

Project and Professionalism

(6CS020)

Smart Calculator using OCR

Student Id : 2039244

Student Name : Sandip Thapa

Supervisor : Mr. Manish Deuja

Cohort/Batch : 4

Words Count :

Submitted on : 23-09-2020

Declaration

Abstact

Contents

# Introduction

## Introduction to OCR

Optical Character Recognition is a technology that allows machine to recognize the text whether it is scanned or printed text images or handwritten text. The machine can do further processing on the data extracted from that text. It can be considered same as the combination of human eye and mind. An eye can see the text from some source but mind is the one that actually processes and interprets that text.

OCR system is made up of combination of both hardware and software. Hardware such as, optical scanner or some specialized circuit board is used to read or extract text. And software does the advance processing.

The most common use of OCR is to convert hard copy documents into softcopy files such as PDFs. This will make easier to edit the document. There are many other applications of OCR, such as: image text extraction, extracting texts from scanned documents, License plate recognition and answer paper checker. (Patel, et al., 2012)

The main concept of this project is to implement OCR in an android based calculator application. The objective of this application is to scan handwritten or printed numbers and perform mathematical calculations on it.

## Problem Domain

In this modern era of technology, people don’t want to waste time in less important tasks like performing mathematical calculation in traditional ways which are generally time consuming. For that they have calculators, but they still need manual input from the users. They need to follow some sets of guidelines and go through each processes to get correct answer. It would be much more time saving if user don’t have to manually enter the data for calculation. If the calculator gets the data with just one click, it will definitely be more efficient and time saving.

Besides, while checking the answer paper, teachers need to solve the question first by themselves and then only can correct the paper. Students are also in dilemma after solving a problem whether their answer was correct or not. It would be great help for them if they knew if their answer was correct.

## This project as solution

The main objective of this project is to develop an app that can perform mathematical calculations without even having to type to give input. It scans for a mathematical problem mentioned in any paper and gives back the answer. Since it is a mobile application, user can use their camera to scan the problem. The app then extracts the problem from paper using OCR and perform operations on it and return the result. Teachers and students can check whether their answers were right or not just by a click. Since this app knows all the mathematical rules and principles, users don’t need to bother about remembering rules like BODMAS, which they had to keep in mind if they were to perform calculation manually in traditional calculators.

# Literature Review

## Development of OCR

### First Generation OCR System

Character Recognition was first originated in early 1870s with the invention of retina scanner. The first generation OCR appeared only in the beginning of 1960s with the development of the digital computers. This generation machines could read symbols specially designed for them. The first commercialized OCR of this generation was *IBM 1418*, which was designed to read special IBM font, *407*. The recognition method was template matching, which compares the character image with a library of prototype images for each character of each font. (Shodh Ganga, 2015)

### Second Generation OCR System

This generation machines were able to recognize machine printed as well as hand written characters. But the character set was limited to numerals and a few letters and symbols. Such machines appeared in between middle of 1960s to early 1970s. (Shodh Ganga, 2015)

### Third Generation OCR System

This generation OCR systems mainly focused on overcoming the challenges like poor document quality, large printed and hand written character sets. Low cost and high performance were also important concerns. (Shodh Ganga, 2015)

### Fourth Generation OCR System

This generation system focuses on complex documents which contain texts, graphics, tables, mathematical symbols, unconstrained handwritten characters, low-quality noisy documents and many more. (Shodh Ganga, 2015)

In this project we are going to use OCR to recognize handwritten or machine printed numerals and mathematical symbols. We can then perform mathematical calculations on retrieved data. There already exists some products that use OCR to do mathematical calculations. Some of them are presented below in the section *2.4.*

## Working Principles

The objective of OCR is to extract the text and convert it into editable form. For that, a document is first scanned using an optical scanner which produces an image form of the document. Now this text image is converted into editable character code such as ASCII. The basic working principle of OCR can be show as following figure.

