A MINI-PROJECT REPORT ON

“Food Expiry Tracker”

SUBMITTED BY

Riteshkumar Singh 18IT1068

Rupal Sonje 18IT1086

Soham Salkar 18IT1050

Supervisor:

Mr. Sachin Bhopi

Department of Information Technology

Dr. D. Y. Patil Group’s

Ramrao Adik Institute of Technology

Nerul, Navi Mumbai

(Affiliated to University of Mumbai**)**

(2021)

CERTIFICATE

This is to certify that the project entitled ‘Food Expiry Tracker’ is being submitted by Soham Salkar 18IT1050, Rupal Sonje 18IT1086, Riteshkumar Singh 18IT1068 to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of ‘B. E. I. T’ in “Android Apps Development Lab (Mini Project)”.

Project Guide	External Examiner	Head of the Department
(Sachin Bhopi)	( )	(Dr. Ashish Jadhav)

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Name and Roll No. of Students
Signature

1. Soham Salkar (18IT1050) ( )
2. Riteshkumar Singh (18IT1068) ( )
3. Rupal Sonje (18IT1086) ( )

Date:

Place

ACKNOWLEDGEMENT

The project "**Food Expiry Tracker"** is the creative work of many minds. A proper synchronization between individuals is a must for any project to be completed successfully. One cannot imagine the power of the force that guides us all and neither can we succeed without acknowledging it.

We would like to express our gratitude to Principal **Dr. Mukesh D. Patil** and **Dr. Ashish Jadhav,** our Head of the department, Information Technology engineering for encouraging and inspiring us to carry out the project in the department lab.

We would also like to thank our Guide **Sachin Bhopi**, Department of the Information Technology Engineering for her expert guidance, encouragement and valuable suggestions at every step.

We also would like to thank all the staff members of the Department of Information Technology Engineering for providing us with the required facilities and support towards the completion of the project.

Last but not least we are thankful to our parents and friends for their constant inspiration, encouragement, and well wishes by which we have made a challenging project.

Riteshkumar Singh 18IT1068

Rupal Sonje 18IT1086

Soham Salkar 18IT1050

PREFACE

We take great opportunity to present this Mini Project report on"**FOODEXPIRY TRACKER"** and put before readers some useful information regarding our project.

We have made sincere attempts and taken every care to present this matter in precise and compact form, the language being as simple as possible. We are sure that the information contained in this volume will certainly prove useful for better insight into the scope and dimension of this project in its true perspective.

The task of the completion of the project, though difficult, was made quite simple, interesting, and successful due to the deep involvement and complete dedication of our group members.

TABLE OF CONTENTS

TABLE OF CONTENTS

ABSTRACT

In India, the household food waste estimate is 50 kg per capita per year, or 68,760,163 tonnes a year. One of the reasons for this food waste is that the consumer forgets the expiry date of the food after purchase and does not consume the food before it expires, and when he/she finds out that the food is expired, it is tossed out. In the proposed system the user has to input some information regarding food products like product name, date of purchase, and date of expiry. On the basis of that data, an automatic notification will be sent to the customer's smartphone several days before the purchased food expires.

SRS DOCUMENT

1. INTRODUCTION

Food wastage due to expiration is an issue that is evident in almost every household and needs to be addressed. It has been ascertained by a study conducted by National Resources Defense Council (NRDC) that 931 million tonnes of food was wasted globally in 2019.In India, the household food waste estimate is 50 kg per capita per year, or 68,760,163 tonnes a year. The household food waste estimate in the US is 59 kg per capita per year, or 19,359,951 tonnes a year, while for China these estimates are 64 kg per capita per year or 91,646,213 tonnes a year. Not only does this imply that globally $165 million each year, but also 25 percent of all freshwater and enormous amounts of chemicals, energy and land are being thrown away. It is estimated that if wastage of food is reduced 5 percent, it would be enough to feed 4 million people Anecdotal evidence suggests that the following are the major factors that drive household losses -

i. undervaluing food

ii. unideal storage

iii. misjudged food needs

iv. partially used ingredients

v. bulk purchases

vi. poor planning, and

vii. over preparation

1.1 MOTIVATION

The motivations behind developing such an application are the following:

1. Economic Benefits
    a. Reduce over-purchasing - We can save a significant amount of money if the food we purchase is not wasted when we use it before it expires. It may also deter us from over purchasing.
    b. Lower disposal costs - If each household decreases food wastage there may be an overall reduction in the food disposal costs on the economy.
2. Social Benefits
    a. Feed people - By keeping track of when the food is going to expire, it prevents wastage and if we have surplus food we can donate it to food rescue organizations.

3. Environmental Benefits
    a. Reduce resource use essential to food production - Many resources are needed to grow food such as water, pesticides and energy. By wasting food, these resources are also wasted that went into growing it
    b. Reduce Landfills - By reducing food wastage we also in turn reduce Landfills which are a major source of the greenhouse gas Methane.

1.2 PURPOSE

The purpose of this SRS is to outline both the functional and non-functional requirements of the Food Expiry Tracker. In addition to said requirements, the document also provides a detailed profile of the external interfaces, performance considerations and design constraints imposed on the subsequent implementation. It is

the intention that the presented set of requirements possesses the following qualities; correctness, unambiguousness, completeness, consistency, verifiability, modifiability and traceability. Consequently, the document should act as a foundation for efficient and well-managed project completion and further serve as an accurate reference in the future. The primary audience of this SRS document will be the development team employed to implement the application. It will not only provide an extensive capacity for project planning and progress assessment but it will further assist with developer/stakeholder interactions. The secondary document audience comprises the stakeholders of the project, that is, the user. To this audience group, this SRS should convey and confirm the required functionality and represent a contractual agreement between the involved parties.

1.3 SCOPE

It has been proven by a study that household food wastage is less in developing countries as compared to developed countries. Hence, a smart phone application has been proposed by this thesis, which will help keep track of food expiration dates and notify the user when a food is about to expire so that food wastage is prevented. This application will enable the user to scan the barcode of the food product to automatically discover the product's name. The user will then be able to use the optical character recognition feature to capture the date of expiry from the product, after which the user can add the item to a database and the date is added to the calendar which will then enable notifications to pop up before food is about to expire. This project is novel because it tries to implement an optical character recognition module to identify the expiry date, which has never been used in this kind of application before, along with a barcode scanning module.

The definitive objective of this project is reducing food wastage by developing an application to track food expiration. This application will be developed with the consideration that it does not take up significant time for the users to feed the food

item details into the application.

2. Overall Description

This section will give an overview of the Food Expiry Tracker application. In particular, the product has been put into perspective through a detailed assessment of the system, user, hardware, software and communication interfaces, memory considerations, operational modes and site adaptation requirements. Further, the characteristics of the system's end-users are discussed along with the identified system constraints and assumptions.

2.1 Product Perspective

As mentioned a comprehensive solution is needed to tackle food wastage and targeting households will significantly reduce a large amount of food that is wasted. The application needs to be very easy to use as well as it needs to significantly reduce user effort and time. Current applications do not implement OCR which may save a considerable amount of time in data entry. They also do not display the list of items efficiently. For this project, the design of the application would then consist of a Graphical User Interface (GUI) where the user can view the current list of items that is already present in the database. In order to add a new item the user will have options such as scanning the barcode of the product, scanning the expiry date of the product and finally adding the product to the database to get the notifications.

2.2 Product Functions

Given below are the major functions that can be performed. The system will:

- Allow users to add food items.
- Allow users to add food items using ocr.
- Allow users to delete food items.

- Allow users to edit food items.
- Allow users to add expiry date manually.
- Allow users to add expiry date through scanning.

2.3 Assumptions and Dependencies

One assumption is that customers will have an android mobile phone containing a camera to scan the QR code and enough storage to download the application.

3. External Interface Requirements

3.1 Hardware Interfaces

Our system will directly interact with the hardware devices like mobile phones, tablets, etc. There is no need for other hardware devices. Android Mobile Phone is required.

3.3 Software Interfaces

- For Database services, the system shall use SQlite database latest version.

4. System Requirements

The components of this application are as follows:

1. The Application with GUI

   This is the main graphical user interface of the application. When the application is launched, the initial interface consists of a view that shows the list of items that are already added in the database. The list of items will be sorted in the order of closest date of expiry and will also have color bar indicators that will make it easier to know which items are high priority. There will be a settings option where we can set different preferences of the application. There will also be an option of auto removal of items once they

expire and the number of days after expiry that an item should be removed from the list as well as the database. In addition to this, there will be an option in the view to add new items to the list which when clicked on will open a new view where the details of the new item can be filled out. This view will have buttons that will help users to access other modules of the application such as the barcode scanner and the optical character recognition module for automatic product discovery as well as automatic expiration date discovery respectively. Furthermore, it will also have the feature to manually enter the product name if the product name is not automatically discovered and to manually enter the expiration date if the date is not recognized by the OCR. There will also be a button to finally save all the information of the food item to the database.

2. Barcode Scanner Module :

   For scanning the barcode information on the back of the food products this project uses Google's ZXing (Zebra Crossing) multi-format 1D/2D barcode image processing library implemented in Java. This library is open source and very easy to integrate in the project but requires an additional application called Barcode Scanner to be installed on the phone. It is able to read both UPC (Universal Product Code) barcode format as well as EAN(European Article Numbering) barcode format, now known as International Article Numbering. The authors of ZXing made it very easy for it to be integrated with other projects. It is achieved with the help of using Intents. Therefore, a barcode is scanned by calling the Barcode Scanner application via Intents. The authors provide a small library of code, which correctly handles all the important details, such as setting category flags and handling the case where the Barcode Scanner application is not installed. This library of code makes sure to prompt the user to install the Barcode Scanner application if it is not already installed on the smart phone as the ZXing library requires it to work completely. The scanning result is handled very easily by using some more code examples, given by the author, in the activity of the barcode scanner. The scanning result is then used by the JSON Parser module which contacts the web service to extract the name of the product. The graphical user interface of the barcode scanner has a view that accesses the smart phone's camera. It overlays a

selectable region in the view so that the barcodes can be comfortably scanned within the region. Once the barcode is scanned it displays a success message briefly along with a snapshot of the barcode. Then the main view where the item information is entered is pulled up.

