KSUSOY YIELDCALC: AN INNOVATIVE NATIVE ANDROID APP TO ESTIMATE SOYBEAN YIELD BEFORE HARVEST USING CONVENTIONAL APPROACH

by

TANIA BANDYOPADHYAY

B.Tech., West Bengal University of Technology, 2012

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2015

Approved by:

Major Professor
Dr. Daniel Andresen

Abstract

This report considers a native Android application called “KSUSoy YieldCalc” that assists in yield estimation of soybean before harvest following conventional approach. Android is one of the most popular installed base of any mobile platform, powering many mobile devices in more than 190 countries for users of diverse economic backgrounds, making it more popular than iOS devices (Android developers n.d.).The project “KSUSoy YieldCalc” adopted the Android platform as its base to serve farmers, agronomists, and consultants and deliver performance to save time and enhance farmers’ their confidence. The native application uses “conventional approach” of estimation of yield for calculations and eliminates the need for having Internet connection to access, thereby increasing the application’s flexibility. The project utilized Android Software Development Kit (SDK) as its development platform with extensive Java and Extensible Markup Language (XML) coding. The Department of Agronomy at Kansas State University (KSU) tested the application with promising results. Dr. Ignacio Ciampitti of the Department of Agronomy at KSU currently demonstrates the application to farmers. User feedback has been very satisfactory to date.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to acknowledge Dr. Daniel Andresen, Associate Professor, Department of Computing and Information Sciences, Kansas State University, for being the major advisor and assisting and guiding throughout the project. I would also like to thank Dr. Ignacio Ciampitti, Assistant Professor, Department of Agronomy, Kansas State University, for his auspices and guidance in the nuances of Agronomy. I would also like to thank my other committee members, Dr.Torben Amtoft and Dr.Mitch Neilsen, Department of CIS, KSU, for their courses and knowledge that helped build a strong base for analytical and real time programming.

Dedication

The project is dedicated to every agronomists, consultant, and farmer all over the world who have been always an innate part of any civilization, providing society with food and encouraging rural employment which significantly impacts economic stability. The world's food sources will be dwindled without farmers leading to poverty and starvation. I would also like to dedicate the project to my family and friends for their constant support and encouragement.

Preface

The project investigates a native android application called "KSUSoy YieldCalc" which was first suggested by Dr. Ignacio Ciampitti, Assistant Professor, Department of Agronomy, Kansas State University, and Dr. Daniel Andresen, Associate Professor, Department of Computing and Information Sciences, Kansas State University. The purpose of the application is to deliver an estimated yield of soybean before harvesting season. The project aims its use of estimating soybean yield before harvest for agronomists, farmers, and consultants.

Chapter 1 - Introduction

1.1. Explanation of Problem

This report considers use of a native android application (app) to address the problem of yield estimation before soybean harvest using conventional approach. The app could potentially save time allowing agronomists, farmers, and consultants to more effectively estimate yields. Yield estimation increases farmers' confidence and indicates necessary change to the factors in order to obtain a reasonable estimate. Determination of estimation involves four primary parameters: Plant Population, Seeds Per Pod, Seed Size, and Pods per Plant. Yield estimates become more accurate as the time for harvest approaches because the plants have already responded to the growing season. Yield estimation can commence as crops enter into their R5 state (first seed) when a majority of pods have developed seeds that fill the plant and flowering continues until termination (Figure 1). Yield potential can be low or high at this point depending on the 4-6 weeks of the growing season. Yield estimates typically improve as the plants grow over the course of ~15 days following initiation into the R5 state and then enter R6 (Figure 2) which lasts for another ~20 days. As plants reach their R6 state, pod retention and seeds per pod become more discernable, thereby improving yield estimates (Casteel 2012).

Figure 1. R5 State of soybean (first seed) (Casteel 2012)

Figure 2. R6 State of soybean (Casteel 2012)

Yield estimates can be calculated using the conventional approach or the simplified approach (Figure 3). In the conventional approach, soybean yield estimates are based on the following components:

- Total number of pods per acre [number of plants per acre x pods per plant] (1)
- Total number of seeds per pod (2)
- Number of seeds per pound (3)
- Total pounds per bushel, or test weight (60 lbs./bu for soybeans) (4)

The equation for the estimation of the potential soybean yield is

[(1) x (2) / (3)] / (4) = Soybean yield in bushels/acre

Figure 3. Calculations to estimate yield (Herbek 2005)

Location	Plants per Acre	x	Pods per Plant	x	Seeds per Pod	÷	Seeds per Pound	÷	Pound per Bushel	=	Bushels per Acre
1		x		x		/		/	60	=	
2		x		x		/		/	60	=	
3		x		x		/		/	60	=	
4		x		x		/		/	60	=	
5		x		x		/		/	60	=	
6		x		x		/		/	60	=	
7		x		x		/		/	60	=	
8		x		x		/		/	60	=	
9		x		x		/		/	60	=	
10		x		x		/		/	60	=	
Average										=	

The simplified approach utilizes a constant row length to represent 1/10,000th area of an acre instead of the number of plants per acre. In this approach, the farmers need to sample 21 inches of row length in a single row if the soybean plants are spaced in 30-inch rows; in two rows if row spacing is 15 inches, and in four rows if row spacing is 7.5 inches, as shown in Figure 4.

Figure 4. Simplified method (Ciampitti 2014)

The conventional approach was selected for this research in order to more effectively calculate yield estimates and ensure convenience of calculation for agronomists and consultants. The conventional approach was chosen because the simplified approach considers only 21 in. of the final plant population for estimation instead of a larger area. In addition, the simplified approach considers seed weight only as a modified (dimensionless) coefficient instead of seeds per pound, creating interpretation difficulties for calculation users.

1.2. Reasons for Ineffective Solutions

The problem has not previously been resolved due to the inability to obtain this information of estimated yield in a conventional electronic format. Typically, farmers manually calculate yield estimates on paper, thereby requiring extensive amounts of time and allowing opportunity for human error. With the boom of smartphones, however, many farmers have begun using android devices (Gerke n.d.), negating the common misconception that farmers do not utilize modern technology. According to Zach Hunnicutt, a fifth-generation Nebraska farmer, technology "is a pretty indispensable part of our operation and we're not unique in the farming world in that aspect." Float Mobile Learning, a consulting firm that develops mobile strategies and apps for agricultural organizations and Fortune 500 companies, has found that 94% of farmers own a smartphone or a mobile phone. In mid 2011, android devices made news when an app that could gather data and deliver it to the salesforce benefited poor Ugandan farmers (Christopher Doering, Gannett Washington Bureau n.d.). As one of the most popular technologies available in wide range of prices, Android can deliver benefits to farmers irrespective of economic condition. Certain web-based apps are targeted for android devices, but they utilize the simple approach that requires farmers to conduct sampling in order to account for natural field variability. In addition, farmers must have Internet connection in the fields in order to access the websites. This application eradicates the need for an Internet connection after installation, ensuring consistent performance independent of the Internet or cellular connectivity.

1.3. Effective Solution

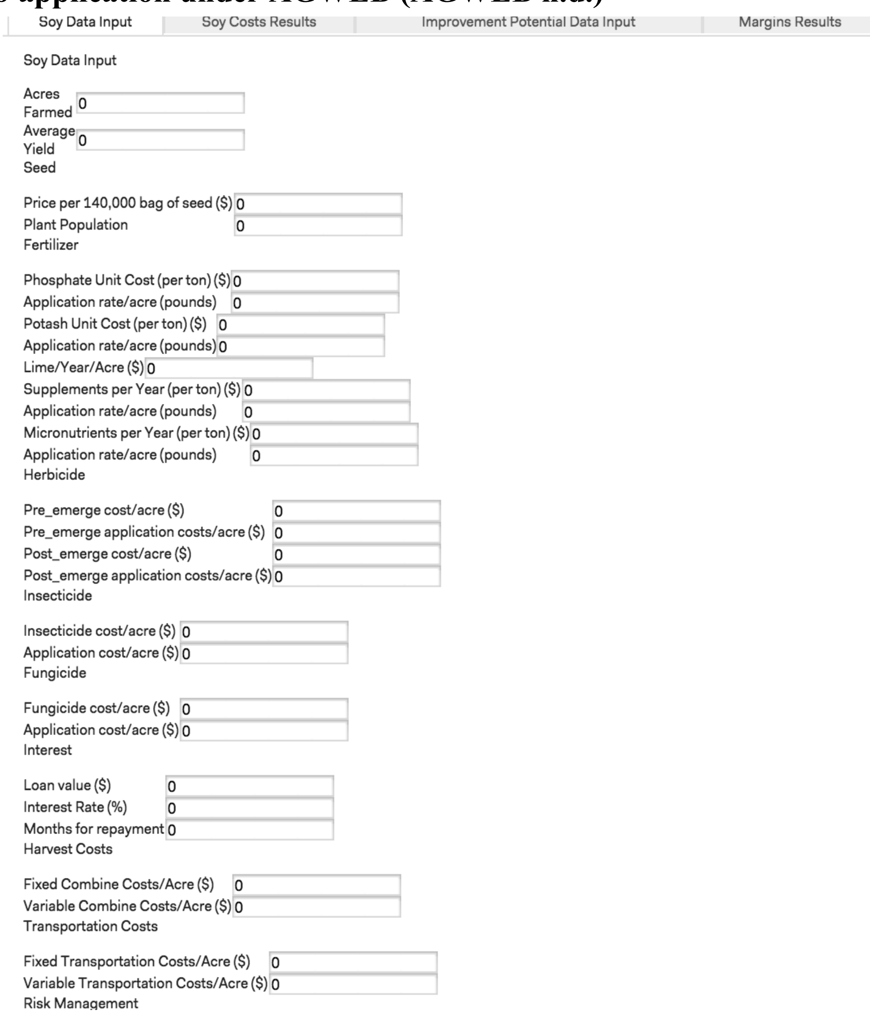
This application is effective because it decreases the likelihood of human error and allows acquisition of yield estimate within a few seconds without needing paper or Internet connection. Calculation automation allows farmers to reduce the number of man-hours they have to dedicate to crunching numbers, consequently reducing the total cost and the probability of loss. Obtaining yield estimates much before harvest enables farmers to plan accordingly and accurately estimate potential values of the parameters. Web applications for mobile devices require Internet connection at the fields. Diminished data speed also negatively affects the performance and time required to display results. Calculation on an Excel spreadsheet requires prior training and knowledge about the application. However, the native app in this study does not require Internet connection or training, thereby eliminating the need for a data plan, reducing cost and time, and improving convenience.

Chapter 2 - Related Work

Farmers and agronomists have begun using technologies to make advancements in food processing, such as web applications that are beginning to assist in soybean production.

One web application, under AGWEB (Figure 5) is designed to calculate the total cost of soybean production using yield as input. Farmers and consultants can use this app to calculate estimated yield and predict total cost of production.

Figure 5. Web application under AGWEB (AGWEB n.d.)


