EGR 4402
Capstone Design

Fatima Ezzahra Zdadou

Final Report

Project: The Optical Capture Recognition Mobile Application

Supervisor: Dr. Falah

Fall 2015
Optical Capture Recognition Mobile Application

Capstone Report

Approved by the Supervisor

SUPERVISOR : Dr. Falah
Acknowledgments:

I would like to express my deepest gratitude and thanks to my supervisor, Dr. Falah, who suggested the idea of the Optical Capture Recognition and who gave me the needed information to start working on the project. Also, I would like to thank him for being supportive and for his guidance through this semester and for giving me the necessary advices to be able to realize this project. I am really grateful to his contribution. Moreover, I would like to thank him for his supervising methodology that made my tasks easier and motivated me through this period.

Finally, I would like to thank my parents, brothers, and all my friends for their support. They have supported me during the difficult moments and encouraged me in carrying out this project work.
# Table of Content:

Abstract: .......................................................................................................................... 5  
1. Introduction:.................................................................................................................. 6  
2. Overall Description: .................................................................................................... 8  
2.1. Product Description: ............................................................................................... 9  
2.1.1. Android Mobile Application Description: ......................................................... 10  
2.2. The Communication Description: ......................................................................... 10  
2.3. Main Actors ........................................................................................................... 10  
2.4. Constraints of the Implementation and Design: ..................................................... 11  
3. Requirement Specifications: ...................................................................................... 12  
3. 1. Interface requirements: ......................................................................................... 12  
3.2. Functional Requirements of the system: ................................................................. 13  
3.3. Non-Functional requirements: .............................................................................. 14  
3.4. Attributes of the Software: .................................................................................... 16  
3.5. Analysis: ................................................................................................................. 17  
4. Application Design: .................................................................................................... 20  
4.1. Architecture of the System: .................................................................................. 20  
4.2. Architecture Diagram: .......................................................................................... 21  
4.2.1. Description of the Attributes: .......................................................................... 22  
4.2.2. Description of the relationships between the modules: ................................... 22  
4.3. Detailed Design: .................................................................................................... 22  
4.3.1. Class Diagram: ................................................................................................. 23  
4.3.2. Classes Description: ......................................................................................... 24  
5. Software Components and Technology used: ............................................................ 26  
6. Testing: ....................................................................................................................... 27  
7. Implementation Results: ............................................................................................ 28  
8. STEEPLE Analysis: .................................................................................................... 32  
9. Future Work: .............................................................................................................. 33  
10. Conclusion: ................................................................................................................ 34  
References ..................................................................................................................... 35  
Appendix: ....................................................................................................................... 36
Abstract:

The aim of this project is to develop an Optical Capture Recognition (OCR) for Android based mobile devices. Scanned text documents, pictures stored in mobile phones having Android as operating system, and pictures taken by any Android device are the mainly focus of this application. This capstone project is divided into two parts, the desktop application that has been implemented by Mehdi Barakat, and the Android based application that I have implemented. The purpose of this application is to recognize text in scanned text documents, text images, and any picture taken by an Android based device in order to reuse it later. This application will allow its users to perform many actions in few minutes, such as copy text from these aforementioned documents and modify it, instead of wasting time on retyping it.
1. Introduction:

Mobile applications grew in less than two decades to achieve the status of the largest information repository in human history. By providing efficient, fast, consistent and authentic tools in the form of internet and mobile applications, information technology is penetrating human life and is playing an important role in changing lives of so many people around the globe.

Today many traditional industrial firms are moving towards utilizing information technology including mobile applications. Mobile applications run banking transactions, air traffic, and emergency room equipment. With this increasingly wide range of technology, both the hardware and software elements of the system can have failures which may result in catastrophic effects.

Nowadays, there is an enormous demand in storing any information available on papers, such as books or newspapers in mobile phones. There is an existing way to store information by scanning the desired text, but it will be stored as an image that won't help for further processing. For instance, if we store scanned text images, we can't read the text word by word, or line by line; the text in these scanned images can't be reused unless we rewrite the whole content by ourselves. For this reason, we need an Optical Capture Recognition (OCR).

