以下に、**C#版のWinForms GUIアプリケーションテンプレート**を提示します。  
これは Python 版と同様の機能（PDF OCR → 伝票番号抽出 → 分割 → CSV出力 → PostgreSQL挿入）を持つ Windows アプリです。

**✅ C# WinForms OCRアプリ 概要**

* .NET 6/7 用 WinForms プロジェクト
* PDF を開いて OCR し、伝票番号（例: DEN12345678）ごとに PDF を分割
* CSV 出力と PostgreSQL への INSERT を実施
* GUI でファイル選択・ログ表示

**🔧 使用ライブラリ（NuGet）**

以下のパッケージをインストールしてください：

Install-Package Tesseract

Install-Package PdfPig

Install-Package Npgsql

Visual Studio の「NuGetパッケージマネージャー」からインストール可能

**📁 プロジェクト構成**

PdfOcrSplitter/

├─ MainForm.cs ← WinForms UIロジック

├─ PdfProcessor.cs ← OCR + DB挿入 + 分割PDFロジック

├─ app.config ← PostgreSQL接続設定（省略可）

├─ tessdata/ ← Tesseract日本語モデル (jpn.traineddata)

├─ Program.cs ← エントリーポイント

**🪟 MainForm.cs（UI側）**

using System;

using System.IO;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace PdfOcrSplitter

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

}

private void btnSelectPdf\_Click(object sender, EventArgs e)

{

var ofd = new OpenFileDialog

{

Filter = "PDFファイル|\*.pdf"

};

if (ofd.ShowDialog() == DialogResult.OK)

{

txtPdfPath.Text = ofd.FileName;

}

}

private async void btnRun\_Click(object sender, EventArgs e)

{

string pdfPath = txtPdfPath.Text;

string outputDir = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "output");

if (!File.Exists(pdfPath))

{

MessageBox.Show("PDFファイルを指定してください");

return;

}

txtLog.AppendText("処理開始...\r\n");

await Task.Run(() =>

{

var processor = new PdfProcessor();

string log = processor.Process(pdfPath, outputDir);

Invoke(() => txtLog.AppendText(log + "\r\n"));

});

txtLog.AppendText("完了しました。\r\n");

}

}

}

**🎨 MainForm.Designer.cs（最低限のUI）**

Visual Studio のデザイナでも追加できますが、コードからもOK：

private Button btnSelectPdf;

private Button btnRun;

private TextBox txtPdfPath;

private TextBox txtLog;

private void InitializeComponent()

{

this.btnSelectPdf = new Button();

this.btnRun = new Button();

this.txtPdfPath = new TextBox();

this.txtLog = new TextBox();

this.SuspendLayout();

this.btnSelectPdf.Text = "PDF選択";

this.btnSelectPdf.Location = new Point(20, 20);

this.btnSelectPdf.Click += new EventHandler(this.btnSelectPdf\_Click);

this.txtPdfPath.Location = new Point(120, 22);

this.txtPdfPath.Width = 400;

this.btnRun.Text = "実行";

this.btnRun.Location = new Point(20, 60);

this.btnRun.Click += new EventHandler(this.btnRun\_Click);

this.txtLog.Location = new Point(20, 100);

this.txtLog.Multiline = true;

this.txtLog.Size = new Size(500, 300);

this.txtLog.ScrollBars = ScrollBars.Vertical;

this.ClientSize = new Size(550, 450);

this.Controls.Add(this.btnSelectPdf);

this.Controls.Add(this.btnRun);

this.Controls.Add(this.txtPdfPath);

this.Controls.Add(this.txtLog);

this.Text = "PDF OCR 分割ツール";

this.ResumeLayout(false);

this.PerformLayout();

}

**🔧 PdfProcessor.cs（処理本体）**

using System;

using System.Collections.Generic;

using System.IO;

using System.Text.RegularExpressions;

using Tesseract;

using PdfPig;

using PdfPig.Writer;

using Npgsql;

public class PdfProcessor

{

public string Process(string pdfPath, string outputDir)

{

var log = "";

Directory.CreateDirectory(outputDir);

var csvPath = Path.Combine(outputDir, "data.csv");

var dennoPages = new Dictionary<string, List<int>>();

var connStr = "Host=localhost;Username=your\_user;Password=your\_password;Database=your\_db";

using var conn = new NpgsqlConnection(connStr);

conn.Open();

using var doc = PdfDocument.Open(pdfPath);

using var writer = new StreamWriter(csvPath);

writer.WriteLine("denno,text");

using var engine = new TesseractEngine(@"./tessdata", "jpn", EngineMode.Default);

for (int i = 0; i < doc.NumberOfPages; i++)

{

var page = doc.GetPage(i + 1);

var text = page.Text;

var match = Regex.Match(text, @"(DEN\d{8})");

if (match.Success)

{

var denno = match.Value;

if (!dennoPages.ContainsKey(denno))

dennoPages[denno] = new List<int>();

dennoPages[denno].Add(i + 1);

writer.WriteLine($"{denno},\"{text.Replace("\"", "\"\"")}\"");

using var cmd = new NpgsqlCommand("INSERT INTO orders (denno, content) VALUES (@d, @c)", conn);

cmd.Parameters.AddWithValue("d", denno);

cmd.Parameters.AddWithValue("c", text);

cmd.ExecuteNonQuery();

log += $"ページ{i + 1}: {denno} 登録\r\n";

}

}

// 分割PDF出力

foreach (var kvp in dennoPages)

{

var outPdf = Path.Combine(outputDir, $"{kvp.Key}.pdf");

using var newPdf = new PdfDocumentBuilder();

foreach (var pageNum in kvp.Value)

{

newPdf.AddPage(doc.GetPage(pageNum));

}

using var fs = File.OpenWrite(outPdf);

newPdf.Build(fs);

}

return log;

}

}

**🧪 PostgreSQL側のテーブル作成SQL**

CREATE TABLE orders (

id SERIAL PRIMARY KEY,

denno TEXT,

content TEXT

);

**✅ 実行方法**

1. Visual Studio でプロジェクトをビルド（Release推奨）
2. 出力 .exe を直接実行
3. 「PDF選択」→「実行」で分割・出力されます

**🧰 必須ファイル補足**

| **ファイル** | **備考** |
| --- | --- |
| tessdata/jpn.traineddata | 日本語OCR対応モデル（Tesseract公式GitHubにあり） |

**📦 配布方法**

* Release ビルド後、bin\Release\netX\ フォルダ内をそのまま zip 配布可能
* .NET 6 or .NET 7 ランタイムがインストールされているPCならOK
* 完全配布型が必要なら ClickOnce または Inno Setup 等でインストーラー化可能

ご希望があれば、このテンプレートを ZIP にして共有用 GitHub リポジトリや Google Drive 形式で提供することも可能です。  
また、**WPF 版**のテンプレートも必要であればご案内します。ご希望はありますか？