**详细设计说明书**

**《智慧食堂系统》**

**编写日期：2024.6.5**

**项目组：**

|  |  |  |
| --- | --- | --- |
| **学号** | **姓名** | **角色** |
|  |  | **组长** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**修改日志**

|  |  |  |
| --- | --- | --- |
| 修改者 | 修改日期 | 备注说明 |
| 章洋 | 2024.6.18 | 完成引言和概述 |
| 章洋 | 2024.6.19 | 完成模块1的设计 |
| 章洋 | 2024.6.20 | 完成模块2的设计 |
| 章洋 | 2024.6.21 | 完成模块3 的设计 |
| 章洋 | 2024.6.22 | 完成模块4的设计 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

目录

[1. 引言 3](#_Toc170345108)

[1.1 编写目的 3](#_Toc170345109)

[1.2 项目背景 3](#_Toc170345110)

[1.3 定义 3](#_Toc170345111)

[**1.3.1** 引用名词1 名词1的解释 3](#_Toc170345112)

[1.4 参考资料 3](#_Toc170345113)

[2 软件结构概述 3](#_Toc170345114)

[3 模块设计 3](#_Toc170345115)

[3.1 模块1 菜品查看模块 3](#_Toc170345116)

[**3.1.1** 功能描述 3](#_Toc170345117)

[**3.1.2** 接口描述 3](#_Toc170345118)

[**3.1.3** 内部元素结构 4](#_Toc170345119)

[**3.1.4** 人机界面设计 4](#_Toc170345120)

[**3.1.5** 子程序设计 5](#_Toc170345121)

[**3.1.6** 模块测试设计 7](#_Toc170345122)

[3.2 模块2 订单处理模块 7](#_Toc170345123)

[**3.2.1** 功能描述 7](#_Toc170345124)

[**3.2.2** 接口描述 7](#_Toc170345125)

[**3.2.3** 内部元素结构 8](#_Toc170345126)

[**3.2.4** 人机界面设计 8](#_Toc170345127)

[**3.2.5** 子程序设计 9](#_Toc170345128)

[**3.2.6** 模块测试设计 12](#_Toc170345129)

[3.3 模块3 营养分析模块 12](#_Toc170345130)

[**3.3.1** 功能描述 12](#_Toc170345131)

[**3.3.2** 接口描述 12](#_Toc170345132)

[**3.3.3** 内部元素结构 13](#_Toc170345133)

[**3.3.4** 人机界面设计 13](#_Toc170345134)

[**3.3.5** 子程序设计 13](#_Toc170345135)

[**3.3.6** 模块测试设计 15](#_Toc170345136)

[3.4 模块4 菜品制备规划模块 16](#_Toc170345137)

[**3.4.1** 功能描述 16](#_Toc170345138)

[**3.4.2** 接口描述 16](#_Toc170345139)

[**3.4.3** 内部元素结构 16](#_Toc170345140)

[**3.4.4** 人机界面设计 17](#_Toc170345141)

[**3.4.5** 子程序设计 17](#_Toc170345142)

[**3.4.6** 模块测试设计 18](#_Toc170345143)

# 引言

## 编写目的

本详细计划说明书（DSD）在概要设计的基础上，进一步的细化系统结构，展示了软件的结构，数据结构设计等，介绍了系统各个模块是如何实现的，包括涉及到的逻辑流程等，为下一步系统的实现和测试做准备。

## 项目背景

a.软件名称：“柚有好饭”食堂管理系统

b.用户：食堂需要面对食物有不同需求的大学生

c.传统食堂运营缺乏一定便携性

## 定义

### 引用名词1：功能描述

解释：准确地描述本模块的功能。

### 引用名词2：接口描述

解释：准确地描述本模块的接口规范。

### 引用名词3：内部元素结构

准确地描述每个模块包含的数据、子程序等。

### 引用名词4：人机界面设计

解释：用图形展示该模块的用户界面。

### 引用名词5：子程序设计

解释：采用PDL语言、程序流程图等过程设计模型设计每个子程序的算法和逻辑。

### 引用名词6：模块测试设计

解释：给出本模块的主要测试要求

## 参考资料

 项目选题计划书​​

 用户需求说明书

 技术可行性报告

# 软件结构概述

1.菜品查看：提供一个用户界面，让用户浏览当前的菜品列表。该模块应支持搜索、排序和筛选功能，以方便用户快速找到感兴趣的菜品。

2.菜品预约：允许用户选择菜品并进行预约。该模块需要处理用户选择、预约时间和数量的记录，以及预约确认和取消功能。

3.营养分析：根据用户选择的菜品，提供营养信息分析。该模块应能够计算并展示菜品的营养成分，如卡路里、蛋白质、脂肪等，并可能提供个性化的营养建议。

4.食堂菜品制备规划：为食堂管理者提供菜品制备的规划工具。该模块应包括菜品的采购、库存管理和分析菜品制备数据等功能，以优化食堂的运营效率。

# 模块设计

## 模块1 菜品查看模块

### 功能描述

对于用户，用户可以登录系统，在系统中查看今日的食堂菜品，菜品介绍及评价。

### 接口描述

菜品管理模块提供了一套完整的API接口，用于实现对菜品信息的添加、修改、删除和查询操作。这些接口允许系统用户和开发者高效地管理菜品数据。

接口列表

1.getDishById(dishId)

