# 新闻搜索引擎

#### 实验目标

开发一个新闻搜索网站。构建一个完整的系统，实现网页的基本增删改查的功能以及其它扩展功能。

#### 开发环境

php运行环境：phpmystudy的集成环境

数据库图形化工具：phpMyadmin

项目开发软件：Vscode

#### 项目结构

+---css

+---html

+---imgs

| +---icons

| \---user\_imgs

+---js

+---php

+---sql

+---index.html

#### 项目预期功能

1. 用户登录注册功能
2. 用户密码修改功能

2.输入关键字搜索新闻功能

3.新闻排序分页显示功能

4.新闻详情页显示点赞收藏功能

5.用户详情页显示个人信息功能

6.收藏网页展示功能

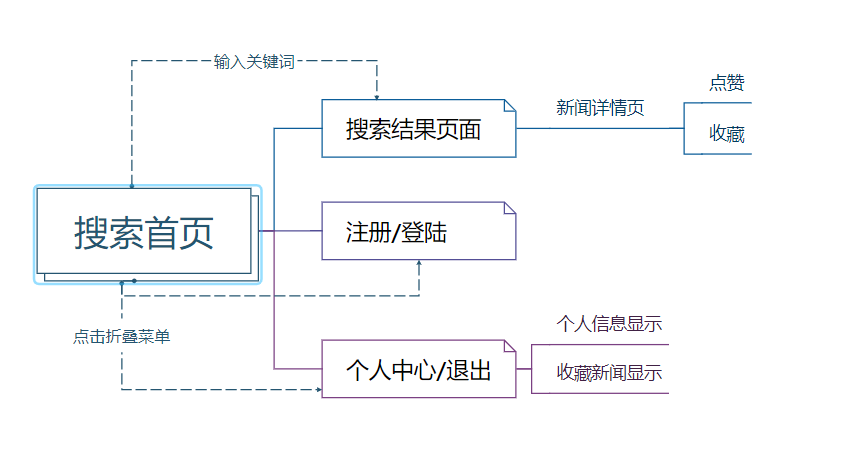
#### 项目预期开发方式

前后端分离开发，前端网页不放入php代码

渲染模式采用客户端渲染，动态更新页面内容

页面分页处理采用后端分页处理，减少一次传输的数据量

#### 网页结构设计



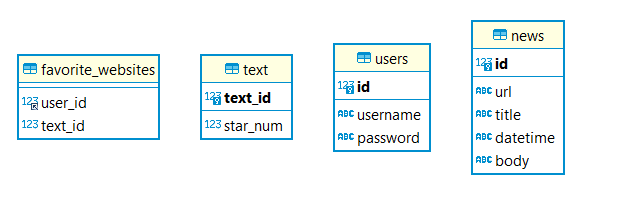
具体网页详情见PPT

#### 数据库设计

用户表

新闻表

用户新闻表



use search\_engine;

 CREATE TABLE  **IF** NOT EXISTS users (

  `id` int(11) NOT NULL,

  `username` varchar(50) COLLATE utf8\_unicode\_ci NOT NULL,

  `password` varchar(255) COLLATE utf8\_unicode\_ci NOT NULL

) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci;

CREATE TABLE **IF** NOT EXISTS news (

    id INTEGER PRIMARY KEY AUTO\_INCREMENT,*--设置为自增属性*

    url TEXT,

    title TEXT,

    datetime TEXT,

    body TEXT

);

CREATE TABLE **IF** NOT EXISTS favorite\_websites (

    user\_id int(11) ,

    text\_id INTEGER,

    FOREIGN KEY (user\_id) REFERENCES users(id)

);

CREATE TABLE **IF** NOT EXISTS text (

    text\_id INTEGER PRIMARY KEY,

    star\_num INT DEFAULT 0,

    FOREIGN KEY (text\_id) REFERENCES news(id)

);

