

# 北京邮电大学



## 《学生游学系统》项目周报

### ——第九周周报

学院：计算机学院（国家示范性软件学院）

专业：计算机科学与技术

班级：2022211305

小组：第 09 小组

成员：张晨阳 2022211683

廖轩毅 2022211637

徐路 2022211644

2024 年 4 月 28 号

# 一、本周工作进度

## 1. 后端算法模块（完成度 25%+25%+20%+20%=90%）

- (1) 实现场所查询模块根据类别、关键词查询场所功能
- (2) 实现场所查询模块根据距离排序(快速排序)显示功能(下图不完全显示)

```

// 查询指定公里内的设施
std::cout << "Please input the search radius (in meters): ";
double searchRadius;
std::cin >> searchRadius;
// double searchRadius = 1000; // 单位为米 (或者Node类位置单位相同)
auto nearbyFacilities = query.findNearbyFacilities(startLocation, searchRadius);

query.sortFacilitiesByDistance(nearbyFacilities, 0, nearbyFacilities.size() - 1);

// 输出查询结果
std::cout << "Facilities within " << searchRadius << " m radius:" << std::endl;
for (const auto& facility : nearbyFacilities) {
    std::cout << "Facility Name: " << facility->getName() << "    Distance: " << facility->getDis();
}

// 过滤查询结果
std::cout << "\nPlease input the interested facility type: ";
// std::string interestedType;
// std::cin >> interestedType;
std::string interestedType = "餐饮";
auto filteredFacilities = query.filterResultsByCategory(nearbyFacilities, interestedType);

// 输出过滤后的结果
std::cout << "\nFiltered Facilities (Type - " << interestedType << "):" << std::endl;
for (const auto& facility : filteredFacilities) {
    std::cout << "Facility Name: " << facility->getName() << "    Distance: " << facility->getDis();
}

std::string interestedType1 = "卫生间";
auto filteredFacilities1 = query.filterResultsByCategory(nearbyFacilities, interestedType1);

// 输出过滤后的结果
std::cout << "\nFiltered Facilities (Type - " << interestedType1 << "):" << std::endl;

```

Please input the search radius (in meters): 250  
Facilities within 250 m radius:  
Facility Name: 超市1号 Distance: 30  
Facility Name: 卫生间2号 Distance: 80  
Facility Name: 咖啡店 Distance: 150  
Facility Name: 饭店1号 Distance: 150  
Facility Name: 医务室 Distance: 150  
Facility Name: 饭店2号 Distance: 240  
Facility Name: 体育馆 Distance: 250

Filtered Facilities (Type - 餐饮):  
Facility Name: 咖啡店 Distance: 150  
Facility Name: 饭店1号 Distance: 150  
Facility Name: 饭店2号 Distance: 240

Filtered Facilities (Type - 卫生间):  
Facility Name: 卫生间2号 Distance: 80

PS D:\Users\SevenGrass\Documents\WILLIAMCHANG\DataStruct  
Please input the search radius (in meters): 1000  
Facilities within 1000 m radius:  
Facility Name: 超市1号 Distance: 30  
Facility Name: 卫生间2号 Distance: 80  
Facility Name: 咖啡店 Distance: 150  
Facility Name: 饭店1号 Distance: 150  
Facility Name: 医务室 Distance: 150  
Facility Name: 饭店2号 Distance: 240  
Facility Name: 体育馆 Distance: 250  
Facility Name: 超市2号 Distance: 270  
Facility Name: 卫生间3号 Distance: 280  
Facility Name: 洗衣房 Distance: 280

Filtered Facilities (Type - 餐饮):  
Facility Name: 咖啡店 Distance: 150  
Facility Name: 饭店1号 Distance: 150  
Facility Name: 饭店2号 Distance: 240

Filtered Facilities (Type - 卫生间):  
Facility Name: 卫生间2号 Distance: 80  
Facility Name: 卫生间3号 Distance: 280

