SRS DOCUMENT FOR OCR SOFTWARE

SUBMITTED BY

NAME	USN
S D HRUDHAY	1MS20CS098
SAFWAN G A	1MS20CS099
SANSKAR R GONDKAR	1MS20CS107

As part of the Course **Software Engineering - CS46**

SUPERVISED BY

Faculty

Dr. Mohan Kumara S

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RAMAIAH INSTITUTE OF TECHNOLOGY

(Autonomous Institute, Affiliated to VTU)

Bangalore – 560054

TABLE OF CONTENTS

1. ABSTRACT

The aim of the project Multilingual OCR is to develop OCR software for online/offline handwriting recognition. OCR is an Optical character recognition and is the mechanical or electronic translation of images of handwritten or typewritten text (usually captured by a scanner) into machine-editable text. OCR is a field of research in pattern recognition, artificial intelligence and machine vision. Handwritten recognition is used most often to describe the ability of a computer to translate human writing into text. This may take in one of the two ways, either by scanning of written text or by writing directly on peripheral input devices.

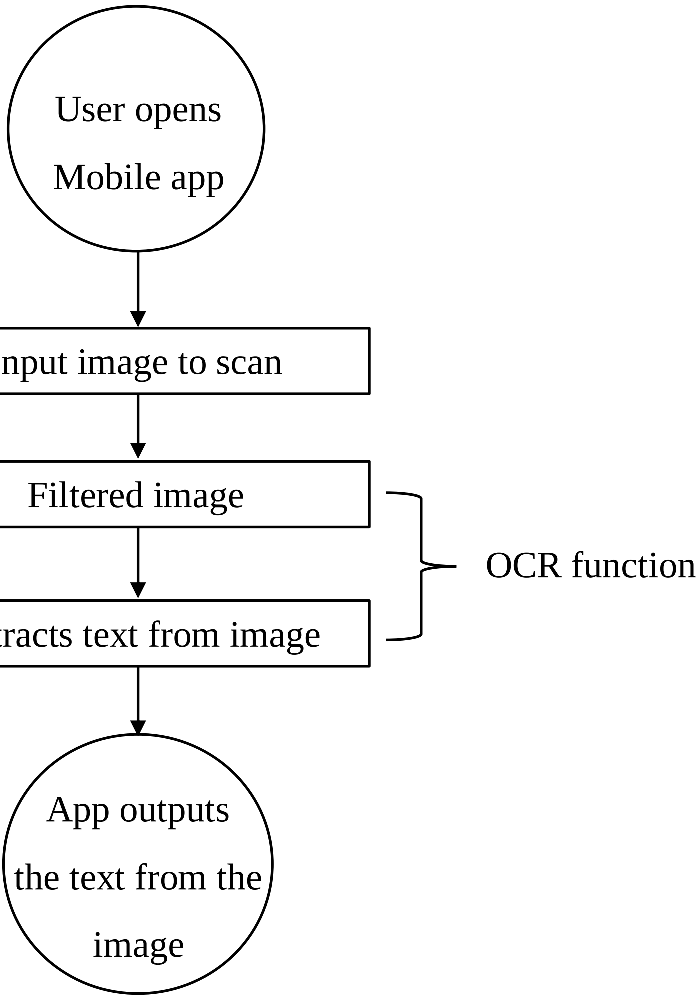


Figure 1: Basic working of application.

2. INTRODUCTION

Description:

Optical Character Recognition, or OCR, is a technology that enables us to convert different types of documents, such as scanned paper documents, PDF files or images captured by a digital camera or phone into editable and searchable data. This technology is very useful since it saves time without the need of retyping the document. It can perform the action in few minutes. It is able to recognize text in images and convert it into editable text by going throughout a simplified process as illustrated in figure 1.

Problem statement:

To implement the software which will recognize the characters from online or offline document (in image format) and use it as individual user profile.

Software Process Model:

For this particular case the best solution would be a concurrent model.

It delivers a series of releases that improve based on customer feedback over time. At the start, the core product is made and tested. If no changes need to be made with respect to functionality, then it proceeds forward. Otherwise, if the customer provided feedback, then changes are made and revised and sent to the testing stage again. Otherwise, the product is cleared for release.

The product can enter development again upon customer feedback for changes to be made, and the cycle repeats whenever necessary. At each point in time other than the starting core development, there always exists a public release, and different components of the system can be updated individually. This is because the denoised image is the only output of the pre-processing module, and its format isn't changing. So, any change in that doesn't require any change in the text recognition module, and thus any subsystem or component can be in any stage of development, without affecting the others. Thus, they can be developed **concurrently**, thus the concurrent model of software model.