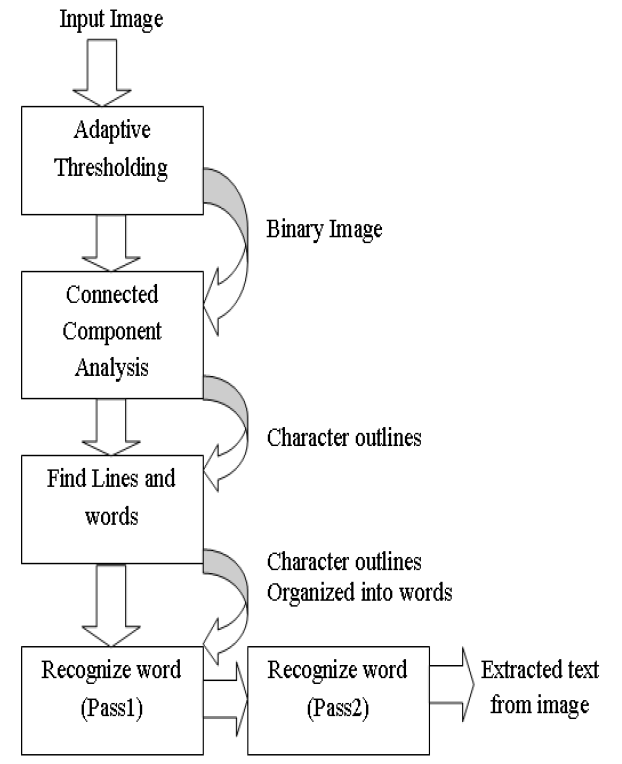


Figure 1: Architecture of Tesseract OCR (Patel & Patel, 2012)

## Handwritten Character Recognition

Tesseract performs well for printed numbers with the detection rate more than 85% for the fonts within its database. But for the handwritten numbers it drops to about 50%. The main reason for this difference in result is due to the variation in the size of numbers that are handwritten and also due to the lack of matching fonts in its database.

To overcome this problem a machine learning algorithm based on Support Vector Machines (SVM) can be applied. This algorithm analyzes data and recognize patterns. This algorithm first converts character images into vector form. After using line segmentation region labels are used to determine the bounding box for each individual characters. A small amount of padding is added to the border, as shown in Figure 2. The segmented character is now downsampled to 32x32 pixels and then divided into 64 4x4 regions. The count in each region is the determined vector value, as shown in Figure 3. This conversion thus results in a 64 dimensional vector for each character image. (Sikka & Wu, 2012)