功能描述：根据提供的菜品ID，获取该菜品的详细信息。

请求方式：GET

请求参数：

(1).dishId：菜品的唯一标识符，类型为字符串或数字。

响应数据：返回请求的菜品的详细信息，如果菜品不存在，则返回错误信息。

2.addDish(dishInfo)

功能描述：添加一个新的菜品到系统中。

请求方式：POST

请求参数：

(1).dishInfo：一个包含菜品信息的对象，通常包括菜品名称、描述、价格、图片URL等。

响应数据：添加成功后返回菜品的详细信息，如果添加失败，则返回错误信息。

3.updateDish(dishId, dishInfo)

功能描述：根据菜品ID更新菜品的信息。

请求方式：PUT

请求参数：

(1).dishId：要更新的菜品的唯一标识符。

(2).dishInfo：包含更新信息的对象，可能包含菜品名称、描述、价格等字段。

响应数据：更新成功后返回更新后的菜品信息，如果更新失败，则返回错误信息。

4.deleteDish(dishId)

功能描述：根据菜品ID从系统中删除一个菜品。

请求方式：DELETE

请求参数：

(1).dishId：要删除的菜品的唯一标识符。

响应数据：删除成功返回成功消息，如果菜品不存在或删除失败，则返回错误信息。

### 内部元素结构

1. 菜品预约模块应有的数据包括：用户信息、菜品信息（名称、数量、价格、预订时间）、评价信息（评分、评价内容）。

2. 本模块应该有以下子程序：

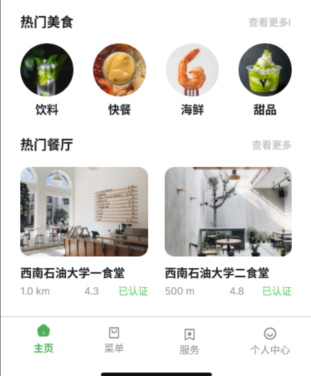
1.用户登录子程序：验证用户信息，以确保能够进行后续预约操作。

2.查看菜品子程序：用户登录后可以查看今日的食堂菜品信息。

3.查看评价子程序：用户可以查看其他用户对菜品的评价

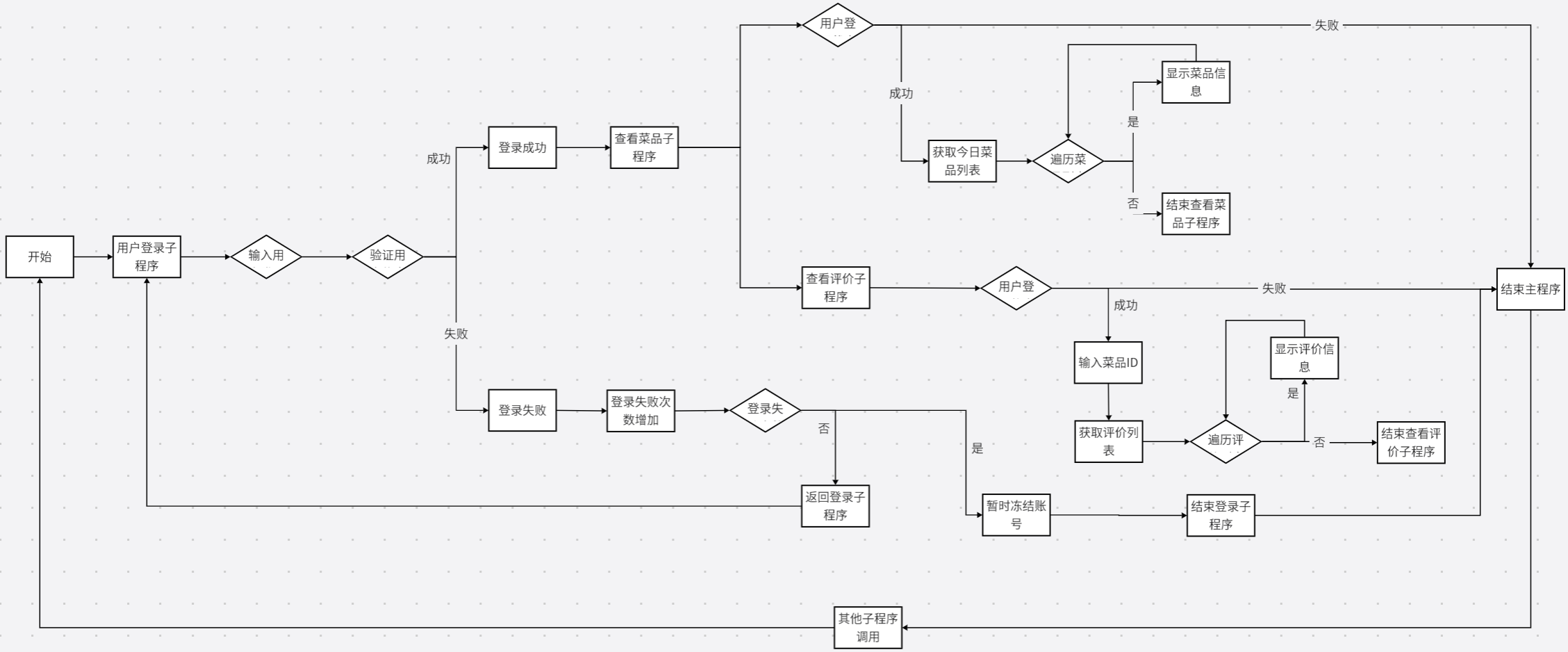
### 人机界面设计

（用图形展示该模块的用户界面。）



### 子程序设计

（采用PDL语言、程序流程图等过程设计模型设计每个子程序的算法和逻辑。）



定义数据结构

数据结构 用户信息 {

用户名

密码

}

数据结构 菜品信息 {

菜品名称

数量

价格

预订时间

}

数据结构 评价信息 {

评分

评价内容

评价用户

}

1. 用户登录子程序

Begin 用户登录子程序

输入 用户登录信息

如果 用户信息验证成功 THEN

登录成功

ELSE

登录失败

登录失败次数 = 登录失败次数 + 1

IF(登录失败次数 > 5) THEN

暂时冻结账号

ENDIF

ENDIF

End 用户登录子程序

2. 查看菜品子程序

