**修订历史:**

| 修改或初始编写日期 | SEPG | 版本 | 说明 | 作者 | 评审时间 | 评审参与人员 | 评审后修改批准日期 | 确认签字人员 |
|---|---|---|---|---|---|---|---|---|
| 2025-03-25 | | 1.0 | 完成功能建模 | 刘文美<br>高妤婕<br>邵任飞 | 2025-03-26 | 杜庆峰<br>邵任飞 | 2025-03-26 | |
| 2025-03-30 | | 1.1 | 完成数据建模与行为建模 | 刘文美<br>高妤婕<br>邵任飞 | 2025-03-31 | 杜庆峰<br>邵任飞 | 2025-03-31 | |
| 2025-04-01 | | 1.2 | 完成文档编写 | 刘文美<br>高妤婕<br>邵任飞 | 2025-04-02 | 杜庆峰<br>邵任飞 | 2025-04-03 | |

# 1 引言

## 1.1 概要设计依据

1．依据概要设计的基本概念进行设计

2．依据前述的需求分析规约进行设计

## 1.2 参考资料

在编写本文档时，我们参考了以下资料：

1．行程规划相关的文章和热门媒体平台的发帖数据

2．现有的行程规划和账目管理的软件与系统

## 1.3 假定和约束

在设计和实现旅游规划小程序时，我们做出了以下假设和约束：

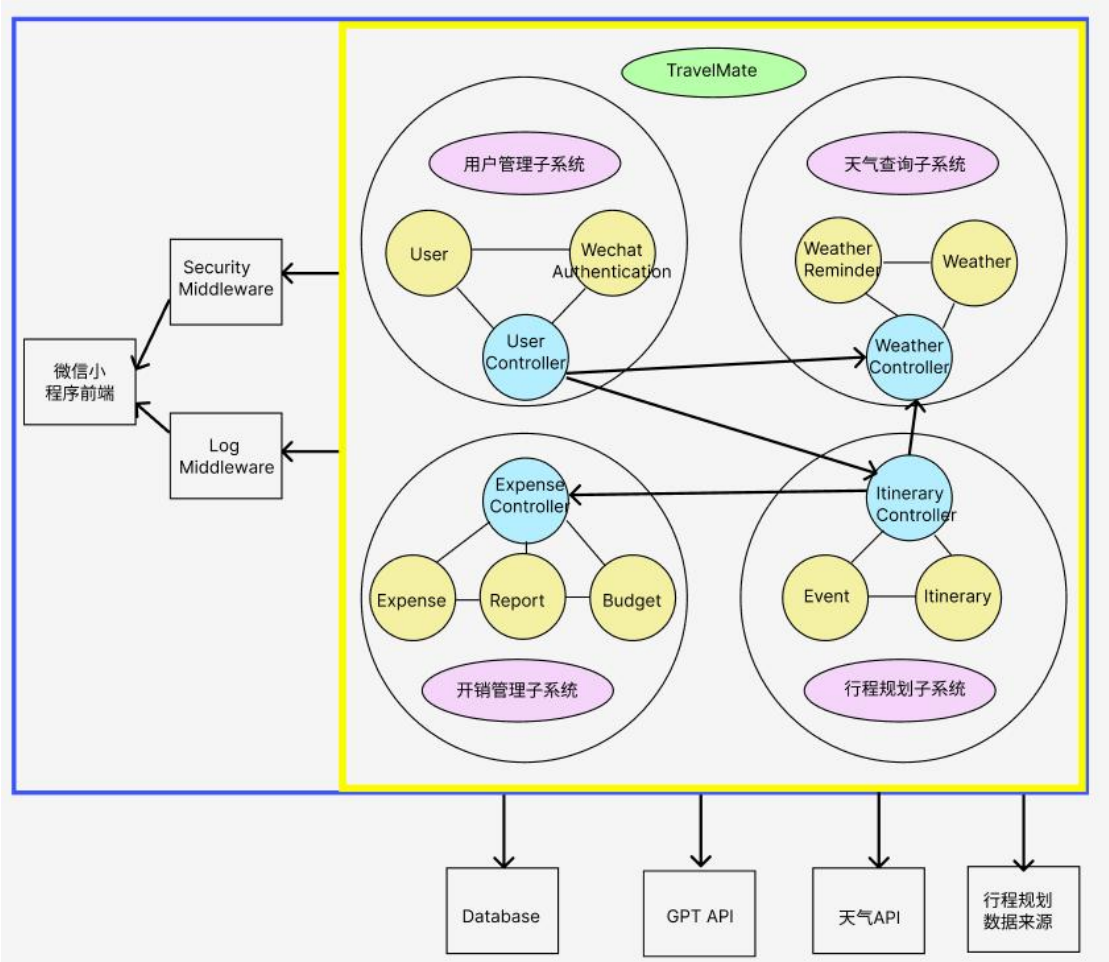
1．系统基于微信小程序平台运行，所有功能依赖于互联网连接以实现数据访问和存储。

2．系统将严格保护用户的个人信息、支出信息和旅游数据，采用适当的安全措施（如数据加密和权限管理）以防止未经授权的访问和数据泄露。

3．系统将提供直观、简洁且用户友好的界面设计，使用户能够轻松规划行程、管理预算并接收智能推荐和提醒功能。

# 2 概要设计

## 2.1 系统体系架构设计




系统体系架构包括以下几个关键部分：

**软件架构**：共分为四个子系统，包括用户管理子系统、天气查询子系统、开销管理子系统和行程规划子系统，详见 2.2。

**中间件**：包括安全中间件和日志中间件。安全中间件用于实现用户认证与授权，可以进行输入校验，避免 SQL 注入或其他恶意请求，可以确保 Web App 的每个请求都符合安全要求，防止未授权访问后端的各个子系统；日志中间件记录每个请求的详细信息，例如时间戳、请求参数、响应时间等，用于排查错误或监控系统运行状态，生成审计日志，满足合规需求。

**外部接口**：包括 Database、GPT API、天气 API 和行程规划数据来源。数据库存储系统的核心数据，如用户信息、行程、预算、天气提醒等，与所有子系统相关联；GPT API 用于行程智能推荐，与行程规划子系统相关联；天气 API 获

取实时天气数据，与天气查询子系统相关联；行程规划数据源提供第三方行程规划数据，如景点推荐、交通信息等，将此数据传给 GPT API 后，实现更好地行程推荐，与行程规划子系统相关联。

