

**Campus Meal Ordering System**

**System Requirement Specifications (SRS)**

**By *Team Foodie***

**Lab Group: TS3**

**Date: September 2020**

**Instructors:**

**Bo An**

**Leong Yin Yoke Junie**

**Qiu Wei**

**Wang Rundong**

# **SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**NANYANG TECHNOLOGICAL UNIVERSITY**

[**Problem Statement**](#_2lo5ees95qje) **4**

[**Overview**](#_r20hh7b8822q) **4**

[Background](#_l1kdrerxu8) 4

[Overall Description](#_jvu29smlrqzf) 4

[**Investigation and Analysis Methodology**](#_ool4su1w1zo4) **4**

[System Investigation](#_xb5y9xi9y5yf) 4

[Analysis Methodology](#_biy43c5xtcwd) 5

[Feasibility study and requirements elicitation](#_psxjamqmlhb4) 5

[System analysis and requirements specification](#_yh0grq6urjdj) 5

[Object-oriented design using UML](#_i47virbz1cor) 6

[Prototyping](#_3wa6ds2de66d) 7

[**Constraints**](#_j7xj8rchvjet) **8**

[Scalability](#_yzyi712xnyd2) 8

[Data and function mapping](#_5q6k0ophunlq) 8

[Proprietary hardware and software](#_kmpus2vhbx6g) 8

[Batch updates vs. Real-time updates](#_s503acwstjua) 8

[Project Schedule](#_2coxf2bmsf6j) 8

[**Operational Requirements**](#_lzqc0miva3j7) **8**

[Help Center](#_vy69x9m91pwh) 8

[Application Services and Technical support](#_zfkh41fxsxui) 9

[Administration Features](#_bjyocm30pmip) 9

[System hardware failure and backup](#_zeqcv5nqh35i) 9

[**Functional Requirements**](#_uu6lsgkk4s8d) **9**

[Client Application](#_csrx1sc241jn) 9

[Registration and Login](#_dslj066fwpms) 9

[Browse Promotions](#_hec1m0ibxj62) 10

[Browse Restaurants](#_m0ye0vlnjaqh) 10

[Browse Restaurant](#_b02lhx7c2lx7) 10

[Browse Dish and Add dish to Cart](#_a8d2cqylctga) 11

[Browse Cart and check out](#_wfbh35lusjyp) 11

[Vendor Application](#_6t8y03idkh3l) 11

[Registration and Login](#_yr6zdiymmcxw) 11

[Browser placed orders](#_4y9flkp6l984) 11

[Deliveryman Application](#_iil7oofbasy) 12

[Registration and Login](#_9m85ajhivrir) 12

[Browser orders to pick up](#_k1wskzwt947z) 12

[Browser orders to deliver](#_4k88ckrevsly) 12

[Browser orders delivered](#_xtqolqyq46e0) 12

[**Input Requirements**](#_ckb85k16y38s) **13**

[The Login Form and Account Access](#_96513w4twzkh) 13

[Order Form](#_ry44ohv492t6) 13

[Sorting Preferences](#_jhl2v13kard4) 13

[**Process Requirements**](#_9h1ybyjlpnfx) **13**

[Database Transactions](#_1pb4oxpoih9b) 13

[Data Integrity](#_ebjtmyg2ucwt) 14

[Data Validation](#_472e4yknfxqh) 14

[Performance](#_p33t46233333) 14

[**Output Requirements**](#_cee1e0jtfud4) **14**

[Checkout Form](#_1501ijq8ipjg) 14

[Order Retrieval System](#_o0tq804o87r6) 14

[Reports and Summaries](#_jnqryxg9j8a2) 15

[**Hardware Requirements**](#_f88tk2fp2f9c) **15**

[Client](#_15yacoq3wfjm) 15

[Backend server](#_2d1oe5dzn64a) 15

[**Software Requirements**](#_50ltz2gvgq4r) **15**

[Client Operating Systems](#_a7vd441iywju) 15

[Client Application](#_491av8fch8np) 15

[Database and Storage system](#_wt93t0wewzk5) 15

[Licenses](#_7a5f6xce0xd8) 15

[**Deployment Requirements**](#_pa1b5etk2xq7) **16**

[Deployment on Android mobile platform](#_dwpm12gxnp2k) 16

[Deployment on IOS mobile platform](#_o5xciwic7z9a) 16

