****

美食拍

需求规约

**Version 1.0**

**2020.6**

小组成员：

**1651290** 夏宇宁，

**1754188** 谢尚汝，

所在院系：软件学院

学科专业：软件工程

指导老师：曹布阳

1. 介绍

本产品为面向美食及其所涉及对象的垂直分享社交型微信小程序，为有相同兴趣的用户提供先进科学以及高效的沟通，并且为此类用户提供安全便捷的网络周边环境。

1. 整体描述

2.1项目特点

(1)随着人民生活水平的改善，越来越多的人开始关注生活质量，”吃得好”变成了 大多数人的追求，产品的潜在客户数量巨大，技术要求简单发展前景广阔。

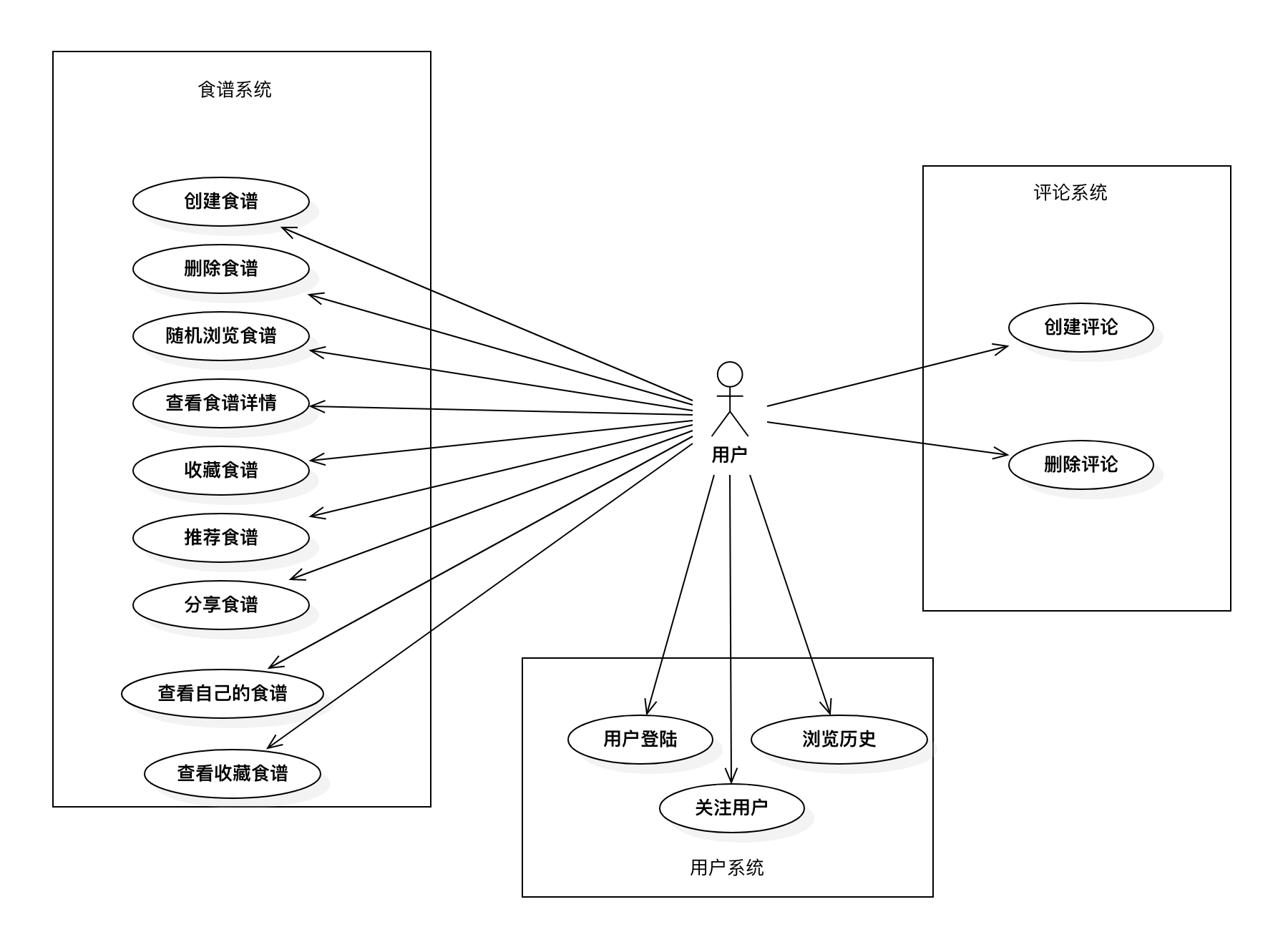
(2)当下媒体社交流行，以美食为契机分享的同时，打开社交功能，适应潮流。

(3)微信的高覆盖率，其小程序的潜在用户也十分巨大，同时开发难度并不太高。

1. 系统与其他系统的接口
2. 主要的功能需求描述

4.1文字描述

4.2用例模型



|  |
| --- |
| 用例: 登录 |
| ID：UC1 |
| 参与者： 用户 |
| 前置条件: 用户拥有微信账号 |
| **基本事件流:**  1. 用户点击“登录”按钮  2. 用户调用微信进行快捷登陆 |
| 后置条件：用户可以使用系统提供的所有功能 |

|  |
| --- |
| 用例: 关注用户 |
| ID：UC2 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号 |
| **基本事件流:**   1. 用户点击他人头像查看详情 2. 用户点击关注加入关注列表 3. 用户可以在个人关注列表界面看到关注的人 |
| 后置条件：用户可以从关注列表访问关注的人 |

|  |
| --- |
| 用例: 查看浏览历史 |
| ID：UC3 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号 |
| **基本事件流:**   1. 用户点击浏览历史按钮 2. 显示用户浏览过的历史 |
| 后置条件：用户可以查看历史食谱详情 |

|  |
| --- |
| 用例: 创建评论 |
| ID：UC4 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号、用户展开食谱详情界面 |
| **基本事件流:**   1. 用户点击评论按钮 2. 用户在弹出的文本编辑界面编辑评论 3. 用户点击发送按钮发表评论 |
| 后置条件：用户可以查看自己发布的评论 |

|  |
| --- |
| 用例: 删除评论 |
| ID：UC5 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号、用户已发表某条评论 |