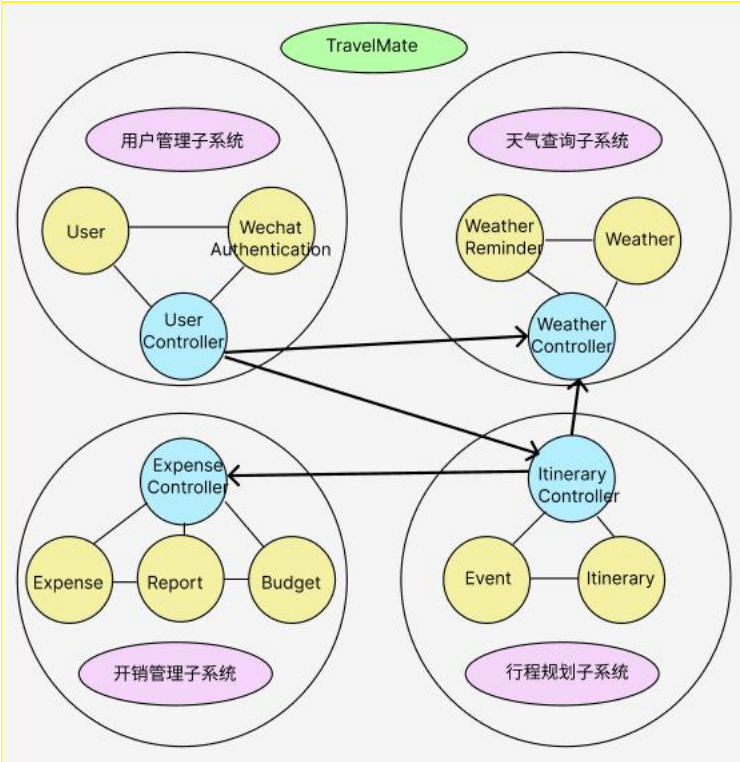
系统组件之间的交互包括下面几个部分：

**前端交互：**前端（微信小程序）通过中间件（Security 和 Log）与后端进行交互。请求分发到具体控制器。

**不同子系统之间的交互：**控制器负责协调不同子系统的数据流通。如 User Controller 和 Weather Controller 协作实现天气提醒绑定功能。

**系统与外部接口：**天气查询子系统依赖天气 API。行程规划子系统依赖 GPT API 和行程规划数据源。

## 2.2 软件体系结构设计




共分为四个子系统，包括用户管理子系统、天气查询子系统、开销管理子系统和行程规划子系统。每个子系统都有一个控制器（Controller），它是子系统的入口。控制器负责处理请求，并与其他控制器协作，实现跨模块的功能。各子系统职责单一，数据流通过控制器完成，避免模块间的直接依赖，便于维护和扩展。

- **用户管理子系统**

  User：用于存储用户的基础信息。

  WechatAuthentication：实现微信登录功能和提供用户认证

UserController：与 WeatherController 交互用于设置天气提醒；与 ItineraryController 交互用于用户的行程管理。

- **天气查询子系统**

  WeatherReminder：提供天气提醒功能。

  Weather：管理和存储天气信息。

  WeatherController：与 ItineraryController 交互为事件提供相关天气信息；与 User Controller 交互为用户绑定天气提醒。

- **开销管理子系统**

  Expense：存储和管理开销。

  Budget：存储和管理预算。

  Report：生成开销和预算的统计报表，用于数据可视化。

  ExpenseController：与 ItineraryController 交互，表示与事件相关的开销和预算。

- **行程规划子系统**

  Event：存储和管理事件。

  Itinerary：存储和管理行程。

  ItineraryController：与 WeatherController 交互，根据行程事件地点和时间，提供天气信息；与 ExpenseController 交互：与开销管理子系统协作，管理事件费用；与 User Controller 交互绑定用户与行程信息。

## 2.3 接口设计

### （一）前后端接口

- **用户管理子系统**

  **（1）用户登录**

  **请求方式：**GET

  **请求路径：**/user/login

  **参数示例：**code=0e1lM7Ga1enqMI0FE0Ha1T0FGv2lM7Gg

  **返回：**

```
{
    "code": 1,
    "msg": "success",
    "data": 681295881
}
```

  **（2）完善用户信息**

  **请求方式：**PUT

请求路径：/user/info

参数示例：

```
{
    "id":681295877,
    "name":"hello",
    "gender":1
}
```

返回：

```
{
    "code": 1,
    "msg": "success",
    "data": 681295881
}
```

（3）得到用户信息

请求方式：GET

请求路径：/user/info

参数示例：userID=681295878

返回：

```
{
    "code": 1,
    "msg": "success",
    "data": {
        "openID": "sadq1dwqeq1aqsgadwd",
        "name": "hahaha",
        "gender": null,
        "id": 681295882
    }
}
```

- 天气查询子系统

（1）查询天气

请求方式：GET

请求路径：/weather/get

参数示例：location=保定&date=2024-12-31

返回：

```
{
    "code": 1,
    "msg": "success",
    "data": {
```

```
            "maxTemperature": 6.0,
            "minTemperature": -3.0,
            "description": "晴",
            "wind": "1-3",
            "date": "2024-12-31",
            "location": "保定"
        }
}
```


**（2）添加提醒**

**请求方式：**POST

**请求路径：**/reminder/add

**参数示例：**


```
{
   "time": "2025-01-03T10:00:00",
   "location": "上海",
   "userId": 681295877,
   "date": "2025-01-03"
}
```


**返回：**


```
{
      "code": 1,
      "msg": "success",
      "data": 1
}
```


**（3）修改天气提醒**

**请求方式：**PUT

**请求路径：**/reminder/modify

**参数示例：**


```
{
   "id":13,
   "time": "2025-01-02T10:00:00"
}
```


**返回：**


```
{
      "code": 1,
      "msg": "success",
      "data": 1
```

}

**（4）删除天气提醒**

**请求方式：**delete

**请求路径：**/reminder/delete

**参数示例：**id=1

**返回：**

{
"code": 1,
"msg": "success",
"data": true
}

**（5）获取某用户天气提醒列表**

**接口说明：天气提醒列表**

**请求方式：**get

**请求路径：**/reminder/get

**参数示例：**userID=681295877

**返回：**

{
"code": 1,
"msg": "success",
"data": [
{
"id": 14,
"time": "2025-01-03T10:00:00",
"location": "上海",
"userId": 681295877,
"date": "2025-01-03"
},
{
"id": 15,
"time": "2025-01-03T10:00:00",
"location": "济南",
"userId": 681295877,
"date": "2025-01-03"
}
]
}

- 行程规划子系统

（1）新建行程

**请求方式：**POST

**请求路径：**/itinerary/add

**参数示例：**

```
{
   "userID": 681295877,
   "name": "带我回北京",
   "startDate": "2024-02-21",
   "endDate": "2024-03-21",
   "location": "北京"
}
```

**返回：**

```
{
      "code": 1,
      "msg": "success",
      "data": 3
}
```

（2）得到某用户的所有行程

**请求方式：**GET

**请求路径：**/itinerary/getall

**参数示例：**userID=681295878

**返回：**

```
{
      "code": 1,
"msg": "success",
"data": [
            {
                  "userID": 681295877,
                  "name": "Trip to Paris",
                  "startDate": "2024-11-21",
                  "endDate": "2024-11-30",
                  "location": "Paris",
                  "id": 1
            },
            {
                  "userID": 681295877,
                  "name": "带我回北京",
```

```
                "startDate": "2024-02-21",
                "endDate": "2024-03-21",
                "location": "北京",
                "id": 3
            }
]
}
```


