天津宴预订系统设计与实现

曹翔,王新博,钱伶玥,周宗奇,綦新贺

（南开大学电子信息与光学工程学院，天津 300350）

摘要：针对传统手工餐厅预订系统存在的效率低下与数据管理困难等问题，设计并实现了一套基于 C++与 Qt 框架的餐厅预订管理系统.系统采用面向对象程序设计方法，结合三层架构，实现顾客、员工与经理的多角色协同操作.通过封装、继承与多态等机制，完成预订状态跟踪、餐桌资源动态管理、菜单维护及业务数据分析等功能，并借助 Qt 实现图形化交互界面.

关键词：餐厅预订系统；面向对象设计方法；Qt 开发；类与对象；协同

一 引言

1.1 项目背景与意义

随着餐饮业的快速发展，传统手工预订系统已无法满足现代餐厅的运营需求.某餐厅目前使用手工预订表单记录餐桌预订信息，这种系统存在一些问题，比如效率低下、易出错、难以识读、难以获取管理数据等.手工预订表单由多行组成，每行对应一个餐桌，记录预订时间、用餐人数、顾客姓名和电话等信息.餐厅每天分为三个用餐时段，但预订可以跨时段安排.

这些问题严重影响了餐厅的运营效率和顾客体验.因此，开发一个预订管理系统成为当务之急.对比市场上已有的餐厅管理系统，大多功能复杂、成本高昂，且不符合该餐厅的操作习惯.因此，需要开发一个简洁、易用、符合餐厅实际需求的定制化系统.

1.2 项目意义

我们认为开发本系统具有以下重要意义：

[1] 提升运营效率：通过数字化管理取代手工记录，减少人为识读错误，提高预订处理速度.
[2] 优化资源配置：实时监控餐桌使用状态，最大化餐桌利用率.
[3] 改善体验：提供便捷的在线预订服务，支持个性化需求.
[4] 数据分析：为管理者提供业务统计和决策支持数据.
[5] 教学实践：作为面向对象程序设计的实践项目，展示封装、继承、多态

等核心概念的应用.

二 系统分析

2.1 需求分析

基于 1.1 节中对餐厅要求的分析，我们整理出了以下要点，并按照这些需求进行设计.

2.1.1 功能性需求

1.多角色系统

该系统应该是一个多角色用户系统，支持不同用户的使用需求.

顾客：能够进行餐厅预订、查看菜单、管理个人预订记录

员工：能够管理预订状态、处理顾客到店/离开、安排临时顾客

经理：能够管理餐厅资源（餐桌、菜品）、查看业务统计、生成报表

2. 预订管理功能

支持预订，包括选择时间、人数、特殊需求

支持预订状态的全程跟踪（待确认、已确认、已取消、已拒绝）

支持智能餐桌匹配，根据人数和偏好自动推荐合适餐桌

3. 餐厅资源管理

餐桌管理：增加、删除、修改餐桌信息（容量、位置、状态）

菜单管理：菜品的增删改查、价格调整、售卖与否的调整

4. 业务分析与报告

为了统计信息,为日后调整菜单、餐桌等提供依据,应当可以进行如下操作：

预订数据统计与分析；餐桌使用率计算；预订报表生成与导出

2.1.2 非功能性需求

1. 可用性

界面友好，操作简单，用户学习成本低；响应时间合理.

2. 可维护性

遵循面向对象设计原则；代码结构清晰，注释完善；模块化设计，便于功能扩展和修改.

2.2 关键问题分析

开发全流程中我们识别了以下关键而棘手的问题，这些将在 3.3.3 节中进行说明：

[1] 对象生命周期管理问题: 顾客对象何时创建？何时销毁？预订数据如何与餐桌和顾客关联？谁与谁关联？

[2] 界面与逻辑分离问题: 如何设计清晰的分层架构？如何确保业务逻辑的可测试性？

[3] 数据持久化问题: 如何在不使用数据库的情况下实现数据存储(或暂时存储)？如何保证数据的一致性和完整性？

三 系统设计与实现

3.1 用例图

基于 2.1.1 节中对多用户角色的描述，我们最终确定了以下用例：顾客、员工和经理.他们之间的关系表示如下.

