**PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA**
**Ministry of Higher Education and Scientific Research**

**UNIVERSITY ECHAHID HAMMA LAKHDAR - EL OUED**
**FACULTY OF EXACT SCIENCES**
**Computer Science Department**

**End Of Study Memory**
**Presented For The Diploma Of**

# ACADEMIC LICENCE

Domain : **Mathematics and Computer Science**
Faculty : **Exact Sciences**
Specialty : **Information Systems**

Presented by :

- **Bennour Mohammed Ramzi**
- **Laggoun Ahmed**
- **Lassel Hadjer**

*Theme*

# Reading Aid Application For Elders

Proposed and Supervised by : Miss. Chourouk Guettas

Sustained on xx-xx- 2021 In front of the jury:

| | | | |
|---|---|---|---|
| M. | …… | MCA | Président |
| M. | …… | MAA | Rapporteur |

**Academic year:** 2020-2021

# ACKNOWLEDGEMENTS

**W**e are honored to write these lines to express our thanks and gratitude to anyone who helped us directly or indirectly from afar or from near.

**F**irst, our greetings and thanks to our fathers and our mothers who were with us throughout our studies.

**O**ur greetings and thanks to the framed professor: Chourouk Guettas.

**W**e would also like to thank the members of the jury for having devoted a part of their time reading this memoir and for their interest in this work.

**F**inally, without exception, We thank all our friends and all who helped us from near or afar.

# Abstract

This work we did during the graduation project presents an academic Licence's degree in computer science. Where we worked on developing an application for the smartphone with the Android operating system dedicated to helping the elderly and the visually impaired to read. We first started by diagnosing the problem of difficulty reading. Then we suggested our solution of developing an Android app to read documents. For development, we used the Android Studio integrated development system with Java.



**Keywords:** Android Studio, Text-to-Speech (TTS), Text Recognition, Aid Elders.



# ملخص

يقدم هذا العمل الذي قمنا به خلال مشروع التخرج للحصول على شهادة ليسانس أكاديمي في الإعلام الآلي. حيث عملنا على تطوير تطبيق للهاتف الذكي ذو نظام التشغيل أندرويد مخصص لمساعدة كبار السن وضعاف البصر على القراءة. بدأنا أولا بتشخيص مشكلة صعوبة القراءة. ثم قمنا باقتراح حلّنا المتمثل في تطوير تطبيق اندرويد لقراءة المستندات. بالنسبة للتطوير، استعملنا منظومة التطوير المدمجة أندرويد ستوديو مع لغة الجافا.



**كلمات مفتاحية:** اندرويد ستيديو، نص الى كلام، التعرف على النص، مساعد الكبار.

# Contents

# List of Figures

# Introduction General

Nowadays, the smartphone is becoming an essential companion and a multifunctional tool. Millions of people tap on their touch screens all day long, unable to do without this small device. For this reason, the publishers of the operating systems for these devices support are encourage developers to develop mobile applications. In Algeria, as in other regions of the world, the use of mobile devices is rapid growth. The uses of mobile devices are carried out through Mobile applications. Whether to communicate and use services on the Internet, to have fun in watching videos, playing games or sharing photos, to make transactions such as purchases and banking. the need to develop and updating of mobile applications is growing. The mobile application development activity increasingly requires a workforce capable of carrying out projects for the development of mobility-specific computer programs, on different platforms. Elders suffer a lot when trying to read writing especially the small one. To help them so This project arises as an initiative proposing the study, design and realization of a android application to help Elders to read . this application can read the documents for you after detecting it through the camera. where the user takes a picture of the text to be read, and the application extracts the text from the image and then reads it audibly.

# Chapter 1

# Introduction

## 1 Introduction

In this chapter we will clarify the details some key points of this topic, objectives and required work. We will give a definition of Android applications and recognizing pictures, language detection, finally we will give an overview of our application.

## 2 Android Applications

An Android application is a mobile application specifically developed for Smartphones using the Android application system purchased and developed by Google.
Like the iPhone applications of which they are often replicas, Android applications are very variable in nature:

- Mobile games.
- Trade.
- Utility.
- Information service.

Can be written using Kotlin, Java, and C++ languages. The Android SDK tools compile your code along with any data and resource files into an APK, an Android package, which is an archive file with an "apk" sufix. One APK file contains all the contents of an Android application and is the file that Android-powered devices use to install the application.
Components of the android applications:

- Activities: which are the presentation layer of your application.
- Services: the components that run in the background.
- Content providers: Shareable data sources.
- Intents: Inter-application communication framework.
- Broadcast receivers: Consumers of the messages broadcast by the intents. [1]

## 2.1 Utility Applications

Today many efforts are taken for improving of the human interface to the computer, Because no longer people want to sit in front of monitor to read data. Since it needs too much effort to be taken, which involves strain to their eyes. In this point of view Speech Synthesis is becoming one of the most important steps towards improving the human interface to the computer. Speech synthesis is the artificial production of speech. This system is called a speech synthesizer, and it can be implemented in software or hardware products.

A text-to-speech (TTS) system converts normal language text into speech, other systems render symbolic linguistic representations like phonetic transcription into speech. Voice is one of the best alternatives for hours of eye strain involved in reading any document. In case of illiterate people Voice is a better interface rather than Graphic User Interface in English. For that reason research is being done throughout the world for improving the Human Interface to the computer and one of the best options is the ability of a computer to speak to humans. Text-To-Speech is a process in which input text is first analyzed, then processed and understood, and then the text is converted in digital audio and then "spoken". It is a small piece of software, which will speak out the text given to it, such as reading from a newspaper.
Application of Image Recognition  text to speech:

- Text Reader
- Voice Aloud Reader(TTS Reader)
- Text Reader - Read The Text For Me
- Copy Text On Screen – Translate  Copy Anywhere

# 3 Reading Aid Application

Elders suffer a lot when trying to read writing especially the small one. To help do so This project arises as an initiative proposing the study, design and realization of a android application to help Elders to read. this application can read the documents for you after detecting it through the camera. Where the user takes a picture of the text to be read, and the application extracts the text from the image and then reads it audibly.