Begin 查看菜品子程序

用户登录状态 = 用户登录子程序()

Utile 用户登录状态 THEN

获取今日菜品列表

对于 每个菜品 in 菜品列表 DO

显示 菜品信息

ENDFOR

ENDIF

End 查看菜品子程序

3. 查看评价子程序

Begin 查看评价子程序

用户登录状态 = 用户登录子程序()

Utile 用户登录状态 THEN

输入 菜品ID

获取菜品ID对应的评价列表

对于 每个评价 in 评价列表 DO

显示 评价信息

ENDFOR

ENDIF

End 查看评价子程序

主程序

Begin 主程序

调用 查看菜品子程序

调用 查看评价子程序

// 其他子程序调用，例如预订菜品、提交评价等

End 主程序

### 模块测试设计

对于菜品管理模块，测试要求应该全面覆盖所有功能点，确保模块的稳定性、可靠性和性能。应有以下几点测试要求：

1.功能测试：

验证每个接口是否按预期工作，包括添加、修改、删除和查询菜品。

2.参数验证测试：

检查接口对于无效、边界和异常输入参数的处理能力。

3.数据一致性测试：

确保在数据更新或删除后，数据库中的数据与预期一致。

4.并发测试：

测试在高并发情况下，接口的性能和稳定性。

5.异常和错误处理测试：

测试系统对于异常情况的处理，包括错误代码和错误消息的正确返回。

6.性能测试：

评估接口的响应时间和处理能力，确保满足性能要求。

## 模块2 订单处理模块

### 功能描述

用户在登录后，可以在系统中提前预约菜品，并得到相应的取餐号。

### 接口描述

订单处理模块提供了一组API接口，用于实现订单的创建、修改、查询和支付功能。这些接口支持学生和食堂管理员对订单进行有效管理。

接口列表：

1.createOrder(studentId, canteenId, dishList, totalPrice)

功能描述：创建一个新的订单。

请求方式：POST

请求参数：

(1).studentId：学生的唯一标识符。

(2).canteenId：食堂的唯一标识符。

(3).dishList：订单中包含的菜品列表，每个菜品应包含菜品ID和数量。

(4).totalPrice：订单的总价格。

响应数据：创建成功后返回新订单的详细信息，包括订单ID等。

2.getOrderById(orderId)

功能描述：根据订单ID获取订单的详细信息。

请求方式：GET

请求参数：

(1).orderId：订单的唯一标识符。

响应数据：返回订单的详细信息，如果订单不存在，则返回错误信息。

3.updateOrder(orderId, orderInfo)

