《Web 应用开发》课程大作业报告评阅表

项目	报告写作和格式规范，无抄袭情况（30%）	符合课程要求（20%）	功能合理且完善，代码无抄袭情况（30%）	具有特色或创新性（10%）	按时提交材料，材料无遗漏（10%）	总分（100%） 签名：
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：GO 语言简历解析系 （命题/自选）

所在教学班：XX 班 姓名：张三 学号：123

一、功能概述

Go 语言编写的简历解析系统是一个 web 应用程序，它能够处理用户上传的简历文件（PDF 格式），提取关键信息，并将其转换为结构化数据。这个系统可以为人力资源部门自动化简历筛选过程，提高效率。以下是该系统的一个概述：

1. 系统功能：

文件上传：允许用户上传简历文件。

PDF 解析：使用 PDF 解析库来读取和解析 PDF 文件中的内容。

数据结构化：将解析出的文本转换为结构化的格式，例如，将“教育背景”、“工作经验”等信息分类。

结果展示：将解析结果展示给用户，可能包括原始文本、高亮关键词和结构化数据。

结果保存：可以将解析后的 html 保存到网页。

2. 系统架构：

Web 服务器：使用 Go 的`net/http`包来搭建一个简单的 HTTP 服务器。

路由处理：定义路由来处理文件上传和结果页面的请求。

模板渲染：使用`html/template`包来渲染上传页面和结果页面。

文件 I/O：处理文件的读取、写入和临时存储。

PDF 处理：集成第三方库，如`unidoc`，来处理 PDF 文件的解析。

3. 技术栈：

编程语言：Go（Golang）。

Web 框架：标准库`net/http`（可能使用第三方框架如 Gin）。

模板引擎：Go 的`html/template`。

PDF 解析：unidoc。

前端：HTML/CSS/JavaScript。

4. 用户流程：

1. 用户访问简历解析系统的网页界面。
2. 用户上传简历文件（通常是 PDF 格式）。
3. 系统后端接收文件，并使用 PDF 解析器处理文件。
4. 系统提取并高亮简历中的关键词。
5. 系统将解析结果以结构化的方式展示给用户。
6. 用户可以查看结果，并通过下载链接保存解析后的内容。

二、实现原理和方法

这段 Go 语言编写的代码实现了一个简单的 HTTP 服务器，它提供了简历解析的功能。以下是代码实现原理和方法的详细说明：

1. 包的导入

代码开始处导入了一系列 Go 的标准库和第三方库，这些库提供了文件操作、正则表达式处理、HTTP 服务器搭建、模板渲染等功能。

2. 结构体定义

`Resume`结构体用于存储简历的 HTML 内容，其中`Content`字段使用`template.HTML`类型，以便在模板中安全地渲染 HTML。

3. 保存 HTML 内容函数

`saveContentAsHTML`函数用于将 HTML 内容保存到文件中。它创建一个文件，并将传入的 HTML 内容写入该文件。

4. PDF 解析函数

`parsePDF`函数负责打开和读取 PDF 文件，提取其中的文本，并逐页进行处理。它使用`unidoc`库来读取 PDF 文件的页面内容。解析出的文本通过`highlightKeywords`函数进行关键词高亮处理。最终，解析和处理后的简历内容被保存为 HTML 文件，并返回一个包含 HTML 内容的`Resume`结构体。

5. 文件上传处理函数

`uploadHandler`函数处理 HTTP 请求，它根据请求方法（GET 或 POST）执行不同的操作：

- 对于 GET 请求，它加载并显示一个上传简历的 HTML 表单。
- 对于 POST 请求，它处理文件上传，将上传的文件存储到服务器上的临时文件中，然后调用`parsePDF`函数来解析 PDF 文件。解析完成后，它加载并渲染一个展示结果的 HTML 模板，并将解析后的简历内容传递给模板进行显示。

6. 静态文件服务

在`main`函数中，服务器配置了静态文件服务，用于提供 CSS、JavaScript 等静态资源。

7. HTTP 服务器启动

`main`函数最后启动 HTTP 服务器，监听 8080 端口，并处理传入的请求。服务器使用`http.HandleFunc`注册路由和相应的处理函数，`http.FileServer`和`http.StripPrefix`用于服务静态文件。