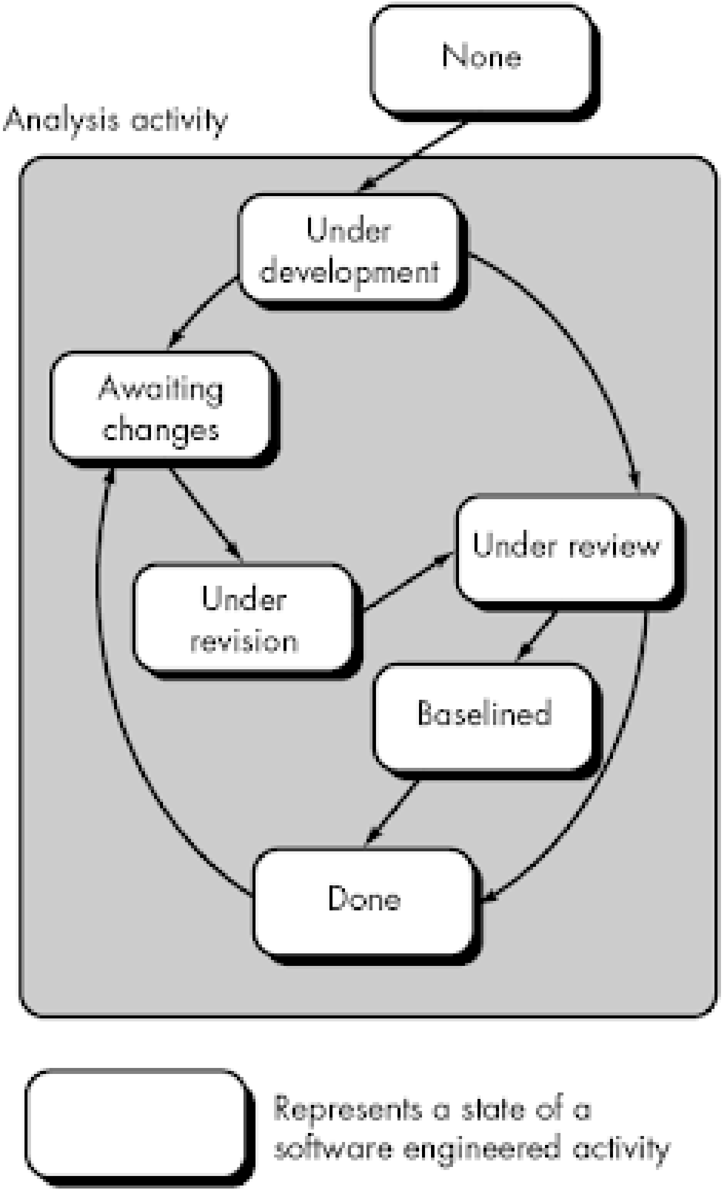


Figure 2: Illustration of concurrent model.

3. SOFTWARE REQUIREMENT SPECIFICATION

3.1 Functional requirements:

User Interface:

Must use an Android mobile phone. He/she will take a picture of the desired text or choose one from the phone's directory. The OCR will ignore the non-textual region of the picture and will print only the text. Also, the user has to follow the required steps in order to avoid any error while using the application. The application will work as follow:

a. Taking a picture using the phone's camera or choosing one from the phone's directory.

b. Recognition of the text.

c. Retrieving the text and make it editable. With a simple click, the user can take advantage of this application and perform many actions in few minutes. By using the mobile's camera, different text images can be scanned, copied, and saved.

Taking/ choosing the desired text image:

The most important thing here is the use of an Android mobile phone and its camera. The user can take a picture of a text image or choose one from the mobile's directory. The user must use a camera of typical resolution and take a picture of a text image or choose one from existing ones in his phone.

a. Android Mobile Phone.

b. Text Image.

c. Images containing text.

Recognition of the text:

The text will be recognized from the image taken by the mobile's camera or from any chosen image from the phone directory. The text will be recognized and ready to be used.

a. Recognition of the text from the image.

b. Ready to be used.

Copying the text for different uses:

Once the text is recognized and ready to be used, the user will be able to copy, edit, and modify it. He/she may also be able to retrieve the data from the image and store it directly on the phone such as the contact information taken from a Business Card. The recognized text may be retrieved to make it editable or store it directly on the phone.

a. Copy the text from the text from the image and modify it.

b. Retrieve data from the text image and store it on the phone.

3.2 Non-Functional requirements:

Performance:

The application response time shall be adequate and sufficient enough, that's why the time required for this application to response to its user's actions has to been managed and controlled. But in order to maintain the performance of the application, the user has to follow the required steps to get the desired result.

Portability Requirements:

The application should be compatible with different version of Android.

Maintainability:

The application shall respond to any change on the requirements.

Reliability Requirements:

The application should not have any unexpected failure. In order to avoid any failure's occurrence, the specifications have been respected and followed correctly. The only problem that may occur in some cases is that the application does not get 100% of the characters in the picture.

Security:

The security signature and certificate of the application is required.

3.3 Requirement Analysis:

3.3.1 Scenario-based elements with Use Cases:

To analyse our requirements, a use case diagram that will show the possible actors, use cases, and the relationships between the actors and the use cases and that is illustrated in figure 3.