功能描述：更新现有订单的信息。

请求方式：PUT

请求参数：

(1).orderId：要更新的订单的唯一标识符。

(2).orderInfo：包含更新信息的对象，可能包括菜品列表、订单状态等。

响应数据：更新成功后返回更新后的订单信息，如果更新失败，则返回错误信息。

4.deleteOrder(orderId)

功能描述：取消订单。

请求方式：DELETE

请求参数：

(1).orderId：要取消的订单的唯一标识符。

响应数据：取消成功后返回成功消息，如果订单不存在或取消失败，则返回错误信息。

### 内部元素结构

1. 订单处理模块应有的数据包括：用户信息、菜品信息（名称、数量、价格、预订时间）、取餐号信息、单号状态（未出餐、代取餐）

2. 本模块应该有以下子程序：

1.预约菜品子程序：用户登录后，将预约的菜品信息存储在系统中。

2.支付子程序：用于支付预约菜品以生成取餐号。

3.生成取餐号子程序：根据预约信息生成取餐号。

4.更新状态子程序：系统更新单号状态并展示。

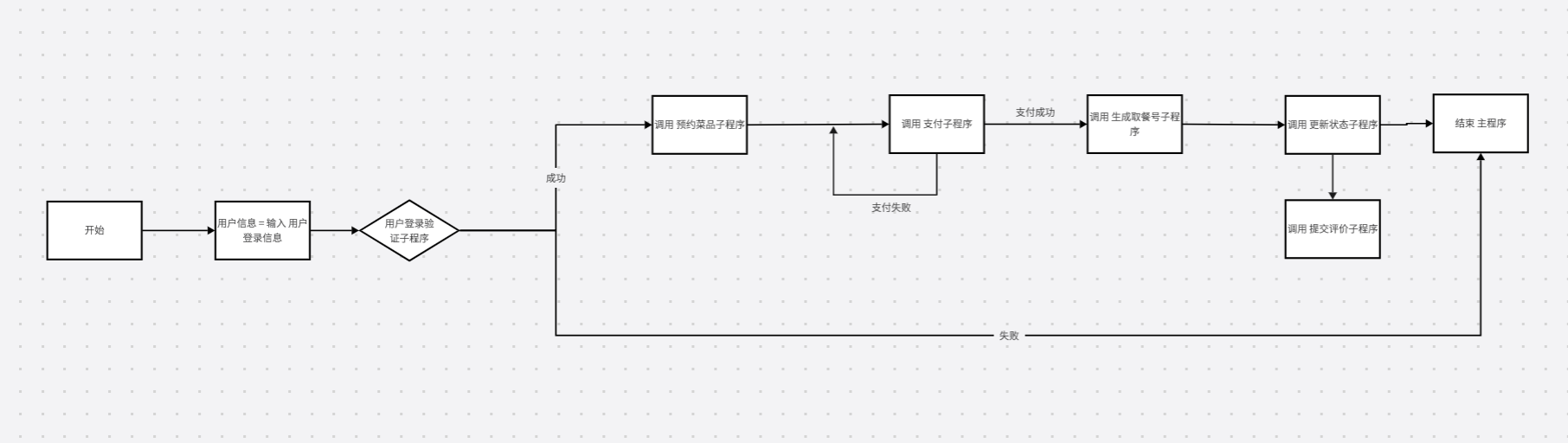
5.提交评价子程序：用户支付后可以对当前菜品进行评价。

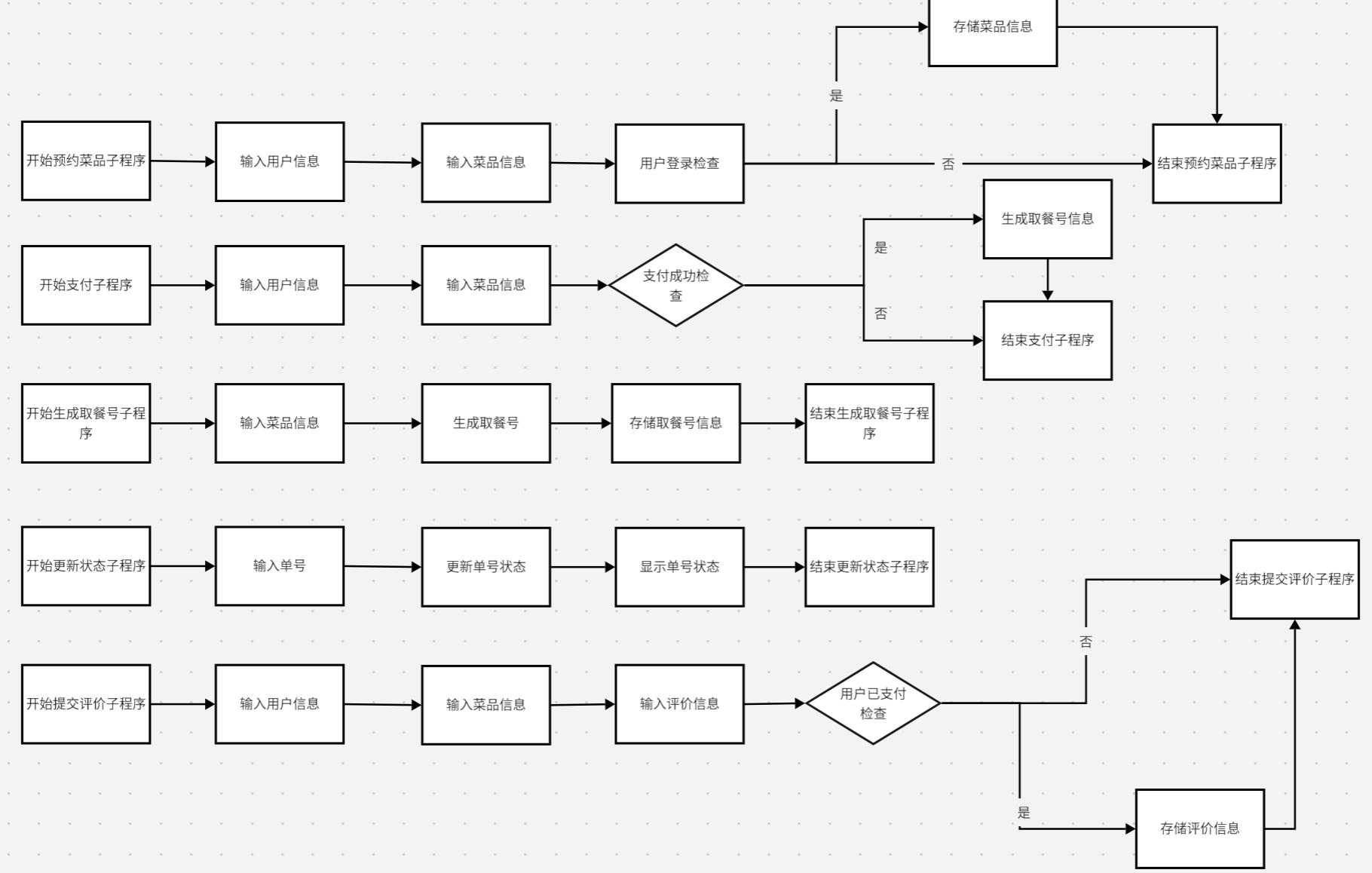
### 人机界面设计

（用图形展示该模块的用户界面。）



### 子程序设计





伪代码：

数据结构 取餐号信息 {

取餐号

}

数据结构 单号状态 {

状态（未出餐，代取餐）

}

1. 预约菜品子程序

Begin 预约菜品子程序

输入 用户信息

输入 菜品信息

如果 用户登录 THEN

