木屋项目部署文档

1. python部分配置

本项目的python部分主要集成了以下功能：

- PDF 文本提取
- OCR 识别
- 文档格式转换（Word ↔ PDF）
- 图像与文本匹配
- 图像描述生成
- 二维码生成
- 思维导图生成（Markdown 转 HTML）
- 图像超分辨率处理

同时通过 RabbitMQ 实现消息队列管理，确保任务的高效处理和响应。功能模块化设计，便于维护和扩展。

1.1. 项目结构

项目python部分的大致结构如下所示：

```
project_root/
├── main.py
├── requirements.txt
├── config.json
├── sr_extend.py
├── qrcode_extend.py
├── markdown_extend.py
├── pdf_extend.py
├── word_extend.py
├── photo_extend.py
├── models/
│   ├── blip_model/
│   │   ├── processor
│   │   └── model
│   ├── clip_model/
│   │   └── ViT-B-32.pth
│   └── sr_model/
│       └── srgan_generator_final.pth
└── qrcodes/
```

- **main.py**：主脚本，负责消息队列的监听和任务调度。
- **sr_extend.py**、**qrcode_extend.py**、**markdown_extend.py**、**pdf_extend.py**、**word_extend.py**、**photo_extend.py**：各功能模块的扩展脚本。
- **config.json**：配置文件，存储项目相关配置。
- **models/**：存放预训练模型的目录。
- **qrcodes/**：生成的二维码图片存放目录。

1.2. 系统要求

- **操作系统**：Windows 10 或更高版本
- **Python**：Python 3.7 及以上
- **硬件**：
  - 推荐使用具备 CUDA 支持的 NVIDIA GPU，以加速深度学习模型的推理
  - 足够的内存

1.3. 安装 Python

确保已安装 Python 3.7 及以上版本。可以通过以下命令检查 Python 版本：

```
python --version
```

如果未安装，请前往 Python 官方网站 下载并安装适合的版本。

1.4. 创建虚拟环境

建议使用虚拟环境以隔离项目依赖。

```
python -m venv chatbot_env
```

激活虚拟环境：

```
chatbot_env\Scripts\activate
```

1.5. 依赖安装

1.5.1. 依赖库安装

本项目所需的python依赖库如下所示

```
pika
json
paddlehub
shutil
opencv-python
qrcode
Pillow
comtypes
pdfplumber
pytesseract
spire.pdf
torch
transformers
```

```
torchvision
clip
```

可直接运行如下命令安装依赖：

```
pip install -r requirements.txt
```

注意：有些库可能需要特定版本，建议根据项目需求调整版本号，依次单独安装各依赖

1.5.2. 特殊依赖安装

- **comtypes**：仅适用于 Windows，确保系统为 Windows 环境。

  ```
  pip install comtypes
  ```

- **spire.pdf**：此库为 Spire.PDF 的 Python 接口，建议访问 Spire.PDF for Python 官方网站，下载并安装相应的包。

- **paddlehub**：安装 PaddlePaddle 后再安装 PaddleHub。

  ```
  pip install paddlepaddle
  pip install paddlehub
  ```

1.6. RabbitMQ 配置

1.6.1 安装 RabbitMQ

前往 RabbitMQ 官方网站 下载并安装适合的版本。推荐使用 RabbitMQ 的稳定版本。

1.6.2. 启动 RabbitMQ 服务

安装完成后，启动 RabbitMQ 服务。默认情况下，RabbitMQ 会在 `localhost:5672` 端口运行。

在当前代码中，RabbitMQ 使用以下凭据（也可修改为自己的用户）：

- **用户名**：`admin`
- **密码**：`123456`

1.6.3. 启用管理插件（可选）

启用 RabbitMQ 管理插件，以通过浏览器管理队列。

```
rabbitmq-plugins enable rabbitmq_management
```

访问 http://localhost:15672 登录 RabbitMQ 管理界面，使用 `admin` 用户名和 `123456` 密码登录。

1.7. 安装外部工具

1.7.1 Tesseract OCR

本项目使用 `pytesseract` 进行 OCR 识别，需要安装 Tesseract OCR 引擎。

- 前往 Tesseract OCR GitHub 下载 Windows 安装包。
- 安装完成后，将 Tesseract 的安装路径（例如 `C:\Program Files\Tesseract-OCR`）添加到系统环境变量 `PATH` 中。
- 下载中文和英文语言包（`chi_sim` 和 `eng`），确保 `tessdata` 文件夹中包含这些语言包。

