SE 3XA3: Test Report
CamRuler

Kshitij Mehta, mehtak1
Meet Patel, patel16
Prince Kowser, kowserm

December 7, 2017

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
11/25/2017	1.0	Being Test Cases
11/30/2017	1.1	Add results to tests completed so far
11/06/2017	1.2	Final Copy Rev1

1 Functional Requirements Evaluation

1.1 Home Screen

1. FR-HS-1
   Type: Functioanl, Dynamic, Manual

   Initial State: Application on home screen

   Input: Press About button

   Expected Output: A new page with instructions on how to use the application opens.

   Result: A new page with instructions on how to use the application opens.

2. FR-HS-2
   Type: Functioanl, Dynamic, Manual

   Initial State: Application on home screen

   Input: Press Start button

   Expected Output: A pop-up dialog box opens for the user to specifiy how many objects to measure.

   Result: A pop-up dialog box opens for the user to specifiy how many objects to measure.

3. FR-HS-3
   Type: Functioanl, Dynamic, Manual

   Initial State: Start button is pressed and pop-up dialog box is displayed.

   Input: User chooses 1 object to measure

   Expected Output: The system stores the number 1

   Result: The system reads the selection and stores the number 1 for the number of objects.

4. FR-HS-4
   Type: Functioanl, Dynamic, Manual

   Initial State: Start button is pressed and pop-up dialog box is displayed.

Input: User chooses 2 objects to measure

Expected Output: The system stores the number 2

Result: The system reads the selection and stores the number 2 for the number of objects.

1.2 Picture taking

5. FR-PT-5

   Type: Functional, Dynamic, Manual

   Initial State: Android application with a "take photo" button for the user to press.

   Input: Press button

   Expected Output: Open phones camera application

   Result: Upon the press of the button, the phone's camera application opens.

6. FR-PT-6

   Type: Functional, Dynamic, Manual

   Initial State: Phone camera

   Input: The user takes a picture

   Expected Output: Sets the picture to the specified image area on application

   Result: After the picture is taken, the picture is set as the background of the application

1.3 User Drawing

7. FR-UD-3

   Type: Functional, Dynamic, Manual

Initial State: A picture is taken and set on to the specified image area on application.

Input: Draw two dots.

Expected Output: A line is drawn between the two dots.

Result: Upon touching the screen, a dot is drawn at the touch location. After the second touch is detected, a line is automatically drawn.

8. FR-UD-4

   Type: Functional, Dynamic, Manual

   Initial State: A line is drawn between the two dots on the picture.

   Input: Draw another pair of dots.

   Output: A line drawn between the new pair of dots only.

   Result: After the first line is drawn, two new dots can be drawn where a line is drawn between those dots only.

9. FR-UD-5

   Type: Functional, Dynamic, Manual

   Initial State: All lines and dots are drawn.

   Input: Press the redraw button.

   Expected Output: All lines and dots are cleared and prompts the user to redraw.

   Result: Upon pressing the clear button, a dot is cleared from the screen.

10. FR-UD-6

    Type: Functional, Dynamic, Manual

    Initial State: User selected 1 object to measure and the frist line is drawn

    Input: User presses OK button

    Expected Output: Allow user to draw another pair of dots

    Result: Upon pressing the OK button, another dot is drawn on the screen.

11. FR-UD-7

    Type: Functional, Dynamic, Manual

    Initial State: User selected 1 object to measure and the second line is drawn

    Input: User presses OK button

    Expected Output: Opens dialog box

    Result: Upon pressing the OK button, a dialog box opens to enter in the measurements.

12. FR-UD-8

    Type: Functional, Dynamic, Manual

    Initial State: User selected 2 objects to measure and the first line is drawn

    Input: User presses OK button

    Expected Output: Allow user to draw another pair of dots

    Result: Upon pressing the OK button, another dot is drawn on the screen.

13. FR-UD-9

    Type: Functional, Dynamic, Manual

    Initial State: User selected 2 objects to measure and the second line is drawn

    Input: User presses OK button

    Expected Output: Allow user to draw another pair of dots

    Result: Upon pressing the OK button, another dot is drawn on the screen.

