title: OCR Scan Vision API emoji: 🔍 colorFrom: indigo colorTo: blue sdk: docker pinned: false license: mit

Arabic OCR API

A powerful REST API for extracting Arabic text from images and PDF documents using PaddleOCR. Built with FastAPI and deployed on Hugging Face Spaces.

🔗 Live Demo

Base URL: `https://sharshar1-ocr.hf.space`

🚀 Features

- 🔍 Arabic text recognition optimized with PaddleOCR
- 🖼️ Support for images (JPG, PNG, JPEG, etc.)
- 📄 Support for PDF documents (multi-page)
- 📍 Bounding box coordinates for each detected text
- 📊 Confidence scores for each recognition
- 🌐 RESTful API design
- 🔒 CORS enabled for cross-origin requests

📡 API Endpoints

1. Health Check

Check if the API is running.

Request:

```
GET /health
```

Response:

```
{
  "status": "healthy"
}
```

---

2. API Info

Get API information and status.

Request:

```
GET /
```

Response:

```
{
  "name": "OCR Scan Vision API",
  "status": "ok",
  "pdf_support": true
}
```

---

3. OCR for Images

Extract text from image files.

Request:

```
POST /ocr
Content-Type: multipart/form-data
```

Parameters:

Parameter	Type	Required	Description
`file`	File	Yes	Image file (JPG, PNG, JPEG, etc.)
`min_conf`	Float	No	Minimum confidence threshold (0.0-1.0). Default: 0.0

Example using cURL:

```
curl -X POST "https://sharshar1-ocr.hf.space/ocr" \
  -F "file=@contract.jpg"
```

Example with confidence filter:

```
curl -X POST "https://sharshar1-ocr.hf.space/ocr?min_conf=0.5" \
  -F "file=@contract.jpg"
```

Response:

```
{
  "items": [
    {
      "box_id": 1,
      "text": "عقد إيجار",
      "confidence": 0.9723,
      "bbox": [100, 50, 300, 90]
    },
    {
      "box_id": 2,
      "text": "الطرف الأول",
      "confidence": 0.9456,
      "bbox": [100, 100, 280, 140]
    }
  ],
  "text": "عقد إيجار\nالطرف الأول",
  "total_boxes": 2
}
```

---

4. OCR for PDF Documents

Extract text from PDF files (supports multi-page documents).

Request:

```
POST /ocr-pdf
Content-Type: multipart/form-data
```

Parameters:

Parameter	Type	Required	Description
`file`	File	Yes	PDF file
`dpi`	Integer	No	Resolution for PDF conversion (72-600). Default: 300
`min_conf`	Float	No	Minimum confidence threshold (0.0-1.0). Default: 0.0

Example using cURL:

```
curl -X POST "https://sharshar1-ocr.hf.space/ocr-pdf" \
  -F "file=@contract.pdf"
```

Example with custom DPI:

```
curl -X POST "https://sharshar1-ocr.hf.space/ocr-pdf?dpi=200&min_conf=0.5" \
  -F "file=@contract.pdf"
```

Response:

```
{
  "pages": 3,
  "items": [
    {
      "box_id": 1,
      "text": "عقد إيجار",
      "confidence": 0.9723,
      "bbox": [100, 50, 300, 90],
      "page": 1
    },
    {
      "box_id": 2,
      "text": "الطرف الأول",
      "confidence": 0.9456,
      "bbox": [100, 100, 280, 140],
      "page": 1
    }
  ],
  "text": "--- Page 1 ---\nعقد إيجار\nالطرف الأول\n\n--- Page 2 ---\n...",
  "total_boxes": 25
}
```

📱 Mobile Integration Examples

Flutter (Dart)

```
import 'dart:convert';
import 'package:http/http.dart' as http;

Future<Map<String, dynamic>> extractTextFromImage(String imagePath) async {
  var request = http.MultipartRequest(
    'POST',
    Uri.parse('https://sharshar1-ocr.hf.space/ocr'),
  );

  request.files.add(await http.MultipartFile.fromPath('file', imagePath));

  var response = await request.send();
  var responseBody = await response.stream.bytesToString();

  return json.decode(responseBody);
}

// Usage
void main() async {
  var result = await extractTextFromImage('/path/to/image.jpg');
  print('Extracted text: ${result['text']}');
  print('Total boxes: ${result['total_boxes']}');
}
```

