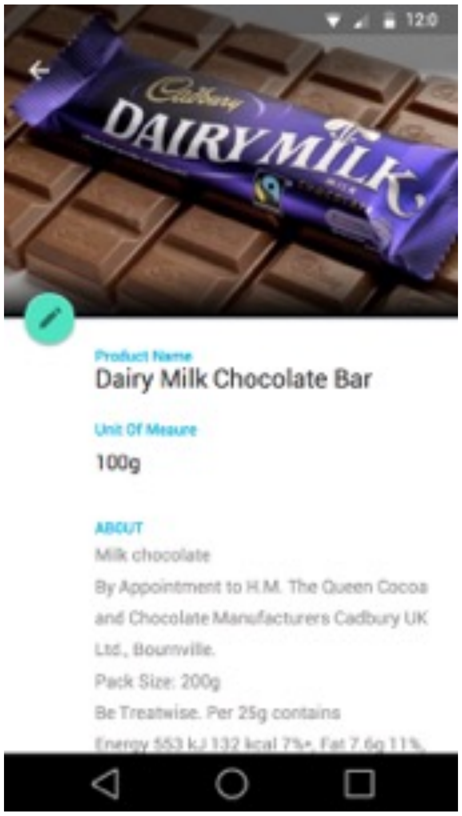
Caster Sugar
500g 25.5

Wholemeal Flour

This activity will show when user select Stock Count Product button on the home screen. It display all the product in a list view manner, include product name, unit of measure, current stock count and the product thumbnail. User can tap on the fab button to add a new product. The fab button will show a floating action menu to allow user to choose between: Add product using barcode, Add product using search product name, Manual input product info.

Both option (Add product using barcode, Add product using search product name) will use Tesco API to create product record automatically.

Screen 5 : Product Info / Edit Product



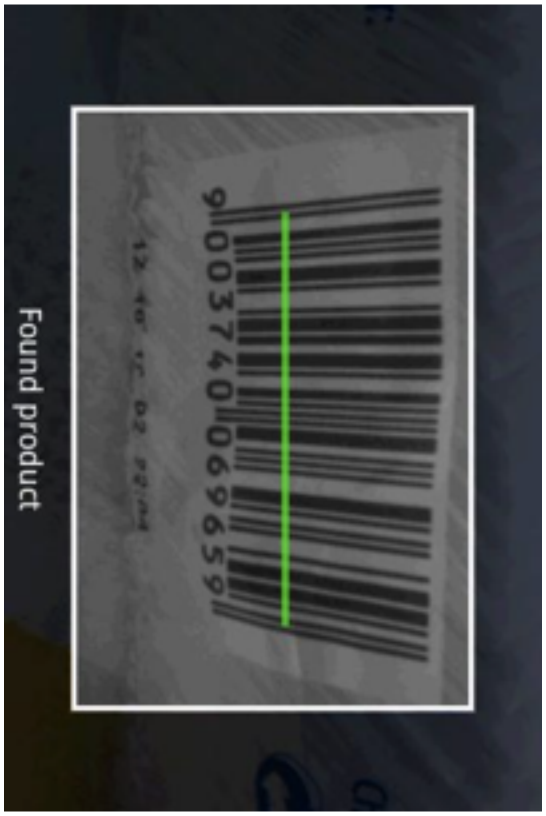
Product Name
Dairy Milk Chocolate Bar

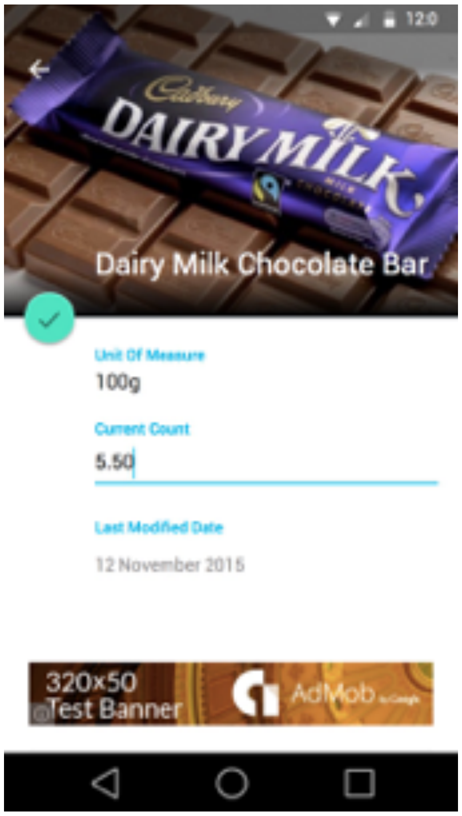
Unit Of Measure
100g

ABOUT
Milk chocolate
By Appointment to H.M. The Queen Cocoa
and Chocolate Manufacturers Cadbury UK
Ltd., Bournville.
Pack Size: 200g
Be Treatwise. Per 25g contains
Energy 553 kJ 132 kcal 7%*, Fat 7.6g 11%,

Product info activity is shown after a product is created (using barcode / search) or when user selected a product from the stock count product activity. This activity will allow user to view, edit and save the product information. When user tap on the edit fab button, product name, unit of measure and product description will be shown as editable text box.

