**QuickEats 餐饮及外卖软件需求文档**

**1. 引言**

**1.1 目的**

本文档旨在定义 QuickEats 软件的功能需求、非功能需求以及系统设计约束，为开发团队提供清晰的需求指导，并确保产品满足用户对速度、便利性和用户体验的期望。

**1.2 范围**

QuickEats 是一款专注于餐饮及外卖服务的应用程序，目标用户为需要快速点餐的消费者。本软件的主要功能包括餐厅浏览、菜品选择、购物车管理、订单提交、实时跟踪和支付处理。范围涵盖从前端用户界面到后端服务接口的完整流程。

**1.3 定义、缩写与缩略语**

* **QuickEats**：餐饮及外卖软件名称。
* **API**：应用程序编程接口（Application Programming Interface）。
* **UI**：用户界面（User Interface）。
* **GPS**：全球定位系统（Global Positioning System）。

**1.4 参考文献**

* 《软件需求工程》（作者：Dean Leffingwell）
* 《人机交互设计指南》（作者：Jenifer Tidwell）

**2. 总体概述**

**2.1 产品视角**

QuickEats 是一款移动优先的应用程序，通过整合本地餐厅资源，为用户提供从浏览到下单的一站式服务。它依赖于地理位置服务（GPS）获取附近餐厅信息，并通过支付网关完成交易。

**2.2 产品功能**

* 快速浏览附近餐厅及其热门菜品。
* 支持自取或外卖模式的订单提交。
* 提供实时订单状态跟踪功能。
* 无缝集成多种支付方式（如信用卡、电子钱包）。

**2.3 用户特征**

* **普通用户**：希望通过手机快速完成点餐操作的消费者，关注便捷性和效率。
* **餐厅管理员**：负责上传菜单、更新库存并查看订单信息。

**2.4 竞品分析**

* **美团外卖**：功能全面但界面复杂，加载速度较慢。
* **饿了么**：注重促销活动，但部分区域配送时间较长。
* **QuickEats 的优势**：专注于速度和简洁性，减少用户操作步骤，提升整体体验。

**2.5 设计约束**

* 数据库需支持高并发访问，以应对多用户同时点餐场景。
* 系统需兼容主流支付网关（如支付宝、微信支付）。

**2.6 假设与依赖关系**

* 假设用户设备已开启 GPS 功能，且网络连接稳定。
* 系统依赖第三方地图 API 获取用户位置。

**3. 功能需求**

**功能点 1：餐厅浏览**

* **描述**：用户可通过地图或列表形式浏览附近的餐厅信息，包括名称、评分和推荐菜品。
* **输入**：用户当前位置坐标（经度、纬度）。
* **输出**：显示距离最近的餐厅列表，每条记录包含餐厅图片、名称、评分等信息。

**功能点 2：菜品选择**

* **描述**：用户点击某个餐厅后可查看其菜单，并选择菜品加入购物车。
* **输入**：餐厅 ID，用户选择的菜品 ID 和数量。
* **输出**：更新后的购物车详情，包括总价和所选菜品清单。

**功能点 3：购物车管理**

* **描述**：用户可查看购物车中已选菜品，并修改数量或删除菜品。
* **输入**：购物车 ID，菜品 ID，操作类型（增加、减少或删除）。
* **输出**：更新后的购物车数据，包含总价和菜品清单。

**功能点 4：订单提交**

* **描述**：用户确认购物车中的菜品后，填写送餐地址和备注信息并提交订单。
* **输入**：用户 ID，购物车数据，送餐地址，备注信息。
* **输出**：生成订单号，并跳转至支付页面。

**功能点 5：实时订单跟踪**

* **描述**：用户可在订单详情页查看当前订单状态（如“已接单”、“配送中”、“已完成”）。
* **输入**：订单号。
* **输出**：实时更新的订单状态信息，可能附带配送员位置（通过 GPS 获取）。

**功能点 6：支付处理**

* **描述**：用户通过第三方支付平台完成订单付款。
* **输入**：订单金额，支付方式（如信用卡、电子钱包）。
* **输出**：支付成功或失败的通知，更新订单状态为“已支付”。

**4. 外部接口**

**接口 1：支付网关**

* **描述**：系统需与第三方支付网关集成，支持多种支付方式（如支付宝、微信支付）。
* **协议**：HTTPS
* **数据格式**：JSON

**接口 2：地图服务**

* **描述**：系统通过调用第三方地图 API 获取用户位置，并计算餐厅距离。
* **协议**：RESTful API
* **数据格式**：JSON

**5. 非功能需求**

**5.1 性能指标**

* 页面加载时间不超过 2 秒。
* 订单提交响应时间不超过 1 秒。
* 支持同时在线用户数不少于 10,000 人。

**5.2 质量属性**

* **可靠性**：系统需保证 99.9% 的正常运行时间。
* **安全性**：用户数据加密存储，支付过程遵循 PCI DSS 标准。
* **可用性**：即使在网络不稳定的情况下，核心功能仍可使用。