#### 具体功能实现

1. **数据收集**

使用python爬虫收集来自中国新闻网的近五天的一千多条新闻

def **crawl\_news**(news\_pool, min\_body\_len):

*# 初始化计数器 i 为 1*

    i = 1

*# 遍历新闻池中的每个新闻*

    for n, news in **enumerate**(news\_pool):

**print**('%d/%d' % (n, **len**(news\_pool)))

*# 构造 HTTP 请求头部信息*

        req = **urllib**.**request**.**Request**(news[1], headers=headers)

        try:

*# 尝试打开指定 URL，设置超时时间为 10 秒*

            response = **urllib**.**request**.**urlopen**(req, timeout=10)

            html = response.read()

        except **socket**.timeout as err:

*# 处理超时异常，打印错误信息，等待 1 分钟后继续*

**print**('socket.timeout')

**print**(err)

**print**("Sleeping for 1 minute")

**time**.**sleep**(60)

            continue

        except **Exception** as e:

*# 处理其他异常，打印异常信息，等待 1 秒后继续*

**print**("--1---%s:%s %s-----" % (**type**(e), e.reason, news[1]))

**print**("Sleeping for 1 second")

**time**.**sleep**(1)

            continue

*# 使用 BeautifulSoup 解析 HTML 内容*

        soup = **BeautifulSoup**(html, "html.parser")  *# http://www.crummy.com/software/BeautifulSoup/bs4/doc.zh/*

*# 移除 HTML 中的所有脚本标签*

        [s.extract() for s in soup('script')]

        try:

*# 尝试找到包含新闻正文的 div 元素，并获取所有段落标签*

            ps = soup.**find**('div', class\_="left\_zw").**find\_all**('p')

        except **Exception** as e:

*# 处理异常，打印异常信息，等待 1 分钟后继续*

**print**("--2---%s: %s-----" % (**type**(e), news[1]))

**print**("Sleeping for 1 minute")

**time**.**sleep**(60)

            continue

*# 初始化新闻正文内容为空字符串*

        body = ''

*# 遍历每个段落，获取文本内容并添加到正文中*

        for p in ps:

            cur = p.get\_text().strip()

            if cur == '':

                continue

            body += '\t' + cur + '\n'

*# 移除正文中的空格*

        body = body.**replace**(" ", "")

*# 过滤掉不包含关键词的新闻*

        if keyword not in body:  *# 过滤掉乱码新闻*

            continue

*# 过滤掉正文长度小于等于 min\_body\_len 的新闻*

        if **len**(body) <= min\_body\_len:

            continue

*# 创建 MySQL 连接*

        connection = None

        try:

            connection = **mysql**.**connector**.**connect**(\*\*mysql\_config)

            cursor = connection.**cursor**()

*# 插入新闻数据，不包含 id 列*

            sql = "INSERT INTO news (url, title, datetime, body) VALUES (%s, %s, %s, %s)"

            val = (news[1], news[2], news[0], body)

            cursor.**execute**(sql, val)

*# 提交事务*

            connection.**commit**()

*# 更新计数器*

            i += 1

*# 每插入 500 条新闻后，等待 3 分钟*

            if i % 500 == 0:

**print**("Sleeping for 3 minutes")

**time**.**sleep**(180)

        except **Error** as e:

*# 处理数据库错误*

**print**("MySQL Error:", e)

        finally:

*# 关闭数据库连接*

            if connection:

                connection.**close**()

1. **搜索功能以及分页方式**

预想加入页面排序算法，信息检索的大作业使用的是python代码，还没来得及写份php版本的，就先用最基础，只是用关键字执行sql语句去数据库匹配来检索信息，也没有涉及到页面的排序算法。

分页方式采用后端分页，每次返回十条新闻信息，记录当前页面序号。没有采用前端分页，考虑到数据量可能过大，一次全部发送给前端可能响应慢，以及占用客户端存储空间多。

*// 检查连接是否成功*

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

*// 处理搜索逻辑*

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

*// 获取用户提交的关键词*

    $key = **isset**($\_POST['key\_word']) ? $\_POST['key\_word'] : null;

