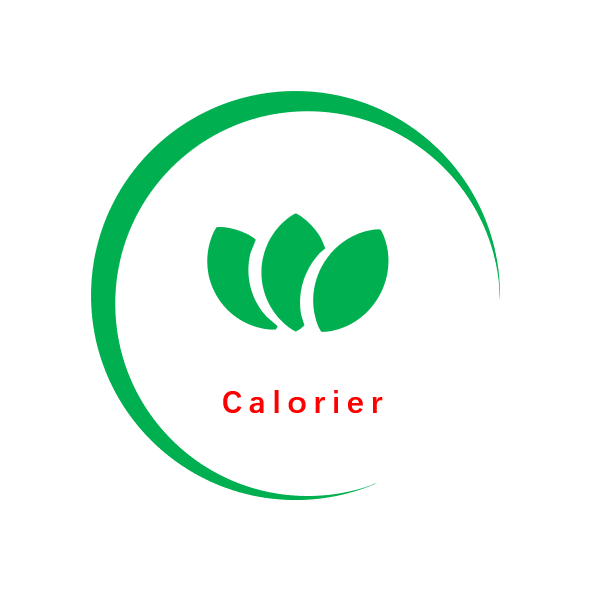
文件编号：Calorier-SWC2020-20200184

受控状态：■受控 □非受控

保密级别：□公司级 □部门级 ■项目级 □普通级

采纳标准：CMMI DEV V1.2





卡路里

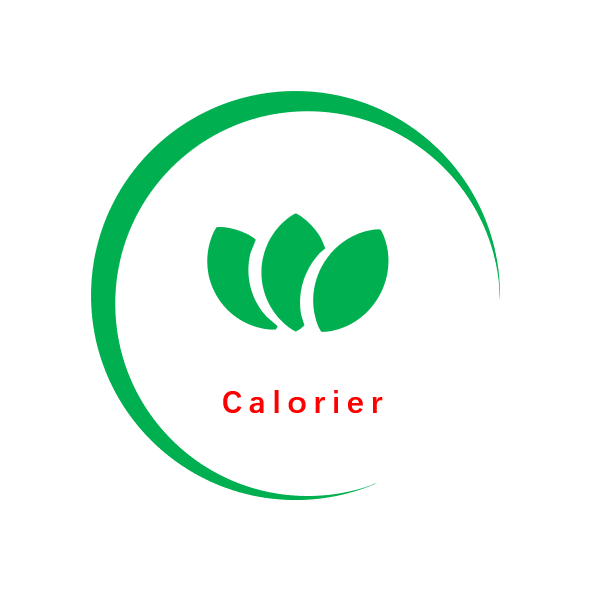
**Calorier**

项目开发文档

**Version 1.4.0**

2019.12.27

**Written by Calorier team**



**All Rights Reserved**

目录

[1 项目概述 1](#_Toc22846313)

[1.1 项目背景 1](#_Toc22846314)

[1.2 项目定位 1](#_Toc22846315)

[1.2.1 应用场景 1](#_Toc22846316)

[1.2.2 目标人群 1](#_Toc22846317)

[1.3 项目方案 1](#_Toc22846318)

[1.4 项目目标 1](#_Toc22846319)

[1.5 项目价值 1](#_Toc22846320)

[2 开发计划 1](#_Toc22846321)

[2.1 最终呈现形式 1](#_Toc22846322)

[2.2 主要功能描述 1](#_Toc22846323)

[2.3 运行环境 1](#_Toc22846324)

[2.4 验收标准 1](#_Toc22846325)

[2.5 关键问题 1](#_Toc22846326)

[2.6 进度安排 1](#_Toc22846327)

[2.7 开发预算 1](#_Toc22846328)

[3 可行性分析 2](#_Toc22846329)

[3.1 技术可行性分析 2](#_Toc22846330)

[3.2 资源可行性分析 2](#_Toc22846331)

[3.3 市场可行性分析 2](#_Toc22846332)

[4 需求分析 2](#_Toc22846333)

[4.1 数据需求 2](#_Toc22846334)

