## About This Document

<table>
<thead>
<tr>
<th>Version No.</th>
<th>Author(s)</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Stockstill, Tuider, Welch</td>
<td>Feb 7, 2019</td>
</tr>
<tr>
<td>1.1</td>
<td>Stockstill, Tuider, Welch</td>
<td>Feb 20, 2019</td>
</tr>
</tbody>
</table>

## Document Properties

<table>
<thead>
<tr>
<th>Item</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document Title</td>
<td>System Design Document</td>
</tr>
<tr>
<td>Author</td>
<td>Stockstill, Tuider, Welch</td>
</tr>
<tr>
<td>Creation Date</td>
<td>Feb 1, 2019</td>
</tr>
<tr>
<td>Last Updated</td>
<td>Feb 20, 2019</td>
</tr>
</tbody>
</table>
# Contents

1 Introduction 4
   1.1 Purpose of the System ........................................... 4
   1.2 Design Goals ................................................... 4
   1.3 Definitions, Acronyms, and Abbreviations ........................ 5
   1.4 References .................................................................. 6

2 Proposed Software Architecture 6
   2.1 Overview ............................................................... 6
   2.2 Subsystem Decomposition ............................................ 7
   2.3 Hardware/Software Mapping ......................................... 10
   2.4 Persistent Data Management ......................................... 11
   2.5 Access Control and Security ......................................... 12
   2.6 Global Software Control .............................................. 12
   2.7 Boundary Conditions ................................................ 12

3 Subsystem Services ......................................................... 12
1 Introduction

The purpose of the System Design Document is to specify how the system should be implemented. This document is meant to be provided to a development team (in this case, we are the team), outlining everything they need to know to develop the system.

Specifically, this document will cover aspects of MLS, such as subsystem decomposition, hardware/software mapping, persistent data management, access control and security, global software control, boundary conditions, and subsystem services.

1.1 Purpose of the System

The purpose of MLS is to provide a solution for medicine misinformation and forgotten dosage times. To accomplish this, we will deliver two end-user systems: an Android mobile app for general users, and an administrative web portal for a select amount of administrators to moderate mobile app content.

1.2 Design Goals

This section will describe in detail the properties of MLS. These properties are closely related to the non-functional requirements specified in the RAD.

- **Efficiency**: Mobile users should get search results instantaneously. This will involve caching some database resources locally, such as photos. Also, efficient calls should be made to the database, and the schema should be streamlined and easy to query.
- **Accessibility**: Mobile users should be able to enjoyably and easily navigate the app. This will be accomplished by employing clean, modern design and intuitive layouts.
- **Security**: Users should not have to worry about their accounts being compromised. Sanitizing user input and encrypting passwords will help with the overall security of the system. Another facet of security involves information about medicines. This data should not be modified by anyone other than the administrator. By ensuring this lock-down, users will not have to worry about viewing information that has been tampered with.
- **Reliability**: Mobile users should be able to consistently use the app’s online features. This will involve deploying the back-end on reliable hardware.
- **Safety**: Due to the nature of the mobile app’s contents, disclaimers will be present on all pages. These will serve the purpose of reminding the user to contact a medical professional instead of consulting the mobile app.

In addition to the design goals derived from the RAD, we have additional criteria described below.
Storage Criteria

Our preliminary evaluation of this criteria is that end user mobile devices will need 15MB to 40MB of free storage space, and the target deployment server will need enough space for PHP server, MySQL, and the required scripts and data for those two softwares.

Maintenance Criteria

It should be easy to implement new features for MLS. Since its inception, the project has evolved in its scope. This has led us to set up the system in a way that makes it easy to integrate changes and additions. Also, we will incorporate good programming practices by following coding standards and writing detailed comments.

Cost Criteria

We will minimize the cost of development and deployment by using open source software and technologies such as Android Studio, Cordova, and MySQL.