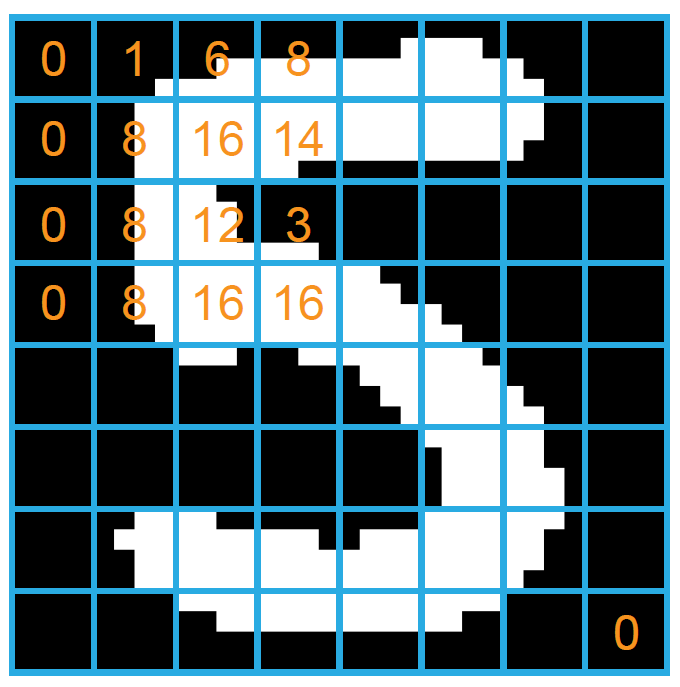


Figure 2: Segmented Character from input image

Figure 3: Downsampled Character

## Similar Systems

### Photo Math

Photomath is one of the best math problem solver application. It uses phone’s camera to capture a picture of the math problems. Then the picture is scanned by the application. The application uses advanced OCR technology in order to recognize both, handwritten and printed characters. The recognized characters are then processed through Photomath’s own algorithm that examines every character and determines the formula for the scanned problem. Finally, a problem solving algorithm is applied to the formula and the solution is provided with every solving steps. (Photomath, 2020)

Figure 4: PhotoMath Application

### Math Solver

Microsoft Math Solver can not only solve simple mathematical calculations, it can solve various math problems like quadratic equations, calculus and statistics. The application can also show graphs of the equations.

We can either type our problem query using a scientific calculator in the application or draw it on the phone’s screen. But most importantly we can just use our phone’s camera to scan the problem on our books or on the copy written by us. (Microsoft, 2019)

Figure 5: Math Solver Application

### Mathway

Mathway is little bit different system than the other two. It works by letting the user to choose the field of mathematics of which the problem is to be solved and then allows user to input the problem by either typing or scanning. The problem is then processed and provides the result in conversational style like the chat bot does. (Mathway, 2020)

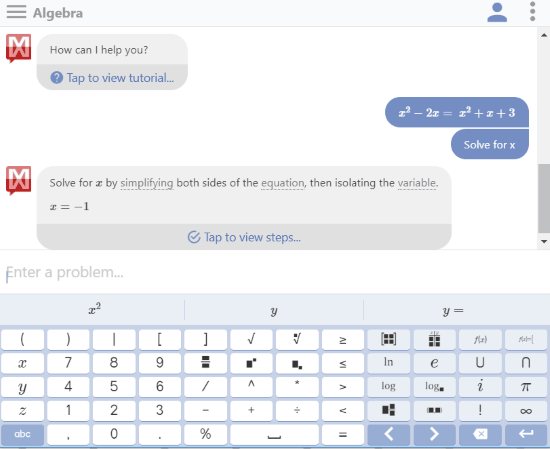
This system implements Lexical analysis in order to solve the problem. It first breaks the problem into tokens. In an expression 1+2, the tokens are 1, + and 2. Then the tokens are fed into the parser, which has the knowledge about relationships between tokens and can call the appropriate function which in this case is add.

Figure 6: Mathway Application

## Review of Similar Systems

All of the above mentioned systems have their separate math content team. Because of this there is solution to every math problem from arithmetic to calculus. These systems read and solve mathematical problems by just using the camera of mobile phones. The most astonishing feature of these systems is that they provide step-by-step solutions too. We can even choose multiple explanation methods for same problems. Moreover, they also provide animated calculation steps.

The only bad aspect or the limitation of these systems is that they support only English language. They can only perform calculations on English numerals. This is the aspect where my project is going to work on. My smart calculator will be able to perform calculations on Nepali numerals too.

## Why Tesseract OCR?

Some of the reasons to use Tesseract OCR are as follows:

* It is platform independent
* Supports multiple languages (Google Open Source, 2020)
* High accuracy
* Open source
* Ease of access and use
* Tesseract has the font accuracy in the range of 85-90%

# Artefact

## FDD

## SRS Legend

Sub Systems:

MCS : Mathematical Calculation System

CRS : Character Recognition System

HCS : Handwritten Character Recognition System

IPS : Image Processing System

MCS - F - 1.0

Types of Requirements:

F : Functional Requirements

NF : Non-Functional Requirements

UR : Usability Requirements

Numbering

**Legend**

## SRS Table

|  |  |
| --- | --- |
| **Requirement Code** | **Requirement Description** |
| MCSF **1.0** | The system should use the built in ALU to perform arithmetic calculations. |
| MCSNF 1.1 | The system should provide the result in less than 1.5 seconds. |
| MCSNF 1.2 | The system should follow BODMAS rule for calculations. |
| MCSUR 1.1 | The system should have “All Clear” button to reset the entry. |
| MCSUR 1.2 | The system should have “Clear Entry” button to erase the latest entry. |
| MCSUR 1.3 | The system should display result in bigger font than the input. |
|  |  |
| MCSF **2.0** | The system should be able to perform scientific calculations. |
| MCSNF 2.1 | The system should be able to point out the error if any occurs. |
| MCSNF 2.2 | The scientific notations should be placed separately. |
|  |  |
| MCSF **3.0** | The system should keep records of the calculations history. |
| MCSNF 3.1 | The system should display maximum five history records. |
| MCSNF 3.2 | The system should have clear history option. |
| MCSUR 3.1 | The system should have black background with white text color. |
|  |  |
| CRSF **1.0** | The system should recognize numbers as well as mathematical notations. |
| CRSUR 1.1 | The notations should be clear and familiar to the users. |
| CRSF **2.0** | The system should recognize characters with noises as well. |
| CRSNF 2.1 | The system should recognize the characters in less than 2 seconds. |
|  |  |
| HCSF **1.0** | The system should recognize hand written characters. |
| HCSNF 1.1 | The system must have handwritten accuracy over 80%. |
| HCSUR 1.1 |  |
|  |  |
| IPSF **1.0** | The system should be able to capture image to perform calculations. |
| IPSNF 1.1 | The system should use mobile camera to capture the image. |
| IPSNF 1.2 | The system should capture the in 720\*720 resolution. |
| IPSNF 1.3 | After capturing the image there should be crop option. |
| IPSNF 1.4 | There should be save option for image. |
| IPSUR 1.1 | There should be camera icon to switch between standard and scanning calculator. |
|  |  |
| IPSF **2.0** | The system should be able to perform calculations on images from gallery. |
| IPSNF 2.1 | While importing, the images should be shown from latest to old. |
| IPSNF 2.2 | While importing only one image should be selectable. |
| IPSUR 2.1 | The selected image should be faded. |
|  |  |