# **Problem Statement**

The campus population including students and staff usually have a dense and strict schedule crammed with various activities. Dining is an essential part of one’s life. However, when it comes to dining, traveling to a canteen and queuing are time-consuming and the campus population are thus sometimes denied pleasant dining experience.

# **Overview**

## **Background**

There are various food delivery applications serving the general public. However, the prices of the food and delivery fees can be prohibitive to students. Therefore, a food delivery application which makes available myriads of foods of the canteens scattered around the campus is needed to improve and enrich the students’ and staff’s dining experience of their daily life.

## **Overall Description**

“Foodie” is a food delivery application specifically designed to serve the entire campus population. With its mass delivery design in mind, the app provides food ordering and delivery services free of charge. Supported by the Flutter development framework, it is portable on both IOS and Android mobile platforms. Empowered by the firebase backend and database services, it is also highly scalable.

# **Investigation and Analysis Methodology**

## **System Investigation**

The application uses firebase for the backend and database services. The firebase database is a realtime database, which can provide users update to date data in the first place. Firebase also provides features such as google authentication and map, and functions that directly interact with front end and databases. For the front end, flutter is the framework equipped with modern designed libraries, and can directly communicate with firebase as well. Besides software and servers, the hardware required from users is either android phones or iphones, which is used by most people. Users’ login, orders, and other information will be inputted from front end prompts, to firebase, which provides functions to make logic responses, redirect third party plugins, and interactions with the realtime database to keep secure and efficient.

## **Analysis Methodology**

### **Feasibility study and requirements elicitation**

We do surveys to team members and friends to know what functions they would like to have most, and how they can be implemented. We also looked into other food ordering applications to find some intuitions and learn from them. The Project proposal is finished in advance to give big directions and also use cases in detail, while quality assurance and project management schedule is created to measure the progress and mitigate possible blockers and risks.

### **System analysis and requirements specification**

* + - 1. **Perform an analysis of the problem using object-oriented techniques.**

An external view of the model of the user ordering information including user login records, food orders, preferences, address and times will be developed as each one is a table stored in the database and an object while it is retrieving. The diagram is visualized with UML. This System Requirement Specifications documents will form part of the documentation for the project. Some desired features of the new system include:

* Choose desired food at restaurants
* Have update to data order status (pending, ready to pick up, on the way, arrived)
* Have accurate address and time for delivery
* If possible, promotion or user preferences can be added
  + - 1. **Scope and Limitations**