（3）得到某一个行程的信息

**请求方式：**GET

**请求路径：**/itinerary/getone

**参数示例：**ID=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": {
        "userID": 681295881,
        "name": "巴黎三日游",
        "startDate": "2024-12-11",
        "endDate": "2024-12-13",
        "location": "巴黎",
        "events": [
            {
                "itiID": 1,
                "startTime": "2024-12-11T09:00:00",
                "endTime": "2024-12-11T18:00:00",
                "location": "巴黎教堂",
                "description": "Annual Conference",
                "name": "参观巴黎教堂",
                "type": 2,
                "expenses": [],
                "budgets": [
                    {
                        "id": 1,
                        "eveID": 1,
                        "money": 40.0
                    }
                ],
                "id": 1
```

```
            },
            {
                "itiID": 1,
                "startTime": "2024-12-11T19:00:00",
                "endTime": "2024-12-11T23:00:00",
                "location": "Little Red Door",
                "description": "放松一下，体验世界第五的酒吧",
                "name": "drink something",
                "type": 3,
                "expenses": [],
                "budgets": [],
                "id": 2
            }
        ],
        "id": 1
    }
}
```


（4）**新建事件**

**请求方式：**POST

**请求路径：**/event/add

**参数示例：**


```
{
  "itiID": 1,
  "startTime": "2024-11-22T09:00:00",
  "endTime": "2024-12-22T18:00:00",
  "location": "巴黎教堂",
  "description": "Annual Conference",
  "name": "参观巴黎教堂",
  "type": 2
}
```


**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": 3
}
```


（5）**删除事件**

**请求方式：**delete

**请求路径：**/event/delete

**参数示例：**id=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": true
}
```


（6）修改事件

**请求方式：**PUT

**请求路径：**/event/modify

**参数示例：**


```
{
  "id": 4,
  "startTime": "2029-12-12T19:00:00",
  "endTime": "2029-12-12T23:00:00"
}
```


**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": 4
}
```


（7）修改行程

**请求方式：**PUT

**请求路径：**/itinerary/modify

**参数示例：**


```
{
  "id": 3,
  "name": "北京欢迎你"
}
```


**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": 3
}
```


（8）删除行程

**请求方式：**delete

**请求路径：**/itinerary/delete

**参数示例：**id=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": true
}
```


**（9）行程智能推荐**

**请求方式：**get

**请求路径：**/itinerary/getintrec

**参数示例：**id=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": "# 旅游攻略\n\n## **Day 1: 哈尔滨....."
}
```


- **开销管理子系统**

**（1）新建开销**

**请求方式：**POST

**请求路径：**/expense/add

**参数示例：**


```
{
    "eveID": 1,
    "type": 3,
    "time": "2024-11-21T12:30:00",
    "money": 20.2,
    "name": "buy some tickets"
}
```


**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": 2
}
```


**（2）新建预算**

**请求方式：**POST

**请求路径：**/budget/add

**参数示例：**

```
{
  "eveID": 1,
  "money": 100.0,
}
```

**返回：**

```
{
    "code": 1,
    "msg": "success",
    "data": 1
}
```

（3）修改开销

**请求方式：**PUT

**请求路径：**/expense/modify

**参数示例：**

```
{
    "id":3,
    "money": 50
}
```

**返回：**

```
{
    "code": 1,
    "msg": "success",
    "data": 3
}
```

（4）修改预算

**请求方式：**PUT

**请求路径：**/budget/modify

**参数示例：**

```
{
    "id":1,
    "money": 50
}
```

**返回：**

```
{
    "code": 1,
    "msg": "success",
```

```
    "data": 1
}
```


（5）删除开销

**请求方式：**delete

**请求路径：**/expense/delete

**参数示例：**id=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": true
}
```


（6）删除预算

**请求方式：**delete

**请求路径：**/budget/delete

**参数示例：**id=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": true
}
```


（7）得到某一事件的开销列表

**请求方式：**get

**请求路径：**/expense/event

**参数示例：**eveID=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": [
        {
            "id": 2,
            "eveID": 1,
            "type": 3,
            "time": "2024-11-21T12:30:00",
            "money": 20.0,
            "name": "buy some tickets"
```

```
            },
            {
                "id": 3,
                "eveID": 1,
                "type": 1,
                "time": "2024-11-21T12:50:00",
                "money": 50.0,
                "name": "taking a taxi"
            }
        ]
    }
```


（8）得到某一事件的预算列表

**请求方式：**get

**请求路径：**/budget/event

**参数示例：**eveID=1

**返回：**


```
    {
        "code": 1,
        "msg": "success",
        "data": [
            {
                "id": 2,
                "eveID": 1,
                "money": 100.0
            }
        ]
    }
```


（9）得到某一时间段内的开销列表

**请求方式：**get

**请求路径：**/report/time

**参数示例：**

startTime=2024-11-23T00:00:00&endTime=2024-11-29T23:59:59&userID=681295877

**返回：**


```
    {
        "code": 1,
        "msg": "success",
        "data": {
            "totalExpense": 9063，
```

```
            "expenses": [
                {
                    "id": 8,
                    "eveID": 6,
                    "type": 2,
                    "time": "2024-11-23T12:50:00",
                    "money": 3021.0,
                    "name": "buy food"
                },
                {
                    "id": 9,
                    "eveID": 8,
                    "type": 2,
                    "time": "2024-11-23T12:50:00",
                    "money": 3021.0,
                    "name": "buy food"
                },
                {
                    "id": 10,
                    "eveID": 8,
                    "type": 2,
                    "time": "2024-11-25T12:50:00",
                    "money": 3021.0,
                    "name": "buy food"
                }
            ]
        }
}
```


(10) 得到用户的全部开销列表

**请求方式：**get

**请求路径：**/report/getall

**参数示例：**userID=681295877

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": {
        "totalExpense": 12084,
        "expenses": [
            {
```

```
                "id": 6,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-21T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
                "id": 7,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-21T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
                "id": 8,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-23T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
                "id": 10,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-25T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            }
        ]
    }
}
```


（11）得到某几种类型的开销列表

**请求方式：**get

**请求路径：**/report/type

**参数示例：**userID=681295877&types=1,2,3

**返回：**


```
{
```

```
"code": 1,
"msg": "success",
"data": {
    "totalExpense": 12084,
    "expenses": [
        {
            "id": 6,
            "eveID": 6,
            "type": 2,
            "time": "2024-11-21T12:50:00",
            "money": 3021.0,
            "name": "buy food"
        },
        {
            "id": 7,
            "eveID": 6,
            "type": 2,
            "time": "2024-11-21T12:50:00",
            "money": 3021.0,
            "name": "buy food"
        },
        {
            "id": 8,
            "eveID": 6,
            "type": 2,
            "time": "2024-11-23T12:50:00",
            "money": 3021.0,
            "name": "buy food"
        },
        {
            "id": 10,
            "eveID": 6,
            "type": 2,
            "time": "2024-11-25T12:50:00",
            "money": 3021.0,
            "name": "buy food"
        }
    ]
}
}
```

（12）展示全部行程预算和开销数据

**请求方式：**get

**请求路径：**/report/budgetandexpense

**参数示例：**userID=681295877

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": {
        "totalExpense": 12084,
        "totalBudget": 6000,
        "expenses": [
            {
                "id": 6,
                "eveID": 6,
                "type": 3,
                "time": "2024-11-21T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
                "id": 7,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-21T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
                "id": 8,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-23T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
                "id": 10,
                "eveID": 6,
                "type": 2,
```