## System Modeling

### Context Modeling

#### Use Case Diagram

* Use Case Diagram for Basic Calculation System

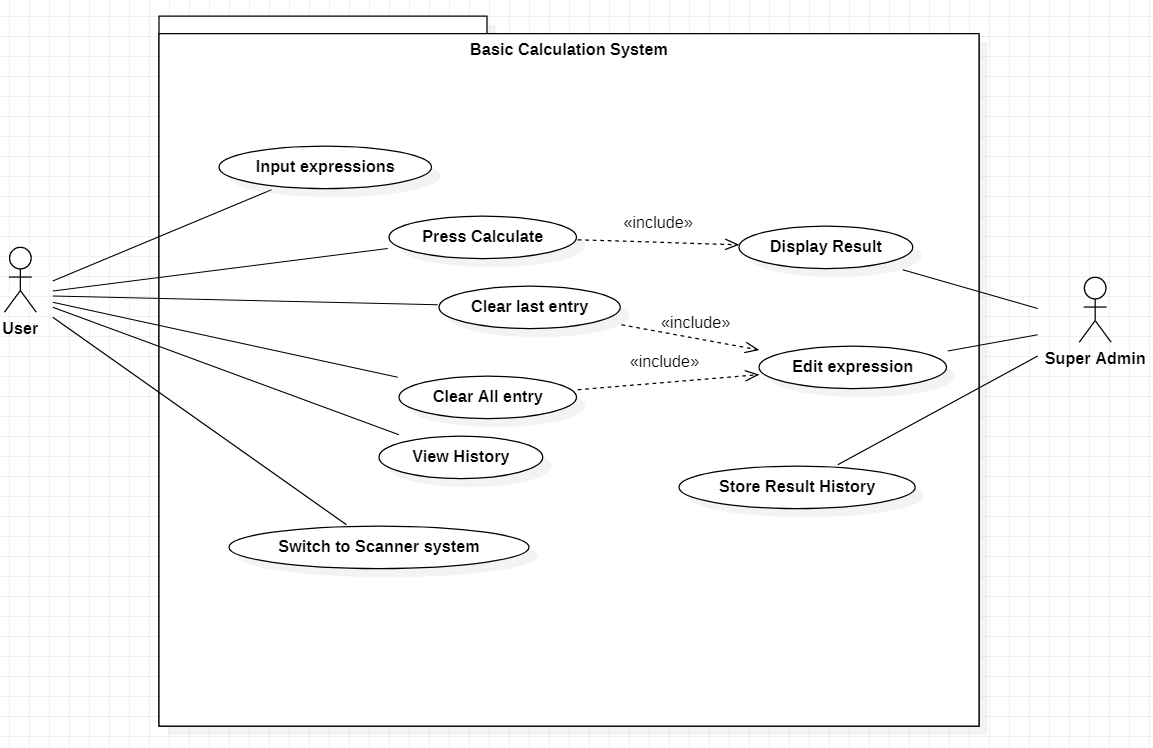


Figure 7: Use Case Diagram for Basic Calculation System

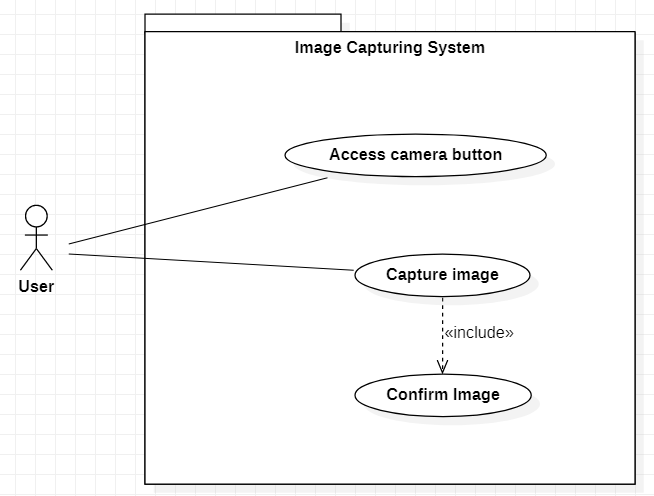
* Use Case Diagram for Image Capturing System

Figure 8: Use Case Diagram for Image Capturing System

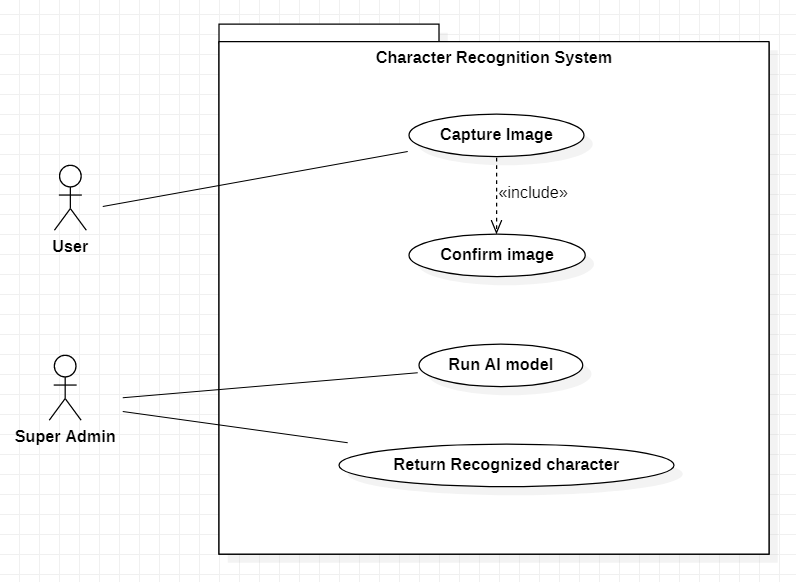
* Use Case Diagram for Character Recognition System

