**《程序设计进阶与实践》大作业报告**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 姓名（组长） | | 闫泽轩 | 学号 | PB21111696 |
| 姓名（组员1） | | 李牧龙 | 学号 | PB21111639 |
| 姓名（组员2） | | 陈治非 | 学号 | PB21000096 |
| 姓名（组员3） | | 左桑贝 | 学号 | PB21000047 |
| 姓名（组员4） | | 徐航宇 | 学号 | PB21050996 |
| 项目名称 | | USTCpermanent（网页快照保存，网址http://124.221.239.13:8000/） | | |
| **一、项目需求分析（可选）**  **随着现代网络迅速的更新迭代，许多网站被篡改，升级或者删除，导致网站的时效性无法得到保证。固定某个网站的数据，保存证据防止丢失成为一种日益广泛的需求。**  **二、项目功能设计**  **1.总体功能说明**  **实现用爬虫或Javascript爬取网页内容并保存至服务器上，防止原网页内容的丢失或变更。项目采用前端后端互相分离的设计，前端由网页和插件组成，后端由python爬虫和django框架组成。项目还实现了用户的注册，登陆，修改密码，退出登录的管理操作。网址在**[**http://124.221.239.13:8000/**](http://124.221.239.13:8000/)  **域名<http://www.ustcpermanent.cn>(网站正在备案中，域名有可能不可用，可使用ip地址访问)**  **2.具体功能点说明**  **前端：**   1. **网页**   **主要负责各个页面UI的美术风格，保持统一，高效的设计语言，同时加以美感的辅助。**   1. **JS爬虫实现**   **通过扫描插件发送的目标网页的HTML源代码，爬取网页中的图片、JS文件、CSS文件等资源，并以压缩包的形式返回给插件去解包、配置文件路径。**  **3. 插件：**  **运行在chrome内核的浏览器中的插件，可以为用户的使用提供一定的便利，同时，也可以提供一些适用性更广的功能（多用于需要登录的网站等）。**  **使用：可通过工具栏或快捷键弹出操作界面，或使用右键菜单快速使用，具体使用教程在官网上有详细介绍。**  **功能：**  **Normalshot：自动抓取当前页面的url,发送到后端抓取页面信息及截屏。适用于不需要登录的网站等。**  **Screenshot：可以获取用户屏幕的截屏，同时发送url。适用于需要登录的网站等。需要注意的是：在截屏过程中，屏幕会有较迅速的滚动，需要耐心等待直到提示截屏结束，否则可能会造成截屏失败。**  **Pageshot：可以在前端抓取网页信息及截屏。适用于需要登录的网站等。目前还在开发中，不具有实际功能。**  **后端：**   1. **django**    1. **总管项目的结构逻辑，结构如下：USTCpermanent为网站总路径，下设有homepage,user\_backend,upload为应用，templates文件夹下保存各种页面，static下保存页面的js,css,img等静态资源，media下保存爬虫爬取的mhtm文件。**    2. **管理数据库的关系映射，所涉表及字段如下：User为用户表，有用户名，密码（已SHA256加密处理）,邮箱字段。Caches为快照表，有用户关联字段，更新时间，快照地址，描述信息。info\_bind为验证表，有用户关联字段，验证码字段。**    3. **负责网站数据保存。使用腾讯云对象存储技术，建立存储桶，地址'ustcpermanent-1308044039'，可以上传用户文件并提供访问地址和下载功能。**    4. **部署和上线服务器。使用腾讯ubuntu服务器，安装宝塔管理系统，在python项目管理器中部署django网站，并配置静态文件地址，出入端口口管理，FTP文件服务系统，域名映射地址和项目运行环境的配置，并设置了一个管理员后台。** 2. **爬虫**   **用户上传希望保存快照的网页，后端将该网页保存为mhtml文件，同时保存一张网页的长截屏。**  **3.功能点设计细节**  **前端：**   1. **网页**   **使用现代化的html，css及JavaScript三者结合进行网页的编写，分散式的编写方式使得根据需求调整代码更加方便快捷。**  **同时，设计语言的统一也使得代码减少了重复工作量。