# 4 Image Recognition

Image recognition, a subcategory of Computer Vision and Artificial Intelligence, represents a set of methods for detecting and analyzing images to enable the automation of a specific task. It is a technology that is capable of identifying places, people, objects and many other types of elements within an image, and drawing conclusions from them by analyzing them. Photo or video recognition can be performed at different degrees of accuracy, depending on the type of information or concept required. Indeed, a model or algorithm is capable of detecting a specific element, just as it can simply assign an image to a large category. So there are different "tasks" that image recognition can perform:

1. Classification: It is the identification of the "class", i.e. the category to which an image belongs. An image can have only one class.

2. Tagging: It is also a classification task but with a higher degree of accuracy. It can recognize the presence of several concepts or objects within an image. One or more tags can therefore be assigned to a particular image.

3. Detection: This is necessary when you want to locate an object in an image. Once the object is located, a bounding box is placed around the object in question.

4. Segmentation: This is also a detection task. Segmentation can locate an element on an image to the nearest pixel. For some cases, it is necessary to be extremely precise, as for the development of autonomous cars.

# 5 Language Detection

Most of the works devoted to language identification are designed to deal with electronic documents, where the text is directly available [2, 3, 4, 5, 6, 7, 8, 10], These approaches rely on language models and statistical analysis of characters [3], or on the detection of keywords/short words [5] or n-grams of characters [2, 4, 6, 5, 7] made a combination of these three types of analysis with a ranking combination strategy to improve the identification rate on two electronic document databases. Also based on n-grams [3], relies on Markov models to model each language and tries to find the best fitting model for a new sequence of characters [4]. More recently, has defined a n-gram method able to identify the language on short texts of same language and on texts composed of multiple languages [8]. combines n-grams with heuristics and the Lin's similarity measure to identify 12 languages (Danish, English, Italian, Spanish, French . . . ) [10]. proposes a graph-based n-gram approach for its system called LIGA to identify the language on short and ill-written texts.

As said before, only few methods are dedicated to language identification on document images [10, 11, 12, 13] and in most cases, language identification is performed on printed documents without any OCR [11, 13]. Both apply shape coding approaches [11]. creates character shape codes gathering family of characters [12], builds word shape codes based on character extremium points and the number of horizontal word runs. Once shape codes extracted [12], measures the similarity between the language templates and the document vector [13]. English and German languages are identified using language models.

A general model (gathering the most frequent words unigram in the five Latin languages) is first generated applying a Latin OCR on the documents of a training set. This general model is used to generate each language model measuring the number of occurrences of each word of the general models in the training set of the language. The language identification is then performed computing the word unigram relative entropy for each language. Regarding the language identification on handwritten documents [1],proposes an approach based on the shape analysis of the connected components of the handwritten document to discriminate the script (Arabic, Cyrillic, Devanagari, Japanese, Latin) and the language (English, German). A document is characterized by the means, the standard deviation and the skew of five features encoding connected components properties (aspect ratio, compactness, number of holes, centroid positions). The classification is performed using a linear discriminant analysis and the system was tested on a private database composed of cleaned images (the irregularities are removed after scanning).This review of the literature devoted to language identification has shown that works were mainly focused on digital documents.

These approaches are based on statistical text analysis or on the detection of keywords or n-grams and all achieve high performance with an average accuracy classification around 99%.On the other hand, approaches dedicated to language identification on document images are very few and the problem is more complicated given that text information is not available. Existing approaches in the literature are focused on printed documents. They work at the document level and use mainly shape analysis [11, 12]. Both reach an average accuracy above 90% using shape coding approaches considering respectively 23 and 8 languages [13], whereas , combining spatial features with the analysis of OCR outputs, achieves an average precision of 94.76% on a private dataset composed of fax images, considering 7 languages. The only approach dedicated to handwritten documents [13] achieves a classification average accuracy around 85% for the discrimination of German/English languages, on images previously cleaned with Adobe Photoshop in order to remove any irregularities (illustrations, doodles, anomalous writing, etc...).

# 6 Text To Speech

The goal of speech synthesis is to automatically generate artificial speech waveforms from another data format. Especially, a TTS synthesizer is a computer-based system that should be able to read any text aloud. The main components of a TTS are NLP (Natural Language Processing) module responsible for the production of the phonetic transcription of an input text and, in some cases, the generation of prosody (intonation, duration, intensity or power) DSP (Digital Signal Processing) module, which converts the data output from the NLP module into speech.There are two main approaches to speech synthesizers: the rule-based and data-driven methods. The rule-based speech synthesis approach aims at re-creating the same sounds as those created by human speech system in modeling the human speech production system. In contrast, data-driven synthesizers generate a speech by arranging in a specific order, various segments of prerecorded speech. One of the most used types of data-driven method is the unit selection approach [14, 15]. It can be achieved by concatenating the speech audio signal from various units selected from a large database of speech spoken by a single speaker. With a database containing a wide range of units of speech having various contexts, prosodic and spectral characteristics, it can synthesize an utterance that would seem more close to human speech [16].

# 7 Conclusion

In this chapter we gave a basic overview of image recognition, and mobile applications on Android platform, with a general description about the application developed in this project. In the second chapter, we will present the architecture and the design of our Application.

