目　录

第一章 需求分析

1.1 需求概述

- 实现 PDF 到 EPUB 的批量转换服务，支持分布式任务调用。
- 转换时需保留原有 PDF 文档的排版样式、标题格式和目录格式。
- 转换后的文档支持保留原文件名和重新命名。
- 转换过程有完整日志记录便于查看转换完成进度。

1.2 项目简要说明

本项目主要对标文档转换网站和文档转换工具，面向有 pdf 文档转换成 epub 文档需求的客户，主要功能是实现 pdf 文档到 epub 文档的高质量转换，同时对扫描版本的 pdf 提供提取转换功能。

1.3 竞品分析

竞品名称	特色功能			
	高精度转换	Ocr 识别转换	支持转换文件重命名	良好的用户交互界面
CleverPDF 网站	×	×	×	√
Convertio 网站	×	×	×	√
PDF24 Tools	√	×	×	√
本项目	√	√	√	√

表 1.1 竞品分析表

第二章 概要设计

2.1 整体模块图

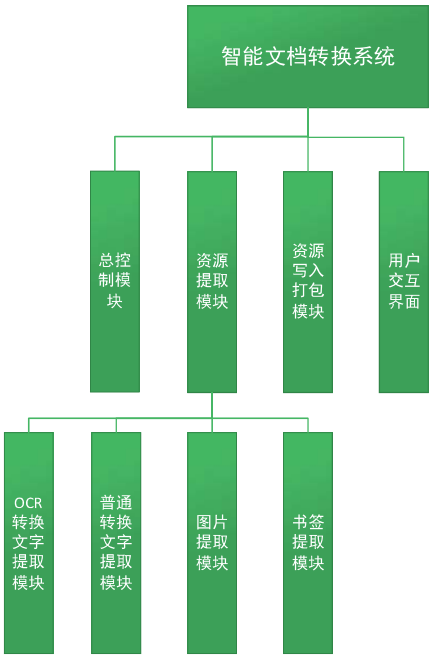


图 2.1 整体模块图

2.2 模块调用图

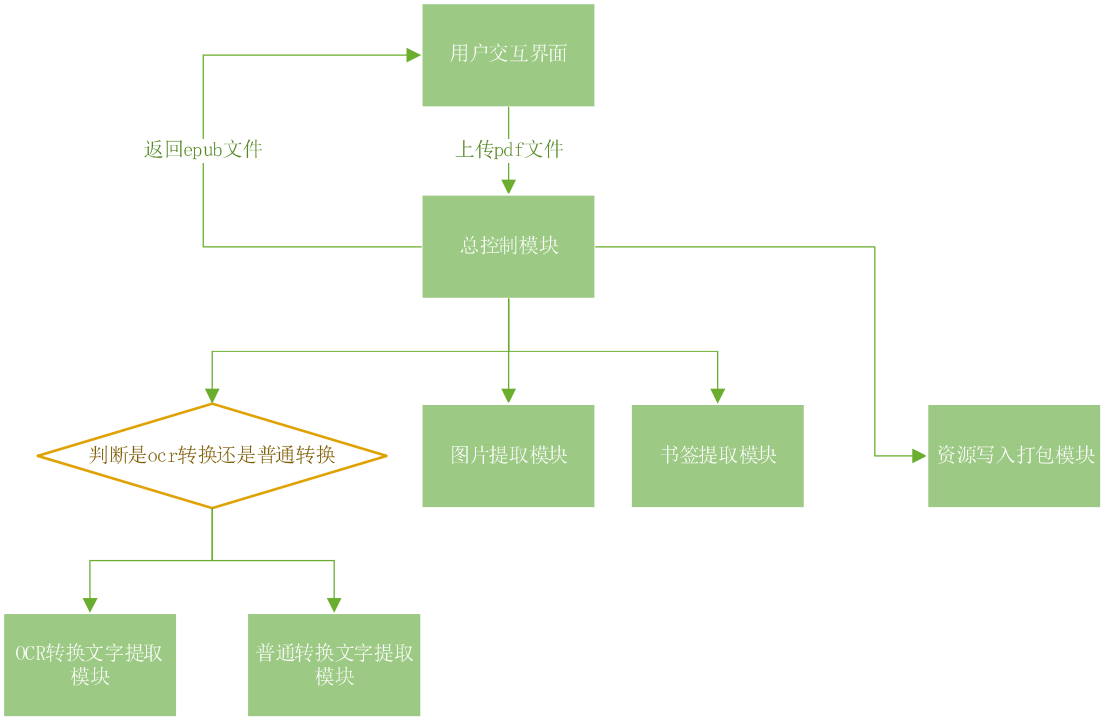


图 2.2 模块调用图

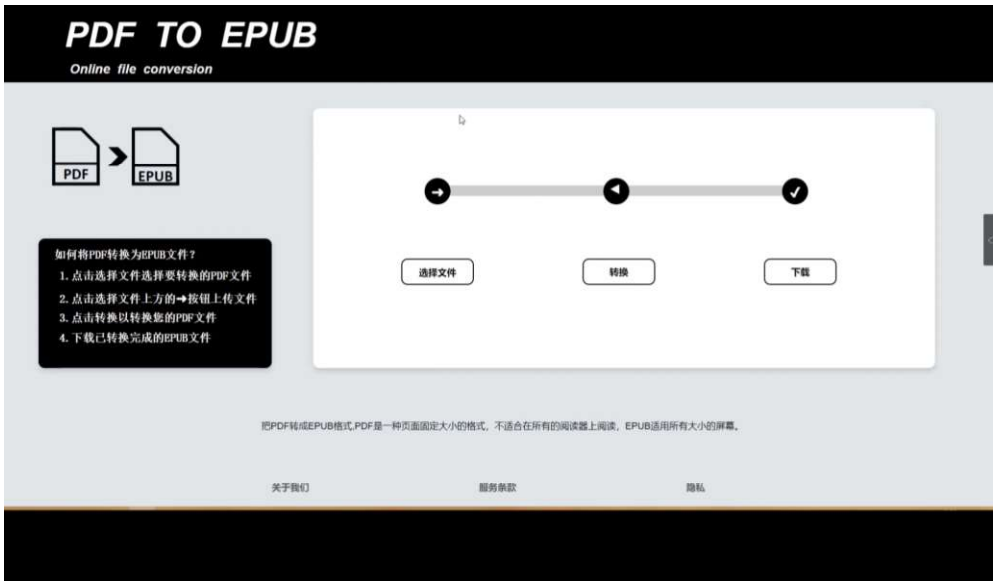
2.3 接口设计

模块名称	接口函数	包含参数	说明
总控制模块	1.PdfConverter() 2.intoEpubNoneExtract()	1.File pdf 2.String title,File output HttpServletRequest requeset	1.传入 pdf 文件 2.传入文件名、输出文件夹和 http 请求
图片提取模块	Extract()	File input,File output	传入文件和输出文件
书签提取模块	PrintBookmark()	PDOutlineNode bookmark,File saveDir	Pdf 文件的书签结点和输出保存文件夹
OCR 转换文字提取模块	1.create() 2.ocrEng()	1.无参数 2.File imgsDir,File output	需要识别的图片文件夹和输出保存文件夹
普通转换文字提取模块	1.PrintTextLocations() 2.extractTextStyle()	1.File input,File output 2.无参数	输入文件夹和输出文件夹
资源写入打包模块	create()	String title,File input,File output,Integer pageNumber,HttpServletRequest request	文件名、输入文件夹、输出文件夹、pdf 页数、http 请求