Scanned documents are great. They let us archive stacks of paper into folder, taking up far less space and being infinitely easier to organize, move, and copy. What's not so great is finding content stored away inside one of our hundreds of scanned documents. By default, they're little more than a picture of our document—and if we want to find info inside them, we will have to open each one and read it for ourselves. Or, we could let our device do the heavy lifting for us, by turning your image into text and letting us search through our scanned
documents as easily as we search through any other documents. That's what OCR—Optical Character Recognition—does. It uses our computer's smarts to recognize letter shapes in an image or scanned document, and turn them into digital text we can copy and edit as needed.
2. Overall 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.

![Figure 1: OCR Process](image)

This process generally consists of three stages: Open (Scan) the document, Capture and recognize data and then Save in a convenient format.
2.1. Product Description:

OCR also called Optical Character Reader is a system that provides a full alphanumeric recognition of printed or handwritten characters at electronic speed by simply scanning the form. Forms containing characters images can be scanned through scanner and then recognition engine of the OCR system interpret the images and turn images of handwritten or printed characters into ASCII data (machine-readable characters).

Therefore, OCR allows users to quickly automate data capture from forms, eliminate keystrokes to reduce data entry costs and still maintain the high level of accuracy required in forms processing applications.

The technology provides a complete form processing and documents capture solution. Usually, OCR uses a modular architecture that is open, scalable and workflow controlled. It includes forms definition, scanning, image pre-processing, and recognition capabilities.

This project has two motivations, a desktop application and Android based mobile application.

Scanned documents, pictures stored in Android devices, and pictures taken by an Android device containing a text are scanned by this OCR. This application will allow its users to retrieve text from these documents to make it editable or to reuse it to perform other actions. For example, people can scan any ID Card, and store its data in their phones directly, or scan books or any important article found in a newspaper and retrieve the needed information and store them in their devices in few minutes.
2.1.1. Android Mobile Application Description:

Android mobile OCR is a software development kit (SDK) based on OCR technologies. Its powerful image processing algorithms enable mobile devices to perform highly accurate text recognition. Optimized for efficiency and featuring comprehensive language support, the application is ideal for wide range of individuals and users of android mobile phones who seek to convert their scanned paper documents, PDF files or images captured by their android mobile phones into editable and searchable data.

Using the code OCR technology, highly accurate text recognition functionality can be included in applications for tablets, Smartphones and other mobile devices using android. Mobile devices using iOS is out of our scope and can be done as a future work.

For the Android, it needs to use the procedures of the Android operating system. In other words, in order to have some tasks done, the system’s implementation uses some libraries, so the application will interact with these libraries as well.

2.2. The Communication Description:

This application doesn't require any network connection. It will work as a device in-house application. This is very useful since sometimes people need to extract data from documents while internet is not working, so this application will allow them to so.

2.3. Main Actors
2.3.1. 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 to translate it to their native language in few minutes.

2.4. Constraints of the Implementation and Design:

The application has been developed for Android based devices with high camera's resolution. Also, the printed text without skew is the mainly focus of the application because of time constraint, but it can be added later on.
As aforementioned, we need to retrieve text from scanned documents or any text image and make it editable to reuse it and read it word by word. For instance, there are plenty of books that are only available on printed format, so even if we scan them, they will be stored only as images. With the use of the Optical Capture Recognition (OCR), these scanned documents will be available for later editing and can be reused by the user.

After the feasibility study and a deep understanding of the functionalities of the project, and after looking at the tools that will be needed to develop and realize the application, My teammate and I have been able to collect the main user classes and the different requirements and classify them as follows:

3. 1. Interface requirements:

User: 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:

- Taking a picture using the phone's camera or choosing one from the phone's directory.
- Recognition of the text.
- 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.

3.2. Functional Requirements of the system:

We have classified these functional requirements as follow:

1. Taking/choosing the desired text image.
2. Recognition of the text.
3. Copying the text for different uses.

3.2.1. Taking/choosing the desired text image:

➢ **Description:**

- **For the mobile application:**

  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.

