QuickEats 项目文档-User1

组别：工具组

**1. 引言**

1.1 背景

在当今快节奏的生活中，消费者对餐饮服务的需求日益倾向于快速与便捷。QuickEats 应运而生，它是一款专为快速浏览、订购和获取本地餐厅精选菜品而设计的移动应用平台。QuickEats 致力于通过其直观易用的界面，为用户提供快速便捷的外卖点单流程。相较于其他同类应用，QuickEats 更注重用户体验，提供从餐厅浏览、菜品展示、购物车管理、订单创建、实时状态跟踪以及支付和评价等一系列功能，为用户提供全流程无缝体验。

背景进一步细分为以下两部分：

市场需求

随着都市生活节奏的加快，人们对于便捷、快速的就餐方式的需求日益增加。特别是在上下班高峰期、加班或没有时间外出就餐时，外卖服务成为重要的饮食解决方案。然而，市场上仍存在一些软件需时过多、用户体验不完善的问题，这为我们开发更加用户友好且快速的点餐软件提供了契机。

技术发展

移动互联网技术的进步使得外卖服务成为可能，同时伴随着大数据分析、云计算服务、人工智能等领域的长足发展，使得在提升订餐效率的同时保证交易安全的前提下，能够打造出更能满足用户需求的外卖软件。

1.2 业务目标

本文档旨在明确 QuickEats 的软件需求，为开发团队提供清晰的指导和方向，以确保所有功能的实现都符合用户的期望和公司的业务要求。通过本需求规约文档的制订，管理层、技术实施小组以及项目管理人员能够围绕共同的需求目标开展合作，减少因理解偏差而导致的效率低下的可能，确保产品最终能满足广大用户群体的需求，并达到预期的市场效益。

1.3 范围

本文档覆盖 QuickEats 的所有功能需求、非功能需求以及相关的约束条件。包括但不限于：

- 用户管理（如创建、修改及删除用户信息）

- 餐厅及菜品浏览与订单管理（如订单建立、支付处理及状态更新）

- 评价机制（提供用户对服务的反馈）

- 支付方式管理以及支付记录的管理

1.4 定义、缩写与缩略语

为保证文档能够清晰准确地传达内容，以下列出本文档中所有专业术语、缩写和缩略语及其定义：

- QuickEats（QE）：本软件的名字，是专为用户提供便捷的餐饮选择和点餐服务的应用。

- UE/UI（用户体验/用户界面）：指用户在与软件互动过程中所面对的体验和界面，设计内容涉及界面布局、视觉表现和交互流程等。

- API（应用程序编程接口）：用于访问内部功能集，允许外部应用通过调用定义良好的接口来访问数据或执行任务。

- SDK（软件开发工具包）：应用程序开发者在创建应用及接口时所需要的软件工具，通常包括库文件、示例代码以及文档等。

- CRUD（Create Read Update Delete）：对数据的添加、查询、更新和删除四个基本操作的英文缩写，广泛应用于数据库管理及软件开发中。

- ECMA：欧洲计算机制造商协会，其定义了包括 JavaScript 等在内的一系列规范，是标准化工作的重要参与者。

- ISO：国际标准化组织，制定了一系列全球性标准，如信息安全 ISO 27000 系列。

- DBMS（数据库管理系统）：用于创建和管理数据库的软件。

- Web API：Web 应用程序编程接口，是传统 API 的一个子类型，用于浏览器和服务器间的网络通信。

- HTTP/HTTPS：传输协议，HTTPS 在 HTTP 基础上增加了信息安全传输的功能，提供加密和身份验证等功能。

- RESTful：一种基于 Web 的软件架构风格与设计约束，通常使用 HTTP 作为协议，该风格通过资源的方式看待整个系统的设计。

- OAuth/OIDC：由 OpenID 基金会维护的一种开放标准，用于处理 API 授权，能够访问通过重定向身份验证令牌的资源。

- JWT（JSON Web Token）：一种开放标准（RFC 7519），用于在双方之间安全地传输信息作为 JSON 对象。

- PWA（Progressive Web App）：渐进式 Web 应用，是一种使用现代 Web 技术构建的 Web 应用，具有类似原生应用的体验，能够在低质量网络下良好运行。

- DPI：密度像素，用来标称智能手机屏幕等级，用于界面元素尺寸调整与图像显示优化。

- TDD（测试驱动开发）：一种软件开发方法，开发人员编写测试用例，驱动程序的开发过程，确保程序符合预期的功能需求。

1.5 参考标准

为了更好地满足本项目的开发要求，本文档主要参考以下文档、标准和其他参考资料：

- ISO/IEC 25010:2011 - 软件工程——软件产品质量需求和评价（产品质量框架）：用于指导此项目质量要求以及质量度量的制定。

- ISO/IEC 9126-1:2001 - 软件工程——产品质量：该项目所遵循的质量模型，以确保所提供功能的健壮性和鲁棒性。

- National Institute of Standards and Technology (NIST) Special Publication 800-53：提供了安全管理、网络技术等领域国际推荐标准以及与之相关的控制和评估规范。

- HTTP/1.1 和 HTTP/2：该协议规定了网络交互的规则，将在与互联网相关的功能实现时予以参照。

1. **总体概述**

2.1 产品视角

2.1.1 愿景与目标

QuickEats 致力于为用户提供一个全面、高效且用户友好的在线订餐系统，涵盖从餐厅浏览到菜品选择、购物车管理、订单创建、支付、订单状态跟踪以及评价发布等功能。其愿景是成为市场领先的在线订餐平台，不仅仅提供基本的订餐功能，还通过以下方式提供超越传统电商平台的价值：

