CIS4398 - FALL 2016

MEDLOG REQUIREMENTS

REVISION HISTORY

Revision #	Author	Revision Date	Comments
1.0	John Allison, Brian Gander, Jesse Gardner, Dan Kauffman, Hong Nguyen	9/17/2016	Initial Draft
1.1	John Allison, Jesse Gardner	9/29/2016	Sentimental Diary and Activity Logger combined – now at 5 core components; minor assorted changes
1.2	Jesse Gardner, Dan Kauffman	9/29/2016	Added Software Development Plan section
1.3	Jesse Gardner	10/12/2016	Modified personal journal feature for log forwarding
1.4	Jesse Gardner, John Allision	10/14/2016	Removed exercise logging component, added productivity scaling system to personal journal, and added list common foods to dietary restrictions.
1.5	Jesse Gardner	12/11/2016	Removed Dietary Restrictions component, modified Journal requirements to remove categorization, modified medication/supplementation log to remove backlogs, modified user stories based on actual web and mobile requirements functionality, modified healthcare provider component to not include place(s) of practice, and added Tone Report

TEMPLE UNIVERSITY

CIS4398 - FALL 2016

Table of Contents

REVISION HISTORY 1
OVERVIEW 3
SYSTEM DIAGRAM (more to be added in future revision) 4
Web Application (Online) 4
Mobile Application (Offline & Online) 4
REQUIREMENTS 5
General Requirements 5
Features 5
Medication/Supplementation Logger 6
Health Provider Information Logger 6
SOFTWARE DEVELOPMENT PLAN 6
Front-end 10
USE CASES WITH ASSOCIATED STORIES 11
Case #1 “Patient logs journal entry via mobile device (offline)” 11
Case #2 “Patient logs activity related information via web app (similar to using sentimental diary logger)” 11
Case #3 “Patient forwards journal report to health care provider via web app” 12

TEMPLE UNIVERSITY

CIS4398 - FALL 2016

OVERVIEW

MedLog is an application that focuses on patients as the end-user, allowing them to log and manage health-related information in order to better maintain personal livelihood. MedLog enables the user to log information on the go as necessary and removes the burden of having to organize and consolidate these different types of records while also allowing logged information to be easily sent to a health-care provider as the need arises.

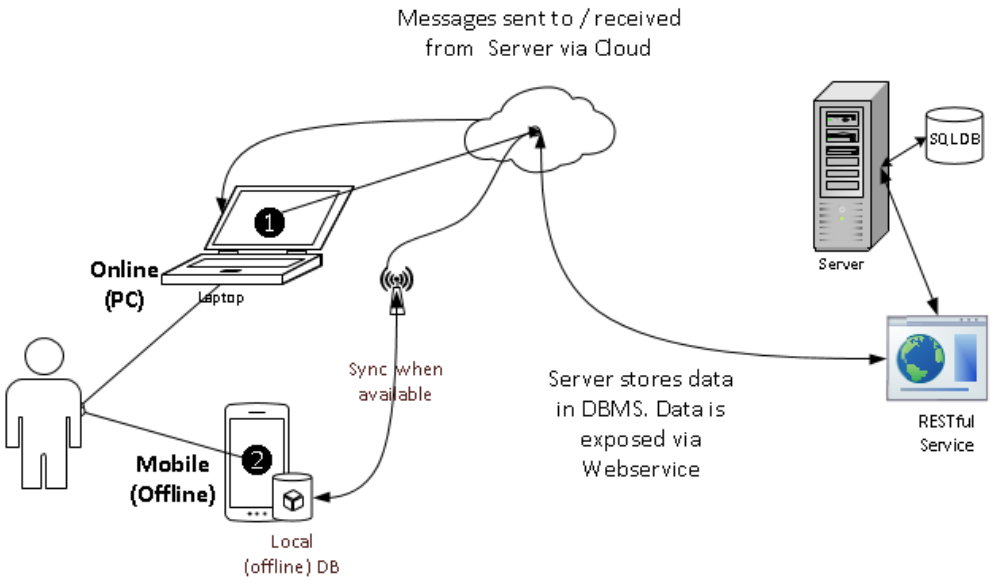
Logs are separated between three core components/categories of information archiving — personal journal, medication/supplementation, and health provider contact information. All logs will have the capability of being forwarded to a specific health provider, with special consideration regarding the personal journal, where health providers can be forwarded an averaged mood scaling estimate on mood per entry as well as a chart depicting the difference overtime. By giving patients the ability to create and manage health-related logs, not only are they able to more accurately keep tabs on lifestyle by developing a heightened sense of personal awareness, but also develop and maintain more improved interactions with their health providers, reinforcing follow-up treatment.