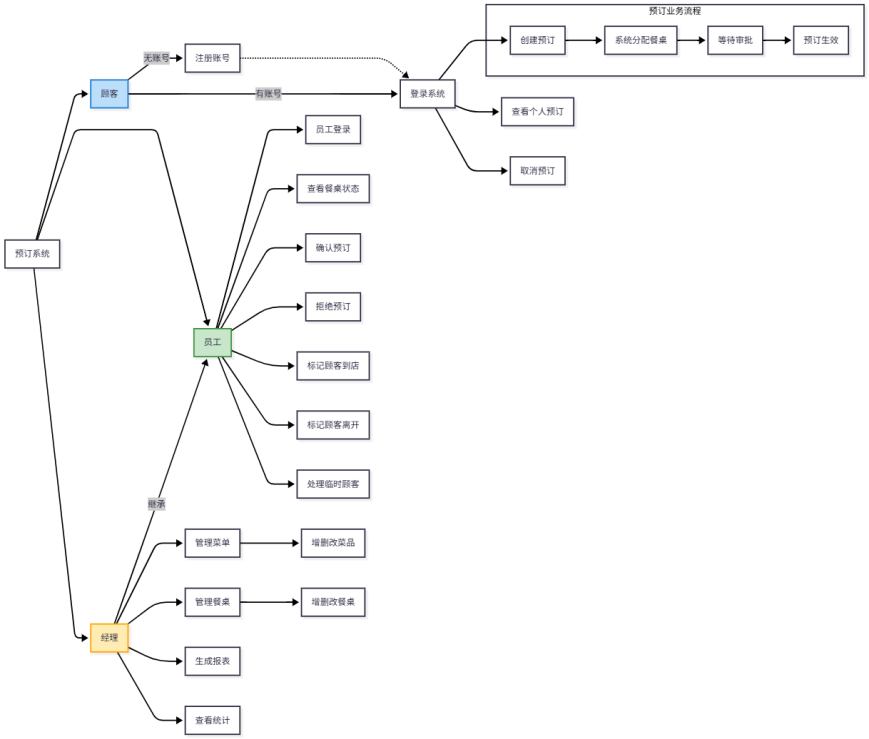

图 3-1 餐厅预订系统用例图

3.2 面向对象设计分析

3.2.1 架构设计法

本系统采用典型的三层架构设计，使得面向对象的设计更快更清晰.同时用三层架构指导开发顺序，也即先开发命令行后 Qt 可视化版本.

[1] 数据层：包含 FoodItem、CustomerInfo、StaffInfo 等数据结构，以及 Table、Reservation、Menu 等核心业务类
[2] 业务层：包含 Customer、Staff、Manager、Restaurant 等类，实现系统的核心业务逻辑
[3] 表示层：基于 Qt 框架的各个界面类，提供用户交互界面

3.2.2 面向对象分析

在类设计过程中，我们首先确定了系统的核心实体：Table、Reservation、Customer、Menu、Restaurant、Staff、Manager.通过分析这些实体之间的关系，我们绘制了类图，明确了各类的职责和关联.下方的类图分为核心类和用 Qt 重构之后的全部类，它们之间的关系表示如下.