表 2.1 接口设计表

2.4 界面设计

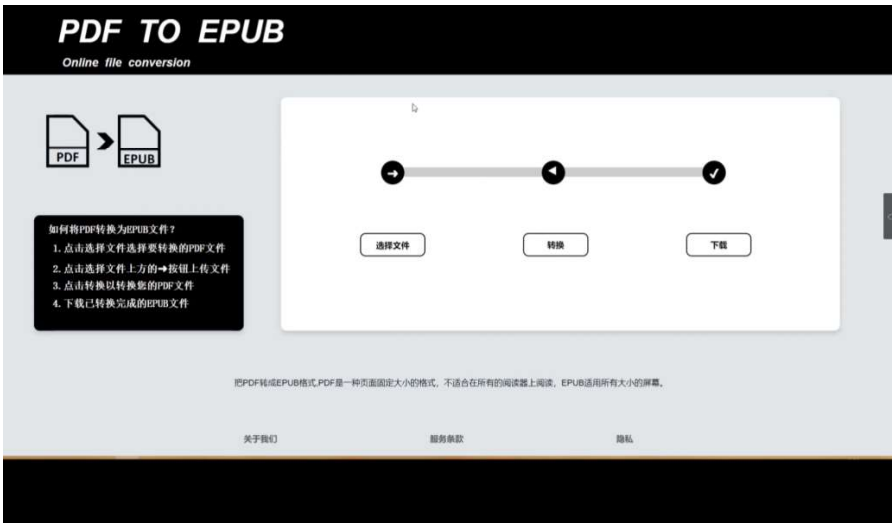
本项目的前端界面设计如下图：



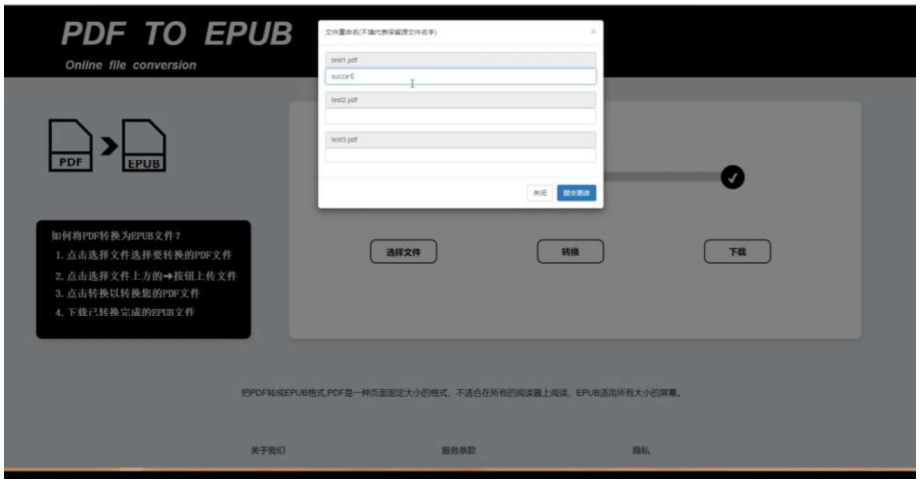
第三章 详细设计

3.1 界面设计

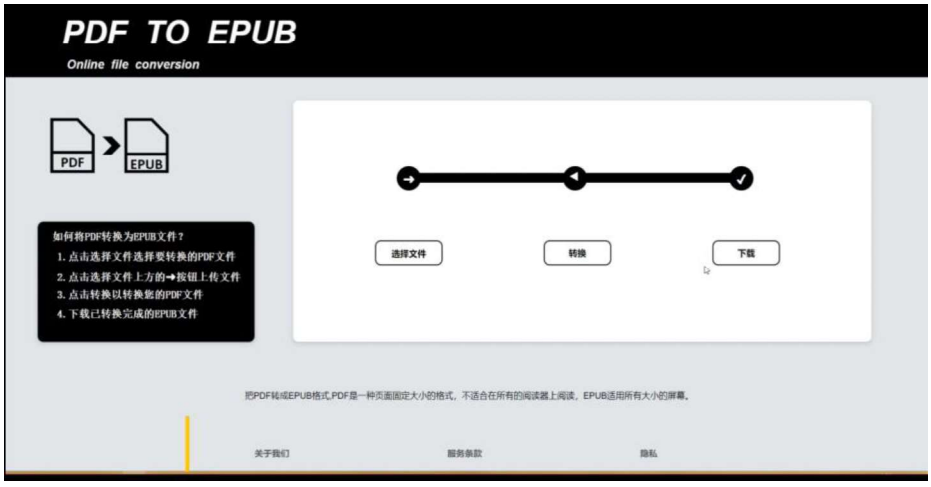
流程 1:进入网站首页



流出 2:上传文件并点击转换，弹出模态框选择是否修改名称



流程 3:等待转换完毕，下载转换文件



3.2 数据库设计

3.2.1 标识符和状态

数据库软件的名称：Mysql 5.7

数据库的名称：labtest

表名	标识符或名称	描述信息	状态
错误表	错误 id	用来保存程序运行时产生的错误信息	使用

表 3.1 标识符和状态表

3.2.2 使用它的模块

模块名称	访问的数据表
图片信息提取模块算法	错误表
文字信息提取模块	错误表
书签提取模块	错误表
资源写入打包模块	错误表
用户交互界面	错误表

表 3.2 模块使用表

3.2.3 逻辑结构设计

字段名	数据类型	长度	主键	非空	描述
id	bigint	100	是	是	主键 id
call_name	varchar	255	-	否	名称
call_time	timestamp	0	-	否	调用时间

call_function_full_name	varchar	255	-	否	调用方法名称
exception_message	longtext	0	-	否	异常信息