```
                "time": "2024-11-25T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            }
        ],
        "budgets": [
            {
                "id": 5,
                "eveID": 6,
                "money": 2000.0
            },
            {
                "id": 6,
                "eveID": 7,
                "money": 4000.0
            }
        ]
    }
}
```


（13）得到部分行程预算和开销数据

**请求方式：**get

**请求路径：**/report/iti

**参数示例：**itiIDs=1,2

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": {
        "totalExpense": 12084,
        "totalBudget": 6000,
        "expenses": [
            {
                "id": 6,
                "eveID": 6,
                "type": 3,
                "time": "2024-11-21T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
```

```
                "id": 7,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-21T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
                "id": 8,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-23T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            },
            {
                "id": 10,
                "eveID": 6,
                "type": 2,
                "time": "2024-11-25T12:50:00",
                "money": 3021.0,
                "name": "buy food"
            }
        ],
        "budgets": [
            {
                "id": 5,
                "eveID": 6,
                "money": 2000.0
            },
            {
                "id": 6,
                "eveID": 7,
                "money": 4000.0
            }
        ]
    }
}
```


## （二）子系统间调用

（1）行程规划子系统向开销管理子系统请求某一事件的开销列表

**请求方式：**get

**请求路径：**/expense/event

**参数示例：**eveID=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": [
        {
            "id": 2,
            "eveID": 1,
            "type": 3,
            "time": "2024-11-21T12:30:00",
            "money": 20.0,
            "name": "buy some tickets"
        },
        {
            "id": 3,
            "eveID": 1,
            "type": 1,
            "time": "2024-11-21T12:50:00",
            "money": 50.0,
            "name": "taking a taxi"
        }
    ]
}
```


（2）行程规划子系统向开销管理子系统请求某一事件的预算列表

**请求方式：**get

**请求路径：**/budget/event

**参数示例：**eveID=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": [
        {
            "id": 2,
            "eveID": 1,
            "type": 1,
```

```
            "money": 100.0,
            "name": "Test Budget"
        },
        {
            "id": 3,
            "eveID": 1,
            "type": 2,
            "money": 40.0,
            "name": "breakfast"
        }
    ]
}
```


（3）行程规划子系统向开销管理子系统请求删除某一事件的所有开销记录

**请求方式：**delete

**请求路径：**/expense/deletebyeveid

**参数示例：**eveID=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": true
}
```


（4）行程规划子系统向开销管理子系统请求删除某一事件的所有预算记录

**请求方式：**delete

**请求路径：**/budget/deletebyeveid

**参数示例：**eveID=1

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": true
}
```


（5）开销管理子系统向行程规划子系统请求某一用户对应的事件 **id**

**请求方式：**get

**请求路径：**/event/getall

**参数示例：**userID=681295877

**返回：**


```
{
```

```
    "code": 1,
    "msg": "success",
    "data": [
        6,
        7
    ]
}
```


**（6）开销管理子系统向行程规划子系统请求某些行程对应的事件 id**

**请求方式：**get

**请求路径：**/event/getbyitiIDs

**参数示例：**itiIDs=1,2

**返回：**


```
{
    "code": 1,
    "msg": "success",
    "data": [
        6,
        7
    ]
}
```


## （三）子系统内部调用

- **用户管理子系统**

User 类调用 WechatAuthentication 类的 getOpenId 函数，参数为 code，返回值为 openId。

- **天气查询子系统**

Weather 类调用 WeatherData 类的 fetchData 函数，参数为 time 和 location，返回值为 maxTemperature、minTemperature、description、wind、airQuality。

WeatherReminder 类调用 Weather 类的 getWeather 函数，参数为 time 和 location，返回值为 maxTemperature、minTemperature、description、wind、airQuality。

- **行程规划子系统**

Itinerary 类调用 Event 类的 getEventList 函数，参数为当前行程(this)，返回值为事件列表(List<Event>)。

- **开销管理子系统**

Report 类调用 Budget 类的 getList 函数，参数为不同的筛选条件，返回值为预算列表（List<Budget>）。

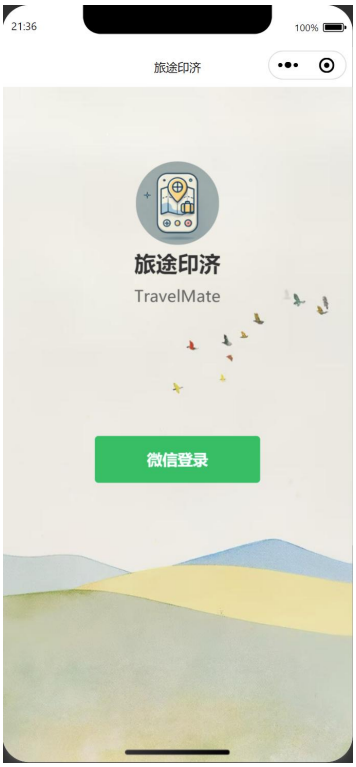
Report 类调用 Expense 类的 getList 函数，参数为不同的筛选条件，返回值为开销列表(List<Expense>)。

## 2.4 界面设计

### 2.4.1 导航和总体结构

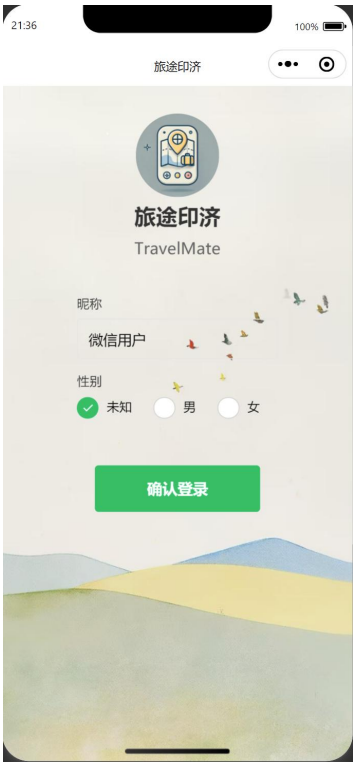
1．登录页面

点击“微信登录”按钮，首次登录用户进入填写用户信息页面，非首次登录用户直接进入小程序主页面（行程页面）。




2．填写用户信息页面

若用户同意授权微信昵称及性别，系统自动补充用户微信昵称及性别，用户可在此基础上进行修改；反之，用户直接填写昵称及性别，二者均不可为空。



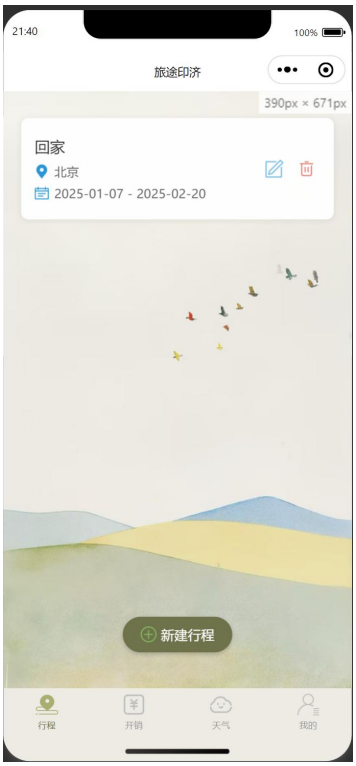

3．导航栏

系统使用全局导航栏，通过导航栏，用户可以快速访问其他功能页面（“行程”、“开销”、“天气”、“我的”）。


### 2.4.2 功能模块页面设计

1．行程页面

页面显示用户的行程列表，点击某行程的编辑按钮进入编辑行程页面，点击某行程的删除按钮删除该行程，点击“新建行程”按钮进入新建行程页面，点击行程框的其他位置进入该行程的事件页面，左右翻页查看更多行程。