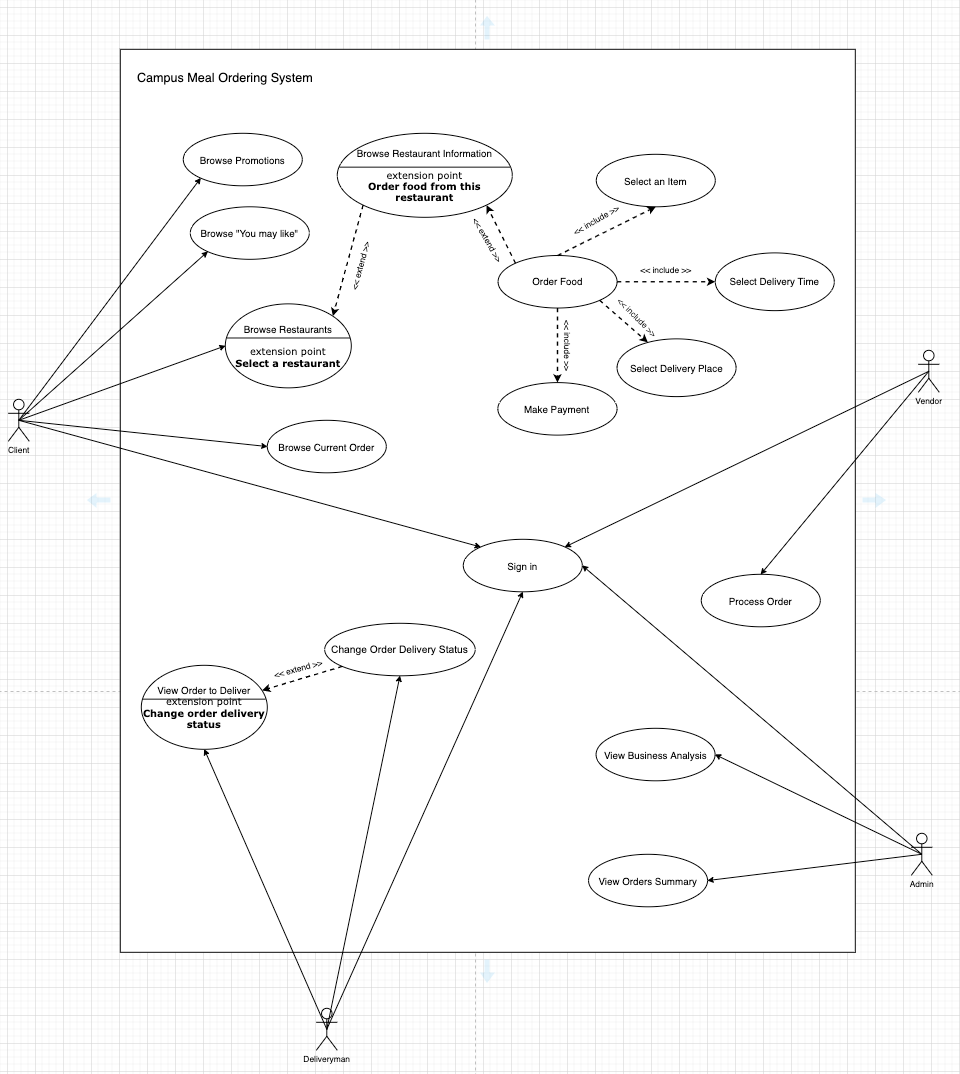
Analysis methodology will involve input/output analysis, requirement analysis, data analysis, process analysis, and application architecture:

* Requirement analysis: System documentation, functional and nonfunctional requirements.
* Input/output analysis: System I/O description, error message or help message
* Data analysis: Involve data storage, manipulation and retrieval, if possible, can do some analysis on user data
* Process analysis: Data and process flow analysis, system interfaces, functions interactions
* Application architecture: Usability, reliability, user interface design, interaction and application architecture design.

### **Object-oriented design using UML**

UML is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for visualizing and constructing. For our use cases, ordering is one of the most important features, and also how orders change status along the way. The order will be created, stored, and modified in the realtime database, while the system will display the up to time changes to users through neat interfaces. The whole use case is listed below.

* + - 1. **Use Case Diagram**



### Prototyping

The Object Oriented Rapid Prototyping (OORP) method will be used to implement a limited and functional prototype for the registration system. We will cover the most important parts including the ordering food and status changing for demonstration.

# **Constraints**

## **Scalability**

The application scales well as firebase handles all the servers and communication among them.

## **Data and function mapping**

We will use MVC in this application. For example, for the dish, we have a class named dish to encapsulate dish data into one object for use. When new dish data is created, we can easily create an object for this dish.

## **Proprietary hardware and software**

The hardware are from google firebase, the software is owned by our team.

## **Batch updates vs. Real-time updates**

For our application, we use cloud firestore, which is a realtime database supported by firebase. It can achieve the real-time updates and return users with the up to time changes.

## **Project Schedule**

There is a three-month timeframe to implement a production system for food ordering from project commencement in Aug 2020.

