**Fooder**

**Software Design Document**

**Prepared by:**  **FOD Project Management Solution, Inc.**

**Issue date:**  **9 November 2019**

**Version:** **1.0**

**Revision Control**

*This section serves to control the development and distribution of revisions to the project plan. It should be used together with a change management process and a document management system. It is recommended that changes to the project plan be documented only by adding appendices to the original project charter. This will keep an accurate history of the original document that was first approved.*

| **Revision Number** | **Date** | **Description of Changes** | **Author/Editor** | **Communication of Change** |
| --- | --- | --- | --- | --- |
| 1.0 | 9-nov-19 | Initial draft | FOD |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**TABLE OF CONTENTS**

[1. Introduction 4](#_Toc25563720)

[1.1. Purpose 4](#_Toc25563721)

[2. Assumptions and Constraints 4](#_Toc25563722)

[2.1. Assumptions and Constraints 4](#_Toc25563723)

[3. Class Diagram 4](#_Toc25563724)

[4. System Setup 4](#_Toc25563725)

[4.1. Training Setup 4](#_Toc25563726)

[4.2. Testing setup 5](#_Toc25563727)

[4.3. Fooder System 5](#_Toc25563728)

[5. System Design 6](#_Toc25563729)

[5.1. Overview of the Fooder - Vision Based Measurement (VBM) System 6](#_Toc25563730)

[5.1.1. Pre-Processing 7](#_Toc25563731)

[5.1.2. Image Analysis: Food Portion Segmentation 7](#_Toc25563732)

[5.1.3. Measurand Identification: Food Recognition 7](#_Toc25563733)

[5.1.4. Measurement: Food Calorie Measurement 7](#_Toc25563734)

[5.2. Image Processing Component 7](#_Toc25563735)

[5.2.1. Using Distance Estimation and Deep Learning to Simplify Calibration 8](#_Toc25563736)

[5.2.2. Selective Search followed by Region Mining 9](#_Toc25563737)

[5.3. Artificial Intelligence Component 10](#_Toc25563738)

[5.4. Cloud Computing Component 10](#_Toc25563739)

[5.5. UI Component 11](#_Toc25563740)

[6. UI Design 13](#_Toc25563741)

[7. References 15](#_Toc25563742)

# Introduction

## Purpose

The purpose of this document is to introduce the design of the principal components of Fooder mobile application: the environment setup, the architecture, the computing techniques used, the UI and their functionalities and interactions.

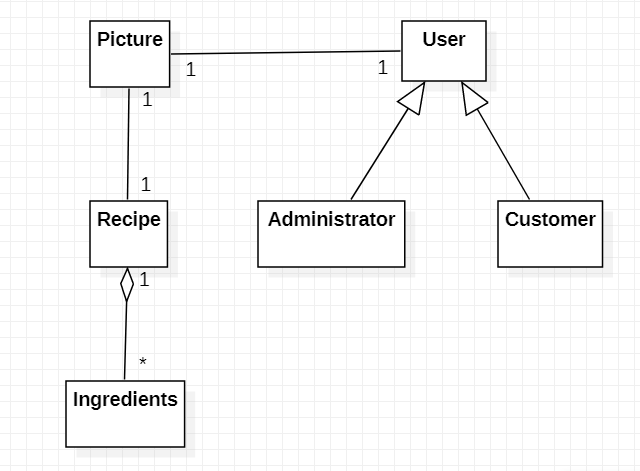
# Assumptions and Constraints

## Assumptions and Constraints

The app must achieve high accuracy in both food identification and calorie calculation. It has to be easy to use and pleasant. Moreover, the app must detect each food ingredient that is visually detectable. It must also use auto-calibration techniques, it should not use external objects in the image. Furthermore, the app must be highly scalable, it must use Amazon Cloud. It must also use Artificial Intelligence (AI) for food detection. The AI training should be incremental and no re-train itself when every new picture is uploaded. The app should run on both Android and iOS.

# Class Diagram

This class diagram represents a high-level view of the system.



