Государственное бюджетное профессиональное образовательное учреждение
Свердловской области
«Уральский государственный колледж им. И.И. Ползунова»

РАЗРАБОТКА МОДУЛЯ РАСПОЗНОВАНИЯ ТЕКСТА

Студент группы ИП-402
Кириллов Вадим Максимович

Руководитель
Ярочкина Екатерина Дмитриевна

Екатеринбург
2025

Цель курсового проекта –

Это разработать модуль распознавания текста для веб-приложения.

Для реализации поставленной цели, необходимо решить следующие задачи:

1. Проанализировать существующие OCR-решения
2. Выбрать стек технологий для реализации проекта
3. Разработать архитектуру модуля
4. Реализовать модуль распознавания текста
5. Реализовать клиентское приложение
6. Протестировать работу приложения

OCR

Optical Character Recognition (OCR) или оптическое распознавание текста – это технология с использование которой становится возможна обработка физического носителя информации (бумажных носителей информации и др.) компьютером в машиночитаемом формате.

Современные OCR-системы применяют методы компьютерного зрения и нейронные сети, что позволяет распознавать текст даже на изображениях низкого качества. Технология широко используется в веб-приложениях, мобильных сервисах и корпоративных системах, где требуется быстрое и точное извлечение информации.

Используемые технологии

В ходе выполнения курсового проекта были задействованы следующие технологии:

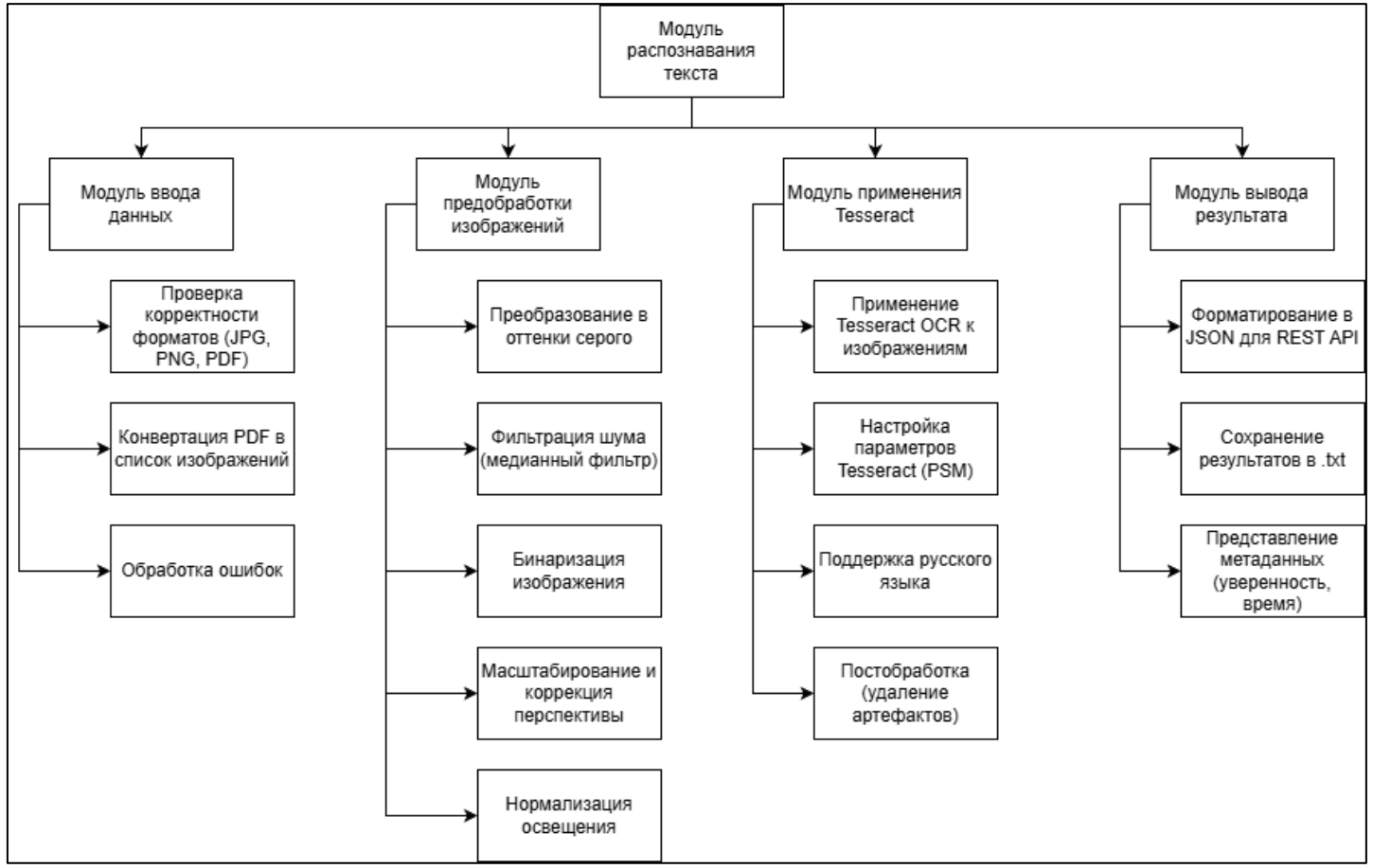
1. Для реализации модуля:
   - Язык программирования : Python 3.x
   - Библиотеки Python: OpenCV, pytesseract, pdf2image, PIL, numpy, json, time, sys.
   - Внешние технологии: Pillow, Tesseract OCR