[4.1.1 静态数据 2](#_Toc22846335)

[4.1.2 动态数据 2](#_Toc22846336)

[4.1.3 数据词典 2](#_Toc22846337)

[4.1.4 数据采集 2](#_Toc22846338)

[4.2 功能需求 2](#_Toc22846339)

[4.2.1 \*\*功能模块 2](#_Toc22846340)

[4.3 性能需求 4](#_Toc22846341)

[4.3.1 时间特性 4](#_Toc22846342)

[4.3.2 适应性 4](#_Toc22846343)

[4.4 界面需求 4](#_Toc22846344)

[4.5 接口需求 4](#_Toc22846345)

[4.5.1 硬件接口 4](#_Toc22846346)

[4.5.2 软件接口 4](#_Toc22846347)

[4.6 其他需求 4](#_Toc22846348)

[5 概要设计 4](#_Toc22846349)

[5.1 处理流程 4](#_Toc22846350)

[5.2 总体结构设计 4](#_Toc22846351)

[5.3 功能设计 4](#_Toc22846352)

[5.4 用户界面设计 4](#_Toc22846353)

[5.5 数据结构设计 4](#_Toc22846354)

[5.6 接口设计 4](#_Toc22846355)

[5.6.1 外部接口 4](#_Toc22846356)

[5.6.2 内部接口 4](#_Toc22846357)

[5.7 错误/异常处理设计 5](#_Toc22846358)

[5.7.1 错误/异常输出信息 5](#_Toc22846359)

[5.7.2 错误/异常处理对策 5](#_Toc22846360)

[5.8 系统配置策略 5](#_Toc22846361)

[5.9 系统部署方案 5](#_Toc22846362)

[5.10 其他相关技术与方案 5](#_Toc22846363)

[6 数据库设计 5](#_Toc22846364)

[7 详细设计 5](#_Toc22846365)

[7.1 \*\*功能模块 5](#_Toc22846366)

[7.1.1 功能描述 5](#_Toc22846367)

[7.1.2 性能描述 5](#_Toc22846368)

[7.1.3 输入 5](#_Toc22846369)

[7.1.4 输出 5](#_Toc22846370)

[7.1.5 程序逻辑 5](#_Toc22846371)

[7.1.6 限制条件 5](#_Toc22846372)

记录更改历史

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **更改原因** | **版本** | **作者** | **更改日期** | **备 注** |
| 1 | 初次填写内容 | 1.0.0 | 王炳楠 | 2019.11.27 |  |
| 2 | 完成项目概述 | 1.1.0 | 曹玉琪 | 2019.11.29 | 添加概述部分 |
| 3 | 完成开发计划 | 1.2.0 | 龚慧琳 | 2019.12.02 |  |
| 4 | 完成可行性分析 | 1.3.0 | 龚慧琳 | 2019.12.03 |  |
| 5 | 完成需求分析 | 1.4.0 | 曹玉琪 | 2019.12.10 |  |
| 6 | 完成界面需求 | 1.5.0 | 王炳楠 | 2019.12.11 |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# 项目概述

## 项目背景

随着生活水平的提高，人们对于饮食平衡方面也关注得越来越多，但是人们对于自己每天的摄入却缺少相应的应用进行记录和跟踪，大家知道自己吃的什么，但是对于自己摄入了什么却不是非常清楚。当今时代健康类软件层出不穷，有关于运动的，有关于疾病的，也有关于食物的。然而目前可获取的关于食物的软件却并不完善，大都存在一些问题导致使用不便。这些问题正是我们考虑优化的地方。