- 打造精美的用户体验界面。

- 提供增强的订单跟踪和评价反馈功能。

- 在安全性和可靠性方面力求卓越，保护用户数据安全并提供稳定的系统性能，赢得用户的信任。

2.1.2 目标用户群体

QuickEats 主要面向以下用户群体：

- 年龄层：以 18-45 岁为主，涵盖学生、职员等多个职业阶层。

- 订餐习惯：经常光顾餐厅或使用网络订餐平台。

- 需求特点：对菜品口味、服务质量有较高要求，注重便捷性、个性化体验和即时反馈。

2.1.3 市场定位

在竞争激烈的在线订餐市场中，QuickEats 将抓住核心用户体验和服务质量两大驱动力。与传统订餐平台相比，QuickEats 的主要优势包括：

- 用户体验优化：提供个性化的菜品推荐、快速响应的订单服务和丰富的餐饮选择。

- 合作餐厅支持：为合作餐厅提供在线业务管理工具，帮助其更好地服务顾客。

2.2 产品功能

2.2.1 功能模块概述

QuickEats 的核心功能模块包括：

| **功能模块** | **描述** |
| --- | --- |
| ****用户管理**** | 用户可以创建账户、管理个人资料、创建及管理订单、添加及删除支付方式、浏览支付记录。 |
| ****餐厅管理**** | 管理平台合作餐厅的资料及数据，提供特定餐厅的信息（如营业时间、联系方式等）。 |
| ****菜品浏览与选择**** | 用户可在选定的餐厅中浏览和选择菜品，查看菜品详细信息（如价格、描述、评分等）。 |
| ****购物车管理**** | 用户可以管理购物车内的菜品，包括添加和移除菜品，预览总消费金额。 |
| ****订单管理**** | 用户可在购物车内容基础上创建订单，修改订单信息，取消订单，查询订单状态。 |
| ****支付处理**** | 用户通过注册的支付方式支付订单，包括支付前的信息认证、支付后的错误处理。 |
| ****订单状态跟踪**** | 用户可随时查看订单的状态变化情况。 |
| ****评价管理**** | 用户在订单完成后可发布评价，餐厅需要定期跟踪评价。 |