存储 菜品信息

ENDIF

End 预约菜品子程序

2. 支付子程序

Begin 支付子程序

输入 用户信息

输入 菜品信息

如果 支付成功 THEN

生成 取餐号信息

ENDIF

End 支付子程序

3. 生成取餐号子程序

Begin 生成取餐号子程序

输入 菜品信息

生成 取餐号

存储 取餐号信息

End 生成取餐号子程序

4. 更新状态子程序

Begin 更新状态子程序

输入 单号

更新 单号状态

显示 单号状态

End 更新状态子程序

5. 提交评价子程序

Begin 提交评价子程序

输入 用户信息

输入 菜品信息

输入 评价信息

如果 用户已支付 THEN

存储 评价信息

ENDIF

End 提交评价子程序

// 主程序

Begin 主程序

用户信息 = 输入 用户登录信息

如果 用户登录验证子程序 THEN

调用 预约菜品子程序

调用 支付子程序

调用 生成取餐号子程序

调用 更新状态子程序

调用 提交评价子程序

ENDIF

End 主程序

### 模块测试设计

对于订单处理模块，以下是几个主要的测试要求：

1. 功能测试：

- 验证每个接口（创建、查询、更新、取消订单）是否按预期工作。确保订单的创建、修改、查询和删除操作能够正确执行，并且返回正确的数据和状态。

2. 参数验证测试：

- 确保接口能够正确处理各种输入参数，包括有效参数、无效参数、边界值和异常值。例如，验证`studentId`、`canteenId`、`orderId`是否接受正确的格式和范围的值，`dishList`是否包含必要的菜品信息，`totalPrice`是否为有效的货币值。