8. 实现方法

模板渲染：使用`html/template`包来渲染 HTML 模板。

文件操作：使用`os`和`io`包进行文件的读写操作。

PDF 解析：使用`unidoc`库来解析 PDF 文件。

HTTP 服务器：使用`net/http`包来创建和运行 HTTP 服务器。

错误处理：通过检查函数返回的错误来实现基本的错误处理。

9. 简历解析

通过使用千问通义大模型来对系统解析并使用正则表达式对结果进行优化。

通过以上步骤实现了结合 Go 语言的标准库和第三方库来创建一个提供简历解析的 Web 服务。

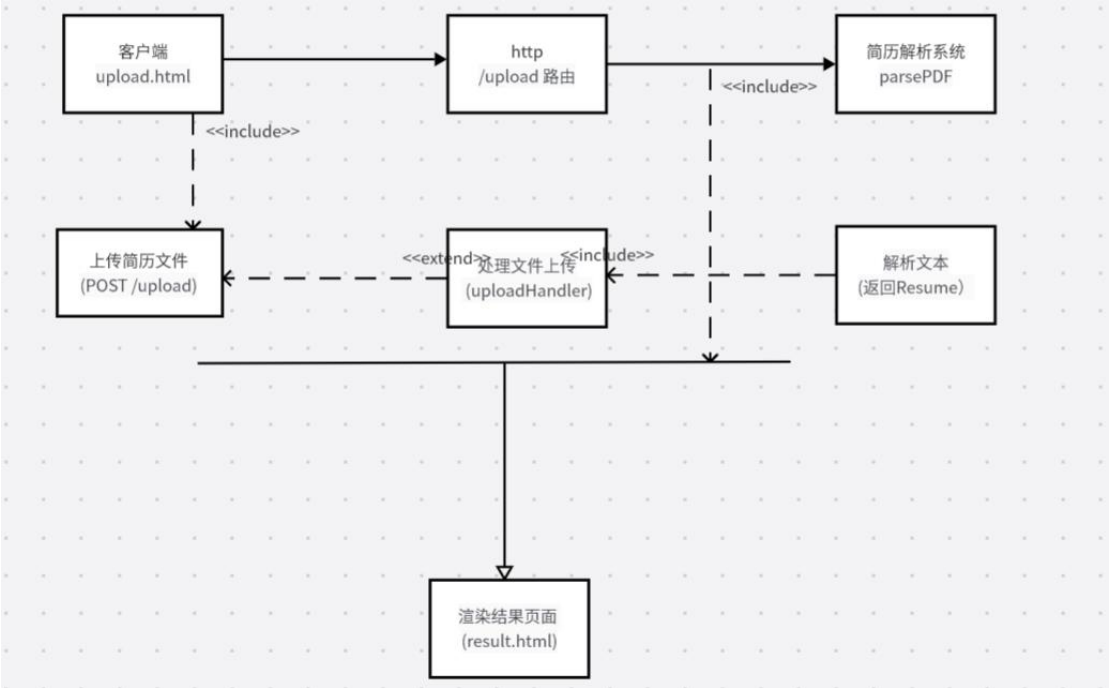

图 1 简历解析系统 UML 图

三、实验和测试环境

Vscode

Go 1.22.2

```
module cvapp

go 1.22.2

Check for upgrades | Upgrade transitive dependencies | Upgrade direct dependencies
require github.com/unidoc/unidoc v2.2.0+incompatible

require golang.org/x/image v0.15.0 // indirect

// require gpt/tongyi/        v0.0.0
```

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

图 3 简历上传功能运行效果

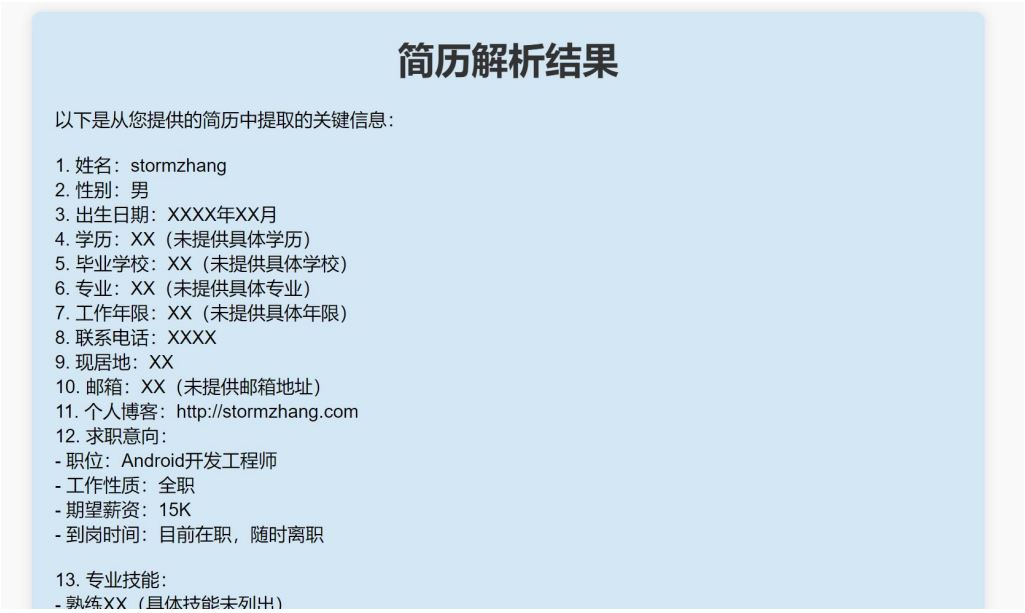

图 4 简历解析功能运行效果

五、结论和总结

1. 功能实现：系统成功实现了简历文件的上传、解析、结果保存和展示的功能.