2.2.2 模块关联关系

- 用户管理模块：贯穿整个平台的核心机制，从账户创建到消费行为的记录。

- 订购流程：用户浏览菜品，将其添加至购物车并创建订单。

- 支付模块：订单成功创建后，支付模块获取用户数据并确认支付状态。

- 订单状态跟踪与评价管理：记录用户通过订单支付流程后的反馈。

2.3 用户特征

2.3.1 技能水平

目标用户群体平均具备基础的互联网使用技能，如使用移动设备和计算机。不同人群对软件的使用成熟度不同，因此需要确保各功能模块的直观性和易用性。

2.3.2 使用习惯

- 移动优先：大多数用户习惯使用智能手机或平板设备进行交易，因此需要确保移动端的适配性。

- 便利性需求：用户希望操作流程简单直观，按钮和选项布局简单易懂。

- 个性化体验渴望：用户喜欢根据自己的口味偏好看到推荐菜品或餐厅，希望订购服务能满足个人需求。

- 即时反馈需求：用户希望在完成订购、支付、取消订单或评价反馈等操作后能获得即时反馈。

2.4 竞品分析

2.4.1 市场竞争力对比

市面上与 QuickEats 定位相似的产品包括美团、饿了么等主流订餐平台。这些平台已有相当数量的商家入驻，并享有广泛的用户基数，积累了多年的市场运营经验。

2.4.2 市场痛点

- 价格压榨与服务质量：大平台间的恶性竞争导致中小商家利润空间被挤压，服务体验和菜品质量得不到有效保障。

- 用户界面：部分平台的用户界面设计较为复杂，操作元素繁多，使用难度较高。

- 个性化推荐与反馈处理：部分订餐平台的个性化服务反馈不及时，导致用户对产品产生负面评价。

- 支付与退单政策：部分平台存在支付退款政策执行不到位、操作步骤繁琐的问题。

2.4.3 产品优势

- 数据隐私加强：利用最先进的安全措施，如用户权限控制、加密传输等。

- 高质量的服务体验：通过整合用户数据提供个性化推荐，提供商家评价建议。

- 界面简洁化：小巧的界面设计，减少冗余信息。

- 简易支付方案：支持多种支付方式，并确保交易过程中用户支付信息的安全性。

- 流畅系统优化：从主机、应用到传输，提供全方位的稳定服务。

2.5 业务风险与机遇

2.5.1 当前市场情况分析

尽管在线订餐市场仍在发展阶段，但市场增长过程中会面临多种风险和机遇，现存的市场环境要求全新的理念和策略。

2.5.2 业务风险

- 用户数据安全：用户数据一旦泄漏，会严重破坏品牌声誉，影响市场认可。

- 支付系统稳定性：支付过程中可能出现延时或错误，影响用户体验。

- 技术迭代速度：随着技术和用户需求的不断变化，长期维护平台的核心竞争力成为挑战。

- 短期盈利难以保障：前期流量导入需要长期积累，短期难以获得丰厚盈利，存在亏损的风险。

2.5.3 业务机遇

- 新市场拓展：扩大市场覆盖率，通过建立用户基础和声誉，吸引更多商家入驻，增加佣金收入。

- 大数据应用：通过用户数据的采集与分析获取用户习惯，进行菜品推荐和偏好预测，形成具有商业价值的大数据资产。

- 国际化布局：随着品牌实力逐渐增强，可以进入国际市场进行拓展，获取该地区的市场份额及用户基数。

2.6 设计约束

2.6.1 技术栈规范

- 前端技术：基于 HTML5、CSS3 和 JavaScript，集成 React 或 Vue 用于数据交互。

- 后端技术：运用 Java 或 Python，借助 Spring Boot 或 Django 等框架完成后端数据处理。

- 数据库：使用 MySQL 或 MongoDB 来处理订单、用户数据，选择数据保存和恢复能力强、访问速度快的数据库类型。

- 安全与隐私：确保用户的个人信息、支付数据以及交易记录的安全完整保存，对传输的数据进行加密处理。

2.6.2 硬件平台规范

- 移动设备：兼容各种尺寸的智能手机和平板电脑，支持 Android 和 iOS 系统。

- PC 端：适用于多种操作系统环境，如 Windows、Linux、macOS，并以此作为开发、测试的主要平台。

- 服务器：选用高性能服务器硬件，保证并发请求的处理能力，并配置可靠的数据存储方案，利用云服务在整体成本维持相对低的情况下保证数据的安全和冗余，增强业务应对能力。