**   1. **JS爬虫**   **借助cheerio（类jQuery）模块，对插件发送过来的HTML字符串load出相应的$变量，并进一步提取img、script等标签中的资源URL，在set对象中通过简单的去重和域名补全工作后，再借助http/https模块发送网络请求，将得到的文件资源封装进事先创建好的JSzip对象中（此过程中可以设置二级文件目录），待所有文件下载完毕后输出zip压缩包给插件，完成工作。**  **3. 插件：**   1. **插件程序通过manifest.json，按照插件的开发原则进行统筹。其中设置了插件的主要信息，为插件运行提供了必要的权限许可（比如对一些chrome api的使用权限），以及对弹出界面（popup），背景页（background）等进行了规划。** 2. **弹出界面由popup.html来编写及统筹，其中的css文件由陈治非同学完成。弹出界面中设置了各中对用户开放的接口。** 3. **右键菜单由menu.js直接插入背景页中实现：在menu.js中，通过调用chrome.contextMenus.create()方法添加按钮、选项卡与它们的点击效果。点击选项卡会切换抓取模式，点击按钮则会根据所选的抓取模式进行对应的抓取。** 4. **登录：编写了函数login()，用以获取用户登录后留在本地有关用户信息的cookie值，如果未登录或登录过期，则会弹出官网要求登录。当插件开始运行时或打开popup界面后均会调用，以实现自动登录。** 5. **抓取页面：popup中通过chrome.tabs.query()获取当前页面的tab标签，menu.js中直接使用contextMenus获取到的tab标签。Normalshot中直接获取url等信息，连同其他信息一同上传。Screenshot较为复杂：在如上操作的同时，通过chrome.tabs.executeScript()向页面中插入内容脚本，并通过chrome.runtime下的相关方法实现页面间的通讯。通过反复的通讯传递各项指令，操纵用户界面滚动，并逐一使用chrome.tabs.captureVisibleTab()截屏，并最终合并，将图片的url上传后端下载。Pageshot的主要功能由左桑贝同学设计，难度较大，实现效果不太好，且由于时间及本人状态不佳，无法对接，故没有投入使用。** 6. **上传：将获取的信息通过$.post()方法（jQuery）以json格式传输到后台。具体的：各种抓取模式由字段uploadtype区分。除了抓取到的页面信息外，获取了页面中的sessionid与后端比对，防止恶意修改。description通过prompt（）弹出输入框并由用户设置。同时设置了回调函数以获取后端发来的信息并给予用户提示，此外，还设置了有关超时的提示。**   **后端：**   1. **django**    1. **django作为重型网站框架，其提供了大而全的网站服务解决方案。在访问某一项服务时，django会通过路由配置按一定的规则将访问的请求转发到各个应用的函数里去。为此配置合理的项目结构，划分应用范围十分重要。homepage应用负责处理网站的首页界面，upload应用负责处理用户上传信息的交互功能，user\_backend应用负责处理与用户交互主要功能，包括登录，注册，退出登录，更改密码/邮箱，查看书签的实现。与php不同，django资源之间的访问往往不能用简单的相对路径的方法实现，需要运用{%load static%}标签，并且django自带的标签和语法可以轻松实现网页的定制和简单逻辑实现。**    2. **此处管理的数据库是sqlite,是轻量级的数据库并且易于编写。最重要的表是User和Caches，它们要管理大多数的用户数据，同时两张表之间要建立外键的映射，即每一个用户有许多份网页快照，每份网页快照都被一个用户拥有，通过数据库的管理可以让我们轻松的查找出符合条件的信息并返回给用户。**    3. **网站信息的保存使用腾讯云对象存储，为此需要专门了解其相应的API调用方式，如设定用户信息，地址信息，文件信息，访问接口等，为此专门封装了一个python模块，方便调用。**    4. **使用Linux系统与windows系统的最大区别是命令行的使用并且几乎没有图形界面。而宝塔系统可以提供图形化的管理方式，极大方便了删除，编辑和更改文件权限等操作。在部署网站时，项目路径的选择，uswgi配置文件的编写，错误信息的查看，项目启动与调试等都需要反复操作试错。如FTP文件传输功能开通需要在宝塔界面中开启21端口，同时腾讯云内也要将21端口打开。** 2. **爬虫**   **各个网站的结构各不相同，通过程序访问网站还有被反爬的风险，要做一个通用的快照程序并不容易。