- (3) 实现日记管理模块根据热度和评分排序（快速排序）功能
- (4) 实现日记管理模块通过日记名、作者 id、作者名、景点、日记内容搜索日记功能（下图不完全展示）

```

40 diary diary1("HomeDisire", "666", "北京邮电大学", "I want to go back to shenzhen!");
41 diary diary2("NoFunAnyMore", "250", "北京邮电大学", "Here is not fun!");
42 diary diary3("ShitWeather", "555", "北京首都机场", "My plane was cancelled because of the fxxking weather!");
43 diary diary4("SooooooTired", "555", "北京首都机场", "I want to SLEEP!!!!!!!!!!!!!!!!!!!!!!");
44 diary1.popularity=12000;
45 diary1.rating=15000;
46 diary2.popularity=13000;
47 diary2.rating=14000;
48 diary3.popularity=20000;
49 diary3.rating=50000;
50 diary4.popularity=30000;
51 diary4.rating=40000;
52
53 diaryManager.addDiary(diary1);
54 diaryManager.addDiary(diary2);
55 diaryManager.addDiary(diary3);
56 diaryManager.addDiary(diary4);
57
58 //排序测试
59 diaryManager.diarySearch("-1", "-1", "-1", "北京邮电大学", "-1", 0);
60
61 //编译测试
62 //编译测试
63 //编译测试
64 //编译测试
65 //编译测试
66 //编译测试
67 //编译测试
68 //编译测试
69 //编译测试
70 //编译测试
71 //编译测试
72 //编译测试
73 //编译测试
74 //编译测试
75 //编译测试
76 //编译测试
77 //编译测试
78 //编译测试
79 //编译测试
80 //编译测试
81 //编译测试
82 //编译测试
83 //编译测试
84 //编译测试
85 //编译测试
86 //编译测试
87 //编译测试
88 //编译测试
89 //编译测试
90 //编译测试
91 //编译测试
92 //编译测试
93 //编译测试
94 //编译测试
95 //编译测试
96 //编译测试
97 //编译测试
98 //编译测试
99 //编译测试
100 //编译测试

```

Title: ShitWeather  
AuthorID: 555  
Destination: 北京首都机场  
Content: My plane was cancelled because of the fxxking weather!  
Popularity: 20000  
Rating: 50000

Title: HomeDisire  
AuthorID: 666  
Destination: 北京邮电大学  
Content: I want to go back to shenzhen!  
Popularity: 12000  
Rating: 15000

Title: NoFunAnyMore  
AuthorID: 250  
Destination: 北京邮电大学  
Content: Here is not fun!  
Popularity: 13000  
Rating: 14000

Title: SoooooTired  
AuthorID: 555  
Destination: 北京首都机场  
Content: I want to SLEEP!!!!!!!!!!!!!!!!!!!!!!  
Popularity: 30000  
Rating: 40000

Press any key to continue . . . |

Diary diary1("HomeDisire", "666", "北京邮电大学", "I want to go back to shenzhen!");  
Diary diary2("NoFunAnyMore", "250", "北京邮电大学", "Here is not fun!");  
Diary diary3("ShitWeather", "555", "北京首都机场", "My plane was cancelled because of the fxxking weather!");  
Diary diary4("SooooooTired", "555", "北京首都机场", "I want to SLEEP!!!!!!!!!!!!!!!!!!!!!!");  
diary1.popularity=12000;  
diary1.rating=15000;  
diary2.popularity=13000;  
diary2.rating=14000;  
diary3.popularity=20000;  
diary3.rating=50000;  
diary4.popularity=30000;  
diary4.rating=40000;  
diaryManager.addDiary(diary1);  
diaryManager.addDiary(diary2);  
diaryManager.addDiary(diary3);  
diaryManager.addDiary(diary4);  
//排序测试  
diaryManager.diarySearch("-1", "-1", "-1", "北京邮电大学", "-1", 0);

## 2. 用户登录后端模块（完成度 90%）

(1) 实现防止注册重复用户名功能（下图不完全展示）

The screenshot shows a Visual Studio IDE with a C++ file named `test1.cpp`. The code includes `<iostream>` and `<mysql.h>`, and uses the `std` namespace. It defines two functions: `bool jsUsernameRegistered(MYSQL* conn, const string& username)` and `bool retrievePassword(MYSQL* conn, const string& username)`. The first function checks if a username is already registered by querying a MySQL database. The second function retrieves the password for a given username. The console window on the right shows the output of the program, including prompts for registration and login, and the successful execution of an SQL insert statement.

```
1 ~\include <iostream>
2 ~\include <mysql.h>
3
4 using namespace std;
5
6 //函数声明
7 bool jsUsernameRegistered(MYSQL* conn, const string& username) {
8
9     string query = "SELECT COUNT(*) FROM users WHERE username = '" + username + "'";
10
11     if (mysql_query(conn, query.c_str())) {
12         cout << "查询失败: " << mysql_error(conn) << endl;
13         return false;
14     }
15
16     MYSQL_RES* result = mysql_store_result(conn);
17     if (result == NULL) {
18         cout << "获取查询结果失败: " << mysql_error(conn) << endl;
19         return false;
20     }
21
22     MYSQL_ROW row = mysql_fetch_row(result);
23     int count = atoi(row[0]);
24
25     mysql_free_result(result);
26
27     return (count > 0);
28 }
29
30 bool retrievePassword(MYSQL* conn, const string& username) {
31     string query = "SELECT password FROM users WHERE username = '" + username + "'";
32
33     // 执行查询
34     if (mysql_query(conn, query.c_str())) {
35         cout << "查询失败: " << mysql_error(conn) << endl;
36         return false;
37     }
38 }
```

