基于Web的编程教学演示系统设计与实现

报告人：陈年域

目录 CONTENTS

PART 1

选题背景及意义

PART1 选题背景及意义

TITLE

基于Web的编程教学演示系统主要用于实现用户在线编辑代码并显示编译结果、提交Pdf文件提取关键词生成相应的词云帮助用户了解文件的关键内容、提交Pdf/Ppt文件转换成html的功能，根据系统要实现的功能可以把系统划分为在线编译部分、词云生成部分、Pdf/Ppt课件转换三个主要部分。

在线编译器

在线编译器是指在不需要安装任何的编译环境和软件的情况下，就能够编写、编译并执行程序。用户不需要承担传统编译器所带来的内存消耗、硬盘存储空间的消耗就能开始快速编写程序。用户还可以到任何一个可以上网的 PC 上编写代码，不需要费时去安装和配置传统的编译环境

词云

“词云”就是对文本中出现频率较高的“关键词”予以视觉上的突出，形成“关键词云层”或“关键词渲染”（即词云图），从而过滤掉大量的文本信息，使浏览网页者只要一眼扫过文本就可以领略文本的主旨。

PART1选题背景及意义

在线编译器

更换现如今在线编译器种类繁多，但仍存在没有代码提示，不能折叠代码等问题，本系统采用Ace editor实现代码提示，折叠等功能。

词云

现如今大部分的在线词云生成工具都需要手动输入文字提交生成，不支持文本文件（如Pdf文件）提取文件生成词云，本系统提供提交Pdf文件生成词云的功能，方便用户快速了解Pdf文件的内容。