Figure 9: Use Case Diagram for Character Recognition System

### Structural Modeling

#### Class Diagram

Figure 10: Class Diagram

### Process Modeling

#### Context Diagram

* Context Diagram for Character Recognition System

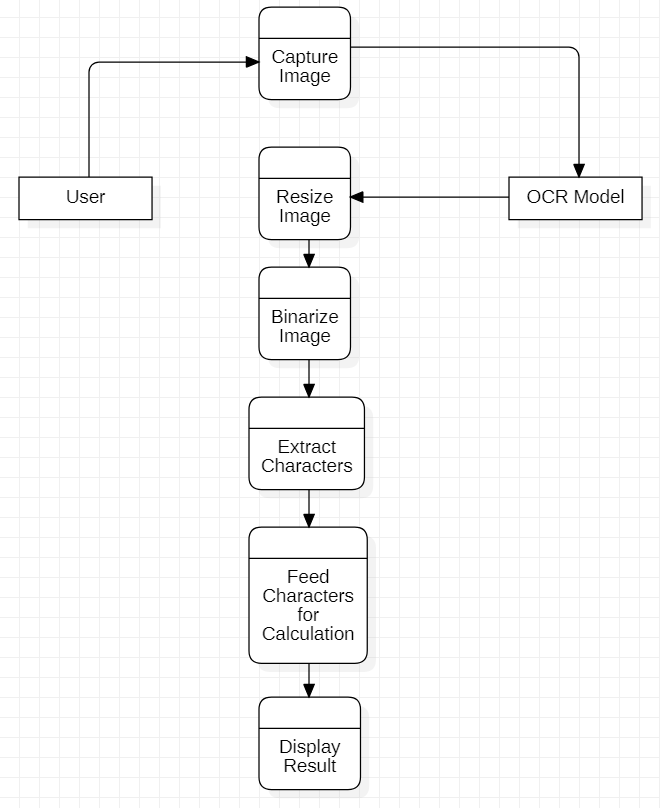


Figure : Context Diagram for Image Capturing System