首先，目前市面上可以做食物图像识别得到每种食物重量及营养素含量的算法或者软件很少，有的识别算法或者软件对食物重量的识别的准确度只能达到50％左右，所以在用户想通过食物图像识别获取食物重量及营养素含量的时候，误差是可能达到40％以上。食物重量的识别的准确度只能达到50％左右。市面上可获取的此类软件很多都只能手动输入摄取的食物种类和数量，操作起来很不方便。在这个方面，目前可以通过拍照上传的方式对图片进行识别，这样就可以更加准确具体地记录食物。其次，很多软件都只是简单记录却无法给出相应的饮食建议。在这一点上，我们也可以根据用户上传的食物信息，调用相应的API，给出较为中肯的建议。最重要的一点是，越来越来的人不再是想简单记录食物种类和数量，更想通过简单的方式来直观看到自己摄取的热量，而市面上基本没有能够实现此效果地软件。为了实现此效果，目前最优的方案是可以根据内置的图像识别和三维重建技术对体积、密度进行估算，从而给出热量、功效及各种营养素等信息。

我们的项目旨在为用户提供一个方便智能的饮食记录软件。每日饮食看似影响不大，日积月累起来就能发现用户的饮食习惯，我们的软件——Calorier可以及时发现用户饮食不当之处，并给予适当的建议。

## 项目定位

### 应用场景

该应用需要用户在Android手机快应用平台上运行，需要用户连接互联网。用户在用餐的时候，在连接互联网的情况下使用该应用上传自己的食物照片，然后该应用返回食物识别结果，用户根据识别结果，了解食物的类型，以及卡路里，原料等详细数据。用户还可以查看自己的食物历史识别记录，同时系统会将用户的每次识别结果做一个统计，并进行分析，给用户生成获取食物的建议，以帮助用户更好的调配饮食。用户可以把其他使用该应用的用户加为好友，并可向好友分享自己的食物识别结果，以及识别记录分析。

### 目标人群

1. 想要健康减肥的年轻群体
2. 注重养生的中老年群体
3. 想要记录饮食的用户

## 项目方案

1.对于识别食物热量问题，该项目通过拍照上传即可识别食物热量，快速记录热量，控制摄入。

2.对于管理摄入问题，该项目可以根据用户的信息计算出每日的热量预算以及碳水、蛋白质和脂肪的合理比例，用户也可以自定义比例。

3.对于食物识别准确性的问题，该项目可以通过提高提高机器学习分类算法的准确率来提高食物识别地准确性。

## 项目目标

1. 帮助减肥者健康减肥
2. 为养生者提供饮食建议
3. 为广大用户记录日常饮食
4. 给用户提供更健康，更能自主控制的饮食方式

## 项目价值

本项目的成果可以为广大减肥爱好者提供准确有效的减肥建议，为养生爱好者提供健康有效的饮食记录方式，引领新的健康生活方式的潮流。

# 开发计划

## 最终呈现形式

新用户可以注册账号，已有账号的用户可以使用该账号登陆应用。用户在登录成功的情况下，可以拍照或从本地选取食物照片上传，以获取该食物的能量，原料、类型以及描述等信息，用户可以查看自己的所有历史识别记录，以及识别历史数据分析。用户可以将其他用户加为好友，可以与好友分享一次的食物识别结果以及与好友聊天。该应用可以定期向用户生成一段时间的饮食摄入报告，并生成数据分析图标，以及结合合理饮食标准给予用户一定的饮食建议。

## 主要功能描述

1. 注册/登录账号

用户可以注册一个账号，使用手机号或邮箱均可。使用注册成功的账号，可以登录账号。

1. 食物识别

用户拍照或从本地选取食物的图片上传，可以获取食物名称，能量（卡路里/g），原料以及该食物描述信息。

1. 用户食物摄入管理

根据用户上传的结果分析用户摄入的热量以及碳水化合物、蛋白质、维生素和脂肪等的比例，并与合理比例作对照，分析用户个营养摄入情况，给用户建议。用户也可以自己科学设置适合自己的合理比例。