| **基本事件流:**   1. 用户找到自己发表的评论 2. 点击删除按钮删除评论 |
| 后置条件：用户评论被移除 |

|  |
| --- |
| 用例: 创建食谱 |
| ID：UC6 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号 |
| **基本事件流:**   1. 用户点击“创建食谱”按钮，进入食谱创建界面 2. 用户上传食谱图片并编辑文字 3. 点击“我要发布”按钮发布菜谱 |
| 后置条件：用户成功创建一条食谱 |

|  |
| --- |
| 用例: 删除食谱 |
| ID：UC7 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号、用户已创建过至少一条食谱 |
| **基本事件流:**   1. 用户点击待删除食谱，进入目的食谱详情界面 2. 用户点击删除食谱按钮 3. 系统弹出“是否确认删除”界面 4. 用户在弹出的“是否确认删除”界面选择是，则系统删除该条食谱；选择否，则系统不删除该条食谱 |
| 后置条件：用户成功删除一条食谱、用户成功取消删除一条食谱 |

|  |
| --- |
| 用例: 随机浏览食谱 |
| ID：UC8 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号、用户自己或其他用户已创建过至少一条食谱 |
| **基本事件流:**   1. 用户进入“主页”页面 2. 系统检索并以信息流方式显示系统所拥有的所有食谱 3. 用户可以上下滑动信息流浏览系统中的食谱 |
| 后置条件：用户可以在“主页”查看系统中所有用户已发布的食谱信息流 |

|  |
| --- |
| 用例: 查看食谱详情 |
| ID：UC9 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号、用户自己或其他用户已创建过至少一条食谱 |
| **基本事件流:**   1. 用户进入“主页”页面 2. 系统检索并以信息流方式显示系统所拥有的所有食谱 3. 用户可以上下滑动信息流浏览系统中的食谱 4. 用户可以选择他（她）感兴趣的食谱，查看食谱详情 |
| 后置条件：用户可以查看系统中所有用户已发布的食谱详情 |

|  |
| --- |
| 用例: 收藏食谱 |
| ID：UC10 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号、用户自己或其他用户已创建过至少一条食谱 |
| **基本事件流:**   1. 用户点击待收藏食谱，进入目的食谱详情界面 2. 若该食谱未被用户收藏过，用户点击收藏食谱按钮，系统将该条食谱加入用户收藏列表；若已被收藏过，用户点击收藏食谱按钮，系统将该条食谱移出用户收藏列表 |
| 后置条件：用户成功收藏一条食谱、用户成功取消收藏一条列表 |

