CS101 · Introduction to Computer Science

方磊，王泓璋 /
12211631,12211633

Homework 1

June 7, 2024

1 组员贡献与相关信息

本项目由以下两位同学合作完成：

- 方磊，学号 12211631
- 王泓璋，学号 12211633

实验课时间：周四 3-4 朱悦铭综合组员贡献权重为方磊：55%，王泓璋：45%。分别工作如下：

- Task1：基础 API 设计 方磊和王泓璋共同完成
- Task2：高级任务
  - Task2.1：使用 mysql 数据库完成项目 方磊和王泓璋共同完成
  - Task2.2：高级 api 设计 方磊和王泓璋共同完成
  - Task2.3：封装并实现后端服务器 方磊完成
  - Task2.4：web 前端设计和实现 方磊完成
  - Task2.5：数据库高级功能 方磊完成
  - Task2.6：高并发测试 方磊完成
- Task3：项目报告编写 方磊和王泓璋共同完成

相关文件说明如下：

文件路径	作用
config.py	项目的配置文件
config_test.py	测试环境的配置文件
pressure_test.py	性能压力测试脚本
requirements.txt	项目依赖包列表
run.py	启动 Flask 应用的主脚本
app/extensions.py	扩展库初始化文件
app/models.py	数据模型定义
app/__init__.py	应用初始化文件
app/admin/routes.py	管理员相关路由
app/admin/__init__.py	管理员模块初始化文件
app/auth/routes.py	认证相关路由
app/auth/__init__.py	认证模块初始化文件
app/main/routes.py	主功能相关路由
app/main/utils.py	主功能相关工具函数
app/main/__init__.py	主功能模块初始化文件
app/static/index.html	静态主页文件
app/templates/index.html	动态主页模板文件
migrations/alembic.ini	Alembic 配置文件
migrations/env.py	Alembic 环境配置文件
migrations/README	Alembic 说明文件
migrations/script.py.mako	Alembic 脚本模板文件

tests/test_models.py	数据模型测试文件
tests/__init__.py	测试模块初始化文件

2 Task1：基础 API 设计

- 站点的增删改：
  - station_insert: 接收"chinese_name", "english_name", "district", "intro", "status" 来描述站点的信息并将其插入到数据库内，其中 status 表示站点的状态如“关闭中”，“运营中”。该函数返回用于表示操作成功与否的提示。
  - station_delete: 接收"chinese_name" 或"english_name" 来描述站点的信息并将其从数据库中删除。该函数返回用于表示操作成功与否的提示。
  - station_update: 接收"old_english_name", "chinese_name", "english_name", "district", "intro", "status", 其中"old_english_name" 用于在数据库中搜索待修改的站点，其他参数为待更新内容。该函数返回用于表示操作成功与否的提示。
  - 参数类型："chinese_name", "english_name", "district", "intro", "status", "old_english_name" 均为 Varchar 类型。
- 线路的增删改：
  - line_insert: 接收"line_name", "start_time", "end_time", "intro", "mileage", "color", "first_opening", "url" 来描述线路的信息并将其插入到数据库内。该函数返回用于表示操作成功与否的提示。
  - line_delete: 接收"line_name" 来描述线路的信息并将其从数据库中删除。该函数返回用于表示操作成功与否的提示。
  - line_update: 接收"old_name", "line_name", "start_time", "end_time", "intro", "mileage", "color", "first_opening", "url", 其中"old_name" 用于在数据库中搜索待修改的线路，其他参数为待更新内容。该函数返回用于表示操作成功与否的提示。
  - 参数类型："line_name", "district", "intro", "color", "url", "old_name" 均为 Varchar 类型；"start_time", "end_time" 均为 Time 类型；"first_opnening" 为 Date 类型。
- 站点与线路的关系的增删：
  - relation_insert: 接收"english_name_list", "line_name", "after_name", 其中"english_name_list" 用于描述待插入的若干站点，"line_name" 用于描述插入的目标线路，"after_name" 用于描述在哪一站之后插入。该函数返回用于表示操作成功与否的提示。
  - relation_insert_at_position: 接收"english_name_list", "line_name", "position", 其中"english_name_list" 用于描述待插入的若干站点，"line_name" 用于描述插入的目标线路，"position" 用于描述在什么位置插入。该函数返回用于表示操作成功与否的提示。
  - relation_delete: 接收"english_name", "line_name" 用于在数据库中搜索需要删除的关系。该函数返回用于表示操作成功与否的提示。
  - 参数类型："english_name_list", "line_name", "after_name" 均为 Varchar 类型；"position" 为 Int 类型。
- 某条线路上某站前后 n 站的查询：
  - get_station: 接收"english_name", "line_name", "direction", "english_name" 描述基准站点，"line_name" 描述在哪条线上搜索，"direction" 描述前后以及多少站，向前时为负，向后时为正。该函数返回搜索得到的站点名称。
  - 参数类型："english_name", "line_name" 均为 Varchar 类型；"direction" 为 Int 类型。
- 乘客或公交卡上车功能：