**6. 完整的用例描述**

**功能点 1：餐厅浏览**

**用例名称: 餐厅浏览**

**用例编号: UC-001**

**参与者: 用户**

**前置条件: 用户已打开应用程序，并授权获取当前位置信息。**

**后置条件: 显示距离最近的餐厅列表，用户可选择某个餐厅查看详情。**

**主事件流:**

**1. 用户启动应用并进入餐厅浏览界面。**

**2. 系统通过用户的当前位置坐标（经度、纬度）查询附近餐厅信息。**

**3. 系统以地图或列表形式展示餐厅信息，包括每条记录的餐厅图片、名称、评分和推荐菜品。**

**异常事件流:**

**- 如果无法获取用户位置信息，系统提示用户手动输入地址或重新授权位置权限。**

**- 如果附近无可用餐厅数据，系统显示“暂无餐厅信息，请稍后再试”。**

**---**

**功能点 2：菜品选择**

**用例名称: 菜品选择**

**用例编号: UC-002**

**参与者: 用户**

**前置条件: 用户已浏览餐厅列表并点击某个餐厅进入详情页。**

**后置条件: 用户成功将菜品加入购物车，购物车更新显示总价和所选菜品清单。**

**主事件流:**

**1. 用户点击某个餐厅，系统显示该餐厅菜单。**

**2. 用户选择菜品并指定数量，点击“加入购物车”按钮。**

**3. 系统接收餐厅 ID、菜品 ID 和数量，更新购物车数据。**

**4. 系统显示更新后的购物车详情，包括总价和所选菜品清单。**

**异常事件流:**

**- 如果菜品库存不足，系统提示“当前菜品库存不足，请选择其他菜品”。**

**- 如果网络请求失败，系统提示“操作失败，请检查网络连接后重试”。**

**---**

**功能点 3：购物车管理**

**用例名称: 购物车管理**

**用例编号: UC-003**

**参与者: 用户**

**前置条件: 用户已将至少一个菜品加入购物车。**

**后置条件: 购物车数据更新，显示最新的总价和菜品清单。**

**主事件流:**

**1. 用户进入购物车页面，查看已选菜品清单和总价。**

**2. 用户对某菜品执行操作（增加、减少或删除）。**

**3. 系统根据操作类型更新购物车数据。**

**4. 系统显示更新后的购物车详情，包括总价和菜品清单。**

**异常事件流:**

**- 如果用户尝试删除最后一个菜品，系统提示“购物车为空，请添加菜品”。**

**- 如果网络请求失败，系统提示“操作失败，请检查网络连接后重试”。**

**---**

**功能点 4：订单提交**

**用例名称: 订单提交**

**用例编号: UC-004**

**参与者: 用户**

**前置条件: 用户已确认购物车中的菜品，并准备提交订单。**

**后置条件: 系统生成订单号，并跳转至支付页面。**

**主事件流:**

**1. 用户点击“提交订单”按钮。**

**2. 系统要求用户填写送餐地址和备注信息。**

**3. 用户确认信息后提交订单。**

**4. 系统生成订单号，并跳转至支付页面。**

**异常事件流:**

**- 如果用户未填写送餐地址，系统提示“请填写送餐地址”。**

**- 如果生成订单失败，系统提示“订单提交失败，请稍后重试”。**

**---**

**功能点 5：实时订单跟踪**

**用例名称: 实时订单跟踪**

**用例编号: UC-005**

**参与者: 用户**

**前置条件: 用户已成功提交订单并进入订单详情页。**

**后置条件: 用户实时了解订单状态及配送员位置（如有）。**

**主事件流:**

**1. 用户进入订单详情页，查看当前订单状态。**

**2. 系统定期更新订单状态（如“已接单”、“配送中”、“已完成”）。**

**3. 如果订单处于“配送中”，系统显示配送员位置（通过 GPS 获取）。**

**异常事件流:**

**- 如果无法获取订单状态更新，系统提示“暂时无法获取订单状态，请稍后刷新”。**

**- 如果配送员位置信息不可用，系统提示“配送员位置信息暂不可用”。**

**---**

**功能点 6：支付处理**

**用例名称: 支付处理**

**用例编号: UC-006**

**参与者: 用户**

**前置条件: 用户已跳转至支付页面并选择支付方式。**

**后置条件: 支付成功或失败的通知显示，订单状态更新为“已支付”或保持不变。**

**主事件流:**

**1. 用户在支付页面选择支付方式（如信用卡、电子钱包）。**

**2. 用户确认支付金额并完成支付操作。**

**3. 系统调用第三方支付平台接口处理支付请求。**

**4. 如果支付成功，系统更新订单状态为“已支付”，并显示“支付成功”通知。**

**异常事件流:**

**- 如果支付失败，系统提示“支付失败，请检查支付信息后重试”。**

**- 如果网络请求超时，系统提示“支付请求超时，请稍后重试”。**