3. 异常和错误处理测试：

- 测试系统对异常情况的响应，比如尝试取消一个不存在的订单或更新一个已经完成的订单。验证系统是否返回合适的错误代码和错误消息。

4. 并发和性能测试：

- 模拟高并发环境下的订单操作，确保系统能够稳定处理大量订单请求，同时验证系统的响应时间和吞吐量是否满足性能要求。

5. 安全性测试：

- 确保订单处理接口实现适当的安全措施，如身份验证、授权和数据加密，以防止未授权访问和数据泄露。

6. 集成和端到端测试：

- 测试订单处理模块与其他相关模块（如用户管理、菜品管理、支付处理等）的集成情况，确保整个订单流程从创建到支付的每个环节都能无缝协同工作。

## 模块3 营养分析模块

### 功能描述

用户在食堂的消费会上传至指定数据库，系统会根据用户在食堂的菜品消费进行营养分析，给出健康建议。

### 接口描述

1.uploadConsumptionData(studentId,dishRecords)

功能描述：将用户的消费记录上传至数据库。

请求方式：POST

请求参数：

(1).studentId：学生的唯一标识符。

(2).dishRecords：用户消费的菜品记录列表，每个记录包括菜品名称、菜品编号、菜品价格、消费时间和消费量。

响应数据：上传成功后返回操作结果，失败则返回错误信息。

2.analyzeNutrition(studentId)

功能描述：对用户在食堂的消费数据进行营养分析。

请求方式：POST

请求参数：

(1).studentId：用户的唯一标识符。

响应数据：返回营养分析结果，包括各类营养素的摄入量、建议摄入量等。

3.generateHealthSuggestions(studentId)