表 3.3 errolog 逻辑结构设计表

3.3 关键算法

3.3.1 图片信息提取模块算法

- 算法流程图

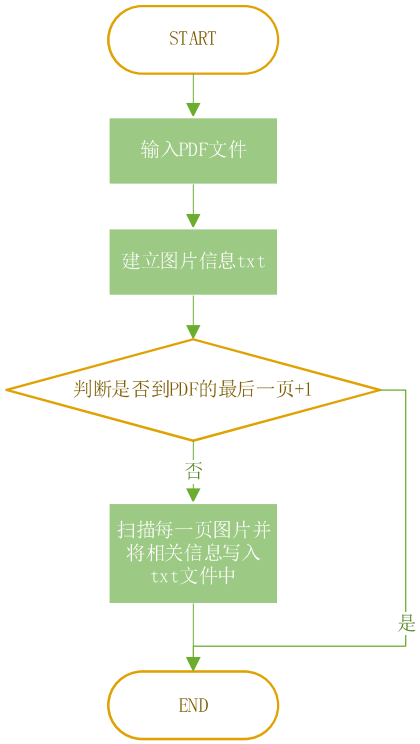


图 3.1 图片信息提取模块流程图

- 核心代码

```
//图片的解析输出
        if (page.extractImages() != null) {
          //输出图片文件
          for (BufferedImage image : page.extractImages()) {
            if (image != null) {
              //指定输出图片名，指定图片格式,后缀自己换
              File Img = new File(ImgPack.getAbsolutePath()+String.format("/image_%d.png",index
++));
              ImageIO.write(image, "PNG", Img);
            }
          }

          //输出图片的格式信息
```

```
            PdfImageInfo[] imageInfo=page.getImagesInfo();

            for (int k=0;k<imageInfo.length;k++){
              //获取指定图片的边界属性
              Rectangle2D rect = imageInfo[k].getBounds();
              //图片的信息内容:页数,宽,高,左上角 x 坐标， 左上角 y 坐标
              int pagenumber=i+1;
              String ImgInfo=String.format("image_%d.png",imgNumber++)+"*"+pagenumber+"*"+rect.g
etWidth()+"*"+rect.getHeight()+"*"+rect.getX()+"*"+rect.getY()+"\n";
              FileWriter imgStylewriter=null;
              imgStylewriter=new FileWriter(imgStyle,true);//以追加的方式写入文件
              imgStylewriter.write(ImgInfo);
              imgStylewriter.flush();
              imgStylewriter.close();
            }
```

3.3.2 普通转换文字信息提取模块算法

- 算法流程图

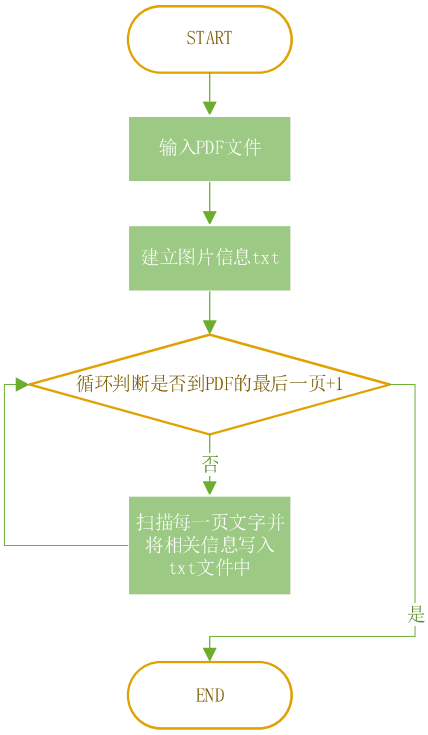


图 3.2 普通转换文字信息提取模块流程图

- 核心代码

```
1.  public void extractTextStyle() throws IOException {
2.  //    PDDocument document = null;
3.      try {
4.        TextStylePack.mkdir();
5.        PDDocument document = PDDocument.load(input);
6.        textPath=TextStylePack.getAbsolutePath()+String.format("/pageTextStyle.txt");
```

```
7.          Text=new File(textPath);
8.
9.          Text.createNewFile();
10.         PDFTextStripper stripper = new PrintTextLocations(input,output);
11.         stripper.setSortByPosition(true);
12.         stripper.setStartPage(0);
13.         stripper.setEndPage(document.getNumberOfPages());
14.         Writer dummy = new OutputStreamWriter(new ByteArrayOutputStream());
15.         stripper.writeText(document, dummy);
16.
17.         BufferedWriter writer=null;
18.         FileWriter fileWriter=new FileWriter(Text);
19.         writer=new BufferedWriter(fileWriter);//将每页的文本写入 txt 文件
20.
21.
22.         for (String textInfo: lines) {
23.           writer.write(textInfo);
24.           writer.flush();//强制全部写出缓冲区
25.         }
26.         lines.clear();
27.
28.       }catch (IOException e){
29.         e.printStackTrace();
30.         ErrorLog errorLog = new ErrorLog();
31.         StringWriter sw = new StringWriter();
32.         PrintWriter pw = new PrintWriter(sw);
33.         e.printStackTrace(pw);
34.         errorLog.setCallName("extractTextStyle");
35.         errorLog.setCallFunctionFullName("com.example.pdfconvertepub.convert.extract.PrintTextLocations");
36.         errorLog.setExceptionMessage(sw.toString());
37.         errorLogMapper.insert(errorLog);
38.       } finally {
39.         if (document != null) {
40.           document.close();
41.         }
42.       }
43.     }
44.     //getX()x 坐标,getY()y 坐标,getFontSize()获取文字尺寸,pagenumber 出现页号,getUnicode()获取文字
45.     @Override
46.     protected void writeString(String string, List<TextPosition> textPositions) throws IOException {
47.       for (TextPosition text : textPositions) {
48.
```

```
49.         String textInfo=text.getX() + "*" + text.getY()+ "*" + text.getFontSize()+ "*" + text.g
    etHeightDir() + "*" + text.getWidthDirAdj()
50.             +"*"+getCurrentPageNo()+ "*" + text.getUnicode()+"\n";
51.         lines.add(textInfo);
52.       }
53.
54.     }
```