|  |
| --- |
| 用例: 推荐食谱 |
| ID：UC11 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号、用户自己或其他用户已创建过至少一条食谱 |
| **基本事件流:**   1. 用户点击待推荐食谱，进入目的食谱详情界面 2. 用户点击“推荐”按钮 |
| 后置条件：用户成功推荐一条食谱 |

|  |
| --- |
| 用例: 分享食谱 |
| ID：UC12 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号、用户自己或其他用户已创建过至少一条食谱 |
| **基本事件流:**   1. 用户点击待分享食谱，进入目的食谱详情界面 2. 用户点击“分享”按钮 3. 系统弹出分享界面 4. 用户在分享界面的文字输入框中输入分享理由（可选） 5. 用户点击“确认分享”按钮 |
| 后置条件：用户成功分享一条食谱 |

|  |
| --- |
| 用例: 查看自己的食谱 |
| ID：UC12 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号 |
| **基本事件流:**   1. 用户点击已发布的食谱，进入相应界面 2. 显示用户已发布的食谱 |
| 后置条件：用户可查看食谱详情 |

|  |
| --- |
| 用例: 查看收藏食谱 |
| ID：UC12 |
| 参与者： 用户 |
| 前置条件: 用户拥有登录的账号 |
| **基本事件流:**   1. 用户点击已收藏的食谱，进入相应界面 2. 显示已收藏的食谱 |
| 后置条件：用户可查看食谱详情 |

1. 功能需求描述

6.其他需求等

6.1 易用性

界面风格统一，其友好的用户交互设计，让用户更方便地获取自己想要的信息。

尽量减少用户的学习成本，方便操作，操作流程合理，操作完成时有统一规范的提示信息。

支持没有计算机使用经验、计算机使用经验较少及有较多计算机使用经验的用户均能方便地使用本系统

系统具有一定的容错和抗干扰能力，在非硬件故障或非通讯故障时，系统能够保证正常运行，并有足够的提示信息帮助用户有效正确地完成任务。

系统具有一定的灵活性，可以提示用户如何操作

系统的通用性、操作简单易学，实用性强

网站功能精简，不含冗余恼人的其它功能。

6.2 可维护性

可维护性是指在不影响系统其他部分的情况下修改现有系统功能中问题或缺陷的能⼒。开发⼈员创建和设计系统架构时，为了提高系统的可维护性，必须考虑以下几个方⾯的要素：低耦合、高内聚合系统。

本系统将采用严格的软件工程的规范进行开发，并采用良好的设计模式保证系统各模块之间的低耦合 及模块之间的高内聚。

本系统的所有代码将会被详细注释，对于系统所有代码，我们会生成详尽的技术文档。对于系统开发过程可能出现的报错，我们将以文档的方式详细罗列报错码及对应的报错信息。

根据兼职行业的现状，本系统不仅要考虑现有的数据集成和功能集成，还要充分考虑因时新需求导致的对现有功能的新要求。

因而系统的设计方案应易于理解

相关的文档资料应清晰、完整、一致

同时要有良好的功能、数据可扩展性以及对环境适应能力

6.3 可扩展性

系统建成后，应在现行系统上不需要做⼤的改动或不影响整个系统结构，就可以增加功能模块，这就必须在系统设计时留有接口，使其具有可扩展性和维护性，这样就方便在后期的维护过程中根据用户的需求添加相应的功能，同时也不会影响系统其他功能模块的正常运行。

除此之外，还应当保证所有文档说明的详细、完整度，使其他接收者或新加入开发者能够快速地了解系统的情况，并且正确地在原有基础上进行更新和加强。

6.4 稳定性

网站能够正常完成所有的功能需求，且保证服务器长时间运行无障碍。故障修复时间一般控制在3小时内，除非出现恶性攻击、病毒破坏、系统资源不足、硬件故障、操作系统崩溃网络中断等情况，不允许出现内存泄露。要求系统可以承受告警风暴和大数据量的冲击，不出现内存溢出等现象。

6.5 灾难恢复

即使黑客入侵随意篡改数据，或者发生灾难造成数据丢失，但由于区块链系统中，任一节点的损坏或者失去都会不影响整个系统的运作，并且可在链上取下当前准确数据，因此对数据恢复尤为容易