Soy Data Input | Soy Costs Results | Improvement Potential Data Input | Margins Results

Soy Data Input

Acres Farmed 0
Average Yield 0
Seed

Price per 140,000 bag of seed ($) 0
Plant Population 0
Fertilizer

Phosphate Unit Cost (per ton) ($) 0
Application rate/acre (pounds) 0
Potash Unit Cost (per ton) ($) 0
Application rate/acre (pounds) 0
Lime/Year/Acre ($) 0
Supplements per Year (per ton) ($) 0
Application rate/acre (pounds) 0
Micronutrients per Year (per ton) ($) 0
Application rate/acre (pounds) 0
Herbicide

Pre_emerge cost/acre ($) 0
Pre_emerge application costs/acre ($) 0
Post_emerge cost/acre ($) 0
Post_emerge application costs/acre ($) 0
Insecticide

Insecticide cost/acre ($) 0
Application cost/acre ($) 0
Fungicide

Fungicide cost/acre ($) 0
Application cost/acre ($) 0
Interest

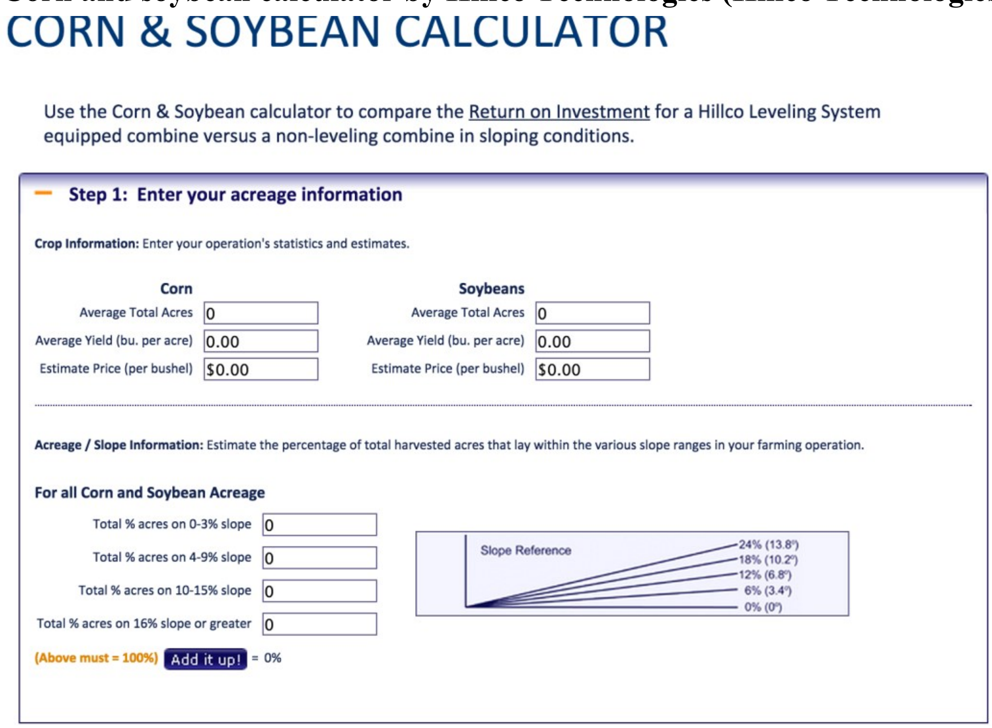
Loan value ($) 0
Interest Rate (%) 0
Months for repayment 0
Harvest Costs

Fixed Combine Costs/Acre ($) 0
Variable Combine Costs/Acre ($) 0
Transportation Costs

Fixed Transportation Costs/Acre ($) 0
Variable Transportation Costs/Acre ($) 0
Risk Management

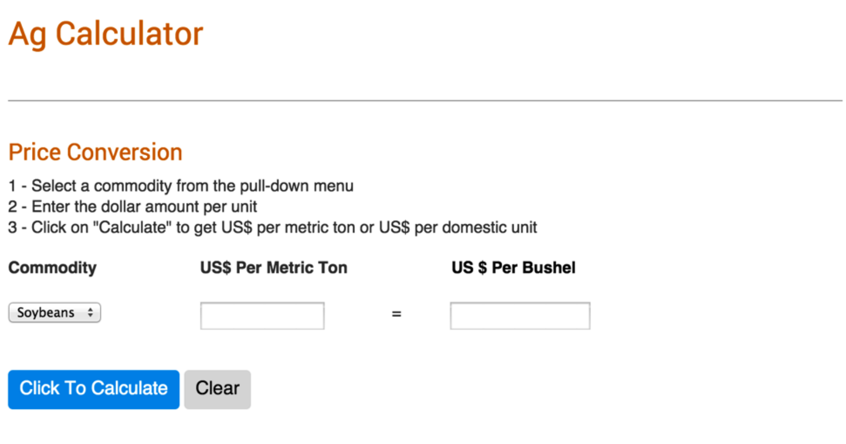
Another app, a product of Hillco Technologies (Figure 6) is used to calculate return on investment. This app also requires yield as input, allowing farmers to calculate yield estimate yield in order to estimate returns.

Figure 6. Corn and soybean calculator by Hillco Technologies (Hillco Technologies n.d.)



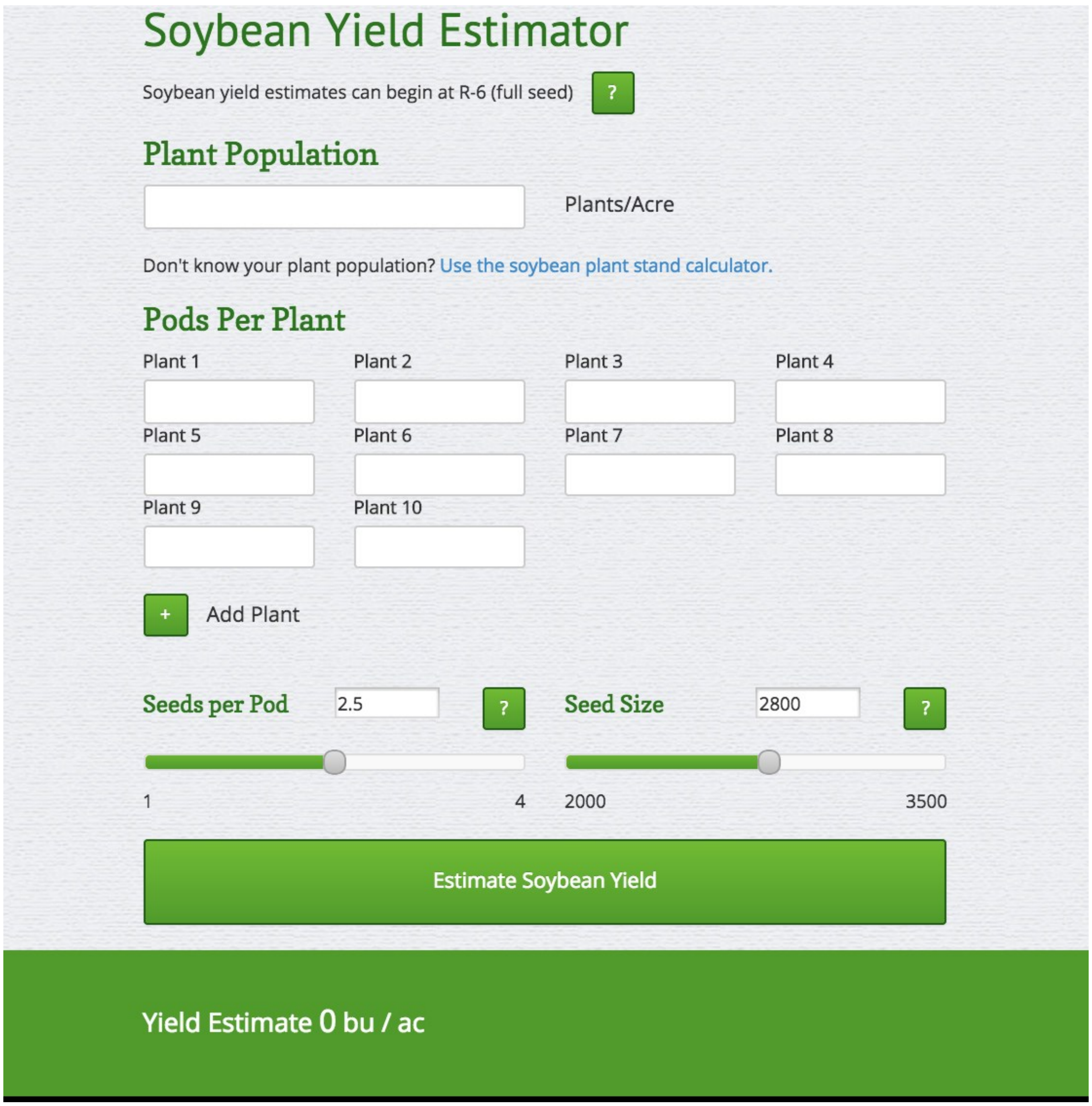
The CME Group created an application (Figure 7) to calculate United States Dollars (USD) per metric or USD per domestic yield. This app can be used for trading purposes.

Figure 7. Ag calculator by CME Group (CME Group n.d.)



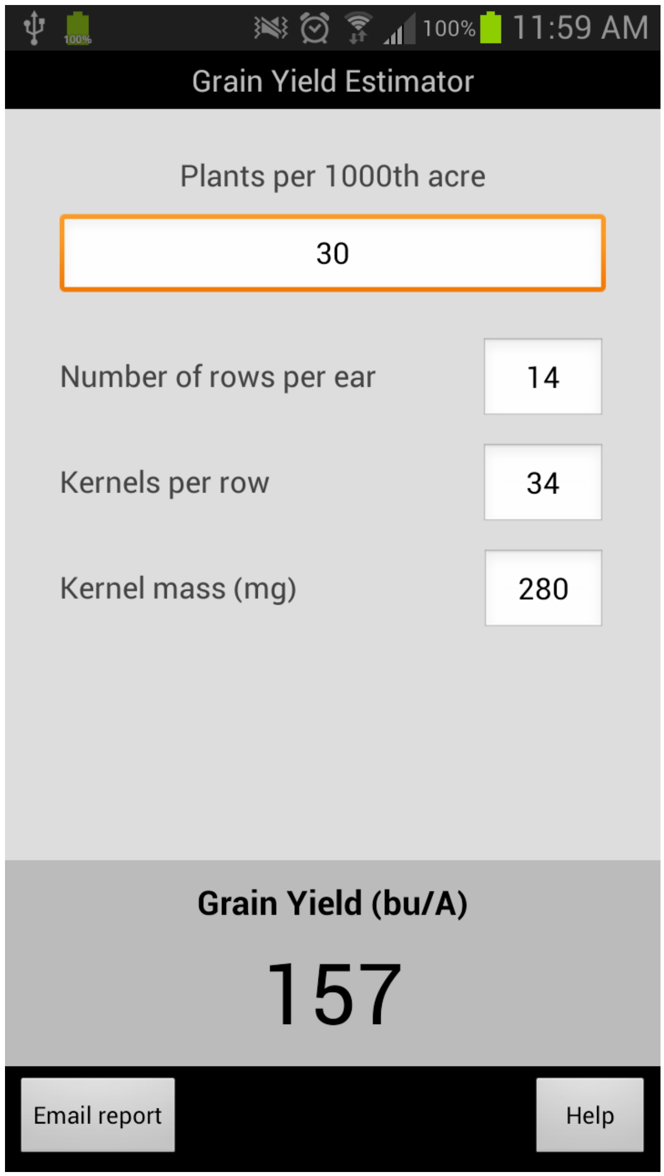
The KSUSoy YieldCalc application was primarily inspired from an application (Figure 8) developed by Manitoba Pulse Growers Association (MPGA) that provides a very sophisticated approach to calculate estimated yield. Therefore, the native KSUSoy YieldCalc android app was built with a sophisticated interface while accounting for lack of Internet connection on farms and eliminating the need for cellular data or browsing balance after app installation.

Figure 8. Soybean Yield Estimator by MPGA (Association 2013)



Initiatives have also begun for building mobile apps to estimate corn yield. The University of Wisconsin has developed many apps within its “Corn Calculator” project, including the “Silag Moisture Cost Adjuster” app, the “Maturity Date Predictor” app, and the “Grain Yield Estimator” (Figure 9).

Figure 9. Grain Yield Estimator app by the University of Wisconsin (Crop Calculator| Integrated Pest and Crop Mangement n.d.)