2. Для реализации серверной части:
   - Язык программирования : Go
   - Встроенные библиотеки: encoding/json, fmt, io, log, **net/http**, os, os/exec, path/filepath.

3. Для реализации клиентской части:
   - Язык программирования: JavaScript
   - Фреймворк: Vue.js

Структура модуля

Модуль состоит из 4 файлов (с расширением .py) и одного конвейера, по которому обрабатываются файлы.



Модуль ввода данных

- Поддержка JPG, PNG, PDF.
- Конвертация PDF в изображения.
- Проверка корректности форматов.
- Обработка ошибок.

```python
from pdf2image import convert_from_path
from PIL import Image

def load_image(file_path):
    """
    Загружает файл (изображение или PDF) и возвращает:
    - объект PIL Image для обычного изображения;
    - список объектов PIL Image для многостраничного PDF.
    Всегда конвертирует в 'RGB', чтобы обеспечить единообразие.
    """
    image = None

    try:
        if file_path.lower().endswith('.pdf'):
            # Конвертируем все страницы PDF в изображения
            pages = convert_from_path(file_path)
            if pages:
                # Приводим каждую страницу к формату RGB
                image = [page.convert('RGB') for page in pages]
        else:
            # Открываем файл изображения
            image = Image.open(file_path).convert('RGB')

    except Exception as e:
        # Эта ошибка будет поймана в main.py
        raise ValueError(f"Ошибка при загрузке файла: {e}")

    if image is None:
        raise ValueError("Не удалось обработать файл. Убедитесь, что это изображение или PDF-документ.")

    return image

```

Предобработка изображений

```python
import cv2
import numpy as np

def deskew(binary_image):
    try:
        coords = np.column_stack(np.where(binary_image > 0))
        angle = cv2.minAreaRect(coords)[-1]

        if angle < -45:
            angle = -(90 + angle)
        else:
            angle = -angle

        (h, w) = binary_image.shape[:2]
        center = (w // 2, h // 2)
        M = cv2.getRotationMatrix2D(center, angle, 1.0)

        rotated = cv2.warpAffine(binary_image, M, (w, h),
                                 flags=cv2.INTER_CUBIC,
                                 borderMode=cv2.BORDER_CONSTANT,
                                 borderValue=0)

        return rotated

    except Exception as e:
        return binary_image
```

```python

def preprocess_image(image):
    """
    Выполняет полную предобработку изображения для OCR.
    """
    # 1. Конвертация из PIL в CV2 (OpenCV)
    img_cv = cv2.cvtColor(np.array(image), cv2.COLOR_RGB2BGR)

    # 2. Конвертация в оттенки серого
    gray = cv2.cvtColor(img_cv, cv2.COLOR_BGR2GRAY)

    # 3. Масштабирование
    h, w = gray.shape
    if w < 1000:
        scale = 1.5
        gray = cv2.resize(gray, (int(w * scale), int(h * scale)), interpolation=cv2.INTER_CUBIC)

    # 4. Удаление шума
    denoised = cv2.medianBlur(gray, 3)

    # 5. Бинаризация (Порог Оцу)
    _, binary_image = cv2.threshold(denoised, 0, 255, cv2.THRESH_BINARY | cv2.THRESH_OTSU)

    # 6. Выравнивание
    deskewed_image = deskew(binary_image)

    return deskewed_image

```

OCR – модуль

- Поддержка русского языка.
- Использование модели LSTM.
- Параметры: OEM 3, PSM 6.
- Извлечение текста и данных уверенности.

```python
import pytesseract
from pytesseract import Output

def get_text_from_image(processed_image):
    """
    Извлекает текст и данные об уверенности из обработанного изображения.
    Принимает CV2 Image (numpy array).
    """

    # --psm 6, работает стабильнее, чем 3
    custom_config = r'-l rus --oem 3 --psm 6'

    # Используем image_to_data для получения детальной информации
    data = pytesseract.image_to_data(processed_image, config=custom_config, output_type=Output.DICT)

    full_text = []
    confidences = []

    for i in range(len(data['text'])):
        if int(data['conf'][i]) > -1:
            word = data['text'][i]
            if word.strip():
                full_text.append(word)
                confidences.append(float(data['conf'][i]))

    return " ".join(full_text), confidences