2. 技术选型：系统采用了 Go 语言进行后端开发，利用了`unidoc`库进行 PDF 文件的解析，`html/template`进行模板渲染，以及标准库中的`net/http`来搭建 HTTP 服务器。

3. 用户体验：用户可以通过简洁的 Web 界面上传简历，并得到一个格式化的解析结果，提高了简历信息的可读性和易用性。

4. 扩展性：系统设计允许未来添加更多功能，例如更复杂的文本分析、机器学习集成以提高关键词识别的准确性等。

5. 性能：系统性能在很大程度上取决于 PDF 解析库和服务器硬件。对于大量用户或大型文件，可能需要进一步的性能优化。

自动化的简历解析过程节省了人力资源部门的时间和精力。结构化数据：将非结构化的 PDF 文本转换为结构化数据，便于进一步处理和分析。用户友好：提供了一个简单的 Web 界面供用户上传和查看简历。

这个系统提供了一个基本但有效的简历解析服务，可以作为进一步开发的起点。在实际部署前，需要进行彻底的测试和安全审查，以确保系统的稳定性和安全性。

附：关键代码（不超过 100 行）

```
// uploadHandler 处理文件上传和结果显示
func uploadHandler(w http.ResponseWriter, r *http.Request) {
    // 如果是 GET 请求，加载并显示上传表单
    if r.Method == http.MethodGet {
        tmpl, err := template.ParseFiles("upload.html") // 尝试解析上传表单的 HTML 模板
        if err != nil {
            http.Error(w, "模板加载失败", http.StatusInternalServerError) // 模板加载出错时返回 500 状态码
            return
        }
        tmpl.Execute(w, nil) // 渲染并发送表单到客户端
        return
    }

    // 如果是 POST 请求，处理文件上传
    if r.Method == http.MethodPost {
        file, _, err := r.FormFile("file") // 从表单中获取名为"file"的文件
        if err != nil {
            http.Error(w, "文件上传错误", http.StatusBadRequest) // 文件读取错误时返回 400 状态码
            return
        }
        defer file.Close() // 确保文件句柄在操作完成后关闭
```

```
        // 创建一个临时文件来存储上传的 PDF
        tempFile, err := os.CreateTemp("", "upload-*.pdf")
        if err != nil {
            http.Error(w, "创建临时文件失败", http.StatusInternalServerError) // 创建临时文件失败时返回 500 状态码
            return
        }
        defer os.Remove(tempFile.Name()) // 上传处理后删除临时文件
```

```
        // 将上传的文件内容复制到临时文件
        _, err = io.Copy(tempFile, file)
        if err != nil {
            http.Error(w, "保存文件失败", http.StatusInternalServerError) // 文件保存出错时返回 500 状态码
            return
        }
```

```
        // 解析临时 PDF 文件内容（此函数需自定义实现）
        resume, err := parsePDF(tempFile.Name())
        if err != nil {
```

```
            http.Error(w, "解析 PDF 失败", http.StatusInternalServerError) // PDF 解析出错时返回 500 状态码
            return
        }
```

```
        // 加载结果显示的 HTML 模板
        tmpl, err := template.ParseFiles("result.html")
        if err != nil {
            http.Error(w, "结果页面模板加载失败", http.StatusInternalServerError) // 模板加载失败时返回 500 状态码
            return
        }
        // 渲染结果模板并发送给客户端
        err = tmpl.Execute(w, resume)
        if err != nil {
            http.Error(w, "渲染结果页面失败", http.StatusInternalServerError) // 渲染结果页面出错时返回 500 状态码
        }
    }
}
```

```
// main 函数启动 HTTP 服务器
func main() {
    // 注册上传处理的路由
    http.HandleFunc("/upload", uploadHandler)
```

```
    // 配置静态文件服务，用于提供 CSS、JS 等静态资源
    fs := http.FileServer(http.Dir("./static"))
    http.Handle("/static/", http.StripPrefix("/static/", fs))
```

```
    // 启动 HTTP 服务器并监听 8080 端口
    log.Println("服务器正在监听:8080")
    err := http.ListenAndServe(":8080", nil)
    if err != nil {
        log.Fatalf("服务器启动失败: %v", err) // 启动失败时打印错误信息并退出程序
    }
}
```

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）。