Gas Scan

Taylor Coomer, Tanner Thurman  
{taypcoom, wiltthur}@ut.utm.edu

***Abstract*— Gas Scan is an Android application that uses optical character recognition (OCR) to assist the user in fuel economy record-keeping. Records are stored in an online real-time database and used to calculate valuable statistics about fuel consumption to the user such as: miles per gallon, total miles driven, total price and more. The calculations can be adjusted and performed only on receipts from a given time period or on those from a certain gas station to give more specific results. Gas Scan is a useful tool that can help make users aware of their fuel consumption.**

***Keywords- mobile, Android, OCR, receipt, gasoline, fuel, economy, firebase***

# INTRODUCTION

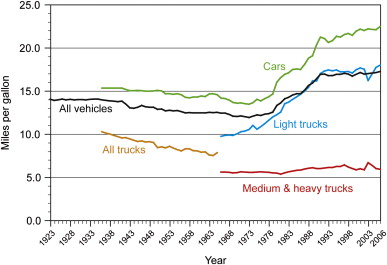
## Our goal with Gas Scan is to use new technology to create a useful application while learning along the way. We believe we have met both goals, especially since we have learned so much about OCR technologies as well as the usefulness of an online database such as Firebase.

The short-term goal of Gas Scan was to provide functionality in the form of record-keeping gasoline receipts. The receipts were stored, retreived, and calculated successfully upon. Gas Scan works well with the APIs we used, and we found ways around the short-comings of the APIs (which we will discuss later).

# BACKGROUND

The idea for this project was born out of the desire to work with mobile development, and we decided that an application that had real-world application to people who aren’t focused on computer science in their daily lives would be the way to go. So, we decided to make Gas Scan to help anyone who drives a vehicle keep track of their fuel economy. The user can compare their vehicle’s fuel economy measurements to our application, or they can use it as a substitute.

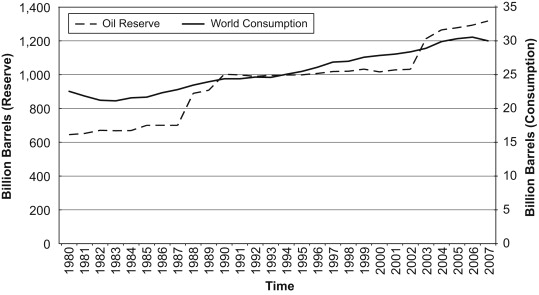
Fuel economy has been improving as the years have gone by: the graph below in figure 1 shows how America’s fuel economy as increased nearly linearly since the advent of commercially available vehicles in the early 1900’s [1]. Starting at an average of about 14 miles per gallon in 1923 and resulting in an average of about 17 miles per gallon; not much has seemingly changed in the 83 years that the graph displays. However, the forefront of the improvement would be the Cars and Light trucks, which have both increased by 6 and 8 miles per gallons respectively. The important thing to note about these graphs is the upward trend in fuel efficiency for most vehicles.



**Figure 1**

The upward trend began in the 1980s when fuel-injection and computer-aided engines became more prevalent [2]. These systems allowed for tracking of fuel-economy and a more effective use of gasoline in engines than the previously used carburetors. Even with the increase in fuel-economy and fuel-monitoring, vehicles still manage to be inefficient. An increase in only 6 to 8 miles per gallons over 83 years is not a lot; especially when our two most recent Presidents have requested an increase from the average 17 miles per gallon. George Bush’s administration passed the *Energy Independence and Security Act* which request an increase in average fuel economy to 27.5 mpg by 2020. Barack Obama’s administration recently passed a law as well which doubles the previous mandate, requesting an average fuel economy of 54.5 mpg by 2025 [3].

The reasoning behind each of these seemingly extreme requests is that our fuel supplies are dwindling at our current rate of consumption. Figure 2 contains a graph of the crude oil supplies juxtaposed with a graph of world consumption of oil supplies:

**Figure 2**

Through the timeline of the graph, the consumption is equal to or greater the oil reserve. As you can see at the very end, the consumption begins to decrease slightly at the 2007 year when the *Energy Independence and Security Act*  was enacted. With the rates shown, the known oil reserves would be depleted approximately by the year 2042 [4]. Each of these acts deleays the inevitable, giving us more time to find alternate energy solutions. Gas Scan is built to make sure that your vehicle is giving you the correct government mandated fuel-efficiency that it claims.