尝试各种方案后，决定通过selenium调用chrome，利用chrome自带的保存网页为mhtml文件和进行网页截屏的功能实现这一需求。**  **在爬虫程序中，我们构建了一个webCrawler类。由于chrome保存网页的效率较低，为了提高运行速度，在该类的init方法中，我们一次性打开4个无头模式下的chrome。这个类中实现需求的方法为saveWebpage方法。该方法中，每有一个url传入，我们就调用一个空闲的浏览器，生成一个线程进行爬取，并返回该线程。如果4个浏览器都正忙，则返回空对象（None）。爬到的文件保存在调用函数时指定的路径中。**  **爬取网页的具体流程图如下：**  输入url  检查有无可用浏览器  连接到url  处理url[1]  返回None  有  无  设置浏览器宽高[3]  处理网页弹窗[2]  等待网页加载  创建并返回爬虫线程  在线程中：  保存mhtml和截图数据  输出至文件  **[1]由于selenium调用chrome时，输入的网页必须带上协议头，但用户在输入网页时很可能不会带上这一部分。为解决这一问题，我们在访问用户输入的url时，会检查它是否带有协议头，如果没有则尝试加上http或https。**  **[2]部分网页会弹出alert，如果不处理这个alert则selenium无法继续执行。因此会检测有无alert并进行处理。对部分有登录弹窗的网站（如知乎），则进行特判把弹窗点掉。**  **[3]selenium可以通过chrome获得网页的宽度和高度。但如果直接采用原始宽高，则在保存截图时，部分网页会出现截图宽度不全的情况。因此默认将浏览器宽度设为网页宽度的2倍。如果只想要1倍宽度，也可在实例化webCrawler类时传入参数进行修改。**  **三、测试、运行情况**          **四、设计、开发过程中的难点**  **前端：**   1. **网页**   **难点主要是对屏幕缩放时改变分辨率产生的问题进行处理，好在html5已经极大地方便了此类问题的解决，因此也并未花费过多的时间。**   1. **JS爬虫**   **难点主要有：**  **1.从零开始学习JS语法，到最后的cheerio、fs、http/https等JS自带模块以及外部引入模块JZzip的综合应用，代码实现是很大的一层困难。并且编写爬虫本身也伴随着痛苦的debug过程，对于新上手的JS语言更是如此，整个编写过程磕磕绊绊，**  **2.部分网站的HTML源代码采用内嵌CSS样式，将图片URL置于CSS元素中，而不是img标签下，对cheerio的工作造成了困难；**  **3.更有的网站本身实行“反爬”机制，部分文件资源无法通过爬虫访问。**  **3. 插件：**  **总的来说，由于js在前端修改bug的难度较大，以及报错的信息时常没有太大作用，另外，由于插件具有一系列特殊的设定、chrome api中各种没有点名的特性、网上资料极少等，且开发者初次开发插件，使得程序开发和调试有一定难度。主要的困难罗列如下：**   1. **外部脚本无法直接引用，原因是违背Content Security Policy，缺少对应的权限设置（但由于我们的插件不在固定域名的网页上进行访问，不方便设置权限），最终直接将外部脚本下载到本地引用。** 2. **插件无法发送post请求，最后通过调试工具的network提示等，我们发现，若未指定域名，html的表单post和js的post方法均会在定向的网址前添加当前域名。比如，我们这里，www.ustcpermanent.cn/upload/urlupload，会被添加插件由chrome分配的域名，从而没有定向到正确的网址。** 3. **页面间信息交互：在chrome中，插件有专门分配的域名，独立于用户页面运行，无法直接访问用户页面的DOM等信息。插入内容脚本可以读取页面信息，但内容脚本具有极大的局限性，即不能引用任何js文件及绝大多数的chrome api 。故需要通过频繁的信息传递，在插件与用户两侧来回传递信息：在用户端读取节点、操纵网页，在插件端调用各种方法处理传来的数据。** 4. **截屏的实现：最初采htmlCanvas2截屏，但该方法存在较多问题，且效果不佳。后来采用翻动滚动条逐页使用chrome自带的方法进行截屏的方法实现。（chrome没有截全屏的api）此外，最初插件截屏极不稳定，时常会出现部分截屏结果为undefined的情况，这是由于chrome每秒只允许调用默认最多2次captureVisibleTab()，后来在每次翻页前进行了0.5秒的等待，运行稳定。** 5. **执行顺序：前端调用的大量api都是异步实现的，导致代码的运行顺序可能并不符合预期。