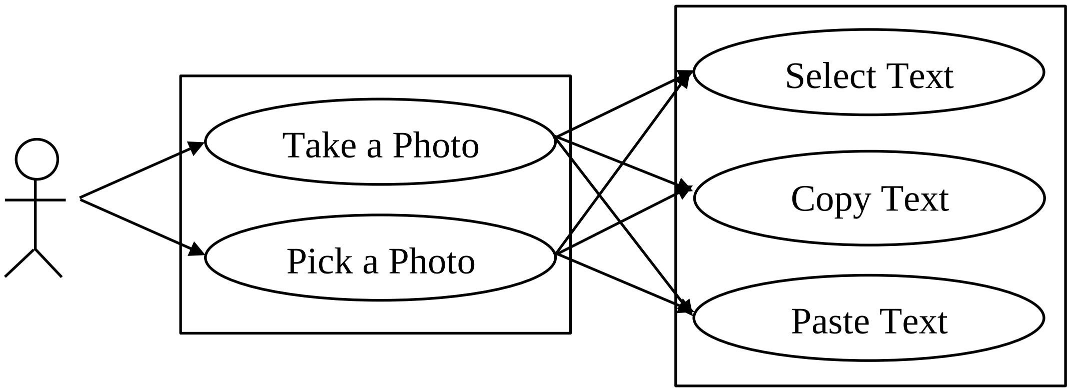


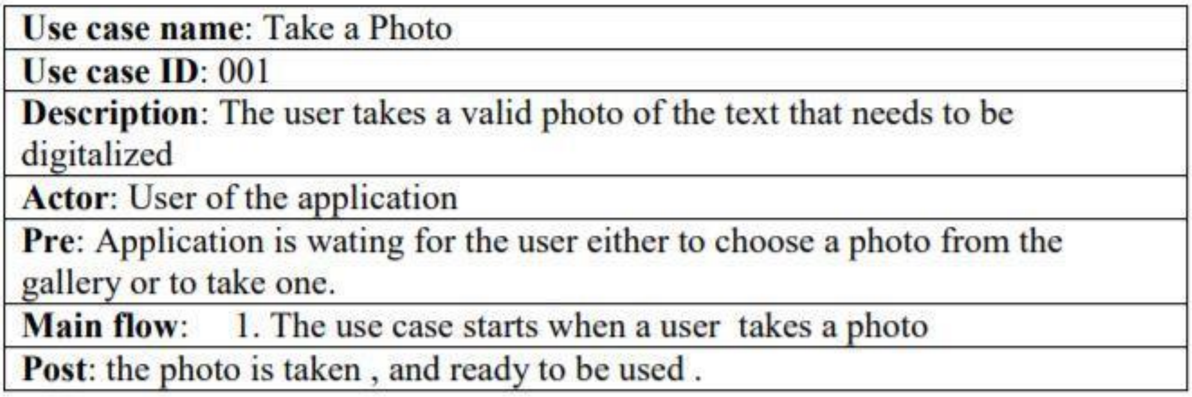
Figure 3: OCR Mobile Application Use Case Diagram

Use case Specification:

Main Actors - Public Users:

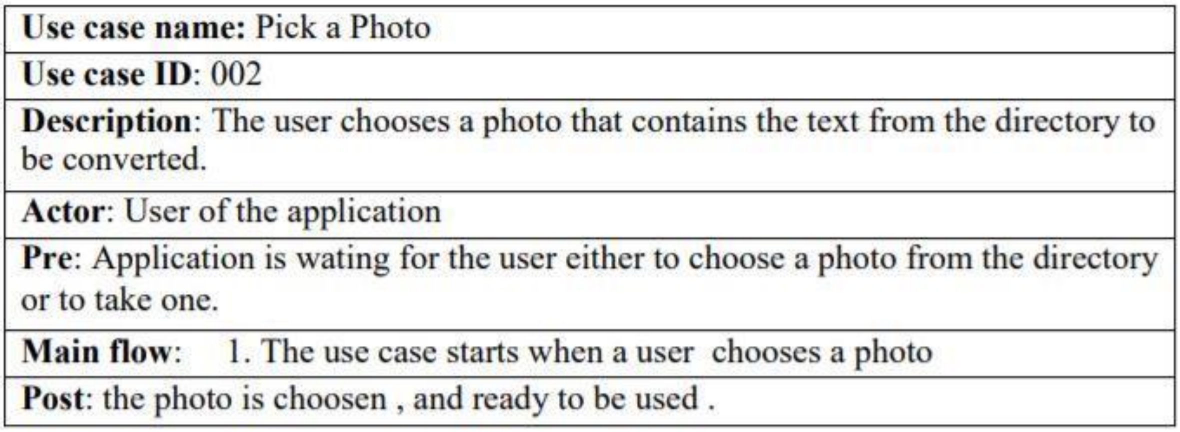
The application will be used to read texts and cards. For instance, it can be used to retrieve information from a business card and save it in the phone directly, or retrieve text from a text image taken by an Android device and store it to further reuse. Moreover, this application will also facilitate retrieving text from books in order to edit it and to convert any images such as slides into notes. Also, instead of taking notes in class, students can directly take a picture of whatever it is written on the board and convert it to text and save it. Finally, the application can also help its users such as foreigners to take a picture of any text image written in any language, retrieve it in order translate it to their native language in few minutes.

