《Web 应用开发》课程大作业报告评阅表

项目	报告写作	符合课程	功能合理	具有特	按时提交	总分
	和格式规	要求	且完善,	色或创	材料,材	(100%)
	范,无抄	(20%)	代码无抄	新性	料无遗漏	签名:
	袭情况		袭情况	(10%)	(10%)	
	(30%)		(30%)			
得分						
得分说						
明/评						
语						

2023-2024 第 2 学期

《Web 应用开发》

课程大作业报告

大作业题目:_	基于 B	Beego 框架实现	见简历解析系统	(命题/自选	<u>;)</u>
所在教学班:	XX 班	姓名: _ 张	<u>:三</u> 学号:	123	

一、功能概述

1. 简历上传与解析:

用户可以通过系统上传简历文件,系统能够自动解析其中的文本内容。 使用了第三方库 `github.com/ledongthuc/pdf` 来解析 PDF 文件,提取其中的文本信息。

2. 关键字高亮处理:

实现了 `HighlightKeywords` 函数,用于在简历文本中高亮显示关键字,如姓名、电话、教育背景等。

通过正则表达式匹配和字符串替换实现关键字高亮。

3. HTML 内容保存:

提供了 `SaveContentAsHTML` 函数,用于将解析后的 HTML 内容保存为 HTML 文件。使用了 `html/template` 包来处理 HTML 内容,确保内容的安全性。

4.PDF 文件验证:

实现了 `ValidatePDF` 函数,用于验证上传的文件是否为有效的 PDF 格式文件。通过读取文件头部的字节数据,判断文件是否以 `%PDF-` 开头,来进行简单的 PDF 文

件验证。

Web 页面处理:

使用了 Beego 框架来处理 Web 页面的路由和请求。

包含了一个 `PDFController` 控制器,用于处理 PDF 文件的上传和解析。

临时文件处理:

在文件上传和解析过程中,使用了临时文件来存储上传的 PDF 文件内容,避免了直接操作原始文件带来的风险。

安全性考虑:

对文件上传、解析过程中可能出现的错误进行了处理和错误提示,提高了系统的健壮性和用户体验。

使用了 `html/template` 包来处理 HTML 内容, 防止 XSS 攻击和其他安全问题。

总的来说,这个系统通过简单而实用的功能实现,为用户提供了方便快捷的简历上传与解析服务,并考虑了安全性和错误处理等方面,为用户提供了更加稳定和安全的使用体验。

二、实现原理和方法

这是一个基于 beego 框架的简单 Web 应用,用于上传和解析 PDF 文件中的简历内容。以下是应用的功能概述:

导入依赖项:代码导入了一些标准库和第三方库,包括 Beego(用于 Web 开发框架)和 Ledongthuc 的 PDF 库(用于处理 PDF 文件)。 初始化函数:代码包含了一个 init() 函数,其中注册了一个名为 safeHTML 的函数到 Beego 模板引擎中,以便在模板中安全地呈现 HTML 内容。

数据结构:

定义了一个名为 Resume 的结构体,用于存储简历内容。SaveContentAsHTML:将 HTML 内容保存为 HTML 文件。

HighlightKeywords: 用于在简历文本中高亮显示关键字。

ParsePDF: 用于解析 PDF 文件,提取文本内容,并调用外部 API 进一步处理关键信息。

PDFController: 控制 PDF 文件的上传和解析过程。

Get 方法用于显示上传页面。

Post 方法用于处理上传的 PDF 文件,进行解析,并将解析后的内容传递给模板进行渲染。

ValidatePDF: 用于验证文件是否为有效的 PDF 格式,检查文件开头是否包含 %PDF-。

整体来说,这个应用允许用户上传 PDF 格式的简历文件,然后解析其中的文本内容,最终将解析后的内容渲染到一个结果页面中供用户查看。

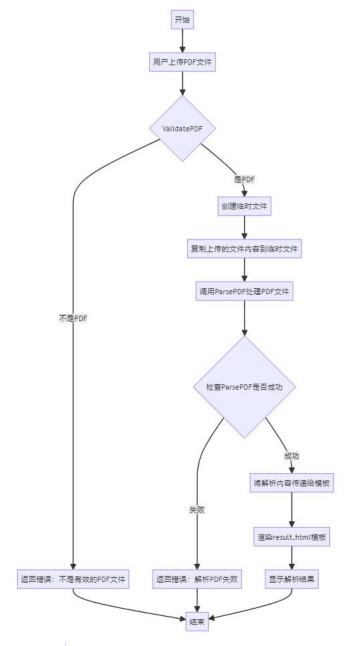


图 1 beego 简历解析系统流程图

三、实验和测试环境

Vscode Go 1. 22. 2

```
■ upload.html
                                                            3 sty
eso pdf.go 2 X eso api.go eso main.go
tempgo > controllers > € pdf.go > ♦ ValidatePDF
       func ValidatePDF(file *os.File) error {
              return err
 170
          // 读取文件的前5个字节,检查是否以"%PDF-"开头
 171
          buf := make([]byte, 5)
          _, err = file.Read(buf)
 172
          if err != nil {
 174
              return err
175
 176
          if string(buf) != "%PDF-" {
              return fmt、Errorf("文件不是以 %PDE- 开头")
178
 179
          return nil
```