3.3.3 OCR 转换文字信息提取模块算法

- 算法流程图

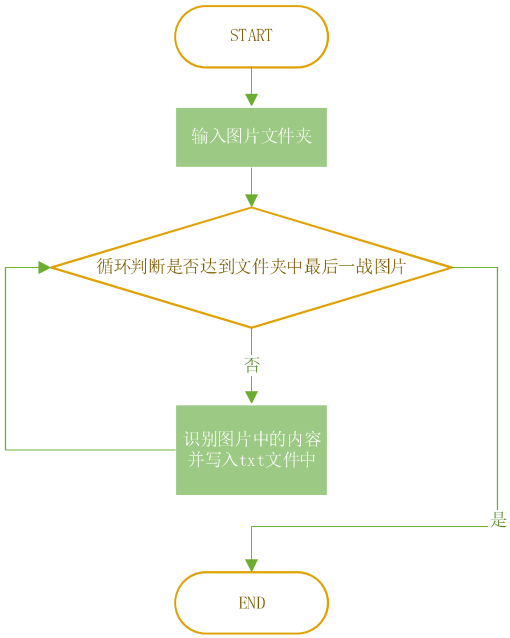


图 3.3 OCR 转换文字信息提取模块流程图

- 核心代码

```
public void create() throws IOException, TesseractException {
    Tesseract tesseract=new Tesseract();
    tesseract.setDatapath(tessDataPathString);
    tesseract.setLanguage("chi_sim");
    int count=0;
    for (File file:listFiles()){
        String pdfInfo = tesseract.doOCR(file);
        FileWriter writer=null;
        String textPath=TextStylePack.getAbsolutePath()+String.format("\\pageText_%
d.txt",count);
        File Text=new File(textPath);
        writer=new FileWriter(Text);
        writer.write(pdfInfo);
        writer.flush();;
        writer.close();
```

```
        count++;
    }


}

private List<File> listFiles(){
    File[] files = imgsDir.listFiles((dir, name) -> name.toLowerCase().endsWith(".p
ng"));
    List<File> sorted = Lists.newArrayList(files);
    sorted.sort(Comparator.comparing(File::toString));
    return sorted;
}
```