14. FR-UD-10

    Type: Functional, Dynamic, Manual

Initial State: User selected 2 objects to measure and the third line is drawn

Input: User presses OK button

Expected Output: Opens dialog box

Result: Upon pressing the OK button, a dialog box opens to enter in the measurements.

15. FR-UD-11

    Type: Functional, Dynamic, Manual

    Initial State: User draws 2 dots.

    Input: User touches a dot and drags it to a new location

    Expected Output: The dot follows the user's finger until the user releases the dot.

    Result: The dot is dragged to a new location and the line is automatically readjusted.

16. FR-UD-12

    Type: Functional, Dynamic, Manual

    Initial State: Use draws 1 dot.

    Input: User presses OK button

    Expected Output: Error message is displayed

    Result: An error message is displayed

17. FR-UD-13

    Type: Functional, Dynamic, Manual

    Initial State: The pop-up dialog box to input measurements is empty.

    Input: User presses OK button

    Expected Output: Error message is displayed

    Result: An error message is displayed

1.4 Calculation

18. FR-C-6

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information.

    Input: Enter reference object in cm and select output unit as cm

    Expected Output: A new window pops up with the length of the object

    Result: The length is displayed on a new dialog box with the correct conversion.

19. FR-C-7

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information.

    Input: Enter reference object in cm and select output unit as mm.

    Expected Output: A new window pops up with the length of the object.

    Result: The length is displayed on a new dialog box with the correct conversion.

20. FR-C-8

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information.

    Input: Enter reference object in cm and select output unit as m.

    Output: A new window pops up with the length of the object.

    Result: The length is displayed on a new dialog box with the correct conversion.

21. FR-C-9

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information.

Input: Enter reference object in mm and select output unit as cm.

Expected Output: A new window pops up with the length of the object.

Result: The length is displayed on a new dialog box with the correct conversion.

22. FR-C-10

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information

    Input: Enter reference object in mm and select output unit as mm.

    Expected Output: A new window pops up with the length of the object.

    Result: The length is displayed on a new dialog box with the correct conversion.

23. FR-C-11

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information.

    Input: Enter reference object in mm and select output unit as mm.

    Expected Output: A new window pops up with the length of the object.

    Result: The length is displayed on a new dialog box with the correct conversion.

24. FR-C-12

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information.

    Input: Enter reference object in m and select output unit as cm.

    Expected Output: A new window pops up with the length of the object.

    Result: The length is displayed on a new dialog box with the correct conversion.

25. FR-C-13

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information.

    Input: Enter reference object in m and select output unit as mm.

    Expected Output: A new window pops up with the length of the object.

    Result: The length is displayed on a new dialog box with the correct conversion.

26. FR-C-14

    Type: Functional, Dynamic, Automatic

    Initial State: Pop up window requiring measurement information.

    Input: Enter reference object in m and select output unit as m.

    Expected Output: A new window pops up with the length of the object.

    Result: The length is displayed on a new dialog box with the correct conversion.

2 Nonfunctional Requirements Evaluation

2.1 Look and Feel

1. SS-1

    Type: Structural, Static, Manual

    Initial State: The application is installed on the phone.

    Input: Several users are asked to launch the application and give feedback about the application's layout and theme.

    Expected Output: All users provide their feedback and rate the application on a scale of 1-5. At least 70% of users will rate the application 5/5.

    Result: Rate your experience with this mobile application out of 5 (1 being horrible, 5 being fantastic)

Rating	1	2	3	4	5
Percentage of Users Testesd	0%	0&	2%	24%	74%

Table 2: Look and Feel User Survey

2.2 Usability

2. SS-2

   Type: Structural, Static, Manual

   Initial State: The application is not running.

   Input: The user launches the application.

   Expected Output: The application provides an option for the user to see instructions on how to use the application and an option to start the option.

   Result: Upon launching the application, the About and Start buttons are displayed.

3. SS-3

   Type: Structural, Static, Manual

   Initial State: The application is on the Take A Picture button page.

   Input: The user takes a picture

   Expected Output: The instructions on what to do next are displayed on the bottom of the screen.

   Result: After the picture has been set as the background, there is a box at the bottom of the screen on what to do next.

4. SS-4

   Type: Structural, Static, Manual

   Initial State: The first pair of dots are drawn.