# System Setup

## Training Setup

For GPU processing:

* + Amazon EC2 cloud instance g2.2xlarge
  + CUDA SKD version 6.5
  + Ubuntu Server 14.04 LTS (HVM)
  + Python 2.7, Numpy, Scipy

## Testing setup

For testing the deep learning models:

* + Multiple Amazon cloud instances t2.micro
  + HDF5
  + OpenCV

## Fooder System

* + 1 central cloud broker Amazon t2.medium
  + 3 other cloud instances Amazon t2.micro
  + Multiple mobile phones:
    - Android x86 operating system (Android 4.4.2 KitKat)
    - Runs the Fooder application

As mentioned in the Assumptions and Constraints section, the system must be highly scalable, so different Amazon clouds instances are used throughout the system.

We also used CUDA which is a parallel computing platform and programming model developed by Nvidia for general computing on its own GPUs (graphics processing units). It helps to accelerate deep learning.

Ubuntu is the operating system.

Python 2.7 is the programming language. Numpy and Scipy are Python packages for scientific computing.

HDF5 is a versatile data model that can represent very complex data objects and a wide variety of metadata. It is compatible with Python.

OpenCV is used for image processing.

More details of Fooder System are depicted in Figure 1: Architecture of the Fooder System.

# System Design

## Overview of the Fooder - Vision Based Measurement (VBM) System

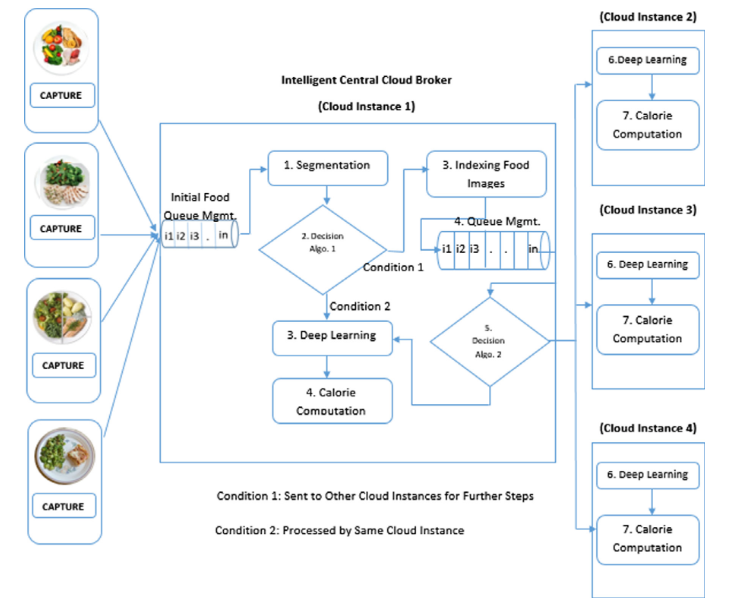


Fig. 1: Architecture of the Fooder System [1]

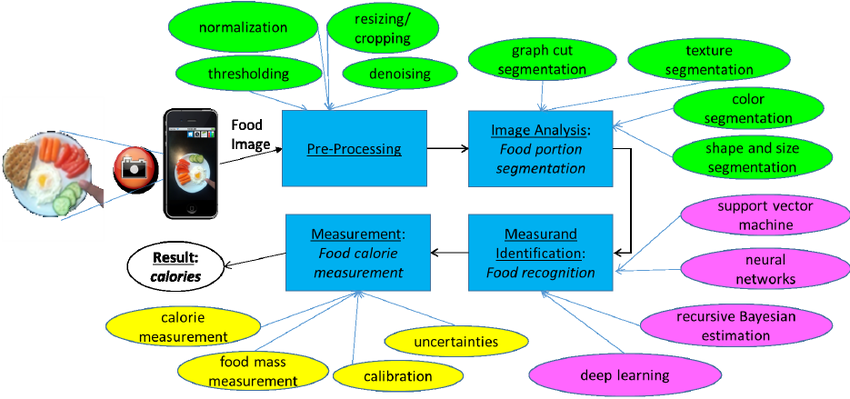


Fig. 2: Various stages of VBM calories measurement. Left to right: food image is acquired by a smartphone, and is fed to image processing (green), computational intelligence (violet), and measurement (yellow) operations. [2]

As shown in the figure above, the VBM system architecture is divided into four main stages: Pre-processing, Image Analysis: Food Portion Segmentation, Measurand Identification: Food Recognition and Measurement: Food Calorie Measurement.