2.6.3 法规要求

- 合法性：严格遵守当地法律法规，尤其是涉及用户数据隐私保护、支付渠道相关规定等。

- 合规性：确保所有的广告宣传、服务内容以及条款都符合法律法规要求。

2.7 假设与依赖关系

2.7.1 文档中的假设

- 用户基础逐渐扩大，带来商业增长的可能性。

- 合作商家数量不断增多，满足用户订购多样性的需求。

- 系统稳定性高，能够应对突发数据流量。

2.7.2 与系统和其他组件的依赖关系

- 第三方支付系统：接入安全、快捷的支付机构，确保交易操作稳定，且与在线支付系统、操作系统等功能强兼容。

- 用户反馈系统：与各大社交媒体平台和客服系统联动，建立多渠道用户反馈机制。

- 物流系统及 API：对用户下单后的配送流程进行跟踪，与第三方物流合作伙伴合作，确保配送的可靠性。

1. **功能需求**

3.1 用户管理功能

3.1.1 用户注册

- 描述：用户填写注册信息并保存到系统数据库。

- 输入：用户名、密码、电子邮箱、电话号码。

- 输出：在用户表中新增用户记录，字段包括 `UserID`、`UserName`、`Password`、`Email`、`PhoneNumber` 等。

3.1.2 用户登录

-描述：用户输入注册时的用户名和密码，系统验证登录信息是否正确。

-输入：用户名、密码。

-输出：若用户名和密码正确，返回登录成功信息，并允许用户进入系统；若信息错误，返回错误提示并要求重新输入。

3.1.3 管理用户信息

- 描述：管理员或用户更新用户信息或删除用户记录。

- 输入：用户表中的 `UserID` 及更新的信息或删除指令。

- 输出：更新用户表中的信息或删除相关记录。

3.1.4 删除用户

- 描述：管理员从系统中删除用户信息。

- 输入：用户表中的 `UserID`。

- 输出：从用户表中删除用户记录。

3.2 餐厅浏览功能

3.2.1 浏览餐厅

- 描述：用户查看系统中的餐厅列表。

- 输入：无。

- 输出：展示餐厅列表，每条记录包含 `RestaurantID` 及其他相关信息。

3.2.2 搜索餐厅

- 描述：用户根据搜索条件查找餐厅信息。

- 输入：搜索关键词。

- 输出：展示符合搜索条件的餐厅列表，餐厅记录包括 `RestaurantID` 及其他相关信息。

3.2.3 查看餐厅详情

- 描述：用户查看特定餐厅的详细信息。

- 输入：餐厅的 `RestaurantID`。

- 输出：展示该餐厅的详细信息。

3.3 购物车管理功能

3.3.1 添加菜品到购物车

- 描述：用户将选择的餐厅菜品添加到自己的购物车中。

- 输入：用户表中的 `UserID`，菜品表中的 `DishID`。

- 输出：在购物车表中新增购物项，包含 `ShoppingCartID` 及 `DishID`。

3.3.2 从购物车移除菜品

- 描述：用户从自己的购物车中移除菜品。

- 输入：用户表中的 `UserID`，菜品表中的 `DishID`。

- 输出：从购物车表中移除选择的菜品记录。

3.4 订单管理功能

3.4.1 创建订单

- 描述：用户生成订单。

- 输入：用户表中的 `UserID`，用户购物车中的所有菜品信息（`DishID`）。

- 输出：新订单创建后，包括 `OrderID`、`UserID`、`OrderDate` 等字段，并添加到订单表中。

3.4.2 修改订单信息（此需求经人工确认判定为不可行，理由：输入应当为订单号+菜品信息+修改后的订单信息）

- 描述：用户修改其未完成的订单信息。

- 输入：订单表中的 `OrderID`，修改后的菜品信息（`DishID`）。

- 输出：更新订单表中相关订单的信息。

3.4.3 取消订单

- 描述：用户取消未完成订单。

- 输入：订单表中的 `OrderID`。

- 输出：从订单表中删除该订单记录。

3.4.4 支付订单

- 描述：用户为未支付的订单进行支付。

- 输入：订单表中的 `OrderID`，用户选择的支付方式信息，支付金额。

- 输出：在支付记录表中新增一条支付记录（`PaymentRecordID` 及支付金额）。

3.4.5 查看订单状态

- 描述：用户查看其订单的状态信息。

- 输入：订单表中的 `OrderID`。

- 输出：展示选定订单的状态信息。

3.4.6 评价订单

- 描述：用户为已完成的订单进行评价。

- 输入：订单表中的 `OrderID`，用户评价信息（评分、评论）。

- 输出：在评价表中新增一条评价记录（`ReviewID`、`OrderID`、`UserID` 等）。

3.5 支付方式管理功能

3.5.1 管理支付方式

- 描述：用户更新或删除其支付方式信息。

- 输入：用户表中的 `UserID`，更新或删除的支付方式信息。

- 输出：更新或删除支付方式表中的相应记录。

3.6 系统管理功能

3.6.1 创建订单状态

- 描述：管理员添加新的订单状态。

- 输入：订单状态描述信息。

- 输出：在订单状态表中新增一条订单状态记录（`OrderStatusID` 及描述）。

3.6.2 发布评价

- 描述：用户发布新的评价信息。

- 输入：用户表中的 `UserID`，评价对象 ID，评价内容。

- 输出：在评价表中新增一条评价记录。

3.6.3 查看支付记录

- 描述：用户查看与自己相关的支付记录。

- 输入：用户表中的 `UserID`。

- 输出：展示所有属于该用户的支付记录。

1. 外部接口

4.1 介绍

本章节详细描述了 QuickEats 系统与其外部系统间的所有接口。外部接口概述了系统如何与不同外部环境（如操作系统、硬件、其他软件以及数据库）进行通信。以下内容将会列出每个接口的形式、通信协议和数据交换格式等。