1. 历史数据统计

用户可以查看自己历史识别过的所有食物情况以及各营养元素的摄入情况的统计表。

1. 好友功能

用户可以添加同使用该应用的用户为好友，并可以向该用户聊天以及将自己的食物识别结果以及历史识别数据分享给好友。

1. 饮食分析和建议

记录用户每次的识别情况，每隔特定的时间段生成一份营养摄入小节，并给出用户一份饮食分析以及建议报告。

## 运行环境

本应用需要在内置了快应用平台1050版本及其以上的Android操作系统运行。

服务端要求系统应为 CentOS6.0 及以上或 ubuntu14.04 及以上，需要安装有python3，java EE等软件。

## 验收标准

体验收标准是结合国家标准、软件行业惯例所提出的对于软件系统质量的最低要求，所有交付的软件必须满足本标准的定义。

标准的定义为

1. 测试用例覆盖全部需求且测试用例不通过数的比例< 1.0%;
2. 不存在错误等级为1的错误；
3. 不存在错误等级为2的错误；
4. 错位等级为3的错误数量≤5；
5. 所有提交的错误都已等到更正

验收项目的划分参照GB/T 16260标准，将软件的质量特性分为6大特性和21给子特性。对于该项目，我们选取了最通用的子特性部分进行测试评价：

1. 功能测试：对软件需求规格说明书中的所有功能项进行测试，该详细内容部分参见测试文档

2. 非功能测试：容错测试、安全性测试、性能测试、压力测试、易用性测试、适应性测试,该部分详细内容参见测试文档

3. 安装测试：数据恢复测试、数据接入测试、数据服务测试

4. 文档测试：

1. 操作、维护文档是否齐全，是否包含产品所需的信息和所有的功能模块
2. 用户文档描述的信息是否正确，是否没有歧义和错误的表达
3. 用户文档是否容易理解，是否通过使用适当的术语、图形表示、详细的解释来表达
4. 用户文档对主要功能和关键操作是否提供应用实例
5. 用户文档是否有详细的目录表和索引表

## 关键问题

本项目的关键问题在于如何通过用户上传的食物照片获取到食物的卡路里量，以及如何将用户的历史数据进行可视化的展示。

## 进度安排

2019.11.01-2019.12.31：完成项目文档的初步编写，确定用户需求，完成需求文档等相关文档。完成食物识别模型的训练；完成数据库的搭建和设计；完成界面原型的设计以及项目总框架的设计。

2020.01.06-2020.02.13：完成复赛所需文档的编写。对第一阶段的工作进行汇总和评价；完成各个模块的代码，完成模块的单元测试和集成测试。

2020.3.30-2020.5.11：完成相应文档的编写。整合各个模块，挖成系统测试和验收测试。

## 开发预算

阿里云服务器：

# 可行性分析

## 技术可行性分析

本项目的技术难点主要在于通过图片识别出食物。而深度学习在图像识别中有着天然的优势。我们采用深度学习的方法来进行食物识别。在用神经网络做识别需要大量的数据集进行训练，所幸，由于机器学习如火如荼地发展，互联网上有十分多的相关数据集，例如food-101可以支撑我们来训练模型。

此外，也有十分多开源的优秀项目提供了食物识别的可行性办法。在2017年，有相关科研人员使用基于Google InceptionV3的模型对food-101数据集进行了训练，并且取得了不错的成果，以10种作物为例，以最常见的预测类别为例，可以达到86.97%的Top-1精度和97.42%的Top-5精度。同年，麻省理工学院发布了Recipe1M+数据集，并且同时发了了Pic2Recipe系统，可用通过单张图片推断出事物的菜谱。

## 资源可行性分析

有大量的开源食物识别数据集如food-101，Recipe1M+作为模型训练的支撑。同时有食物-卡路里的数据集，基本覆盖日常所有食物。

## 市场可行性分析

随着人民的生活水平逐渐提高，家家户户都有智能手机，只要在内置了快应用框架的安卓系统平台就可以安装此软件，因此用户安装此软件的成本较低。而生活水平提高其中一方面就在用户有了更多的饮食选择以及可以有更健康的饮食方案，但大多数用户却不知道什么目前市场上存在一些有关饮食健康的应用软件，但是却都有着一定的缺陷，有的只能提供饮食标准，却不能记录用户的饮食情况，并且不能给予用户良好的建议。