1.Scan Photo Use Case



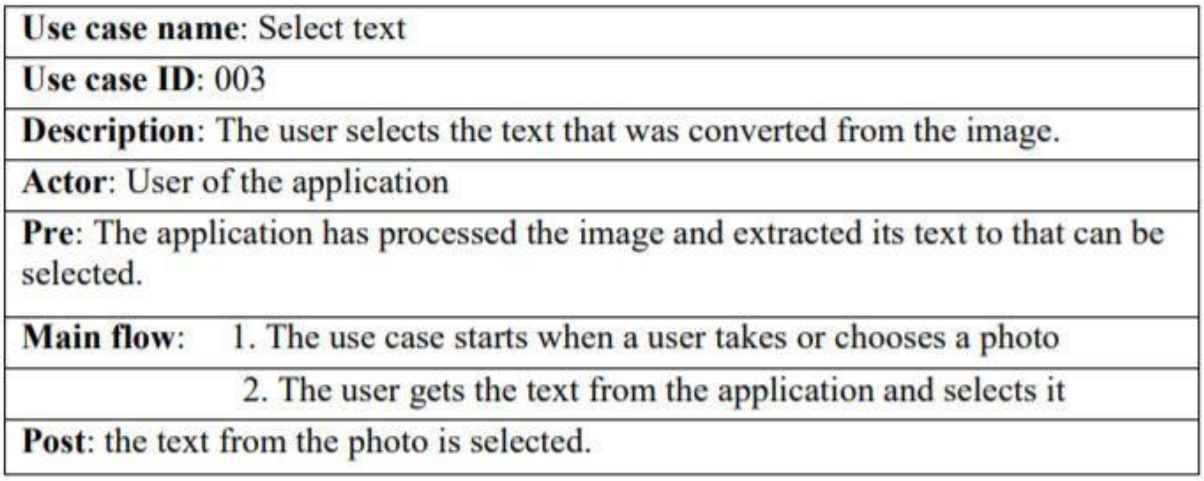
Use case name: Take a Photo
Use case ID: 001
Description: The user takes a valid photo of the text that needs to be digitalized
Actor: User of the application
Pre: Application is wating for the user either to choose a photo from the gallery or to take one.
Main flow: 1. The use case starts when a user takes a photo
Post: the photo is taken , and ready to be used .

2. Pick a Photo



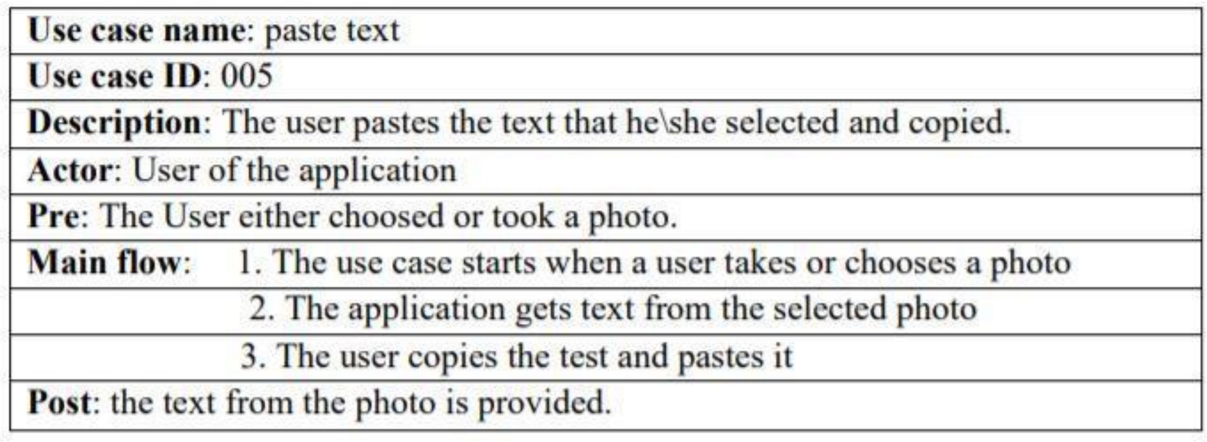
Use case name: Pick a Photo
Use case ID: 002
Description: The user chooses a photo that contains the text from the directory to be converted.
Actor: User of the application
Pre: Application is wating for the user either to choose a photo from the directory or to take one.
Main flow: 1. The use case starts when a user chooses a photo
Post: the photo is choosen , and ready to be used .