# Chapter 2

# Conception

## 1 Introduction

Before starting the development of our application, we must specify the objective of this application by determining the functional and non-functional needs to which our application should respond. In this chapter, we will first present a general architecture of our system which contains the different interactions between actors and others system components.

## 2 Architecture

### 2.1 Introduction

To read the text phonetically it must passes through four stages which is illustrated by the following schema (Figure below).

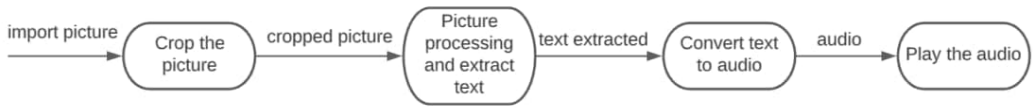



Figure 2.1: Main stages

### 2.2 Import The Image

To enter the pictures to the app we have two ways, if the picture already stored in phone storage we have to import it from the gallery or if it is not already stored we have to take it through camera.

### 2.3 Processing the image and extract the text

#### 2.3.1 Crop The Picture

After uploading the photo, a obvious field seems over it that indicates the bounds of the photo, then the user adjustments the scale of the field in share to the brand new dimensions of the photo from which the textual content is to be extracted.

The running precept of the photo cropping set of rules is primarily based totally on choosing factors from the chord, and on the premise of which the brand new borders of the photo are decided[17].

### 2.3.2 Extract Text

In order to extract the text from the picture we use OCR (Optical Caracter Recognition). There are two main methods for extracting features in OCR:

1. In the first method, the algorithm for feature detection defines a character by evaluating its lines and strokes.
2. In the second method, pattern recognition works by identifying the entire character.

We can recognize a line of text by searching for white pixel rows that have black pixels in between. Similarly, we can recognize where a character starts and finishes.

The following pictures show the visualization of these methods respectively:

Figure 2.2: Method 1. Feature detection.


Figure 2.3: Method 2. Pattern recognition on a row of text.


Figure 2.4: Method 3. Pattern recognition on a single character.

Next, we convert the image of the character into a binary matrix where white pixels are 0s and black pixels are 1s as shown in the following figure:


Figure 2.5: Simple of binary matrix.

Then, by using the distance formula, we can find the distance from the center of the matrix to the farthest 1.

$$d = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}$$

Figure 2.6: The distance formula.

We then create a circle of that radius and split it up into more granular sections.
At this stage, the algorithm compares each subsection to a database of matrices representing characters with different fonts to identify the character it has most in common statistically.
It makes it easy to bring printed media into the digital world by doing this for every line and character[18].

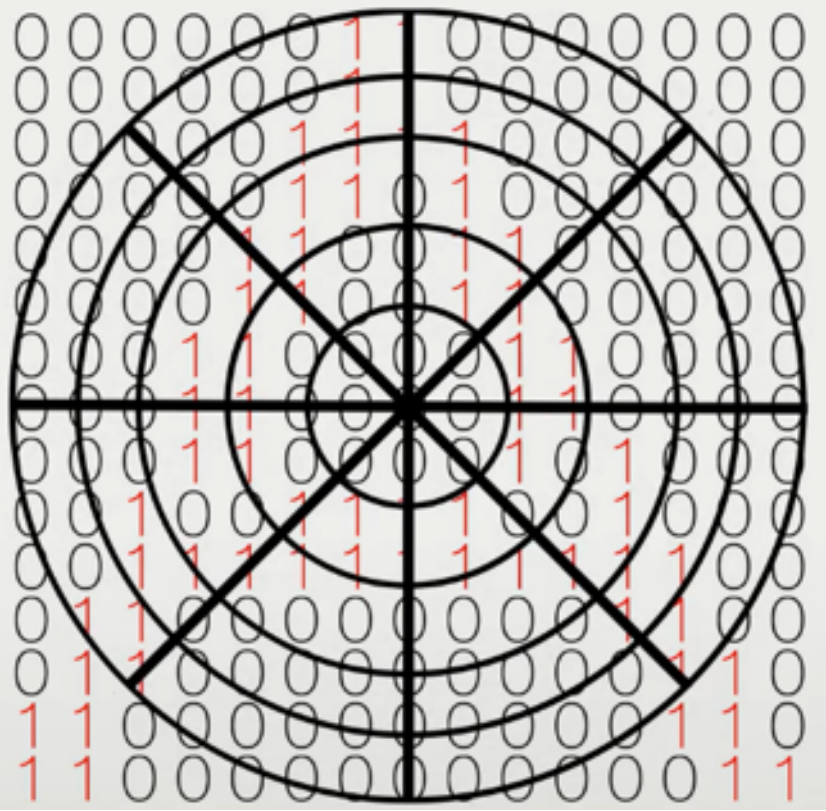


Figure 2.7: Compare each subsection against the matrix database.

## 2.4 Read The Text

A text-to-speech (TTS) system converts normal language text into speech, Synthesized speech can be created by concatenating pieces of recorded speech that are stored in a database.

A text-to-speech system (or "engine") is composed of two parts:[19] a front-end and a back-end.

1. The front-end has two major tasks. First, it converts raw text containing symbols like numbers and abbreviations into the equivalent of written-out words. This process is often called text normalization, pre-processing, or tokenization. The front-end then assigns phonetic transcriptions to each word, and divides and marks the text into prosodic units, like phrases, clauses, and sentences. The process of assigning phonetic transcriptions to words is called text-to-phoneme or grapheme-to-phoneme conversion. Phonetic transcriptions and prosody information together make up the symbolic linguistic representation that is output by the front-end.

2. The back-end (often referred to as the synthesizer) then converts the symbolic linguistic representation into sound. In certain systems, this part includes the computation of the target prosody (pitch contour, phoneme durations),[20] which is then imposed on the output speech.