*// 查询匹配的总数据条数*

    $countSql = "SELECT **COUNT**(\*) as total FROM news WHERE title LIKE '%$key%'";

    $countResult = $conn->**query**($countSql);

    $totalCount = $countResult->**fetch\_assoc**()['total'];

*// 每页显示的条数*

    $itemsPerPage = 10;

*// 计算总页数*

    $totalPages = **ceil**($totalCount / $itemsPerPage);

*// 获取用户请求的页码，默认为第一页*

    $currentPage = **isset**($\_POST['page']) ? (int)$\_POST['page'] : 1;

*// 计算偏移量，用于数据库查询*

    $offset = ($currentPage - 1) \* $itemsPerPage;

    $sql = "SELECT \* FROM news WHERE title LIKE '%$key%' LIMIT $offset, $itemsPerPage";

    $result = $conn->**query**($sql);

*// 处理查询结果*

    $docs = [];

    if ($result->num\_rows > 0) {

        while ($row = $result->**fetch\_assoc**()) {

            $doc = [

                'id' => $row['id'],

                'url' => $row['url'],

                'title' => $row['title'],

                'datetime' => $row['datetime'],

                'body' => $row['body'],

            ];

            $docs[] = $doc;

        }

    } else {

*// 没有匹配的新闻*

        $docs = [];

        $error = true;

    }

*// 将结果以 JSON 格式返回，包括分页信息*

**header**('Content-Type: application/json');

**echo** **json\_encode**(['docs' => $docs, 'total\_pages' => $totalPages, 'current\_page' => $currentPage, 'error' => $error]);

}

*//分页逻辑在后端处理*

$conn->**close**();

?>

1. **用户登录信息保存**

使用cookie保存在浏览器中

后端

*// 处理用户登录*

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $username = $\_POST["username"];

    $password = $\_POST["password"];

*// 查询数据库*

    $sql = "SELECT \* FROM users WHERE username = '$username'";

    $result = $conn->**query**($sql);

    if ($result) {

        if ($result->num\_rows > 0) {

*// 用户存在，验证密码*

            $row = $result->**fetch\_assoc**();

            if (**password\_verify**($password, $row["password"])) {

*// 登录成功*

*// 设置Cookie，这里假设使用用户的ID作为标识信息*

                $userID = $row["id"];

**setcookie**("userID", $userID, **time**() + (7 \* 24 \* 60 \* 60), "/"); *// 7天有效期，根据需求设置*

**echo** **json\_encode**(["success" => true, "message" => "Login successful"]);

            } else {

*// 密码错误*

**echo** **json\_encode**(["success" => false, "message" => "Incorrect password"]);

            }

        } else {

*// 用户不存在*

**echo** **json\_encode**(["success" => false, "message" => "User not found"]);

        }

    } else {

*// 查询失败*

**echo** **json\_encode**(["success" => false, "message" => "Query failed: " . $conn->error]);

    }

}

前端

 function **setCookie**(name, value, days) {

        const expires = new **Date**();

        expires.**setTime**(expires.**getTime**() + (days \* 24 \* 60 \* 60 \* 1000));

        document.cookie = `${name}=${value};expires=${expires.**toUTCString**()};path=/`;

    }

1. **用户注册密码哈希加密**

后端处理

*// 处理用户注册*

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

    $username = $\_POST["username"];

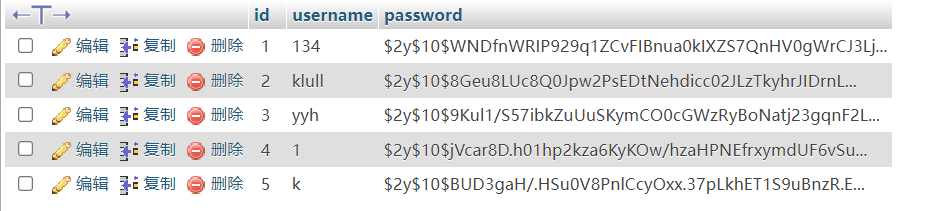
    $password = **password\_hash**($\_POST["password"], PASSWORD\_DEFAULT); *// 使用密码哈希进行存储*