功能描述：根据用户的消费数据和营养分析结果，生成相应的健康建议。

请求方式：POST

请求参数：

(3).studentId：用户的唯一标识符。

响应数据：返回个性化的健康饮食建议

### 内部元素结构

1.营养分析模块应该有的数据包括：菜品名称、菜品编号、菜品价格、消费时间、消费量。

2.本模块应该有以下子程序：

1.数据上传子程序：将用户的消费记录上传至数据库。

2.数据分析子程序：对用户在食堂的消费数据进行营养分析

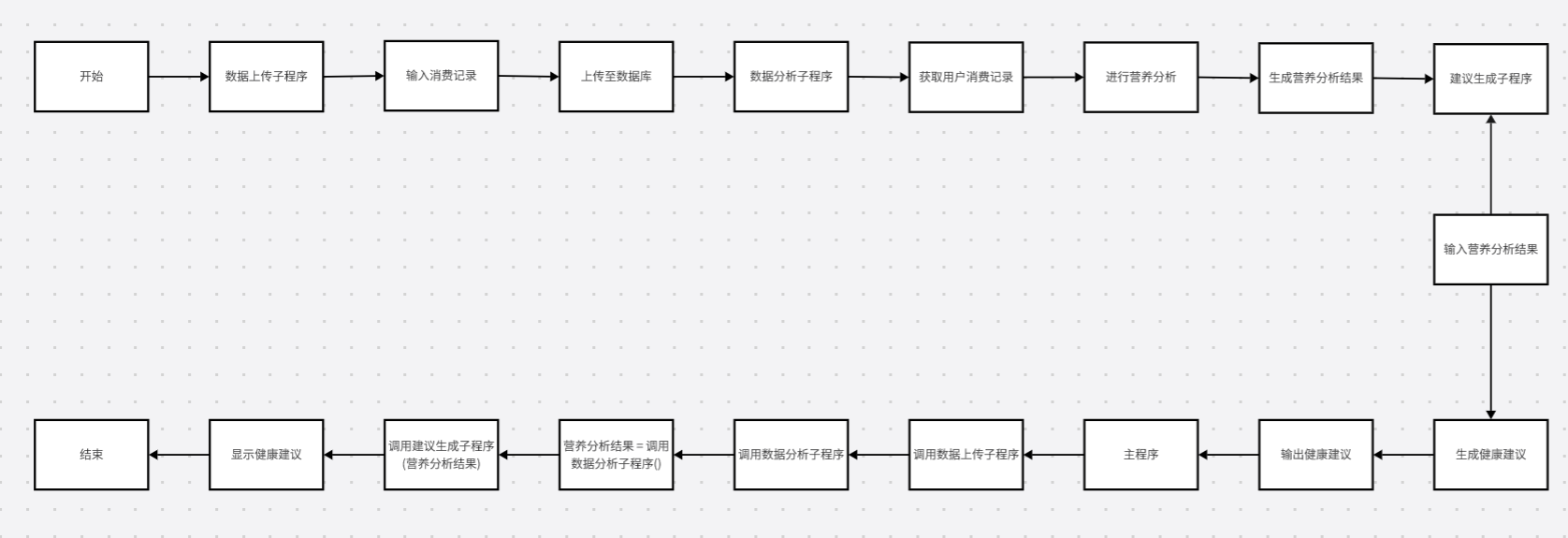
3.建议生成子程序：根据用户的消费数据和营养分析结果，生成相应的健康建议。

### 人机界面设计

（用图形展示该模块的用户界面。）



### 子程序设计



伪代码：

数据结构 消费记录 {

菜品信息

消费时间

消费量

}

数据结构 营养分析结果 {

营养指标

营养评价

}

数据结构 健康建议 {

建议内容

}

1. 数据上传子程序

Begin 数据上传子程序

输入 消费记录

将 消费记录 上传至数据库

End 数据上传子程序

2. 数据分析子程序

Begin 数据分析子程序

获取 用户消费记录

对 消费记录 进行营养分析

生成 营养分析结果

End 数据分析子程序

3. 建议生成子程序

Begin 建议生成子程序

输入 营养分析结果

根据 营养分析结果 生成 健康建议

输出 健康建议

End 建议生成子程序

主程序

Begin 主程序

调用 数据上传子程序

调用 数据分析子程序

营养分析结果 = 调用 数据分析子程序()

调用 建议生成子程序(营养分析结果)