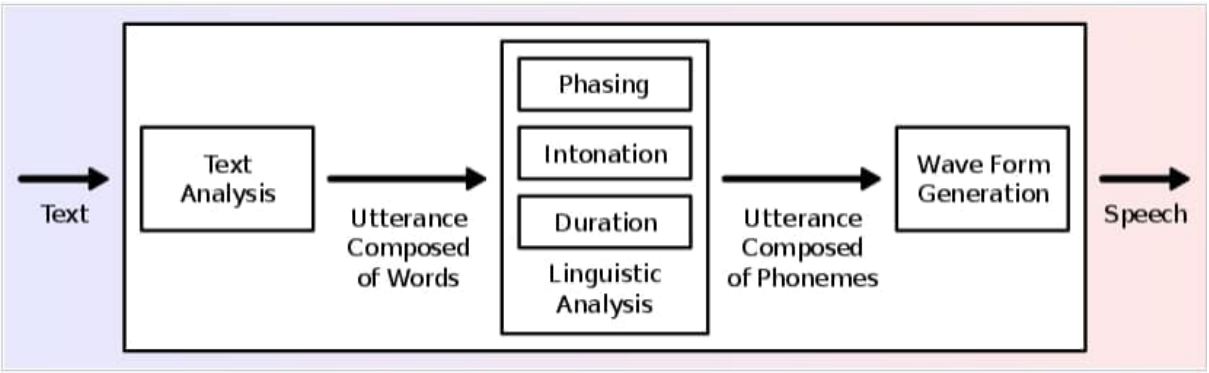



Figure 2.8: Text to Speech stages

# 3 Conclusion

In this chapter we discussed the architecture of our application step by step that represented by importing the picture then processing it and extract the text, the converting that text to speech, this chapter has allowed us to prepare the phase of realization that will concretize everything that has been presented so far.

# Chapter 3

# Implementation

## 1 Introduction

This chapter contains a description of the development tools that we used to create this project. It includes a detailed explanation of how to use the application with some images and the available services in the application. We have chosen open source development tools to avoid legal aspects, contracts, licenses and links. Thus, reducing costs.

## 2 Work Environment

The used development environments, both hardware and software are described below.

### 2.1 Development device information:

For the realization of our project, we used a Lenovo computer characterized by:

- Operating system: Windows 10 (x64)
- Processor: Intel Core i7 5500U / 2.4 GHz.
- RAM: 16 GB.
- Hard Disk: 1 TB.

In order to install and run the application we used a Samsung smart phone with:

- Name of the Device: Samsung Galaxy M21.
- Android version: 11
- RAM: 4GB.
- Internal storage: 64GB.
- Connection: LTE, 3G, 2G.
- WI-FI: 802.11 a.
- GPS: Yes.

## 2.2 Java development kit (JDK)

The Java Development Kit (JDK) is a software development environment that offers a collection of tools and libraries necessary for developing Java applications. You need the JDK to convert your source code into a format that the Java Runtime Environment (JRE) can execute.

## 2.3 Android software development kit (SDK)

The Android Software Development Kit contains the necessary tools to create, compile and package Android applications. Most of these tools are command line based. The primary way to develop Android applications is based on the Java programming language.

## 2.4 Android virtual device (AVD)

Android Virtual Device (AVD) is a configuration that defines the characteristics of an Android phone, tablet, Wear OS, Android TV, or Automotive OS device that you want to simulate in the Android Emulator. It contains a hardware profile, system image, storage area, skin, and other properties.

## 2.5 Android studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA. On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system.
- A fast and feature-rich emulator.
- A unified environment where you can develop for all Android devices.
- Apply Changes to push code and resource changes to your running app without restarting your app.
- Code templates and GitHub integration to help you build common app features and import sample code.
- Extensive testing tools and frameworks.
- Lint tools to catch performance, usability, version compatibility, and other problems.
- C++ and NDK support.
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine.

## 2.6 Used Libraries

The libraries we have used in our project.

### 2.6.1 Butter Knife

Android Butterknife is a view binding tool that uses annotations to generate boilerplate code for us. ButterKnife is developed by Jake Wharton at Square and is essentially used to save typing repetitive lines of code like findViewById(R.id.view) when dealing with views, thus making our code look a lot cleaner. To use ButterKnife in android application we need to add the following lines to our build.gradle file Figure 3.1.

```
android {
  ...
  // Butterknife requires Java 8.
  compileOptions {
    sourceCompatibility JavaVersion.VERSION_1_8
    targetCompatibility JavaVersion.VERSION_1_8
  }
}

dependencies {
  implementation 'com.jakewharton:butterknife:10.2.3'
  annotationProcessor 'com.jakewharton:butterknife-compiler:10.2.3'
}
```

Figure 3.1: Butter Knife tool

### 2.6.2 Play services vision

Google Play Services is a proprietary background service and API package for Android devices from Google, one of those APIs is called Text Recognition, it can recognizes text in any Latin based language. It also represents the structure of recognized text, including paragraphs and lines. To use this API we need to add the following line to our build.gradle file Figure 3.2.

```
  // Google Play services APIs (play service vision).
  implementation 'com.google.android.gms:play-services-vision:20.1.3'
```

Figure 3.2: Play Services Vision

### 2.6.3 Material design

Material is an adaptable system of guidelines, components, and tools that support the best practices of user interface design. Backed by open-source code, Material streamlines collaboration between designers and developers, and helps teams quickly build beautiful products. To use material design we need to add the following ling to our build.gradle file Figure 3.3.

```
// Material design.
implementation 'com.google.android.material:material:1.3.0-alpha03'
```