3. Optical Character Recognition Module

   This app uses the Tess Two library which is based on the Tesseract OCR Engine, developed by HP and now maintained by Google, and Leptonica image processing libraries. The Tess Two library is a fork of Tesseract tools for Android, which provides a set of Android APIs and build files for the Tesseract OCR and Leptonica, and adds some functions to it. It downloads the already trained language files when integrated with the application. The OCR module will capture the date of expiry of the product and display it on the screen after some processing. The application then adds that date to the calendar on submission of the food item into the database. The Tess Two library was readily available as an open source library which was used to integrate with the project. The major challenge was that the library was written in C/C++ language and in order to integrate it into the project the Android Native Development Kit (NDK) was required. Android NDK is a companion tool to the Android Software Development Kit (SDK) and makes it possible to port libraries written in C/C++ to Android. Since OCR does CPU intensive work, the authors developed it using native code. The author of Tess Two also provided an example implementation of an Android project which was modified to suit this project's requirements. The graphical user interface of the optical character recognition module has a view that accesses the smart phone's camera like in the case of a barcode scanner. It also overlays a selectable region in the view so that the required date can be conveniently scanned within the region. Once the required date of expiry is in the small region, the camera button on the interface can be clicked to scan the date. If the scan results are accurate the done button can be clicked or if scan results are not satisfactory the skip button can be clicked to skip the scanning process. In this case, the main view where the food item information is added is shown and the date can be manually picked in the available date picker widget. A pure Java OCR could have been used but on

experimentation it was found that even after training it with a significant number of samples, it did not give satisfactory results and hence the Tess Two OCR was preferred over it. In addition to it, already trained language files were available for Tess Two.

4. SQLite

   SQLite is the database that is used if an application needs to manage its own private database. This database is used to store the list of items and their expiration dates. It's database management classes are readily available to store the application's content. Simple SQL queries can be used to query the database. If the application was working on data sent to it by a provider then only the generic database classes would have been used. Android comes with the sqlite3 database tool. This tool can be used to browse or run SQL commands on the device. It is run by typing sqlite3 in a shell window. A separate preferences file is used to save the preferences of the application. The database will only consist of one table with a column each for id, name and date of expiry.

5. Nonfunctional Requirements

5.1 Performance Requirements

The system must be interactive, and the delays involved must be less. So, in every action-response of the system, there are no immediate delays. Updates must be made with little delay.

5.2 Safety Requirements

The software is completely environmentally friendly and does not cause any safety violations. The UI will have a flexible font that can be zoomed so as to not over-constrain the eyes.

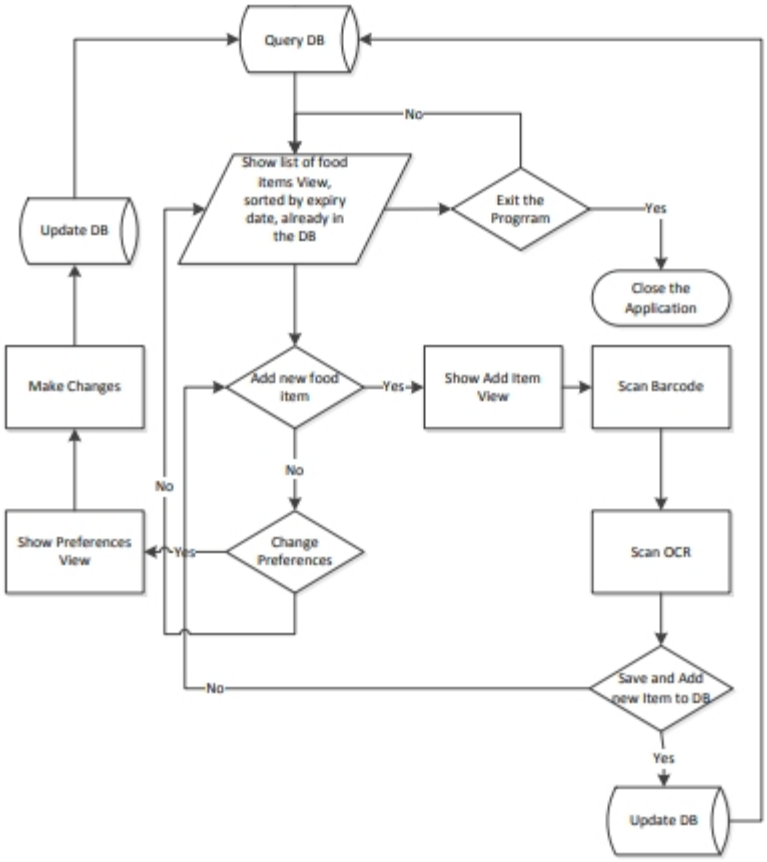
5.3 FLOW CHART

The flow chart of the application is given in Figure 5.3. It shows that when the application is launched it calls the main activity that has a view which displays the list

of all the food items that are already added to the database. This view has the items in the order of closest day of expiry, so the item which is going to expire soon will show up on top of the list. In the options, the user has a choice to add a new item, change the preferences for the application and exit from the application. If the user chooses to add a new item, the AddItem view shows up which has all the options to add the necessary information related to the food item.

First, the user can scan the barcode of the food item to be added by clicking the scan barcode button, which calls the barcode scanner module and opens up another view. This view accesses the smart phone's camera and overlays a selectable region in the view with a red horizontal line in the middle, so that the barcodes can be comfortably scanned within the region. Now the product with the barcode is placed in front of the camera so that the barcode is scanned, after which it displays a success message briefly along with a snapshot of the product with the barcode. The AddItem view shows up with the name of the product if found in the database, else it will prompt to add the name manually in the text field provided. Secondly, the user then tries to scan the date of expiry of the product by clicking on the scan OCR button, which calls the OCR module and brings up another view. This view accesses the smart phone's camera and overlays a small selectable region in the view so that the dates can be scanned easily. The product's date is placed in front of the camera so that the date can be scanned. Once the date is scanned, the user can either click on a button called continue if he deems the scan result to be satisfactory or try to scan the date again. If the result is not satisfactory, they will also have a button to skip the OCR step, which will then bring back the AddItem view. Meanwhile, in the background the scanned date will be processed and then converted to the appropriate date format. The date picker widget will be updated with the new date. If the widget has not been updated, 16 then the user can manually set the date. Once all the information is complete the user can then click on the save button to add the item to the SQLite database and update it. The main view is then shown again with the updated list. Now, if the user wants to change the application's preferences he can select the change preferences options in the settings. The change preferences view then comes up on the screen which has all the different preferences of the application. When we are done making the changes, we can press the back button which will then update the preferences in

the preferences file. The main view will show up again from where we can click on the exit button in the settings to exit the application.



5.4 Technical Process

Following would be the languages I would use to develop my application within the stipulated time period:

Front-end development: XML,

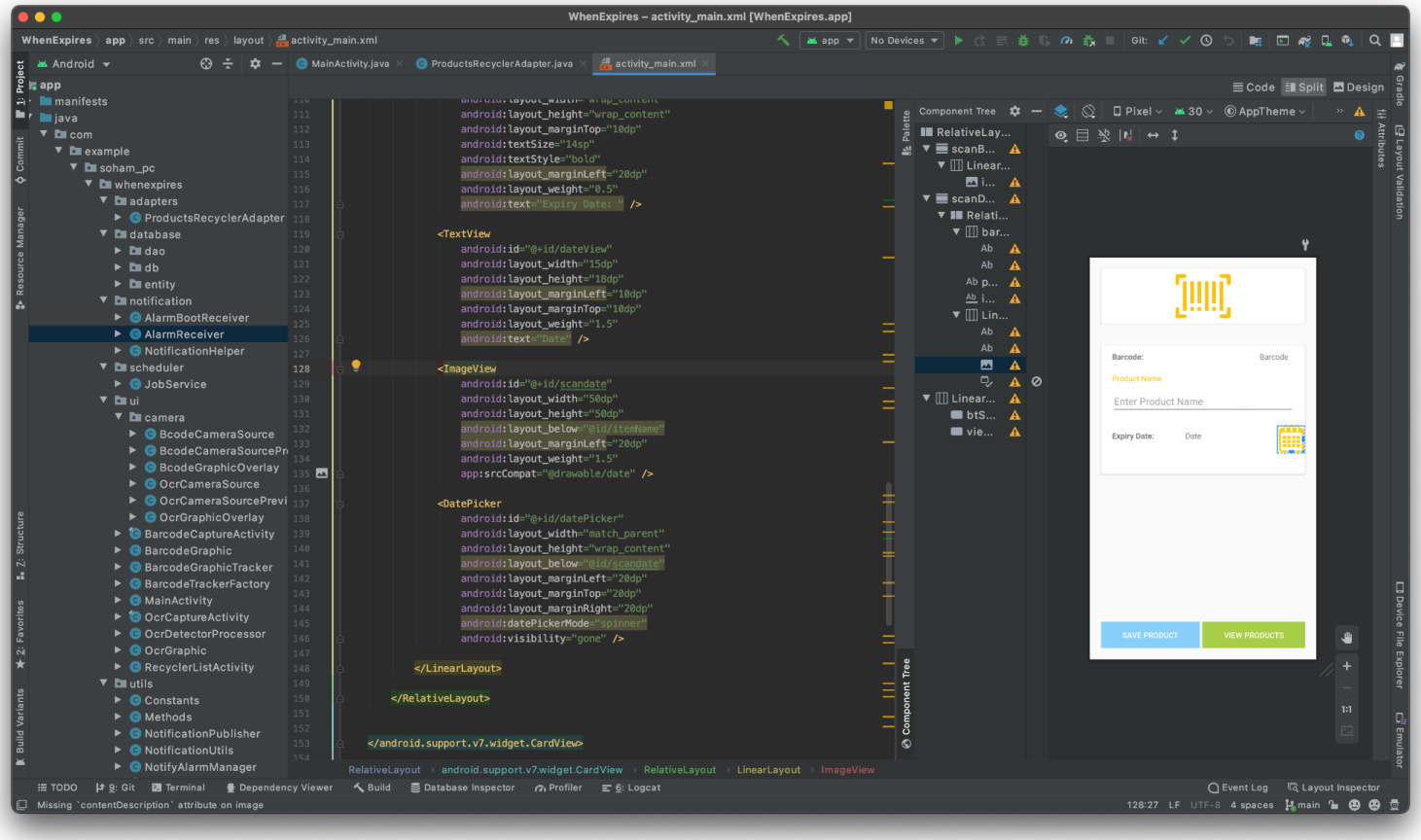
Back-end development: Java, SQLite.