- 1: Android Mobile Phone.
- 2: Text Image.
- 3: Images containing text.

3.2.2. Recognition of the text:

➢ **Description:**

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.
  - 1: Recognition of the text from the image.
  - 2: Ready to be used.

3.2.3. Copying the text for different uses:

- Description:
  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.
  - 1: Copy the text from the text from the image and modify it.
  - 2: Retrieve data from the text image and store it on the phone.

3.3. Non-Functional requirements:

After the functional requirements, My teammate and I have been able to classify the non-functional requirements as follows:

3.3.1. Product Requirements:

3.3.1.1. Usability Requirements:

The application shall be used friendly and doesn’t require any guidance to be used. In other words, the application has to be as simple as possible, so its users shall use it easily. Actually, the interface is quite simple and straightforward so that anyone can
understand it. The user should only click on the take or pick picture button and then directly click on the button extract text without any complications.

3.3.1.2. 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 do not get 100% of the characters in the picture.

3.3.1.3. Efficiency Requirements:

3.3.1.3.1. 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.

3.3.1.3.2. Portability Requirements:

The application should be compatible with different version of Android, so if the version of Android is upgraded, the application should be upgraded as well.

3.3.2. Organizational Requirements:

3.3.2.1. Delivery Requirements:

I agreed with my client, to deliver the mobile based application by the beginning of December, and exactly on December 4th, 2015.

3.3.2.2. Implementation Requirements:
I used Java as a programming language for the implementation of the project.

3.3.2.3. Standards Requirements:

The application shall follow the AUI standard form.

3.3.3. External Requirements:

3.3.3.1. Ethical Requirements:

This application should protect the confidentiality of the user's personal information and any personal data stored on his/her mobile phone.

3.3.3.2. Legislative Requirements:

3.3.3.2.1. Security:

The security signature and certificate of the application is required as in any mobile application.

3.3.3.2.2. Privacy:

The application shall protect the user's data and make sure to keep it confidential. The device can be protected by a pin code or fingerprints in order to ensure the privacy.

3.4. Attributes of the Software:

3.4.1. Maintainability:

The application shall respond to any change on the requirements.

3.4.2. Adaptability:
The application shall be compatible to any Android OS version.

**3.4.3. Availability:**

The application shall be available on the store whenever users want to download it.

**3.4.4. Flexibility:**

The architecture shall be flexible to any change of the requirements.

**3.5. Analysis:**

To analyze our requirements, my teammate and I used 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 2

**3.5.1. Use case Diagram:**

![Use case Diagram](image-url)

*Figure 2: OCR Mobile Application Use Case Diagram*
3.5.2. Use case Specification:

1. Scan Photo Use Case

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

2. Pick a Photo

<table>
<thead>
<tr>
<th>Use case name:</th>
<th>Pick a Photo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use case ID:</td>
<td>002</td>
</tr>
<tr>
<td>Description:</td>
<td>The user chooses a photo that contains the text from the directory to be converted.</td>
</tr>
<tr>
<td>Actor:</td>
<td>User of the application</td>
</tr>
<tr>
<td>Pre:</td>
<td>Application is waiting for the user either to choose a photo from the directory or to take one.</td>
</tr>
<tr>
<td>Main flow:</td>
<td>1. The use case starts when a user chooses a photo</td>
</tr>
<tr>
<td>Post:</td>
<td>the photo is chosen, and ready to be used.</td>
</tr>
</tbody>
</table>

3. Select text

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

4. Copy text

<table>
<thead>
<tr>
<th>Use case name:</th>
<th>Copy text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use case ID:</td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td></td>
</tr>
<tr>
<td>Actor:</td>
<td></td>
</tr>
<tr>
<td>Pre:</td>
<td></td>
</tr>
<tr>
<td>Main flow:</td>
<td></td>
</tr>
<tr>
<td>Post:</td>
<td></td>
</tr>
<tr>
<td>Use case ID: 004</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong> The user copies the text that was converted from the image.</td>
<td></td>
</tr>
<tr>
<td><strong>Actor:</strong> User of the application</td>
<td></td>
</tr>
<tr>
<td><strong>Pre:</strong> The User has selected the text or the part of the text that wants to copy</td>
<td></td>
</tr>
</tbody>
</table>
| **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  
3. The user copies the text or part of the text that he\'she needs |
| **Post:** the text from the photo is copied and ready to use. |