本软件着重于拍照识别食物，并分析出其名称以及能量值和原料等信息，并给用户良好的饮食建议，市场上有这样的需求，实现这样需求的却很少很少。

# 需求分析

## 数据需求

### 静态数据

静态数据作为快应用的一部分直接保存在程序中，静态数据从程序内部分离出来，形成独立的外部数据文件，在本系统中定义的静态数据有如下三类信息：

用户信息

用户ID；用户名；密码；邮箱；手机号码；头像；创建日期

食物信息

食物ID；食物名称；食物信息；食物链接；食物成分；食物卡路里

历史记录信息

历史记录ID；食物ID；用户ID；是否删除；记录时间

### 动态数据

动态输入数据

不同食物的图片

历史记录的年月日

动态输出数据

事物的种类、卡路里、成分等信息

不同年月日的历史记录

### 数据词典

food 食物表存储食物信息

user 用户表存储用户信息

record 历史记录表存储用户上传记录

### 数据采集

在本项目中，用户的数据来自于用户的登录、注册和图片上传等操作，我们通过收集用户的操作来获取用户的数据，而我们用于匹配食物信息的数据来源于网上搜索的api。

## 功能需求

### \*\*功能模块

表 核心功能模块描述

|  |  |  |  |
| --- | --- | --- | --- |
| **功能模块** | **功能** | **功能描述** | **优先级** |
| **用户上传图片获取食物信息** | 图像识别 | 识别图片中的食物种类和数量 | 高 |
| 图片存储 | 将用户上传图片存储到服务器端 | 高 |
| **食物历史记录** | 获取历史记录 | 获取不同年月日的历史记录 | 高 |
| 获取历史食物 | 获取某一时期上传的食物信息 | 高 |
| **历史数据可视化** | 将卡路里数据可视化 | 将历史记录食物卡路里数据画成图片 | 高 |
| **推荐与建议功能** | 推荐食物 | 根据用户的历史记录为用户推荐适当的食物 | 中 |
| 饮食建议 | 根据用户的历史记录为用户提供饮食建议 | 中 |
| **分享功能** | 分享分析结果 | 将系统分析出的结果分享给其它用户 | 低 |

表 \*\*\*\*用例规约

|  |  |
| --- | --- |
| 用例名称 | 实物图片识别 |
| 功能简述 | 根据用户上传的图片识别出对应的食物信息返回给用户 |
| 用例编号 | W001 |
| 执行者 | 用户 |
| 前置条件 | 用户成果登录，用户成功上传一张图片 |
| 后置条件 | 用户得到返回的食物信息且图片存储到服务器中 |
| 涉众利益 | 用户希望上传图片后能够较快的得到食物信息，且能够保存到历史记录中。 |
| 基本路径 | 1. 用户登录系统 2. 用户选取或拍摄一张图片 3. 用户上传图片 4. 系统对图片进行识别 5. 系统返回给用户识别结果 6. 食物信息记录在历史记录中 |
| 扩展路径 | 1. 服务器压力大，提示用户稍后尝试，保存用户上传的图片，在服务器恢复后进行图片识别，并返回结果。 2. 用户上传的图片并不包含食物，提示用户未检测到食物信息。 |
| 字段列表 | 食物信息=上传者+上传时间 |
| 设计规则 | 用户应在正常的网络环境下进行图片上传 |
| 未解决的问题 | 受到服务器限制，用户无法快速得到食物识别结果 |
| 备注 | 无 |

