Software Requirements Specification (SRS)

# Extracting and Parsing Information from Emirates ID Card PDF

## 1. Introduction

#### 1.1 Purpose

The purpose of this document is to define the software requirements for a system that extracts and parses information from PDF files containing images of Emirates ID cards. The extracted information includes fields such as card number, ID number, name, date of birth, nationality, occupation, issuing place, sex, issuing date, and expiry date.

#### 1.2 Scope

The system will convert PDF files to images, preprocess these images to enhance text clarity, perform Optical Character Recognition (OCR) to extract text, and then parse the extracted text to identify and extract the required fields. The system will log the extracted text for review and debugging purposes and output the parsed information.

#### 1.3 Definitions, Acronyms, and Abbreviations

- \*\*OCR\*\*: Optical Character Recognition

- \*\*PDF\*\*: Portable Document Format

- \*\*DPI\*\*: Dots Per Inch

- \*\*ID\*\*: Identification

#### 1.4 References

- Python documentation: [https://www.python.org/doc/](https://www.python.org/doc/)

- Tesseract OCR: [https://github.com/tesseract-ocr/tesseract](https://github.com/tesseract-ocr/tesseract)

- pdf2image: [https://github.com/Belval/pdf2image](https://github.com/Belval/pdf2image)

- Pillow (PIL): [https://python-pillow.org/](https://python-pillow.org/)

- OpenCV: [https://opencv.org/](https://opencv.org/)

#### 1.5 Overview

This document describes the functional and non-functional requirements, system architecture, and the steps involved in processing the PDF files to extract and parse the required information.

### 2. General Description

#### 2.1 Product Perspective

The system will be a standalone script written in Python. It will use various libraries such as `pdf2image`, `pytesseract`, `Pillow`, and `OpenCV` to achieve the desired functionality.

#### 2.2 Product Functions

- Convert PDF files to images.

- Preprocess images to enhance text clarity.

- Perform OCR on preprocessed images.

- Parse the extracted text to identify and extract required fields.

- Log the extracted text for review and debugging.

- Output the parsed information.

#### 2.3 User Characteristics

The users of this system will be developers and data analysts who need to extract and parse information from Emirates ID card PDFs.

#### 2.4 Constraints

- The system should handle PDF files that contain images of Emirates ID cards.

- The system should process each page of the PDF efficiently.

- The system should accurately extract and identify the required fields.

#### 2.5 Assumptions and Dependencies

- The PDF files are well-formed and contain clear images of the ID cards.

- The required libraries (`pdf2image`, `pytesseract`, `Pillow`, `OpenCV`) are installed and configured correctly.

- Poppler is installed for PDF conversion.

### 3. Specific Requirements

#### 3.1 Functional Requirements

##### 3.1.1 Convert PDF to Images

- \*\*Description\*\*: The system shall convert each page of the PDF file to high-resolution images (default DPI of 300).

- \*\*Input\*\*: PDF file

- \*\*Output\*\*: List of images

##### 3.1.2 Preprocess Images

- \*\*Description\*\*: The system shall preprocess the images to enhance text clarity. This includes converting images to grayscale, applying adaptive thresholding, and denoising the images.

- \*\*Input\*\*: List of images

- \*\*Output\*\*: List of preprocessed images

##### 3.1.3 Extract Text from Images

- \*\*Description\*\*: The system shall use Tesseract OCR to extract text from each preprocessed image.

- \*\*Input\*\*: List of preprocessed images

- \*\*Output\*\*: Extracted text

##### 3.1.4 Parse Extracted Text

- \*\*Description\*\*: The system shall parse the extracted text to identify and extract the required fields using regular expressions.

- \*\*Input\*\*: Extracted text

- \*\*Output\*\*: Dictionary containing parsed information (card number, ID number, name, date of birth, nationality, occupation, issuing place, sex, issuing date, expiry date)

##### 3.1.5 Log Extracted Text

- \*\*Description\*\*: The system shall log the entire extracted text for review and debugging purposes.

- \*\*Input\*\*: Extracted text

- \*\*Output\*\*: Log file

##### 3.1.6 Output Parsed Information

- \*\*Description\*\*: The system shall print the parsed fields to the console.

- \*\*Input\*\*: Dictionary containing parsed information

- \*\*Output\*\*: Console output

#### 3.2 Non-Functional Requirements

##### 3.2.1 Performance

- \*\*Description\*\*: The system shall process each page of the PDF efficiently and within a reasonable time frame.

##### 3.2.2 Accuracy

- \*\*Description\*\*: The OCR and parsing techniques shall accurately extract and identify the required fields from the ID card images.

##### 3.2.3 Error Handling

- \*\*Description\*\*: The system shall handle errors gracefully, including file not found errors for the PDF, errors during PDF to image conversion, and errors during OCR text extraction.

##### 3.2.4 Logging

- \*\*Description\*\*: The system shall log significant steps in the processing pipeline, including conversion of PDF to images, text extraction from images, and parsing of extracted text.

### 4. System Architecture

#### 4.1 System Components

- \*\*PDF to Image Converter\*\*: Converts PDF pages to high-resolution images.

- \*\*Image Preprocessor\*\*: Preprocesses images to enhance text clarity.

- \*\*OCR Engine\*\*: Uses Tesseract to extract text from preprocessed images.

- \*\*Text Parser\*\*: Parses the extracted text to identify and extract required fields.