3. Select Text



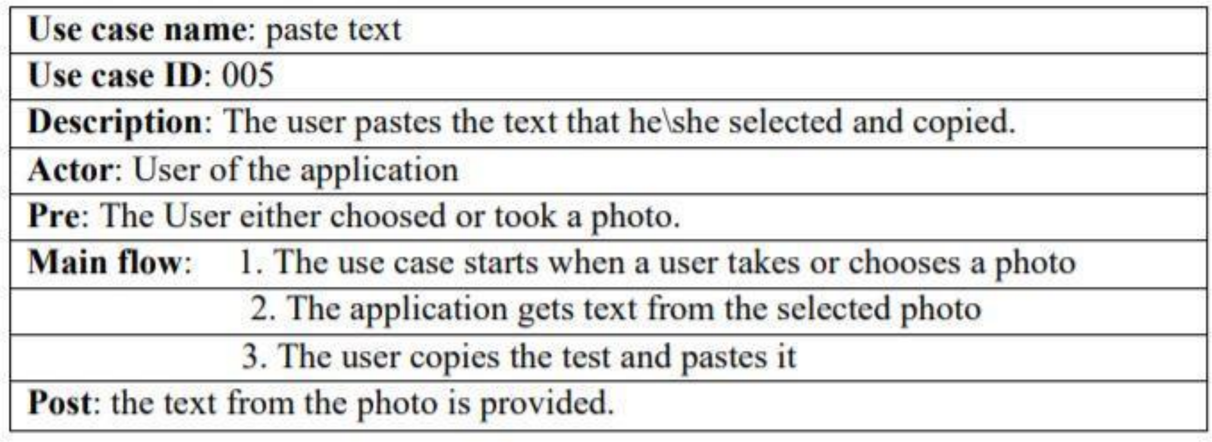
Use case name: Select text
Use case ID: 003
Description: The user selects the text that was converted from the image.
Actor: User of the application
Pre: The application has processed the image and extracted its text to that can be selected.
Main flow: 1. The use case starts when a user takes or chooses a photo
2. The user gets the text from the application and selects it
Post: the text from the photo is selected.

4.Copy Text



Use case name: paste text
Use case ID: 005
Description: The user pastes the text that he\she selected and copied.
Actor: User of the application
Pre: The User either choosed or took a photo.
Main flow: 1. The use case starts when a user takes or chooses a photo
2. The application gets text from the selected photo
3. The user copies the test and pastes it
Post: the text from the photo is provided.

5. Paste text



Use case name: paste text
Use case ID: 005
Description: The user pastes the text that he\she selected and copied.
Actor: User of the application
Pre: The User either choosed or took a photo.
Main flow: 1. The use case starts when a user takes or chooses a photo
2. The application gets text from the selected photo
3. The user copies the test and pastes it
Post: the text from the photo is provided.