这导致了大量bug的产生，也使得它实际上成为最令人困扰的地方。解决方法是，分析需要的运行顺序并灵活使用回调函数的功能：回调函数在异步函数执行完毕后才被调用，能够较好地保证运行的顺序。**   **后端：**   1. **django**   **几乎处处是难点。理解和实现一个网站框架已是一个很重的负担，如何在茫如烟海的错误信息中定位错误，甚至复现错误更是件困难的事情。更不用说在CSDN恶劣的信息垃圾中翻找有用的解答，相互传抄，信息过时，会员付费......程序的崩溃很难理解错误原因，如在调用函数时莫名出现的死循环但是只有一直在刷屏的"WARNING"但是并不终止，还有一次不小心删除数据库时，恢复数据库成了玄学的操作，最后解决办法竟然是建了一个空文件夹。操作Linux系统服务器时的命令行，更改队友提交的代码，甚至是给腾讯云上的错误文档纠错......此外作为后端和项目主要负责人，需要经常与队友沟通，催促进度，如何有效率地开展讨论会议，分配任务和划分ddl也是作为组长需要深刻反思的难题。**   1. **爬虫**   **难点主要在于完整保存网页快照。现有的各类工具可以轻松保存网页的html或是网页上的文件，但几乎没有工具可以较为简单地将整个页面资源（包括图片，css等）保存下来，部分能实现该功能的工具（如PyWebCopy）对中文页面的支持也大多很差，容易出现乱码。如果直接逐一下载html和css文件中的资源并修改路径，则工程量巨大，难以实现。最终在搜寻资料时发现chrome的cdp命令中有将网页打包保存为mhtml的功能，并可被selenium调用，于是决定使用selenium+chrome的方案保存快照。**  **五、小组分工**  **前端：**  **陈治非（网页界面开发），徐航宇（插件功能开发），左桑贝（js爬虫实现）**  **后端：**  **闫泽轩（django项目框架开发，服务器管理，项目总负责人）**  李牧龙（python爬虫实现）  **六、总结与收获**  **前端：对于网页搭建，适配的过程有更深的体验，也深刻地体会到设计语言存在于容易被忽视的细节中；在设计JS爬虫时，不仅学习了JS的基础语法和综合应用，更了解了网站的资源组成和架构，对web技术有了全新的认知。**  **后端：对于网站架构，运行方式，网络协议的了解更加深入，能够开发一个正式的网站项目并且能够部署到真实情况中去。加深了团队协作的经验，能够运用软件开发的一些策略提高项目进度，同时大大提高了编程能力。**  **七、参考资料（文献、书籍、网络资源等）**  **Django说明文档：**  <https://docs.djangoproject.com/zh-hans/4.0/>  **达内互联网教程：**  <https://www.bilibili.com/video/BV1vK4y1o7jH?spm_id_from=333.788.top_right_bar_window_custom_collection.content.click&vd_source=94255fb0c00d533f8918ab39154faf84>  **Chrome拓展程序教程：**  <http://www.kkh86.com/it/chrome-extension-doc/extensions/getstarted.html#update-code>  **Chrome API:**  <http://www.kkh86.com/it/chrome-extension-doc/extensions/api_index.html>  **Chrome浏览器网页截全屏算法以及实现：**  <http://t.zoukankan.com/lovelylife-p-4137287.html>  **用JavaScript爬取网页**  <https://segmentfault.com/a/1190000023034660>  **How to use JSzip**  <https://stuk.github.io/jszip/documentation/examples.html>  **黑马程序员JavaScript全套教程**  <https://www.bilibili.com/video/BV1ux411d75J?spm_id_from=333.999.0.0&vd_source=134ef05209efb3c8ae74fc36da31e913>  **其他：CSDN帖子，博客园，StackOverflow，菜鸟教程** | | | | |
| 附件说明（代码、程序、网站资源） |  | | | |

（每组交一份报告、报告不超过6页）