*// 检查用户名是否已经存在*

    $checkUserQuery = "SELECT \* FROM users WHERE username = '$username'";

    $checkUserResult = $conn->**query**($checkUserQuery);

数据库



**5.新闻点赞收藏功能**

在新闻的详情页面下面放入点赞和收藏功能。进行点赞和收藏之前首先要对用户的登陆状态进行检测，如果未登陆，则提醒用户先登陆。如果已登陆，则将用户的点赞收藏信息存入数据库对应表中。

<script>

*// 点赞收藏*

    var likeIcon = document.**getElementById**('likeIcon');

    var favoriteIcon = document.**getElementById**('favoriteIcon');

    var isLiked = false; *// 初始状态为未喜欢*

    var isFavorite = false; *// 初始状态为未收藏*

    function **toggleLike**() {

*// 检查用户登录状态*

        if (**checkLoginStatus**()) {

*// 切换喜欢状态*

            isLiked = !isLiked;

*// 根据喜欢状态切换图标*

            if (isLiked) {

                likeIcon.src = 'imgs/icons/heart-solid.svg'; *// 切换为实心心形图标*

*// 在这里添加执行喜欢相关的操作*

**likeOrLogin**();

            } else {

                likeIcon.src = 'imgs/icons/heart-regular.svg'; *// 切换为空心心形图标*

*// 在这里添加执行取消喜欢相关的操作*

**unlikeOrLogin**();

            }

        } else {

*// 用户未登录，提示登录*

**alert**("Please log in to like the news.");

        }

    }

    function **toggleFavorite**() {

*// 检查用户登录状态*

        if (**checkLoginStatus**()) {

*// 切换收藏状态*

            isFavorite = !isFavorite;

*// 根据收藏状态切换图标*

            if (isFavorite) {

                favoriteIcon.src = 'imgs/icons/star-solid.svg'; *// 切换为实心星星图标*

*// 在这里添加执行收藏相关的操作*

**favoriteOrLogin**();

            } else {

                favoriteIcon.src = 'imgs/icons/star-regular.svg'; *// 切换为空心星星图标*

*// 在这里添加执行取消收藏相关的操作*

**unfavoriteOrLogin**();

            }

        } else {

*// 用户未登录，提示登录*

**alert**("Please log in to favorite the news.");

        }

    }

**6.用户个人主页实现**

预期功能，展现用户的昵称，提供修改密码功能，设置用户头像功能，如果没有设置头像，则展示默认头像。头像功能还未完全实现，目前只提供在前端显示，还没设置后端存入数据库。

      <script>

            function **showChangePasswordModal**() {

                var modal = document.**getElementById**('passwordModal');

                modal.style.display = 'flex';

            }

            function **closeChangePasswordModal**() {

                var modal = document.**getElementById**('passwordModal');

                modal.style.display = 'none';

            }

            function **previewImage**(input) {

                var userPic = document.**getElementById**('userPic');

                if (input.files && input.files[0]) {

                    var reader = new **FileReader**();

                    reader.**onload** = function (e) {

                        userPic.src = e.target.result;

                    };

                    reader.**readAsDataURL**(input.files[0]);

                }

            }

        </script>

#### 项目总结

目前预期的功能都已大部分实现，小部分功能还等待优化完善。项目也还有很大的迭代空间，许多其它新颖功能也可以继续添加。这次项目很锻炼我的动手能力，很多前后端接口的bug一调就是几个小时，极大锻炼了我的耐力。但是能做出来这么一个项目，还是很带来成就感的，后面也打算继续完善这个项目然后开源到github上面，也算是一个项目经历了。