3.3.2 Activity diagram:

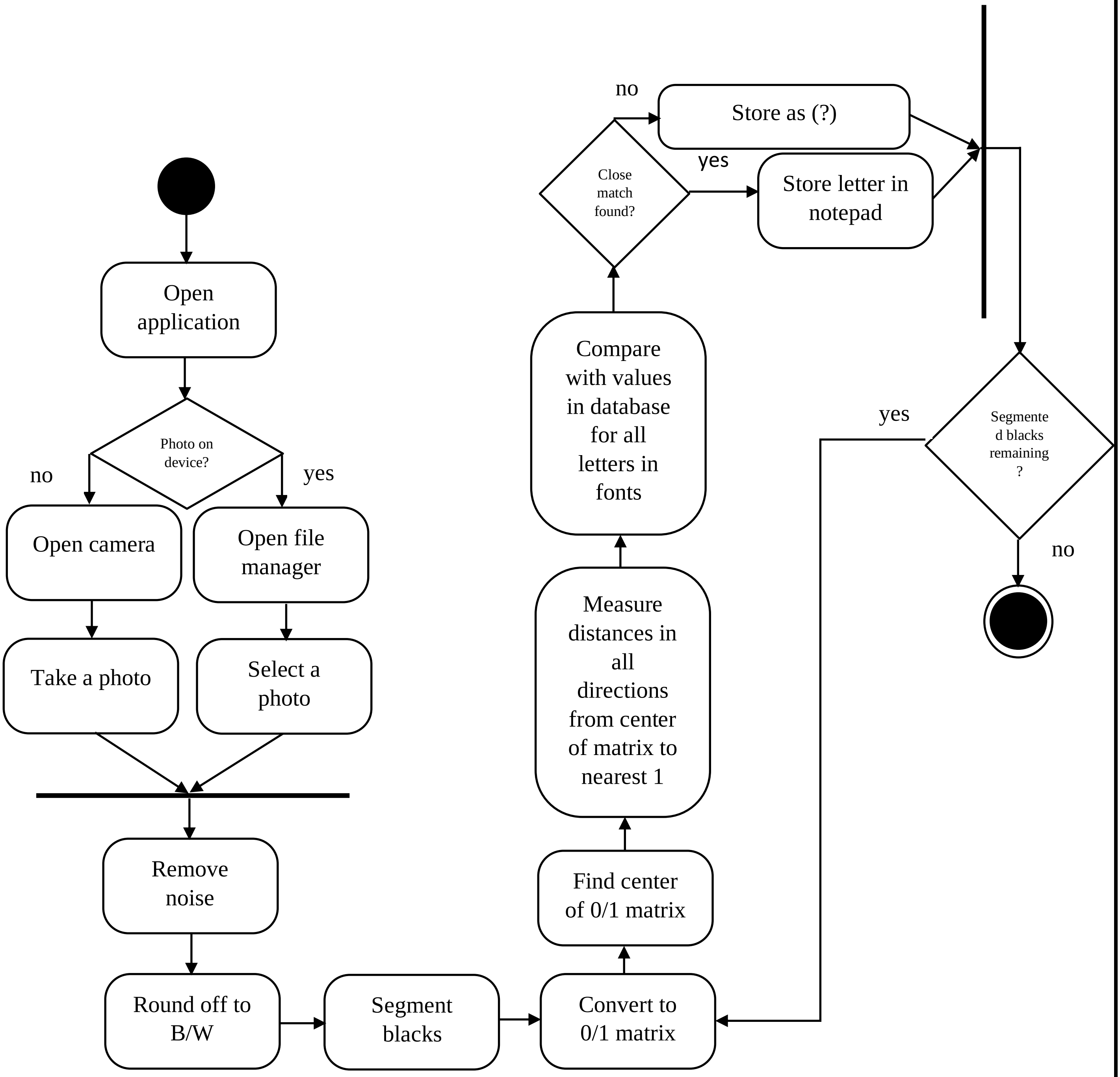


Figure 4: Activity diagram.

3.3.3 Class-Based elements:

Description of the Attributes:

1. **User**: The one that will interact with the Optical Capture Recognition (OCR).

2. **The Image**: Taken: must have a certain resolution and must contain text. Selected from the phone: must contain text.

3. **The Text**: must have words.

4. **Words**: English Alphabet.

5. Optical Capture Recognition Application.

Description of the relationships between the modules:

1. **User-OCR Application**: It is one to one relationship because only one user interacts with the application at a time.

2. **OCR Application-Image**: The application is able to process one image at a time.

3. **User-Image**: The user can either take a picture or choose one from the phone directory.

4. **Image-Text**: The image must contain text.

5. **Text-Words**: The text can contain many words.

6. **Words-Characters**: Each word can have any number of English Alphabet's characters.

7. **User-Text**: The user can copy, paste, or select the whole text or just a part of it.