TEMPLE UNIVERSITY

MedLog

CIS4398 - FALL 2016

SYSTEM DIAGRAM (more to be added in future revision)[1]



This chart illustrates two basic usage diagrams, beginning with the user's entry point (client) and flows through hardware and software components that facilitate data access and entry.

Web Application (Online)

When the user connects to MedLog using an internet-connected device such as a laptop, data is transmitted to the Server and stored in the central DBMS immediately.

Mobile Application (Offline & Online)

When the user connects to MedLog from a mobile device that is not connected to the internet (using the mobile application), logged data is cached in a local store until such time as a network connection becomes active. When the device connects to the internet, the locally cached data is synchronized with the MedLog server using a series of API calls to the MedLog web service.

[1] *System diagram rev 1.0*

TEMPLE UNIVERSITY

CIS4398 - FALL 2016

REQUIREMENTS

General Requirements

Web Application: support latest versions of standard web-browsers (e.g. Microsoft IE, Mozilla Firefox, as well as Google Chrome/Safari and other standard WebKit based browsers)

Mobile Application : support Android OS based devices

Features

All core components will be accessible using the web application. The mobile application will support at least two of the five core components — medication/supplementation and dietary restriction. All logs will be archived with a corresponding timestamp the moment they are logged based on day and time as well as the capability of forwarding journal log information to specified health providers. Journal report information will be only forwardable via mobile.

Journal

- Provides a place for users to make use of a virtual personalized diary
- Logs will appear on screen in descending order (by date and time)
- Sentimental dairy component
    - Allows the patient to document “how they are feeling”.
- Activity logger component
    - Allows the patient to document “what they have achieved” or rather “accomplished” throughout a specified timeframe
- Implement a mood scaling and productive scaling system for each entry that will be averaged for total number of entries that can also be displayed as a graphic visualization
    - Capability to be forwarded to health providers along with other logs (doesn’t breach HIPAA guidelines)
    - Use a graphed line chart of mood/productivity ratings according to entry
    - Forwardable only through android mobile platform

Prediction Based Analysis on Mood For Future Journal Entries

- Use of IBM Watson’s Tone-Analyzer API
- Devise algorithm to draw correlation between mood rating and Sentimental Diary geared towards the latest entries (mainly focused on last entry with a minor focus of a historical range of previous entries)

- Correlate the mood rating with matching phrasing of context to the variables of the Tone-Analyzer API
  - Variables such as joy, fear, sadness, disgust, and anger (emotional tone used here as example) are weighted between 0 to 1 to identify accurate correlation compared to the context of the entries due to the phrasing of grouped words (sentence structures)
    - Use of emotional tone, social tone, and language tone to find variation all weighed between 0 to 1 and correlate those variations to the patient's mood and phrasing of entries
- Generate a predictive analysis visualization based on correlation

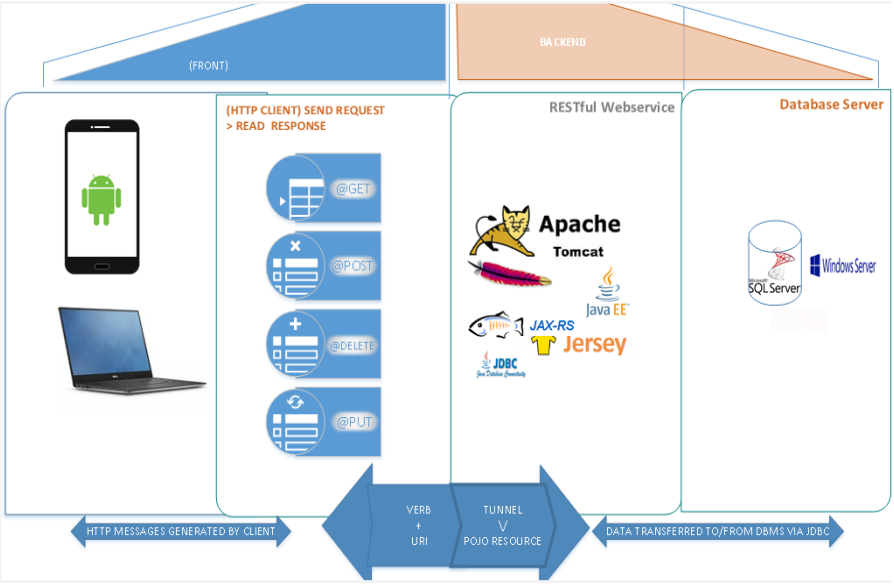
Medication/Supplementation Logger

- Allows the patient to keep track of a number of important variants regarding a particular medication or supplement.
  - Name of medication/supplement
  - Medication type/supplement categorization (prescribed, over-the-counter, and supplement) — if medication type is prescribed then health provider information of prescriber is necessary
  - Dosing information
    - Amount
    - Times-per-day