### Pre-Processing

In this stage, any noise and blurs of the food image captured by the user are removed in order to obtain analyzable image for the next step. Image processing methods such as denoising, normalization, thresholding, cropping and resizing are performed here.

### Image Analysis: Food Portion Segmentation

This step determines the boundary of the food portion by using segmentation methods such as Color and Texture Segmentation, K-mean Clustering, Graph Cut Based Segmentation.

### Measurand Identification: Food Recognition

The features of each detected food portion are extracted and classified using deep learning and deep neural networks. Cloud computing was applied in order to process these classification methods and to improve the accuracy and response time.

### Measurement: Food Calorie Measurement

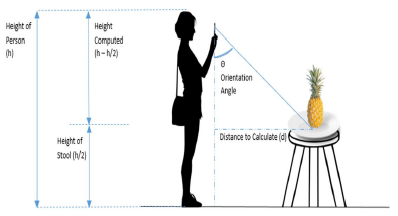
To obtain the calorie of the food portion, we need to determine the mass of the food which is used in combination with the nutritional tables. To estimate the real dimensions of the food from the image, we use the distance captured by the mobile sensors between the smartphone and the food portion for calibration.

## Image Processing Component

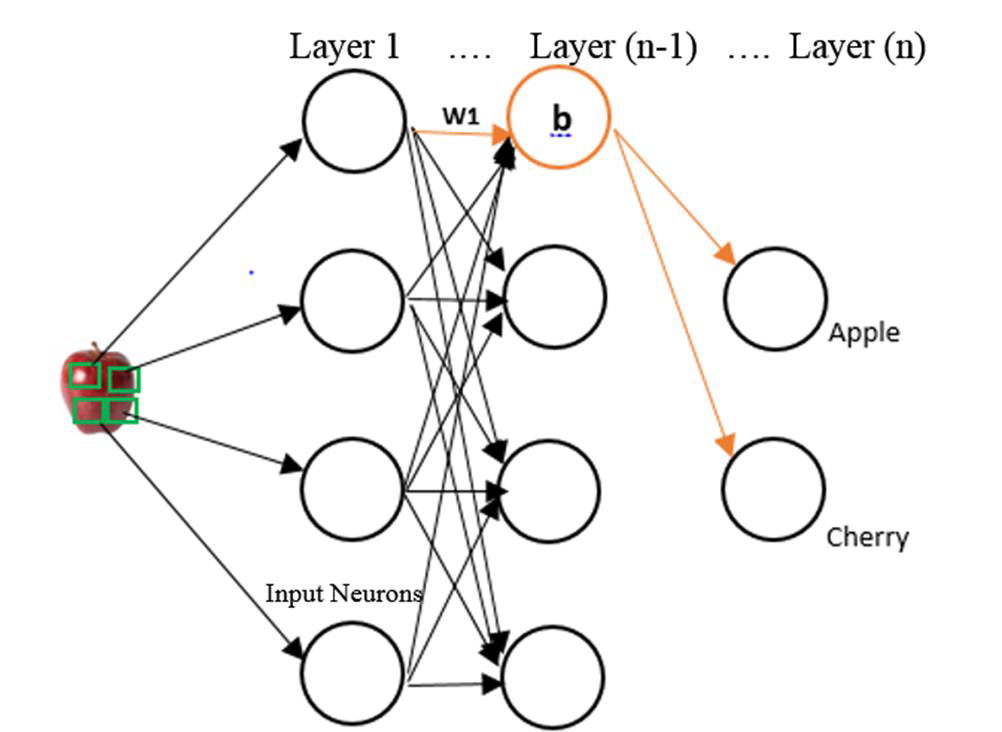
There are two possible alternatives for the project’s image processing component.

### Using Distance Estimation and Deep Learning to Simplify Calibration

The first method uses distance estimation and deep learning to simplify calibration P. Kuhad [4]. Once the user captures the image of the food item on the plate, the image is sent to the cloud for food recognition. The only assumption about the user is that in a normal scenario he would capture the image from the phone’s camera, of a plate which is placed on a stool, at height *L*, which is assumed to be half the height of the user, that is, *L= H/2.* If the user fails to capture the photo from the mentioned distance, the system will smartly recalibrate the block size on the distance measure. Hence calibration will always remain accurate, irrespective of the distance from which the user captures the photo.