Figure 3.3: Material Design

### 2.6.4 Bubble seek bar

A beautiful Android custom seek bar, which has a bubble view with progress appearing upon when seeking. Highly customizable. To use it we need to add the following line in our build.gradle file Figure 3.4.

```
// Bubble seek bar.
implementation 'com.xw.repo:bubbleseekbar:3.5-lite'
```

Figure 3.4: BubbleSeekBar library

### 2.6.5 ML Kit LanguageID

we used ML Kit to identify the language of a string of text. You can get the string's most likely language as well as confidence scores for all of the string's possible languages. ML Kit recognizes text in more than 100 different languages in their native scripts including Arabic, English and chinese. To use it we need to add the following line in our build.gradle file Figure 3.5.

```
// Machine learning for mobile developer
// we used google machine learning to identify the language.
implementation 'com.google.mlkit:language-id:16.1.1'
```

Figure 3.5: ML Kit

### 2.6.6 CanHub (Image cropper)

Image Cropping Library for Android, optimised for Camera / Gallery. To use it we need to add the following line in our build.gradle file Figure 3.6.

```
// CanHub Library (Image cropper).
implementation 'com.github.CanHub:Android-Image-Cropper:2.2.1'
```

Figure 3.6: CanHub library

## 2.7 SQLite

SQLite is a opensource SQL database that stores data to a text file on a device. Android comes in with built in SQLite database implementation. SQLite supports all the relational database features. In order to access this database, you don't need to establish any kind of connections for it like JDBC,ODBC ...etc. The main package is android.database.sqlite that contains the classes to manage your own databases. In order to exploit it we have to define a new class we named it SQLite, then declare two principal variables Figure 3.7.

```
/**
 * When you change the SQLite schema you should increase the version.
 **/
public static final String DATABASE_NAME = "DocsEvee";
public static final int DATABASE_VERSION = 11;
```

Figure 3.7: SQLite version and name.

It have two main methods.

1. onCreate() method is called only once throughout the application lifecycle. It will be called whenever there is a first call to getReadableDatabase() or getWritableDatabase() function available in super SQLiteOpenHelper class. So SQLiteOpenHelper class call the onCreate() method after creating database and instantiate SQLiteDatabase object. Database name is passed in constructor call. So we add the line to create the database in this method Figure 3.8.

```
@Override
public void onCreate(SQLiteDatabase db) {

    // Create table for uploaded files.
    db.execSQL("create table files (id INTEGER PRIMARY KEY AUTOINCREMENT, title TEXT,text TEXT, image BLOB,created_at TEXT)");

    // Create table for configuration.
    db.execSQL("create table IF NOT EXISTS config (id INTEGER PRIMARY KEY CHECK (id = 0), language TEXT)");
    // Insert into config.
    db.execSQL("INSERT INTO config(id, language) VALUES(0, 'en')");
}
```

Figure 3.8: onCreate method

2. onUpgrade() is only called whenever there is a updation in existing version. So to update a version we have to increment the value of version variable passed in the superclass constructor. In onUpgrade method we can write queries to perform whatever action is required. so we write queries to drop tables and remove the generated audio files Figure 3.9.

```
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

    db.execSQL("DROP TABLE IF EXISTS files");
    db.execSQL("DROP TABLE IF EXISTS config");
    onCreate(db);
    String fileName = Environment.getExternalStorageDirectory().getAbsolutePath()
            +"/DocsEyee/";
    File f = new File(fileName);
    try {
        FileUtils.deleteDirectory(f);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Figure 3.9: onUpgrade method

We created a public methods so that we can use them in other classes to insert Figure 3.10, update Figure 3.11, delete Figure 3.12 and get files Figure 3.13 and more Figures 3.[14/15/16].

```
// Insert uploaded file.
public int insertFile(String title,String text, Bitmap image) throws SQLiteException{
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues value = new ContentValues();
    byte[] byteImage = BitmapUtility.getBytes(image);
    c = Calendar.getInstance();
    String date = df.format(c.getTime());
    value.put("title", title);
    value.put("text", text);
    value.put("image", byteImage);
    value.put("created_at", date);
    int rz =(int) db.insert( table: "files",  nullColumnHack: null, value);
    return rz;
}
```

Figure 3.10: Insert file method

```
// Update file title.
public void updateFileTitle(int id, String newTitle){
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues value = new ContentValues();
    value.put("title", newTitle);
    int result = db.update( table: "files", value,  whereClause: "id ="+id,  whereArgs: null);
}
```

Figure 3.11: Update file method

```java
// Delete uploaded file.
public void deleteFile(int id){
    SQLiteDatabase db = this.getWritableDatabase();
    db.execSQL("DELETE FROM files WHERE id="+id);
}
```

Figure 3.12: Delete file method

```java
// Get all uploaded files.
public ArrayList<FileUploaded> getFiles(){
    SQLiteDatabase db = this.getReadableDatabase();
    ArrayList<FileUploaded> results = new ArrayList<>();

    Cursor cs = db.rawQuery( sql: "SELECT * FROM files", selectionArgs: null);
    if (cs.moveToFirst()) {
        while (cs.isAfterLast() == false) {
            FileUploaded file = new FileUploaded();
            file.setId(cs.getInt(cs.getColumnIndex( columnName: "id")));
            file.setTitle(cs.getString(cs.getColumnIndex( columnName: "title")));
            file.setText(cs.getString(cs.getColumnIndex( columnName: "text")));
            file.setImage(BitmapUtility.getImage(cs.getBlob(cs.getColumnIndex( columnName: "image"))));
            file.setCreatedAt(cs.getString(cs.getColumnIndex( columnName: "created_at")));
            results.add(file);
            cs.moveToNext();
        }
    }
    return results;
}
```

Figure 3.13: Get files method

