OpenLMIS Re-Architecture Concept Note

February 12, 2016

The global initiative to develop shared, open source solutions for managing medical commodity distribution in low and middle income countries. OpenLMIS: Configurable. Interoperable. Open Source.

www.openlmis.org // info@openlmis.org

Table of Contents

OpenLMIS Background

Billions of dollars in life-saving medical commodities are distributed annually to low income countries, however ensuring these commodities reach those most in need continues to be a challenge. Large populations are unable to access essential healthcare services because of inadequate distribution systems. The result is devastating to these communities: as many as 1.5 million children die annually from vaccine-preventable diseases. Many of these deaths could be prevented by improvements in supply chain performance.

Public health supply chains need accurate, timely logistics data from health facilities to make decisions about procurement and supply. Too often this information is inaccurate or simply not available. As a result, health systems in many countries continue to experience stock-outs of essential medicines, leaving people vulnerable to treatable illness and diseases.

In low- and middle income countries, the distribution of medicines relies on a fragmented mix of information systems that often focus on isolated parts of the supply chain or on a single health program. Less emphasis has been placed on ensuring that all programs and all facilities are supported, and that information flows between supply chain layers from the national warehouse to the service delivery point.

In many cases, countries have had to build and support their own custom solutions, or rely on costly commercial systems. Globally, this approach is expensive, inefficient, and often results in inadequate solutions that become increasingly difficult to maintain over time. Countries are not able to leverage the work already done by their neighbors, and instead have to "reinvent the wheel" when new features are needed. This leads to information systems that are outdated and unable to effectively handle the growing number of health programs and products.

OpenLMIS is a global initiative to support the development of shareable, interoperable, open-source software for electronic logistics management information systems. OpenLMIS was designed by a community of countries and international stakeholders who believe in working together to solve common challenges. Countries and donors have pooled resources to create a non-proprietary product that is built on shared user requirements across countries. The end result is a more flexible and powerful information system than what any one country or organization could create individually. As an open source project, the software is available free of charge, and enhancements made by individual countries are contributed back to the community for others to use.

Challenge to Address

In the fall of 2012, with strong ongoing guidance provided by the Tanzania and Zambia ministries of health and JSI through the USAID | DELIVER PROJECT, VillageReach and ThoughtWorks began development on the first release of OpenLMIS, an open-source logistics management information system (LMIS).

Significant contributions from PATH, USAID, Rockefeller Foundation, the Bill & Melinda Gates Foundation, the UN Commission on Life-Saving Commodities, and others first helped shape OpenLMIS. JSI customized and extended the initial OpenLMIS code base (v0.9) for national deployments in Tanzania and Zambia in late 2013 under the name "eLMIS." In 2015, eLMIS was also deployed to Cote D'Ivore. OpenLMIS software development continued with the v1.0 release, which was deployed by VillageReach to manage vaccine distribution in Mozambique and Benin.

As new installations of OpenLMIS were developed and deployed, challenges surfaced related to the OpenLMIS software internals. A key challenge was the inability to easily extend the code base, which resulted in a "code fork" between the Tanzania and Zambia eLMIS implementations and the Mozambique and Benin OpenLMIS v1.0 implementations.

In an effort to address the fork, the community agreed to begin working toward a common master branch hosted on GitHub. ThoughtWorks created the "2.0" branch based on the eLMIS code line and the community has kept it current with updates from both eLMIS and a new facility stock management project led by CHAI in Mozambique. Merges of project code to this branch were halted in December 2015 to prepare for a stable "2.0" release of OpenLMIS. This version is slated to be released March 2, 2016.

During the all-community meeting held in September 2015, three governance groups were formed to help guide and manage the activities of the OpenLMIS community. The Governance, Product, and Technical groups meet regularly to discuss and find consensus on key issues relating to OpenLMIS development and management.

In September, the community also agreed that a single "core" code line was required. This alone does not resolve many of the difficulties that may cause forking. Four of the primary pain points identified by the community are listed below:

- Monolithic Architecture: The lack of modularity, extension points or similar structures discourages open source development and encourages forking. As such, it is difficult to define a core set of OpenLMIS features and services. Projects often require custom code that should not be included in the shared OpenLMIS codebase.
- No data collection extensions: Deployments may have specialized data collection needs and a need a way to collect and report on custom data that can migrate through releases.
- Hardcoded User Interface (UI): The current UI does not contain sufficient configurability and ability to brand/modify look and feel.
- No Suitable API: OpenLMIS also lacks a standard, reusable Application Program Interface (API) with which to build a UI, and support integrations with other Health Information Systems.