4.2 系统与硬件的接口

4.2.1 描述

对于硬件接口的要求，主要包括设备端兼容性与运行参数，确保 QuickEats 能在各种硬件环境下稳定运行。

4.2.2 设备支持

- 移动设备：兼容 iOS 和 Android 不同版本的操作系统。

- PC 端：支持 Windows、Linux、macOS 操作系统。

4.2.3 性能要求

- QuickEats 需要确保在不同硬件配置下的良好性能，包括 RAM、存储和处理器性能。

- 设备支持多任务，以支持后台操作，如订单状态更新、通知接收等。

4.2.4 UI 设计

- 界面设计需考虑不同屏幕大小与 DPI 密度，确保按钮和字体在不同设备上皆能清晰可见。

- 提供触摸友好界面以适合移动设备操作。

4.3 系统与外部软件的接口

4.3.1 描述

QuickEats 通过 API 和 SDK 与第三方系统进行合作，包括但不限于支付平台、物流追踪系统和社会网络系统。

4.3.2 支付系统集成

- 接口类型：RESTful API。

- 通信协议：HTTP/HTTPS。

- 数据格式：JSON。

- 功能：提供支付接口能力，包括但不限于信用卡、借记卡、支付宝、微信支付等功能。

- 需求：确保支付过程顺畅、支付记录准确，具有自动化的支付失败处理机制。

4.3.3 物流合作

- 接口类型：Web API。

- 通信协议：HTTP/HTTPS。

- 数据格式：JSON 或 XML。

- 功能：通过第三方物流服务，实现实时订单状态更新及物流跟踪。

- 需求：提供可靠的物流跟踪数据，以通知用户订单的当前状态，如配送路线、预计到达时间。

4.3.4 社交媒体接口

- 接口类型：OAuth/OIDC。

- 通信协议：HTTP/HTTPS。

- 数据格式：JSON。

- 功能：用户可通过其社交媒体账户（如微信、微博）登录 QuickEats，同时将评价内容分享到社交媒体。

- 需求：支持社交媒体登录及分享功能，确保信息传输安全。

4.4 与数据库的接口

4.4.1 描述

与数据库交互是系统数据存储与管理的重要组成部分，本节定义了相应的接口。

4.4.2 数据库类型

- 支持关系型数据库（如 MySQL）和文档型数据库（如 MongoDB）用于存储用户信息、订单数据以及评价。

- 表结构需满足性能需求，尤其是高并发场景下的数据读写能力。

4.4.3 数据库操作

- 用户管理数据库操作：实现用户数据的创建、检索、更新及删除（CRUD）操作。

- 订单与支付数据库操作：用于订单生成、支付信息的记录、支付成功后的订单更新等操作。

- 评价管理数据库操作：记录用户的评价反馈及相应评价更新操作。

4.4.4 数据格式

- 数据库中的各项信息以结构化形式存储（对于关系型数据库，是以表格的形式）并确保数据格式一致及操作符合数据保护法规。

4.5 数据安全与隐私保护接口

4.5.1 描述

保护用户数据安全及隐私是系统的重要基础。

4.5.2 数据加密

- 所有数据在传输及存储时进行加密处理，推荐使用 HTTPS 协议并采用 JWT 方式对信息进行安全加密。

4.5.3 权限设置

- 各接口操作需具备用户权限验证机制，限制非法访问接口的功能。

4.5.4 日志记录

- 所有敏感操作，如用户管理、数据检索、支付信息处理等需记录日志，以备追踪和审计。

1. 非功能需求

5.1 性能需求

5.1.1 响应时间

为确保系统的高性能和用户友好性，每类操作的响应时间要求如下：

- 登录与登出功能：在正常情况下，登录和登出应在 0.5 秒内完成。在高峰时段（例如晚餐高峰期间），用户登录和登出时间不应超过 1 秒。

- 页面加载时间：无论页面大小如何复杂，食品选择页面（包括浏览和搜索餐厅、菜品显示）的加载时间需要在不到 3 秒内。

- 支付处理时间：支付页面的加载时间必须确保在 2 秒内完成。在用户点击付款后，支付完成后信息显示一般不应超过 10 秒。

- 订单状态更新时间：对于刚刚下完的订单，状态更新时间不应超过 30 秒。对于所有其他正在进行中的订单，订单状态需要每 2 分钟刷新一次或更短周期。

强需求说明：系统需要确保在负载增加的情况下，各页面和功能可用且响应时间合理。对于认证和支付功能，必须始终保持快速响应，即使可能出现高峰期，也要确保其高效运行。

5.1.2 吞吐量

系统在处理大量并发请求时必须保持良好的性能。设定的吞吐量标准如下：

- 浏览操作：系统应能够同时处理 5000 次浏览操作，这包括查看餐厅、浏览菜品、搜索食物等类型的操作。

- 下单操作：在同一时间，下单操作的吞吐量应支持 300 个并发请求。这意味着在晚餐高峰时段，系统容量能够应对大量同时发生的下单行为。

5.1.3 并发用户数

系统具备高并发用户的处理能力：

- 高峰使用场景：在高峰时段，预计系统能够确保至少 5000 名用户同时在线，而不产生任何性能问题或性能下降。

- 正常使用场景：在非高峰时段，系统能够同时支持 10000 个用户使用，包括访问、浏览、下单和查看订单状态等一系列正常操作。

5.1.4 高峰时段性能

在要求最严格的晚餐高峰时段，如下性能指标必须符合：

- 响应时间：所有关键交互的响应时间不能超过 2 秒。包括页面加载时间、数据检索、订单状态更新等。

- 并发用户数：预计晚餐高峰时段会有 10000 名用户在系统中完成关键交易和浏览操作。为此，系统吞吐量必须足以处理这一数量的并发用户。

5.2 质量属性

5.2.1 可靠性

应用必须在各种情况下维持高度的稳定性。

- 故障恢复能力：系统能够在单次故障之后 5 分钟内自行恢复到正常服务水平。其中关键功能需要在 30 秒内恢复正常。

- 冗余设计：为了避免硬件故障导致服务中断，建议使用具有冗余特性的部署来配置服务器资源。这意味着即使一个服务器出现故障，其他同步复制的副本亦可以继续提供无间断服务。

5.2.2 可用性

系统可能因各种原因中断，如网络故障或硬件停机等。为确保系统在这些条件下的持续可用性，需要考虑以下因素：

- 目标可用时间：全年范围内，系统需达到 99.9% 的月度可用时间，即每月最多只有 0.1% 的时间系统可能遇到服务不可用的情况。

- 恢复时间目标（RTO）：在发生故障导致系统暂时中断后，系统应在 1 小时内恢复至有效状态。这意味着任何影响用户服务的问题应在 1 小时内得到解决，以此来最小化停机时间。

- 恢复点目标（RPO）：在系统发生任何故障后，所有数据均要求在 15 分钟内可以完全恢复到故障之前的最新状态。这意味着一旦系统故障，可以尽量减少因为数据丢失而影响运营的风险。

- 定期维护：系统每月进行的定期维护和升级不会影响超过 2 小时的服务，且需提前至少一周通知所有用户。

5.2.3 可维护性

系统和其配置的可维护性对系统稳定性和生命力具有重要意义：

- 错误识别和恢复：系统提供详细的日志文件，可以用于快速识别、查找及解决任何系统错误。维护人员将能通过日志查找任何的问题根源，并通过既定的处理流程恢复系统的正常运作。

- 错误恢复流程：对于简单的系统问题，应能够在 24 小时内解决；复杂问题则需要在 72 小时内完成解决。这将有助于确保系统能够保持可用性。

- 用户文档和培训材料：提供系统使用和维护的详细文档，包括用户手册、系统维护指南、安装和部署文档等。这确保了开发、维护和使用系统的人员对于操作内容具有清晰的理解。

- 易用易理解的功能设计：应用程序应具备清晰和直观的用户界面，所有关键功能均须纳入一个简单明了的设计当中，有助于减少用户使用时的复杂度并简化系统操作步骤。

5.2.4 可移植性

系统的设计应允许轻松地移植到不同的平台和环境上：

- 操作系统兼容性：系统应该能够兼容所有主流的服务器操作系统，包括但不限于 Linux、Ubuntu 和 Redhat 等。这使得在未来的系统升级或者更换硬件供应链上，能够有更多的选择。

- 跨平台支持：确保应用能够在不同的硬件和服务器环境中自行移植和执行，无需做巨大的改动或重写代码以适配不同的基础架构。

5.2.5 数据安全与隐私保护

系统需要利用多种手段保护用户数据安全和个人隐私，具体措施包括：

- 用户数据加密：所有的用户数据（包括个人密码、支付信息等）传输过程均使用安全协议（如 SSL/TLS）进行全程加密传输，确保数据在传输过程中的安全。

- 数据备份及恢复机制：每次重大变化节点后，系统应进行完整数据备份。一旦数据丢失或损坏，系统应在 1 小时内恢复所有备份数据，以确保业务可以平稳地恢复运营。

- 安全合规标准：系统应符合国际和本地的数据保护和信息安全标准，包括但不限于 ISO 27001、以及国家层面的数据保护法律。

- 访问控制和权限管理：系统必须具备严格的用户身份管理和权限分隔。对于不同级别的用户（如管理员和普通用户），应支持配置不同的权限集，保证各个角色只能够访问与其角色匹配的功能，避免不必要的对系统关键领域的访问。

5.2.6 性能和效能评估

为了确保系统持续提供优质服务，系统应定期进行如下诸项性能效能评估：

- 性能测试与评估：系统必须经过“负载测试”和“压力测试”，模拟真实的复杂业务环境，确保系统实际应用场景下的表现符合预期。测试应覆盖常见峰值时间段及各种潜在的使用场景。

- 效能和性能监控：性能监控系统应当安装就位，提供实时的检测与监控功能。这类系统能够持续检查不同组件的性能状态，当系统运行出现问题时，能够发出即时的警告，并指导运维人员进一步诊断和修复问题。