```java
// Update Language.
public void updateLang(String lang){
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put("language", lang);
    int result = db.update( table: "config",values,  whereClause: "id=0", whereArgs: null);
}
```

Figure 3.14: Update language method

```
// Get Language.
public String getLang(){
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cs = db.rawQuery( sql: "SELECT * FROM config WHERE id = 0", selectionArgs: null);
    cs.moveToFirst();
    return cs.getString(cs.getColumnIndex( columnName: "language"));
}
```

Figure 3.15: Get language method


```
// Get Language.
public String getLang(){
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cs = db.rawQuery( sql: "SELECT * FROM config WHERE id = 0", selectionArgs: null);
    cs.moveToFirst();
    return cs.getString(cs.getColumnIndex( columnName: "language"));
}
```


Figure 3.16: Get language method

We used it in our application to store the uploaded files so that the user don't have to upload the same file twice.

# 3 Presentation of DocsEyee:

In this section, we will be presenting the main interfaces of our mobile application.

Figure 3.17: Application's icon

## 3.1 Application Interfaces

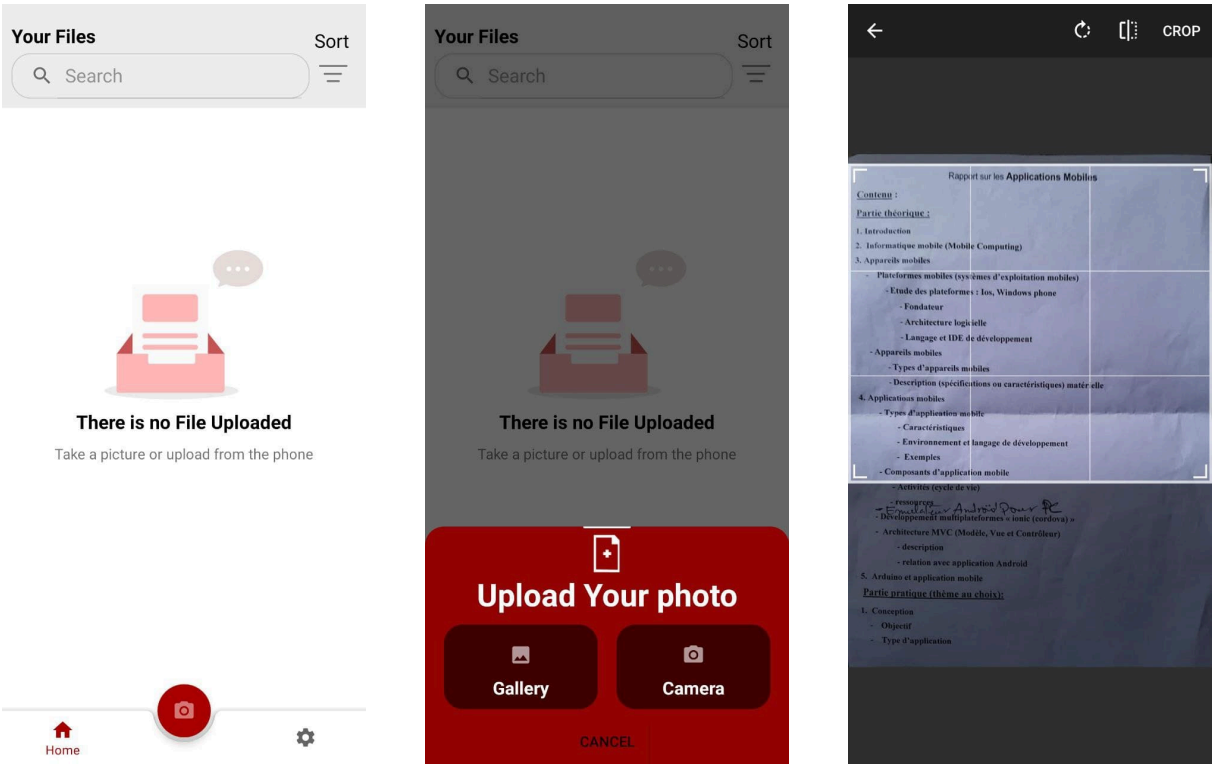
### 3.1.1 SplashScreen:

This is the application loading interface, containing the application logo and it takes about 2 seconds Figure 3.18.


Figure 3.18: SplashScreen

### 3.1.2 HomeScreen

When the splash screen finishes,The application start home interface, for the first use when the user have no file uploaded yet, It tells the user that there is no file uploaded. The user can upload a picture or take one by clicking the red floating button in the bottom Figure 3.19(a). It will open bottom sheet with two options take or capture a picture Figure 3.19(b). If the user chose Gallery it will open the gallery to choose a picture and if he chose Camera it will open the camera to capture the file. Then after the picture has taken successfully, It wilt send the picture to the crop interface to crop the desired part Figure 3.19(c).




(a) There is no file

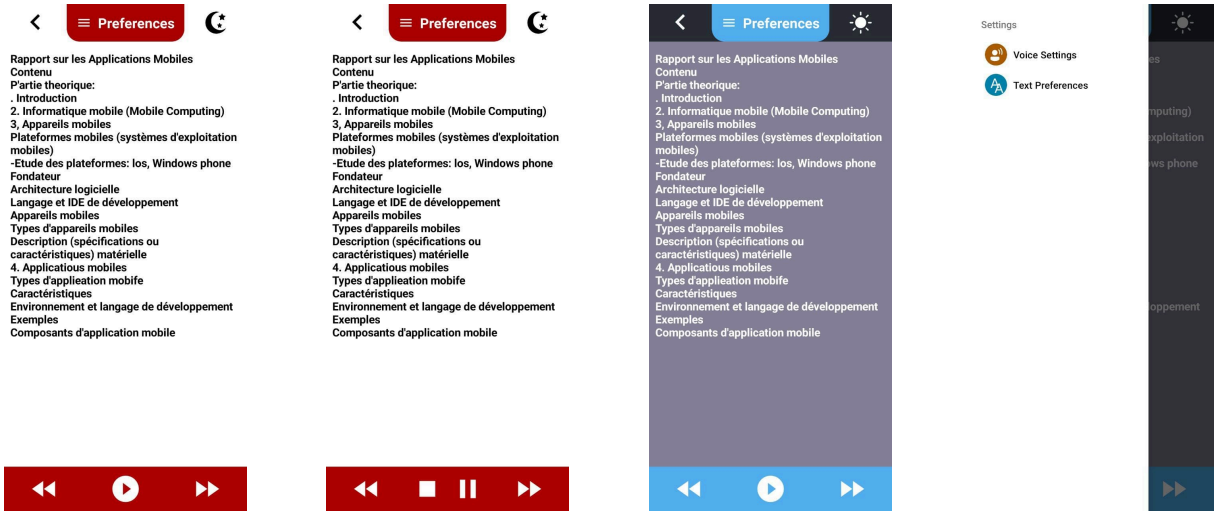
(b) Upload Picture bottom sheet

(c) Upload Picture bottom sheet

Figure 3.19: Home Screen

### 3.1.3 Read Screen

After the user successfully upload a picture, the app detect the text in the picture and send it to Reading interface, and this latter has Play button to read the text phonetically Figure 3.20(a). When the app starts reading, the play button disappears and the stop and pause buttons appear. Also, if the user wants to forward or rewind it's possible with forward and rewind buttons in the bottom Figure 3.20(b). In addition, the user can change the reading theme by clicking the crescent or sun icon it helps well when the user is in low light (it's something like night mode) Figure 3.20(c). The user can adjust the text and voice preferences by clicking on Preferences on the top of the interface to pull the side menu Figure 3.20(d).

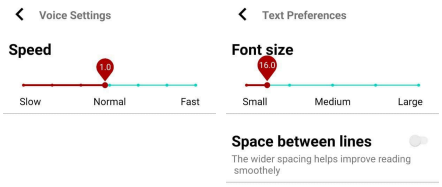



(a) Play button (b) Stop/Pause button (c) Night mode (d) Navigation menu

Figure 3.20: Read Screen

### 3.1.4 Reading Preferences

The user can click on voice settings in order to change the voice speed, by default it is in normal speed Figure 3.21(a). Also, he can change the font size from small size to large by default it is 16sp, and he switch on/off space between lines because the wider spacing helps improve reading smoothly Figure 3.21(b).