React Native (JavaScript)

```
const extractTextFromImage = async (imageUri) => {
  const formData = new FormData();
  formData.append('file', {
    uri: imageUri,
    type: 'image/jpeg',
    name: 'image.jpg',
  });

  const response = await fetch('https://sharshar1-ocr.hf.space/ocr', {
    method: 'POST',
    body: formData,
    headers: {
      'Content-Type': 'multipart/form-data',
    },
  });

  return await response.json();
};

// Usage
const result = await extractTextFromImage('file:///path/to/image.jpg');
console.log('Extracted text:', result.text);
```

Swift (iOS)

```
import Foundation

func extractText(from imageData: Data, completion: @escaping (Result<[String: Any], Error>) -> Void) {
    let url = URL(string: "https://sharshar1-ocr.hf.space/ocr")!
    var request = URLRequest(url: url)
    request.httpMethod = "POST"

    let boundary = UUID().uuidString
    request.setValue("multipart/form-data; boundary=\(boundary)", forHTTPHeaderField: "Content-Type")

    var body = Data()
    body.append("--\(boundary)\r\n".data(using: .utf8)!)
    body.append("Content-Disposition: form-data; name=\"file\"; filename=\"image.jpg\"\r\n".data(using: .utf8)!)
    body.append("Content-Type: image/jpeg\r\n\r\n".data(using: .utf8)!)
    body.append(imageData)
    body.append("\r\n--\(boundary)--\r\n".data(using: .utf8)!)

    request.httpBody = body

    URLSession.shared.dataTask(with: request) { data, response, error in
        if let error = error {
            completion(.failure(error))
            return
        }

        if let data = data,
           let json = try? JSONSerialization.jsonObject(with: data) as? [String: Any] {
            completion(.success(json))
        }
    }.resume()
}
```

Kotlin (Android)

```
import okhttp3.*
import okhttp3.MediaType.Companion.toMediaType
import okhttp3.RequestBody.Companion.asRequestBody
import java.io.File

suspend fun extractTextFromImage(imageFile: File): String {
    val client = OkHttpClient()

    val requestBody = MultipartBody.Builder()
        .setType(MultipartBody.FORM)
        .addFormDataPart(
            "file",
            imageFile.name,
            imageFile.asRequestBody("image/jpeg".toMediaType())
        )
        .build()

    val request = Request.Builder()
        .url("https://sharshar1-ocr.hf.space/ocr")
        .post(requestBody)
        .build()

    client.newCall(request).execute().use { response ->
        return response.body?.string() ?: ""
    }
}
```

Python

```
import requests

def extract_text_from_image(image_path):
    url = "https://sharshar1-ocr.hf.space/ocr"

    with open(image_path, "rb") as f:
        files = {"file": f}
        response = requests.post(url, files=files)

    return response.json()

# Usage
result = extract_text_from_image("contract.jpg")
print(f"Extracted text: {result['text']}")
print(f"Total boxes: {result['total_boxes']}")
```

Response Schema

OCR Response Object

Field	Type	Description
`items`	Array	List of detected text boxes
`text`	String	Combined text from all boxes
`total_boxes`	Integer	Number of detected text regions
`pages`	Integer	Number of pages (PDF only)

Item Object

Field	Type	Description
`box_id`	Integer	Unique identifier for the text box
`text`	String	Recognized text content
`confidence`	Float	Recognition confidence (0.0-1.0)
`bbox`	Array	Bounding box coordinates [x1, y1, x2, y2]
`page`	Integer	Page number (PDF only)

⚠ Error Handling

HTTP Status Codes

Code	Description
200	Success
400	Bad Request (invalid file)
500	Internal Server Error

Error Response Example

```
{
  "detail": "Invalid image file"
}
```

Technology Stack

- **Framework:** FastAPI
- **OCR Engine:** PaddleOCR (PaddleX)
- **Detection Model:** PP-OCRv5_server_det
- **Recognition Model:** arabic_PP-OCRv5_mobile_rec
- **PDF Processing:** pdf2image, pypdfium2
- **Deployment:** Hugging Face Spaces (Docker)

Notes

- First request may take longer due to model loading (~30-60 seconds)
- Subsequent requests are faster as models are cached
- Higher DPI for PDF results in better accuracy but slower processing
- Use `min_conf` parameter to filter low-confidence results

License

MIT License

Author

Mohamed Sharshar

- GitHub: mohamedsharshar
- Hugging Face: sharshar1