Software Environment

One of the first steps of implementation is deciding which languages and technologies the team(s) should utilize. For MLS, our team will be working with the following:

- HTML for basic web page creation.
- Bootstrap for a stylish, modern layout.
- CSS for styling web pages.
- JavaScript (jQuery, AJAX) to dynamically modify web page content and to communicate with PHP scripts on the server.
- PHP to run scripts to retrieve and modify database contents.
- MySQL for the relational database management system.
- Android Studio for developing the Android mobile app.

1.3 Definitions, Acronyms, and Abbreviations

The following are definitions of acronyms and abbreviations that will occur throughout this document.

- MLS: Medication Leaflet System
- SDD: System Design Document
- RAD: Requirements Analysis Document
- RDBMS: Relational Database Management System
- GUI: Graphical User Interface
- MVC: Model, View, and Controller
1.4 References
The precursor to this System Design Document is the Requirements Analysis Document. All information found within this document is derived from the latter, and as such, should be read before this.

2 Proposed Software Architecture
2.1 Overview
Since MLS does not seek to update or replace an existing system, we will only describe the new system we have proposed.

Software Architecture
To describe the software architecture of MLS, we will implement the MVC architectural pattern. This architecture presents three tiers of the system: model, view, and controller.

This MVC architecture describes MLS in a concise manner because:

• The presentation layer describes our end user interfaces: Android devices and web browsers.
• The business layer encompasses the subsystems and their services which communicate with the data layer.
• The data access layer describes the MySQL database that manages persistent storage of MLS content.

2.2 Subsystem Decomposition

The following items are subsystems within MLS, partitioned into high-level descriptions of their functions. Along with these subsystems, we also incorporate a few off-the-shelf components. These components are listed below.
Persistent storage → MySQL
Preferred web browser → Google Chrome
Target mobile platform → Android
APIs:
• Android SMS
• Google Maps
• Cordova Barcode Scanner

The following diagram displays the services provide by previously mentioned off-the-shelf components and APIs.
Note: Due to the implementation progress of MLS, we will not be able to quickly migrate to an object-oriented approach; instead, the system will remain procedural.

Subsystem: User Account Management

The User Account Management subsystem allows mobile users to create an account and log in. This subsystem involves interaction with the MLS Database subsystem to store user account data and to search for existing user accounts. This subsystem also handles blocked users, ensuring that they cannot contribute to public information such as medicine reviews.

Subsystem: Medicine Search

The Medicine Search subsystem grants users the ability to search for a specific medicine and gain details about that medicine. Searching can be accomplished via user typing or barcode scanning. Also has the ability to add medicines to a personal list, thus invoking the Personal Medicine List subsystem.

Subsystem: Medicine Review

The Medicine Review subsystem allows users to post reviews of medicines they have had experience with. It also presents the ability to vote on which side effects a user experienced, allowing side effects to be ranked by probability per medicine.

Subsystem: Personal Medicine List

The Personal Medicine List subsystem will allow users to keep a personal list of the medicines they take. This subsystem interacts with Medicine Search, since a user must first look up the medicine before they can add it to their list.

Subsystem: Notifications

This subsystem handles medicine dosage notifications for both the individual user and external contacts such as friends and family. This subsystem interacts with Personal Medicine List, since a medicine must be added to a personal list before notifications can be set for it.

Subsystem: Pharmacy Locater

This subsystem handles information about pharmacies, such as location and other metadata.
Subsystem: Administration

The Administration subsystem covers many functionalities designed to moderate content that appears on the mobile app. These features include viewing/blocking users, viewing/deleting medicine reviews, and viewing/adding/editing/deleting medicines.

Subsystem: MLS Database

The MLS Database subsystem handles persistent storage of MLS data. All previously mentioned subsystems interact with this subsystem since they all must either store or retrieve data.