5. Paste text

<table>
<thead>
<tr>
<th>Use case name: paste text</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use case ID:</strong> 005</td>
</tr>
<tr>
<td><strong>Description:</strong> The user pastes the text that he'she selected and copied.</td>
</tr>
<tr>
<td><strong>Actor:</strong> User of the application</td>
</tr>
<tr>
<td><strong>Pre:</strong> The User either choosed or took a photo.</td>
</tr>
</tbody>
</table>
| **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. |
4. Application Design:

The design of the application is highly important in order to fulfill the requirements and functionalities of the project.

4.1. Architecture of the System:

At this level, we were able to identify the most important modules that we designed as follows:

The system is modeled as follows:

The interface of the system is separated from its logic by using a Model View Controller. The Model contains the data while the interface constructs the view of the architecture. The controller matches between the model and the view. All communications between the model and the view will go through the controller. Moreover, the interface doesn't follow the logic of the application which makes the architecture of the system easier.
Also, for the user class which has many functions such as pressCaptureImageBotton() and pressImportImageBotton() handles the interface's process without dealing with the application's logic.

Then, the system's controller contains handlers for the system's functionality which is composed into modules. In other words, the handler of CaptureImage is used in Capture image module, the handler of ImportImage is used in Import Picture module, and the RecognizeText is used in the UserApplication module. This controller calls the functions, but do not process them.

Finally, the application's model processes the functions of all invoked functions. This model is made of CaptureImage, ImportImage, RecognizeText, and UserApplication classes that summarize the logic of the application. These classes have different functionalities and process requests of the aforementioned view and controller, retrieve the result, and give it back to the controller.

4.2. Architecture Diagram:
4.2.1. Description of the Attributes:

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

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

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

- **Words**: English Alphabet.

- **Optical Capture Recognition Application**.

4.2.2. 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.

4.3. Detailed Design:
After a deep understanding of the requirements and functionalities of the Optical Capture Recognition (OCR) application, Mehdi Barakat and I have been able to define entities and their relationships and ended up with this following architecture diagram:

4.3.1. Class Diagram:
After the specification of the requirements of the application and their analysis, my teammate and I have been able to get our class diagram. This above diagram characterizes the classes of the application. We ended up with seven classes that hold all the information required to build and run the application. This class diagram describes the relationships among classes and shows their attributes and methods. Moreover, this class diagram has been made after several meetings with our supervisor who helped us to end up with this above final class diagram.

4.3.2. 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 preprocessing 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 is 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.
5. Software Components and Technology used:

The software components and technology used in this project are:

5.1. Technology enablers:

➢ Java as an object oriented programming language.
➢ Android SDK (Software development Kit)
➢ Android Studio.

5.2. Operating Systems:

➢ Windows
➢ Android OS

5.3. Hardware Components:

➢ Personal Computers
➢ Android based mobile phone with high resolution.
6. Testing:

Testing is an important step that helps to detect errors. Testing is a process of finding faults that might occur during the implementation phase. It is also a way to test if the product fulfills the requirements and to check the components functionalities. There exists many ways of testing where each one of them has a distinct requirement, but the only testing that we made is the acceptance testing.

I have tested the application, Android based device, with the supervisor using acceptance testing strategy.
7. Implementation Results:

The implementation results shows the end result of the project. It is an Android based device application that fulfills the requirements set by the client. These are some snapshots of the application with their description.

This above figure represents the Home page of the Optical Recognition Application.
This above figure represents the activities allowed to the user which are picking a picture from the phone's directory, or taking a picture by the phone.
This above figure shows the output of the application where the user picks a picture containing Latin alphabet.
This above picture shows the output of the application where the user picks a picture containing a text in French and it shows also the garbage data gotten from the non textual regions.
8. STEEPLE Analysis:

The STEEPLE Analysis includes seven external factors that can affect or be affected by the industry. Actually, before moving to the step of the implementation, we must know the risks and challenges that we might face later on. In other words, we must analyze the possible chances and threats to the industry as a whole.