These pain points must be resolved for OpenLMIS to continue as a viable open source choice for LMIS needs worldwide. Similar open source projects, such as DHIS2 and OpenMRS, have faced comparable challenges and at some point in their project trajectory had to undertake similar re-architecture efforts. The changes required are significant, particularly for modularity and extensibility.

Proposed Approach

Re-architect OpenLMIS to **better enable contribution** and **maximize shared benefit** by promoting code reuse and transferrable customizations while simultaneously providing the ability to maintain and modify a stable OpenLMIS Core.

This includes maintaining processes, policies and technical support for the contribution of features and fixes to OpenLMIS. Transferrable customizations—modules—are a mechanism for projects to build non-global features that may be applied to future versions of OpenLMIS, thereby easing the upgrade or migration process.

Key Objectives

The following list outlines the key objectives for the re-architecture project.

- Introduce Modularity and Extensibility to OpenLMIS such that:
    - OpenLMIS has a concept of Core: features and services that are foundational to OpenLMIS and included with every deployment
    - Core features are *extensible*, meaning that their behavior, workflow and data collection may be modified in sanctioned ways by a module
    - Additional functionality may be added at run or build time via a module/plugin-style architecture. These modules may include new functionality, new UI components, new database tables and columns, new data, and new Web Services
    - Modules are backward-compatible with new releases to Core OpenLMIS
    - Modular architecture can be achieved primarily with standard technologies (e.g. OSGi, Spring, micro services, etc.) that facilitate open source development

- Refactor the Infrastructure:
    - Introduce technologies and structure that support an extensible domain model
    - Introduce a unified persistence model: a means to store and retrieve data by both Core and modules. Modules must be able to extend the data model

- Implement Structural changes:
    - Break up the monolithic OpenLMIS project into several that, along with any desired modules, form an OpenLMIS deployment:
        - Core: foundational services and logic
        - Reference application: a basic OpenLMIS application that may be used and modified by deployments
        - Reference UI

- Provide for Migrations/Upgrades:

    - A deployment, with existing modules and possibly custom code, can upgrade to new releases of Core OpenLMIS and/or updated modules via a well-defined process
    - Define a migration path for current OpenLMIS deployments
    - The application must be easily deployable as a cloud-hosted, Web-based application on a well-known managed-service provider.

- Enable UI Customization:
    - Modules may add additional user interface elements, such as menu items and screens
    - Modules may modify existing Core screens to capture new data elements, or accommodate new functionality provided by a module/plugin

- Maintain Rights and Roles:
    - Regarding changes above, access should continue to be enforced by OpenLMIS' permissions structure. Changes may be required to enforce permissions at all levels (UI, web service/API, reporting, etc.)

What is Core OpenLMIS?

The envisioned core of OpenLMIS is the set of services and features that compose its essential value: an open-source LMIS solution for low-resource environments. During the September meeting, the community agreed with the concept of Core and embarked on a group exercise to list the primary aspects of core OpenLMIS. The raw results can be found here.

While the community must refine and ratify this list, initial points of agreement on key aspects of core are listed below:

- Requisitions
- Stock Management
- Program-based (Multiple "verticals" of Programs with their own products, requisition workflows, costing data, etc.)
- Offline Capability for Select Business Processes
- Essential objects: Programs, Products, Facilities, Users, Supply Chains, Delivery Zones, Requisitions
- Reference Data: out-of-box data essential to OpenLMIS operation
- Basic cold chain equipment support
- Data source for analytics (ideally through an interface vs. direct data store access)
- Basic Reporting Module, with out-of-box reports reflecting supply chain good practices

Architectural Vision

Moving forward, the vision for the re-architecture of OpenLMIS includes a number of key developments critical to the functionality and flexibility of the product.

These terms may be considered the primary components of the OpenLMIS vision, and figure greatly in the overall structure of the product as it evolves. Figure 1 presents the architecture vision.