* Context Diagram for Character Recognition System

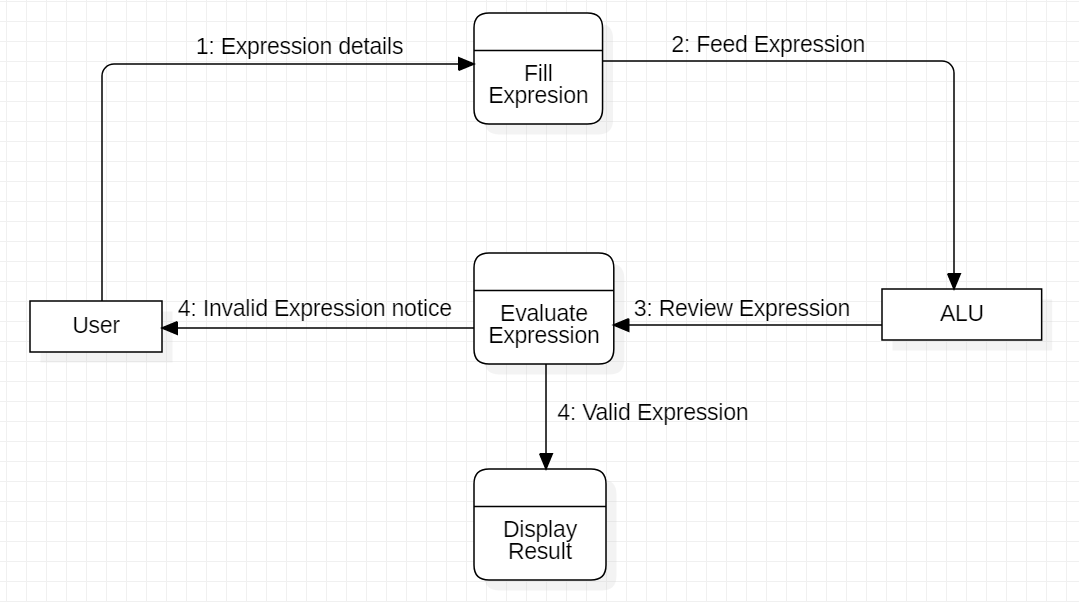


Figure 12: Context Diagram for Character Recognition System

### UI Model

#### Wireframe

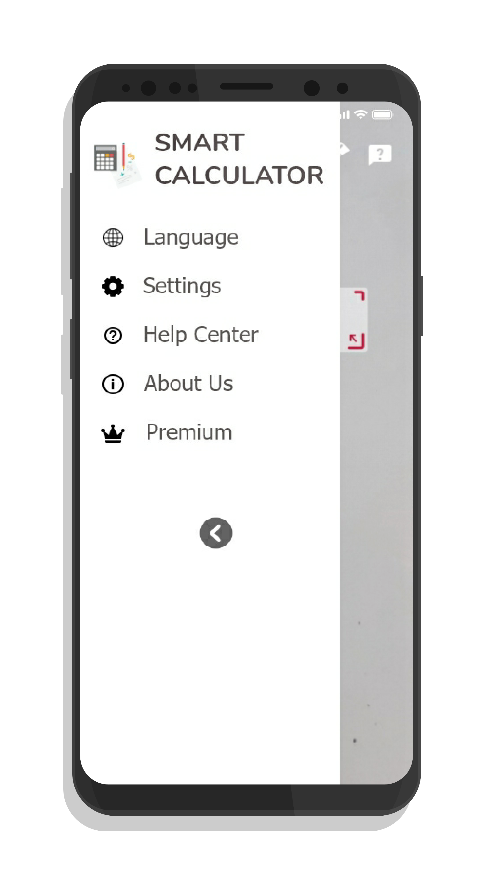
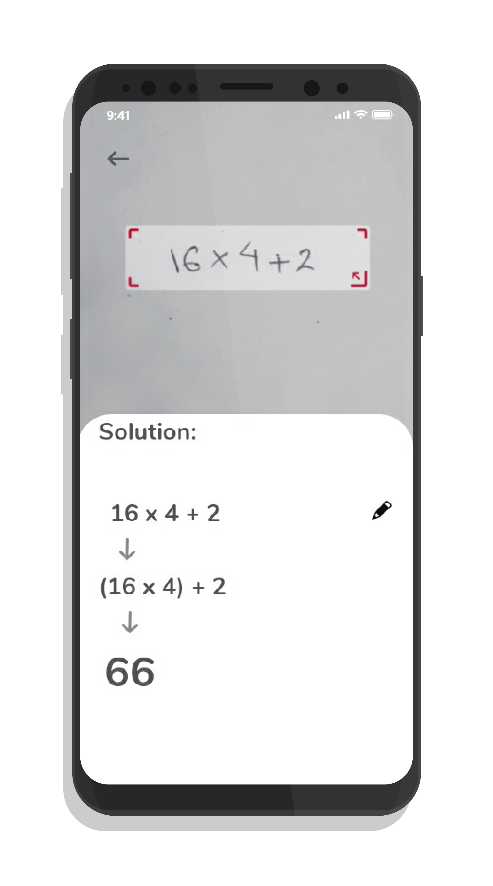
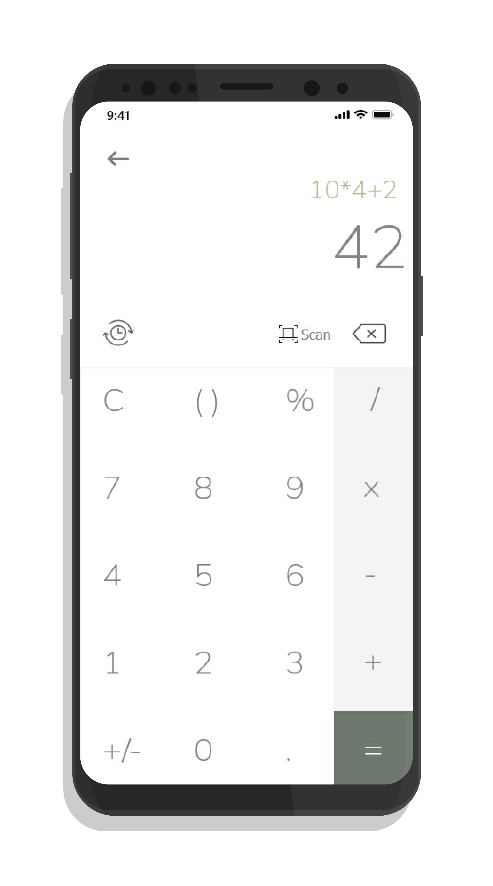
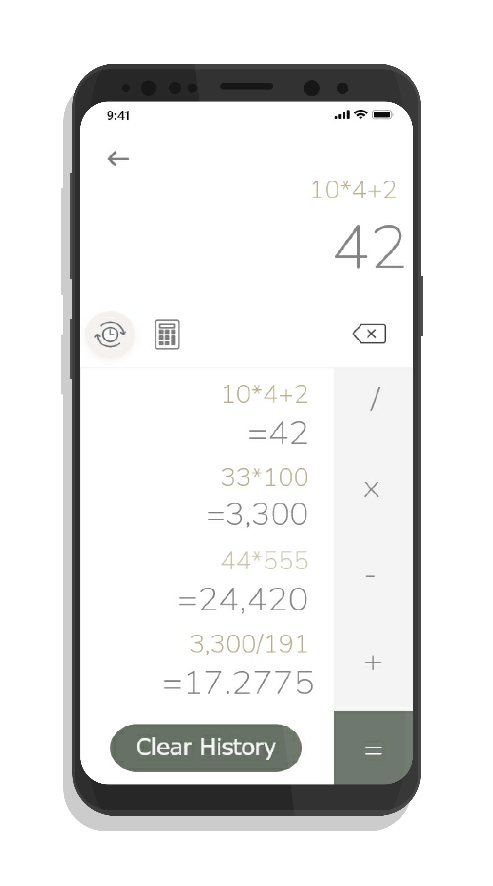