8.1. **Economic**: Since people will be using a such technology, so instead of wasting time typing on their keyboards or writing on papers, many actions might be performed in few minutes which will contribute on the growth of the economy.

8.2. **Political**: No political trend.

8.3. **Technological**: This application is introducing a new technology that will help its users to save time and perform many actions in few minutes. For instance, students can take a picture of the slides and copy them directly instead of wasting time typing and then copying the desirable data.

8.4. **Legal**: No legal trend.

8.5. **Environmental**: Since the use of papers will reduce, so the number of waste will decrease as well.

8.6. **Social**: This application provides a social issue which is saving time and performing actions in few minutes, helping foreigners to copy and translate the desirable text just by taking a picture of whatever they need.

8.7. **Ethical**: This application should protect the confidentiality of the user's personal information and any personal data stored on the mobile phone.
9. Future Work:

During this period of the implementation of this capstone project, the Android based devices that I have implemented, I was only concentrated on the text images and documents without skew that contain Latin Alphabet due to the time constraint. I could not further extend my work to make my application process text documents in Arabic or Berber as examples.

For the future, I will try to integrate other languages such as Japanese, Chinese, Arabic, or any language that doesn't contain Latin alphabet. Also, I will concentrate also on the text with skew to ensure that the application will be efficient and employed in further uses.
10. Conclusion:

This capstone project is the biggest and most important project that I worked on during our entire academic career. First, it is not only the final step to getting my bachelor degree, but also it is a great self-capacity-check as I stepped away from what I am used to work with to a new development field. During this period, which was less than 4 months, I got to work on a project that helped me improve my capacities in the computer science field especially that it is a different and new platform using Android. It also added a lot to the knowledge that I have accumulated throughout my academic journey at Al Akhawayn University, it widened up my experience in dealing with different platforms, and enhanced my aptitudes to work with similar platforms in real life application which is going to work in our favor in future internships or jobs. Putting the technical part aside, this project had a positive impact on some of my personal traits. It upgraded our punctuality because I had to manage my time to provide the best work I can do in no later time than the deadline by sending our weekly dairies to my supervisor, Dr. Falah. It also helped me develop my team spirit and improved my attitudes regarding listening carefully to others’ opinions and respecting them as I have collaborated with each other which smoothened out the flow of this project. Moreover, having to deal with constant pressure, I learnt to be more efficient and confident with what I am doing, and I managed to prevent stress from getting over me and pushing me to lose control over myself. To conclude, this project was a great experience in my academic career that pushed me on so many levels, technically, professionally, and personally. I am quite satisfied with the outcome that I ended up with, but none of it could have been possible without the help of my supervisor, Dr. Falah, whom I would love to thank extremely.
References


Appendix:

Libraries Used:

```java
package com.zdadou.app;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;

import android.content.SharedPreferences;
import android.graphics.Bitmap;
import android.graphics.Typeface;
import android.text.SpannableString;
import android.text.Spanned;
import android.text.style.StyleSpan;
import android.util.Log;
```

The Words Order:

```java
public static int GetWordsIndex(int iPosInString) {
    int i = 0;
    int prev = 0;
    for (; i < m_asWords.length; i++) {
        if (iPosInString >= prev && iPosInString < m_asWords[i].m_iEnd) {
            break;
            prev = m_asWords[i].m_iEnd;
        }
    }
    if (i == m_asWords.length) return -1;
    return i;
}
```
Getting the picture:

```java
public static int GetWordsIndex(int iPosInString)
{
    int i=3;
    int prev = 0;
    for (; i<m_asWords.length; i++)
    {
        if (iPosInString >= prev & iPosInString < m_asWords[i].m_iEnd)
            break;
        prev = m_asWords[i].m_iEnd;
    }
    if (1 == m_asWords.length) return -1;
    return 1;
}
```