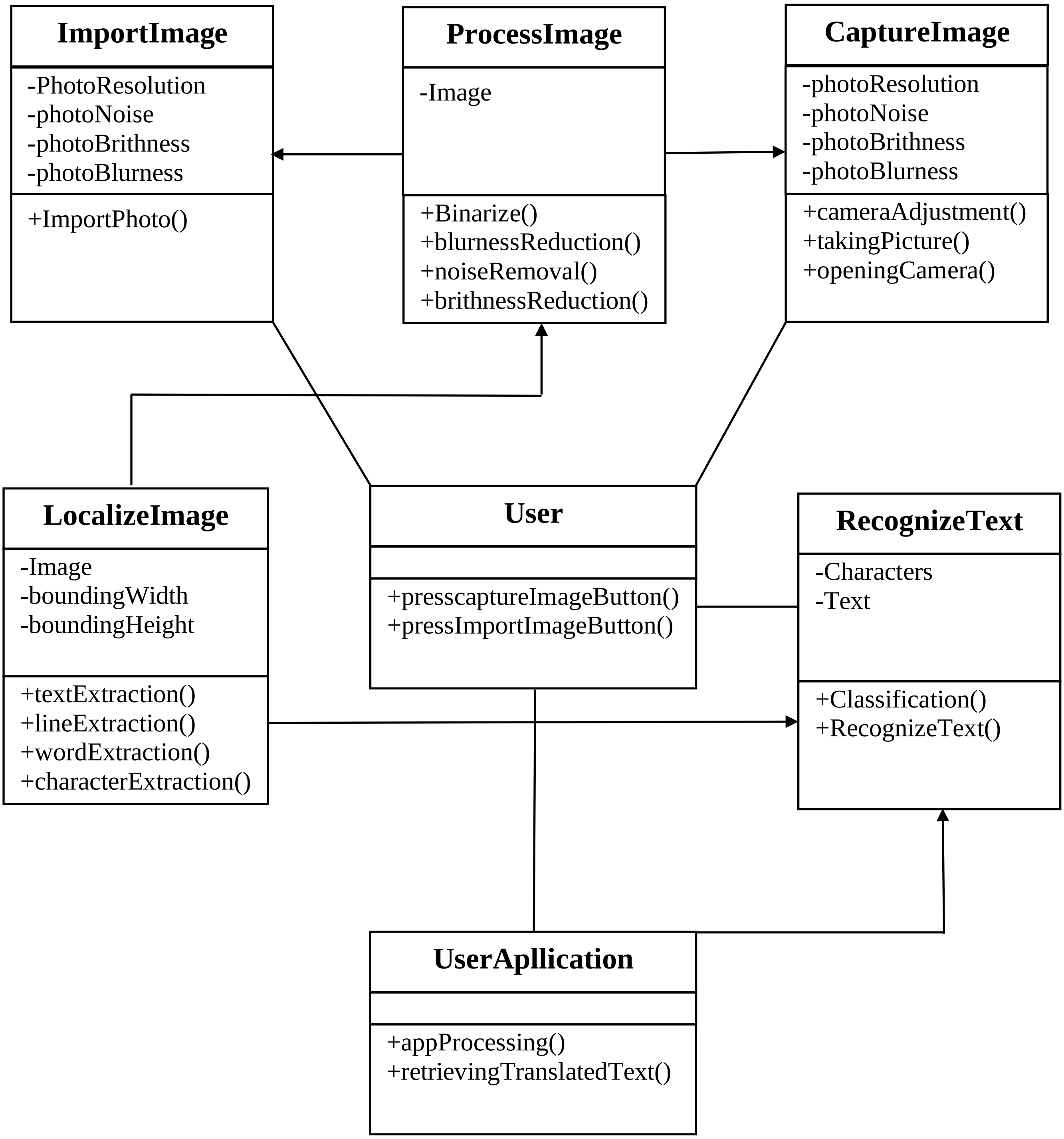


Figure 5: Class diagram

Classes Description:

User:

This class refers to the user of the application. This class contains the presscaptureImageButton() and pressImportImageButton() methods that invoke the CaptureImage and ImportImage classes respectively and that are responsible of the processes of each user's request.

CaptureImage:

This class contains PhotoResolution, PhotoNoise, PhotoBrightness, and PhotoBlurness as attributes. These attributes are related to the photo taken by the user, meaning that the photo has to be taken by a high quality device with a high resolution. Also, the class contains cameraAdjustment(), takingPicture(), openingCamera() as methods that are exclusively responsible for the capturing of the picture.

ProcessImage:

This class's attribute is the image. It is mainly responsible of the pre-processing of the image which done by these following methods: Binarize(), blurnessReduction(), noiseRemoval, and brightnessReduction(). This processImage is done in order to get the image from the CaptureImage class that meets the needed criteria.

ImportImage:

This class contains PhotoResolution, PhotoNoise, PhotoBrightness, and PhotoBlurness as attributes. These attributes are related to the photo imported or picked from the phone's directory or from the computer storage disk.. Also, the class contains ImportPhoto() as method that solely performs the import of the picture from the computer or the phone's directory.

LocalizeImage:

This class is responsible for the localizing the text in the image imported or taken by the user. It contains the image, boundingWidth, and boundingHeight as attributes, meaning that the image must be in certain range and contains text as well in order to be processed. Moreover, this class contains the textExtraction(), lineExtraction(), wordExtraction(), and characterExtraction() which are the methods that enable the localization of the text respectively. The image uses in this class is the one processed by the ProcessImage class.

RecognizeText:

The attributes of this class are characters and text. It mainly matches the characters of the LocalizeImage and classify them using the classification() method. In order words, it recognizes only Latin Alphabet which is sent to the UserApplication class by the method RecognizeText(), else it returns garbage data.

UserApplication:

This class contains the end result of these aforementioned processes of the application which is handled by the appProcessing() method, and then this retrieved text is handled by the retrievingTranslatedText() method that sends it to the user.