2．编辑/添加行程页面

用户可以在此页面填写行程名称，选择开始日期、结束日期（结束日期必须>=开始日期），目的地，任何信息都不能为空，点击“保存”按钮保存行程。

3．事件页面

本页面显示某行程的所有事件以及该行程的基本信息，点击“智能推荐”按钮进入该行程的智能推荐页面，点击“添加事件”按钮进入添加事件页面，点击某事件的“预算开销”按钮进入某事件的预算开销页面，点击某事件的编辑按钮进入该事件的编辑页面，点击某事件的删除按钮可以删除本事件。


4．智能推荐页面

本页面可以显示本行程的智能推荐。


5．添加/编辑事件页面

用户可以填写事件的名称、选择开始时间和结束时间（结束时间必须>=开始时间），填写地点，选择类别，填写描述（除描述外都必须为非空）。点击“保存”按钮保存本事件。


6．事件的预算开销页面

预算使用情况框显示预算与支出的比较，总支出用红色显示，预算用绿色显示，若总支出超过预算会提示总支出超过预算的数额。点击“编辑预算”弹出编辑预算弹窗（当没有设置预算时页面显示“添加预算”按钮，点击弹出添加预算弹窗）。页面下方显示本事件的开销列表，点击“添加开销”按钮进入添加开销页面，点击某开销的编辑按钮进入编辑开销页面，点击删除按钮可以删除本开销。


7．添加/编辑开销页面

用户可以输入开销的名称，金额，选择时间、类别（均不能为空），点击“保存”按钮保存本开销。

8．开销页面-按时间查看报表

用户可以选择筛选的开始时间、结束时间，以查询特定时间段内的总支出及支出详细信息。


9．开销页面-按类别查看报表

用户可以选择筛选的类别，以查询特定类别的总支出及支出详细信息。


10．开销页面-按行程查看报表

用户可以选择筛选的行程，以查询特定行程的预算、总支出及支出详细信息。

11．天气页面

本页面上方为天气查询部分，用户首次进入本页面，系统会请求地理位置权限，若同意，则本页面显示用户所在位置当前日期的天气情况；用户可以在最上方的搜索框输入地点、选择时间（只能在当天及之后 3 天以内的日期），点击“搜索”按钮显示用户搜索的地点日期的天气情况。下方显示用户的天气提醒列表，显示用户的所有天气提醒，点击“添加提醒”按钮进入添加提醒页面，点击某提醒的编辑按钮进入编辑提醒按钮，点击删除按钮可以删除该提醒。


12．添加/编辑提醒页面

用户可以选择提醒时间（发送提醒的时间，只能在当前时间及以后的时间）、选择查询日期（要查询的天气信息的日期，只能在提醒时间的当天及以后三天）、填写地点（均不能为空），点击“保存”按钮保存本提醒。


13．个人信息页面

本页面用户的个人信息，点击“编辑资料”按钮，进入编辑个人信息页面。

14．编辑个人信息页面

用户可以填写昵称，选择性别（均不能为空），点击“保存按钮”可以保存个人信息。


### 2.4.3 操作细节和弹窗设计

1．添加/编辑行程页面

未填写和选择时，本页面的选择/输入框中会提示填写/选择的信息，并且在有内容未空时，“保存”按钮为无法点击的状态。


2．删除行程弹窗

点击某行程的删除按钮时，会出现确认删除的弹窗。


3．添加/编辑事件页面

未填写和选择时，本页面的选择/输入框中会提示填写/选择的信息，并且在除描述外有内容未空时，“保存”按钮为无法点击的状态。


4．删除事件弹窗

点击某事件的删除按钮时，会出现确认删除的弹窗。


5．添加预算弹窗

点击“添加预算”按钮时，会出现添加预算的弹窗，以及输入预算金额的提示。

6．编辑预算弹窗

点击“编辑预算”按钮时，会出现编辑预算的弹窗，以及输入预算金额的提示。


7．添加/编辑开销页面

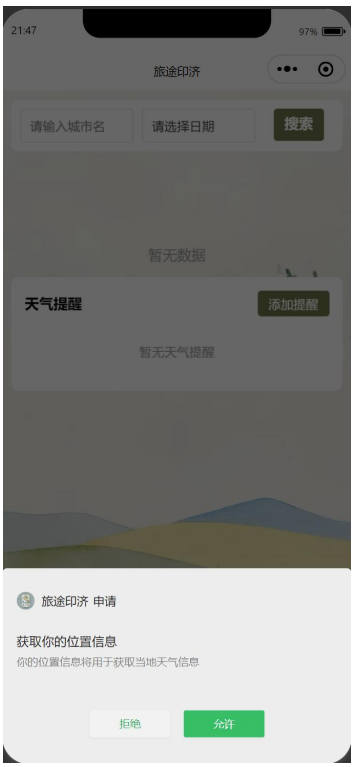
未填写和选择时，本页面的选择/输入框中会提示填写/选择的信息，并且在有内容未空时，“保存”按钮为无法点击的状态。


8．删除开销弹窗

点击某开销的删除按钮时，会出现确认删除的弹窗。


9．请求地理位置信息的弹窗

当用户首次进入天气页面时，会出现请求获取用户的位置信息的弹窗。




10．请求订阅一次消息通知的弹窗

点击“添加提醒”按钮时，会出现请求订阅一次天气提醒消息通知的弹窗。


11．添加/编辑天气提醒页面

未填写和选择时，本页面的选择/输入框中会提示填写/选择的信息，并且在有内容未空时，“保存”按钮为无法点击的状态。

12．删除天气提醒弹窗

点击某天气提醒的删除按钮时，会出现确认删除的弹窗。


## 2.5 外部接口设计

在我们的旅游规划小程序中，系统需要与多个外部接口交互来提供丰富的功能支持，包括旅行推荐、智能分析、用户认证和地图服务等功能。以下是外部接口的详细设计：

### 2.5.1 旅行网站接口

1．**用途**：用于获取旅行相关平台上与目的地相关的旅行信息，如热门景点、旅游攻略、推荐路线和住宿信息。

2．**接口协议**：HTTP RESTful API。

3．**请求方式**：GET 请求。

4．**接口功能**：根据用户输入的目的地关键字、预算、出行事件信息获取相关的旅游指南数据。返回相关攻略，包括景点列表、景点详细信息（名称、地址、图片、简介）和用户评价。

5．**数据格式**：JSON。

6．**接口要求**：需获取旅行网站提供的开发者 API 密钥。

示例如下：

```
{
  "destination": "Shanghai",
  "attractions": [
    {
      "name": "The Bund",
      "address": "Zhongshan East 1st Rd, Shanghai",
      "image": "https://example.com/bund.jpg",
      "description": "A famous waterfront area in central
Shanghai."
    },
    ...
  ]}
```