# DEVELOPMENT TOOLS

## Android Studio

The default, and most frequently used, IDE for Android development. Built off of IntelliJ, Android Studio contains all the bells and whistles that any IDE for Java development might have, plus more since it is made with Android development in mind. Most importantly for our devolpment cycle was the containment of Android emulators for any version of Android, as well as an SDK downloader and updater.

We used the emulators in place of phones for the majority of the development out of necessity. There (of course) were phones available for use, but the understandably strict requirements for use kept us from developing and testing on them continuously. We developed and tested regularly with the emulators, and very occasionally we would test for rare bugs on the phones - we never found anything specific to the hardware that the emulators did not reveal to us prior.

Android Studio has plenty of documentation and tutorials available ranging from beginner layout development to increasingly difficult examples for threading and serialization [5].

## Firebase

## Google’s newly aquired web and application framework which was originally built with just the realtime database in mind. We used the realtime database and the user authentication through Google accounts in Gas Scan. Firebase synchronizes on-the-fly and allowed for a quick set-up.

Just like Android, there is a wealth of material online for Firebase that is free that ranges from documenation to tutorials. We used Firebase mostly because it provided us with a *free* online database that could be easily connected with users that have a Google account to provide security and write protection [6]. Also, the query functionality for Firebase was simple enough to figure out and implement for our needs.

## Google Mobile Vision

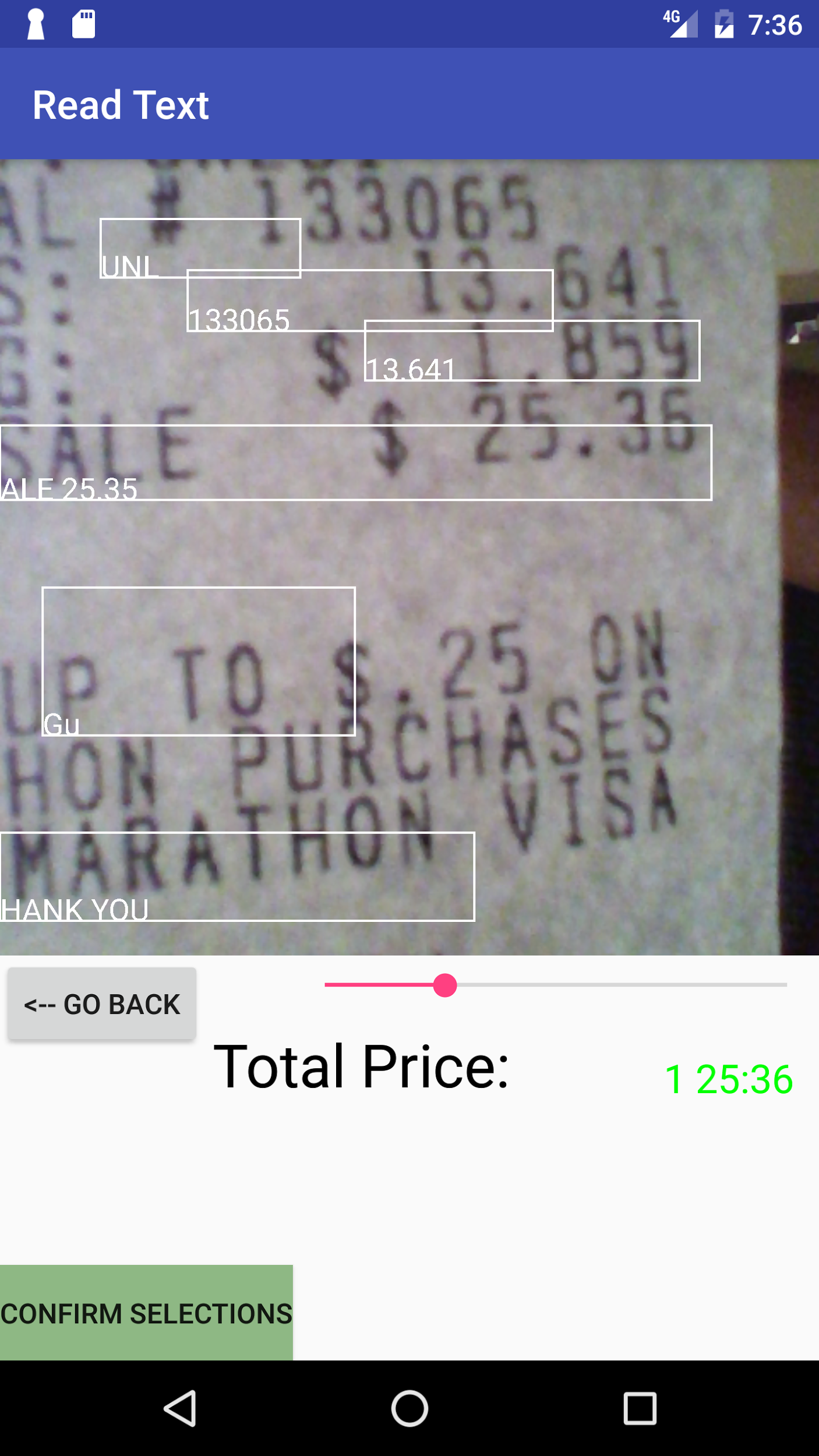
An OCR (optical character recognition) API built by Google which works well on mobile devices. Mobile Vision allows for instantaneous recognition of text in images through a camera feed at a low frame-rate. The error rate is higher this way, but we worked around the higher error rate by allowing the user to correct any mistakes at a later stage in data retrieval. This was the only short-coming in any API that caused us any problem, and the benefits outweighed the drawbacks. We felt that we found a good solution to the problem while maintaining a quick and intuitive application flow.