Health Provider Information Logger

- Allows the patient to document specific health provider information
  - Name of health provider
  - Occupation
  - Contact information — address, email address, skype, phone number
  - Additional comments section

SOFTWARE DEVELOPMENT PLAN



(fig.1) An overview of the software components, and function pieces of MedLog backend. RTL : Messages to and from the database are sent and received by a Java-based web application. Capitalizing on the existing set of tools available for JavaEE, MedLog will run JAX-RS code on Tomcat server with Jersey components, and implementing MS's JDBC. ***Supplemental code from third parties include****: Gson, JAXB, Commons DB Pool, Commons Collections, eclipseLink & ASM.* ***Testing supplements****: HttpUnit, EasyMock and Backbone.js*

Conceptually, the back end components abstract and standardize access. In practice, the back end will ensure data integrity and security. The two software components comprising MedLog's back end are the the RESTful web service architecture and the database server. Both web and db components will consist of several layers of abstraction[2].

Database Server *[fig.above-right]*

MedLog servers will run under Microsoft Windows Server 2008. SQL Server (MS-2008) engine will host MedLog's database. The database engine provides a couple of internal layers including: security, data integrity, concurrency, and adaptive tuning (e.g. index optimization based on key usage).

The MedLog data model adheres to the BC normalization rules. In order to maintain BCNF, CRUD operations will use not DML[3] directly. Instead stored procedures, along with user defined functions are the business and presentation layers of MedLog's DB component. Running a pre-compile procedure

[2]nTier layered architecture: Each layer, builds on previous and improves usability and simplicity.
In addition to usability and simplicity some small logic blocks are need for the layer function.
[3] Data Modification Language: SELECT, UPDATE, INSERT, DELETE

offers a number of advantages; one advantage is that candidate keys violations can be handled by the procedure and the appropriate action taken.

> *For example, the prescription drug table does not use the drug name as a PRIMARY KEY, using SQL Server's search framework and regular expressions, when a patient enters a drug name, logic in the stored procedure will determine if it should INSERT a new record, or UPDATE the existing record.*

Row-level data audit logging became a part of the database engine in R2008. MedLog will enable change data capture (CDC). Initially CDC will run on the diary table; erroneous / accidental changes will have an undo/rollback capability similar to version control.

Restful Web service [*fig1.above, 2nd from right*]

The MedLog web service has four functional requirements:

1. CRUD resources for each relevant entity in the database. API resources for special joins, along with management and administrative tasks.
2. Provide common API for data access for mobile and web applications.
3. Abstractions:
    a. Data access
    b. Business logic/validation
    c. Interface: RESTFul architecture presentation layer consists of a URI, verb, parameter and global ID.
4. Privacy-compliant multi-tenancy .
    a. Assign compliance officer and begin process of complying with EIPPA[4] and HIPPA[5] guidelines

The data access layer consists of two pieces:

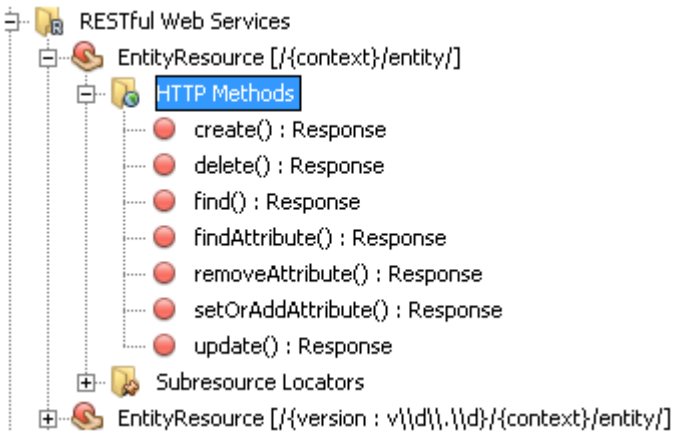
1. Managing the database connection
    a. Initially loading driver for MSSQL (Microsoft's JDBC 4.2)
    b. Persistence service (JPA/Eclipselink unit).
2. Executing SQL code in Java
    a. JDBC Statement[6]: Instantiate and set parameters as defined in SQL stored procedure.
    b. For READ methods, JPA manages (array-backed) cache of "result set".