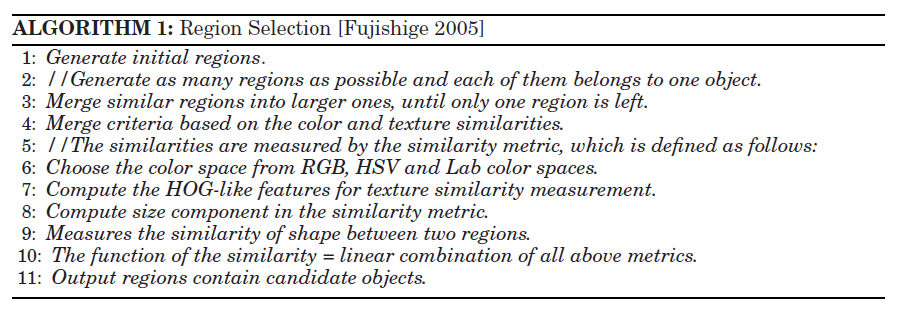


The system makes use of the mobile sensors (accelerometers) in this application to estimate the distance between the food object and the user taking the photo of the food. A deep learning technique is used to enable the system to ascertain the food features based on color, contour, texture and size to classify the food object accurately. Also, due to its efficiency compared with hyperbolic or sigmoid functions, the rectified linear unit (ReLU) is used within the training of the deep convolutional neural network. Furthermore, stochastic gradient descent will be used to tweak the weights and bias to the output closer to the desired output, during the learning phase. The back-propagation algorithm, is the fast way of computing the gradient of the cost function.



### Selective Search followed by Region Mining

For deep learning purposes, we need to process the images by generating region proposals on the images. As discussed in Pouladzadeh [3], we chose Selective Search algorithm because this method creates a hierarchy of small to large regions by grouping super-pixels.

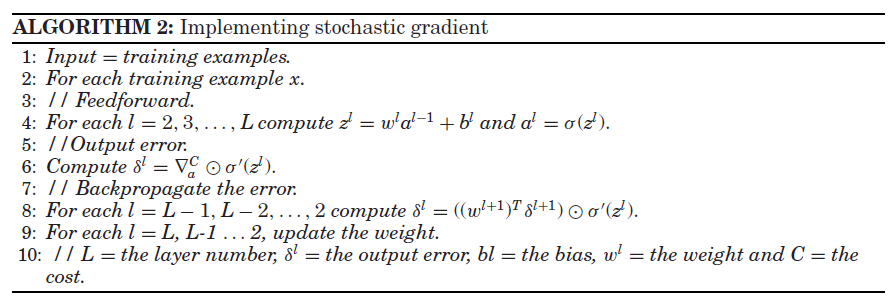


Then, we filtered out the irrelevant regions (i.e., regions that are too small or too large to a certain size or aspect ratio) to decrease the number of proposed regions from 1000 to 150 regions per image.

From these proposed regions, features could be extracted. We use a deep-learning framework called Caffe to obtain the feature matrix of 150 x 4096 for each image, which is used to sort the proposed regions into two types of regions: positive (target object) and hard-negative (background or food belonging to another class). This can be achieved by a greedy algorithm with optimization discussed in Pouladzadeh [3] which finds the minimal region that represents one food category.

## Artificial Intelligence Component

Since food recognition requires highly complex data, deep neural network is used. As opposed to the other machine learning methods, deep neural network learns to extract features itself while training. To achieve this, we use deep belief network (DBN), as mentioned in Pouladzadeh [3]. We first performed a pre-training, which is an unsupervised training without labeling the images. A DBN is like a stack of Restricted Boltzmann Machines (RBM), where the hidden layer of one RBM is the visible layer of the RBM above it. This process is repeated until every layer of network is trained and where the hidden nodes and the edge parameters are determined. However, we do not know how these random training inputs are called. So, we performed a fine tuning of the net with supervised learning by using a very small set of label sample to label these random training inputs. During the supervised training of the DBN, we are constantly calculating a cost value. The cost is typically the difference between the net’s predicted output and the actual output from a set of labelled training data. The cost is then lowered by making slight adjustments to the weights (w) and biases (b) over and over throughout the training process until the lowest possible value is obtained. The training process utilizes the stochastic gradient and its algorithm was proposed by Pouladzadeh [2], which measures the rate at which the cost will change with respect to a change in a weight or a bias.



