# 

# 

**Design Report on Software Maintainability**

***(Mall-E)***

# 

# **Document Change Record**

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Description of Change** | **Approved By** | **Date** |
| 1.0 | Initial Report | Atrik Das | 04/05/2021 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 

# 

# Contents

[**1. Design Strategies**](#_det0ee9cttu6) **2**

[**1.1 The Planning Phase Before Development**](#_uzsly0ak6z3f) **2**

[**1.2 The Process of Development**](#_kyuo1ntlyey1) **2**

[**1.3 Correction by Nature**](#_ro9nzde5rf1p) **2**

[**1.3.1 Corrective Maintainability**](#_lzva8598u1bd) **2**

[**1.3.2 Preventive Maintainability**](#_bslq9yttp0n3) **2**

[**1.4 Enhancement by Nature**](#_pupzmnmtvza1) **2**

[**1.4.1 Adaptive Maintainability**](#_ei6y164fuo7c) **2**

[**1.4.2 Perfective Maintainability**](#_uiauyuoryium) **3**

[**1.5 Maintainability Practices**](#_a0l35miiu1t7) **3**

[**2 Architectural Design Patterns**](#_gg3p6iefpc0) **3**

[**3 Software Configuration Management Tools**](#_n28qct6ui1ie) **3**

[**3.1 MediaWiki**](#_fjcj6ztilwjb) **3**

[**3.2 GitHub**](#_qxnqn0kgb2c9) **3**

[**3.3 Google Drive**](#_seyqinig0cy0) **4**

[**3.4 SVN**](#_c03n41um3ecr) **4**

# 

# **1. Design Strategies**

## **1.1 The Planning Phase Before Development**

During the planning phase of the development of our application, we have analysed and predicted the improvements required after the release of this application. If popularity surges, we would be anticipating high demand / usage thus our application must be scaled up to handle even higher traffic rates. Hence, we targeted scalability as one of the factors that we would have to look into in the future.

## **1.2 The Process of Development**

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.

## **1.3 Correction by Nature**

During the development and testing phase of the application, we are predicting bugs to be found and corrections need to be made.

### ***1.3.1 Corrective Maintainability***

Fault detection done through testing and faults correction to be implemented.

### ***1.3.2 Preventive Maintainability***

Features implemented in atomic manner, each feature, tested independently, error detection and correction.

## **1.4 Enhancement by Nature**

We will enhance our application while testing the application.

### ***1.4.1 Adaptive Maintainability***

Can be easily adapted to a new operational environment.

### ***1.4.2 Perfective Maintainability***

After product delivery, if errors are detected, immediate correction to be made to reduce maintenance cost and time required.

## **1.5 Maintainability Practices**

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

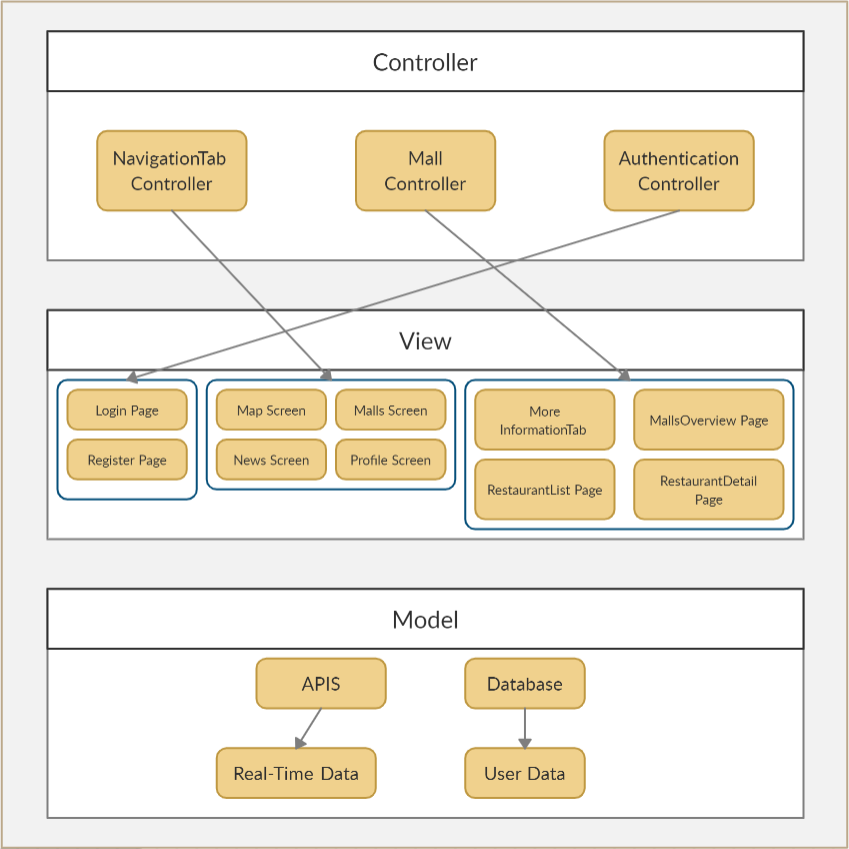
* Readable Code
* Version Control
* Standardized Documentation
* Modularity

# **2 Architectural Design Patterns**

Since Mall-E is heavily dependent on the user experience, we decided to adapt the Model-View-Controller (MVC) model as our architectural mode. This system infrastructure framework will be beneficial for Graphic User Interface design. It allows easy to change and add functions without breaking and affecting the other functions.

Mall-E is using the Model-View-Controller (MVC) architectural design pattern:

* The Model layer is where it holds the user data and records for the usage from the user interface layer.
* The View layer is where it stores/shows the components that display data from the Model layer to the user.
* Lastly, the Controller layer is where the components of the application that receive and carry out commands from the users to alter the View or Model layer.



# **3 Software Configuration Management Tools**

This is where we will discuss on version control management and logs on change / version history.

## **3.1 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 concurrently edit the page at the same time. Hence, editing the pages at the same time will not result in a loss of information.

## **3.2 GitHub**

GitHub is a source code hosting platform using the distributed version control and source code management software named 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. It also allows users to code together in a collaborative manner remotely.

## **3.3 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. It allows easy collaboration remotely too as it allows editing by different users simultaneously.

## **3.4 SVN**

TortoiseSVN is a free open-source Windows client for the Apache™ Subversion version control system. That is, TortoiseSVN manages files and directories over time. Files are stored in a central repository.