```

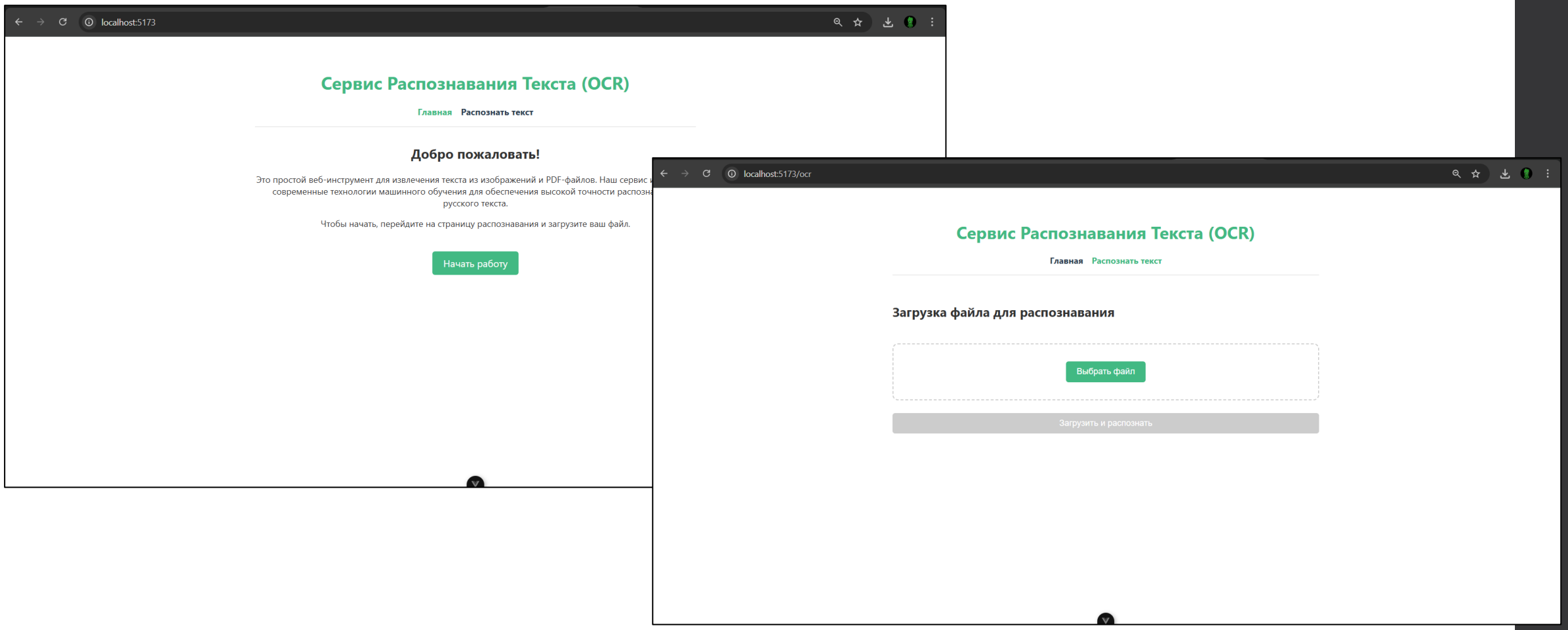
Модуль вывода данных

- Формирование итогового JSON.
- Средняя уверенность распознавания.
- Время обработки.
- Поддержка экспорта текста.