Pdf/Ppt课件转换

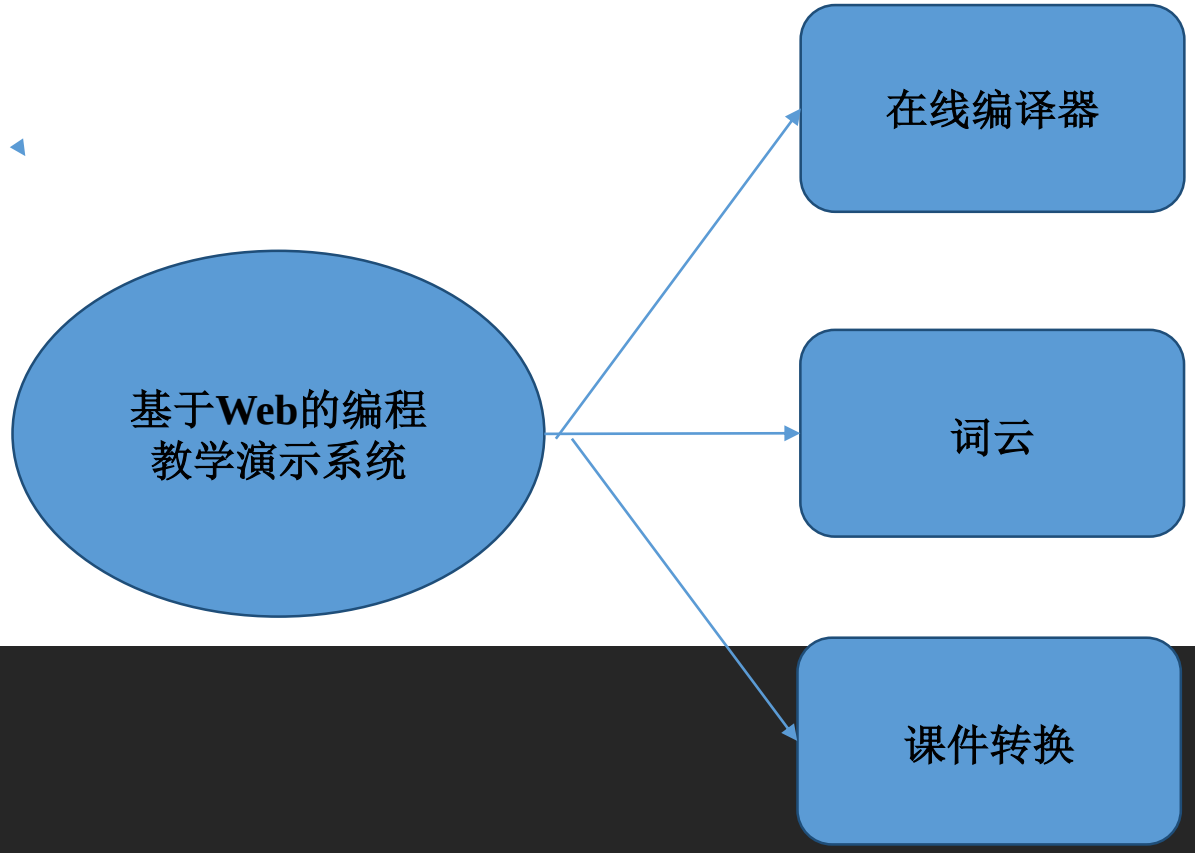
系统能够将用户上传的pdf/ppt课件转换为html网页方便用户在线交流学习

PART 2

系统介绍

PART2系统介绍

基于Web的编程教学演示系统主要可以分为在线编译模块，词云模块，以及课件转换模块（Pdf/Ppt课件转换为Html）三个主要模块，采用基于Python的Flask作为Web系统的应用框架，并选用Pycharm作为开发工具。



PART2系统介绍

在线编译

在线编译模块采用Ace编辑器，ACE 是一个开源的、独立的、基于浏览器的代码编辑器，可以嵌入到任何web页面或JavaScript应用程序中。ACE支持超过60种语言语法高亮，并能够处理代码多达400万行的大型文档, ACE在性能和功能上可以媲美本地代码编辑器。系统支持Python/Java/Html/javascript的在线编译。

PART2系统介绍

词云

词云图，也叫文字云，是对文本中出现频率较高的“关键词”予以视觉化的展现，词云图过滤掉大量的低频低质的文本信息，使得浏览者只要一眼扫过文本就可领略文本的主旨。词云模块主要使用了PdfMiner库、Jieba库、以及Wordcloud库。

PART2系统介绍

1

Jieba

Jieba 库是一个基于Python优秀的中文分词第三方库，对中文有着很强大的分词能力。

2

PDFMiner

PDFMiner是一种从PDF文档中提取信息的工具， 与其他PDF相关工具不同，它完全专注于获取和分析文本数据。PDFMiner允许人们获取页面中文本的确切位置，以及字体或线条等其他信息。