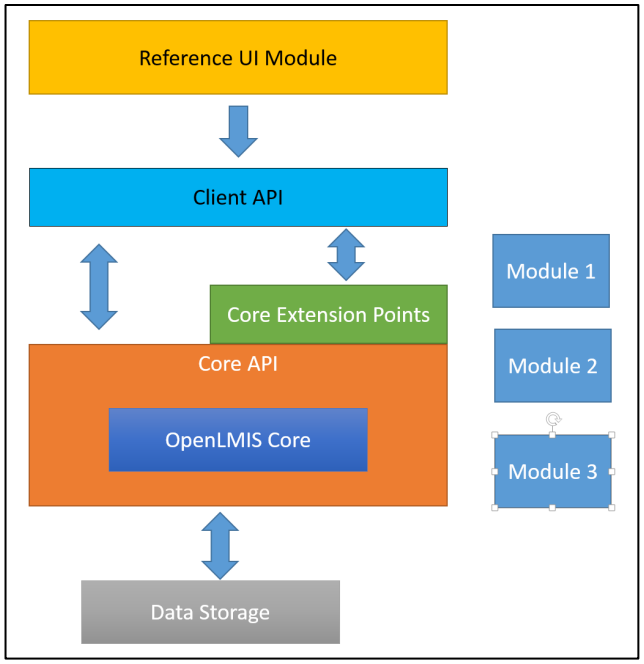

Figure 1 OpenLMIS proposed architecture

Core. OpenLMIS will have a core set of services and functionality—the essence of the product—which will include common LMIS components such as stock management and requisitions as well as first-class objects such as Products, Facilities, and Users.

Core also provides basic application features, such as authentication and authorization, rights and role management. Core features are accessed through standardized APIs and extension points.

Core API. Access to OpenLMIS core is always performed through well-known and well-documented APIs, insulating the rest of the application from changes to core. This layer of API is distinguished from the Client API; the Core API will be used in-process and requires in-depth knowledge of OpenLMIS internals.

Client API. This layer is the public interface of OpenLMIS suitable for all manner of clients, including user interfaces, integrations, point utilities, and reporting. It is optimized for low resource needs and is extensively documented for external use.

Extension Points. Extension points enable custom behavior within core services. These operate alongside the core APIs. An example of an extension point is the calculation of required stock. OpenLMIS offers restock algorithms based on population or historical consumption. Using an extension point, a module may substitute a third algorithm to meet country requirements.

Modules. Modules add functionality not considered part of Core OpenLMIS, and are used for adding new features. The feature may be a very project-specific feature, or a broader add-on that is useful to any number of deployments.

Modules are complete packages of functionality, and include actual logic, data storage extensions, API, and UI (the last few are optional). Modules act as the layers above OpenLMIS Core logic, as depicted in Figure 2.

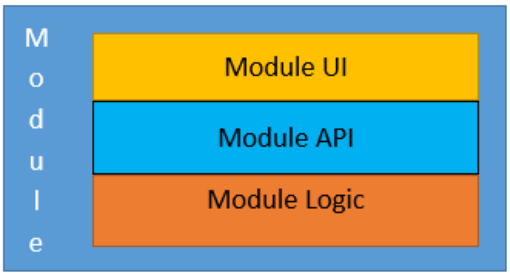

Figure 2 Module Detail

Some examples of features best suited for a module are:

- Dispensing
- Electronic Medical Records
- Bar coding
- Analytics tool
- Budgeting
- Master Product List

Appendix B lists the further characteristics of core and modules. A module can, with agreement from the OpenLMIS technical governance, be made part of Core. It is entirely possible for contributed modules to become community supported.

Reference UI. OpenLMIS offers, as a separate module, a reference user interface. The reference UI is functionally complete, meaning it provides access to all core services as well as everything needed for an LMIS implementation. The reference UI interacts with the OpenLMIS server only through the API layer. The API layer is sufficient to build a completely new interface to meet project needs.

Projects that need to customize (versus configure) portions of the reference UI will fork the reference UI module and make the changes. If the project is upgraded, it simply refreshes its fork with the latest reference UI release.