The business layer validates modifications against rules as well as necessary parameters from respective Entity and passes *new* instances to the DAL method. Business logic examples:

---

[4] Educational Privacy Policy
[5] Health Privacy
[6] Statement: MS CallableStatment implementation translates into transactional calls to stored procedures.

- The current UserID is the only valid ID for entities that require UserID keys.
- A password is valid if it includes: letters, numbers and special characters (a printable non-alphanumeric)
- Validate that:
    - Required fields are non-blank
    - Keyed fields are valid keys.
    - Values that require additional validation are valid.
        - Email address pattern matcher.
        - Date range is non-negative.
    - All fields are sanitized of:
        - any restricted content (e.g. non-printable/control characters (\r \n \t)
        - html markup and other data types of injection 'attacks'



The client interface provides a set of HTTP methods." JAX-RS, core APIs enable developers to rapidly build Web applications in Java that are characteristic of the best designed parts of the Web. The API brings in support for designing and implementing Web resources and application that follow principles of REST (Representational State Transfer) architectural style to the Java Platform.[7] "

Resources are acted upon by using a set of simple, well-defined operations. The REST architectural style constrains an architecture to a client/server architecture and is designed to use a stateless communication protocol, typically HTTP. In the REST architecture style, clients and servers exchange representations of resources by using a standardized interface and protocol.

In the future MedLog would implement a proper token-based authentication system, such as OAuth. However for MVP, we will use Tomcat's authentic interface. We will extend the SQL authentication provider to allow certain resources exposed by JAX-RS as proxies for the server's realm restriction.

[7] JAX-RS (JSR!30)(Oracle - 2013)

Front-end

Web development requires the use of proper scripting, use of HTML and CSS for page layout. Native Javascript and jQuery will be used to mitigate front-end and back-end interweaving as well as dynamic page handling along with manipulating input data within the UI.

Android mobile development platform primarily consists of utilizing Java and XML markup regarding front-end development. Java programmatically allows functioning of activities, broadcast receivers, and use of services necessary for each logging component. XML provides reference to use of the functional components built on the platform in the case of journal entries.Development Environment

Vendor	Type	Software	Purpose	License
Microsoft	SERVER	SQL Server 2008R2	DBMS	MSDN-AA
Microsoft	IDE	SSMS	DB IDE	MSDN-AA
Google/Jetbrains	IDE	Android Studio	Android / Mobile	Apache 2.0
ApexSQL	VCS	SourceControl	Versioning Addin for SSMS	Commercial
Apache	SERVER	Tomcat	Web Server - host Webservice	Apache
Oracle	IDE	NetBeans	JavaEE / Servlets Design IDE	CDDL/GNU
Microsoft	API	JDBC	Java Driver for MSSQL	MIT
Slack	APP	Slack	Team communication	Proprietary
GitHub	VCS/PM	GitHub	Versioning and Issue management	TOS
Togglebox	SAAS	OnApp VM	Virtual Machine Hosting. SQL/Web	Commercial
Google	FRAMEWORK	Gson	Translate between POJO and JSON	Apache 2.0

TEMPLE UNIVERSITY

CIS4398 - FALL 2016

Version Control:

Repository URL	**Download/Clone URL**
https://github.com/tue65786/MedLog	https://github.com/tue65786/MedLog.git

VCS: SQL, WebService and client application projects are versioned with git. The central/push repository is hosted, publicly, on GitHub.

Work-around for MSSQL version control limitation (of using TFS): MedLog will use a third party plug from ApexSQL.

Team communication: MedLog's GitHub repository is [web] hooked to the Team's communication tool, Slack. GitHub will push code-commit alerts as well as changes to tickets (e.g. status, or team member assignment).

USE CASES WITH ASSOCIATED STORIES

Case #1 "Patient logs journal entry via mobile device (offline)"

1. Logs into MedLog app and is presented with journal entry tab along with an icon to the journal report
2. Enters journal entry info; sentimental diary entry and mood/productivity rating
3. Patient's device syncs with cloud. (when network connection becomes available)
4. Patient also has the option to forward updated information to a specific health provider

Case #2 "Patient logs activity related information via web app (similar to using sentimental diary logger)"

1. Signs into MedLog site

2. Goes to the activity logger tab
3. Logs recollection of periodic productivity related behavior
4. Log is saved to the database
5. Before meeting with a specific health provider to discuss lifestyle habits, patient is able to review the log details

Case #3 "Patient forwards journal report to health care provider via web app"

1. Signs into MedLog
2. Selects send tab in dashboard and is presented with drop-down list of specific healthcare providers
3. Selects health provider from a drop down menu corresponding to secure email address inputted in the health provider information logger
4. Submits forwarding request (to MedLog), where journal report information is forwarded