显示 健康建议

End 主程序

### 模块测试设计

对于营养分析模块，以下是几个主要的测试要求：

1. 数据上传子程序测试：

验证数据上传子程序能够正确解析和存储菜品名称、菜品编号、菜品价格、消费时间、消费量等信息。确保数据格式、类型和范围符合预期，并且能够处理异常数据输入。

2. 数据一致性和完整性测试：

确保上传的数据在数据库中保持一致性，没有数据丢失或错误，特别是在处理大量或复杂的数据集时。

3. 数据分析子程序测试：

验证数据分析子程序能够准确计算和分析用户消费的营养成分，包括卡路里、脂肪、蛋白质、碳水化合物等，并确保分析结果的准确性和可靠性。

4. 建议生成子程序测试：

测试建议生成子程序是否能够基于用户的消费数据和营养分析结果提供合理的健康建议，并且建议应符合营养学原则。

5. 异常和错误处理测试：

确保模块能够妥善处理异常情况，例如数据缺失、格式错误或分析失败，并返回清晰的错误信息。

6. 性能测试：

评估模块在处理大量数据时的性能，包括数据上传、分析和建议生成的响应时间和系统稳定性。

## 模块4 菜品制备规划模块

### 功能描述

系统会根据多日内食堂的就餐人数，依据大数据分析，预测未来近几天食堂进餐人数，得到菜品制备数据。

### 接口描述

1.getDiningData（）

功能描述：系统自动收集食堂的每日就餐人数和售出菜品的数据。

请求方式：GET（假设数据收集是自动进行的，不需要额外的请求参数）

请求参数：无（自动数据收集）

响应数据：返回收集到的就餐人数和菜品销售数据。

2.postTrendAnalysis（startDate,endDate）

功能描述：利用收集到的就餐数据预测未来几日食堂的人数趋势。

请求方式：POST

请求参数：

(1).startDate：分析开始日期。

(2).endDate：分析结束日期。

响应数据：返回预测的就餐人数趋势数据。

3.postPrepaPrediction（trendData）

功能描述：根据预测得到的近期进餐人数趋势，系统生成菜品制备数据。

请求方式：POST

请求参数：

(1).trendData：从数据分析子程序得到的就餐人数趋势数据。

响应数据：返回生成的菜品制备数据，包括菜品名称、制备时间、数量等。

### 内部元素结构

1.营养分析模块应该有的数据包括：就餐人数数据（每日就餐人数）、菜品制备数据（菜品名称、制备时间、数量等）。

2.本模块应该有以下子程序：

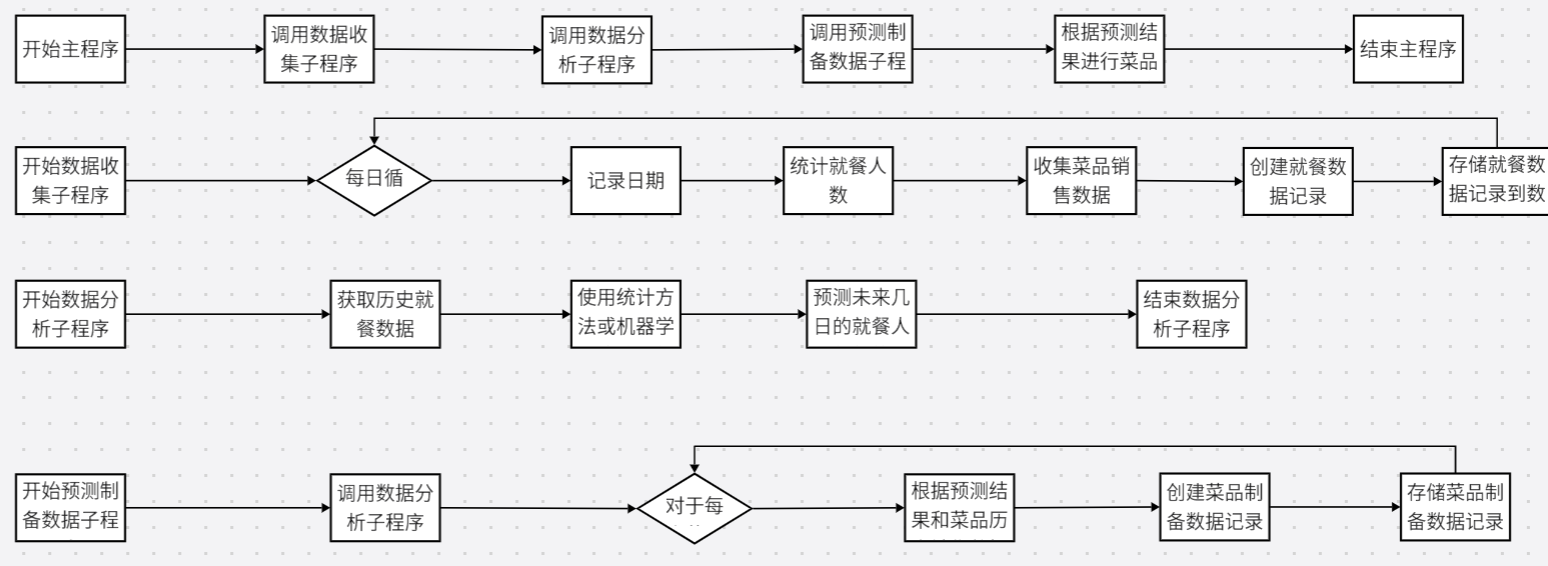
1.数据收集子程序：系统收集食堂的每日就餐人数和售出菜品的数据

2.数据分析子程序：利用收集到的就餐数据预测未来几日食堂的人数趋势

3.预测制备数据子程序：根据预测得到的近期进餐人数趋势，系统生成菜品制备数据。

### 人机界面设计

### 子程序设计



定义数据结构

数据结构 就餐数据 {

日期

就餐人数

菜品销售列表 // 列表中包含每个菜品的销售数量

}

数据结构 菜品制备数据 {

菜品编号

预计制备数量

}

1. 数据收集子程序

Begin 数据收集子程序

对于 每日 DO

记录 日期

统计 就餐人数

收集 菜品销售数据

创建 就餐数据记录

存储 就餐数据记录到数据库

ENDFOR

End 数据收集子程序

2. 数据分析子程序

Begin 数据分析子程序

获取 历史就餐数据

使用 统计方法或机器学习算法 对历史数据 进行分析

预测 未来几日的就餐人数趋势

End 数据分析子程序

3. 预测制备数据子程序

Begin 预测制备数据子程序

预测结果 = 调用 数据分析子程序()

对于 每个菜品 IN 菜品列表 DO

根据 预测结果 和 菜品历史销售数据 计算 预计制备数量

创建 菜品制备数据记录

ENDFOR

存储 菜品制备数据记录

End 预测制备数据子程序

主程序

Begin 主程序

调用 数据收集子程序

调用 数据分析子程序

调用 预测制备数据子程序

// 根据生成的菜品制备数据进行后续的菜品准备和供应工作

End 主程序

### 模块测试设计

对于菜品制备规划模块，以下是几个主要的测试要求：

1.数据收集功能测试：

验证getDiningData接口能够在没有任何输入参数的情况下自动收集并返回正确的就餐人数和菜品销售数据。

2.数据完整性和准确性测试：

确保getDiningData接口收集的数据是完整和准确的，包括就餐人数和菜品销售记录的所有相关细节。

3.趋势分析准确性测试：

对postTrendAnalysis接口进行测试，确保根据提供的startDate和endDate参数，返回的就餐人数趋势预测数据是准确和合理的。

4.菜品制备预测测试：

验证postPreparationPrediction接口能否根据提供的trendData准确生成菜品制备数据，包括菜品名称、制备时间和数量。

5.异常和错误处理测试：

测试接口对无效参数、缺失参数、日期格式错误等异常情况的处理能力，验证是否返回合适的错误代码和错误消息。

6.性能和并发测试：

评估接口在高并发请求下的表现，确保系统能够快速响应并处理大量数据收集和分析请求。