The picture's filter (Get Parts):

```java
public int bwFilterImage(byte[] bw_image, int width, int height, int bpp)
{
    final int THOLD_WHITE = 0x70;
    final int THOLD_BLACK = 0x30;
    final int BLK_SZ = width/16;
    final int RED_P = 3;
    final int GREEN_P = 6;
    final int BLUE_P = 1;
    if (bpp != 1)
    {
        int local_min = THOLD_WHITE;
        int local_max = THOLD_BLACK;
        int local = 0;
        for (int i=0; i<width * height; i++)
        {
            // Code here
        }
    }
    return 1;
}
```
Division Algorithm:

```java
public void SetImgDivisor(int d)
{
    if (d!=2 && d!=4)
        return;
    m_bImgDivisor = (byte)d;
}

public byte GetImgDivisor()
{
    return m_bImgDivisor;
}
```

Word Detection:

```java
public final class Words
{
    String m_sBody;
    int m_iConfidence;
    boolean m_bIsValidWords;
    boolean m_bNewLine;
    int m_iEnd;

    public Words()
    {
        m_sBody = "";
        m_iConfidence = 0;
        m_bIsValidWords = false;
        m_bNewLine = false;
        m_iEnd = 0;
    }

    public Words(String body, int conf)
    {
        if (!mConfig.m_bReplaceUnknownChars)
            m_sBody = body;
        else
        {
            m_sBody = body.replaceAll("[^a-z\s0-9!?,;\:\"\'\-]+", "");
            m_sBody = m_sBody.replaceAll("[^A-Za-z0-9!?,\.;:\"\'\-]+", "");
        }
        m_iConfidence = conf;

        if (m_sBody.contains("\n"))
            m_bIsValidWords = false;
        else m_bIsValidWords = MyIsValidDictionaryWords(body);
        m_bNewLine = body.contains("\n");
        Log.v(TAG, "Words("+m_sBody+","+conf+")");
    }
}
public static int GetWordsIndex(int iPosInString)
{
    int i = 0;
    int prev = 0;
    for (; i < m_asWordsss.length; i++)
    {
        if (iPosInString >= prev && iPosInString < m_asWordsss[i].m_iEnd)
            break;
        prev = m_asWordsss[i].m_iEnd;
    }

    if (i == m_asWordsss.length) return -1;
    return i;
}
public final static class OCRConfiguration
{
    public int m_iMinOveralConfidence;
    public int m_iMinWordsConfidence;
    public int m_iPSMMode = PM_Automatic;
    public boolean m_bUseBWFilter;
    public boolean m_bShowValidWordsOnly;
    public boolean m_bReplaceUnknownChars;

    private byte m_bImgDivisor = 2;
    private String m_sLanguage = "eng";
    public String m_asLanguages[] = null;

    public OCRConfiguration()
    {
        LoadFabricDefaults();
    }

    public void LoadFabricDefaults()
    {
        m_iMinOveralConfidence = 60;
        m_iMinWordsConfidence = 50;
        m_iPSMMode = OCR.PM_Automatic;
        SetImgDivisor(2);
        m_bUseBWFilter = false;
        m_bShowValidWordsOnly = false;
        m_bReplaceUnknownChars = true;
        m_sLanguage = "eng";
        Log.v(TAG, "LoadFabricDefaults(): " + m_sLanguage);
    }
}
Initialize and destruct OCR:

```java
private OCR()
{
    String text = libVer();
    classInitNative();
    initializeNativeDataNative();

    mConfig.n_asLanguages = getLanguagesNative();

    Log.v(TAG, "OCR Initialize() done - libver " + text + " no-langs-installed=" + mConfig.n_asLanguages.length);
}

public static void Initialize()
{
    if (m_OCInstance == null)
    {
        m_OCInstance = new OCR();
    }
}

public static OCR get()
{
    return m_OCInstance;
}

public void Destructor()
{
    Log.v(TAG, "OCR - finalize");
    if (m_bIsLibraryInitialized)
    {
        ClearOCR();
        m_bIsLibraryInitialized = false;
    }
}```