### 2.5.2 大语言模型接口

1．**用途**：提供智能化的旅行推荐、行程规划建议。

2．**接口协议**：HTTP RESTful API 或 WebSocket。

3．**请求方式**：POST 请求。

4．**接口功能**：根据用需求，包括行程预算、时间段、目的地信息以及前面获得的旅行网站数据生成推荐的旅行计划。提供完整的攻略建议，包括具体日程安排、花销数目、热门景点相关信息、推荐的用餐地点等等。

5．**数据格式**：JSON。

6．**接口要求**：需对接 LLM 提供的 API 密钥（如 OpenAI 或其他供应商）。

示例如下：

```
{
  "destination": "Tokyo",
  "days": 5,
  "budget": 3000,
  "preferences": ["cultural sites", "food"]}
{
  "itinerary": [
    {
      "day": 1,
      "activities": [
        "Visit Senso-ji Temple",
        "Try sushi at Tsukiji Market"
      ]
    },
    ...
  ]}
```

### 2.5.3 微信认证接口

1．**用途**：用于用户身份验证，支持微信账户登录，帮助系统获取用户的基本信息（如昵称、头像）。

2．**接口协议**：基于微信小程序提供的 wx.getUserProfile 接口协议。

3．**请求方式**：调用 wx.getUserProfile 方法，用户需主动授权。

4．**接口功能**：提供弹窗提示用户授权，授权后获取用户的基本信息（昵称、头像、性别）。返回用户的微信标识（如 openid）及必要的用户信息。无需通过服务端获取授权码（code），而是直接通过微信客户端调用接口完成。

5．**数据格式**：JSON。

6．**接口要求**：需注册微信开放平台并获取 AppID 和 AppSecret。

示例如下：

```
{
  "openid": "o6_bmjrPTlm6_2sgVt7hMZOPfL2M",
"nickname": "Xie Qing",
  "avatarUrl":
"https://wx.qlogo.cn/mmopen/vi_32/Q0j4TwGTfT...",
  "gender": 1
}
```

### 2.5.4 高德地图接口（天气接口）

1．**用途**：用于获取实时天气信息和地图服务，提供目的地天气提醒功能。

2．**接口协议**：HTTP RESTful API。

3．**请求方式**：GET 请求。

4．**接口功能**：根据用户当前位置（或指定位置）获取实时天气数据，包括温度、空气质量、天气状况等。提供目的地天气预测功能。

5．**数据格式**：JSON。

6．**接口要求**：需向高德开放平台申请开发者 API 密钥。

示例如下：

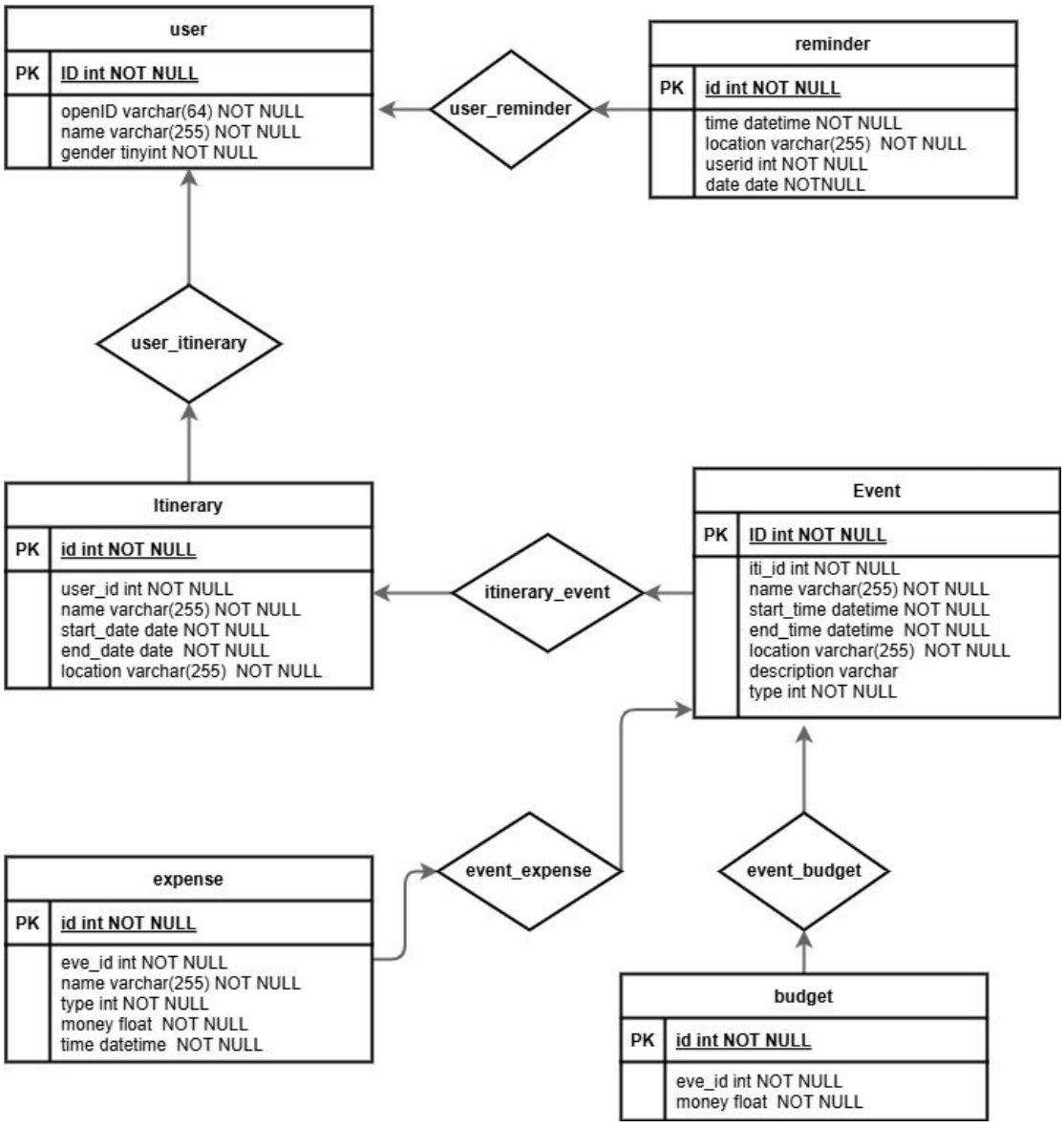
```
URL:
https://restapi.amap.com/v3/weather/weatherInfo?key=YOUR_
API_KEY&city=Shanghai

{
  "status": "1",
  "count": "1",
  "info": "OK",
  "infocode": "10000",
  "lives": [
```

```
    {
      "city": "Shanghai",
      "weather": "Cloudy",
      "temperature": "25",
      "winddirection": "East",
      "windpower": "3",
      "humidity": "78"
    }
  ]}
```