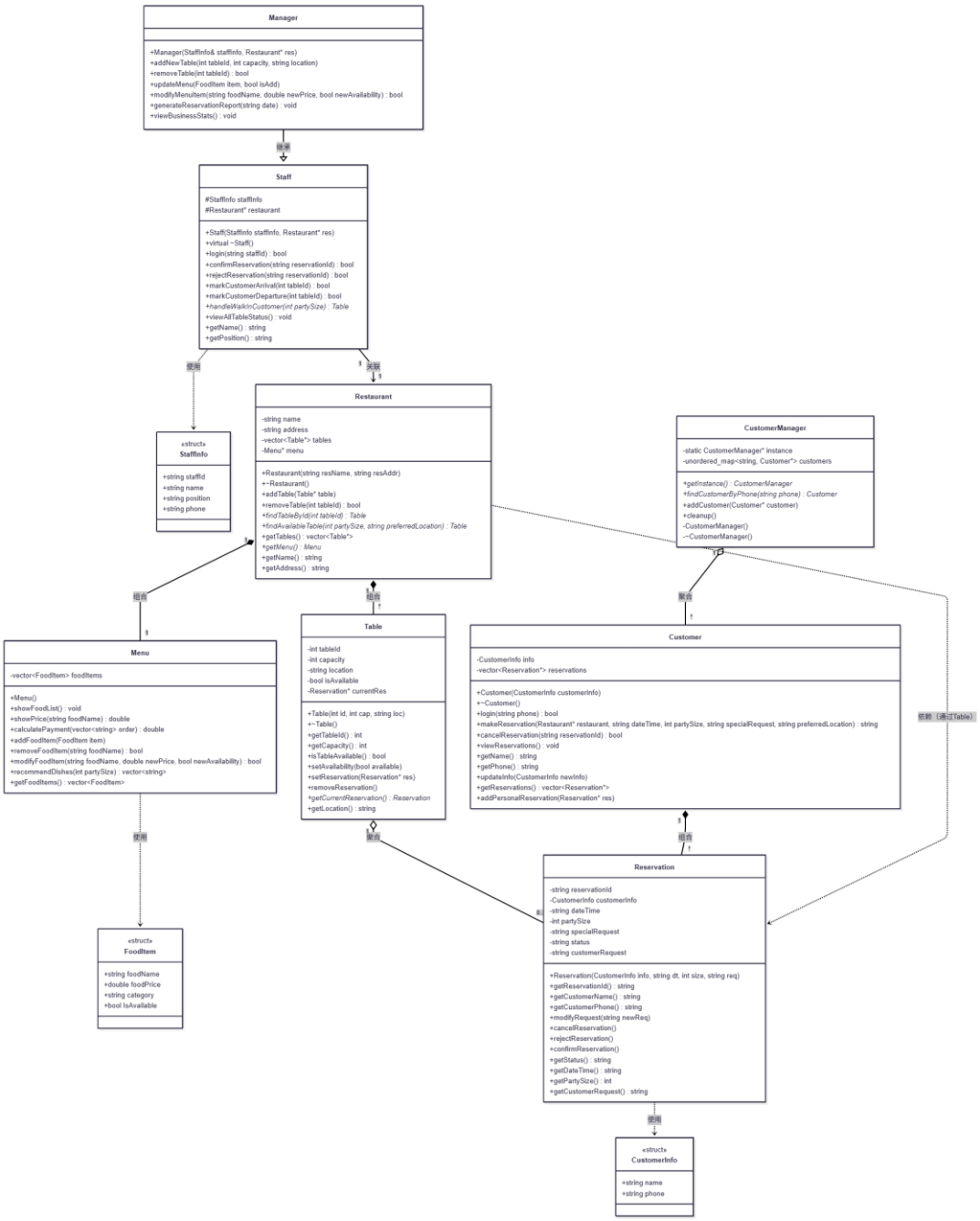

图 3-2 核心类类图

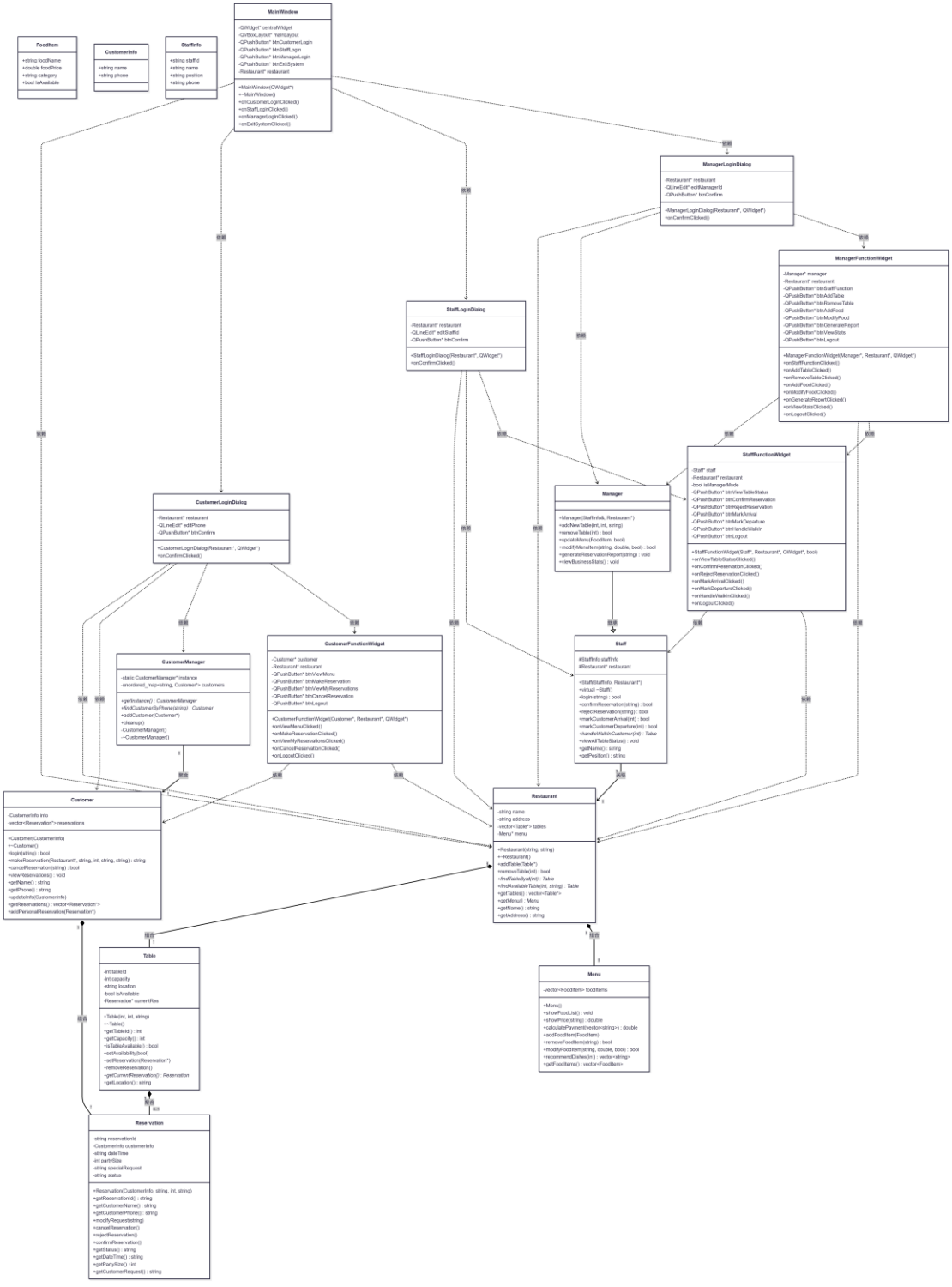
MainWindow
FoodItem
CustomerInfo
StaffInfo
ManagerLoginDialog
ManagerFunctionWidget
StaffLoginDialog
CustomerLoginDialog
Manager
StaffFunctionWidget
CustomerManager
CustomerFunctionWidget
Staff
Customer
Restaurant
Table
Menu
Reservation

图 3-3 系统完整类图

3.2.3 面向对象的具体应用

1.封装性原则

系统严格遵循封装原则，所有类的成员变量均设置为 private 或 protected，通过公有方法提供访问接口.例如 Table 类封装了餐桌 ID、容量、位置、可用状态等属性；Reservation 类封装了预订信息，包括顾客信息、时间、人数等；Menu 类封装了菜品列表和相关操作.