- \*\*Logger\*\*: Logs the extracted text and significant steps in the processing pipeline.

#### 4.2 Data Flow

1. \*\*Input\*\*: PDF file

2. \*\*Convert PDF to Images\*\*: Convert each page of the PDF to high-resolution images.

3. \*\*Preprocess Images\*\*: Preprocess images to enhance text clarity.

4. \*\*Extract Text\*\*: Use Tesseract OCR to extract text from preprocessed images.

5. \*\*Log Extracted Text\*\*: Log the entire extracted text for review and debugging.

6. \*\*Parse Extracted Text\*\*: Parse the extracted text to identify and extract required fields.

7. \*\*Output\*\*: Print the parsed fields to the console.

### 5. User Interface

#### 5.1 Console Interface

- The system will run as a command-line script.

- The user will provide the path to the PDF file as an input argument.

- The system will output the parsed information to the console.

### 6. Appendices

#### 6.1 Installation and Setup Instructions

1. \*\*Install Required Libraries\*\*:

```bash

pip install pytesseract pdf2image pillow opencv-python-headless

```

2. \*\*Install Poppler\*\*:

- \*\*Windows\*\*: Download and install from [Poppler for Windows](http://blog.alivate.com.au/poppler-windows/).

- \*\*macOS\*\*: Install via Homebrew:

```bash

brew install poppler

```

#### 6.2 Sample Script

```python

import cv2

import pytesseract

from pdf2image import convert\_from\_path

from PIL import Image

import re

import logging

import numpy as np

# Configure logging

logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

def preprocess\_image(image):

"""

Preprocess the image to improve OCR accuracy using OpenCV.

:param image: PIL Image object.

:return: Preprocessed image.

"""

# Convert to grayscale

image = np.array(image)

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

# Apply adaptive thresholding

thresh = cv2.adaptiveThreshold(

gray, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C, cv2.THRESH\_BINARY, 11, 2

)

# Remove noise

denoised = cv2.fastNlMeansDenoising(thresh, None, 30, 7, 21)

return denoised

def convert\_pdf\_to\_images(pdf\_path, dpi=300):

"""

Convert PDF to images.

:param pdf\_path: Path to the PDF file.

:param dpi: Resolution for conversion.

:return: List of images.

"""

images = convert\_from\_path(pdf\_path, dpi=dpi)

return images

def extract\_text\_from\_images(images):

"""

Extract text from the provided images using OCR.

:param images: List of image objects.

:return: Extracted text as a string.

"""

text = ""

for image in images:

preprocessed\_image = preprocess\_image(image)

text += pytesseract.image\_to\_string(preprocessed\_image, config='--psm 6')

return text

def parse\_id\_card(text):

"""

Parse the text extracted from the ID card.

:param text: OCR extracted text from the ID card.

:return: Dictionary containing parsed information.

"""

# Extract card number based on the "IARE" pattern

card\_number\_pattern = r'IARE(\d{8})'

card\_number\_match = re.search(card\_number\_pattern, text)

card\_number = card\_number\_match.group(1) if card\_number\_match else None # Extract the 8 digits after

"IARE"

# Extract ID number with the specific format

id\_number\_pattern = r'\b\d{3}-\d{4}-\d{7}-\d\b'

id\_number\_match = re.search(id\_number\_pattern, text)

id\_number = id\_number\_match.group(0) if id\_number\_match else None

# Define regex patterns for other fields

patterns = {

'Name': r'Name\s\*:\s\*(.\*)',

'Date of Birth': r'Date of Birth\s\*:\s\*(\d{2}/\d{2}/\d{4})',

'Nationality': r'Nationality\s\*:\s\*(.\*)',

'Occupation': r'Occupation\s\*:\s\*(.\*)',

'Issuing Place': r'Issuing Place\s\*:\s\*(.\*)',

'Sex': r'Sex\s\*:\s\*(\w)',

'Issuing Date': r'Issuing Date\s\*[:/\s\w-]\*\n\s\*(\d{2}/\d{2}/\d{4})',

'Expiry Date': r'Expiry Date\s\*[:/\s\w-]\*\n\s\*(\d{2}/\d{2}/\d{4})',

}

data = {'Card Number': card\_number, 'ID number': id\_number}

for field, pattern in patterns.items():

match = re.search(pattern, text, re.IGNORECASE)

if match:

data[field] = match.group(1).strip()

else:

data[field] = None

return data

def main(pdf\_path):

logging.info(f"Converting PDF to images: {pdf\_path}")

images = convert\_pdf\_to\_images(pdf\_path)

logging.info("Extracting text from images.")

text = extract\_text\_from\_images(images)

# Log the extracted text

logging.info("Extracted Text:")

logging.info(text)

logging.info("Text extraction complete. Parsing ID card information.")

id\_data = parse\_id\_card(text)

for key, value in id\_data.items():

print(f"{key}: {value}")

if \_\_name\_\_ == "\_\_main\_\_":

pdf\_path = 'EID - SIVAPRASAD\_NANDYALA.pdf' # Replace with your PDF file path

main(pdf\_path)

```

---

This document provides a comprehensive overview of the requirements and steps for extracting and parsing information from PDF files containing images of Emirates ID cards. The provided script demonstrates the implementation, including PDF conversion, image preprocessing, text extraction, and parsing of relevant fields. Adjustments can be made based on specific needs and observed OCR output.