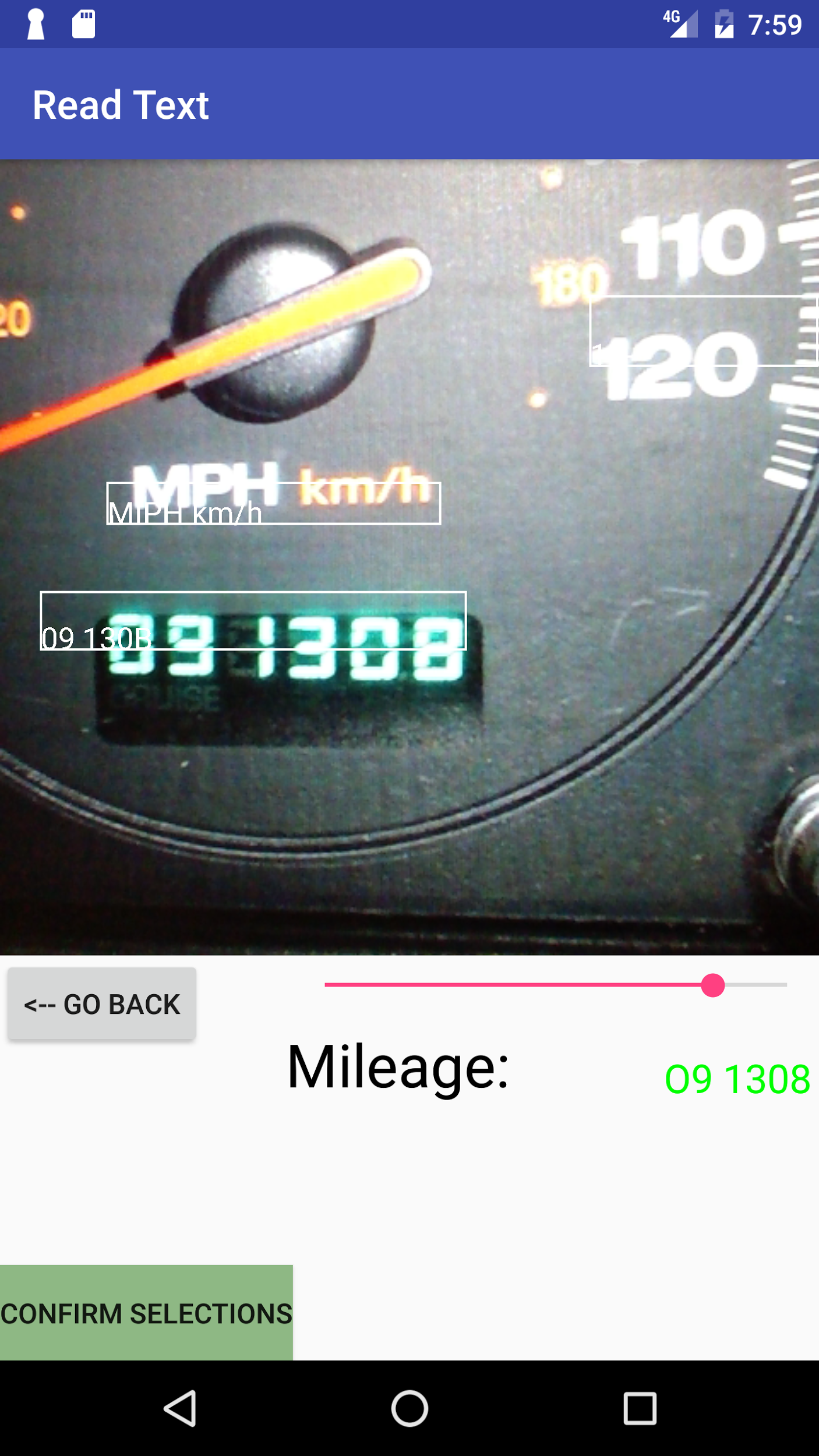
# APPLICATION FEATURES