Reference Application. To complete the separation started with the core and the reference UI, OpenLMIS delivers a reference application. Treated as a separate project, the reference application comprises Core, the reference UI, and any community-supported modules deemed appropriate. The reference application contains all the logic for a modern Web application and will end up looking mainly like OpenLMIS today.

The advantages of a separate reference application are a different level of modularity and protection of core from changes more likely to occur (for example, a change to the application's authentication scheme).

Data Storage and Reports. Core OpenLMIS defines a standard way to persist and retrieve data, including a means to add additional data storage per project requirements. Reporting is essential for any OpenLMIS deployment. Thus, input is required from the community on what level of "out of the box" reporting OpenLMIS should provide.

For now, OpenLMIS will continue offering the embedded Jasper reporting engine. Ideally, any embedded reporting framework tuned for OpenLMIS will be implemented as a module. This includes dashboards viewed within the OpenLMIS UI. OpenLMIS will NOT provide complex analytics or business intelligence capabilities.

Reduced Feature Set (as compared to 2.0). A re-architected OpenLMIS containing only core features will be smaller in size than the current 2.0 branch. The current set of code in the 2.0 branch contains project-specific features that must be removed as, by definition, it does not belong in Core. If the community desires these features to be included, either as core or as a module, the core team will provide guidance to on modularizing these features.

Plan

The work is planned to start as soon as possible. While the duration for different approaches to this solution will vary, based on initial project scoping the development life cycle of the applications should take no more than one year.

VillageReach has staff dedicated for the following roles/function for OpenLMIS:

- Subject Matter Expertise in OpenLMIS
- Consultation, technical and project discussions, daily meetings, and a general project coordination support
- Software Engineering and Architecture
- Project Management

In order to compress the timeline for delivering a re-architected OpenLMIS that achieves the Key Objectives, VillageReach is looking for a Partner to:

- Learn and understand OpenLMIS, its purpose, operation and architecture, and become familiar with the OpenLMIS community;
- Understand the pain points faced by OpenLMIS implementers and developers;
- Collaborate with the OpenLMIS core software engineering team and the OpenLMIS community to create high-level and detailed architectural designs for a major refactor of OpenLMIS that fulfil the *Key Objectives*, above.
- Co-create the re-architected OpenLMIS software according to the project timeline, in partnership with the core OpenLMIS development team.
- Ensure knowledge of re-architecture activities and new application is built within the core OpenLMIS development team and OpenLMIS community, and produce high-quality documentation to support this objective.

It is essential that the Partner collaborate with the OpenLMIS Core team and OpenLMIS Community during the design and development process. On-site presence with the OpenLMIS Core team (Seattle, WA) is preferred.

Scope of Work for Collaboration

- Software Analysis
- Software Architecture
- Software Development
- Documentation
- Build & Deployment Plan
- Design and Documentation for Data migration from old systems/transition
- A comprehensive risk analysis/mitigation strategy

The team size and composition may vary depending on the approach taken.

Ideally, the partner will have experience with the development of distributed open-source projects using a modern Web stack (e.g. Java, Ruby, HTML5, Web Services, and document and object-oriented databases) using Agile.

Envisioned Phases

Phase One: Analysis & Design	I	• Perform technical analysis of current OpenLMIS implementations vis-à-vis the Key Objectives. • Develop Modularity and Domain Model design
	II	• Document findings and design the new Web-browser based application (OpenLMISv3) • Recommended approach to build, deployment & data migrations • Implementation Plan
Phase Two: Software Development and Testing	I	• Assist OpenLMIS Core development team with implementation of design. • Training materials for deployment, monitoring and troubleshooting for tech admins
	II	• Help document and implement a deployment plan to a cloud-based hosting provider, such as Amazon, Google, Heroku, etc., including data migration, communications materials and documentation of software and processes. • Training materials for contributors
Phase Three: Deployment and Rollout	I	• Rollout strategy for all platforms and deployment mechanisms, including mobile devices, locally and globally managed cloud-hosted instances. • Assistance with migrations (TBD depending on if/how many implementations choose to upgrade)

Appendix A: Guiding Principles

The OpenLMIS project strives to follow the *Principles for Digital Development* (http://digitalprinciples.org/):