  - board: 接收”user”, ”start_station”, 并获取被调用时的系统日期时间以得到”start_date”, ”start_time”, 随后将其插入数据库并保持”end_time”, ”end_station” 均为 Null。该函数返回用于表示操作成功与否的提示。
  - 参数类型: ”user”, ”start_station” 均为 Varchar 类型。

- 乘客或公交卡下车功能：
  - exit: 接收”user”, ”end_station”, ”bussiness_type”, 并获取被调用时的系统日期时间以得到”end_date”, ”end_time”, 随后将其插入数据库中”user” 相同而”end_time” 为 Null 的最晚的一行。随后借助预先通过 Prise.xlsx 生成的 station_Price 数据表以及”bussiness_type” 来计算该趟地铁所需的价格，并自动在 passenger 或者 card 表中扣除对应的金额。如果金额不足则无法成功出站。该函数返回用于表示操作成功与否的提示。
  - 参数类型: ”user”, ”end_station” 均为 Varchar 类型；”bussiness_type” 为 boolean 类型。
- 查询已上车但未下车的乘客：
  - active_rides: 不接收参数，直接向数据库发起查询并获取所有”end_time” 为 Null 的数据，并保留其中的”user”, ”start_station”, ”start_date”, ”start_time”。函数返回一个列表，包含所有查询到的行。
  - 参数类型：无参数。

3 Task2：高级任务

3.1 Task2.1：使用 mysql 数据库完成项目

如图所示，本项目的数据库完全使用 mysql 数据库完成。

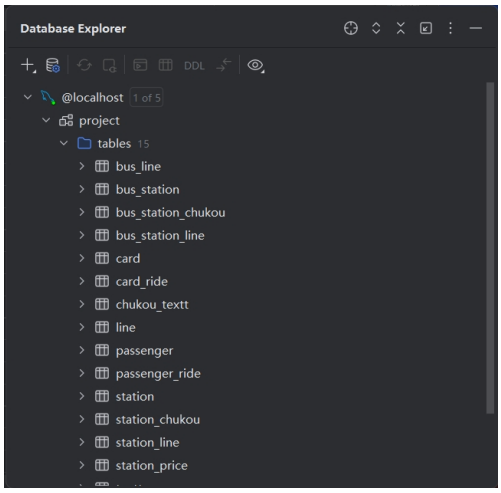

Figure 1: MySQL 数据库

3.2 Task2.2：高级 api 设计

本项目的高级 API 设计如下：

- **POST /admin/get_station_by_position**：根据站点名称和线路名称获取向前和向后第 n 站的站点名称，需提供 station_name、line_name 和 n 参数，成功时返回站点名称，失败时返回错误信息。
- **GET /admin/station_graph**：获取站点图，无需参数，返回站点图的 .xlsx 文件，失败时返回错误信息。
- **POST /admin/shortest_path**：获取从源站点到目标站点的最短路径，需提供 source 和 target 参数，成功时返回路径信息，失败时返回错误信息。

- **POST /admin/alternative_paths**：获取从源站点到目标站点的最多三条最短路径，需提供 source 和 target 参数，成功时返回路径信息，失败时返回错误信息。
- **POST /auth/register**：注册卡或乘客，需提供 register_type 和相应的注册信息，成功时返回注册成功信息，失败时返回错误信息。
- **POST /auth/recharge**：为卡充值，需提供 code 和 money 参数，成功时返回充值后的金额，失败时返回错误信息。
- **GET /**：渲染主页，无需参数，返回主页 HTML。
- **POST /board**：记录乘客上车信息，需提供 user 和 start_station 参数，成功时返回成功信息，失败时返回错误信息。
- **POST /exit**：记录乘客下车信息并计算票价，需提供 user、end_station 和 bussiness_type 参数，成功时返回票价信息，失败时返回错误信息。
- **GET /active_rides**：获取所有未结束的乘车记录，无需参数，成功时返回乘车记录信息，失败时返回错误信息。
- **POST /search_rides**：根据条件搜索乘车记录，需提供搜索条件参数，成功时返回乘车记录信息，失败时返回错误信息。

3.3 Task2.3：封装并实现后端服务器

本项目的后端服务器有以下特点：

- **使用 Flask 框架**：Flask 是一个轻量级的 Python Web 框架，本项目使用 Flask 框架实现后端服务器。
- **使用 Blueprint**：Blueprint 是 Flask 的一个扩展，可以将应用分解为一些小的模块，本项目使用 Blueprint 实现不同的 API。
- **使用 SQLAlchemy**：SQLAlchemy 是一个 Python SQL 工具包和对象关系映射器，本项目使用 SQLAlchemy 实现数据库操作，实现了连接池和 orm 映射。
- **使用 HTTP/RESTful Web API**：本项目使用 HTTP/RESTful Web API 实现前后端交互。

3.4 Task2.4：web 前端设计和实现

本项目的 web 前端设计如下：

- **主页**：选择是乘客页面还是管理员页面。
- **乘客页面**：提供注册、充值、上车、下车等功能。
- **管理员页面**：提供获取站点图、获取最短路径、获取指定位置的站点等功能。