5.2.7 扩展性

系统设计应考虑未来的需求扩展，并支持灵活地增加新功能或改进现有功能，保证其可以随时满足业务增长和用户需求变化的情况：

- 横向扩展能力：系统架构设计使得增加服务器的数量不会显著增加管理复杂性和单个实例的成本，同时能够均匀地分配任务和负载。

- 微服务架构：使用微服务架构，各功能模块独立部署和扩展，减少因系统扩展造成的管理复杂性，提高灵活性。

5.2.8 界面设计及操作简便性

易用性的非功能需求关注的是界面友好度和用户操作便捷性：

- 界面设计：用户界面需简洁、直观，布局合理，符合现代 Web 应用与移动应用的设计规范；所有操作菜单、按钮等应该放在用户能够轻易查找的位置。

- 操作便捷性：用户在进行下单、查看订单状态、支付等操作时步骤应尽可能简化，不需要进行多步导航。此外，应提供一个基于用户反馈的信息检索机制，使用户可通过多种途径查找所需信息。

5.2.9 兼容性和跨设备支持性

工程设计和系统实现需满足跨设备兼容性和不同操作系统与浏览器间兼容：

- 系统兼容性：应用应能够兼容所有主流操作系统的桌面版本、平板电脑端和移动设备版本，包括但不限于 Windows、macOS、ChromeOS、Linux、Android 和 iOS。

- 浏览器兼容性：确保用户无论在任何现代浏览器（如 Chrome、Firefox、Safari、Microsoft Edge 等）上，系统都能正常运行和访问。

5.2.10 可利用的硬件和网络资源

应用程序的安装与部署应能不限制硬件配置，对各种现代处理单元和网络基础设施具备兼容性：

- 硬件要求：系统应能在各种常规的服务器设备和网络环境下平稳运行，硬件需求应根据常规商业系统硬件配置设定。

- 网络效率：系统必须能够在低带宽以及高延迟环境下的跨站点运行。防止和缓解由于网络因素引起的数据缓冲与延迟状况发生，不过分依赖网络带宽，从而确保服务质量。

5.2.11 时间和时区支持性

系统所处理的时间与日期信息应适用不同地方时区和特定的日期格式：

- 时区兼容性：必须处理 UTC 和当前用户配置的本地时间时区，确保所有时间相关的显示和处理均以用户的时间设置为准。

5.2.12 经济考量与月度维护费用

该软件需求规格书也需考虑到经济因素：

- 经济效率：系统需要实现其硬件和托管费用的最小化。通过高效的资源管理和优化利用服务器资源，力求降低维护和运行成本。

- 合理注重成本进行维护：软件成本控制，由于系统运行及维护预计需要一定的经济投入，应综合考虑成本和收益比，控制运维费用维持在可合理接受范围内，确保在不损害服务水平和性能的前提下，保持成本效率。

6.1 用户注册 （User Register）

- 用例编号：UC-01

- 前置条件: 用户未注册，或正在访问注册页面。

- 后置条件：用户成功注册，并跳转至登陆界面。

- 主事件流:

1. 用户打开注册页面。

2. 用户输入用户名、密码、电子邮箱、电话号码。

3. 用户提交注册表单。

4. 系统验证各项信息的有效性：

- 检查用户名是否已存在。

- 检查密码是否符合强密码规则。

- 检查电子邮箱格式是否正确。

- 检查电话号码格式是否正确。

5. 如果输入的所有信息有效，系统将注册信息保存到用户表中。

6. 注册成功，返回成功信息及登录入口。

- 异常情况:

- A1：如果用户输入的邮箱或电话已被注册，提示“该邮箱/电话已被使用”。

- A2：如果密码不符合要求，提示“密码必须包含字母、数字及特殊字符，并且长度至少8位”。

- A3：如果用户名不符合格式，提示“用户名只能包含字母、数字和下划线”。

6.2 用户登录 （Log in）

- 用例编号：UC-02

- 参与者：用户

- 前置条件: 用户已注册并且拥有有效的用户名和密码。

- 后置条件：用户成功登录。

- 主事件流:

1. 用户打开登录页面。

2. 用户输入用户名和密码。

3. 用户点击登录按钮。

4. 系统验证用户名和密码是否匹配。

5. 如果用户名和密码匹配，用户成功登录，并跳转到用户首页或仪表盘。

6. 如果用户名或密码错误，系统提示用户用户名或密码错误，并要求重新输入。

- 异常情况:

- A1：如果用户未注册但尝试登录，提示“该用户名未注册，请先注册”。

- A2：如果密码错误超过限制次数，提示“登录尝试次数过多，请稍后再试”。

6.3 管理用户信息（Manage User Information）

- 用例编号：UC-03

- 参与者：管理员、用户

- 前置条件：管理员或用户已成功登录系统。

- 后置条件：用户信息被更新或删除，系统状态保持一致。

- 主事件流：

1. 管理员或用户登录到系统。

2. 管理员或用户选择“管理用户信息”功能。

3. 系统展示用户信息列表或搜索界面。

4. 管理员或用户选择需要更新或删除的用户记录。

5. 管理员或用户修改用户信息或选择删除操作。

6. 系统验证修改信息的有效性。

7. 系统更新或删除用户记录。

8. 系统返回操作成功的消息给管理员或用户。

- 异常事件流：

- A1：如果尝试更新或删除不存在的用户记录，系统显示错误消息并终止操作。

- A2：如果输入的更新信息无效（如电话号码格式不正确），系统显示错误消息并要求重新输入。

- A3：如果系统在更新或删除过程中遇到错误，系统显示错误消息并终止操作。

6.4 删除用户（Delete User）

- 用例编号：UC-04

- 参与者：管理员

- 前置条件：管理员已成功登录系统。

- 后置条件：用户记录被成功删除。

- 主事件流：

1. 管理员登录到系统。

2. 管理员选择“删除用户”功能。

3. 系统展示用户列表或搜索界面。

4. 管理员选择要删除的用户记录。

5. 系统验证并删除用户记录。

6. 系统返回操作成功的消息给管理员。

- 异常事件流：

- A1：如果管理员尝试删除不存在的用户记录，系统显示错误消息并终止操作。

- A2：如果删除过程中出现错误（如数据库连接失败），系统显示错误消息并终止操作。

6.5 浏览餐厅（Browse Restaurant）

- 用例编号：UC-05

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户看到餐厅列表。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“浏览餐厅”功能。