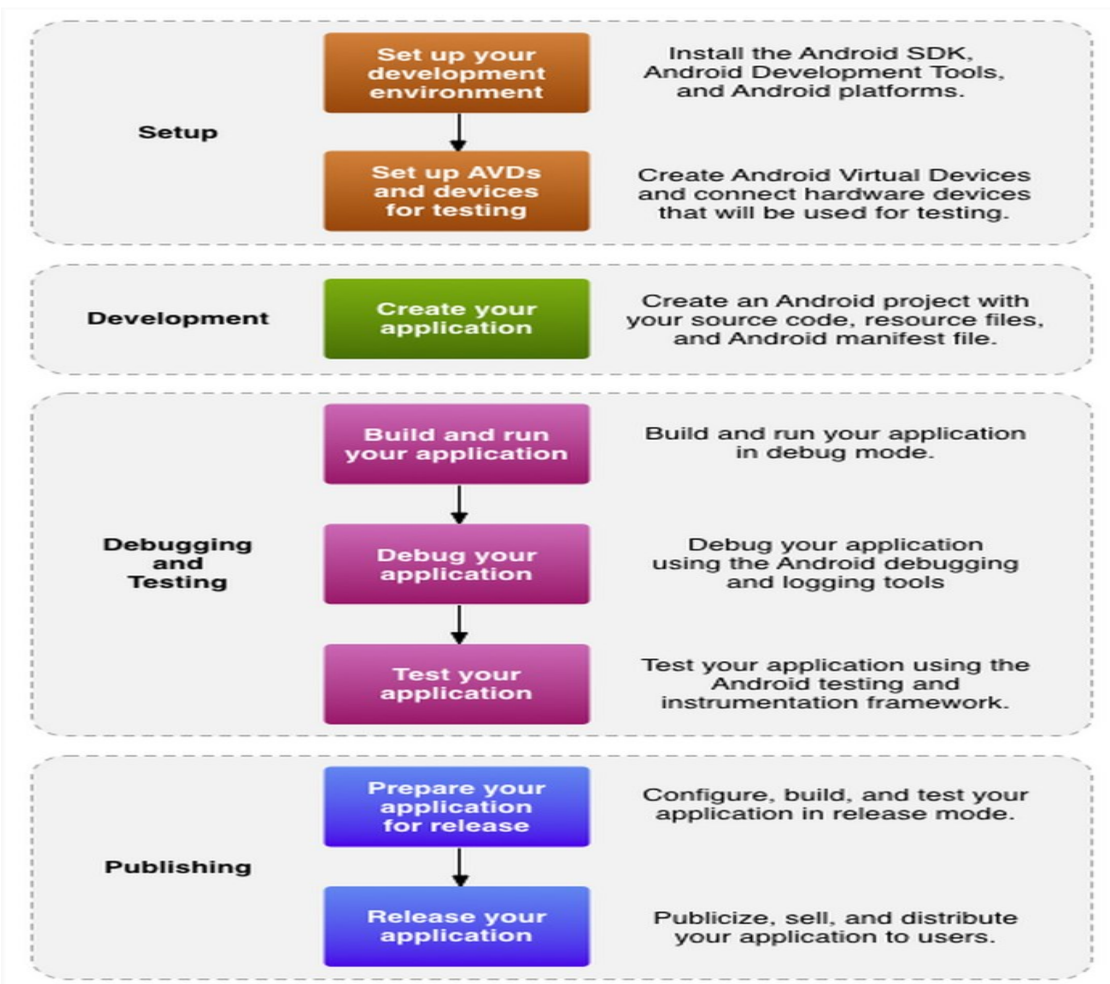


Chapter 3 - Implementation

In this study, the KSUSoy YieldCalc app was implemented using Android Studio, the official integrated development environment (IDE) for development on the Android platform that primarily uses JAVA and XML based on IntelliJ IDEA. The Android build system is the toolkit used to build, test, run, and package the apps. It can run as an integrated tool from the Android Studio menu and it can run independently from the command line. Figure 10 represents a graphical workflow for Android applications.

3.1. Android application workflow

Figure 10. Android workflow (developer.android.com, Developer Workflow 2014)



The following development phases are required to build an Android app.

3.1.1. Environment Setup

The environment setup phase includes software installation, environmental setup, creation of Android Virtual Devices (AVDs), and hardware device connections to install and test the app in the developer's mode. An AVD is an emulator configuration that simulates an actual device by specifying hardware and software options to be emulated by the Android. It can be launched from Eclipse by clicking Window > AVD Manager, or it can be started from command line by calling the android tool with AVD options from the <sdk>/tools/ directory.

3.1.2. Project Setup and Development

The project setup and development phase includes set up and development of the Android Studio project and application modules (Figure 11) that contain all of source code and resource files for an application. An Android project should contain everything that defines the Android app, including app source code to build configurations and test code. SDK tools required this project to follow a specific structure in order to accurately compile and package the KSUSoy YieldCalc application.

Figure 11. Android project files (developer.android.com, Managing Projects Overview 2014)

Android Project Files

Android Studio project files and settings provide project-wide settings that apply across all modules in the project.

`.idea`
Directory for IntelliJ IDEA settings.

`app`
Application module directories and files.

`build`
This directory stories the build output for all project modules.

`gradle`
Contains the gradler-wrapper files.

`.gitignore`
Specifies the untracked files that Git should ignore.

`build.gradle`
Customizable properties for the build system. You can edit this file to specify the default build settings used by the application modules and also set the location of your keystore and key alias so that the build tools can sign your application when building in release mode. This file is integral to the project, so maintain it in a source revision control system.

`gradle.properties`
Project-wide Gradle settings.

`gradlew`
Gradle startup script for Unix.

`gradlew.bat`
Gradle startup script for Windows.

`local.properties`
Customizable computer-specific properties for the build system, such as the path to the SDK installation. Because the content of the file is specific to the local installation of the SDK, the `local.properties` should not be maintained in a source revision control system.

`.iml`
Module file created by the IntelliJ IDEA to store module information.

`settings.gradle`
Specifies the sub-projects to build.

A module is the first level of containment within a project that encapsulates specific source codes and resources depending on the type. Android application modules (Figure 12) are eventually built into .apk files that contain source and resource files.

Figure 12. Android modules (developer.android.com, Managing Projects Overview 2014)

`build/`
: Contains build folders for the specified build variants. Stored in the main application module.

`libs/`
: Contains private libraries. Stored in the main application module.

`src/`
: Contains your stub Activity file, which is stored at `src/main/java//ActivityName>.java`. All other source code files (such as `.java` or `.aidl` files) go here as well.

`androidTest/`
: Contains the instrumentation tests. For more information, see the Android Test documentation.

`main/java/com.>project<.>app<`
: Contains Java code source for the app activities.

`main/jni/`
: Contains native code using the Java Native Interface (JNI). For more information, see the Android NDK documentation.

`main/gen/`
: Contains the Java files generated by Android Studio, such as your `R.java` file and interfaces created from AIDL files.

`main/assets/`
: This is empty. You can use it to store raw asset files. Files that you save here are compiled into an `.apk` file as-is, and the original filename is preserved. You can navigate this directory in the same way as a typical file system using URIs and read files as a stream of bytes using the `AssetManager`. For example, this is a good location for textures and game data.

`main/res/`
: Contains application resources, such as drawable files, layout files, and string values in the following directories. See Application Resources for more information.

  `anim/`
  : For XML files that are compiled into animation objects. See the Animation resource type.

  `color/`
  : For XML files that describe colors. See the Color Values resource type.

  `drawable/`
  : For bitmap files (PNG, JPEG, or GIF), 9-Patch image files, and XML files that describe Drawable shapes or Drawable objects that contain multiple states (normal, pressed, or focused). See the Drawable resource type.

  `layout/`
  : XML files that are compiled into screen layouts (or part of a screen). See the Layout resource type.

  `menu/`
  : For XML files that define application menus. See the Menus resource type.

  `raw/`
  : For arbitrary raw asset files. Saving asset files here is essentially the same as saving them in the `assets/` directory. The only difference is how you access them. These files are processed by aapt and must be referenced from the application using a resource identifier in the `R` class. For example, this is a good place for media, such as MP3 or Ogg files.

  `values/`
  : For XML files that define resources by XML element type. Unlike other resources in the `res/` directory, resources written to XML files in this folder are not referenced by the file name. Instead, the XML element type controls how the resources defined within the XML files are placed into the `R` class.

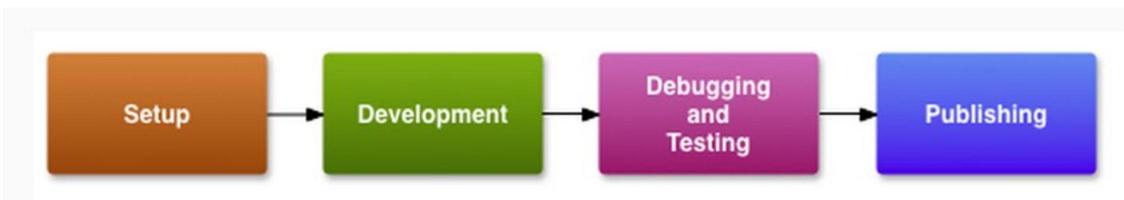
  `xml/`
  : For miscellaneous XML files that configure application components. For example, an XML file that defines a `PreferenceScreen`, `AppWidgetProviderInfo`, or Searchability Metadata. See Application Resources for more information about configuring these application components.

3.1.3. Building, Debugging, and Testing

In the building, debugging, and testing phase, the KSUSoy YieldCalc app was built into a debuggable .apk package(s) able to be installed and run on the emulator or a hardware device connected to a machine. Android uses a build system based on Gradle that provides flexibility, customized build variants, and dependency resolution. The project was then debugged using an Android Debug Monitor and device log messages (logact) along with IntelliJ IDEA coding features. The JDWP-compliant debugger and debugging and logging tools can also be used.

3.1.4. Publishing

Figure 13. Publishing workflow (developer.android.com, publishing 2014)



The publishing phase (Figure 13) involves configuration and building of an application for release and distribution to users. Publication is comprised of two main steps:

1. Preparing the application for release
   During this step, a release version for the project or final application product was built for users to download and install on their Android-powered devices.
2. Release the application to users
   This step involves marketing, selling, and distributing the release version of an application to users.