1. **Design with the User:**
   - Develop context-appropriate solutions informed by user needs.
   - Include all user groups in planning, development, implementation, and assessment.
   - Develop projects in an incremental and iterative manner.
   - Design solutions that learn from and enhance existing workflows, and plan for organizational adaptation.
   - Ensure solutions are sensitive to, and useful for, the most marginalized populations: women, children, those with disabilities, and those affected by conflict and disaster.
2. **Understand the Ecosystem:**
   - Participate in networks and communities of like-minded practitioners.
   - Align to existing technological, legal, and regulatory policies.
3. **Design for Scale:**
   - Design for scale from the start, and assess and mitigate dependencies that might limit ability to scale.
   - Employ a "systems" approach to design, considering implications of design beyond an immediate project.
   - Be replicable and customizable in other countries and contexts.
   - Demonstrate impact before scaling a solution.
   - Analyze all technology choices through the lens of national and regional scale.
   - Factor in partnerships from the beginning, and start early negotiations.
4. **Build for Sustainability:**
   - Plan for sustainability from the start, including planning for long-term financial health, e.g. assessing total cost of ownership.
   - Utilize and invest in local communities and developers by default, and help catalyze their growth.
   - Engage with local governments to ensure integration into national strategy, and identify high-level government advocates.
5. **Be Data Driven:**
   - Design projects so that impact can be measured at discrete milestones with a focus on outcomes rather than outputs.
   - Evaluate innovative solutions and areas where there are gaps in data and evidence.
   - Use real-time information to monitor and inform management decisions at all levels.
   - When possible, leverage data as a by-product of user actions and transactions for assessments.
6. **Use Open Data, Open Standards, Open Source, Open Innovation:**
   - Adopt and expand existing open standards.
   - Open data and functionalities, and expose them in documented APIs (Application Programming Interfaces) where use by a larger community is possible.
   - Invest in software as a public good.
   - Develop software to be open source by default with the code made available in public repositories and supported through developer communities.
7. **Reuse and Improve:**
   - Use, modify, and extend existing tools, platforms, and frameworks when possible.

- Develop in modular ways favoring approaches that are interoperable over those that are monolithic by design.

8. **Address Privacy and Security:**
   - Assess and mitigate risks to the security of users and their data.
   - Consider the context and needs for privacy of personally identifiable information when designing solutions and mitigate accordingly.
   - Ensure equity and fairness in co-creation, and protect the best interests of the end-users.
9. **Be Collaborative:**
   - Engage diverse expertise across disciplines and industries at all stages.
   - Work across sector silos to create coordinated and more holistic approaches.
   - Document work, results, processes, and best practices, and share them widely.
   - Publish materials under a Creative Commons license by default, with strong rationale if another licensing approach is taken

Appendix B: Core Application and Modules

Core Code	• Highly stable and well-tested • Changes infrequently • Regular release schedule • Supported by the OpenLMIS Community (bug fixes, etc)
Community-Supported Module	• Meets community-supported module standards (testing coverage, coding style, etc. – see wiki) • Aligns with OpenLMIS Product Roadmap • Functionality supports multi-country requirements • Supported by the OpenLMIS community (bug fixes, etc)
Contributed Module	• Any individual or organization can submit and share a Contributed Module • No specific code standards • May be supported by the contributor, not supported by the OpenLMIS community • Often will be created via country-specific implementation projects

Appendix C: OpenLMIS v. eLMIS Feature Comparison

This comparison is valid as of 7/15/2015. A new feature variability matrix is currently under analysis and will be releases publically as soon as it is available.

Feature		Description	Options	OpenLMIS v1 (Deployed in Mozambique and Benin)	OpenLMIS v1.x / eLMIS (Deployed in Tanzania and Zambia)
Configuration					
1	**Upload**	Allows for upload of CSV files for system set up		✓	✓
			1. Delivery Zones	✓	✓
			2. Delivery Zone Members	✓	✓
			3. Map Delivery Zones to Program Schedules	✓	✓
			4. Delivery Zone Warehouses	✓	✓
			5. Dosage Units	✓	X
			6. Facilities	✓	✓
			7. Facility FTP Details	✓	✓
			8. Facility Operators	✓	X
			9. Facility Types	✓	X
			10. Facility Approved Products	✓	✓
			11. Geographic Levels	✓	X
			12. 12. Geographic Zones	✓	✓