Microsoft Visual Studio 调试

```
=====
1. 注册
2. 登录
3. 忘记密码
其余为退出
请输入要执行的操作: 1
请输入要注册的用户名: Tom
用户名已被注册
请输入要注册的用户名: Mary
用户名已被注册
请输入要注册的用户名: willian
用户名已被注册
请输入要注册的用户名: Lisa
用户名未被注册
请输入注册账号名: Lisa
请输入密码: 152928
执行sql语句: INSERT INTO users(username, password) VALUES("Lisa", "152928")
sql语句执行成功!

E:\vs2022-files\Test\vs64\Debug\Test.exe (进程 4948)已退出, 代码为 1.
要在调试停止时自动关闭控制台, 请启用"工具"->"选项"->"调试"->"调试停止时自动关闭控制台".
按任意键关闭此窗口. . .
```

(2) 实现根据用户名找回密码、修改密码功能（下图不完全展示）

The screenshot shows a Visual Studio IDE with a C++ file named `test1.cpp`. The code includes `<iostream>` and `<mysql.h>`, and uses the `std` namespace. It defines a function `bool retrievePassword(MYSQL* conn, const string& username)` that retrieves the password for a given username. The console window on the right shows the output of the program, including prompts for password retrieval and the successful execution of an SQL query.

```
28
29 bool retrievePassword(MYSQL* conn, const string& username) {
30     string query = "SELECT password FROM users WHERE username = '" + username + "'";
31
32     // 执行查询
33     if (mysql_query(conn, query.c_str())) {
34         cout << "查询失败: " << mysql_error(conn) << endl;
35         return false;
36     }
37
38     // 获取查询结果
39     MYSQL_RES* result = mysql_store_result(conn);
40     if (result == NULL) {
41         cout << "获取查询结果失败: " << mysql_error(conn) << endl;
42         return false;
43     }
44
45     // 获取查询结果中的数据行
46     MYSQL_ROW row = mysql_fetch_row(result);
47
48     // 检查是否找到了对应用户名的密码
49     if (row == NULL) {
50         cout << "用户名不存在" << endl;
51         mysql_free_result(result);
52         return false;
53     }
54
55     // 打印找回密码
56     cout << "用户名为 " << username << " 的密码是: " << row[0] << endl;
57
58     // 释放查询结果内存
59     mysql_free_result(result);
60
61     return true;
62 }
63
64
```

Microsoft Visual Studio 调试

```
=====
1. 注册
2. 忘记密码
3. 忘记密码
其余为退出
请输入要执行的操作: 3
请输入要找回密码的用户名: Bob
用户名为 Bob 的密码是: 555666
啥也没发生!

E:\vs2022-files\Test\vs64\Debug\Test.exe (进程 5980)已退出, 代码为 0.
要在调试停止时自动关闭控制台, 请启用"工具"->"选项"->"调试"->"调试停止时自动关闭控制台".
按任意键关闭此窗口. . .
```

## 二、待完成任务&待解决问题

### 1. 待完成任务

- (1) 各模块与其对应的数据库的配置与连接
- (2) 所有模块的完全整合
- (3) 一些 stl 的功能、数据库功能的使用改为自实现

### 2. 待解决问题

- (1) 前端知识的缺乏
- (2) 一些边界问题导致 / 由错误输入导致 / 内存管理导致的 bug 的修改

## 三、下周工作安排

### 1. 后端算法部分

- (1) 将所有子模块整合连接在一起，使用同一个主函数，即实现主模块

### 2. 数据库部分

- (1) 配置所有子模块的数据库

### 3. 前端部分

- (1) 学习前端知识，初步实现一些前端界面
- (2) 通过实践前端知识，适当修改原计划中前端部分内容