3

Wordcloud

Wordcloud库是基于Python的优秀的词云展示第三方库

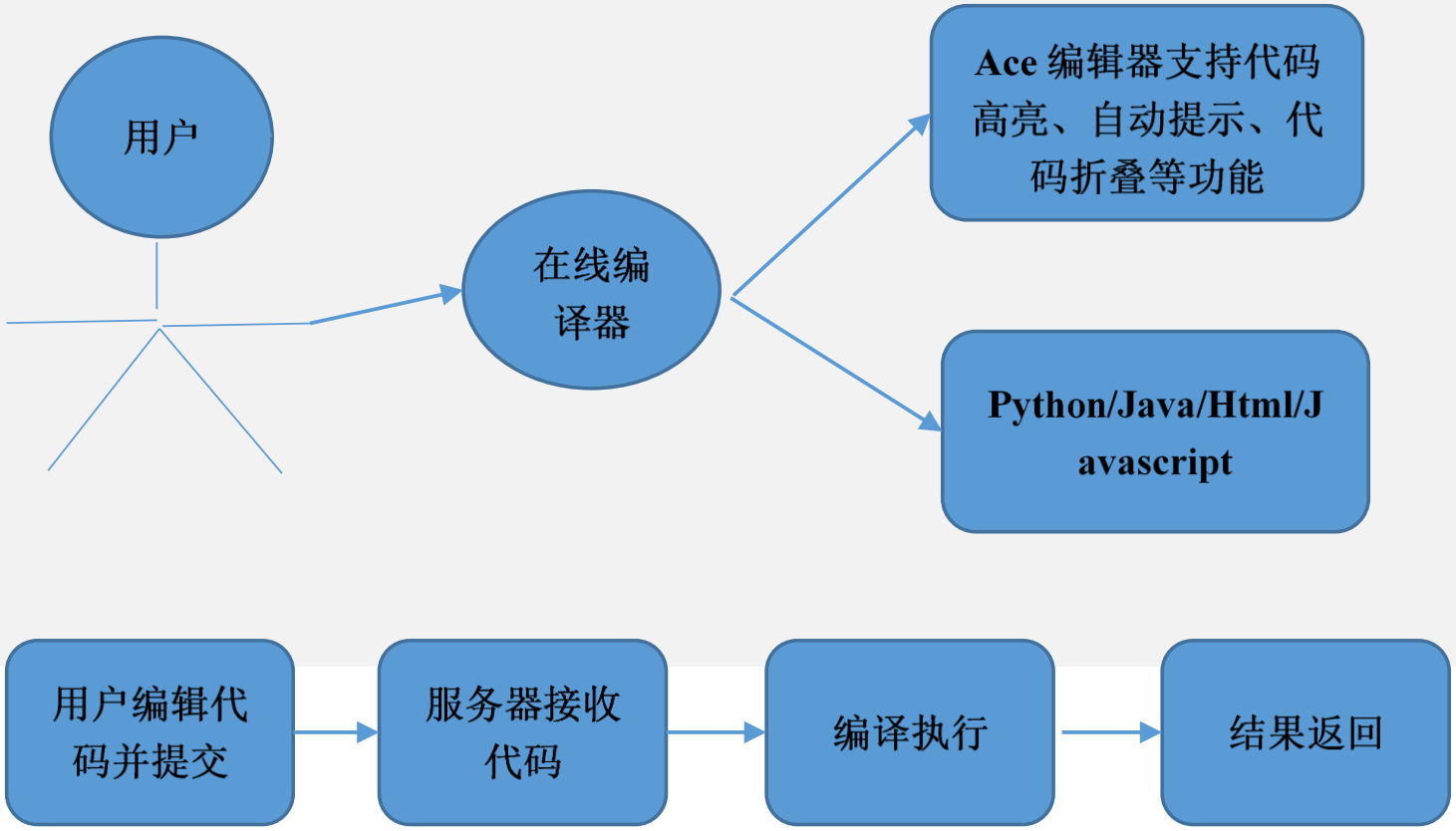
PART 3

完成情况

PART3完成情况

在线编译模块

Python/Java的编译实现：将用户输入的代码放在<form>表单中提交，写入文件后经由系统安装的Python/Java编译器编译后返回编译结果，输入框部分采用Ace编辑器 实现代码自动提示、高亮、代码折叠等功能。