This process used for training a neural net is called back-propagation or back-prop (calculating the gradient from right to left). This results in a better accuracy and a smaller training time.

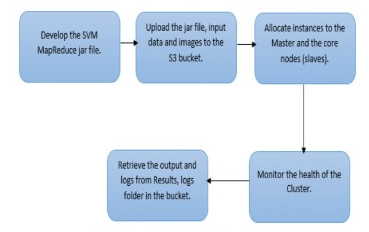
At the end of the DBN training, the desired feature vectors that are used by the classifier are generated. The prediction follows this probabilistic model where p (label |n input image) is the probability of the match with the corresponding label set.

The AI training is incremental, because we would not want to re-train the model every time from the beginning with the whole dataset.

Once the correct type of food is detected, the system can calculate the mass of the food by using mobile sensors to compute the distance between the plate of the food and the user taking the photo of the food. We use this distance to calibrate the dimensions of the food from the image. This auto-calibration method is discussed in P. Kuhad [4]. Consequently, the amount of calorie and nutrition of each food portion can be derived using nutritional tables.

## Cloud Computing Component

It would be practically impossible to compute all the image processing and deep learning on the mobile device. Thus, it is important to be able to run in parallel and schedule all these algorithms and large number of users’ concurrent requests on a cloud server. Additionally, we need a tool that would allow us to process massive amounts of unstructured data in parallel across a distributed cluster of processors or stand-alone computers. To overcome this, as presented in the work by Paddi [1], we choose Elastic MapReduce (Amazon EMR) as a parallel classifier in the cloud to implement SVM on each of the cloud instances. Moreover, we performed the map and reduce based training and testing tasks on multiple Amazon EC2 instances, since it is a firm requirement to use Amazon Web Services (AWS) for this project. As shown in Figure 4.4.1, a large Hadoop cluster was created for master and core instance groups. The master node was assigned to master instance group to manage the cluster and run master components of distributed application. The slave node was then assigned to the core instance group, where data processing takes place. All the input image data, MapReduce job, results and logs are stored in the Amazon S3 bucket. Here is the implementation proposed by Pouladzadeh [3]:



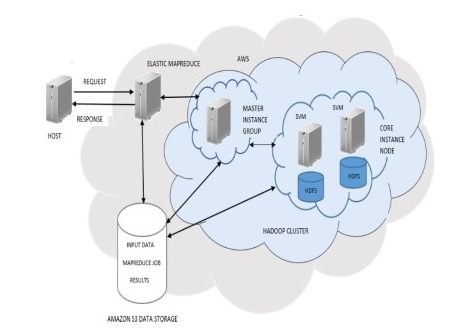


Figure 4.4.1

This also increases the accuracy for food classification in deep learning.

## UI Component

The user takes picture from the two angles: the top view for extracting the food portions and the side view for analyzing the depth of the food to calculate the mass which is needed to calculate the food calories. Then, we asked the user to draw a bounding circle around the plate of the food to separate the background from the food (area of interest).

The android application will be implemented using Android Studio; therefore, the UI will be coded using XML. We are using Android Studio because it is Android’s official IDE, and has all the tools we need to create our application.

The designing on our UI has several principles that it must satisfy. Design guidelines must be followed because good UI allows for a better user experience. We want the interface to be well structured so that related items and unrelated items are grouped together. This allows the design to be more intuitive for the user.