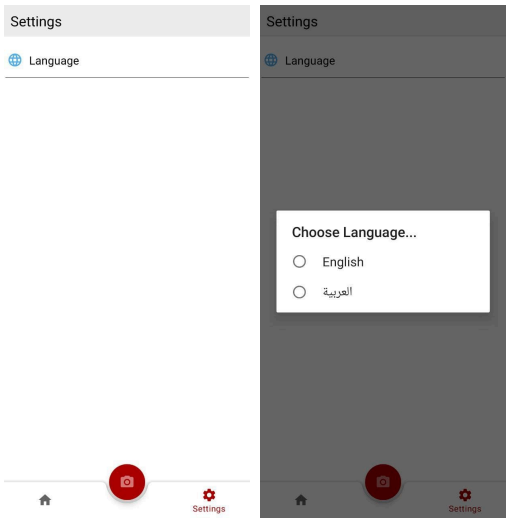



(a) voice setting (b) Text Settings

Figure 3.21: Reading Preferences

### 3.1.5 Application Settings

From home interface the user can click on settings Figure 3.22(a) and click on language to change the application language, he can choose either English or Arabic Figure 3.22(b).

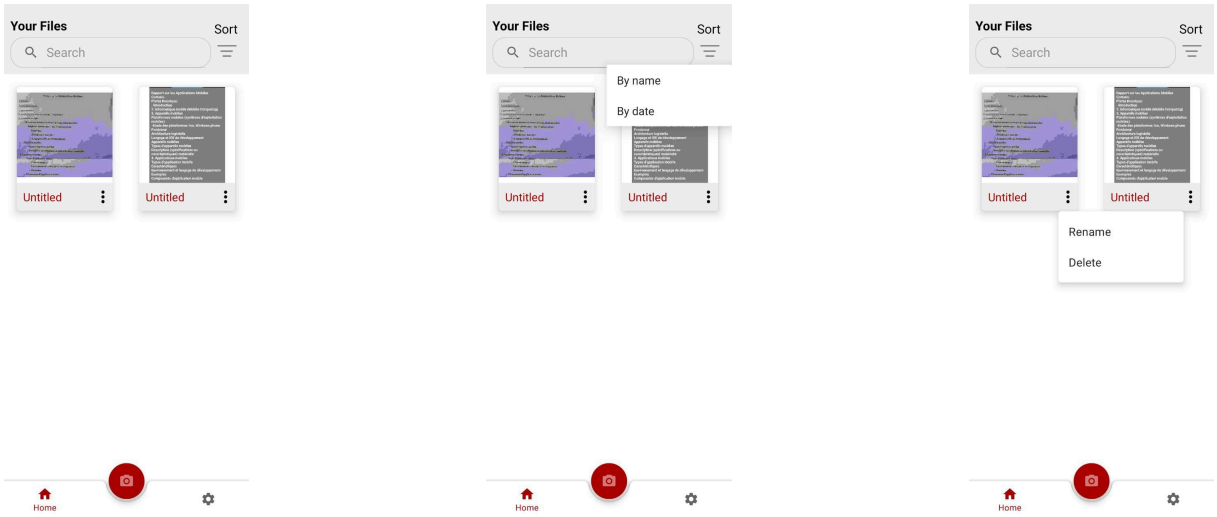



(a) Settings (b) Language

Figure 3.22: Settings

### 3.1.6 Uploaded Files

After the user uploaded files successfully it will be shown as array-list in good view so he can search for any file by its title Figure 3.23(a). He can also order the files by name so it will be (A to Z) or (Z to A), or by upload date so it will be from (new to old) or (old to new) Figure 3.23(b). In addition he can rename or delete the files Figure 3.23(c).




(a) Files (b) Sort files (c) Rename/Delete file

Figure 3.23: Uploaded Files

# 4 Results and discussion

The results obtained from this study is much, one of these the mechanism of storing pictures, we stored pictures in SQLite as bytes and then retrieve it, this mechanism is useful if the user deleted the main picture by mistake so the picture in our app wont be deleted, in addition

this mechanism is faster when retrieving the picture, but one of its downsides is the quality of the picture is decreasing. The second mechanism is based on storing the path of the main picture if it is already in the gallery, if he want to take it by camera so the app stores the picture as file with image extension then takes its path. this mechanism gives us pictures with good quality, but if the user deleted the picture by mistake it wont be available anymore, so we chose the first one.

We noticed also that converting Text-To-Speech results to audio file is more useful than reading the directly because it will give us the speedy in reading so we do not have to convert the text to speech every time the user want to listen so this benefit us the time and device resource, in addition it gives us the forward and rewind in listening.

# 5 Conclusion

In this last chapter, we presented the development environment of our application and the various tools and libraries used. Afterwards, we have presented the different interfaces of our application. We take seriously to respect at most the HCI's standards to achieve our application.

# General Conclusion

We began this note as a general description of the applications of smart mobile initially applications and types of platforms. Then we put in question the most important problem that Elders suffer. We sealed it with a set of solutions in the form of a mobile application. In chapter II of the note we have identified the main specifications and features of the application to be achieved Then we introduced the construction stages where we began analyzing needs, we then modeled the specific functions of our application in a schema. Either in the third part we moved to the practical side and is part of the mobile application realization that works on the android platform and we also put pictures of most of the application screens with explanation. The subject we worked on was good and useful to people's lives specially the Elders. We liked the concrete aspect of this project because we could easily see the result of our work, which was very motivating. At the end of this project, our application is released in its first version. It will always remain open to prospects for improvement. We are considering adding another features for very large use. Thus, we plan to make our application cover the entire national territory. We are therefore considering adding new features based on user recommendations.