2. 类关系设计

系统包含丰富的类间关系：

[1] Table 与 Reservation：聚合关系，一个餐桌可以有 0 个或 1 个当前预订
[2] Customer 与 Reservation：组合关系，一个顾客可以有多个预订记录
[3] Staff 与 Restaurant：关联关系，员工操作餐厅资源
[4] Manager 与 Staff：继承关系，经理继承员工功能并扩展管理功能

3. 继承与多态

系统通过继承实现代码复用和功能扩展，在设计 Manager 类时尤其凸显.Manager 继承 Staff 的基本功能，同时扩展管理特定功能.Staff 类使用虚析构函数，支持多态特性，确保派生类对象的正确销毁.

3.3 核心功能实现

完成后的系统共有 16 个头文件，16 个源文件，按类划分，依面向对象原则分隔函数.展示如下

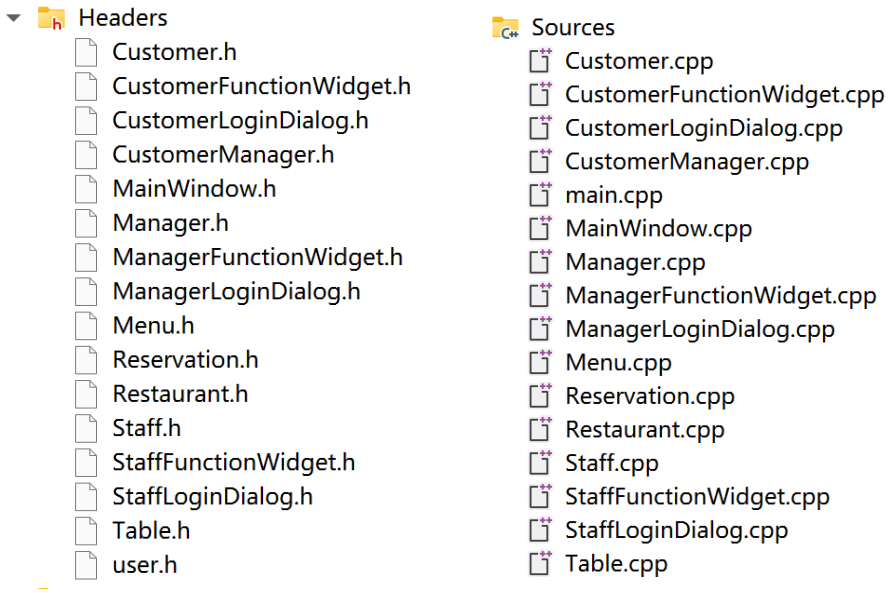

图 3-4 图 3-5 头文件与源文件展示

3.3.1 数据模型模块

数据模型模块定义了系统的基础数据结构：

FoodItem 结构体：存储菜品信息，包括名称、价格、类别、可用状态

CustomerInfo 结构体：存储顾客基本信息，包括姓名、电话

StaffInfo 结构体：存储员工信息，包括工号、姓名、职位、电话

这些结构体在 user.h 中定义，作为系统数据传输的基础单元.

3.3.2 重要模块

系统包含三类用户角色：

[1] Customer 类支持注册和登录，可进行预订、取消、查看等操作，还有个人预订记录管理；
[2] Staff 类支持预订确认和拒绝，可进行顾客到店和离开标记，还有临时顾客处理；
[3] Manager 类继承自 Staff，具备员工所有功能，还可进行餐桌资源管理、菜单内容管理与报表生成和统计查看.

Table 类在 Table.cpp 中定义，负责餐桌资源的管理，主要功能包括餐桌状态的查询和设置、预订的关联和解除、位置和容量信息管理.

Reservation 类负责预订信息的管理，包含预定 ID、顾客信息、预定时间、用餐人数、特殊需求、预定状态等，预订状态包括："Pending"、"Confirmed"、"Cancelled"、"Rejected".

Menu 类负责菜品信息的管理，可支持菜单展示、菜品增删改查等.