3.3.4 目录信息提取模块算法

- 算法流程图

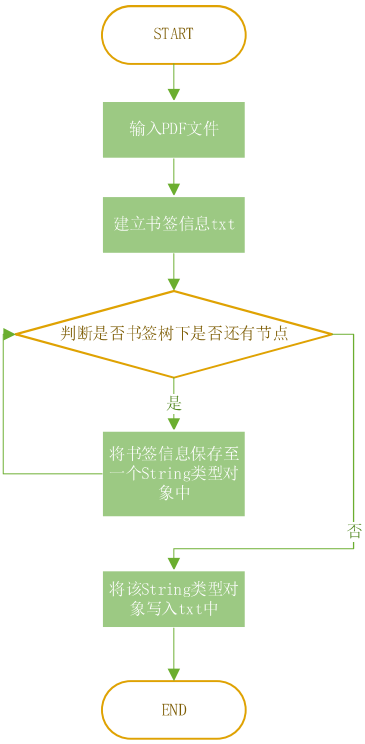


图 3.4 书签信息提取模块流程图

- 核心代码

```
1.  public void printBookmark(PDOutlineNode bookmark,File saveDir,String indentation) throws I
    OException
2.    {
3.      File bookMark=new File(saveDir.getAbsolutePath()+"/bookmark.txt");
4.      String bookMarkInfo=new String();
5.      PDOutlineItem current = bookmark.getFirstChild();
6.
7.      while(current != null)
```

```
8.      {
9.        int pages =0;
10.       int index =0;
11.       if (current.getDestination() instanceof PDPageDestination)
12.       {
13.
14.         PDPageDestination pd = (PDPageDestination) current.getDestination();
15.         pages = (pd.retrievePageNumber() +1);
16.         index = (indentation.length());
17.       }
18.       if (current.getAction() instanceof PDActionGoTo)
19.       {
20.         PDActionGoTo gta = (PDActionGoTo) current.getAction();
21.         if (gta.getDestination() instanceof PDPageDestination)
22.         {
23.           PDPageDestination pd = (PDPageDestination) gta.getDestination();
24.           pages = (pd.retrievePageNumber() +1);
25.           index = (indentation.length());
26.         }
27.       }
28.       if (pages ==0) {
29.         bookMarkInfo= index+"*"+current.getTitle()+"*"+pages+"\n";
30.         lines.add(bookMarkInfo);
31.       }
32.       else{
33.         bookMarkInfo=index+"*"+current.getTitle()+"*"+pages+"\n";
34.         lines.add(bookMarkInfo);
35.
36.         //System.out.println( indentation +"  "+index+"   "+current.getTitle() +"  "+ pages);
37.
38.       }
39.
40.       printBookmark(current, saveDir, "  "); // 递归调用
41.       current = current.getNextSibling();
42.     }
43.     //写文件
44.     BufferedWriter writer=null;
45.     writer=new BufferedWriter(new FileWriter(bookMark));//将每页的文本写入 txt 文件
46.     for (String textInfo: lines) {
47.       writer.write(textInfo);
48.
49.       writer.flush();//强制全部写出缓冲区
50.     }
```