   Input: The user presses the OK button.

Expected Output: The instructions on what to do next keep changing every time a user presses OK.

Result: The instructions displayed at the bottom of the screen changes every time the OK button is pressed.

5. SS-5

   Type: Structural, Static, Manual

   Initial State: The second pair of dots are drawn.

   Input: The user presses the clear button.

   Expected Output: The instructions should change back to the first stage of the drawing sequence (i.e drawing the reference object).

   Result: The instructions displayed at the bottom of the screen changes when clear is pressed depending on the drawing stage. For example, if 4 dots are drawn and the user presses the Clear button, the instructions should not change. If the user presses the Clear button again, there are now 2 dots on the screen meaning that the user is on the stage of drawing the dots on the reference object. Thus, the instruction should change to that accordingly.

6. SS-6

   Type: Structural, Dynamic, Manual

   Initial State: The dialog box to enter the reference object's measurements is open with the measurements filled in.

   Input: Several users are asked to click on "units" to specify a unit of measurement.

   Expected Output: At least 70% of the users understand the metrics being used and select the appropriate unit.

   Result: Rate your uderstanding of the units used in this application out of 5 (1 being horrible, 5 being fantastic)

Rating	1	2	3	4	5
Percentage of Users Testesd	0%	0&	0%	5%	95%

Table 3: Units User Survey

7. SS-7

   Type: Structural, Static, Manual

   Initial State: The application is installed on the phone.

   Input: Several users with different Android phones are asked to launch the application.

   Expected Output: At least 70% of the users should be able to launch and use the application successfully.

   Result: Number of Users that can launch and use the application successfully.

Rating	1	2	3	4	5
Percentage of Users Testesd	0%	0&	0%	15%	85%

Table 4: Launch and Use User Survey

2.3 Performance

8. SS-8

   Type: Structural, Dynamic, Automatic

   Initial State: The dialog box for the measurements of the reference object is open and filled out.

   Input: The user taps the "Done" button.

   Expected Output: The application calculates the actual object's measurements within 3 seconds after the user taps "Done".

   Result: The application calculates the output is 0.113 seconds

9. SS-9

   Type: Structural, Dynamic, Automatic

   Initial State: The dialog box for the measurements of the reference object is open.

Input: User is asked to input varying measurements for the reference object(smaller - larger) and using various units.

Expected Output: The application calculates the actual object's measurements within 3 seconds after the user taps "Done"

Result: Depending on the number and unit converion, the application calculates and displays the measurements between 0.098 - 0.267 seconds.

10. SS-10

    Type: Structural, Dynamic, Automatic

    Initial State: The dialog box for the measurements of the reference object is open and filled out.

    Input: The user taps the "Done" button.

    Expected Output: The application calculates the actual object's measurements and has a precision of 2 decimal places.

    Result: The output is displayed with 2 decimal places.

11. SS-11

    Type: Structural, Dynamic, Manual

    Initial State: The picture is the background of the screen and the reference object, as well as the actual object are both selected.

    Input: The user taps the screen to draw another dot.

    Expected Output: The application display's a message "no more objects can be measured".

    Result: The application does not display this message. Instead, when the user taps the screen, the last dot is moved to the touch location. Due to time constraints, this test could not be explored further.

2.4 Robustness

12. SS-12

Type: Structural, Dynamic, Manual

Initial State: The picture is the background of the application.

Input: The user is asked to retake the picture several times.

Expected Output: The system does not crash and it allows to user the take as many pictures as they want.

Result: Due to time constraints, this test was not performed officially. However by running the application multiple times on a test device, this test was indirecty performed where the system did not crash regardless of the number of times a picture was taken.