3. 系统展示餐厅列表。

4. 用户浏览餐厅列表。

- 异常事件流：

- A1：如果系统没有餐厅数据，系统提示“暂无餐厅数据”。

- A2：如果系统数据不完整或加载失败，系统显示错误消息并提示重试。

6.6 搜索餐厅（Search Restaurant）

- 用例编号：UC-06

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户看到匹配的餐厅列表。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“搜索餐厅”功能。

3. 用户输入搜索条件。

4. 系统根据条件搜索餐厅。

5. 系统展示匹配的餐厅列表。

6. 用户浏览匹配的餐厅列表。

- 异常事件流：

- A1：如果搜索条件无效或无匹配结果，系统提示“未找到匹配餐厅”。

- A2：如果系统搜索过程中出现错误，系统显示错误消息并提示重试。

6.7 查看餐厅详情（View Restaurant Details）

- 用例编号：UC-07

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户看到餐厅的详细信息。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“查看餐厅详情”功能。

3. 用户选择要查看的餐厅。

4. 系统展示该餐厅的详细信息。

5. 用户查看餐厅详细信息。

- 异常事件流：

- A1：如果选择的餐厅不存在，系统显示“餐厅不存在”提示。

- A2：如果系统在获取餐厅数据过程中出现错误，系统显示错误消息并提示重试。

6.8 添加菜品到购物车（Add Dish to Cart）

- 用例编号：UC-08

- 参与者：用户

- 前置条件：用户已成功登录系统，用户位于某个餐厅页面。

- 后置条件：菜品被成功添加到购物车。

- 主事件流：

1. 用户登录到系统。

2. 用户选择某个餐厅。

3. 用户浏览该餐厅的菜单。

4. 用户选择菜品并加入购物车。

5. 系统确认菜品已经被添加到购物车。

6. 系统返回操作成功的消息给用户。

- 异常事件流：

- A1：如果选择的菜品不存在或已售罄，系统显示错误消息并终止操作。

- A2：如果购物车内数量达到上限，系统提示“购物车已达上限”。

6.9 从购物车移除菜品（Remove Dish from Cart）

- 用例编号：UC-09

- 参与者：用户

- 前置条件：用户已成功登录系统，用户拥有购物车数据。

- 后置条件：购物车中的菜品被移除。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“购物车”功能。

3. 用户浏览购物车中的菜品。

4. 用户选择要删除的菜品。

5. 系统确认并删除该菜品。

6. 系统返回操作成功的消息给用户。

- 异常事件流：

- A1：如果尝试删除不存在的菜品，系统显示错误消息并终止操作。

- A2：如果删除过程中出现错误，系统显示错误消息并终止操作。

6.10 创建订单（Create Order）

- 用例编号：UC-10

- 参与者：用户

- 前置条件：用户已成功登录系统，有有效的购物车数据。

- 后置条件：订单成功创建。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“创建订单”功能。

3. 系统展示购物车中的菜品列表。

4. 用户确认购物车中的所有菜品。

5. 系统生成订单并保存至系统数据库。

6. 系统返回订单创建成功的消息给用户。

- 异常事件流：

- A1：如果购物车中有无效的菜品，系统显示错误消息并终止操作。

- A2：如果订单创建过程中出现错误，系统显示错误消息并终止操作。

6.11 修改订单信息（Modify Order Information）

- 用例编号：UC-11

- 参与者：用户

- 前置条件：用户已成功登录系统，用户有未完成的订单。

- 后置条件：订单信息被成功更新。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“修改订单信息”功能。

3. 系统展示订单列表。

4. 用户选择需要修改的订单。

5. 用户修改订单信息（如菜品、数量等）。

6. 系统验证修改信息的有效性。

7. 更新订单信息并保存到数据库。

8. 系统返回操作成功的消息给用户。

- 异常事件流：

- A1：如果订单已成交无法修改，系统显示错误消息并终止操作。

- A2：如果用户输入的信息无效，系统显示错误消息并要求重新输入。

6.12 取消订单（Cancel Order）

- 用例编号：UC-12

- 参与者：用户

- 前置条件：用户已成功登录系统，用户有未完成的订单。

- 后置条件：订单被成功取消。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“取消订单”功能。

3. 系统展示订单列表。

4. 用户选择需要取消的订单。

5. 系统确认并取消该订单。

6. 系统返回操作成功的消息给用户。

- 异常事件流：

- A1：如果订单已成交不允许取消，系统显示错误消息并终止操作。

- A2：如果取消过程中出现错误，系统显示错误消息并终止操作。

6.13 支付订单（Pay Order）

- 用例编号：UC-13

- 参与者：用户

- 前置条件：用户已成功登录系统，用户有未支付的订单。

- 后置条件：订单支付完成。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“支付订单”功能。

3. 系统展示待支付的订单列表。

4. 用户选择需要支付的订单。

5. 用户选择支付方式并输入支付信息。

6. 系统验证支付信息的有效性。

7. 系统执行支付操作。

8. 系统返回支付成功的消息给用户。

- 异常事件流：

- A1：如果支付信息无效或错误，系统显示错误消息并要求重新输入。

- A2：如果支付过程中出现错误，系统显示错误消息并终止操作。

6.14 查看订单状态（View Order Status）

- 用例编号：UC-14

- 参与者：用户

- 前置条件：用户已成功登录系统，用户有订单数据。

- 后置条件：用户看到订单的状态。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“查看订单状态”功能。