# **Operational Requirements**

## **Help Center**

Campus Meal Ordering System users have 24x7 access to telephone assistance for technical support, for example, failing to login, sluggish interface and slow response time ect.

## **Application Services and Technical support**

All application developers have full access to the application source code and the DBAs have access to the system backend Firebase project. They will be able to address application bugs quickly or enhance system features when necessary.

## **Administration Features**

Anyone can register via Google to become a normal user(client) of the system, while the vendor and deliveryman registration should be verified and conducted by the administrators. There are different levels of system access, the system contains 3 applications, client application, vendor application and delivery man application. Only authorized vendors could access the vendor application and only authorized delivery men could access the deliveryman application. Only authorized system administrators have access to all user information.

## **System hardware failure and backup**

System is hosted on Firebase, which is backed up by Google Cloud Platform (GCP). All hardware and data backup are managed on the Cloud. Google Cloud will ensure the hardware security. In case the hardware fails, the GCP side will do a backup.

# **Functional Requirements**

Here are atomic and verifiable functional requirements of the CMOS

## **Client Application**

### **Registration and Login**

* The system shall display “Sign-in with Google” when users first use this application.
* The system shall redirect users to Google OAuth page.
* The system shall redirect users back to the application when the user has authenticated with Google.
* The system shall display the system home page after the user has logged in to the system.

### **Browse Promotions**

* The system shall display a carousel of the restaurants that are being promoted in the system home page.
* The system shall allow users to navigate around different promoted restaurants by moving the carousel in the system home page.

### **Browse Restaurants**

* The system shall display the restaurants list available on the platform in the system home page.
* The system shall display each restaurant’s name in the system home page.
* The system shall display each restaurant’s picture in the system home page.
* The system shall display each restaurant’s position in terms of which canteen the restaurant is located at in the system home page.
* The system shall allow users to click on a specific restaurant to browse the detailed information about the restaurant in the system home page.

### **Browse Restaurant**

* The system shall display the restaurant’s name in the restaurant details page.
* The system shall display the restaurant’s ordering time in the restaurant details page.
* The system shall display the restaurant’s delivery time in the restaurant details page.
* The system shall display the restaurant’s offered dishes in the restaurant details page.
* The system shall display each dish’s name in the restaurant details page.
* The system shall display each dish’s price in the restaurant details page.
* The system shall allow users to click on each dish of the menu in the restaurant details page.

### **Browse Dish and Add dish to Cart**

* The system shall display the dish’s name in the dish page.
* The system shall display the dish’s price in the dish page.
* The system shall display an “Add to Cart” button in the dish page.
* The system shall allow users to click on the “Add to Cart” button to add the dish to the cart in the dish page.

### **Browse Cart and check out**

* The system shall display the “cart” button in the right bottom corner of the application after the user has logged into the system.
* The system shall allow the user to click on the “cart” button to browse the cart.
* The system shall display each added dish’s name in the cart page.
* The system shall display each added dish’s price in the cart page.
* The system shall display total price in the cart page.
* The system shall display a “Check Out” button in the cart page.
* The system shall allow users to click on the “Check Out” button to check out the order.

## **Vendor Application**

### **Registration and Login**

* The system shall display “Continue with Google” when users first use this application.
* The system shall redirect users to Google OAuth page.
* The system shall redirect users back to the application when the user has authenticated with Google.
* The system shall display the system home page after the user has logged in to the system.

### **Browser placed orders**

* The system shall display a list of orders placed during the current session in the home page.
* The system shall display “Confirm” and “Reject” buttons on each order in the home page.
* The system shall allow the user to click on the “Confirm” button to accept the order.
* The system shall allow the user to click on the “Reject” button to reject the order.

## **Deliveryman Application**

### **Registration and Login**

* The system shall display “Continue with Google” when users first use this application.
* The system shall redirect users to Google OAuth page.
* The system shall redirect users back to the application when the user has authenticated with Google.
* The system shall display the “Orders to Pick Up” page after the user has logged in to the system.