图 2 生产环境以及配置图

四、运行和测试结果

描述一下结果, 附带效果截图 简历上传功能

在线简历解析



图 3 简历上传功能运行效果

1. 姓名: (未给出具体姓名)

2. 性别: 男/女

3. 出生年月: 19xx.xx.xx

4. 民族: xx

5. 政治面貌: 无提供

6. 教育背景:

- 学校: xxxxXX大学 - **专业**: 中医药**专业**

- 学历: 20XX届 XX方向 XX学士 - 成绩: 学分点/GPA: x.x/满分x分

- 奖项: 秀班干部、新之星、学生干部、三好学生等

7. 项目/科研经历:

- 20xx年: xx项目(负责人), xxxxxx课题 - 20xx年: xxxxxxx项目(项目组成员)

8. 工作/实习经历:

- 20xx.x-20xx.x: xx中心医药药剂科(实习),职责包括药品采购、管理工作等

图 4 简历解析功能运行效果

```
2024/05/31 21:52:29.758 [D] [server.go:3137]
                                                           ::1 304
                                                                                      match GET
                                                                           770.6µs
PS C:\Users\Golan\Desktop\cvapp3\tempgo> bee run
                  | v1.12.0
2024/05/31 21:52:41 WARN
                            ▶ 0001 Running application outside of GOPATH
2024/05/31 21:52:41 INFO
                            ▶ 0002 Using 'tempgo' as 'appname'
                            ▶ 0003 Initializing watcher...
2024/05/31 21:52:41 INFO
# cd C:\Users\Golan; git -c log.showsignature=false show -s --format=%H:%ct
fatal: your current branch 'main' does not have any commits yet
2024/05/31 21:52:43 SUCCESS ▶ 0004 Built Successfully!
2024/05/31 21:52:43 INFO
                             ▶ 0005 Restarting 'tempgo.exe'...
2024/05/31 21:52:43 SUCCESS ▶ 0006 './tempgo.exe' is running...
2024/05/31 21:52:43.064 [I] [asm_amd64.s:1695] http server Running on http://:8080
```

图 5 简历解析功能运行效果

五、结论和总结

这是一个基于 Beego 框架的简历上传与解析系统。通过该系统,用户可以上传简历文件,系统会自动解析其中的内容,并提取关键信息,如姓名、联系方式、工作经历等。该系统具备了简历上传、解析、关键信息提取等功能,为用户提供了便捷的简历管理和分析服务。通过调用外部 API 或自定义函数,实现了对简历文本的关键字高亮处理和进一步的信息提取,增强了系统的功能和实用性。在文件上传和解析过程中,代码实现了文件类型验证,确保上传的文件为有效的 PDF 格式文件,提高了系统的安全性。使用了临时文件存储上传的 PDF 文件内容,减少了因文件操作而引起的潜在安全风险。通过 Beego 框架提供的控制器结构,实现了简单而灵活的路由管理和请求处理,便于系统的功能扩展和维护。在代码中对可能出现的错误进行了适当的处理和错误提示,提高了系统的稳定性和用户友好性。对文件的合法性进行了严格的验证,避免了无效文件对系统造成的潜在问题。使用了临时文件存储上传的 PDF 文件内容,并通过合适的仓fer 语句及时释放资源,提高了系统的性能和资源利用率。通过合理的文件读取和字符串处理方式,有效地减少了系统的内存占用和运行时的性能开销。

附: 关键代码(不超过100行)

```
// ParsePDF 函数用于从 PDF 文件中提取和处理文本内容 func ParsePDF(filePath string) (Resume, error) {
    // 打开文件
    file, err := os.Open(filePath)
    if err != nil {
        return Resume{}, err
    }
    defer file.Close()

// 获取文件信息
```

```
fileInfo, err := file.Stat()
    if err != nil {
        return Resume{}, err
   }
   // 创建 PDF 阅读器
    reader, err := pdf.NewReader(file, fileInfo.Size())
    if err != nil {
        return Resume{}, err
   }
   // 用于存储 PDF 文本内容的字符串构建器
    var text strings.Builder
    numPages := reader.NumPage()
    for pageNum := 1; pageNum <= numPages; pageNum++ {
        page := reader.Page(pageNum)
        // 获取页面的纯文本内容
        content, err := page.GetPlainText(nil)
        if err != nil {
            return Resume{}, err
        text.WriteString(content)
        if pageNum < numPages {</pre>
            text.WriteString("\n--- 分页符 ---\n")
        }
   }
   // 对文本内容进行关键字高亮处理
    content := HighlightKeywords(text.String())
    log.Print(content)
    // 调用 ParseText 函数处理文本
    apiKey := "sk-705d39237a4a4553900ead7c4bfde6bb"
    prompt := "帮我提取出下面这个简历的关键信息:" + content
    content, err = ParseText(apiKey, prompt)
    content = HighlightKeywords(content)
   // 返回解析后的简历内容
    return Resume{Content: content}, nil // Content 保持为 string 类型
// PDFController 用于处理 PDF 文件的上传和解析
type PDFController struct {
    beego.Controller
```