3.3.5 资源写入打包模块算法

- 算法流程图

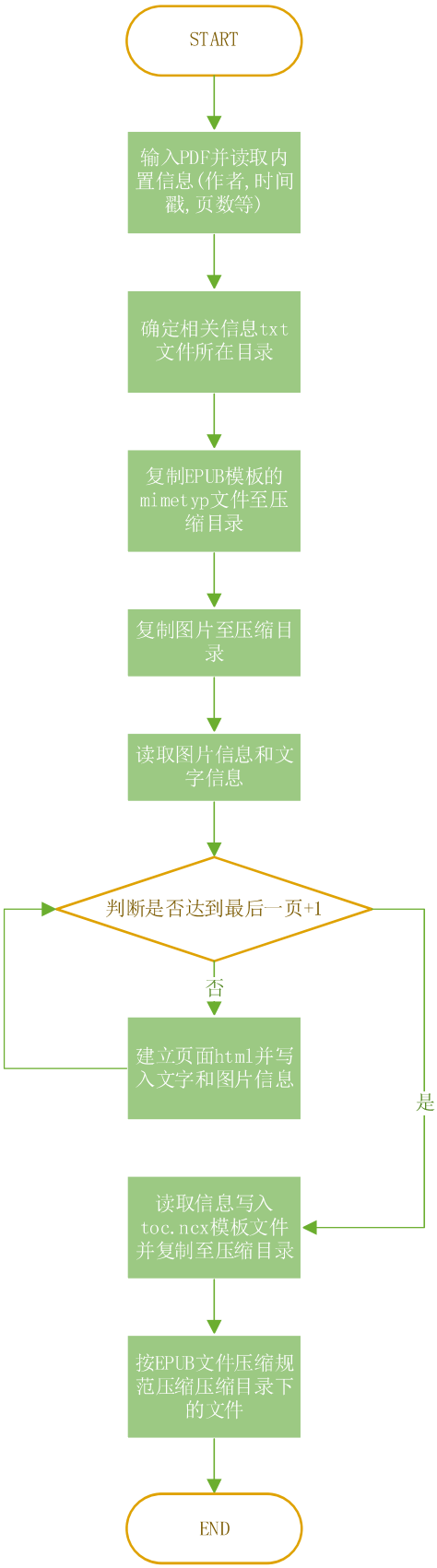


图 3.5 资源写入打包模块流程图

- 核心代码

```
public void create(String title, File input, File output, Integer pageNumber,String Name,HttpServ
letRequest request) throws IOException {
    this.pageNumber=pageNumber;
    timestamp = DateTimeFormat.forPattern("yyyy-MM-dd'T'hh:mm:ssSZZ").print(DateTime.now());
    uuid = UUID.randomUUID().toString();
    this.title = title;
    this.imgsDir = new File(input.getAbsolutePath()+"/img");//图片
    this.textStyleDir =new File(input.getAbsolutePath()+"/textStyle");//文字样式
    this.imgsStyleDir =new File(input.getAbsolutePath()+"/imgstyle");//图片样式
    this.bookmark=new File(input.getAbsolutePath()+"/bookmark.txt");
    this.pdfName=Name;
    this.outputDir=output;
    try {
      basedir = File.createTempFile(uuid, "");
      basedir.delete();
      basedir.mkdirs();

    } catch (IOException e) {
      e.printStackTrace();
      ErrorLog errorLog = new ErrorLog();
      StringWriter sw = new StringWriter();
      PrintWriter pw = new PrintWriter(sw);
      e.printStackTrace(pw);
      errorLog.setCallName("create");
      errorLog.setCallFunctionFullName("com.example.pdfconvertepub.convert.epub.EpubCreatorT
ext");
      errorLog.setExceptionMessage(sw.toString());
      errorLogMapper.insert(errorLog);
    }
    classLoader = getClass().getClassLoader();

    writeMimetype();
    copyImages();//复制图片
    copyStandardFilez();
    createIndex();
    createOPFFile();
    createTOC();
    new_pack(basedir.getAbsolutePath(), request);//epub 的打包