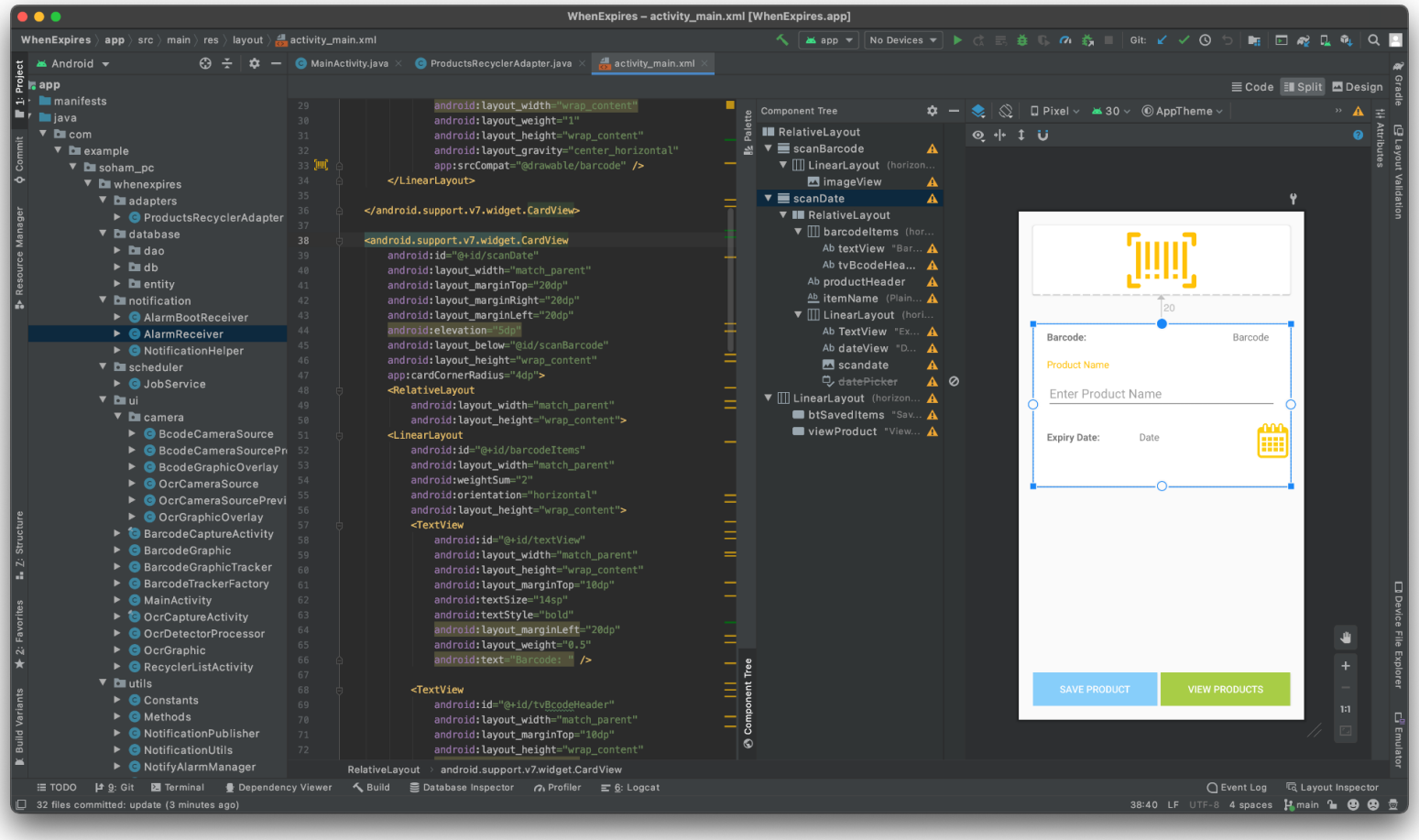
For Android app: Android Studio.

2. Implementation of Widget Box

Theory: Widgets enable users to interact with an Android Studio application page. There are various kinds of widgets, such as Buttons and TextViews. To see all the widgets at your disposal, create a new application project called “Widgets” and select "empty activity"

Output:

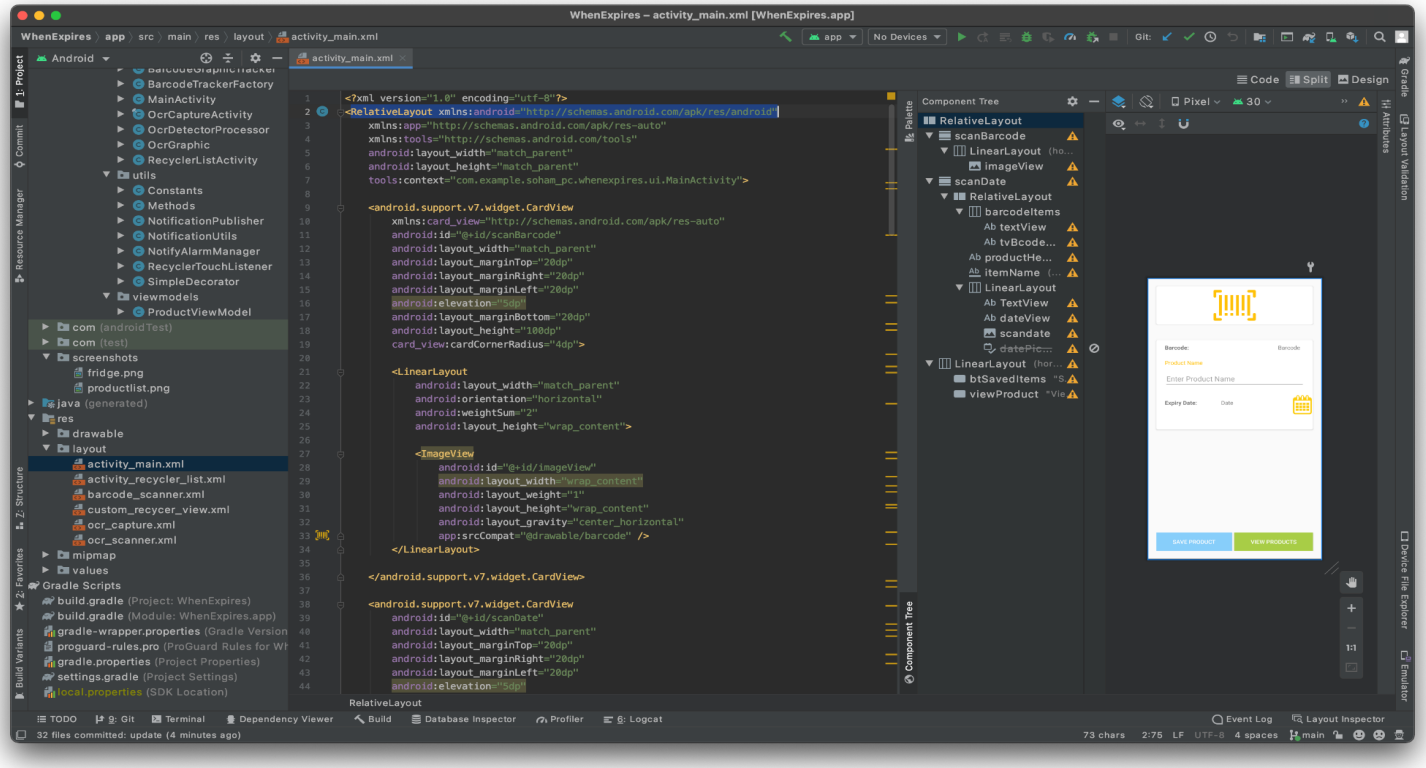



WhenExpires – activity_main.xml [WhenExpires.app]
MainActivity.java
ProductsRecyclerAdapter.java
activity_main.xml
Component Tree
RelativeLayout
scanBarcode
LinearLayout
imageView
scanDate
barcodeItems
textView
productHeader
itemName
scandate
datePicker
btSavedItems
viewProduct
Barcode:
Barcode
Product Name
Enter Product Name
Expiry Date:
Date
SAVE PRODUCT
VIEW PRODUCTS

3. Implementation of Layout:

A layout defines the structure for a user interface in your app, such as in an activity. All elements in the layout are built using a hierarchy of View and ViewGroup objects. A View usually draws something the user can see and interact with. Whereas a ViewGroup is an invisible container that defines the layout structure for View and other ViewGroup.

Output:

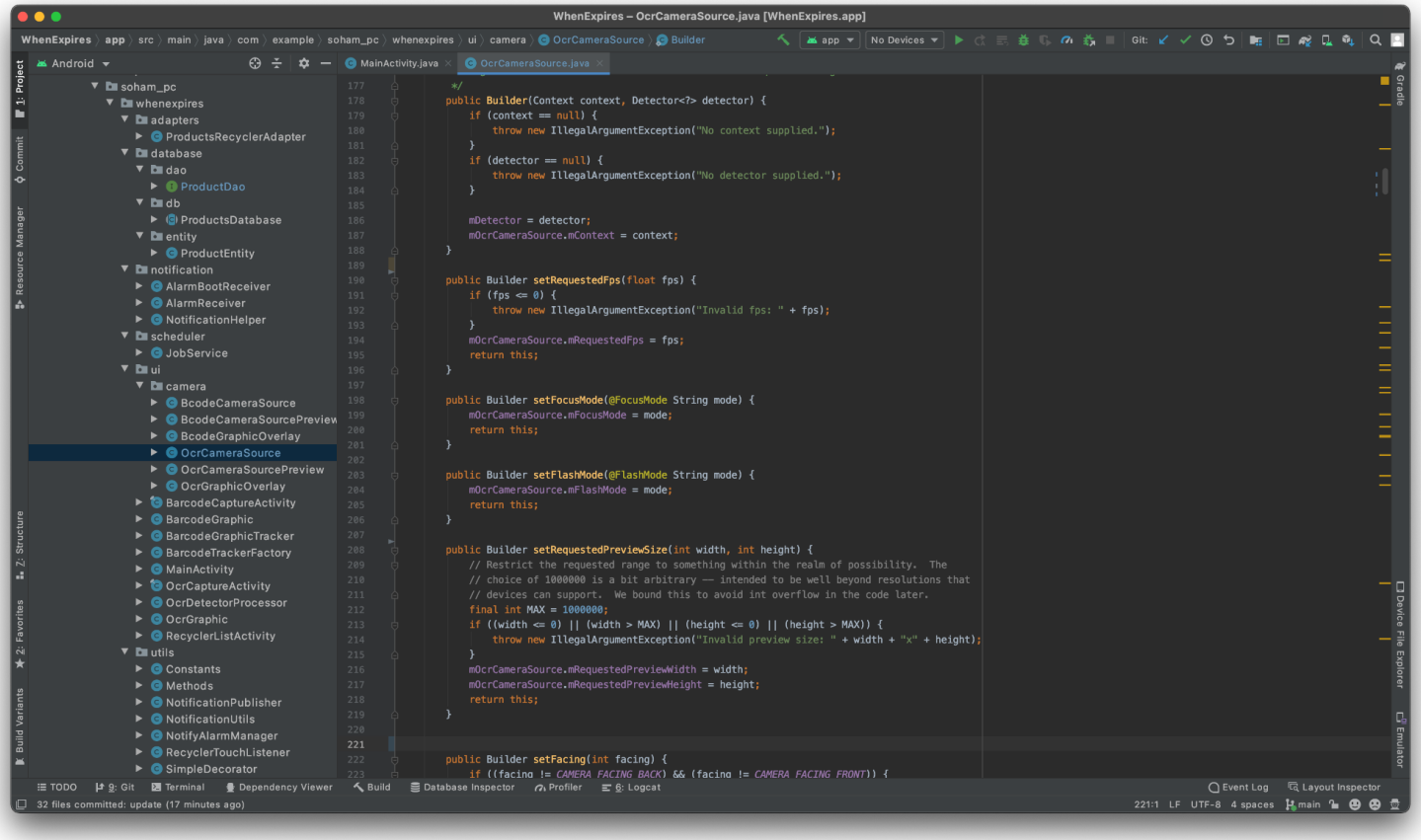


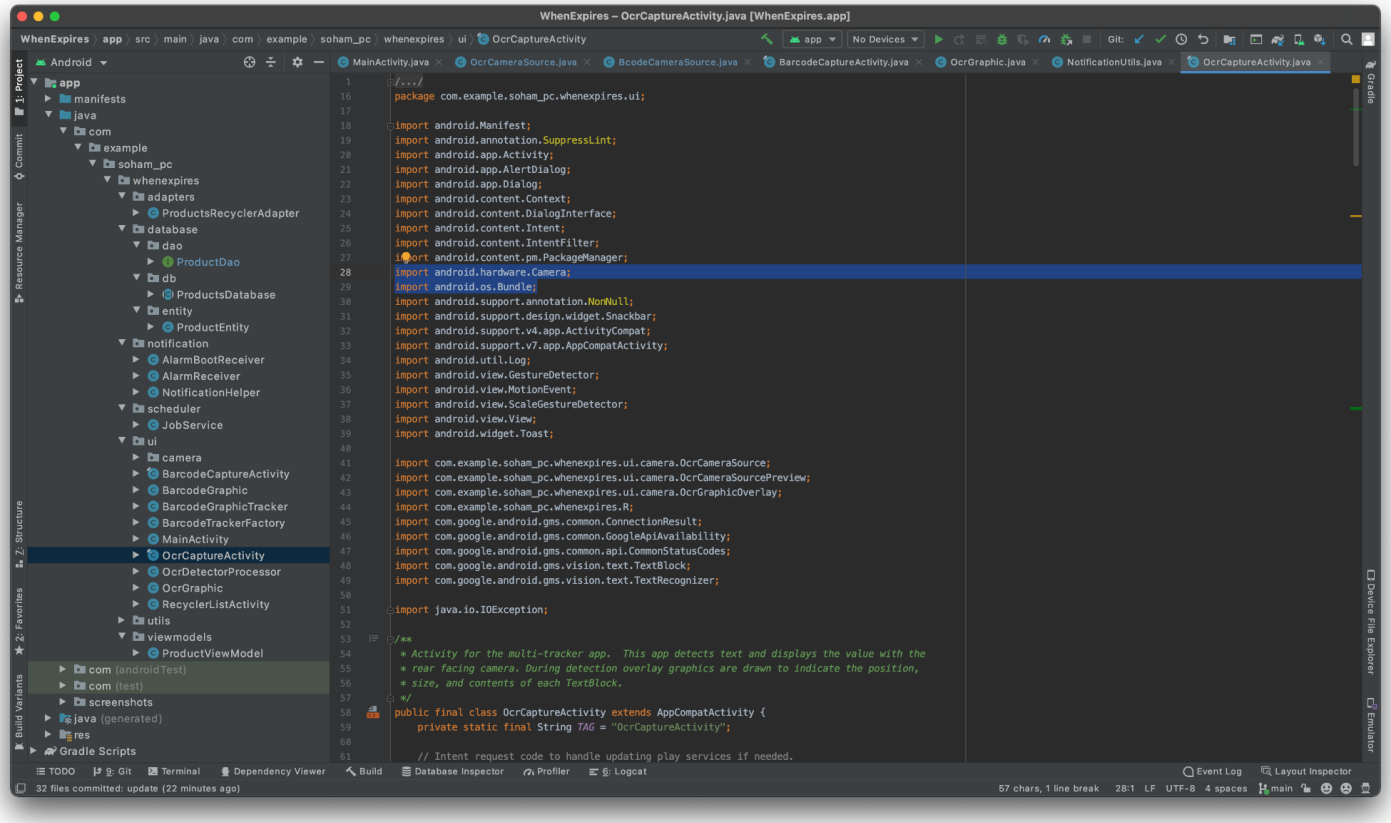
4. Implementation of Camera API

The Android framework includes support for various cameras and camera features available on devices, allowing you to capture pictures and videos in your applications. Before enabling your application to use cameras on Android devices, you should consider a few questions about how your app intends to use this hardware feature.

- Camera Requirement - Is the use of a camera so important to your application that you do not want your application installed on a device that does not have a camera? If so, you should declare the camera requirement in your manifest.
- Quick Picture or Customized Camera - How will your application use the camera? Are you just interested in snapping a quick picture or video clip, or will your application provide a new way to use cameras? For getting a quick snap or clip, consider Using Existing Camera Apps. For developing a customized camera feature, check out the Building a Camera App section.
- Foreground Services Requirement - When does your app interact with the camera? On Android 9 (API level 28) and later, apps running in the background cannot access the camera. Therefore, you should use the camera either when your app is in the foreground or as part of a foreground service.
- Storage - Are the images or videos your application generates intended to be only visible to your application or shared so that other applications such as Gallery or other media and social apps can use them? Do you want the pictures and videos to be available even if your application is uninstalled?

Output:

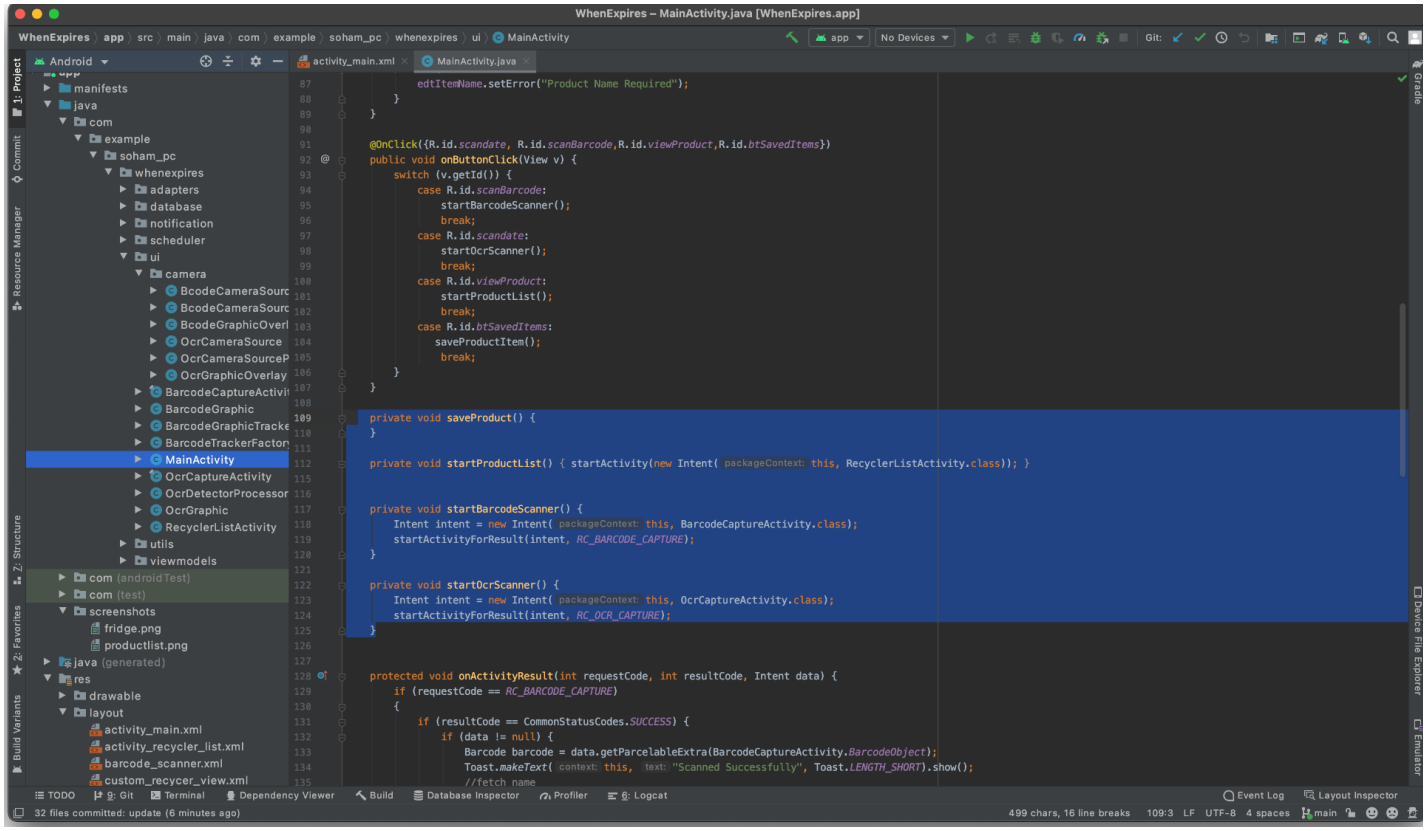

WhenExpires – OcrCameraSource.java [WhenExpires.app]
MainActivity.java
OcrCameraSource.java
soham_pc
whenexpires
adapters
ProductsRecyclerAdapter
database
dao
ProductDao
db
ProductsDatabase
entity
ProductEntity
notification
AlarmBootReceiver
AlarmReceiver
NotificationHelper
scheduler
JobService
ui
camera
BcodeCameraSource
BcodeCameraSourcePreview
BcodeGraphicOverlay
OcrCameraSource
OcrCameraSourcePreview
OcrGraphicOverlay
BarcodeCaptureActivity
BarcodeGraphic
BarcodeGraphicTracker
BarcodeTrackerFactory
MainActivity
OcrCaptureActivity
OcrDetectorProcessor
OcrGraphic
RecyclerListActivity
utils
Constants
Methods
NotificationPublisher
NotificationUtils
NotifyAlarmManager
RecyclerTouchListener
SimpleDecorator
public Builder(Context context, Detector<?> detector) {
if (context == null) {
throw new IllegalArgumentException("No context supplied.");
}
if (detector == null) {
throw new IllegalArgumentException("No detector supplied.");
}
mDetector = detector;
mOcrCameraSource.mContext = context;
}
public Builder setRequestedFps(float fps) {
if (fps <= 0) {
throw new IllegalArgumentException("Invalid fps: " + fps);
}
mOcrCameraSource.mRequestedFps = fps;
return this;
}
public Builder setFocusMode(@FocusMode String mode) {
mOcrCameraSource.mFocusMode = mode;
return this;
}
public Builder setFlashMode(@FlashMode String mode) {
mOcrCameraSource.mFlashMode = mode;
return this;
}
public Builder setRequestedPreviewSize(int width, int height) {
// Restrict the requested range to something within the realm of possibility. The
// choice of 1000000 is a bit arbitrary -- intended to be well beyond resolutions that
// devices can support. We bound this to avoid int overflow in the code later.
final int MAX = 1000000;
if ((width <= 0) || (width > MAX) || (height <= 0) || (height > MAX)) {
throw new IllegalArgumentException("Invalid preview size: " + width + "x" + height);
}
mOcrCameraSource.mRequestedPreviewWidth = width;
mOcrCameraSource.mRequestedPreviewHeight = height;
return this;
}
public Builder setFacing(int facing) {
if ((facing != CAMERA_FACING_BACK) && (facing != CAMERA_FACING_FRONT)) {
32 files committed: update (17 minutes ago)


WhenExpires – OcrCaptureActivity.java [WhenExpires.app]
WhenExpires › app › src › main › java › com › example › soham_pc › whenexpires › ui › OcrCaptureActivity
Android
app
com
example
soham_pc
whenexpires
adapters
ProductsRecyclerAdapter
database
dao
ProductDao
db
ProductsDatabase
entity
ProductEntity
notification
AlarmBootReceiver
AlarmReceiver
NotificationHelper
scheduler
JobService
ui
camera
BarcodeCaptureActivity
BarcodeGraphic
BarcodeGraphicTracker
BarcodeTrackerFactory
MainActivity
OcrCaptureActivity
OcrDetectorProcessor
OcrGraphic
RecyclerListActivity
utils
viewmodels
ProductViewModel
com (androidTest)
com (test)
screenshots
java (generated)
res
Gradle Scripts
MainActivity.java
OcrCameraSource.java
BcodeCameraSource.java
BarcodeCaptureActivity.java
OcrGraphic.java
NotificationUtils.java
OcrCaptureActivity.java
/.../
package com.example.soham_pc.whenexpires.ui;
import android.Manifest;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.pm.PackageManager;
import android.hardware.Camera;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.design.widget.Snackbar;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.GestureDetector;
import android.view.MotionEvent;
import android.view.ScaleGestureDetector;
import android.view.View;
import android.widget.Toast;
import com.example.soham_pc.whenexpires.ui.camera.OcrCameraSource;
import com.example.soham_pc.whenexpires.ui.camera.OcrCameraSourcePreview;
import com.example.soham_pc.whenexpires.ui.camera.OcrGraphicOverlay;
import com.example.soham_pc.whenexpires.R;
import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.GoogleApiAvailability;
import com.google.android.gms.common.api.CommonStatusCodes;
import com.google.android.gms.vision.text.TextBlock;
import com.google.android.gms.vision.text.TextRecognizer;
import java.io.IOException;
/**
* Activity for the multi-tracker app. This app detects text and displays the value with the
* rear facing camera. During detection overlay graphics are drawn to indicate the position,
* size, and contents of each TextBlock.
*/
public final class OcrCaptureActivity extends AppCompatActivity {
private static final String TAG = "OcrCaptureActivity";
// Intent request code to handle updating play services if needed.
TODO
9: Git
Terminal
Dependency Viewer
Build
Database Inspector
Profiler
6: Logcat
Event Log
Layout Inspector
32 files committed: update (22 minutes ago)
57 chars, 1 line break
28:1
LF
UTF-8
4 spaces
main

5. Implementation of Intents:

An Intent object carries information that the Android system uses to determine which component to start (such as the exact component name or component category that should receive the intent), plus information that the recipient component uses in order to properly perform the action.

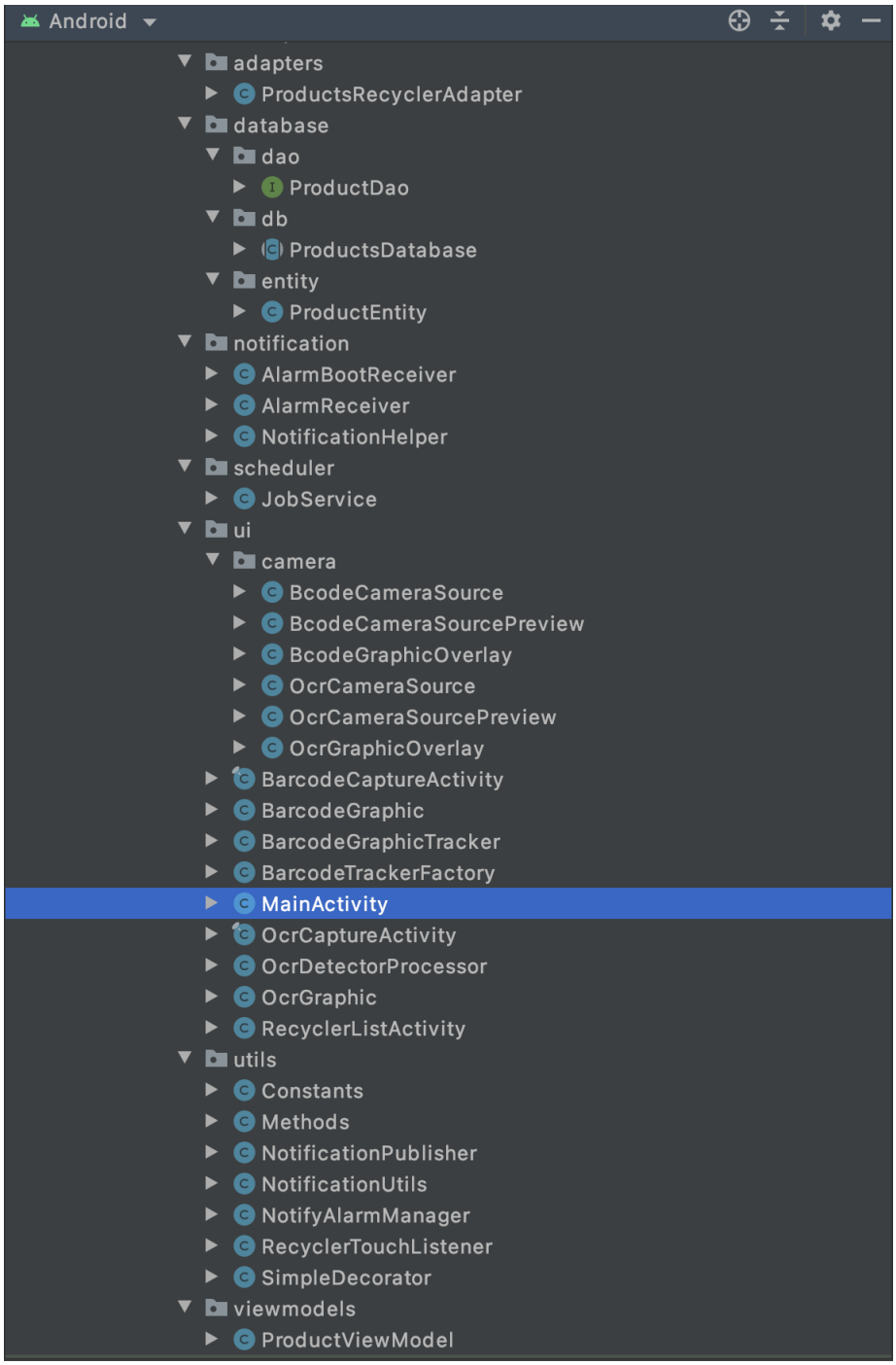
Output:



6. Implementation of Activity

Theory: The Activity class is a crucial component of an Android app, and the way activities are launched and put together is a fundamental part of the platform's application model. Unlike programming paradigms in which apps are launched with a main() method, the Android system initiates code in an Activity instance by invoking specific callback methods that correspond to specific stages of its lifecycle.