4. DATA FLOW

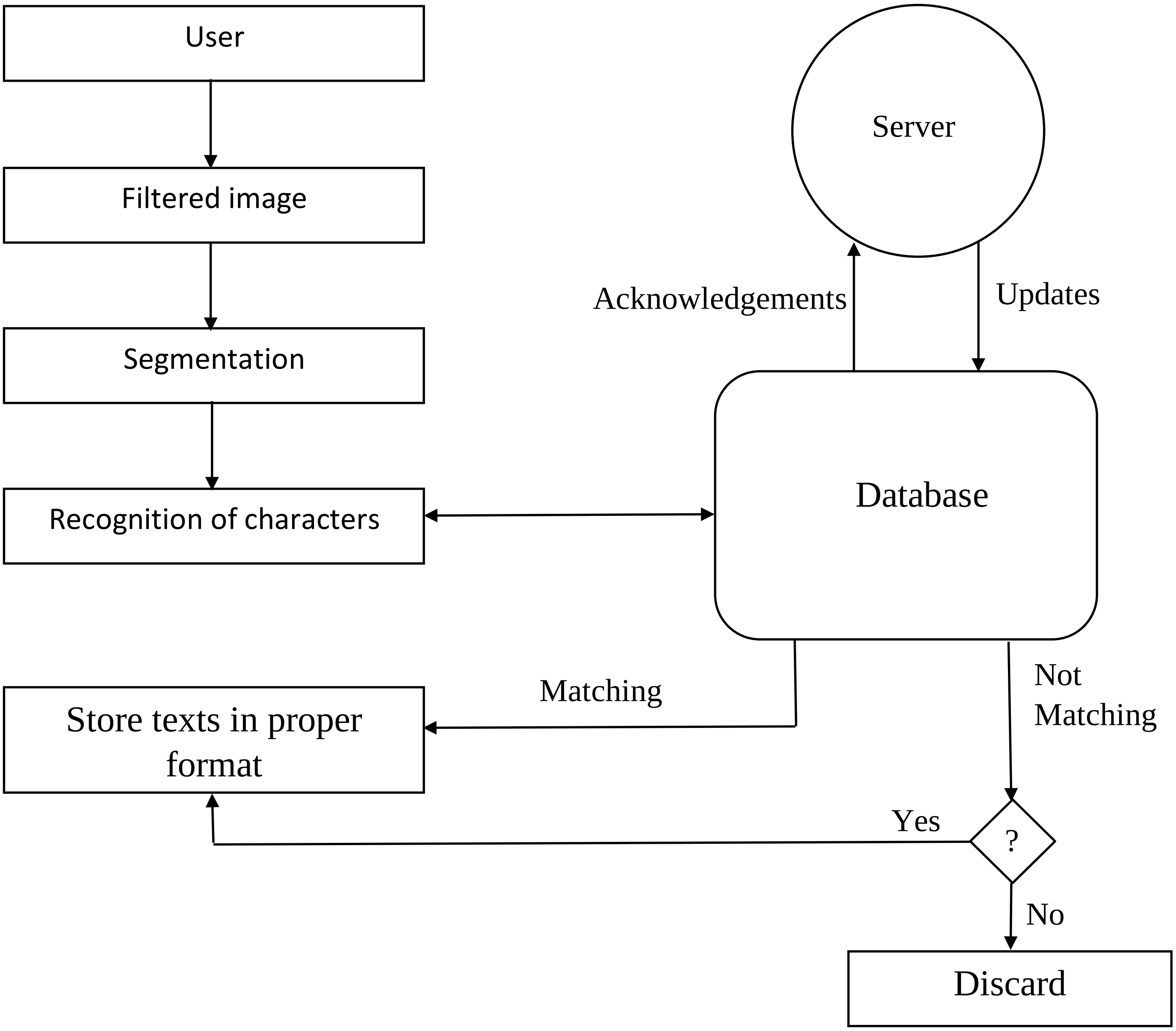


Figure 6.1: Data flow diagram.

- The user shall capture the image (input data).
- The captured image shall pass through a filter which will try to remove the noise like unwanted scratches, blurriness etc.
- The filtered image will then pass through a feature extraction module which will detect and then recognize segmented character with the help of a database.
- The database will receive updates from the server regularly. The duplicates shall be discarded after the updates respectively.
- The characters detected shall further be tallied with the database to obtain a precise text.
- After successful recognition and comparisons with the database, the text data shall be stored in a proper format to the user.

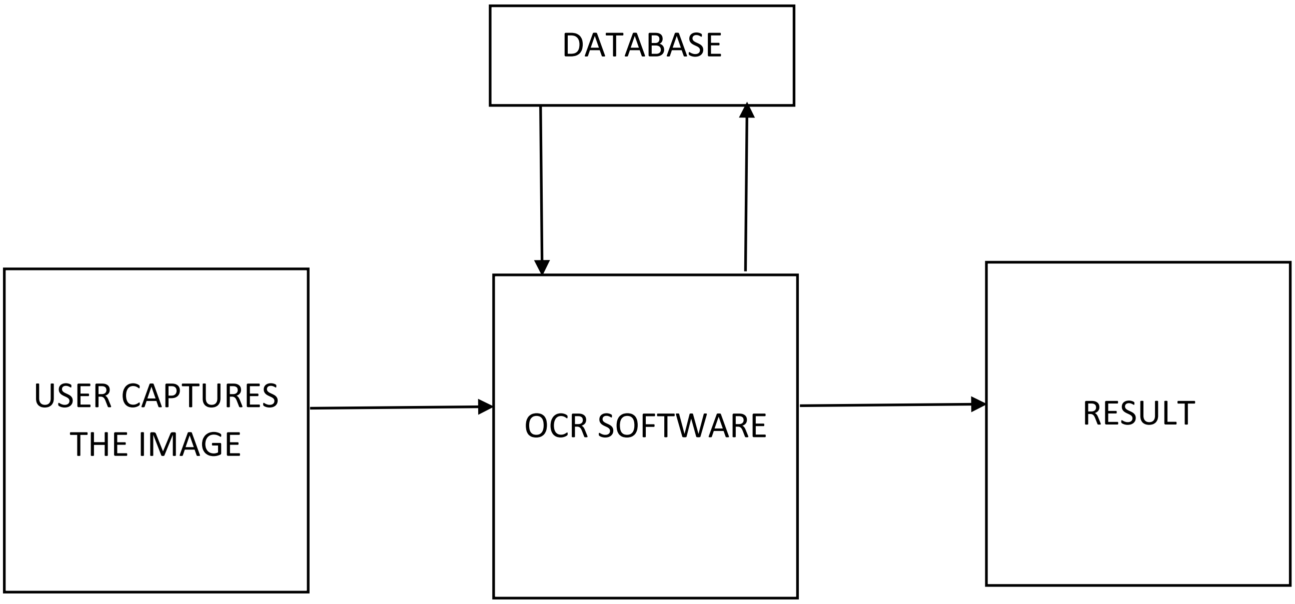


Figure 6.2: zero-level data flow diagram.