1.7.2. Markmap CLI

项目中使用了 `markmap` 生成思维导图，需安装 Node.js 和 Markmap CLI。

- **安装 Node.js**：

  访问 Node.js 官方网站 下载并安装适合的版本。

- **安装 Markmap CLI**：

  ```
  npm install -g markmap-cli
  ```

1.7.3. Microsoft Word

确保系统中安装了 Microsoft Word，以支持 `convert_word_to_pdf` 功能。此外，安装相关的 Microsoft Office 库和 COM 接口（通过 `comtypes` 实现）。

1.8. 修改配置文件`config.json`

根据情况修改配置文件 `config.json`，确保所有字段都正确填写。（下为一个示例）

```
{
  "rabbitmq": {
    "host": "localhost",
    "port": 5672,
    "virtual_host": "/",
    "username": "admin",
    "password": "123456"
  },
  "base_url": "https://511b136e.r21.cpolar.top",
  "paths": {
    "blip_model": "models/blip_model",
    "clip_model": "models/clip_model/ViT-B-32.pth",
    "sr_model": "models/sr_model/srgan_generator_final.pth",
    "temp_dist": "F:\\image.png"
```

```
  }
}
```

2. 前端部署

修改前端代码App.vue和local.vue的后端地址为部署的后端地址

```
<script>
    export default {
        globalData: {
            userInfo: {id:100000},//id初始值为调试用,如果数据库不存在该id则无效
            url:'https://6c15e199.r21.cpolar.top',//修改这里的地址
          },
        onLaunch: function() {
            console.log('App Launch')
        },
        onShow: function() {
            console.log('App Show')
        },
        onHide: function() {
            console.log('App Hide')
        }
    }
</script>

<style>
    /*每个页面公共css */
    page{
    }
</style>
```

```
<script>
        import upload from '@/components/upload.vue'
        export default {
            data() {
                return {
                    show:1,
                    path:'',
                    tag: '',
                    message: '',
                    fileLists: null,
                    files: [],
                    filesApp: '',
                    isUploadServer: true,
                    uploadOptions: {
                        // 上传服务器地址，此地址需要替换为你的接口地址
                        //不能用全局变量不然会炸
                        url: 'https://6c15e199.r21.cpolar.top/file/upload',
```

web

直接在uniapp里将项目运行到web浏览器上

app

在uniapp里生成相应apk安装包安装到手机即可

3. 后端部署

3.1. 数据库初始化

操作说明：需先创建数据库并执行表结构脚本。
步骤：

1. **创建数据库**（通过MySQL命令行）：

   ```
   mysql -u your-mysql-username -p
   CREATE DATABASE muwudb CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
   ```

   - 说明：`muwudb` 需与配置文件 `spring.datasource.url` 中的数据库名一致。

2. **执行表结构脚本**：
   运行项目根目录下的 `DB/createdb.sql` 文件，创建表结构及初始数据。
   执行方式：

   - **MySQL命令行**：

     ```
     mysql -u your-mysql-username -p muwudb < DB/createdb.sql
     ```

   - **图形化工具**（如Navicat/Workbench）：
     1. 连接数据库服务器；
     2. 选择数据库 `muwudb`；
     3. 导入 `createdb.sql` 脚本并执行。

3.2. 完整部署配置文档

3.2.1. 全局变量配置（`GlobalVariables.java`）

```
package com.hnu.muwu.config;

import org.springframework.stereotype.Component;

@Component
public class GlobalVariables {
    // 服务器文件存储根目录（部署时需修改为实际路径，如：/data/muwu/files）
    public static final String rootPath = "/your-server-file-path";
```

```
    // 静态资源目录（开发环境保留，生产环境建议用Nginx代理）
    public static final String sharePath = "src/main/resources/static";
}
```

3.2.2. 消息队列配置（MessageQueueHelper.java）

```
public class MessageQueueHelper {
    // RabbitMQ生产环境配置（替换为实际服务器信息）
    private static final String host = "rabbitmq.your-domain.com";
    private static final String username = "mq_user";
    private static final String password = "mq_password";
    // 其他参数保持默认或按实际环境调整
}
```

3.2.3. 应用配置文件（application.properties）

```
# 服务端口
server.port=8082

# 数据库配置（需与createdb.sql中的库名一致）
spring.datasource.url=jdbc:mysql://your-mysql-host:3306/muwudb?
useUnicode=true&characterEncoding=utf8&serverTimezone=Asia/Shanghai
spring.datasource.username=your-mysql-username        # 数据库用户名（需具备表创建权
限）
spring.datasource.password=your-mysql-password        # 数据库密码

# Redis配置（本地开发可默认，生产环境需修改）
spring.data.redis.host=127.0.0.1
spring.data.redis.port=6379

# 邮件服务（以QQ邮箱为例，需获取授权码）
spring.mail.username=your-email@qq.com
spring.mail.password=your-email-auth-code            # 非登录密码，需在邮箱后台申请

# MyBatis配置
mybatis.type-aliases-package=com.hnu.muwu.bean
```