## 性能需求

### 时间特性

系统应具有快速响应的特性，用户打开界面和提交事务的平均响应时间应低于1.5 秒。用户进行上传图片操作的数据处理时间应低于 5 秒。（响应）

### 适应性

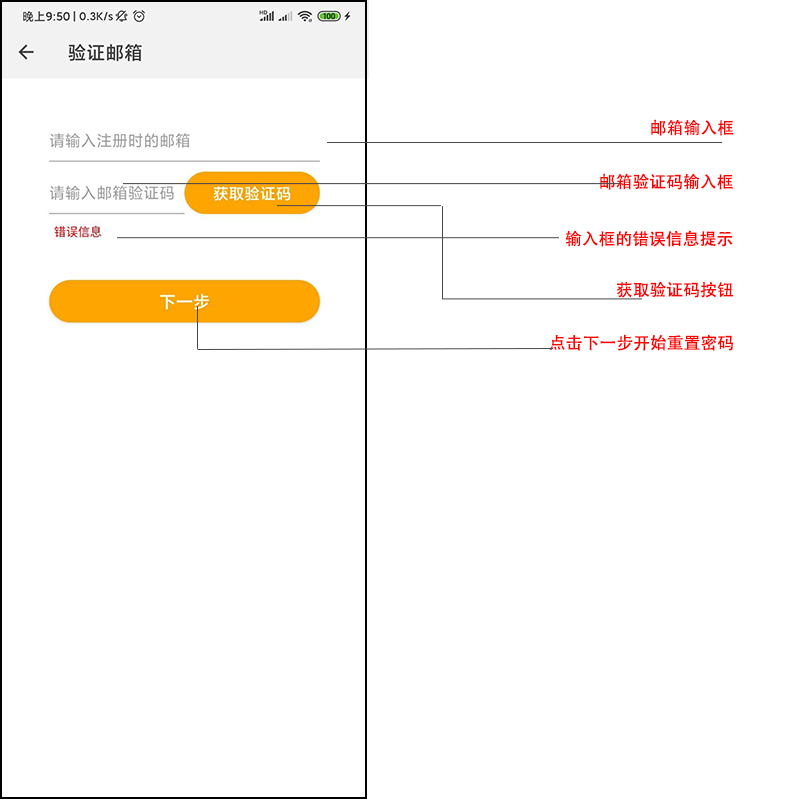
在操作方式、运行环境、软件接口或开发计划等发生变化时，应具有适应能力。

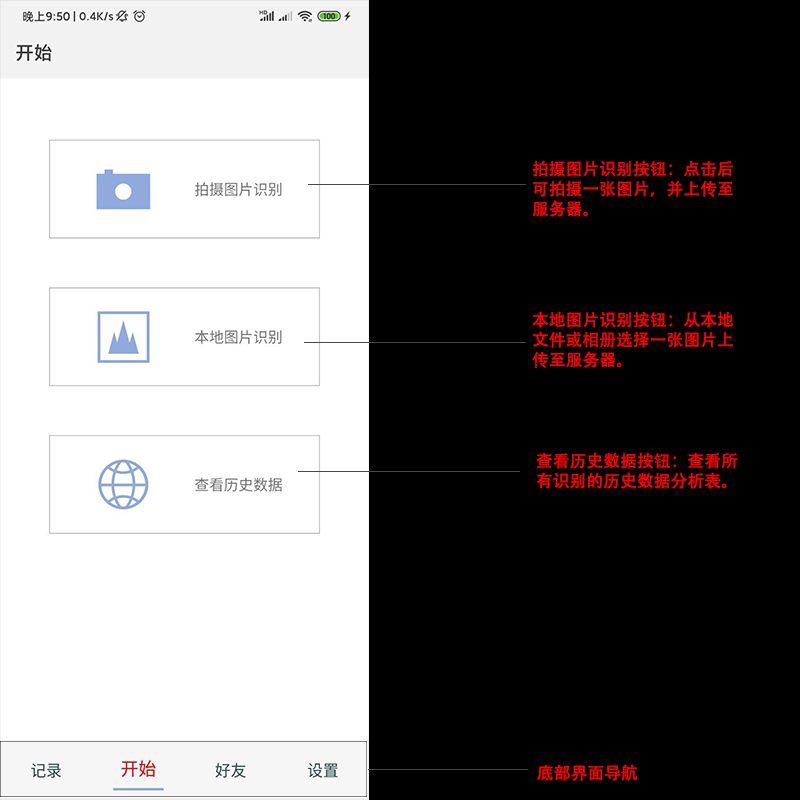
## 界面需求

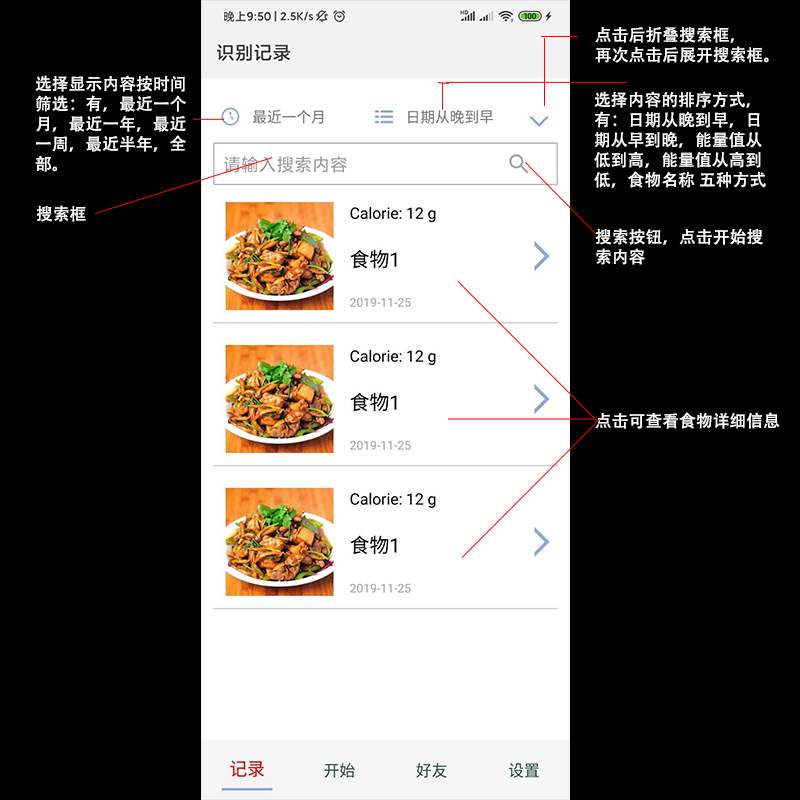
1. 可以适应常用的手机屏幕尺寸；
2. 页面风格采用当前流行的简化风格、扁平化的设计；
3. 把握界面之间的逻辑关系，将主要的内容集中在2、3个页面中，虽然次级界面有助于我们把握逻辑关系，但是过多的页面跳转会使用户感到焦虑；
4. 界面的配色符合快应用的定位；
5. 应用的图标在所有页面都处于同一位置。
6. 用户所需的所有数据内容均按先后次序合理显示。
7. 所有的重要选项都要在主页显示。
8. 重要条目要始终显示。
9. 重要条目要显示在页面的顶端中间位置。
10. 必要的信息要一直显示。
11. 界面原型图