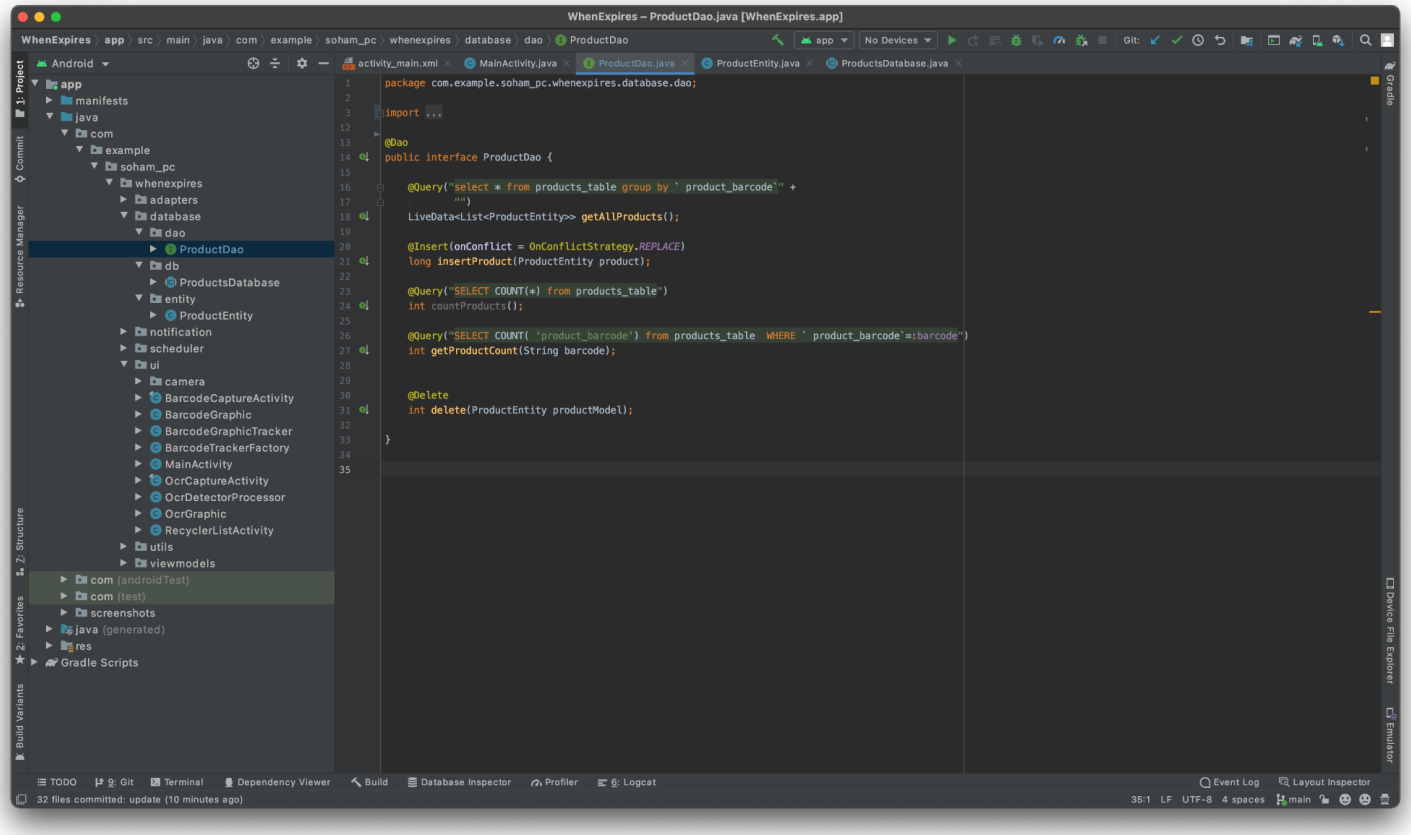
Output:

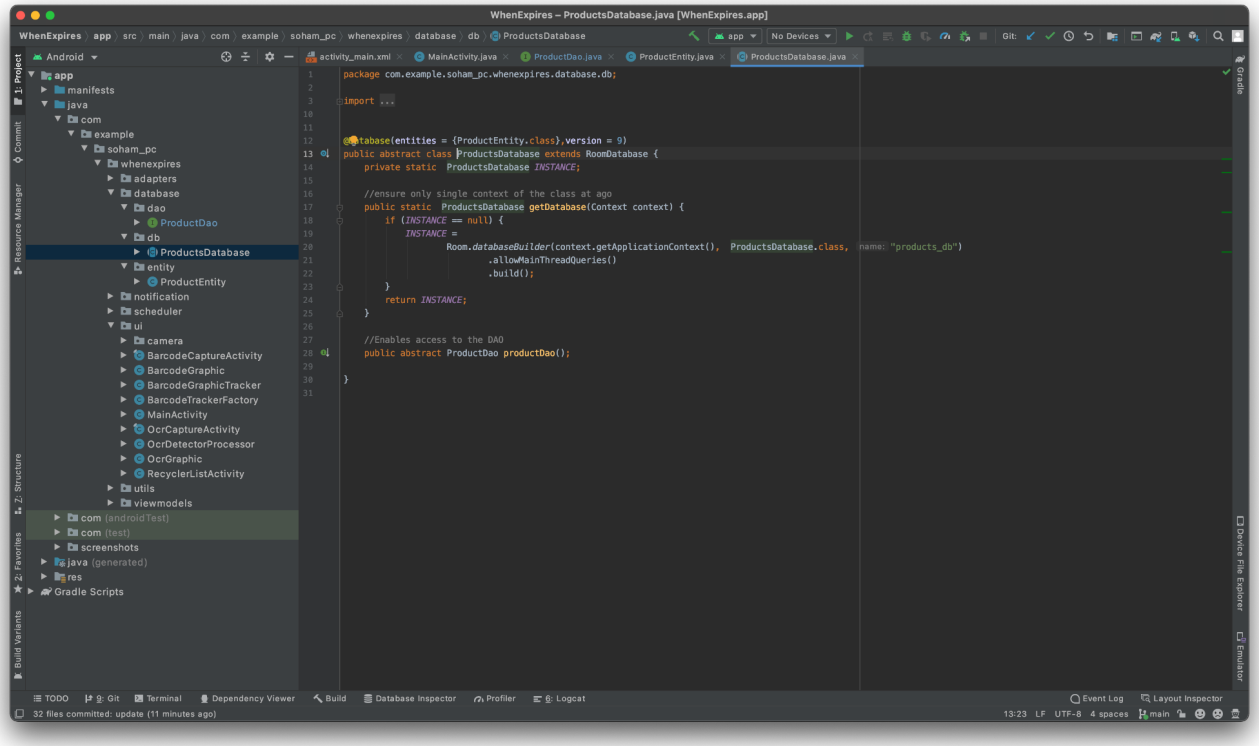

Android
adapters
ProductsRecyclerAdapter
database
dao
ProductDao
db
ProductsDatabase
entity
ProductEntity
notification
AlarmBootReceiver
AlarmReceiver
NotificationHelper
scheduler
JobService
ui
camera
BcodeCameraSource
BcodeCameraSourcePreview
BcodeGraphicOverlay
OcrCameraSource
OcrCameraSourcePreview
OcrGraphicOverlay
BarcodeCaptureActivity
BarcodeGraphic
BarcodeGraphicTracker
BarcodeTrackerFactory
MainActivity
OcrCaptureActivity
OcrDetectorProcessor
OcrGraphic
RecyclerListActivity
utils
Constants
Methods
NotificationPublisher
NotificationUtils
NotifyAlarmManager
RecyclerTouchListener
SimpleDecorator
viewmodels
ProductViewModel

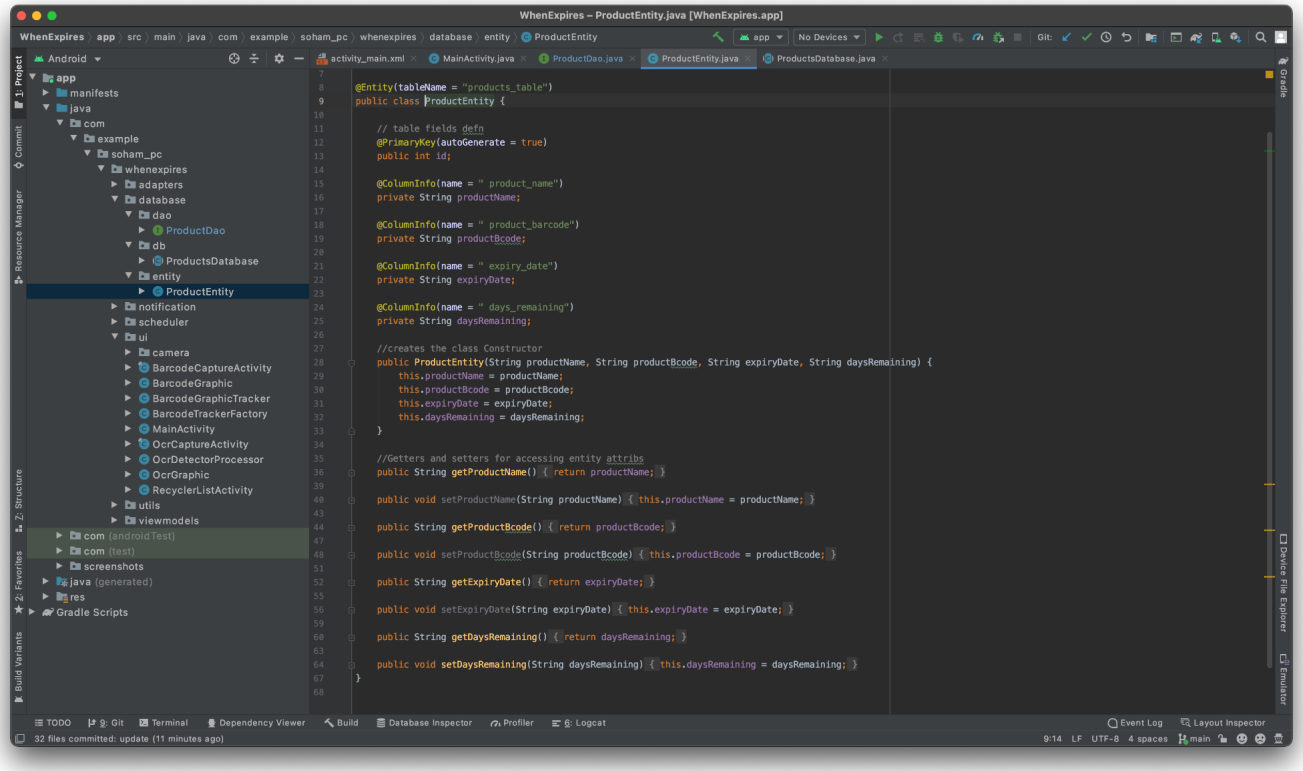
7. Implementation of SQLite Theory:

SQLite is a opensource SQL database that stores data to a text file on a device. Android comes in with built in SQLite database implementation. SQLite supports all the relational database features. In order to access this database, you don't need to establish any kind of connections for it like JDBC,ODBC e.t.c.