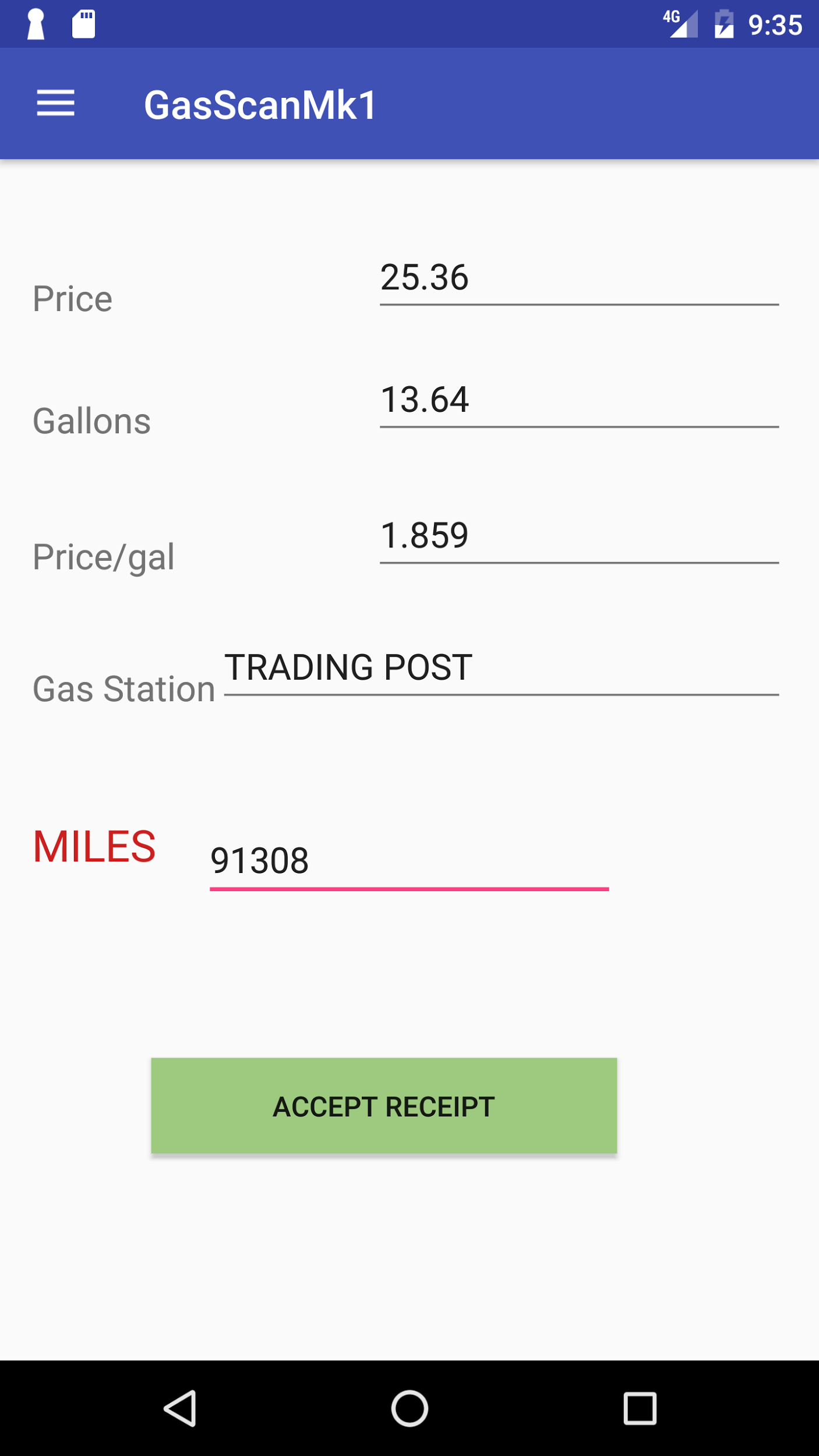
The app currently includes 3 key features.