3.2. Unified Modelling Language (UML) diagram

Figure 14. UML diagram for KSUSoy YieldCalc

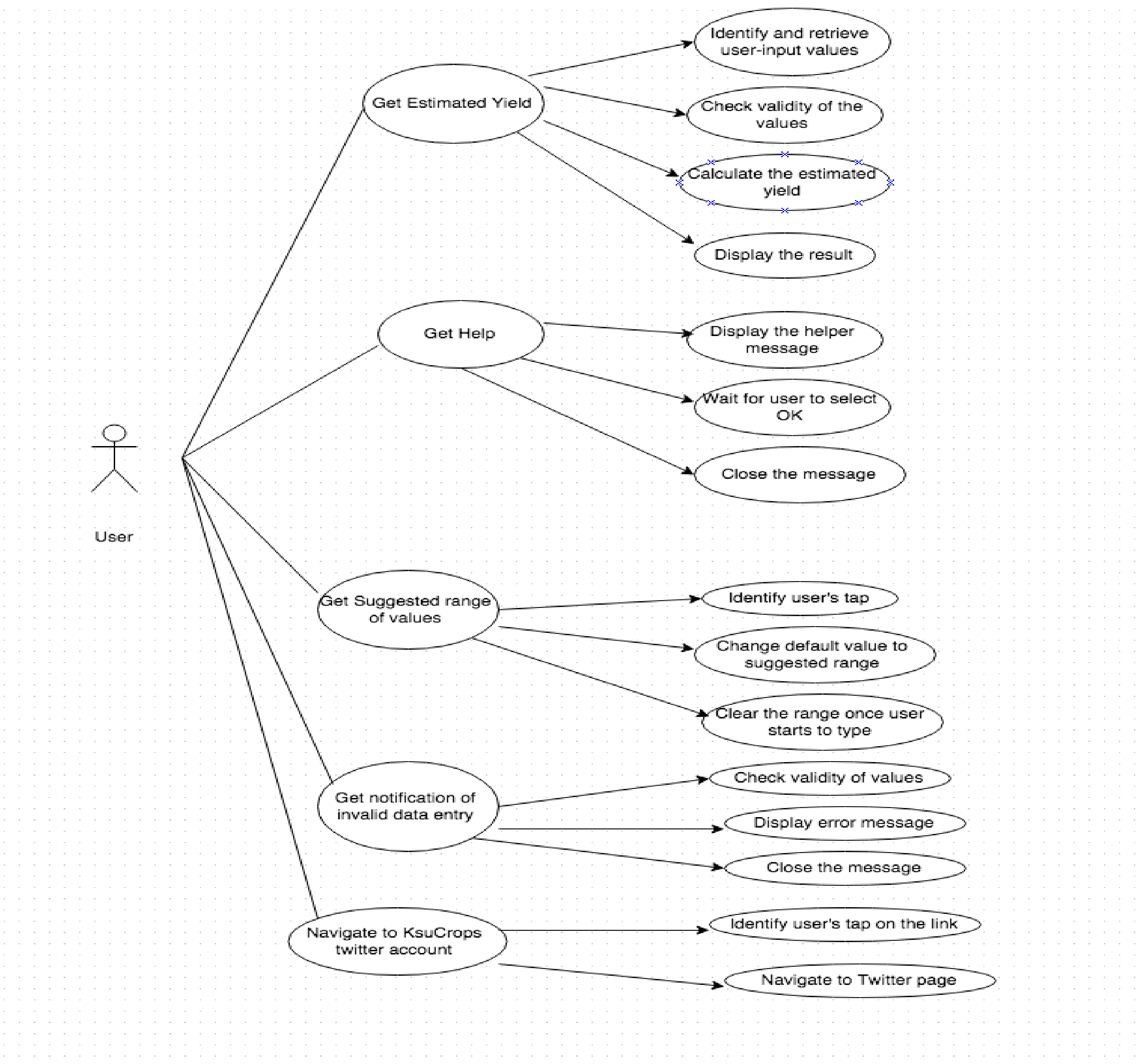


Figure 14 illustrates a UML diagram for KSUSoy YieldCalc. Use of the application allows users to obtain the following functionalities:

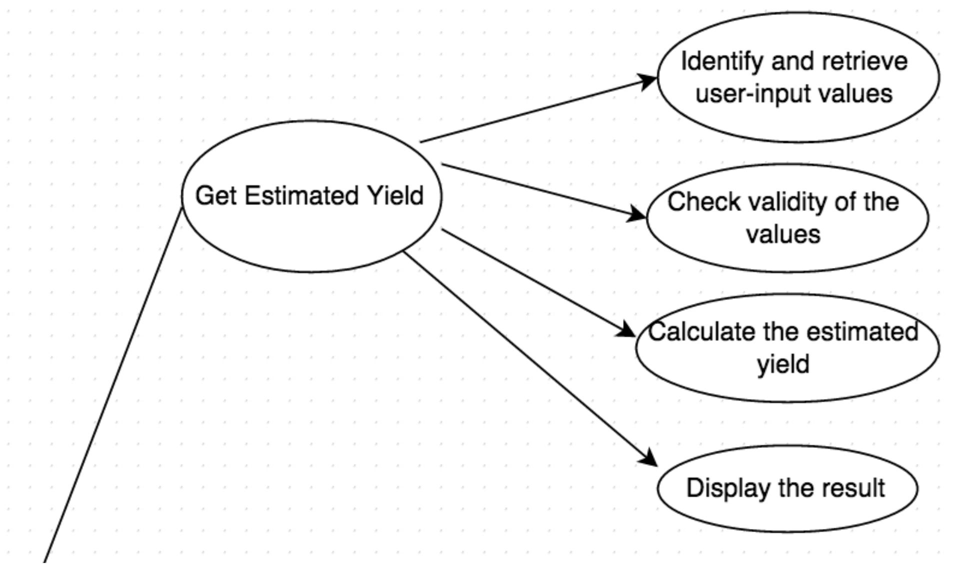
1. Get Estimated Yield

2. Get Help
3. Get Suggested Range of Values
4. Get Notification of Invalid Data Entry
5. Navigate to KSUCrops Twitter Account

These are the basic functionalities provided to the user in Step 1.

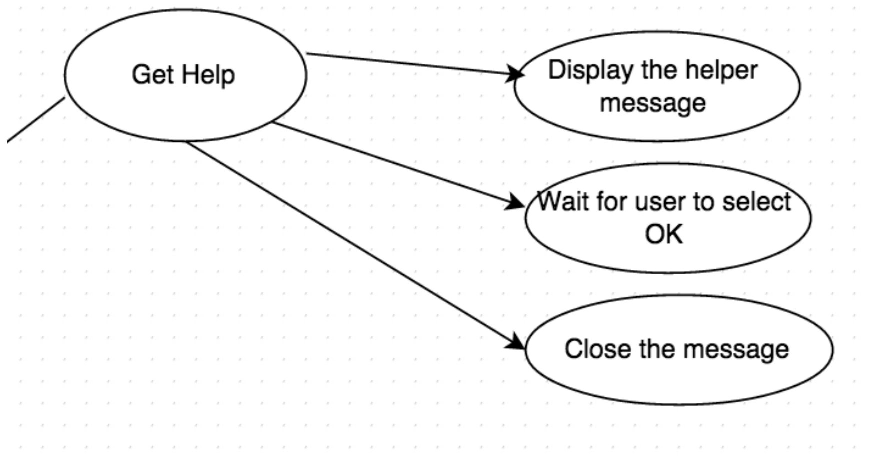
Each functionality was expanded and analyzed for the next step. The functionality of "Get Estimated Yield" (Figure 15) should identify and retrieve user-input values, check value validity, calculate estimated yield, and display results with units.

Figure 15. Analysis of "Get Estimated Yield"



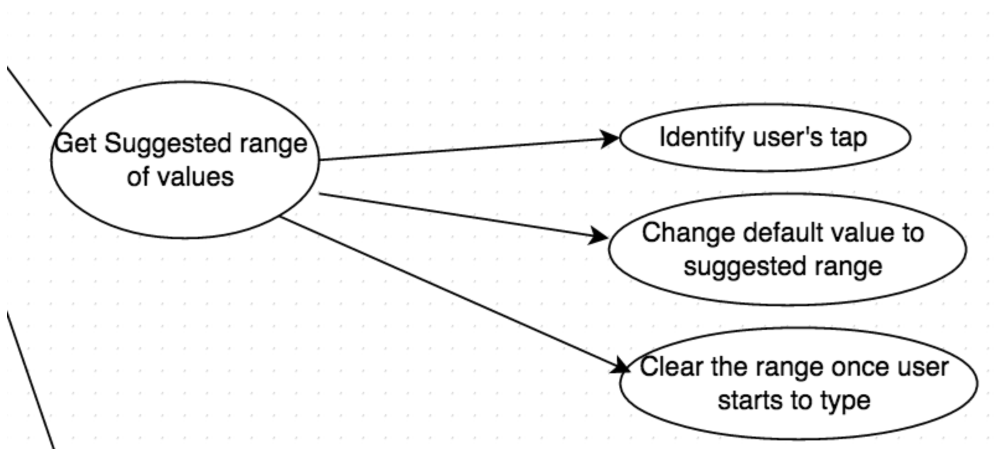
The "Get Help" functionality (Figure 16) should display the helper message and then wait for the user to complete reading and click "OK". After the "OK" selection is identified, it should close the message.

Figure 16. Analysis of "Get Help" function



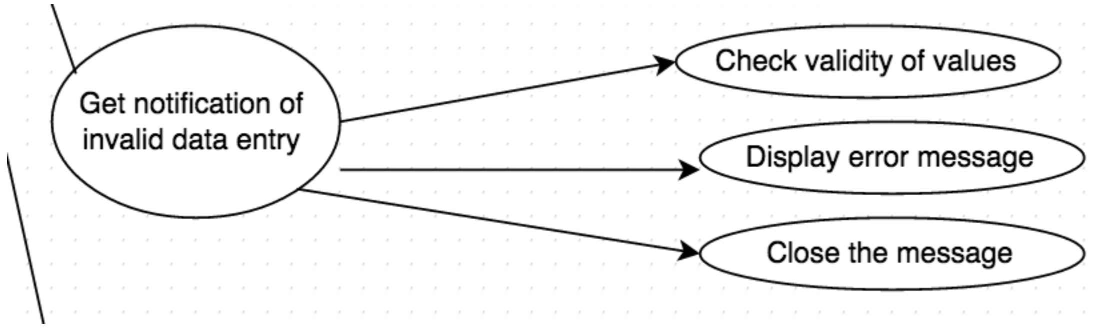
The "Get Suggested Range of Values" (Figure 17) should first identify the tap in the input screen. Every input box has a value by default; when the user taps on the input box and attempts to enter a value instead of default values, the suggested range is shown for the specific entry. It should also clear the suggestion once the user begins to enter values.

Figure 17. Analysis of "Get Suggested Range of Values" function



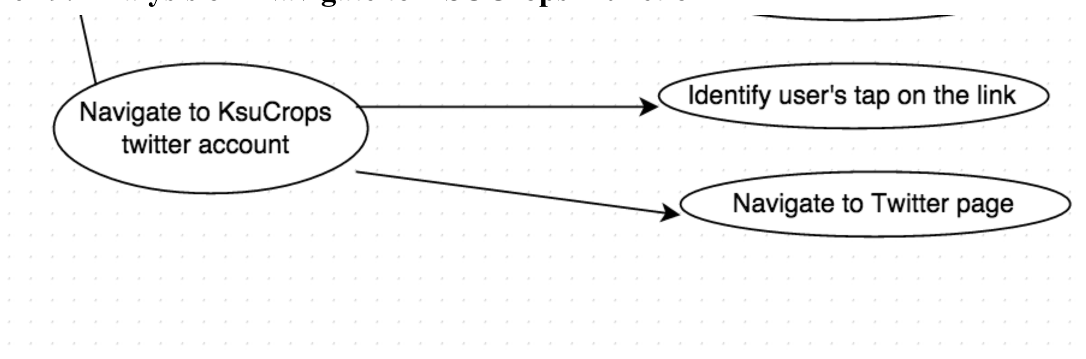
The next functionality (Figure 18) to be analyzed is "Get Notification of Invalid Data Entry". Once the user enters entry and the app retrieves it, the app verifies validity of the entry. In the case of an invalid entry, the app displays a message recognizing the invalid entry and then it waits for the user to click "OK," at which time the display message should close.

Figure 18. Analysis of "Get Notification of Invalid Data Entry" function



The last functionality (Figure 19) extended in this study was "Navigate to KSUCrops Twitter Account." The KSUSoy YieldCalc app was expected to identify the tap on the provided link, leading to a Twitter page.

