LAMI Full-Stack Application

Project Solution Approach

LAMI

Logo

Description automatically generated

**LAMI Team**

Jacob Smith and Jerrel Chapmen

10/4/2022

**Note**: Recall that this writing assignment says:

Length = 5+ pages text + appendices as needed.   
  
Some materials do not count towards this 5 page minimum. These excluded parts include:  
 Cover page  
 table of contents  
 pictures  
 tables  
 images  
 diagrams

Posted as a single self‐contained file (no links to outside resources.)

Posted as a PDF file.

Typed single‐spaced.

Typed with black text.

Typed with #11 font size.

Typed using Arial font.

Typed with one inch margins on sides, top and bottom.

**(Please erase this page in your final document.)**

**TABLE OF CONTENTS**

[I. Introduction 3](#_Toc82521031)

[II. System Overview 3](#_Toc82521032)

[III. Architecture Design 3](#_Toc82521033)

[III.1. Overview 3](#_Toc82521034)

[III.2. Subsystem Decomposition 3](#_Toc82521035)

[I.1.1. [Subsystem Name] 4](#_Toc82521036)

[a) Description 4](#_Toc82521037)

[b) Concepts and Algorithms Generated 4](#_Toc82521038)

[c) Interface Description 4](#_Toc82521039)

[I.1.2. [Include sections III.2, III.3, etc., for other subsystems] 4](#_Toc82521040)

[IV. Data design 4](#_Toc82521041)

[V. User Interface Design 4](#_Toc82521042)

[VI. Glossary 5](#_Toc82521043)

[VII. References 5](#_Toc82521044)

[VIII. Appendices 5](#_Toc82521045)

# Introduction

The introduction begins by stating the purpose of the document. Explain the purpose for providing this design document and specify the intended audience for it. If this is a revision of an earlier document, please make sure to summarize what changes have been made during the revision (keep this discussion brief). Then provide a brief description of your project and state your project goal.

# System Overview

The system overview contains a general description of the functionality and design of the project. The overview will only briefly describe the overall design considerations and the comprehensive explanations will be done in the sections to follow. The overview should serve as an introduction to these sections.

The LAMI project is to create and add on to the already existing code and improve upon it, also to implement a website that can access the same things as the app. The coding will be done using the flutter language since it can be used to program apps and websites. Also, the database we will be using to keep track of profiles and personal info will be firebase. With these the users will be able to communicate through a server to add medications, sleep times for tracking when to take medicine, and surveys so the caretakers can check the medications effectiveness.

# Architecture Design

## Overview

The architectural design of the existing application is structured in accordance with the Client-Server network architecture. With regards to this application, the server refers to a central database and the client refers to two distinct groups, the employees of LAMI and the residents of The Alliance House. Both groups interact with the same server, albeit in slightly different ways. When broken down to its simplest form, the employee group can view and modify any entry in the database through a web or mobile application, whereas the resident group can only view and modify entries associated with their account through the mobile application.

## Subsystem Decomposition

The Client subsystem is implemented with the Flutter application framework

The Server subsystem is implemented with Google Firebase.

Diagram

Description automatically generated

### [Flutter Multi-Platform Application Framework]

#### Description

#### The Flutter framework provides mechanisms to create user interfaces for both mobile and web applications. A consistent framework for the development of all three UI’s (iOS, Android, Web app) increases readability, cohesion, and structure throughout the entire project. One of the most important services Flutter provides is the ability to communicate with the operating system of the phone to send a push notification outside of the application.

#### Concepts and Algorithms Generated

Flutter focuses on utilizing “widgets”, which are classes used to create UI. If we think of each app functionality as a widget we can expect an ‘alarm’ widget, a ‘log-in’ widget, a ‘profile’ widget, and a ‘monitoring’ widget. We can expect that these would take user input but would need to contact the backend to sync data to profiles and authenticate users.

#### Interface Description

#### Services Provided:

#### Service name: Alarm Widget

*Service provided to:* Flutter

*Description:* This service will allow authenticated users to add their medication to a list and set reminders for when they need to take their medicine. It should take user input and when a user adds an alarm to the list it should contact the backend and save it to the users list of current medications. The users’ current medication list should be viewable to those with an admin account.

*Services Required:*

Flutter, Firebase

### [Google Firebase]

#### Description

Firebase is a database system developed by Google as well as Flutter so we decided to use them together. The amount of documentation about connectivity between them also helped us. Firebase provides user authentication, NoSQL database, analytics, monitoring functions and so on for almost free. In our use case, we would not need to pay any extra cost for their service. Firebase offers two kinds of databases for customers, which are Realtime Database and Firestore database.

#### Concepts and Algorithms Generated

#### Realtime database - “The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in real-time to every connected client. When you build cross-platform apps with our Apple platforms, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.”

#### Firebase Cloud Firestore - “Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud. Like Firebase Realtime Database, it keeps your data in sync across client apps through real-time listeners and offers offline support for mobile and web so you can build responsive apps that work regardless of network latency or Internet connectivity. Cloud Firestore also offers seamless integration with other Firebase and Google Cloud products, including Cloud Functions.”

#### Interface Description

#### Services Provided:

#### Service name: Alarm Widget

*Service provided to:* Flutter

*Description:* Firebase provides libraries for their service so there is no difficulty to use their databases while we need to set up transitions among screens by ourselves. The login screen calls a function from Firebase Authentication to authenticate users and the register screen lets users to register themselves to the Authentication. The reset password screen and forgot password screen also utilize the Authentication to function their jobs. In the main screen, each user’s alarms are pulled from Cloud Firestore into the home screen by sending the authorized user’s id after checking if he or she exists in the database. In the home screen, the retrieved alarms are ordered by time without accessing the database any more.

*Services Required:*

Flutter, Firebase Authentication, Cloud Firestore

# Data design

[You may skip this section if your project doesn’t require any data manipulation or storage]

Describe all data structures (including the internal and temporary data structures), and the database(s) created as part of the application. This information is important from the design point of view as it will help the team in properly understanding all the data structures and databases which will be required for the coding.

# User Interface Design

[You may skip this section if your project doesn’t have a GUI component] – but! If the tools is ever to be used by humans (even just starting and stopping it), there’s some form of user interface design. It can be very simple, but it does exist. Make sure you document how you expect people to use your product, even if it’s just:

* Installation
* Configuration file edits
* Launch daemon by running command [x]

Provide a detailed description of user interface. The information in this section should be accompanied with proper images showing how exactly you vision the interface to be like (for example mock-ups). Make sure to mention which use cases in your “Requirements Specification” document will utilize these interfaces for user interaction.

# Glossary

Define technical terms used in the document.

# References

(Dutoit, 2010), 3rd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice Hall, 2010.

Cite your references here.

For the papers you cite give the authors, the title of the article, the journal name, journal volume number, date of publication and inclusive page numbers. Giving only the URL for the journal is not appropriate. You should use either IEEE or Chicago style formatting for your citations

For the websites, give the title, author (if applicable) and the website URL.

# VIII. Appendices

Any larger images, charts, or external materials should be put into appendices. These are attached at the end of the document, so the main materials are kept closer together and the overall flow of the document is preserved. If you include 4 pages of spreadsheets in the middle of a section, it makes it very difficult to track the flow of your presentation. Instead, those sheets go in Appendix [X] and are referred to by the earlier document.

You may have as many appendices as you need for the document to make sense.