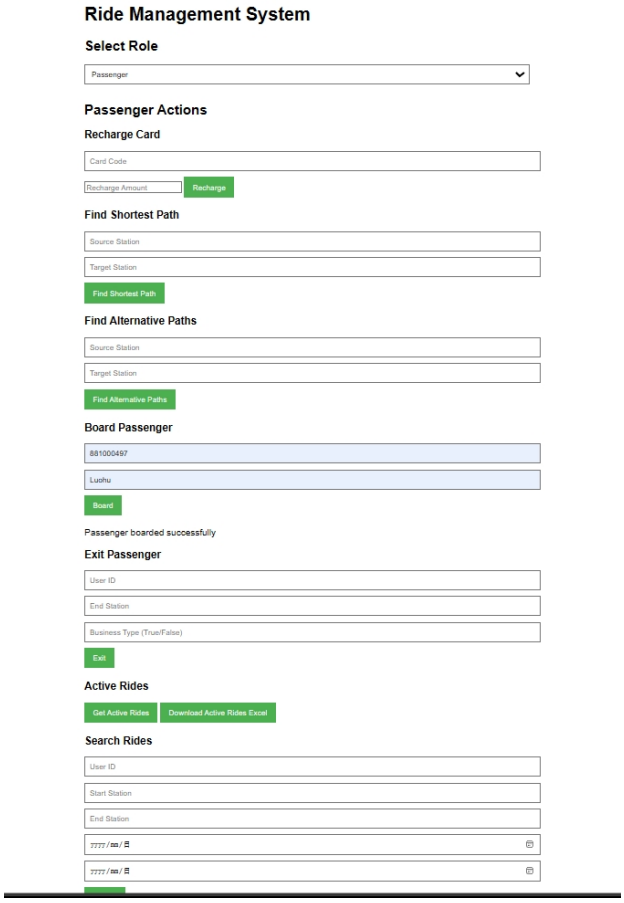

Figure 2: 用户页面

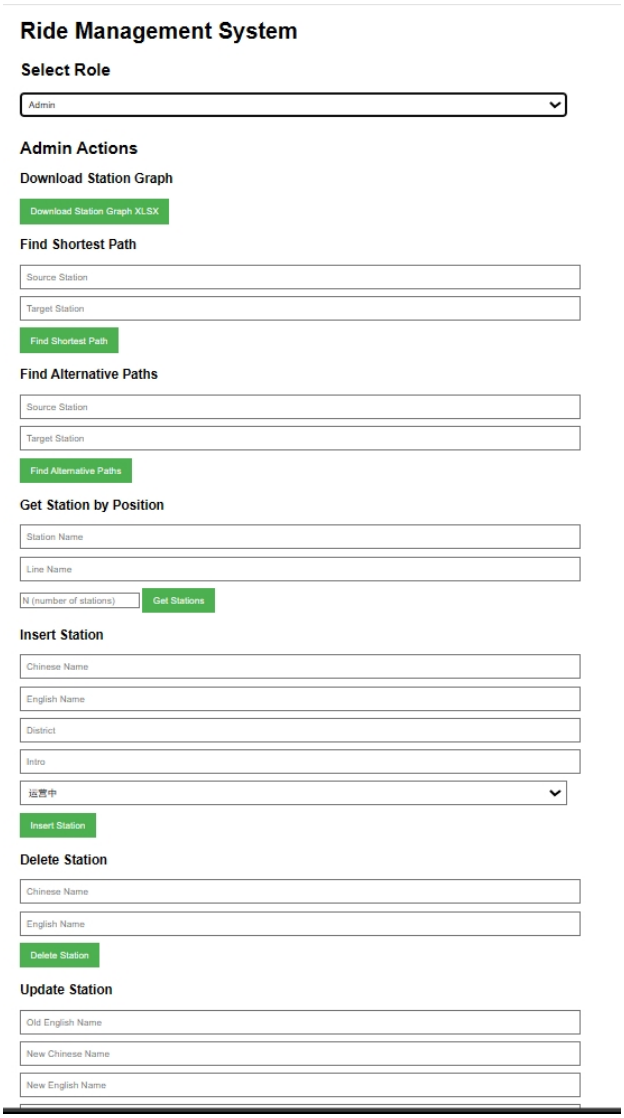

Figure 3: 管理员页面

3.5 Task2.5：数据库高级功能

本项目的数据库高级功能如下：

- **设置权限**：设置管理员和乘客的权限，管理员可以获取站点图、获取最短路径、获取指定位置的站点等功能，乘客可以注册、充值、上车、下车等功能。
- **设置触发器**：设置触发器，当乘客下车时，自动计算票价。当乘客注册时，自动判断身份证号是否合法。

3.6 Task2.6：高并发测试

完成了一个高并发脚本，模拟了 1000 个用户同时访问服务器，测试了服务器的性能。测试结果如下：可以

Total requests	1000
Successful requests	1000
Failed requests	0
Total time (seconds)	25.55
Requests per second	39.13

Table 2: Request Statistics

看到，服务器在高并发下的性能表现良好，没有出现失败请求。