Figure 19. Analysis of "Navigate to KSUCrops" function



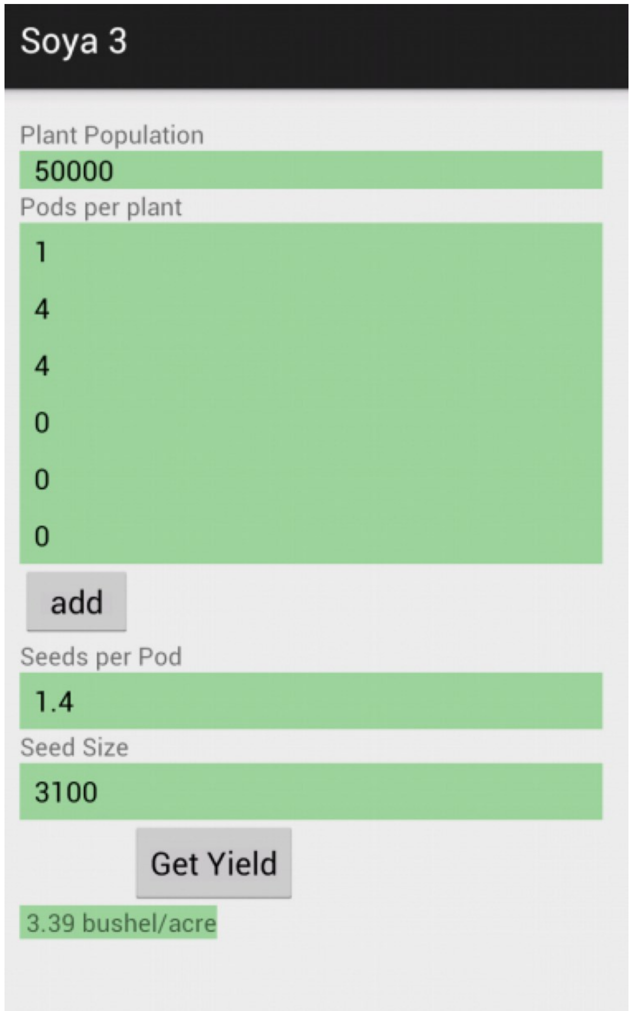
3.3. Building the Project

3.3.1. User Interface

The KSUSoy YieldCalc application project included several design phases that each contained unique advantages and disadvantages.

Design 1: The app was initially designed as shown in Figure 20.

Figure 20. Interface in Phase-1 of design



Advantages of this design phase included the following:

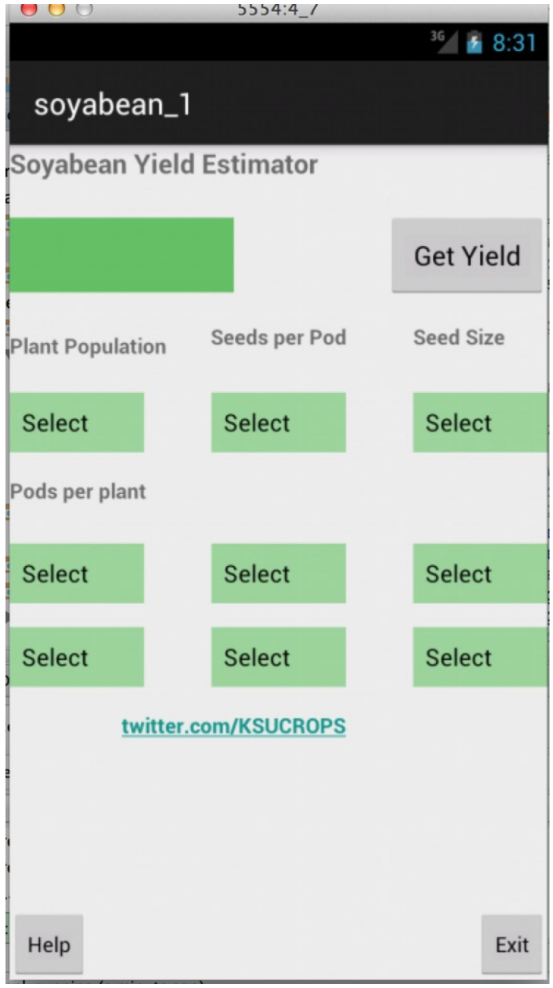
1. Users were provided a drop-down list to select values.
2. Users could add input boxes for Pods/Plant.
3. The display was simple and easy for users.

Disadvantages of this design phase included the following:

1. The view became scrollable depending on the number of additional input boxes and screen size of the user's Android device.
2. The layout of space in this phase was inefficient.

Design 2: Design revisions from the first phase led to a limited number of input boxes for pods per plant (Figure 21).

Figure 21. Interface of Phase-2 of design



Advantages of this design included the following:

1. The layout fit onto one screen irrespective of screen size and density of the android device.
2. Users were provided a drop-down list from which to select values.
3. The layout was convenient and simple.
4. The Twitter link to KSUCROPS was provided in order to obtain current updates.

Disadvantages of this design included the following:

1. The number of “Pods per Plant” input boxes were limited.

Design 3: After analyzing advantages and disadvantages of each design while accounting for farmer feedback and Dr. Ignacio Ciampitti’s (associate professor in the Department of Agronomy at Kansas State University) suggestion, design Phase 3 (Figure 22 and Figure 23) was executed.

Advantages of this phase included the following:

1. This design increased ease of operation for farmers
2. Ten boxes were available for pods per plant.
3. Suggested values and ranges for each input box were provided.
4. Spanish translation of the app was available based on setup locale.
5. Positive feedback was obtained from agronomist and consultants.
6. “Help,” “exit,” and “error” messages were provided.

Disadvantages of this phase included the following:

1. The layout was clustered according to User Interface (UI) norms.
2. No drop-down list was provided.

Figure 22. Final interface design with logos

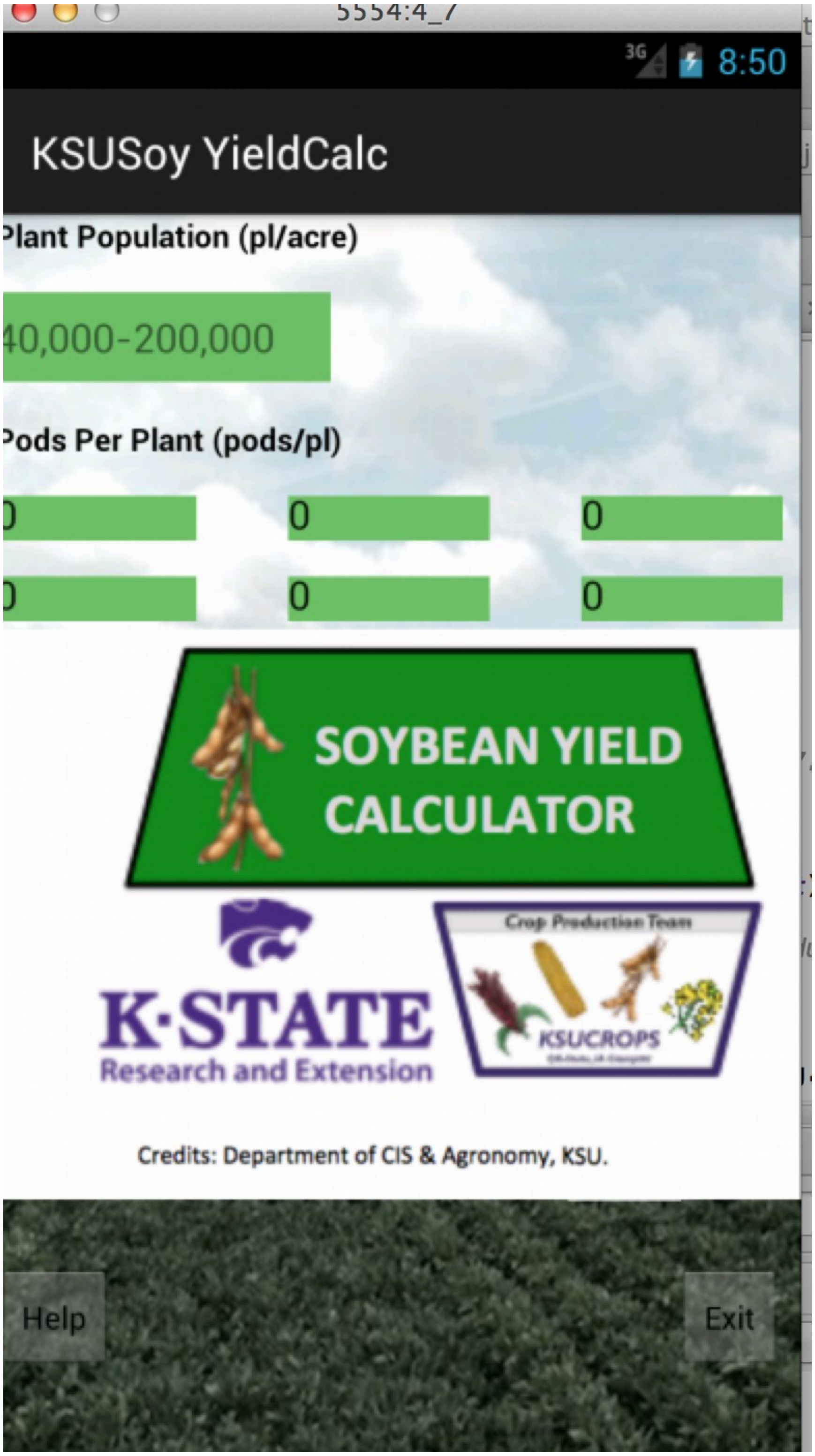
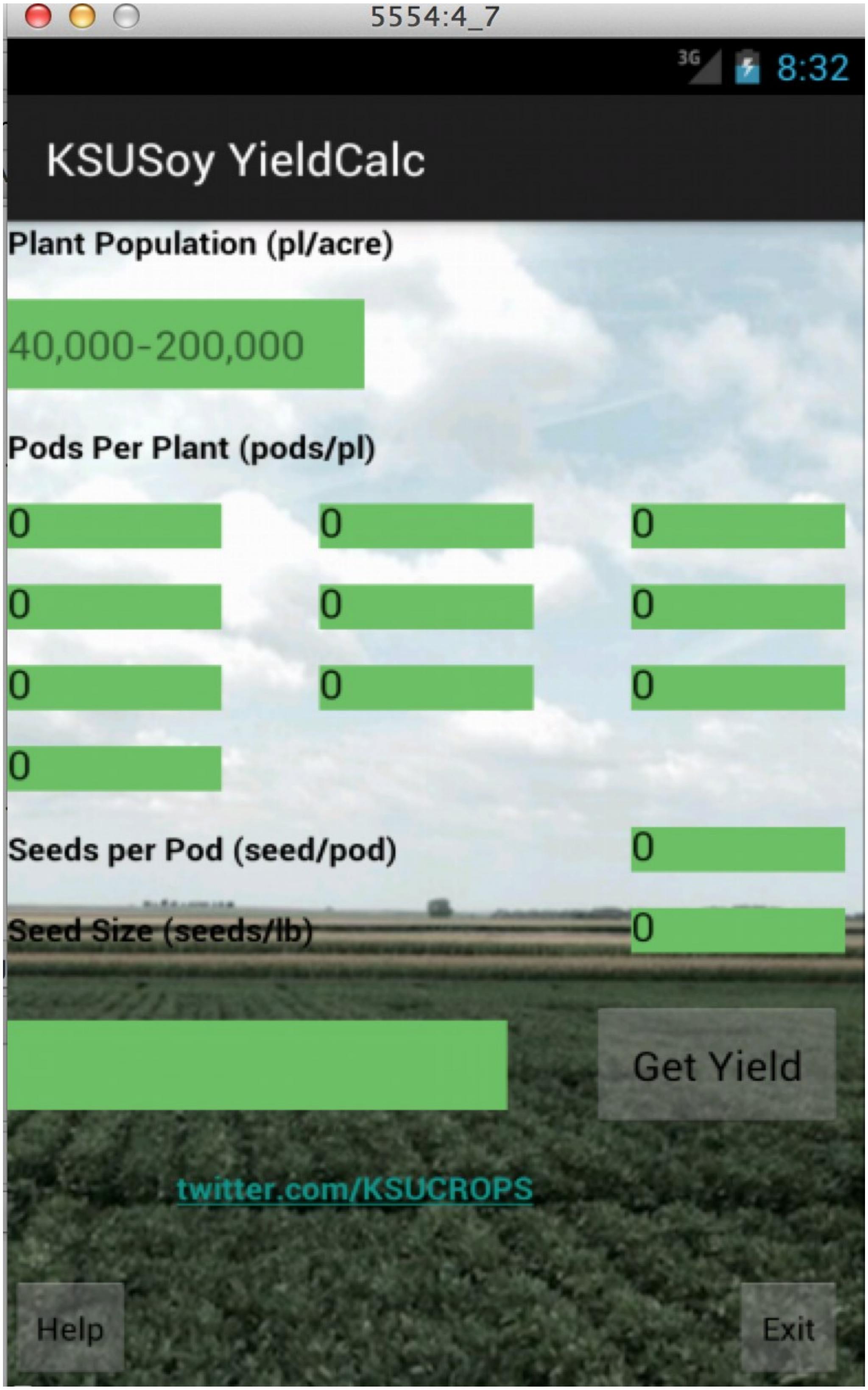


Figure 23. Interface of KSUSoy YieldCalc



3.3.2. Logic behind the Code

Equations for the solution required four user parameters: Plant Population, Pods per Plant, Seeds per Pod, and Seed Size The primary motivations for yield estimate calculation are:

1. Total Number of Pods

The total number of pods per plant within a constant row length should be counted. A final pod number was obtained when all the plants within sections that represented 1/10,000$^{\text{the}}$ of an acre were counted. A similar procedure was exercised in various sections of the field to obtain an overall estimation at the field-scale. (Ciampitti 2014).