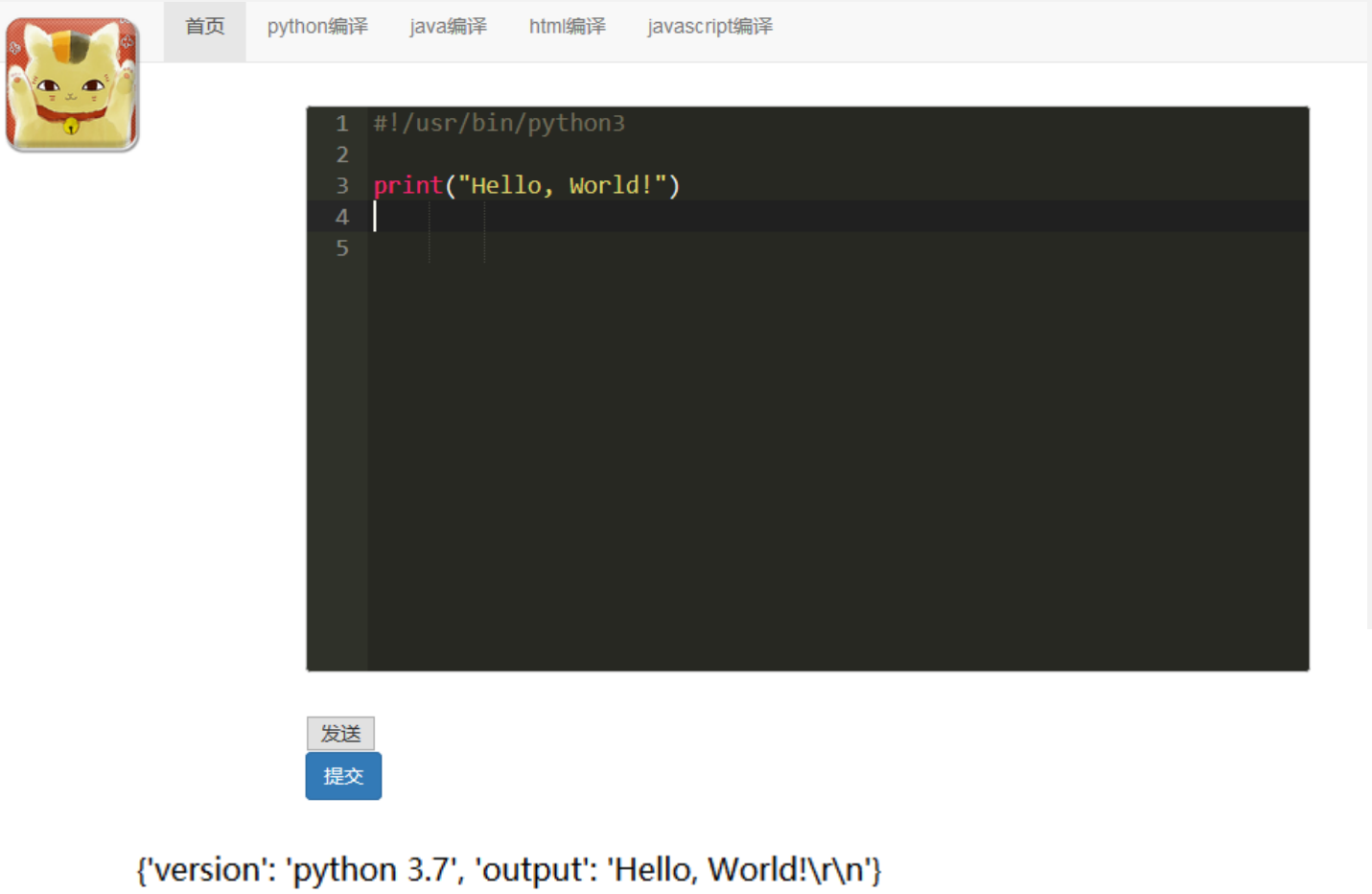


在线编译流程图

PART3完成情况

在线编译模块

在嵌入Ace编辑器的过程中发现Ace编辑器中的内容无法放在form表单中提交（内容为空），故现将Ace中的代码发送至另一普通编辑框，再进行提交。由于在HTML中的多个空格一般会被视为一个故发送时会失去代码的原有格式，查找资料后使用pre标签保持代码的格式不变。



```java
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

发送

提交

编译结果

```
{'output': 'Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF-8\nHello World\r\n', 'code': 'Success
'}
```

Java编译效果图

PART3完成情况