Restaurant 类作为系统的核心管理类，负责餐桌管理方法、查询和匹配功能、资源获取方法.Restaurant 类的 findAvailableTable 方法根据用餐人数和偏好位置查找可用餐桌.优先匹配位置偏好，若无则匹配任何可用餐桌.

3.3.3 开发所遇问题

在设计初期，我们面临一个关键决策，应当以哪个对象作为系统的基本对象，是预订（Reservation）、餐桌（Table）还是顾客（Customer）？经过多次讨论，我们最终确定如下设计：

[1] Table 作为资源核心，餐厅的核心资源是餐桌，有关预定的所有业务要围绕餐桌展开；
[2] Reservation 作为业务核心，是连接顾客和餐桌的桥梁；
[3] Customer 作为用户核心，是服务的对象，需要长期管理，在程序运行的过程中需要暂时存储.

在开发初期，我们对以哪个对象为核心存在疑惑.经过讨论，我们认识到 Table、

Reservation 和 Customer 都是核心对象，但需要明确它们之间的关系.最终，我们确定以 Reservation 作为连接 Table 和 Customer 的桥梁，一次预订关联一个餐桌和一个顾客，这样既保持了餐桌的独立性，也记录了顾客的预订历史.我们编写了 makeReservation 函数，进行查找可用餐桌、创建对象、绑定餐桌、绑定顾客等操作.

建立预订时，系统需进行

[1] 为顾客创建 Reservation 对象；
[2] 将其与匹配的 Table 对象绑定（通过 Table::setReservation）；
[3] 将该预订加入顾客的历史记录及 CustomerManager 管理.

在开发过程中，我们发现顾客数据管理存在问题，问题在于是否使用数据库与如何保证数据不丢失，也即由于未使用数据库，无论是否析构顾客对象，顾客从顾客界面退出后数据丢失，不可理解.为了解决这个问题，我们设计了 CustomerManager 类，采用单例模式，在程序运行期间始终存在，以手机号存储顾客指针，提供查找和添加顾客的方法，并在析构时释放所有顾客对象，用于保存所有顾客对象.这样，顾客登录时可以从 CustomerManager 中获取已有对象，新顾客则被添加到 CustomerManager 中.虽然这种方案在并发和大量数据方面有局限，但对于课程项目来说足够.

3.4 图形用户界面实现

在 Qt 界面中，我们使用信号槽机制将界面事件与业务逻辑连接.每个界面类持有对应的业务对象指针，通过调用业务对象的方法来执行操作，并将结果更新到界面.

3.4.1 主界面设计

MainWindow 类提供系统入口界面，包含四个主要功能按钮：

[1] 顾客登录
[2] 员工登录
[3] 经理登录
[4] 退出系统

界面设计简洁明了，采用垂直布局，按钮样式统一，每个按钮都有明确的功能标识.界面标题为"【天津宴】餐厅预订系统"，突出餐厅品牌.

3.4.2 顾客功能界面

CustomerFunctionWidget 类提供顾客操作界面，包含五个主要功能：

[1] 查看菜单
[2] 预订餐桌
[3] 查看我的预订
[4] 取消预订

[5] 退出登录

预订功能通过对话框收集详细信息，包括时间、人数、特殊需求等.

预订操作有新建预订和取消预订，新建预定：选择餐桌→输入时间、人数、顾客信息→检查冲突→确认创建；取消预订：选择预订→确认取消→系统释放餐桌并更新状态.

3.4.3 员工功能界面

StaffFunctionWidget 类提供员工操作界面，包含七个主要功能：

[1] 查看所有餐桌状态
[2] 确认预订
[3] 拒绝预订
[4] 标记顾客到店
[5] 标记顾客离开
[6] 处理临时到店顾客
[7] 退出登录

界面支持经理模式和普通员工模式，根据登录角色调整功能.