2) Total Number of Seeds per Pod

Soybean plants typically offer an average of 2.5 seeds per pod (the given range is 1 to 4 seeds per pod). The primary determining criterion is the interaction between the environment and genotypes. Under severe drought and heat stress, a pessimistic approach considers an average of 1 to 1.5 seeds per pod, an approximate value that may change throughout the growing season.

3) Seed Size

The seed size factor for soybean ranges from 2,500 (normal to large seed weight) to 3,500 (small seed size) seeds per pound. Conditions throughout the growing season favorably promoted large seed sizes, meaning that smaller seeds per pound were expected. However, if a growing season experiences drought effect, as has been the case for the past two growing seasons, then a smaller final seed size could be expected, meaning a larger number for seeds per pound. According to Dr. Casteel, soybean specialist at Purdue University,

- If conditions are favorable and large seed size is expected, then the conversion to be used is 15 units;

- If abiotic or biotic stresses are imposed during the grain-filling period, a predictable small size of seeds will correlate to a seed size factor of 21 units. (Casteel 2012)

For example, 120,000 plants/acre (plant density) were planted in a 30-in. row, leading to an expectation of approximately 12 plants in 21-in. Within those 12 plants, an average of 22 pods per plant were measured, totaling 264 pods (22 x 12). If a “normal” growing season condition is assumed, then the final seeds per pod would be approximately 2.5 and large seeds would be expected for the seed size factor, thus 15 units.

The following equation can be used for a “favorable” season:

264 pods x 2.5 seeds per pod / 15 = 44 bushels per acre

A “drought” growing season (late reproductive, from R2 to R6 stages) dramatically affects the final seed number and size. Therefore, even if the pod number is identical to the pod number in a “normal” season, the yield calculation would be

264 pods x 1.5 seeds per pod / 21 = 19 bushels per acre. (Ciampitti 2014)

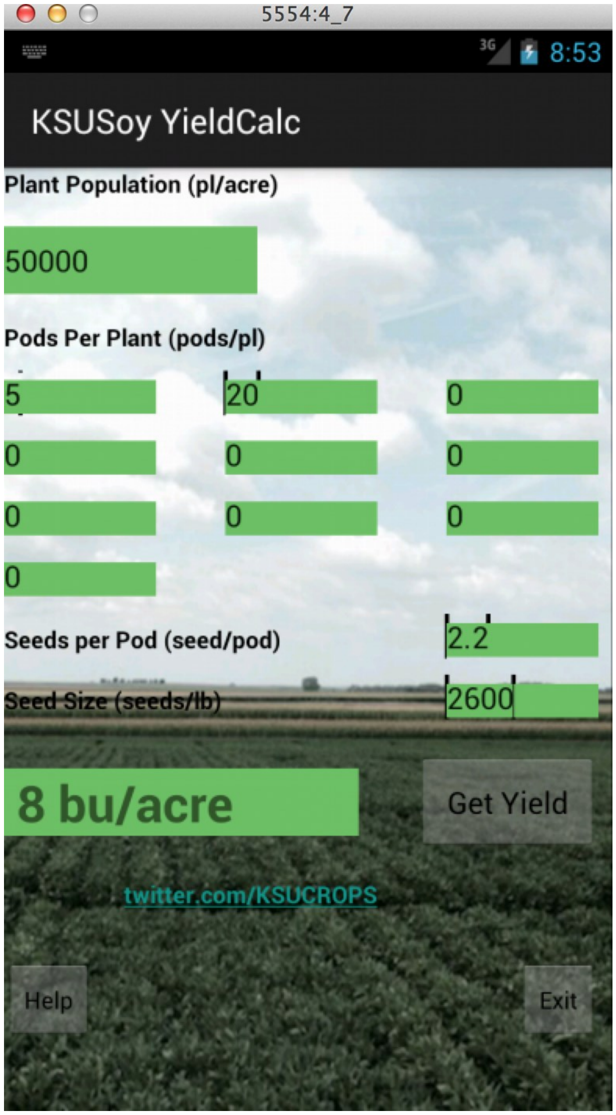
In the conventional approach followed by the KSUSoy YieldCalc app, soybean yield estimates were based on the following components:

1. Total number of pods per acre [number of plants per acre x pods per plant-(1)
2. Total number of seeds per pod-(2)
3. Number of seeds per pound-(3)
4. Total pounds per bushel, or test weight (60 lbs./bu for soybeans)-(4)

The equation for estimation of potential soybean yield used in our solution (Figure 24) is

[(1) X (2) / (3)] / (4) = Soybean yield in bushels/acre

Figure 24. Demonstration of calculated estimate



Chapter 4 - Testing

KSU's Department of Agronomy tested the KSUSoy YieldCalc app in comparison with two methods described in the following subsections.

4.2.1. Methods of Testing

A dataset with 40 cases was generated using the function "randbetween" in Microsoft Excel (Table 1). Generated variables were:

Plants per acre: random population for each case of 10 numbers between 40000 and 120000.

Number of pods per plant: random population for each case of 10 numbers between 1 and 50.

Number of seeds per pod: random population for each case of 10 numbers between 1 and 4.

Seed Size: random population of 10 numbers between 2500 and 3500.

When the dataset was randomly generated and value ranges were presented, three methods were used to estimate the yield:

Method 1: This method (Figure 25) calculates yield in an Excel spreadsheet. The population was multiplied by the number of pods, and then the number of grains per pod were multiplied by the number of pods. Finally, yield was calculated by accounting for seed size; the final value was multiplied by 60 to obtain the accurate unit (bu/ac)

Method 2: This method (Figure 26) calculates yield using an online webpage and the initial dataset, http://mpgabeanapp.com/soybean-yield-estimator .

- Method 3: This method (Figure 27) calculates yield using soybean yield estimation and original datasets.

4.1. Testing Results

The KSUSoy YieldCalc app was tested according to two factors:

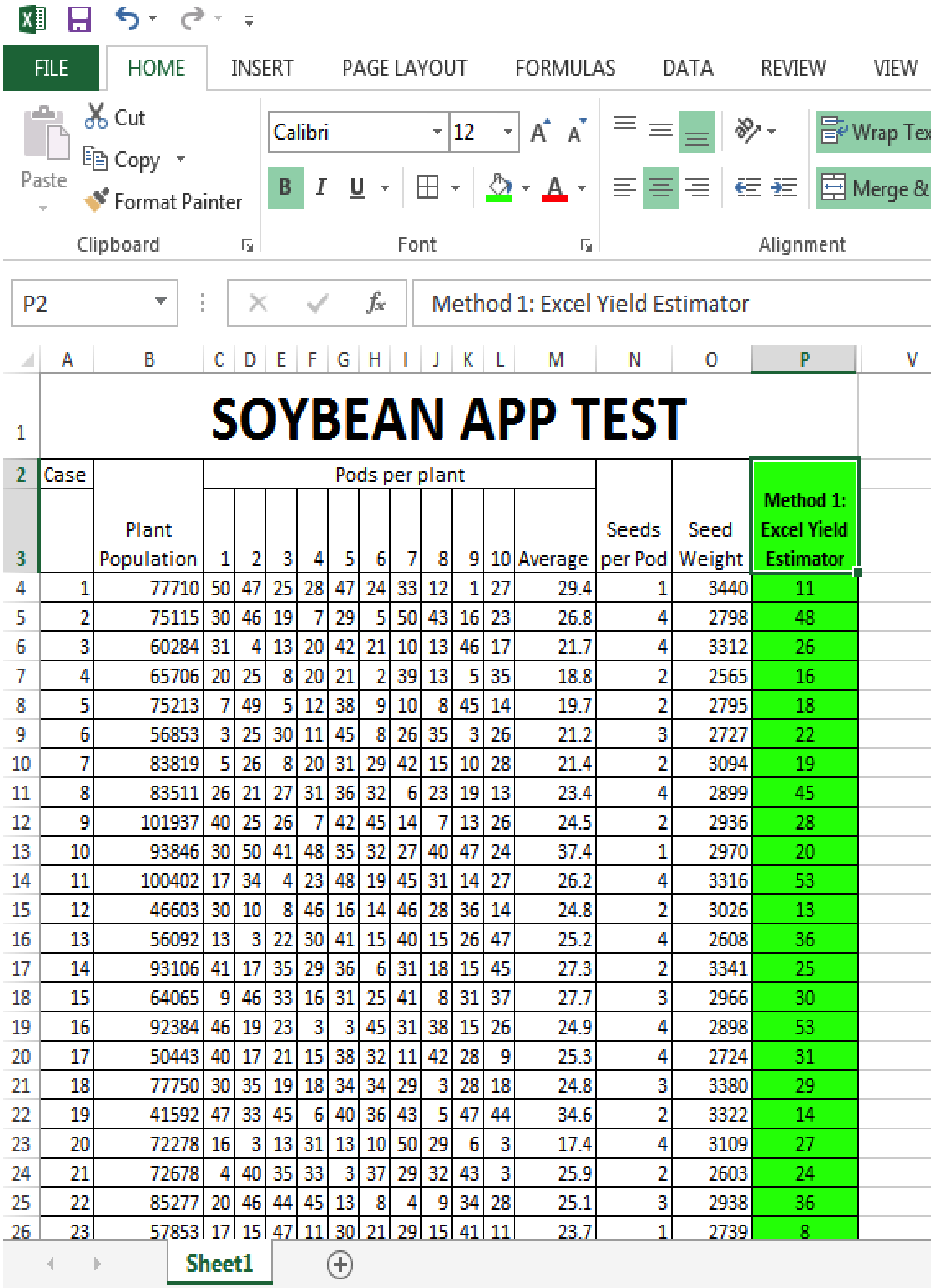
1) Testing based on correctness: The app was tested to validate results and confirm its functional prospect. It was tested against existing methods with a dataset of 80 records; results were very promising. Students in the Department of Agronomy at KSU conducted the testing with real data. The app was demonstrated to farmers who tested the app against their data. Users were pleased with the correctness, accuracy, and speed of results.

2) Testing of the interface: This testing primarily involved testing the app's user interface in relation to ease of layout and convenience of use. The farmers primarily preferred a simple layout with manual entry, as provided to them in the final design phase. Users appreciated a screen layout that did not require scrolling and contained all parameters and displayed results on one page. They also enjoyed receiving suggestions regarding the range of values to be entered against each input box while simultaneously allowing a value outside the range for data entry. The "Help" and "Error" functions proved to be good guides. Feedback was overall very positive and satisfying.

4.1.1. Testing based on Correctness

Results for all methods against the dataset (Table 1) were compared to KSUSoy YieldCalc app results by subtracting the yield for each case. The result of each subtraction showed no differences between the three methods (Table 2).