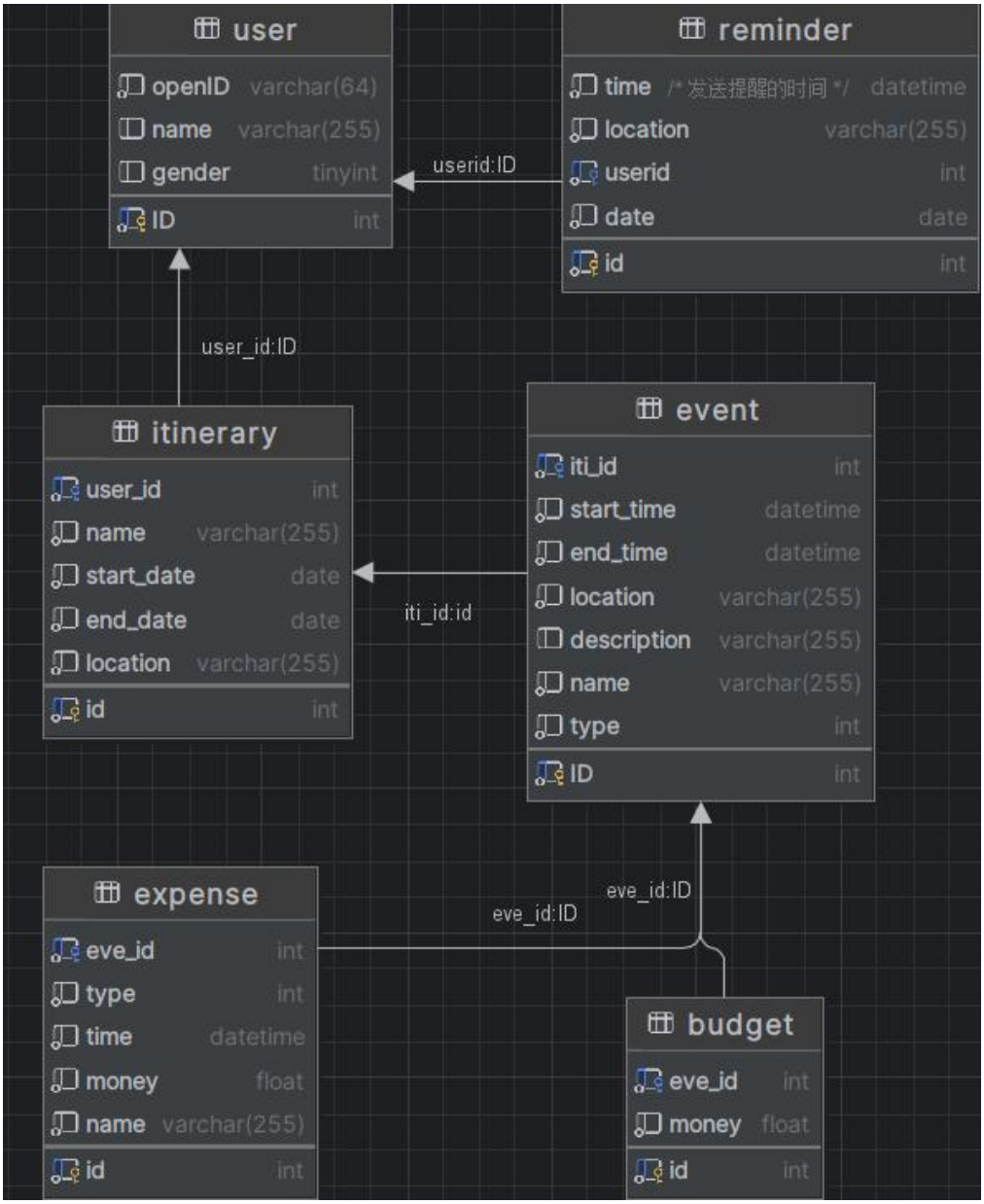

## 2.6 数据库设计

### 2.6.1 数据库逻辑设计

数据库 ER 图：


user
PK
ID int NOT NULL
openID varchar(64) NOT NULL
name varchar(255) NOT NULL
gender tinyint NOT NULL
user_reminder
reminder
PK
id int NOT NULL
time datetime NOT NULL
location varchar(255) NOT NULL
userid int NOT NULL
date date NOTNULL
user_itinerary
Itinerary
PK
id int NOT NULL
user_id int NOT NULL
name varchar(255) NOT NULL
start_date date NOT NULL
end_date date NOT NULL
location varchar(255) NOT NULL
itinerary_event
Event
PK
ID int NOT NULL
iti_id int NOT NULL
name varchar(255) NOT NULL
start_time datetime NOT NULL
end_time datetime NOT NULL
location varchar(255) NOT NULL
description varchar
type int NOT NULL
expense
PK
id int NOT NULL
eve_id int NOT NULL
name varchar(255) NOT NULL
type int NOT NULL
money float NOT NULL
time datetime NOT NULL
event_expense
event_budget
budget
PK
id int NOT NULL
eve_id int NOT NULL
money float NOT NULL

### 2.6.2 数据库实体设计




### 2.6.2.1 User 实体

| 字段 | 描述 |
| --- | --- |
| ID | 主键,用户编号(int) |
| openID | 用户 openID 标识(varchar(64)) |
| name | 用户昵称(varchar(255)) |
| gender | 用户性别(tinyint) |

### 2.6.2.2 Reminder 实体

| 字段 | 描述 |
| --- | --- |
| id | 主键,提醒编号(int) |
| time | 提醒时间(datetime) |
| location | 地点(varchar(255)) |

| | |
|---|---|
| userid | 关联的用户 ID(int) |
| date | 提醒日期(date) |

#### 2.6.2.3 Itinerary 实体

| 字段 | 描述 |
|---|---|
| id | 主键,行程编号(int) |
| user_id | 关联的用户 ID(int) |
| name | 行程名称(varchar(255)) |
| start_date | 开始日期(date) |
| end_date | 结束日期(date) |
| location | 地点(varchar(255)) |

#### 2.6.2.4 Event 实体

| 字段 | 描述 |
|---|---|
| ID | 主键,事件编号(int) |
| iti_id | 关联的行程 ID(int) |
| name | 事件名称(varchar(255)) |
| start_time | 开始时间(datetime) |
| end_time | 结束时间(datetime) |
| location | 地点(varchar(255)) |
| description | 事件描述(varchar(255)) |
| type | 事件类型(int) |

#### 2.6.2.5 Budget 实体

| 字段 | 描述 |
|---|---|
| id | 主键,预算编号(int) |
| eve_id | 关联的事件 ID(int) |
| money | 预算金额(float) |

#### 2.6.2.6 Expense 实体

| 字段 | 描述 |
|---|---|
| id | 主键,支出编号(int) |
| eve_id | 关联的事件 ID(int) |
| name | 支出名称(varchar(255)) |
| type | 支出类型(int) |
| money | 支出金额(float) |
| time | 支出时间(float) |