Figure 13: History Screen

Figure 14: Calculator Screen

Figure 15: Solution Screen

Figure 16: Navigation Screen

Figure 17: Main Screen

# References

Google Open Source, 2020. *Tesseract OCR.* [Online]   
Available at: https://opensource.google/projects/tesseract  
[Accessed 15 May 2022].

Mathway, 2020. *Mathway.* [Online]   
Available at: https://www.mathway.com/  
[Accessed 08 October 2020].

Microsoft, 2019. *Microsoft Math Solver.* [Online]   
Available at: https://math.microsoft.com/en  
[Accessed 20 April 2020].

Patel, C. I., Patel, A. & Patel, D., 2012. Optical Character Recognition by Open source OCR Tool Tesseract. *International Journal of Computer Applications,* 55(10), pp. 50-56.

Patel, C. I. & Patel, D., 2012. Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study. *International Journal of Computer Applications,* 55(10), pp. 50-56.

Photomath, 2020. *Photomath.* [Online]   
Available at: https://photomath.net/en/help/how-does-photomath-work  
[Accessed 20 April 2020].

Shodh Ganga, 2015. *Shodhganga:a reservoir of Indian theses.* [Online]   
Available at: http://14.139.13.47:8080/jspui/handle/10603/130552  
[Accessed 20 April 2020].

Sikka, A. & Wu, B., 2012. *Camera Based Equation Solver for Android Devices,* California: Stanford University.