Figure 25. Method 1: Excel spreadsheet using the function



SOYBEAN APP TEST

Case	Plant Population	Pods per plant 1	2	3	4	5	6	7	8	9	10	Average	Seeds per Pod	Seed Weight	Method 1: Excel Yield Estimator
1	77710	50	47	25	28	47	24	33	12	1	27	29.4	1	3440	11
2	75115	30	46	19	7	29	5	50	43	16	23	26.8	4	2798	48
3	60284	31	4	13	20	42	21	10	13	46	17	21.7	4	3312	26
4	65706	20	25	8	20	21	2	39	13	5	35	18.8	2	2565	16
5	75213	7	49	5	12	38	9	10	8	45	14	19.7	2	2795	18
6	56853	3	25	30	11	45	8	26	35	3	26	21.2	3	2727	22
7	83819	5	26	8	20	31	29	42	15	10	28	21.4	2	3094	19
8	83511	26	21	27	31	36	32	6	23	19	13	23.4	4	2899	45
9	101937	40	25	26	7	42	45	14	7	13	26	24.5	2	2936	28
10	93846	30	50	41	48	35	32	27	40	47	24	37.4	1	2970	20
11	100402	17	34	4	23	48	19	45	31	14	27	26.2	4	3316	53
12	46603	30	10	8	46	16	14	46	28	36	14	24.8	2	3026	13
13	56092	13	3	22	30	41	15	40	15	26	47	25.2	4	2608	36
14	93106	41	17	35	29	36	6	31	18	15	45	27.3	2	3341	25
15	64065	9	46	33	16	31	25	41	8	31	37	27.7	3	2966	30
16	92384	46	19	23	3	3	45	31	38	15	26	24.9	4	2898	53
17	50443	40	17	21	15	38	32	11	42	28	9	25.3	4	2724	31
18	77750	30	35	19	18	34	34	29	3	28	18	24.8	3	3380	29
19	41592	47	33	45	6	40	36	43	5	47	44	34.6	2	3322	14
20	72278	16	3	13	31	13	10	50	29	6	3	17.4	4	3109	27
21	72678	4	40	35	33	3	37	29	32	43	3	25.9	2	2603	24
22	85277	20	46	44	45	13	8	4	9	34	28	25.1	3	2938	36
23	57853	17	15	47	11	30	21	29	15	41	11	23.7	1	2739	8

Figure 26. Method 2: Web application (Association 2013)

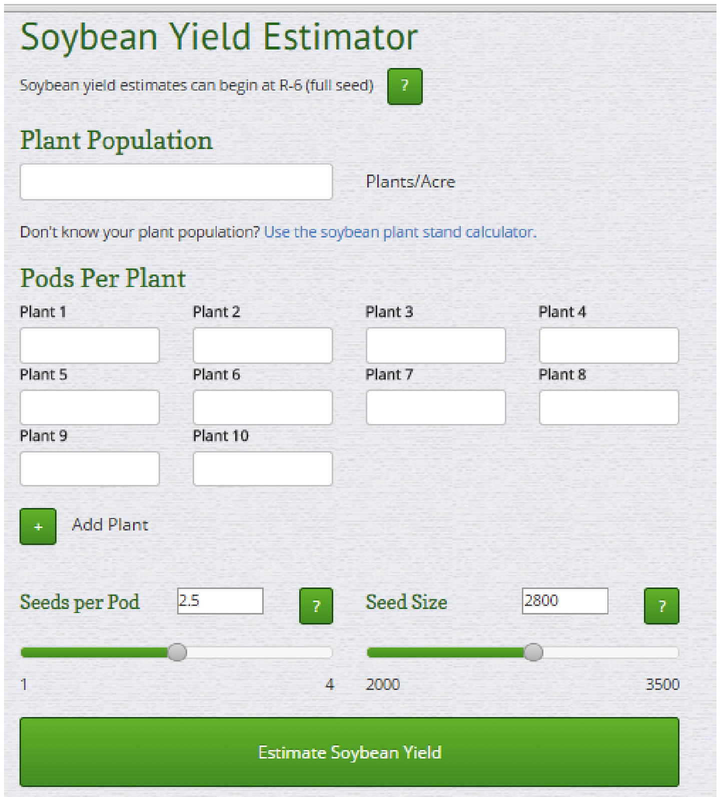


Figure 27. Method 3: KSUSOY YieldCalc

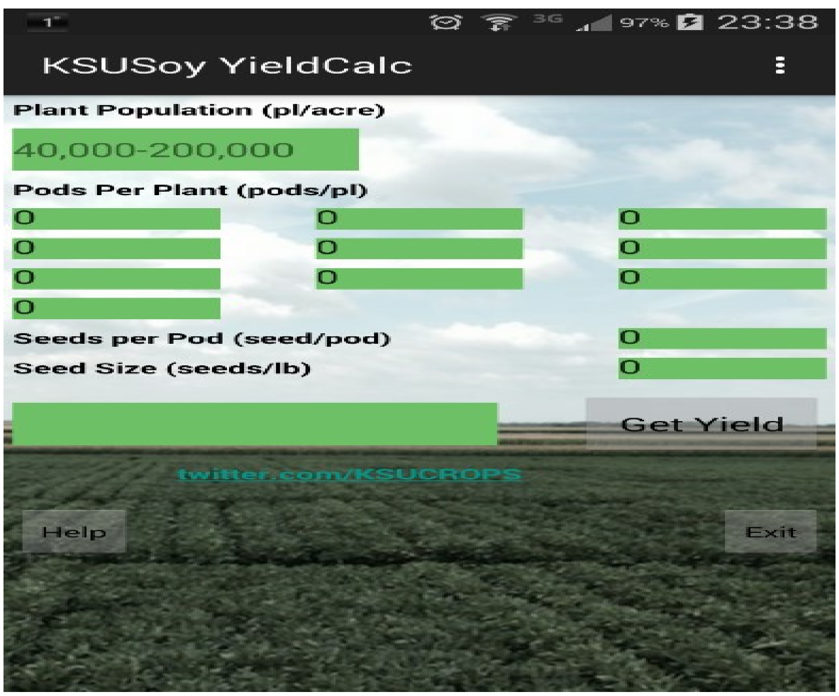


Table 1. Dataset

Case	Plant Population	Pods per plant											Seeds per Pod	Seed Weight
		1	2	3	4	5	6	7	8	9	10	Average		
1	77710	50	47	25	28	47	24	33	12	1	27	29.4	1	3440
2	75115	30	46	19	7	29	5	50	43	16	23	26.8	4	2798
3	60284	31	4	13	20	42	21	10	13	46	17	21.7	4	3312
4	65706	20	25	8	20	21	2	39	13	5	35	18.8	2	2565
5	75213	7	49	5	12	38	9	10	8	45	14	19.7	2	2795
6	56853	3	25	30	11	45	8	26	35	3	26	21.2	3	2727
7	83819	5	26	8	20	31	29	42	15	10	28	21.4	2	3094
8	83511	26	21	27	31	36	32	6	23	19	13	23.4	4	2899
9	101937	40	25	26	7	42	45	14	7	13	26	24.5	2	2936
10	93846	30	50	41	48	35	32	27	40	47	24	37.4	1	2970
11	100402	17	34	4	23	48	19	45	31	14	27	26.2	4	3316
12	46603	30	10	8	46	16	14	46	28	36	14	24.8	2	3026
13	56092	13	3	22	30	41	15	40	15	26	47	25.2	4	2608
14	93106	41	17	35	29	36	6	31	18	15	45	27.3	2	3341
15	64065	9	46	33	16	31	25	41	8	31	37	27.7	3	2966
16	92384	46	19	23	3	3	45	31	38	15	26	24.9	4	2898
17	50443	40	17	21	15	38	32	11	42	28	9	25.3	4	2724
18	77750	30	35	19	18	34	34	29	3	28	18	24.8	3	3380
19	41592	47	33	45	6	40	36	43	5	47	44	34.6	2	3322

20	72278	16	3	13	31	13	10	50	29	6	3	17.4	4	3109
21	72678	4	40	35	33	3	37	29	32	43	3	25.9	2	2603
22	85277	20	46	44	45	13	8	4	9	34	28	25.1	3	2938
23	57853	17	15	47	11	30	21	29	15	41	11	23.7	1	2739
24	51008	2	6	14	23	13	38	13	15	16	43	18.3	2	2758
25	87032	47	26	21	1	40	25	7	24	33	43	26.7	1	2549
26	111902	10	45	6	18	5	43	47	13	15	14	21.6	3	2646
27	85314	42	3	4	1	36	4	5	41	2	33	17.1	3	2921
28	49534	8	6	43	28	12	39	50	5	23	24	23.8	4	2964
29	101706	43	28	17	40	8	50	4	45	12	13	26	4	3031
30	55129	33	35	18	18	31	45	30	46	11	20	28.7	2	3388
31	79049	13	38	26	24	17	30	42	24	22	2	23.8	4	3404
32	63310	17	8	27	47	49	25	26	15	39	20	27.3	2	3000
33	87346	33	28	48	34	3	8	41	28	32	15	27	2	2542
34	47798	30	48	9	19	42	45	29	17	1	1	24.1	1	3226
35	65270	50	35	4	2	21	38	19	2	4	15	19	3	3141
36	79386	38	16	35	4	15	2	41	1	4	35	19.1	1	2971
37	57905	3	10	41	4	50	38	47	37	36	45	31.1	4	2907
38	98985	39	13	31	26	19	27	49	22	40	26	29.2	4	3224
39	113208	9	16	11	32	41	17	3	49	9	10	19.7	4	2720
40	81492	43	34	11	32	50	48	34	4	38	2	29.6	2	2664

Table 2. Comparison of yield by methods for the first 20 cases

Method 1: Excel Yield Estimator	Method 2: Web Soybean Yield Estimator	Method 3: Soybean App Yield	Comparison		
			M1-M2	M1-M3	M2-M3
11	11	11	0	0	0
48	48	48	0	0	0
26	26	26	0	0	0
16	16	16	0	0	0
18	18	18	0	0	0
22	22	22	0	0	0
19	19	19	0	0	0
45	45	45	0	0	0
28	28	28	0	0	0
20	20	20	0	0	0
53	53	53	0	0	0
13	13	13	0	0	0
36	36	36	0	0	0
25	25	25	0	0	0
30	30	30	0	0	0
53	53	53	0	0	0
31	31	31	0	0	0
29	29	29	0	0	0

14	14	14	0	0	0
27	27	27	0	0	0

According to the previous table, the conclusion was made that the three methods produced identical results. Estimated yield was the variable used to compare app performance.

