

**Campus Meal Ordering System**

**Design Report on Software Maintainability**

**By *Team Foodie***

**Lab Group: TS3**

**Date: October 2020**

**Instructors:**

**Bo An**

**Leong Yin Yoke Junie**

**Qiu Wei**

**Wang Rundong**

# **SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**NANYANG TECHNOLOGICAL UNIVERSITY**

**Document Change Record**

| **Revision** | **Description of Change** | **Done by** | **Date** |
| --- | --- | --- | --- |
| 0.10 | Initial Template |  |  |
| 0.30 | Add Architecture Design Strategy | Jun Yi | 28/10/2020 |
| 0.5 | Add Introduction and early planning phase | Renice | 30/10/2020 |
| 0.80 | Add correction and enhancement by nature | Renice | 31/10/2020 |
| 1.0 | Design Diagram | Renice | 1/11/2020 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

1. Introduction 4

2. Design Strategies 4

2.1. The Planning Phase Before Development 4

2.2. The Process of Developing 4

2.3. Correction by Nature 5

2.4. Enhancement by Nature 5

2.5. Maintainability Practices 6

3. Architectural Design Patterns 6

4. Software Configuration Management Tools 7

3.1. MediaWiki 7

3.2. GitHub 7

3.3. OneDrive 7

1. **Introduction**

Over the years, software products exhibited a large growth in size and complexity. Due to increased functional requirements, software changes are required to adapt and appropriate system configurations are made to correspond to those changes. One of the main concerns in the technological industry is the demand to maintain and enhance the software at an efficient and fast rate. Thus, improved and sustainable software design strategies must be practiced to meet this objective.

1. **Design Strategies**

## **The Planning Phase Before Development**

Early planning enables the team to anticipate parts of the system that requires significant development effort or potential issues. It also allows the team to take into consideration future features when designing the architecture of the system. With early planning and limited project schedule, features can be modified or added without incurring additional overhead development time to redesign components.

* 1. **The Process of Developing**

We are testing out in a small, test driven development. Due to safe distancing measures and contact minimization in place in response to COVID-19, we were unable to have members of our target user base to test our product. Instead, team members, non-developers and developers alike, perform the role of testers and provide continuous feedback on the design and usability of the application.

* 1. **Correction By Nature**

We will correct the application while testing. And this is what we will look out for:

* + 1. **Corrective Maintainability Faults**

Through testing, fault detection is done. When a user discovers a bug, an investigation will be conducted to identify the source of fault, to a specific component. A diagnosis will then be performed to identify the cause of failure. The component will then be modified and updated to fix the issue.

* + 1. **Preventive Maintainability**

Features are implemented in an atomic manner and each feature is tested independently. This is done such that error can be detected easily.

Referring to the user’s feedback and system logs, modifications and updates will be done to fix potential issues to ensure that the system is updated and working as expected. This also helps to prevent issues that may affect development.

* 1. **Enhancement By Nature**

We will enhance our application while testing it. And this is what we will look out for:

* + 1. **Adaptive Maintainability**

When there are updates to the platforms or dependencies of the system, the development team will research and check for any incompatibilities with our current system. If there are no known issues, a staging setup will be deployed and the testing will take place according to the testing plan to ensure that all features are working as expected before updating the production setup that is facing end users. This is to ensure that the application can be easily adapted to a new operational environment.

* + 1. **Perfective Maintainability**

Incremental modifications and updates will be done along the lifetime of the product to improve efficiency, performance, reliability and maintainability. The most recent and working deployment packages are backed up to allow restoring of deployment if the perfective maintenance fails and cripples the system. After product delivery, the updated version should be tested thoroughly to ensure that there are no bugs before deployment.

* 1. **Maintainability Practices**

To uphold quality in both process and product, we have implemented the following maintainability practices over the course of our project:

• Readable Code

• Version Control

• Standardized Documentation

• Modularity

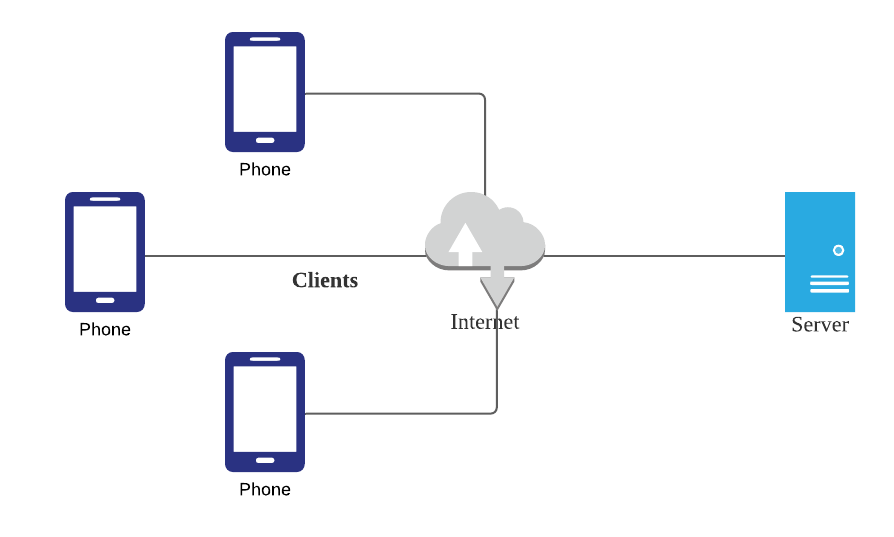
1. **Architecture Design Patterns**

**Client-Server Architecture**

CMOS has a broad target market thus it has to cater to a myriad of users and thus mobile phone hardware. Also, as the amount of users increase, the backend must be able to adapt to changes in customer demand and scale. Taking these points in mind we have pinpointed these traits to be the most essential in our design of the product; Portability and scalability.

As such we have settled on adopting a client-server architecture as the most suitable for CMOS. The client will be made to be lightweight and as it is built upon flutter it will be platform agnostic and Firebase is easily compatible and scalable to the needs of CMOS.

Following is the design:



1. **Configuration Management Tools**

This is where we will discuss version control management, and tracking on who made what changes and when.

* 1. **MediaWiki**

MediaWiki is a free and open-source application. This service is used as it is easy for beginners to pick up. There are many FAQs provided which can teach users the functions required by the users. There is a wide range of functions which allows users to create their information in different styles. It also allows users to concurrently edit the page at the same time. Hence, editing of the page will not result in a loss of information.

* 1. **GitHub**

GitHub is a source code hosting platform using the distributed version control and source code management Git. GitHub is chosen for its familiarity and support provided by various IDE applications. GitHub also supports issue tracking similar to a ticketing system. Whether it’s a software bug, code enhancement or documentation, users can open an issue, label them appropriately and assign them for other team members to resolve. All users involved will receive timely updates on the progress of the issue.

* 1. **Google Drive**

Google Drive service is used as a file storage and for the backup of documents initially created. This service allows users to share and store files within the group easily. Furthermore, OneDrive allows users to edit documents using the full functionality of Microsoft Office desktop applications. This service allows users to edit documents concurrently and supports version control.