3 Comparison to Existing Implementation

Since the existing implementation was successful in terms of its functionality, we could use it to compare our implementation to in order to see if we are going down the correct path. The following tests compare the program to the existing implementation called Camera Ruler:

- FR-PT-5
- FR-PT-6
- FR-UD-3
- FR-C-6
- FR-C-9

4 Unit Testing

As Java was being used to implement the project, the JUNIT Framework was used in order to perform unit testing. Methods which return values were be simplest to test using JUNIT. These methods include the ratio calculations and metric conversions which are test cases:

- FR-C-6
- FR-C-7
- FR-C-8
- FR-C-9
- FR-C-10
- FR-C-11
- FR-C-12
- FR-C-13
- FR-C-14

These test cases required an input value and an expected output value which was then verified by JUNIT. Because unit testing is quick using JUNIT, every method with an input and output was tested for normal values as well as edge case values. Inputs generating exceptions were also tested to make sure of correct error handling. Stubs were created to test all possible cases as we do not want the program to crash on any kind of input. .

5 Changes Due to Testing

5.1 Functional Requirements Testing

Several of the features implemented in CamRuler were inspired due to testing. For example, we decided to implement a dragging feature in our application because it was getting very tiring to clear a dot every time you wanted to move a dot. Therefore, test case FR-UD-11 was added to test this feature.

5.2 Nonfunctional Requirements Testing

5.2.1 Look and Feel

Based upon the Look and Feel User Survey and feedback, we made minor changes to the overall layout of the theme. For example, we changed the apperance of our buttons and enhanced the appearance of the home page.

5.2.2 Usability

Originally, we did not have an About page or specific instructions displayed at the bottom when drawing dots. However after reviewing the user feedback, we decided to implement those ideas to make CamRuler more user friendly. Furthermore, the decision to display units along with the measurements was also made based on the Units User Survey.

5.2.3 Performance

Not applicable.

5.2.4 Robustness

Not applicabale.

6 Automated Testing

This project underwent automated testing through JUNIT. Methods which have an input value such as the mathematical ratios used to calculate measurements of objects were tested through an automated means. These methods include the following test cases:

- FR-C-6
- FR-C-7
- FR-C-8
- FR-C-9
- FR-C-10

- FR-C-11
- FR-C-12
- FR-C-13
- FR-C-14

Stubs of edge cases, normal cases, and negative cases were created in order to ensure that the program methods run properly. Stubs were created to test error handling in case the user enters something wrong. Furthemore, tests that had to do with precision or time were also performed automatically. This includes test cases:

- SS-8
- SS-9
- SS-10

For these tests, the time was automatically compared against the maximum allowable time and the decimal precision was compared to maximum allowable decimal places. Most of the test cases had to performed manually as it was essential to verify each test visually. Test such as checking if a picture is set as the background of the application would have to be done manually.

7 Trace to Requirements

Requirement	Test case
FR2	FR-HS-1
FR3	FR-PT-5, FR-PT-6
FR4	FR-PT-5, FR-UD-3
FR5	FR-UD-4
FR6	FR-UD-5
FR7	FR-C-5
FR8	FR-C-(6-14)
FR9	FR-UD-11
FR10	FR-HS-4, FR-UD-8, FR-UD-9, FR-UD-10
FR11	FR-UD-12, FR-UD-13
NF-AP	SS-1
NF-EU	SS-2, SS-4, SS-5, SS-11
NF-UPR	SS-6
NF-AR	SS-4
NF-SLR	SS-8, SS-9
NF-PAR	SS-10
NF-RAR	SS-7
NF-CR	SS-2

Table 5: Trace to Requirements

8 Trace to Modules

Module	Test case
About	FR-HS-1
MainActivity	FR-HS-2, FR-HS-3, FR-HS-4, SS-1, SS-2, SS-7
Image	FR-PT-5, FR-PT-6, FR-UD-(3-13), SS-3, SS-4, SS-5, SS-6, SS-10, SS-11
SurfaceImage	FR-PT-6, SS-12
Calculate	FR-C-(6-14), SS-8, SS-10
DrawingOnImage	FR-UD-(3-11), SS-11
Utilities	FR-C-(6-14), Ss-8, SS-9

Table 6: Trace to Modules

9 Code Coverage Metrics

We believe that through our tests, we were able to produce approximately 95% code coverage. There were some very small parts of the code we could not test to the full extent like the picture taking component in the Image module and the dragging component in the DrawingOnImage module. Based on the tests that we performed, for the most part we believe that we covered Equivalence Testing, Boundary Testing and Control-flow Testing which is the code coverage criteria. Thus, based on the traceability matrices to Requirements and Modules, we believe we tested 95% of our code.