## Receipt Scanning

Receipts (as well as gasoline pumps) can be scanned to retreive the information needed to calculate the fuel economy. Information needed is: gallons of purchase, price of purchase (and/or) price per gallon, and the total miles on the odometer. The information is scanned using the Mobile Vision API within the application, then the information is submitted to the Firebase database to be stored, safely. Figures 3, 4, and 5 show the process of creating a receipt entry with the app. First, figure 3 shows the scanning of a receipt in which the user selects the desired information contained in the white boxes. The boxes are generated using Mobile Vision. The same technique is done to obtain the odometer reading in Figure 4. Lastly, in Figure 5, the user gets a chance to correct any mistakes or errors that occured in the OCR reading before the data is sent to the Firebase database when the select “accept receipt”.

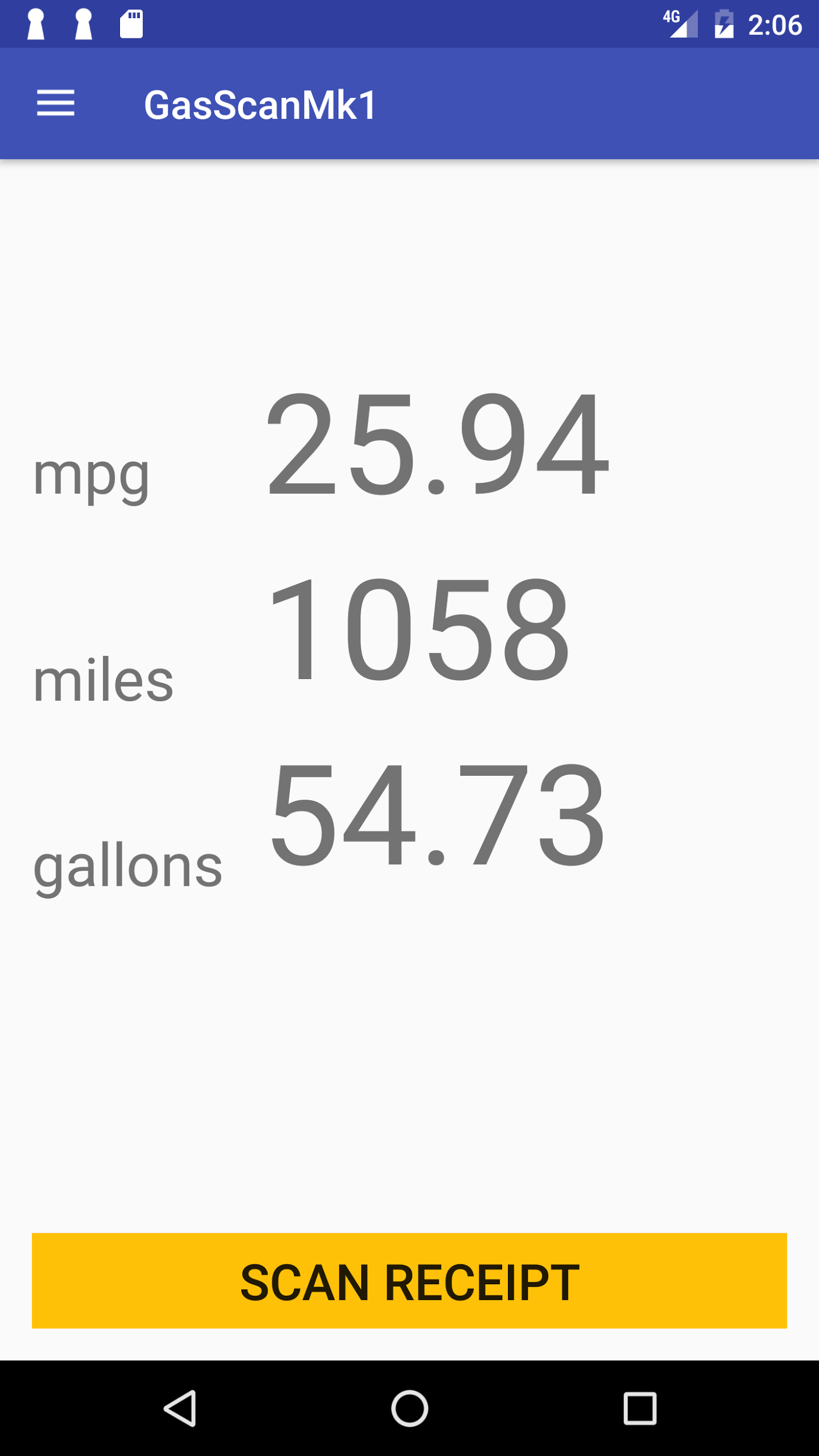
**Figure 3** 

**Figure 4**

**Figure 5**

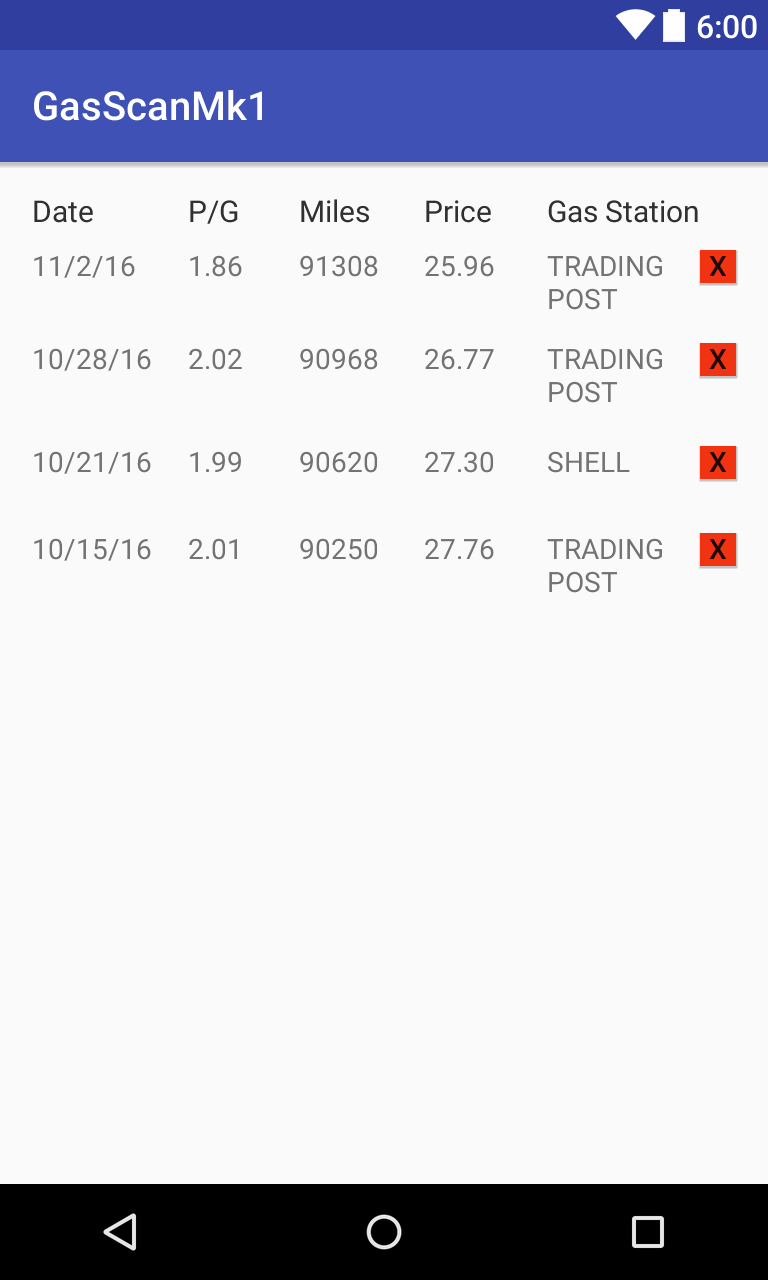
## Calculations

Information is retreived to give the user their totals and averages. These values are then displayed on the main menu, clearly. The information is retreived from each user’s Firebase table by a query that only retreives the most recent keys. The query is done to prevent re-calculating, and it reduces the number of reads and writes to the database - making the application more efficient. These values can allow the user to make educated decisions on their fuel consumption: *Are they spending to much? Should they be getting better gas mileage than they really are?