```
def main(code):
    r = dict()
    r["version"] = get_version()
    pyname = get_pyname()
    fpath = write_file(pyname, code)
    try:
        # subprocess.check_output 是 父进程等待子进程完成，返回子进程向标准输出的输出结果
        # stderr是标准输出的类型
        outdata = decode(subprocess.check_output([EXEC, fpath], stderr=subprocess.STDOUT))
    except subprocess.CalledProcessError as e:
        # e.output是错误信息标准输出
        # 错误返回的数据
        r["code"] = 'Error'
        r["output"] = decode(e.output)
        return r
    else:
        # 成功返回的数据
        r['output'] = outdata
        r["code"] = "Success"
        return r
```

Python编译主要代码

PART3完成情况

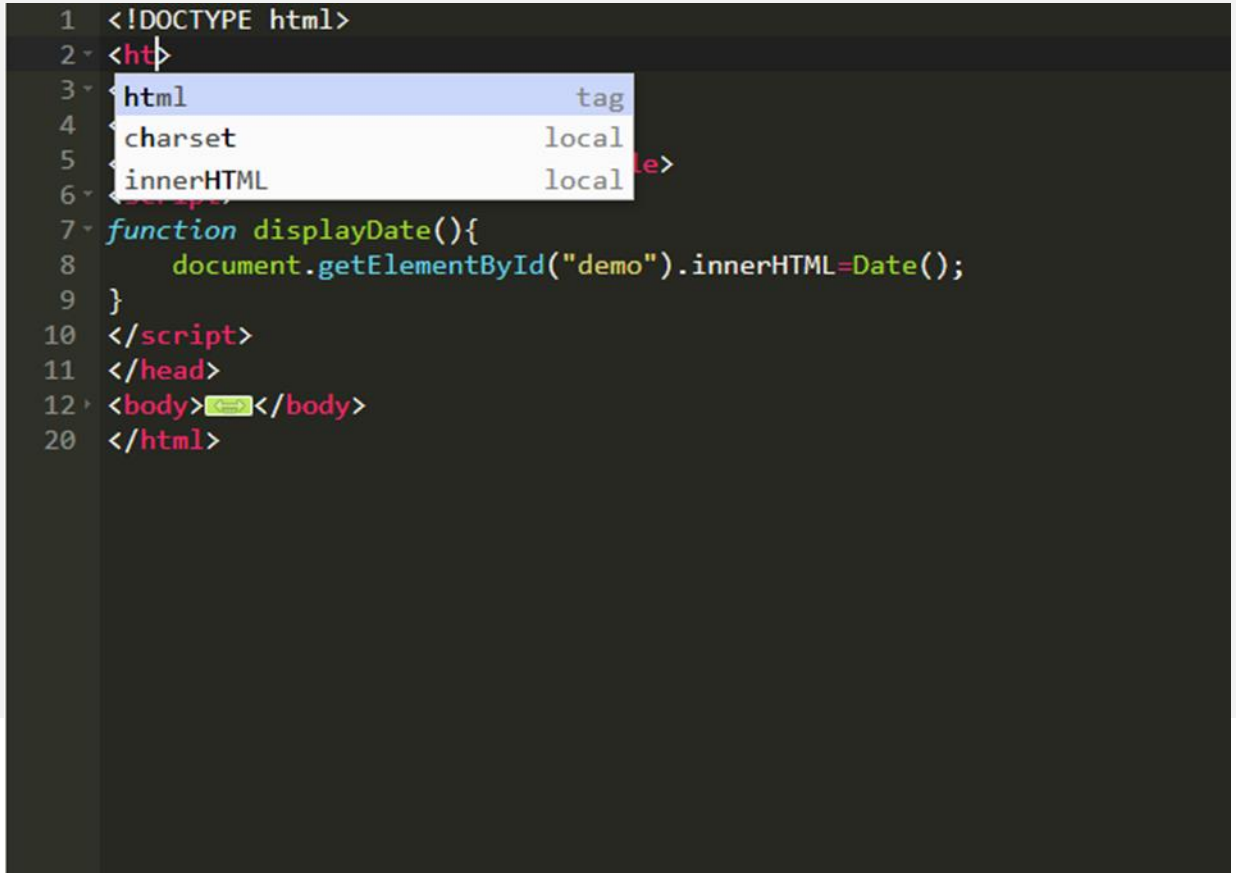
```
def main(code):
    r = dict()
    jname = get_javaname(code)
    fpath = write_file(jname, code)
    try:
        # subprocess.check_output 是 父进程等待子进程完成，返回子进程向标准输出的输出结果
        # stderr是标准输出的类型
        # 编译Java代码
        outdata = decode(subprocess.check_output([JAVAC_EXEC, fpath], stderr=subproces
    except subprocess.CalledProcessError as e:
        # e.output是错误信息标准输出
        # 错误返回的数据
        r["code"] = 'Error'
        r["output"] = decode(e.output)
        return r
    else:
        # 执行Java代码
        # 因为调用原因，bat写绝对路径
        outdata = decode(
            subprocess.check_output(["C:\\Users\\nianyu\\PycharmProjects\\zhiliao\\te
                                     stderr=subprocess.STDOUT))
        # 成功返回的数据
        r['output'] = outdata
        r["code"] = "Success"
        return r
```

java编译主要代码

PART3完成情况

在线编译模块

Html/Javascript在线编译部分依旧采用Ace编辑器代替普通网页编辑框，显示部分采用<iframe>标签将网页隔离出一部分区域用来显示代码执行效果，在<iframe>获取Ace编辑器的代码得时发现，使用document.getElementById().innerText方法会将编辑器的行号也一并获取，导致错误，查找资料和官方指导后使用Ace编辑器自带的api解决了问题。



PART3完成情况