Relationship Between Subsystems

The following diagram displays the relationship between MLS subsystems. Since all subsystems interact with the MLS Database subsystem, the diagram is relatively straightforward.
2.3 Hardware/Software Mapping

The following diagram displays the relationship between aforementioned subsystems/off-the-shelf components and the hardware they will operate on.
2.4 Persistent Data Management

For persistent data management, we have chosen to use MySQL. The MLS database records user information, medicine entries, user notification times, medicine reviews, and pharmacy location information. Since user data will be stored in the database, MLS will utilize prepared statements and parameterized queries to combat malicious SQL. Below is the schema for the MLS database.
2.5 Access Control and Security

Below, we have created a table, or access control matrix, which displays the permissions of each type of end user for MLS. The purpose of this table is to explicitly outline privileges, ensuring that each end user type does not have unwanted permissions.

<table>
<thead>
<tr>
<th>Actor</th>
<th>Privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLS Admin</td>
<td>View all user accounts, medicines, and medicine reviews.</td>
</tr>
<tr>
<td></td>
<td>Block accounts.</td>
</tr>
<tr>
<td></td>
<td>Add/delete/edit medicines.</td>
</tr>
<tr>
<td></td>
<td>Delete medicine reviews.</td>
</tr>
<tr>
<td>Mobile user</td>
<td>Account creation.</td>
</tr>
<tr>
<td></td>
<td>Search medicines.</td>
</tr>
<tr>
<td></td>
<td>Write medicine reviews.</td>
</tr>
<tr>
<td></td>
<td>Vote on side effects.</td>
</tr>
<tr>
<td></td>
<td>Add medicine to personal list.</td>
</tr>
<tr>
<td></td>
<td>Set notification times.</td>
</tr>
<tr>
<td></td>
<td>Set group notification times.</td>
</tr>
<tr>
<td></td>
<td>Locate pharmacies.</td>
</tr>
</tbody>
</table>

2.6 Global Software Control

Since the nature of MLS introduces the possibility of many simultaneous user connections, we will have to balance loads in a manner such as queuing. By default, MySQL can handle a few hundred concurrent connections. When considering a user base of 10,000, it is not likely that there will be over 500 concurrent users at a time. When factoring in concurrent requests, this number is likely a fraction of concurrent users. All of this data will be useful in determining how to balance loads to ensure that MLS scales cleanly.

2.7 Boundary Conditions

Initialization of MLS will occur when the final administrative portal web page, PHP scripts, and MySQL database are active on a target server. Shutdowns can occur from a sudden issue with the server as a whole. Runtime errors may occur if an operator of the server account improperly modifies or deletes necessary files.

3 Subsystem Services

This section will describe the services provided by each subsystem of MLS. All of these services interact with the database in some fashion.
User Account Management

The user account management subsystem provides the following services:
- Registration of mobile users.
- Authenticates user input from login forms.
- Denies blocked users if they attempt to post reviews.

Medicine Search

The medicine search subsystem provides the following services:
- Allows mobile users to search through the database for medicine information.
- Searching is initiated by either user input or barcode search.

Medicine Review

The medicine review subsystem provides the following services:
- Allows mobile users to post reviews for medicines they have taken.
- Includes a side effect ranking system to determine what percentage of users experienced which side effects.

Personal Medicine List

The personal medicine list subsystem provides the following services:
- Allows users to add any medicine to a personal list.

Notifications

The notifications subsystem provides the following services:
- Allows mobile users to set personal notifications for medicines in their own lists.
- Interacts with the Personal Medicine List subsystem.

Pharmacy Locator

The pharmacy locator subsystem provides the following services:
- Handles location and metadata of stored pharmacies.

Administration

The administration subsystem provides the following services:
- Allows administrators to moderate content displayed on the mobile app.
- Users can view all user accounts, medicines, and medicine reviews.
- Users can block accounts.
- Users can add/delete/edit medicines in the database.
- Users can delete medicine reviews from the database.
**MLS Database**

The MLS database subsystem provides the following services:

- Persistent storage of MLS data.
- All previous subsystems interact with this subsystem.