    copyEpubToOutput(request,outputDir);
//    FileUtils.deleteDirectory(basedir);
  }
```

第四章 测试报告

4.1 测试环境

硬件环境	应用服务器	客户端
硬件配置	CPU：Intel(R) Xeon(R) Platinum 8269CY Memory： 2GB HD：20G	CPU: Intel(R) I7-9750H Memory:16GB HD:512G
软件配置	OS:centOS 7 JDK:1.8.0_311 SQL:MySQL 5.7	浏览器:edge,Chrome
网络环境	1M WAN	10M WAN

表 4.1 测试环境表

4.2 测试方案

序号	测试用例	预期结果	实测结果	测试状态	错误类型
测试单元:图片信息提取模块					
1	带有图片的 PDF 文档	输出图片相关信息 txt	输出图片相关信息 txt	1	-
2	不带有图片的 PDF 文档	输出图片关信息 txt，但无内容	输出图片关信息 txt，但无内容	1	-
3	空	输出为空	报错	3	1

表 4.2 图片信息提取模块测试表

序号	测试用例	预期结果	实测结果	测试状态	错误类型
测试单元:普通转换文字信息提取模块					
1	带有文字的 PDF 文档	输出每一页文字的坐标的 txt 集合	输出每一页文字的坐标的 txt 集合	1	-

2	不带有文字的 PDF 文档	输出文字相关信息 txt 集合但所有内容为空	输出文字相关信息 txt 集合但所有内容为空	1	-
3	空	输出为空	报错	3	1

表 4.3 普通转换文字信息提取模块测试表

序号	测试用例	预期结果	实测结果	测试状态	错误类型
测试单元:OCR 转换文字信息提取模块					
1	图片	输出识别文字信息 txt	输出识别文字信息 txt	1	-
2	非图片	输出为空	报错	3	4
3	空	输出为空	输出为空	1	-

表 4.4 OCR 转换文字信息提取模块测试表

序号	测试用例	预期结果	实测结果	测试状态	错误类型
测试单元:书签信息提取模块					
1	包含书签的 PDF 文件	输出书签相关信息 txt	输出书签相关信息 txt	1	-
2	不包含书签的 PDF 文件	输出书签相关信息 txt 但内容为空	输出书签相关信息 txt 但内容为空	1	-
3	空	输出为空	报错	1	4

表 4.5 书签信息提取模块测试表

序号	测试用例	预期结果	实测结果	测试状态	错误类型
测试单元:资源写入打包模块					
1	资源文件	Epub 文件	Epub 文件	1	-
2	空	输出为空	报错	3	4

表 4.6 资源写入打包模块测试表

序号	测试用例	预期结果	实测结果	测试状态	错误类型
测试单元:用户交互界面					
1	单个 PDF 文件	Epub 文件	Epub 文件	1	-
2	多个 PDF 文件	Zip 包	Zip 包	1	-
3	空	无法转换	无法转换	1	-

表 4.7 用户交互界面测试表

测试状态:1-测试合格 2-测试不合格 3-测试有错待处理 4-未测试
错误类型说明:1-功能错/缺 2-语法错 3-注释错 4-结果错 5-窗口错 6-逻辑错 7-链接错 8-变量定义错 9-控件错 11-自定义函数/类错 12-界面错

4.3 修正记录

序号	模块名	修正记录
1	图片信息提取模块	在构造函数处添加判空模块
2	普通转换文字信息提取模块	在构造函数处添加判空模块
3	书签信息提取模块	在总控制模块处添加判断 pdf 文件模块
4	资源写入打包模块	在总控制模块处添加判空模块