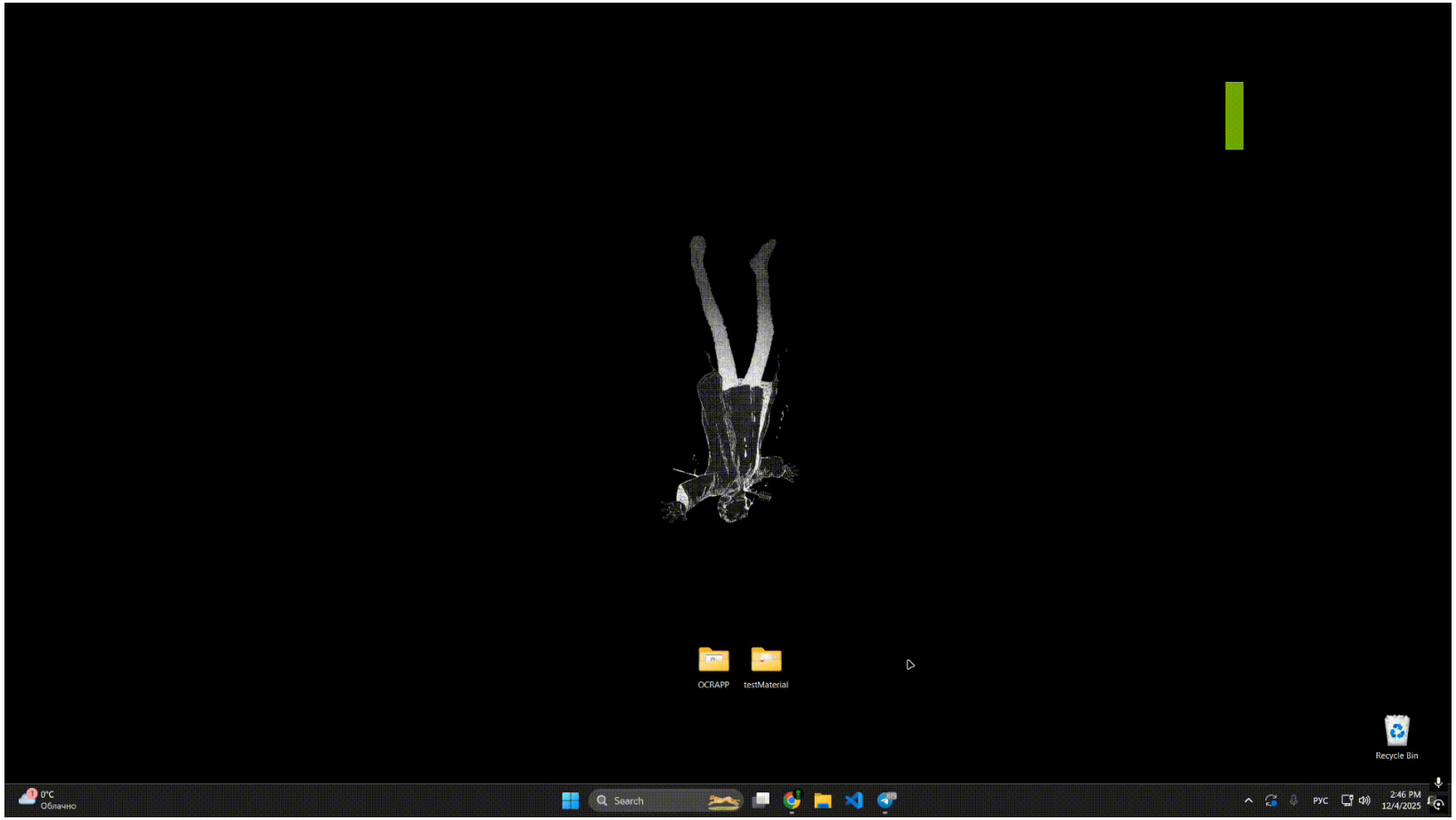
```python
# Модуль 4: Экспорт данных
def export_data(text, confidence_list, start_time):
    """Формирует и выводит итоговый JSON."""
    end_time = time.time()
    processing_time = int((end_time - start_time) * 1000)  # в миллисекундах

    valid_confidences = [conf for conf in confidence_list if conf > 0]
    average_confidence = (
        sum(valid_confidences) / len(valid_confidences)
        if valid_confidences else 0
    )

    result = {
        "text": text,
        "average_confidence": round(average_confidence, 2),
        "processing_time_ms": processing_time
    }

    # ensure_ascii=True, чтобы избежать проблем с кодировкой в Go
    print(json.dumps(result, ensure_ascii=True))

```

Клиентское приложение

Разработан небольшое веб-приложение для работы с OCR-модулем



Тестирование



Заключение

В ходе работы над курсовым проектом:

1. спроектирован и разработан модуль распознавания текста на базе Tesseract OCR;
2. реализовано удобное клиентское приложение на базе Vue.js и серверная часть приложения на Go;
3. протестирована корректность работы модуля распознавания текста. И сделаны выводы на основании уверенности и скорости работы приложения.

Государственное бюджетное профессиональное образовательное учреждение
Свердловской области
«Уральский государственный колледж им. И.И. Ползунова»

РАЗРАБОТКА МОДУЛЯ РАСПОЗНОВАНИЯ ТЕКСТА

Студент группы ИП-402
Кириллов Вадим Максимович

Руководитель
Ярочкина Екатерина Дмитриевна

Екатеринбург
2025