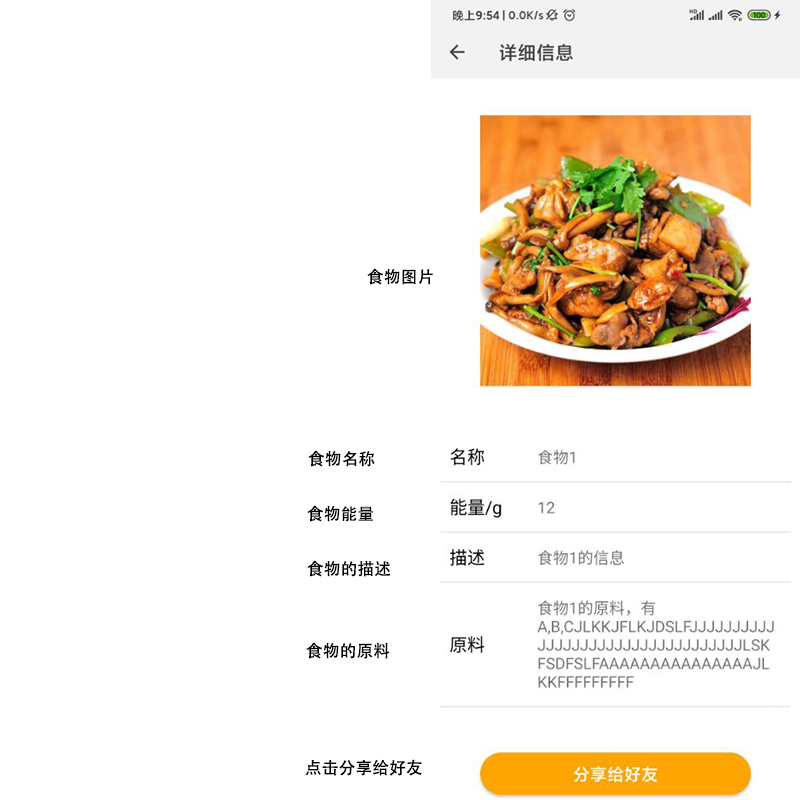


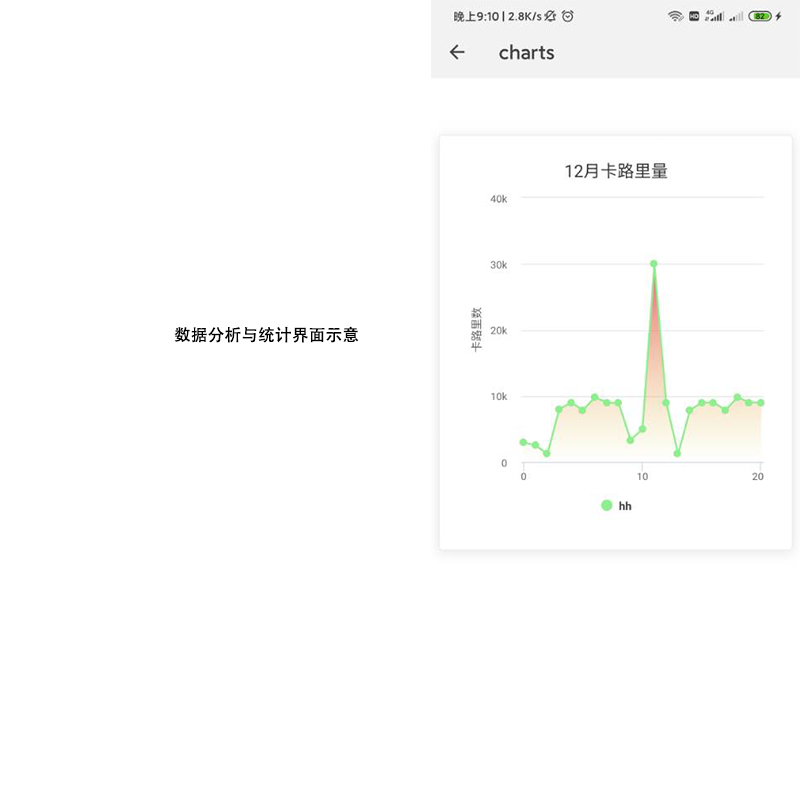
















## 接口需求

### 硬件接口

在支持快应用的手机机型上流畅运行。

### 软件接口

服务端要求系统应为 CentOS6.0 及以上或 ubuntu14.04 及以上，需要安装有python3 等软件。

客户端系统应能够支持快应用的运行。

## 其他需求

1. 可使用性

为提升可使用性，将数据预处理过程封装成相应工具包，使其可以重复利用，可以提供给第三方使用。以类的方式搭建神经网络，使其可以较为方便地作为新网络的部件。

1. 可维护性

为了获得良好的可维护性，严格遵守代码的命名规范，尽量少使用缩写，在关键代码部分提供注释，提升代码的可读性。以面向对象的方式组织代码，提升代码可读性。

1. 安全性

用户对个人信息有安全性要求。只有授权的用户才能动用和修改信息，而且必须防止信息的非法、非授权的泄露。为了保障用户的密码安全，将密码通过加密算法进行加密。

# 概要设计

## 处理流程

## 总体结构设计

## 功能设计

## 用户界面设计

## 数据结构设计

## 接口设计

### 外部接口

### 内部接口

## 错误/异常处理设计

### 错误/异常输出信息

### 错误/异常处理对策

## 系统配置策略

## 系统部署方案

## 其他相关技术与方案

# 数据库设计

# 详细设计

## \*\*功能模块

### 功能描述

### 性能描述

### 输入

### 输出

### 程序逻辑

### 限制条件