表 4.8 修正记录表

4.4 技术指标

运行速度	运行速度主要受机器性能和文档复杂程度影响,在 cpu 为 i7-9750h，内存为 16g 的物理机上，转换页数为 100 页且具有目录结构的文档，耗费时间为 2348ms。
安全性	本项目面向用户仅提供上传和下载功能，且严格限制文件类型，具有良好的安全性
扩展性	本项目提供了多种接口以供二次开发
部署方便性	本项目已打包为 jar 包，仅需配置好服务器环境便能运行

表 4.9 技术指标表

第五章 安装及使用

5.1 安装环境要求

- 操作系统:CentOS 7-2009
- 数据库:MySQL 5.7
- Java JDK:jdk 1.8.0_311

5.2 典型使用流程

1. 首先在服务器(操作系统建议选择 centOS7)安装 java(jdk 版本:1.8.0_311)。
2. 建议安装宝塔面板(方便管理服务器，当然自己安装 MySQL 也行)，然后在宝塔面板中安装 mysql,新建数据库名为 labtest,上传 errlog.sql 建表。
3. 上传该项目的 jar 包至服务器,用 java -jar 命令运行该项目。
4. 在浏览器输入“服务器地址+:8088/epub”即可访问到转换页面。
5. 如需查看错误表的内容可以登录宝塔面板查看。

第六章 项目总结

6.1 项目协调与任务分解

姓名 / 任务内容	黄培	王子涵	李浩
组织协调	70%	15%	15%
作品创意	40%	30%	30%
竞品分析	10%	45%	45%
方案设计	30%	30%	40%
技术实现	40%	30%	30%
文献阅读	10%	45%	45%
测试分析	10%	45%	45%

表 6.1 任务分解表

6.2 克服困难

1. 对 pdf 结构和 epub 结构了解甚少，通过查阅知网相关文献以及 pdf 官网，弥补了知识的不足。

2. 项目初始只针对内容为全图片的 pdf 文件转换为 epub，后续通过分类获取图片和文字坐标及文字等信息，进行算法的调整，从而实现了 pdf 文件各种类型的内容的转换。
3. 如何生成多个 html 文件和分页操作，通过优化算法和获取过程中加入 pagenumber 参数，进行分页和生成相应的 html 文件。
4. 如何处理 html 文件与 epub 文件格式的兼容，通过查阅资料，设置相应的比例，完成文字格式的兼容。
5. 如何压缩提取的各类型文件，转换为 epub，我们尝试了写压缩算法，但效果不佳，后续也采用了 windows 命令，后续优化使用了相应的 epub 压缩函数。
6. 如何获取进度条参数和进行单文件下载和多文件压缩下载，以及单文件上传和多文件上传，通过查阅 csdn 和各类型技术网站，获取相应算法，充分了解相应算法各函数的使用，灵活的调用函数，成功实现了相应功能。
7. 如何部署到 ILinux 平台。
8. 如何进行 Ocr 的文字识别功能，通过不断的测试，最后可进行中文和英文的识别，并提高 Ocr 识别模式。

6.3 水平提升

最初采用的提取 PDF 内容的方法是 OCR，但此方法对中文提取的正确率不高，且只能提取文字，不能提取图片，这样不能保证转换后 EPUB 文件的文本格式和原 PDF 文本格式相同或相近。后续我们采用了提取 PDF 文字、位置、大小、所在页数及图片大小、位置、所在页数，这种方法可以最大程度保持 EPUB 文本和原 PDF 文本格式相同，且转换速度有了较大提升。

6.4 升级演进

后续我们项目将提供针对 pdf 中图片进行识别处理，及多类型文档转换为 epub，例如：敏感信息图片打码，模糊图片变清晰，epub 文件排版结构的优化。

6..5 商业推广

1. 提供极简的页面和酷炫的动画。
2. 提供免费的 pdf 文件转换为 epub 功能，吸引顾客，和进行推广。
3. 提供特定功能的 cr 识别模式，针对 pdf 文件进行图片文字的识别。
4. 提供多平台的使用，电脑段和手机端。

参考文献

[1]张逸. PDF 到 EPUB 电子书格式转换工具的设计与实现[D]. 西安电子科技大学, 2016.
[2]叶兰. 电子图书新规范 EPUB3.0 及其应用[J]. 图书馆杂志, 2012, 31(8):7.
[3]张杰. 数字出版中 PDF 与 EPUB 格式转换技术研究[D]. 杭州电子科技大学, 2016.
[4]牛永洁, 薛苏琴. 基于 PDFBox 抽取学术论文信息的实现[J]. 计算机技术与发展, 2014, 24(12):4.
[5]申冉, 宋俊玲. EPUB 格式电子书的制作[J]. 出版与印刷, 2015(4):3.
[6]林青, 李健. PDF 文档 HTML 化中文本重排问题研究[J]. 电脑与信息技术, 2014, 22(3):4.