# Prospects

We have completed the basic idea of this research and added excellent features that help to enhance the idea of the application, but there are still many additions that can improve the user experience, including:

- Reading text from different files (PDF, Word, Excel, ...).
- Reading text in Arabic language.
- Extract Arabic text from picture.
- Translate the text.

# Bibliography


[1] N.BENBOURAHLA,"Android 4 les fondamentaux fe developpement d'application java",2012.

[2] Google plug-in website for language http://code.google.com/p/language-detection/ detection.

[3] T. Dunning, Statistical Identification of Language, Techreport, 1994.

[4] R. ˇReh˚uˇrek, M. Kolkus, Language identification on the web: extending the dictionary method, CICLing, 2009, 357–368.

[5] G. Grefenstette, Comparing two language identification schemes, JADT, 1995.

[6] W. B. Cavnar, J. M. Trenkle, N-Gram-Based Text Categorization, SDAIR, 1994, 161–175.

[7] L. Grothe, E. W. De Luca, A. N¨urnberger, A Comparative Study on Language Identification Methods, In LREC, 2008.

[8] B. Martins, M. J. Silva, Language identification in web pages, In Proceedings of the 2005 ACM symposium on Applied computing, 2005, 764–768.

[9] E. Tromp, M. Pechenizkiy,Graph-based n-gram language identification on short texts, In Proc. 20th Machine Learning conference of Belgium and The Netherlands, May 2011, 27–34.

[10] P. Sibun, A. L. Spitz, Language determination: Natural language processing from scanned document images, In ANLP, October 1994, 15–21.

[11] L. Shijian, C. L. Tan, Script and language identification in noisy and degraded document images, IEEE PAMI, 2008, 14–24.

[12] D. S. Lee, C. R. Nohl, H. S. Baird, Language identification in complex, unoriented, and degraded document images, In Series in Machine Perception and Artificial Intelligence, 1998, 17–39.

[13] J. Hochberg, K. Bowers, M. Cannon, P. Kelly, Script and Language Identification for Handwritten Document Images, IJDAR, 1999, 45-52.

[14] Zen H., Tokuda K., Black A. W. Statistical parametric speech synthesis . Speech Communication, Elsevier 2009; 51(11): 1039-1064.

[15] Clark, R., Richmond, K., King, S. Multisyn: Open-domain unit selection for the Festival speech synthesis system. Speech Communication , Elsevier 2007; 49(4):317-330.

[16] Ngugi K., Okelo-Odongo W., Wagacha P. W. Swahili Text-To-Speech System. African Journal of Science and Technology, 2005; 6 (1): 80-89.

[17] Article Grid Anchor based Image Cropping: A New Benchmark and An Efficient Model. Authors: Hui Zeng, Lida Li, Zisheng Cao, and Lei Zhang.

[18] https://moov.ai/en/blog/optical-character-recognition-ocr/ viewed at 2021-06-07.

[19] van Santen, Jan P. H.; Sproat, Richard W.; Olive, Joseph P.; Hirschberg, Julia (1997). Progress in Speech Synthesis.

[20] Van Santen, J. (April 1994). "Assignment of segmental duration in text-to-speech synthesis". Computer Speech Language.