Output:




WhenExpires – ProductsDatabase.java [WhenExpires.app]
package com.example.soham_pc.whenexpires.database.db;
import ...
@Database(entities = {ProductEntity.class},version = 9)
public abstract class ProductsDatabase extends RoomDatabase {
private static ProductsDatabase INSTANCE;
//ensure only single context of the class at ago
public static ProductsDatabase getDatabase(Context context) {
if (INSTANCE == null) {
INSTANCE =
Room.databaseBuilder(context.getApplicationContext(), ProductsDatabase.class, "products_db")
.allowMainThreadQueries()
.build();
}
return INSTANCE;
}
//Enables access to the DAO
public abstract ProductDao productDao();
}

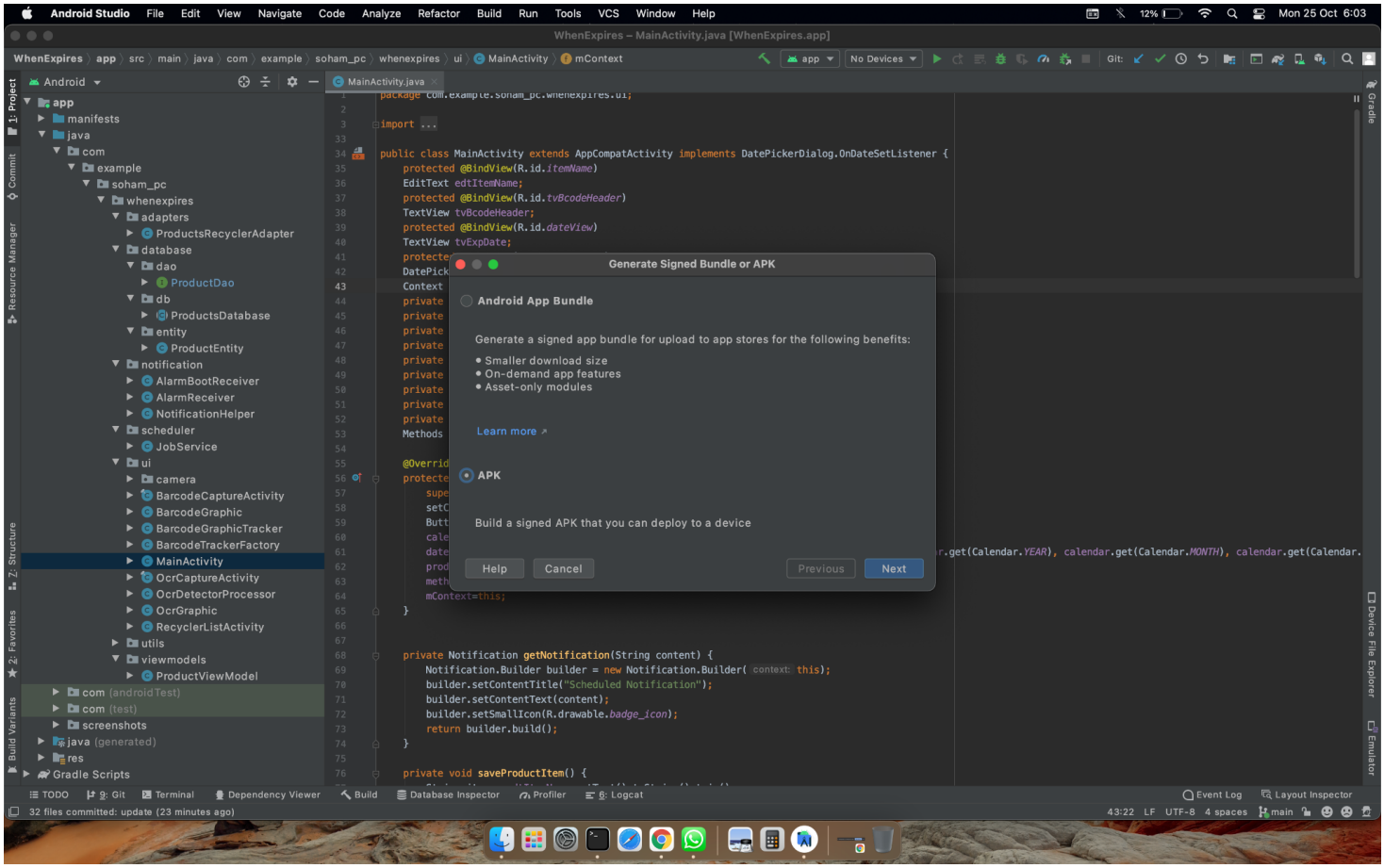

WhenExpires – ProductEntity.java [WhenExpires.app]
@Entity(tableName = "products_table")
public class ProductEntity {
// table fields defn
@PrimaryKey(autoGenerate = true)
public int id;
@ColumnInfo(name = " product_name")
private String productName;
@ColumnInfo(name = " product_barcode")
private String productBcode;
@ColumnInfo(name = " expiry_date")
private String expiryDate;
@ColumnInfo(name = " days_remaining")
private String daysRemaining;
//creates the class Constructor
public ProductEntity(String productName, String productBcode, String expiryDate, String daysRemaining) {
this.productName = productName;
this.productBcode = productBcode;
this.expiryDate = expiryDate;
this.daysRemaining = daysRemaining;
}
//Getters and setters for accessing entity attribs
public String getProductName() { return productName; }
public void setProductName(String productName) { this.productName = productName; }
public String getProductBcode() { return productBcode; }
public void setProductBcode(String productBcode) { this.productBcode = productBcode; }
public String getExpiryDate() { return expiryDate; }
public void setExpiryDate(String expiryDate) { this.expiryDate = expiryDate; }
public String getDaysRemaining() { return daysRemaining; }
public void setDaysRemaining(String daysRemaining) { this.daysRemaining = daysRemaining; }
}