4.1.2. Testing of Interface

Farmers tested the interface to ensure convenience of use. Feedback was very positive primarily due to the following features:

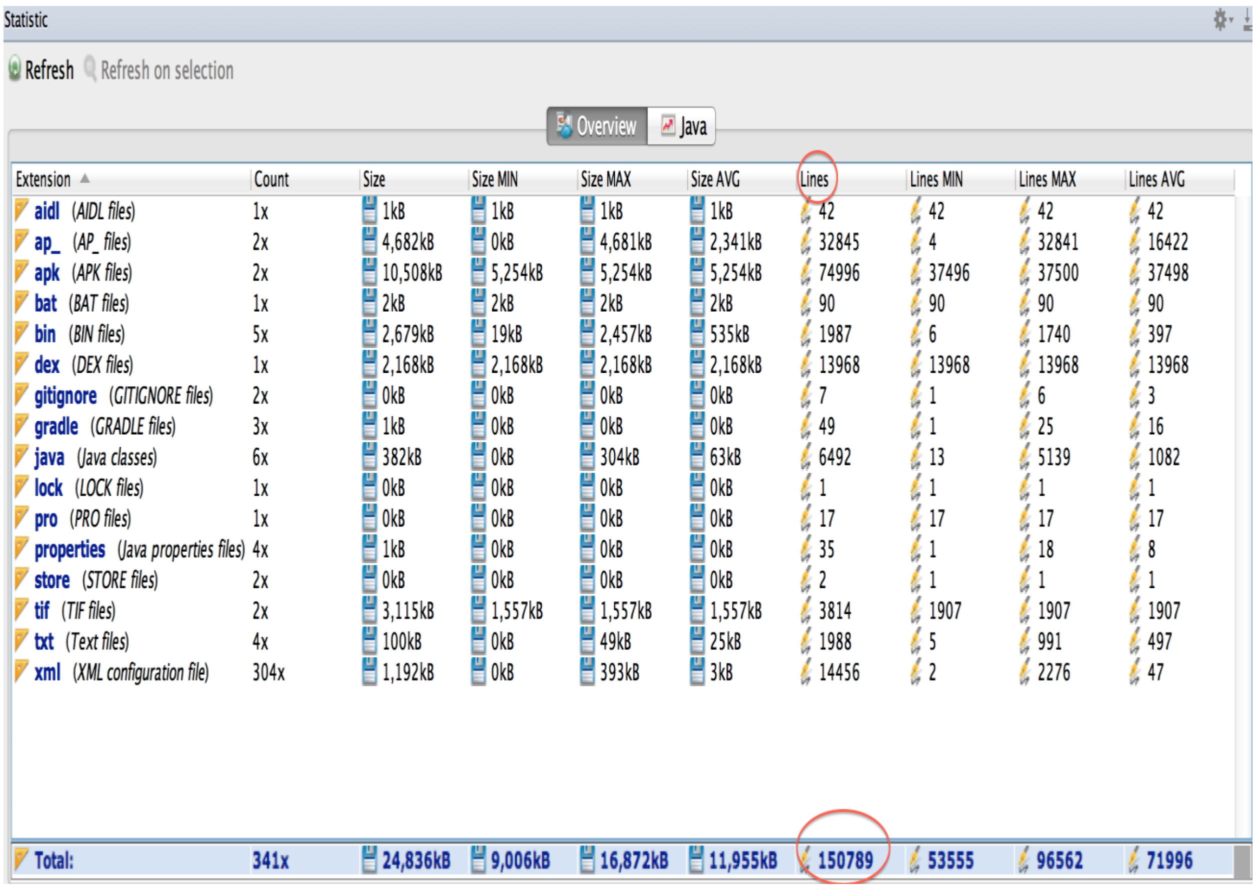
1. Manual entry: Manual entry is a conventional approach for input of values that asks users to select from a list of given values. Farmers preferred manual entry because they were familiar with the parameters. Manual entry also provides ease of use and does not require a user to scroll through a list, especially beneficial for cellular devices small screen sizes. The KSUSoy YieldCalc application provides manual entry.
2. Suggested values: Although the KSUSoy Yield Calc application requires manual entries of values, it also provides suggested value ranges if the user wants to enter a value in the input box, thereby alerting the user to the suggested range of values in addition to manual entry without scrolling.
3. Auto-fit layout: The KSUSoy YieldCalc application adapts itself to the screen size of any android device without requiring scrolling. Users greatly appreciated this feature, and reactions were very satisfying.
4. Helper dialogues: The KSUSoy YieldCalc application provides a short dialog message in simple language to advise farmers on how to use the application.

5. Error messages: Although the application provides suggested ranges, farmers did not want the application to restrict calculation to that range. Therefore, farmers can enter values outside the suggested range. In order to ensure value validity, error functions that display dialogues in the event of data entry errors are provided.
6. Twitter link to KSUCrops: Many agronomists, consultants, and farmers access and follow KSUCrops twitter page. A link has been provided in the application interface through which users can navigate to the Twitter page to view current information.
7. Units: The units for each entry and results are provided for convenience of use.
8. Languages: The KSUSoy YieldCalc app is provided in English and Spanish based on the set locale of the utilized device, consequently targeting a diversified range of farmers.
9. Orientation: The KSUSoy YieldCalc app can be automatically oriented in landscape or portrait mode without requiring settings.

Chapter 5 - Evaluation

This application project was evaluated using the “Statistics” plug-in for Android Studio. The project contained a total of 341x files and 150789 lines of code (Figure 28) , including 6492 lines of code in Java (Figure 29) . The KSUSoy YieldCalc app offered results less than 20 seconds after parameters were given its values.

Figure 28. Metric analysis of the project

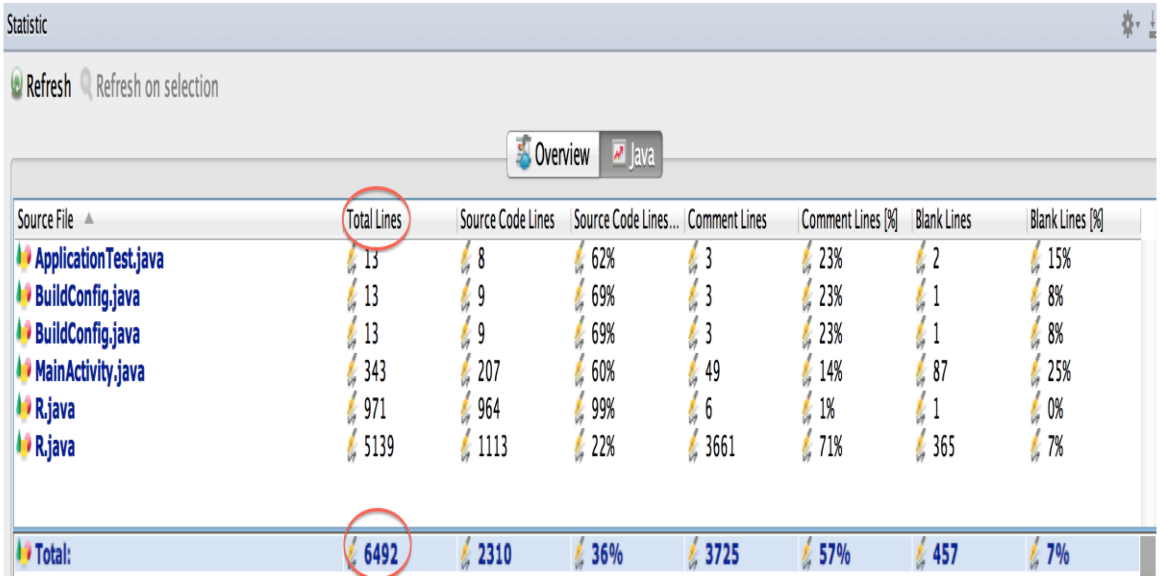

Statistic

Refresh Refresh on selection

Overview Java

Extension	Count	Size	Size MIN	Size MAX	Size AVG	Lines	Lines MIN	Lines MAX	Lines AVG
aidl (AIDL files)	1x	1kB	1kB	1kB	1kB	42	42	42	42
ap_ (AP_ files)	2x	4,682kB	0kB	4,681kB	2,341kB	32845	4	32841	16422
apk (APK files)	2x	10,508kB	5,254kB	5,254kB	5,254kB	74996	37496	37500	37498
bat (BAT files)	1x	2kB	2kB	2kB	2kB	90	90	90	90
bin (BIN files)	5x	2,679kB	19kB	2,457kB	535kB	1987	6	1740	397
dex (DEX files)	1x	2,168kB	2,168kB	2,168kB	2,168kB	13968	13968	13968	13968
gitignore (GITIGNORE files)	2x	0kB	0kB	0kB	0kB	7	1	6	3
gradle (GRADLE files)	3x	1kB	0kB	0kB	0kB	49	1	25	16
java (Java classes)	6x	382kB	0kB	304kB	63kB	6492	13	5139	1082
lock (LOCK files)	1x	0kB	0kB	0kB	0kB	1	1	1	1
pro (PRO files)	1x	0kB	0kB	0kB	0kB	17	17	17	17
properties (Java properties files)	4x	1kB	0kB	0kB	0kB	35	1	18	8
store (STORE files)	2x	0kB	0kB	0kB	0kB	2	1	1	1
tif (TIF files)	2x	3,115kB	1,557kB	1,557kB	1,557kB	3814	1907	1907	1907
txt (Text files)	4x	100kB	0kB	49kB	25kB	1988	5	991	497
xml (XML configuration file)	304x	1,192kB	0kB	393kB	3kB	14456	2	2276	47
Total:	341x	24,836kB	9,006kB	16,872kB	11,955kB	150789	53555	96562	71996

Figure 29. Total lines of JAVA code



Statistic

Refresh Refresh on selection

Overview Java

Source File	Total Lines	Source Code Lines	Source Code Lines...	Comment Lines	Comment Lines [%]	Blank Lines	Blank Lines [%]
ApplicationTest.java	13	8	62%	3	23%	2	15%
BuildConfig.java	13	9	69%	3	23%	1	8%
BuildConfig.java	13	9	69%	3	23%	1	8%
MainActivity.java	343	207	60%	49	14%	87	25%
R.java	971	964	99%	6	1%	1	0%
R.java	5139	1113	22%	3661	71%	365	7%
Total:	6492	2310	36%	3725	57%	457	7%

This project increased understanding of agronomical terms and techniques of which the author had no prior substantial knowledge. Research for this project also increased understanding of the nuances of Android techniques, including its advantages and disadvantages. The KSUSoy YieldCalc application targets a simplistic community. Therefore, a balance of simplicity and sophistication must be maintained throughout interface design. Design interface was achieved after three phases of design.

Chapter 6 - Future Work

The KSUSoy YieldCalc application is a relevant app for farmers and agronomists, but it contains opportunity for future expansion or customization depending on user needs. Various interfaces and functional propositions could be incorporated in future, including the following:

1) A separate activity for display of Pods per Plant: This feature was considered for application but neglected for farmers' convenience. Dr. Daniel Andresen proposed a separate activity for entry in Pods per Plant instead of clustering the page with additional 10 entry box. However, the idea was not implemented due to farmers' feedback stating that most farmers wanted a single page provides input boxes and results.
2) Email: A future extension may include emailing results to the farmers or consultants. However, this extension requires Internet connection which is currently unavailable with the KSUSoy YieldCalc application.

Bibliography

AGWEB. http://www.agweb.com/soybean-calculator/.

Android developers. *Android developers.* http://developer.android.com/about/index.html.

Association, Manitoba Pulse Growers'. *Soybean Yield Estimator.* 01 01, 2013. http://mpgabeanapp.com/soybean-yield-estimator (accessed 08 08, 2-14).

Casteel, Shaun N. *Pest & Crop.* 08 17, 2012. http://extension.entm.purdue.edu/pestcrop/2012/issue21/index.html (accessed 08 31, 2015).

Christopher Doering, Gannett Washington Bureau. *USA Today.* http://www.usatoday.com/story/news/nation/2013/03/03/farming-technology-ipad-apps/1959139/.

Ciampitti, Ignacio. *K-State Agronomy eUpdates.* 08 01, 2014. https://webapp.agron.ksu.edu/agr_social/eu_article.throck?article_id=313 (accessed 09 10, 2014).

CME Group. http://www.cmegroup.com/tools-information/ag-calculator.html.

Crop Calculator| Integrated Pest and Crop Mangement. http://ipcm.wisc.edu/apps/crop-calculators/.

developer.android.com. *Developer Workflow.* 01 01, 2014. http://developer.android.com/tools/workflow/index.html (accessed 08 31, 2014).

—. *Managing Projects Overview.* 01 01, 2014. http://developer.android.com/tools/projects/index.html (accessed 08 31, 2014).

—. *publishing.* 01 01, 2014. http://developer.android.com/tools/publishing/publishing_overview.html (accessed 08 31, 2014).

Gerke, Jason. *Missouri FarmBureau.* http://www.mofb.org/NewsMedia/ShowMeArticles.aspx?articleID=259.

Herbek, Chad Lee and Jim. "UK Cooperative Extension Service." 12 01, 2005. http://www2.ca.uky.edu/agc/pubs/agr/agr188/agr188.pdf (accessed 08 31, 2014).

Hillco Technologies. http://www.hillcotechnologies.com/calculator/corn-soy-calculator.html.