```
<pre id="code" class="ace_editor" style="min-height:400px">
 <textarea  class="ace_text-input"></textarea>
</pre>


<script>
    //初始化对象
    editor = ace.edit("code");
    editor.setOptions({
        enableBasicAutocompletion: true,
        enableSnippets: true,
        enableLiveAutocompletion: true,//只能补全
    });
    //设置风格和语言（更多风格和语言，请到github上相
    {#theme = "clouds"#}
    {#language = "python"#}
    editor.setTheme("ace/theme/monokai");
    editor.session.setMode("ace/mode/python");

    //字体大小
    editor.setFontSize(18);
```

Ace编辑器主要代码

PART3 完成情况

词云

词云模块需要实现用用户上传的pdf文件生成相应的词云图的功能，查找资料发现Pdfminer库可以从Pdf文件中提取信息故采用先将用户上传的Pdf文件转换为txt文件，再用Python的Jieba库对数据进行中文分词再用Wordcloud库生成词云图的方法。

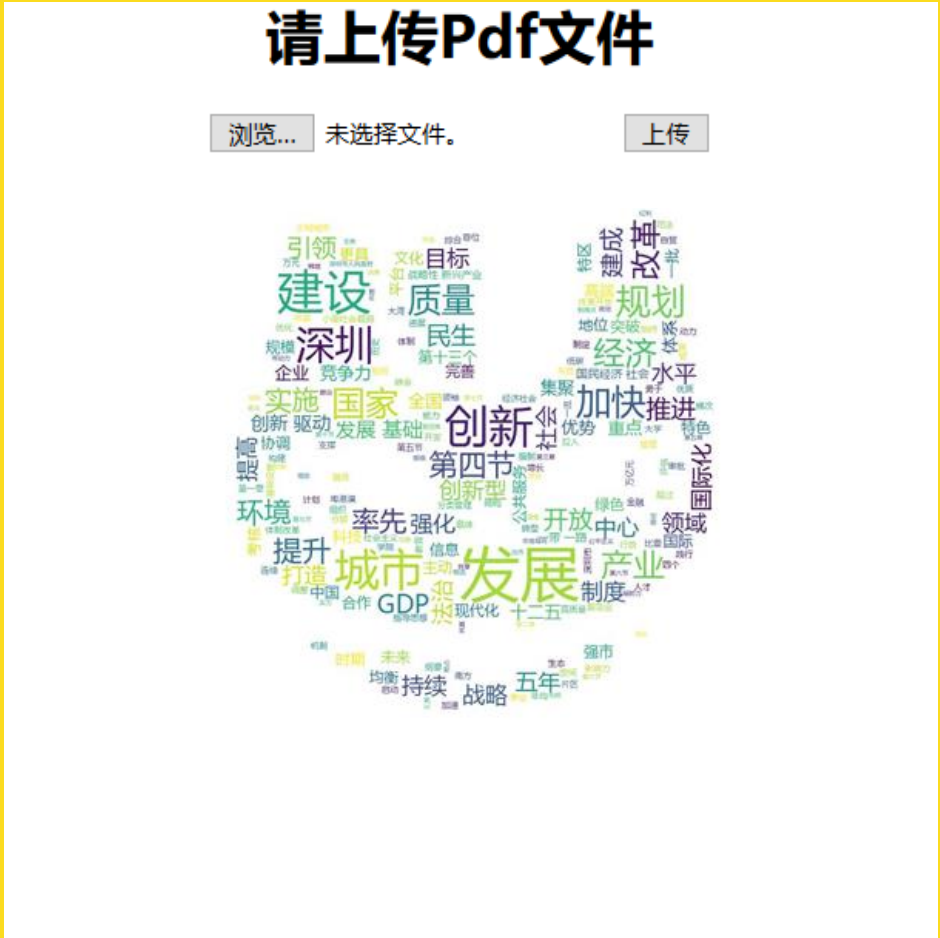


图2：深圳市国民经济和社会发展第十三个五年规划纲要.pdf词云效果图

Wordcloud可通过替换mask背景图来生成不同形状的词云，Wordcloud默认字体不支持中文需要添加期望使用的中文字体路径，此外Wordcloud（按英文习惯）以空格分词，中文不用空格所以WordCloud不能正确对中文进行分词，所以引入Jieba库来进行对中文的分词，Jieba库在进行中文分词可以通过添加停用词来停用一些和内容无关的常用词，例如“今天” “明天” “你好” “第一节”。

PART3完成情况