### **Browser orders to pick up**

* The system shall display a list of orders the user needs to pick up during the current session in the “Orders to Pick Up” page.
* The system shall display “Picked Up” buttons in the form of a tick on each order in the home page.
* The system shall allow the user to click on the “Picked Up” button to update the order delivery status to “Picked Up”.

### **Browser orders to deliver**

* The system shall display a list of orders the user has picked up and needs to deliver during the current session in the “Orders to deliver” page.
* The system shall display “Delivered” buttons in the form of a tick on each order in the home page.
* The system shall allow the user to click on the “Delivered” button to update the order delivery status to “Done”.

### **Browser orders delivered**

* The system shall display a list of orders the user has delivered during the current session in the “Done” page.

# **Input Requirements**

## **The Login Form and Account Access**

The login form is standard for a form of this type. It provides text fields for username and password, which the user must enter before signing in. This form also gives the option for a user to register for the site if they have not yet done so. To login, users must know their registered email address and corresponding case-sensitive password.

## **Order Form**

Users can navigate the numerous order forms using the main menu, each of which corresponds to a specific category of order items, adding items to their shopping cart along the way. At any time, users must be able to view and modify their shopping cart and when they are finally ready to place their order, they can proceed to the checkout form.

## **Sorting Preferences**

Users can choose to select preferences that will influence how their list of restaurants will be generated. Examples of preferences are on promotion, vegetarian, etc.

# **Process Requirements**

The following are among the inherent requirements that the online registration system must be able to handle.

## **Database Transactions**

The system must be able to retrieve, write and update data correctly to the database.

## **Data Integrity**

The system must execute all database transactions atomically, if any statement in the database transaction fails or succeeds, the whole transaction should fail or succeed accordingly.

## **Data Validation**

The system must validate all data entered by the user in a consistent manner.

## **Performance**

Must resolve locking issues and handle concurrent use of the system on a 24x7 basis. Send, receive and display user messages to assist the overall user experience**.**

# **Output Requirements**

## **Checkout Form**

The checkout form generated shall be short and concise. It shall only consist of the contents of the shopping cart to present a summary of the order and to calculate the total cost, in addition to allowing the user to specify all of the necessary delivery details.

## **Order Retrieval System**

The user must be able to view all the order records. User interaction with the order retrieval will be very simple. It shall display only the summary of records. To view the details of an order, the user must simply click on that order number, which will populate the panel the details, displayed in an easy to read. This structure can intuitively be expanded and collapsed to display only the desired information.

## **Reports and Summaries**

Authorised administrators must be able to extract summarized and rolled-up data into meaningful information. All records will be archived but accessible on demand.

# **Hardware Requirements**

## **Client**

Smartphone running Android or iOS operating systems

## **Backend server**

Instances requisitioned dependent on server load and running on Google Cloud datacenters

# **Software Requirements**

## **Client Operating Systems**

* Android
* iOS

## **Client Application**

Built for both Android and iOS. Requires:

* Network connection
* Location data (Delivery man)

## **Database and Storage system**

Implemented with Firebase

* Firestore
* Storage

## **Licenses**

Required licenses/subscriptions required to run and maintain software from third party vendors:

* Firebase Blaze Plan
* Apple Developer Program

# **Deployment Requirements**

## **Deployment on Android mobile platform**

* The application should be prepared for release.
* The application should be configured for release.
* A release version of the application should be built and signed.
* The release version of the application should be thoroughly tested.
* The Firebase backend and database services should be secure and production-ready.
* The application should be released on the Google Play platform.
  + Preparing promotional materials should be prepared.
  + The Google Play settings should be configured.
  + The assets should be uploaded
  + The release version of the app should be published

## **Deployment on IOS mobile platform**

* An iOS distribution provisioning profile and distribution certificate should be created
* An App Store Connect record for the app should be created.
* The app should be tested thoroughly.
* The app should be archived and uploaded using Xcode.
* The app’s metadata and further details in its App Store Connect record should be configured.
* The status of the app should be checked.
* If the app has been rejected, thoroughly test the app again and fix the bugs.