3. 系统展示订单列表。

4. 用户选择查看的订单。

5. 系统展示该订单的状态。

- 异常事件流：

- A1：如果选择的订单不存在，系统显示“订单不存在”提示。

- A2：如果系统在获取订单数据过程中出现错误，系统显示错误消息并提示重试。

6.15 评价订单（Rate Order）

- 用例编号：UC-15

- 参与者：用户

- 前置条件：用户已成功登录系统，用户有已完成的订单。

- 后置条件：用户评价信息被成功提交。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“评价订单”功能。

3. 系统展示已完成订单列表。

4. 用户选择要评价的订单。

5. 用户输入评价信息。

6. 系统确认并保存该评价。

7. 系统返回评价成功的消息给用户。

- 异常事件流：

- A1：如果用户尝试评价未完成的订单，系统显示错误消息并终止操作。

- A2：如果评价过程中出现错误，系统显示错误消息并终止操作。

6.16 管理支付方式（Manage Payment Method）

- 用例编号：UC-16

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户的支付方式信息被更新或删除。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“管理支付方式”功能。

3. 系统展示用户支付方式列表。

4. 用户选择更新或删除支付方式。

5. 系统更新或删除支付方式信息。

6. 系统返回操作成功的消息给用户。

- 异常事件流：

- A1：如果用户尝试更新或删除不存在的支付方式，系统显示错误消息并终止操作。

- A2：如果输入的更新信息无效，系统显示错误消息并要求重新输入。

6.17 删除支付方式（Delete Payment Method）

- 用例编号：UC-17

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户的支付方式信息被删除。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“删除支付方式”功能。

3. 系统展示用户支付方式列表。

4. 用户选择要删除的支付方式。

5. 系统确认并删除支付方式。

6. 系统返回操作成功的消息给用户。

- 异常事件流：

- A1：如果删除过程中出现错误，系统显示错误消息并终止操作。

- A2：如果用户尝试删除不存在的支付方式，系统显示错误消息并终止操作。

6.18 创建订单状态（Create Order Status）

- 用例编号：UC-18

- 参与者：管理员

- 前置条件：管理员已成功登录系统。

- 后置条件：订单状态被保存到数据库。

- 主事件流：

1. 管理员登录到系统。

2. 管理员选择“创建订单状态”功能。

3. 系统展示订单状态创建界面。

4. 管理员输入订单状态信息。

5. 系统确认并保存订单状态信息。

6. 系统返回操作成功的消息给管理员。

- 异常事件流：

- A1：如果输入的订单状态信息无效，系统显示错误消息并要求重新输入。

- A2：如果保存过程中出现错误，系统显示错误消息并终止操作。

6.19 管理订单状态（Manage Order Status）

- 用例编号：UC-19

- 参与者：管理员

- 前置条件：管理员已成功登录系统。

- 后置条件：订单状态信息被更新或删除。

- 主事件流：

1. 管理员登录到系统。

2. 管理员选择“管理订单状态”功能。

3. 系统展示订单状态列表。

4. 管理员选择需要更新或删除的订单状态。

5. 管理员更新订单状态信息或选择删除操作。

6. 系统确认并保存或删除订单状态信息。

7. 系统返回操作成功的消息给管理员。

- 异常事件流：

- A1：如果更新或删除过程中出现错误，系统显示错误消息并终止操作。

- A2：如果尝试更新或删除不存在的订单状态，系统显示错误消息并终止操作。

6.20 发布评价（Publish Rating）

- 用例编号：UC-20

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户的评价信息被成功发布。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“发布评价”功能。

3. 系统展示发布评价界面。

4. 用户输入评价信息并确认发布。

5. 系统保存评价信息。

6. 系统返回发布成功的消息给用户。

- 异常事件流：

- A1：如果输入的评价信息无效，系统显示错误消息并要求重新输入。

- A2：如果发布过程中出现错误，系统显示错误消息并终止操作。

6.21 删除评价（Delete Rating）

- 用例编号：UC-21

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户的评价信息被成功删除。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“删除评价”功能。

3. 系统展示用户评价列表。

4. 用户选择需要删除的评价。

5. 系统确认并删除评价。

6. 系统返回操作成功的消息给用户。

- 异常事件流：

- A1：如果删除过程中出现错误，系统显示错误消息并终止操作。

- A2：如果用户尝试删除不存在的评价，系统显示错误消息并终止操作。

6.22 查看支付记录（View Payment Records）

- 用例编号：UC-22

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户看到自己的支付记录。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“查看支付记录”功能。

3. 系统展示用户的支付记录界面。

4. 用户浏览支付记录。

- 异常事件流：

- A1：如果系统没有支付记录，系统提示“暂无支付记录”。

- A2：如果系统在获取支付记录过程中出现错误，系统显示错误消息并提示重试。

6.23 删除支付记录（Delete Payment Record）

- 用例编号：UC-23

- 参与者：用户

- 前置条件：用户已成功登录系统。

- 后置条件：用户自己的支付记录被成功删除。

- 主事件流：

1. 用户登录到系统。

2. 用户选择“删除支付记录”功能。

3. 系统展示用户支付记录列表。

4. 用户选择需要删除的支付记录。

5. 系统确认并删除支付记录。

6. 系统返回操作成功的消息给用户。

- 异常事件流：

- A1：如果删除过程中出现错误，系统显示错误消息并终止操作。

- A2：如果用户尝试删除不存在的支付记录，系统显示错误消息并终止操作。