3.4.4 经理功能界面

ManagerFunctionWidget 类提供经理操作界面，包含八个主要功能：

[1] 员工端所有功能
[2] 添加新餐桌
[3] 移除餐桌
[4] 添加菜品到菜单
[5] 修改菜品信息
[6] 生成预订报表
[7] 查看业务统计
[8] 退出登录

经理界面集成了管理功能和操作功能，统计信息以文本形式清晰展示，报表生成功能支持日期选择和文件保存.

3.5 开发流程

在整个开发过程中，我们遵循了以下开发流程：

[1] 确定类图：首先绘制类图，明确类的职责和关系；
[2] 命令行版本开发：先开发命令行版本，实现核心业务逻辑，包括 Table、Reservation、Customer 类的编写和测试；
[3] 处理临时顾客功能：实现员工处理临时顾客的功能，即直接分配可用餐桌；
[4] 设计预订逻辑：设计并实现预订的逻辑，确保 Table、Reservation、

Customer 正确绑定；

[5] 处理预订任务：实现顾客预定、员工对预订的确认等功能，完成基本流程；

[6] 完成经理功能：对预定任务进行统计，增删改餐桌、菜单，并输出报表等；

[7] 完成命令行版本：将所有功能在命令行版本中整合测试；

[8] Qt 重构：基于命令行版本，使用 Qt 进行图形界面重构，增加图形化实现类，重构代码，将业务逻辑与界面分离；

[9] 测试与封装：进行系统测试，修复 bug(如添加菜品时的错误)，进行封装.

四 应用前景和展望

4.1 应用前景

本系统设计初衷高度贴近中小型餐厅的实际运营场景，具备良好的直接应用前景.其界面与操作逻辑源于传统表单，员工培训成本低，易于接受.系统功能完整覆盖了从预订接待到营业分析的核心环节，且基于 C++/Qt 开发，具有运行高效、部署简单、维护方便的特点，非常适合作为广大中式餐厅、特色餐馆进行数字化管理升级的入门解决方案.

4.2 不足与展望

首先，在数据方面，系统避开使用 MySQL 等数据库进行数据存储，虽然避免了数据库的复杂部署，但也带来了数据查询效率低下、并发访问控制薄弱等固有缺陷，未来希望能实现完整的数据持久化.

其次，系统的架构设计存在一定的耦合度，特别是业务逻辑与 Qt 界面之间的依赖较为紧密，这为后续的界面重构或移植到其他 GUI 框架增加了难度，且系统的美观性有待进一步加强.

再者，系统缺乏完整的异常处理机制，当出现数据文件损坏或非法输入时仅能提供基础的错误提示，不利于开发人员维护.

五 总结

本餐厅预订系统基于 C++和 Qt 框架开发，采用面向对象设计方法，实现了多角色用户管理、智能餐桌匹配、可视化操作界面等核心功能.系统设计遵循软件工程原则，具有良好的可扩展性、可维护性和用户体验.

通过本项目的开发，我们深入实践了面向对象程序设计的核心概念，包括封装、继承、多态、设计模式等，掌握了大型软件系统的架构设计和实现方法.系统不仅具有实际应用价值，也为相关领域的技术研究和教学提供了参考案例.

任务分工表

任务	曹翔 2412910	王新博 2411048	钱伶玥 2411456	周宗奇 2412620	綦新贺 2414089
讨论应当实现的功能	√	√	√	√	√
确定用例与类 进行系统设计	√	√	√	√	√
代码功能实现	1 设计与编写所有用户类、预定类、餐桌类等基本类，完成预定流程主功能 2 设计与实现经理功能 3 完成命令行版本编写 4 思考解决开发所遇问题	1 完成 Qt 版本重构，设计界面，编写图形化实现类 2 设计预定类，处理预定流程 3 思考解决开发所遇问题	1 完成 Qt 版本重构，设计界面，编写图形化实现类 2 优化主界面 3 设计与优化经理功能 4 思考解决开发所遇问题	1 思考解决开发所遇问题	
代码 Bug 检查	√	√	√	√	√
代码调试与打包	√	√	√		
展示 PPT			√		

表 W-1 任务分工表