* The list goes on. Figure 6 shows the dashboard values in the main menu that contain calculated fields for miles per gallon (mpg), total miles, and total gallons.

**Figure 6**

## Logs

The user can view and edit receipt entries within the logs menu. The sort-filter works by queries just like the intial calculations, but there is more than just a query by key. Users can query the receipts by date, price, gallons, or any attribute of the records. Figure 7 below shows the log menu with four receipt entries. These can be sorted as mentioned above, and each entry corresponds to an object inside the Firebase.

**Figure 7**

## SUMMARY

Gas Scan seems like just a simple application to track fuel economy targeted towards earth-conscious users who wish to get the most of their vehicles and their money. However, the value of the project was from our learning experience. One that exposed us to Android development, the difficulties of OCR programming and parsing, as well as the usefulness of having a real-time database.

# FUTURE DEVELOPMENT

If we continue to work on Gas Scan, we hope to 1.) fine tune our initial concept even further to a point of perfection, so there is no longer a need for the user to double-check receipt entries, and the OCR can become more automatic. This automation would be done through the work of parsing and templating receipts. 2.) Implement a form of community which would assist with the user’s ability to find cheaper gasoline near by. We would implement a new field for each receipt entry that would contain GPS coordinates, and these coordinates would be used to find the nearest gas-station. Once the gas station was found, we would update the price per gallon automatically on a public table that contains gas prices for all known gas stations for our users to compare either manually or through the help of our application.

##### References

1. M. Sivak and O. Tsimhoni, "Fuel efficiency of vehicles on US roads: 1923–2006," Energy Policy, vol. 37, no. 8, pp. 3168–3170, Aug. 2009.
2. M. Davis, "What you need to know about mechanical fuel injection - hot Rod Magazine," in Engine, Hot Rod Network, 2010. [Online]. Available: <http://www.hotrod.com/articles/hrdp-1010-what-you-need-to-know-about-mechanical-fuel-injection/>. Accessed: Nov. 30, 2016.
3. The White House, website, <https://www.whitehouse.gov/the-press-office/2012/08/28/obama-administration-finalizes-historic-545-mpg-fuel-efficiency-standard> (08/28/2012)
4. S. Shafiee and E. Topal, "When will fossil fuel reserves be diminished?," Energy Policy, vol. 37, no. 1, pp. 181–189, Jan. 2009.
5. Android Studio, website, <https://developer.android.com/training/basics/firstapp/creating-project.html> (10/24/2014)
6. Firebase, Documentation, website, <https://firebase.google.com/docs/reference/js/firebase.database.Query> (11/19/2016)