```
    def split_word(self):
        seq = ''
        sw_words = self.get_stopword()
        text = self.text_wash()
        segs = jieba.cut(text,cut_all=False)
        for seg in segs:
            if seg not in sw_words:
                seq = seq + seg +" "
        return seq
```

```
class wordclouder():
    # get parameter
    def __init__(self,text,image):
        self.text = text
        self.imag = image

    # 生成词云图
    def word_cloud(self):
        mask_image = imread(self.imag,flatten=False)
        word_pic = WordCloud(
            font_path='./utils/msyh.ttc',
            background_color='white',
            mask=mask_image
        ).generate(self.text)
        imsave(wc,word_pic)
```

```
class word_spliter():
    def __init__(self,text,stop_path = sw_path):
        self.text = text
        self.stop_word = stop_path

    def get_stopword(self):
        stopwords = {}.fromkeys([line.rstrip() for line in open(self.stop_word, encoding='utf-8')])
        return stopwords

    def text_wash(self):
        self.text = self.text.encode(encoding="utf-8",errors='ignore').decode("utf-8")
        # print(self.text)
        return self.text
```

```
def generate_wordcloud(text,pic):
    sp_word = word_spliter(text)
    words = sp_word.split_word() #分词
    g_word = wordclouder(words,pic)
    g_word.word_cloud()
```

wordcloud主要代码

PART 4

存在问题和方法

PART4存在问题和方法

存在问题

对Python以及Flask框架的运用不够熟悉

Python/Java的编译结果返回页面显示时会刷新原有界面，显示结果，这样就会清空用户已经输入到编辑框的代码，不方便用户修改代码。

拟采取的办法

查阅书籍、上网查询学习有关Python以及Flask的相关知识，了解如何使页面不刷新，或者研究如何使输入框的代码在页面刷新之后依旧保留来解决问题。

PART 5

下步工作

PART5下步工作

01

1、研究如何使用Python实现Pdf文件转化为Html文件。

02

2、研究如何使用Python实现Ppt转Pdf文件。

03

3、研究如何组合PPt转Pdf和Pdf转html，进而实现Ppt转Html。

04

4、组合全部模块并上传到服务器运行。

5、汇总结果并完成最终论文，答辩。


THANK YOU
FOR WATCHING