### 2.6.3 数据库关系设计

#### 2.6.3.1 reminder&user

| 字段 | 描述 |
| --- | --- |
| reminder id | 提醒 id |
| user ID | 事件 id |

#### 2.6.3.2 itinerary&user

| 字段 | 描述 |
| --- | --- |
| Itinerary id | 行程 id |
| user ID | 事件 id |

#### 2.6.3.3 event&itinerary

| 字段 | 描述 |
| --- | --- |
| event ID | 事件 id |
| itinerary id | 行程 id |

#### 2.6.3.4 expense&event

| 字段 | 描述 |
| --- | --- |
| expense id | 支出 id |
| event ID | 事件 id |

#### 2.6.3.5 budget&user

| 字段 | 描述 |
| --- | --- |
| budget id | 预算 id |
| event ID | 事件 id |

#### 2.5.3.6 关系分析

| 联系集名称 | 关系 | 说明 |
| --- | --- | --- |
| **reminder&user** | 多对一 | 一个 user 可以有多个 reminder，一个 reminder 只能提醒一个 user |
| **itinerary&user** | 多对一 | 一个 user 可以有多个 itinerary，一个 itinerary 只能属于一个 user |
| **event&itinerary** | 多对一 | 一个 itinerary 可以有多个 event，一个 event 只能属于一个 itinerary |
| **budget&user** | 多对一 | 一个 user 可以有多个 budget，一个 budget 只能属于一个 user |
| **expense&event** | 多对一 | 一个 user 可以有多个 expense，一个 expense 只能属于一个 user |

### 2.6.4 数据库表设计

#### 2.6.4.1 user 表

| Field | Description | Type | Nullable |
|---|---|---|---|
| ID | 主键, 用户编号 | int | NO |
| openID | 用户 openID 标识 | varchar(64) | NO |
| name | 用户昵称 | varchar(255) | NO |
| gender | 用户性别 | tinyint | NO |

#### 2.6.4.2 reminder 表

| Field | Description | Type | Nullable |
|---|---|---|---|
| id | 主键, 提醒编号 | int | NO |
| time | 提醒时间 | datetime | NO |
| location | 地点 | varchar(255) | NO |
| userid | 关联的用户 ID | int | NO |
| date | 提醒日期 | date | NO |

#### 2.6.4.3 itinerary 表

| Field | Description | Type | Nullable |
|---|---|---|---|
| id | 主键, 行程编号 | int | NO |
| user_id | 关联的用户 ID | int | NO |
| name | 行程名称 | varchar(255) | NO |
| start_date | 开始日期 | date | NO |
| end_date | 结束日期 | date | NO |
| location | 地点 | varchar(255) | NO |

#### 2.6.4.4 Event 表

| Field | Description | Type | Nullable |
|---|---|---|---|
| ID | 主键，事件编号 | int | NO |
| iti_id | 关联的行程 ID | int | NO |
| name | 事件名称 | varchar(255) | NO |
| start_time | 开始时间 | datetime | NO |
| end_time | 结束时间 | datetime | NO |
| location | 地点 | varchar(255) | NO |
| description | 事件描述 | varchar(255) | YES |
| type | 事件类型 | int | NO |

#### 2.6.4.5 Budget 表

| Field | Description | Type | Nullable |
|---|---|---|---|
| id | 主键，预算编号 | int | NO |
| eve_id | 关联的事件 ID | int | NO |
| money | 预算金额 | float | NO |

#### 2.6.4.6 Expense 表

| Field | Description | Type | Nullable |
|---|---|---|---|
| Id | 主键，支出编号 | int | NO |
| eve_id | 关联的事件 ID | int | NO |
| name | 支出名称 | varchar(255) | NO |
| type | 支出类型 | int | NO |
| money | 支出金额 | float | NO |
| time | 支出时间 | datetime | NO |

## 2.7 系统出错处理设计

### 2.7.1 出错信息

**出错有以下分类：**

1．用户输入错误：用户输入非法字符或格式错误，例如日期格式、金额格式、空字段等。

2．数据访问错误：数据库连接失败、查询无结果或数据读写失败。

3．接口调用错误：第三方接口（如微信认证接口、天气接口等）调用失败或超时。

4．权限错误：用户未登录或尝试访问无权限的资源。

5．系统错误：系统内部异常或未知错误，如程序运行时错误、内存溢出等。

**出错信息反馈：**

1．用户输入错误：提示“输入格式有误，请检查后重试”。

2．数据访问错误：提示“数据加载失败，请稍后再试”。

3．接口调用错误：提示“外部服务暂时不可用，请稍后重试”。

4．权限错误：提示“未登录，请先登录系统”。

5．系统错误：提示“系统发生未知错误，请联系管理员”。

**错误日志：**

系统需在后台记录所有错误信息，包括时间戳、错误类型、发生模块和详细堆栈信息，以便运维人员进行排查。

### 2.7.2 补救措施

**（1）用户输入错误**

1．提供输入提示：在输入框附近提供合法输入格式的提示说明（如日期格式为 YYYY-MM-DD）。

2．实时校验：在用户输入时实时检查并高亮错误字段。

3．错误重试：引导用户重新输入。

**（2）数据访问错误**

1．短期补救：提供备用数据库连接池或缓存机制以减轻数据库压力。

2．用户体验优化：如果查询无结果，返回空白状态提示用户“未找到相关数据”。

3．后台处理：自动重试数据库连接一定次数，如果仍失败，记录日志并报警。

**（3）接口调用错误**

1．短期补救：针对超时或调用失败，系统可设置自动重试（如 3 次）。

2．备用机制：若主要接口失败，可尝试调用备用接口或提供本地缓存数据（如天气接口失败时返回最近一次请求的结果）。

3．用户提示：在界面提示用户接口不可用，并引导用户稍后重试。

**（4）权限错误**

1．提示用户登录：未登录用户需跳转至登录界面完成认证。

2．登录状态检查：若会话超时或过期，提示用户重新登录。

**（5）系统错误**

1．异常捕获：所有运行时错误应被统一捕获并优雅退出，避免直接崩溃。

2．降级服务：系统部分功能出现问题时，其他功能应正常运行（例如接口调用失败时仍能使用离线模式）。

3．错误报告：实时发送错误报告至管理员邮箱或监控平台。

**2.7.3 系统维护设计**

**（1）日志管理**

1．日志分类：分为访问日志、错误日志、调试日志等。

2．日志存储：日志存储在服务器本地或云端，按日期分文件管理，支持自

动清理旧日志。

3．日志分析：提供工具对日志进行分析，生成报表（如错误频率、热点模块等）。

**（2）系统监控**

1．实时监控：集成监控工具（如 Prometheus 或 Grafana）监控系统运行状态，包括 CPU、内存、接口响应时间等。

2．报警机制：当系统性能指标异常或发生重大错误时，自动通过邮件、短信或微信发送告警信息。

**（3）系统备份**

1．数据备份：定期对数据库进行备份，保留至少最近 30 天的备份数据。

2．配置备份：保存所有重要配置文件的备份，以便快速恢复。

3．自动恢复：提供一键恢复机制，在数据或系统出现问题时快速还原。

**（4）更新维护**

1．定期更新：定期对系统进行升级和安全补丁的更新。

2．灰度发布：更新功能时采用灰度发布机制，确保系统平稳运行。

3．回滚机制：更新失败时，支持快速回滚到上一个稳定版本。

**（5）文档支持**

提供详细的开发文档和操作手册，便于后续开发和运维。

定期更新文档，保证与实际系统保持一致。