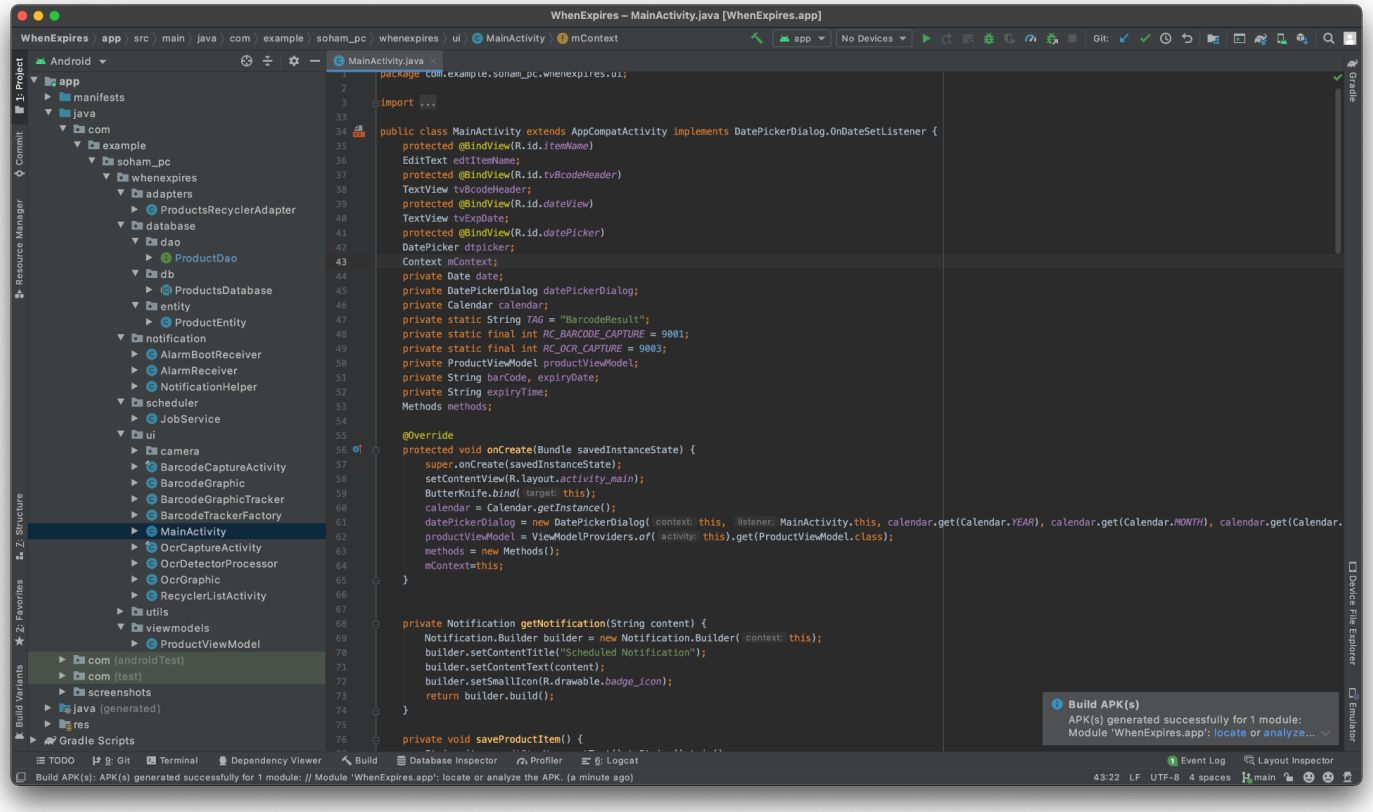
8. Generation of Application APK

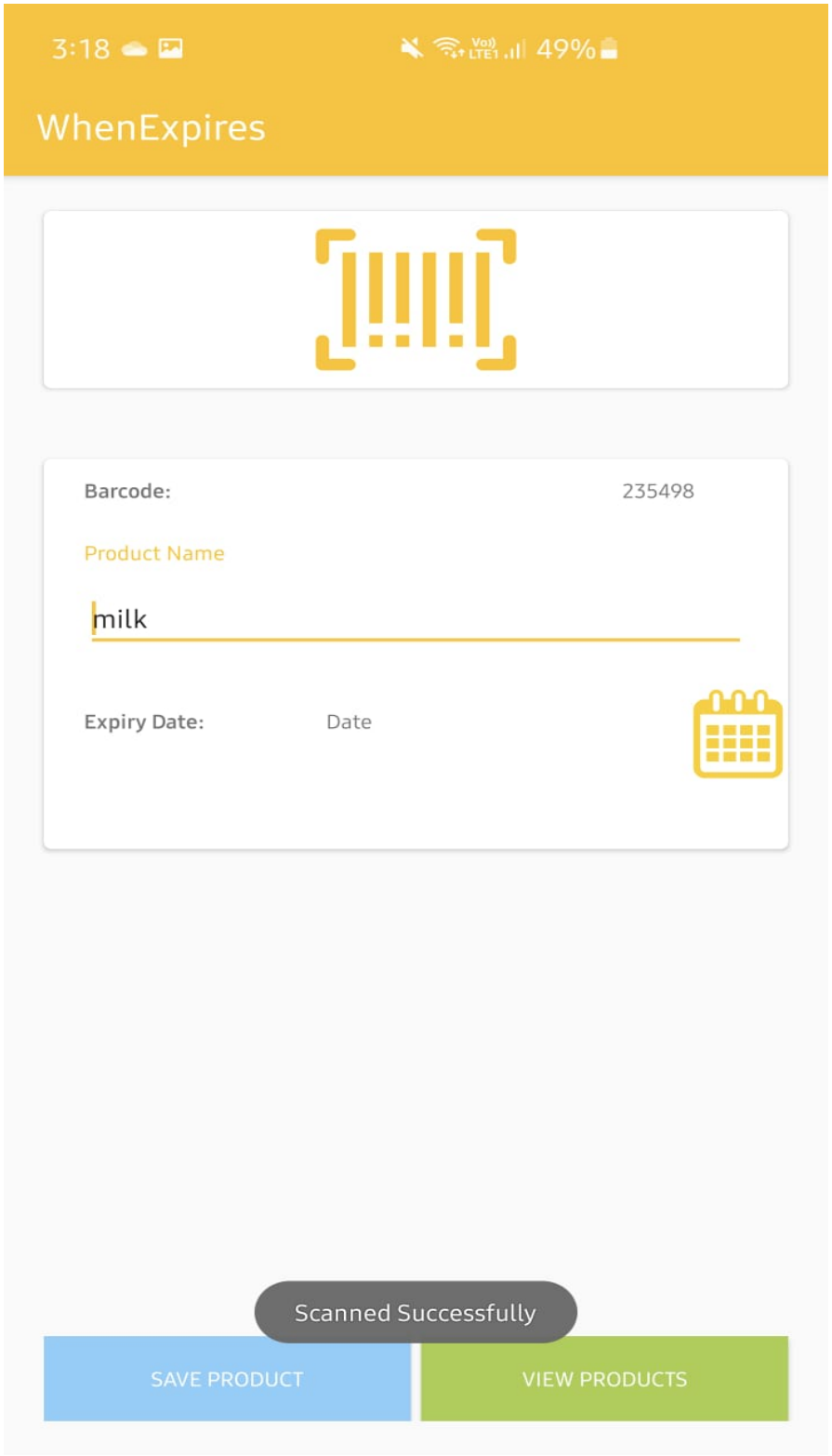
Theory: While developing an Android app, you would usually run it on a physical device or an emulator. If you want to share it with someone for their feedback, you would share an APK that can easily be installed on any Android device. How you can extract an APK file using Android Studio:

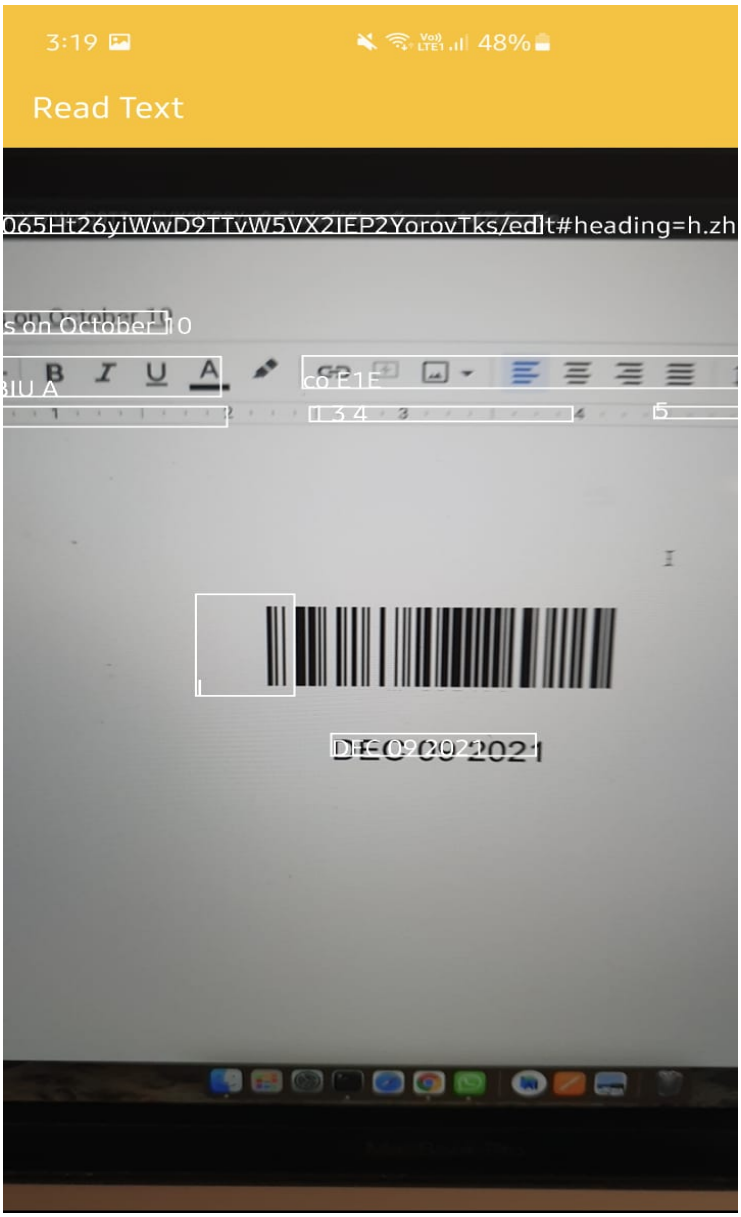
- In the Android menu, go to Build > Build Bundle(s) / APK (s) > Build APK(s).
- Android Studio will start building the APK for you. Once done, a pop-up on the bottom right will notify you of its completion. Click the ‘locate’ button in this dialog.
- The ‘locate’ button should open File Explorer with the debug folder open that contains a file called “app-debug.apk”.
- That’s it. Rename this file and share!
- This APK can only be used for testing. For deployment, you should generate a signed APK for which the process is a little more complicated.
- You need to enable the “Allow unknown sources” option in the settings of the device where you want to install the APK

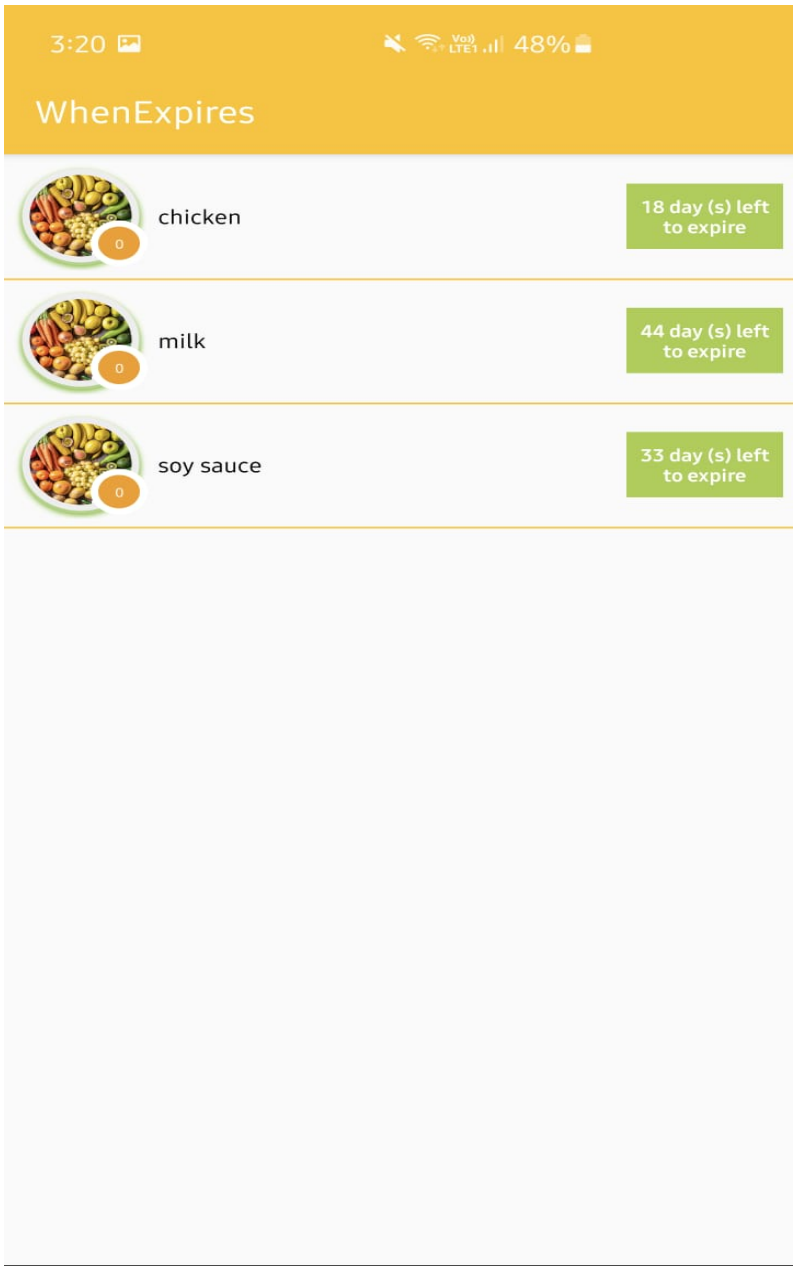
Output:


Generate Signed Bundle or APK
Android App Bundle
Generate a signed app bundle for upload to app stores for the following benefits:
• Smaller download size
• On-demand app features
• Asset-only modules
Learn more
APK
Build a signed APK that you can deploy to a device
Help
Cancel
Previous
Next


WhenExpires – MainActivity.java [WhenExpires.app]
MainActivity.java
Build APK(s)
APK(s) generated successfully for 1 module:
Module 'WhenExpires.app': locate or analyze...


3:18
49%
WhenExpires
Barcode:
235498
Product Name
milk
Expiry Date:
Date
Scanned Successfully
SAVE PRODUCT
VIEW PRODUCTS


3:19
48%
Read Text
065Ht26yiWwD9TTvW5VX2IEP2YorovTks/edit#heading=h.zh
s on October 10
BIU A
co E 1F
DEC 09 2021


3:20
48%
WhenExpires
chicken
18 day (s) left to expire
milk
44 day (s) left to expire
soy sauce
33 day (s) left to expire

9. Conclusion and Future Scope

In conclusion, the application is able to show a list of all items by color coding it as well as sorting it in order of high priority. The application is also able to automatically discover the name of the products using barcode scanning and tries to detect the date of expiry by OCR scanning but is unsuccessful in doing so in most attempts. It was also able to provide preference editing options which makes the application easy to customize. Overall, this application would definitely be very useful in preventing food wastage as well as saving money.