}

```
}
// Get 方法用于显示上传页面
func (c *PDFController) Get() {
    c.TplName = "upload.html"
}
// Post 方法用于处理 PDF 文件的上传和解析
func (c *PDFController) Post() {
    file, _, err := c.GetFile("file")
    if err != nil {
        c.Ctx.WriteString("获取上传文件失败: " + err.Error())
        return
    }
    defer file.Close()
    // 创建临时文件用于存储上传的 PDF 文件内容
    tempFile, err := os.CreateTemp("", "*.pdf")
    if err != nil {
        c.Ctx.WriteString("创建临时文件失败: " + err.Error())
        return
    }
    defer tempFile.Close()
    defer os.Remove(tempFile.Name())
    // 将上传的文件内容复制到临时文件中
    _, err = io.Copy(tempFile, file)
    if err != nil {
        c.Ctx.WriteString("保存上传文件失败: " + err.Error())
        return
    }
    // 验证文件是否为有效的 PDF 格式,通过检查文件开头是否包含"%PDF-"
    if err := ValidatePDF(tempFile); err != nil {
        c.Ctx.WriteString("上传文件不是有效的 PDF: " + err.Error())
        return
    }
    // 解析 PDF 文件内容
    resume, err := ParsePDF(tempFile.Name())
    if err != nil {
        c.Ctx.WriteString("解析 PDF 失败: " + err.Error())
        return
    }
```

```
// 将解析后的内容传递给模板
    c.Data["ContentForTemplate"] = resume.Content
   // 渲染模板
    c.TplName = "result.html"
    c.Layout = "layout.html"
   c.Render()
}
// ValidatePDF 函数用于检查文件是否为有效的 PDF 格式
func ValidatePDF(file *os.File) error {
   // 将文件指针移到文件开头
    _, err := file.Seek(0, io.SeekStart)
   if err != nil {
        return err
   }
   // 读取文件的前 5 个字节, 检查是否以"%PDF-"开头
    buf := make([]byte, 5)
    _, err = file.Read(buf)
   if err != nil {
        return err
   }
   if string(buf) != "%PDF-" {
        return fmt.Errorf("文件不是以 %PDF- 开头")
   }
    return nil
}
package main
//Main.go 注册执行
import (
   _ "tempgo/routers"
    "github.com/astaxie/beego"
)
func main() {
    beego.Run()
}
```

请务必提交以下电子版文件(缺一不可):

- 1.word 版本报告 (.doc 或.docx)。除了电子版外,还需要打印并提交纸质版本。
- 2.将 word 版本报告转为 PDF 格式 (.pdf) ,请确保与 word 版本内容一致。
- 3.全部源代码打包到一个文件中(.zip)。
- 4.关键代码 (100 行左右)。加注释!
- 5.运行效果截图(png 或 jpg 格式),选择 1 张你认为效果最好的提交。更多的效果截图可以放在 word 版本报告中。

文件命名要求:

以上所有文件请以学号开头进行命名:

例如:

32019070299-张三.doc

32019070299-张三.pdf

32019070299-张三.zip

32019070299-张三.go

32019070299-张三.png

以上文件请分别以附件方式上传。

注意:

- 1.源代码中请使用相对路径,不要使用绝对路径(绝对路径会导致老师这里无法运行)。
- 2.请务必包含全部需要的文件,资源文件等(但不要包含没有用的多余文件),尽量不要使用 CDN(有可能也会无法运行)。
- 3.如果你的代码中包含或依赖可能在老师这里无法运行的环境(例如有数据库等),请在 word 版报告中的"实验和测试环境"部分中说明清楚。
- 4.请在 word 版本报告中包含你的运行效果截图(以免老师这里无法运行导致看不到你的效果),截图请涵盖你实现的所有功能,截图可以是整个页面的布局图,也应当包含局部细节图(用来表明你实现的具体功能,局部图能更好地展现细节,同时能让老师重点关注到你想表现的内容,而不是让老师在一幅图中去猜你到底想展现什么内容)。
- 5.word 版报告(及 PDF)务必阐述清晰(你要把老师看作是完全不了解你做了什么的人,然后想办法把你做的内容说清楚,说明白,阐述请用书面语,不要过于口语化),排版规范(字体,字号,图下方要有图 1.xxx,图 2.xxx 这样的题注)。题注是"指出现在图片下方的一段简短描述",表格的题注要放在表格的上方。题注也会帮助老师理解你的图和表想表达的内容。

6.word 版报告没有必要粘贴源代码(如有,请加注释)。

7.请提交最终版本程序(删除代码中没有用的内容,删除不需要和没用的多余文件)。