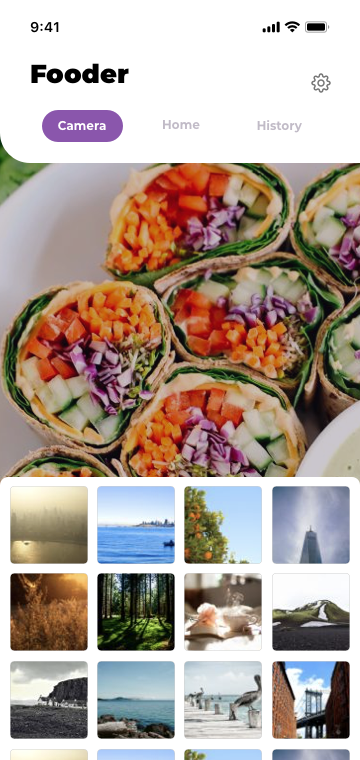
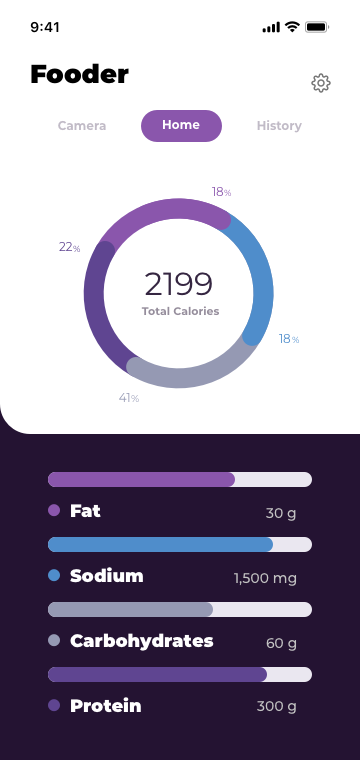
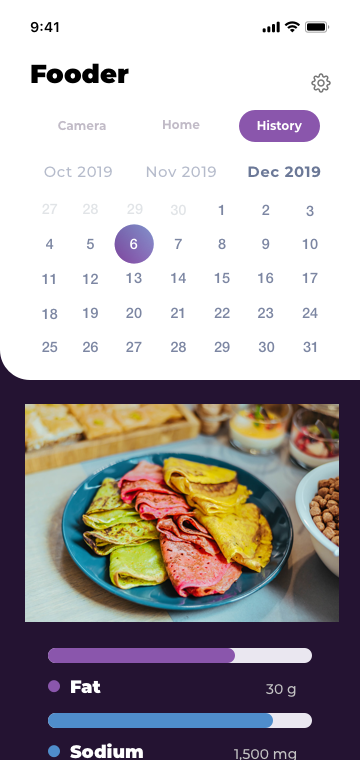
Our UI must also provide good status to the user. The user should always know what has been done, what available steps are possible, and completion. Throughout any process, the current state of the system must be known, and any error messages must be clearly visible with a concise and accurate description.

Simplicity is another principle that we want to have with our interface. Different components in the interface should be well spaced, not cluttered, and easy to follow.

The usage of standard icons such as the cog wheel allows users to experience recognition over recall. Users will be able to associate knowledge from the usage of other applications to have a better understanding of what certain elements in our application do; this is also known as external consistency.

The design shown in the next section of this documents is to be evaluated and tested to ensure that it follows our valued principles.

# UI Design

# References

1. S.V.B. Peddi et al. "An intelligent cloud-based data processing broker for mobile e-health multimedia applications" Future Generation Computer Systems. vol. 66 pp. 71-86 Mar. 2016.
2. Parisa Pouladzadeh, Shervin Shirmohammadi, and Abdulsalam Yassine. 2016. You are what you eat: So,measure what you eat.IEEE Instrum. Meas. Mag.19, 1, 9–15.
3. P. Pouladzadeh S. Shirmohammadi "Mobile Multi-Food Recognition Using Deep Learning" ACM Trans. Multimedia Comput. Commun. Appl., vol. 13, no. 3s, Article 36, Aug. 2017.
4. P. Kuhad A. Yassine S. Shirmohammadi "Using distance estimation and deep learning to simplify calibration in food calorie measurement" Proc. IEEE Int. Conf. Computational Intelligence and Virtual Environments for Meas. Syst. Applications 2015-Jun.
5. P. Pouladzadeh S. Shirmohammadi R. Almaghrabi "Measuring calorie and nutrition from food image" IEEE Trans. Instrum. Meas. vol. 63 no. 8 pp. 1947-1956 Aug. 2014.