Screen 6 : Stock Count Entry

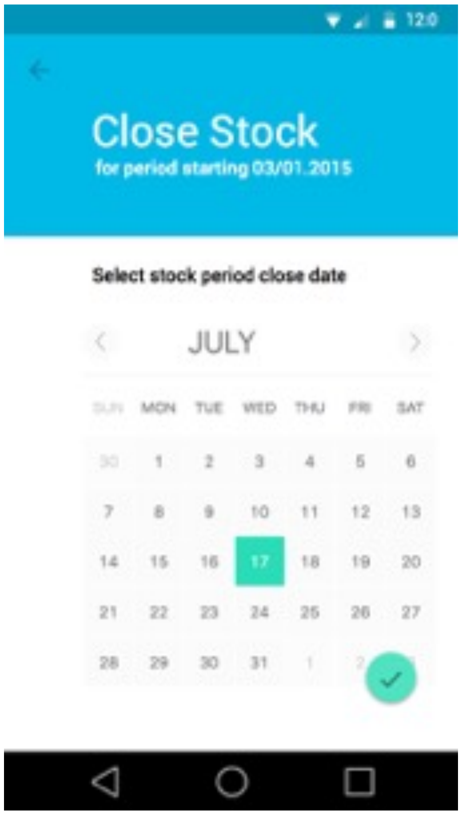




Unit Of Measure
100g
Current Count
5.50
Last Modified Date
12 November 2015

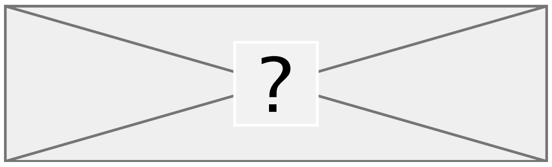
When user select Stock Count with barcode, it will launch the barcode scanning UI provided by Zxing embedded in the application. Once the barcode is recognised, it will trigger the content provider to search for the relevant product. Finally it will display the stock count entry activity which allow user to enter and save the current count for that product.

Screen 7 : Close Stock Period


Close Stock
for period starting 03/01.2015
Select stock period close date
JULY
SUN MON TUE WED THU FRI SAT

This activity will show when user select Close Stock Period from the home activity. It allow user to select the closing stock period date and create new stock period with the following date as stock period start date.

Widget Screen



This widget allow user to keep track all their stock count product and its stock period current count. This widget will refresh every 15 seconds looping through all the stock period product.

Key Considerations

How will your app handle data persistence?

All the data in the application is stored in a local repository using SQLite. A content provide will be use to connect the local repository and the main application. Here are the main entities which will be store in the local repository:

- User
- Product
- Stock Period
- Stock Period Count

Describe any corner cases in the UX.

A widget will provide the stock count information of a product. When use tap on the widget it will open the application displaying the product info and allow user to change the stock count of that product.

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

Here is a list of libraries and API which will be use by the application:

Tesco API - Tesco is one of the largest supermarket retailer in UK and its API allow user to search product information based on barcode and product name.This come in handy when user want to insert product info for the first time when using the application by just scanning the product barcode or search by product name.

Glide - This is a powerful library which will handle all the image loading and caching .

Retrofit - This is type-safe HTTP client library which is useful to wrap around any WEB API call.

Butterknife - This is a view 'injection' library which is widely use in android application for binding field and method with android views using annotation.

Zxing - A open source library which provide the ability of scanning barcode.

Google Play Services - The application will use the identity platform provided by google to authentication the user. Also an ad banner will be show on application using the admob library provided by google play service.

SQLite - This is library is mainly use to create local repository, table structure, querying data and managing CRUD operation.

Next Steps: Required Tasks

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

Task 1: Project Setup

1. Create a new Android Studio project using the following information:

Application name: StoCount
Company Domain: nanodegreeapp.hengtan.stocount.com
Package name: com.hengtan.nanodegreeapp.stocount

Platform : Phone and Tablet
Minimum SDK: API 16 - Android 4.1 (Jelly Bean)

2. Modify build.gradle file in app folder to include all the dependencies libraries.

Task 2: Create a Java Library using the source code provided by Zxing

1. Create a Java library call 'zxing' and import all necessary code for barcode scanning from https://github.com/zxing/zxing

2. Modify the settings.gradle file found in the main project folder by adding the ':zxing' in the include statemetn.

3. Modify build.gradle file in app folder to include the following statement within the dependencies body:

- compile(project(':zxing)) { transitive = true }

Task 3: Implement UI for Each Activity and Fragment

Start building the following screen:

- Build UI for Login Screen
- Build UI for Stock Period Screen
- Build UI for Home Screen
- Build UI for Product Screen : including add product fragment, product info fragment, stock entry fragment
- Build UI for Product List / Search Result Screen
- Build UI for Close Stock Period Screen

Task 4: Implement data layer: Database structure, Content Provider :

1. Create a database helper class which contain all the code for creating the database it self and all necessary tables.

2. Create a content provider which will connect to the database helper class for managing all the queries, insert, delete, update operation.

3. Update UI to use content provider for displaying data.

Task 5: Implement query to Tesco API :

1. Create a wrapper class using Retrofit for making calls to Tesco API.
2. Update add product fragment to use wrapper class to connect to Tesco API.

Task 6: Implement Google Play Services

1. Add identity verification using google play services to the Login Screen
2. Add an add banner to the Home screen

Task 7: Create widget for display product stock count

1. Create a collection widget which will loop through the each product and displaying a thumbnail version of the product picture, product name and its current stock count.

Submission Instructions

i. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
ii. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
iii. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"