			13. 13. Products	✓	✓
			14. Product Category	✓	✓
			15. Product Forms	✓	X
			16. Product Groups	✓	✓
			17. Programs	✓	X
			18. Program Product	✓	✓
			19. Product Prices per Program	✓	✓
			20. Programs supported by Facilities	✓	✓
			21. Regimen Categories	✓	X
			22. Requisition Groups	✓	✓
			23. Requisition Group Members	✓	✓
			24. Map Requisition Group Members	✓	✓
			25. Map Requisition Group Members to Programs + Schedule	X	✓
			26. Supervisory Nodes	✓	✓
			27. Supply Lines	✓	✓
			28. Users	✓	✓
2	**Configure**			✓	✓
	a) R & R Template	Allows configuration of report and requisition template for each program		✓	✓
	b) Regimen Template	Allows configuration of reporting fields and regimens for each program		✓	✓

	c) Program Product ISA	Allows configuration of ISA formula for each product (by program)		✓	✓
	d) System Settings	Allows configuration of system settings		✓	✓
			1. Order file	✓	✓
			2. Shipment file	✓	✓
			3. Budget file	✓	✓
			4. Order number	✓	✓
	e) Settings	Allows configuration of additional settings		X	✓
			1. Analytics	X	✓
			2. General	X	✓
			3. Notification – Email	X	✓
			4. Notification - SMS	X	✓
			5. Order Export	X	✓
			6. R & R	X	✓
			7. Report Labels	X	✓
3	**Messages**	Displays list of SMS messages sent to user account and allows user to send a new SMS		X	✓
4	**Manage**			✓	✓
	a) Facilities	Allows user to add facilities, and/or lookup and edit information for existing facilities		✓	✓
	b) Roles	Allows user to add, define, and edit different user		✓	✓

		roles			
	c) Schedules	Allows user to add or edit different schedules		✓	✓
	d) Users	Allows addition of new users and editing of user information		✓	✓
	e) Supervisory nodes	Allows addition of new and editing of existing supervisory nodes		✓	✓
	f) Geographic zones	Allows addition of new and editing of existing geographic zones		✓	✓
	g) Requisitions groups	Allows addition of new and editing of existing requisition groups		✓	✓
	h) Supply Lines	Allows addition of new and editing of existing supply lines		✓	✓
	i) Facility approved products	Allows addition of new and editing of existing products based on program and facility type		✓	✓
	j) Products	Allows addition of new and editing of existing products		✓	✓
5	**Equipment**			**X**	✓
	a) Equipment types	Allows addition of new and editing of		**X**	✓

		existing equipment types			
	b) Equipment list	Allows addition of new and editing of existing equipment		X	✓
	c) Products for Equipments	Allows association of products to specific equipment and programs		X	✓
	d) Service Types	Allows addition of new and editing of existing service types		X	✓
	e) Service Vendors	Allows addition of new and editing of existing service vendors		X	✓
	f) Service Contracts	Allows addition of new and editing of existing service contracts		X	✓
	g) Donors	Allows addition of new and editing of existing donors		X	✓
6	**Vaccine**			X	✓
	a) Protocols			X	Under development
	b) Logistics Template			X	Under development
	c) Diseases	Allows addition of new and editing of existing diseases for tracking		X	✓
	d) Manufacturer	Allows addition of new and editing of existing manufacturers and product		X	✓

		mapping			
	e) Transaction Type	Allows addition of new and editing of existing transaction types		X	✓
	f) Receive Status	Allows addition of new and editing of existing received statuses		X	✓
	g) Storage	Allows addition of new and editing of existing vaccine storage information		X	✓
	h) Storage Type	Allows addition of new and editing of existing storage types		X	✓
	i) Temperature	Allows addition of new and editing of existing temperatures		X	✓
	j) Country	Allows addition of new and editing of existing countries		X	✓
Requisition (Pull) Supply Process					
7	**Requisitions**			✓	✓
	a) Create/Authorize	Allows the creation of a new requisition for supply periods		✓	✓
	i. Fully Supply (/Priority Drugs and Medical Supplies)	Allows user to enter requisition information for full supply products, optional data validation and display of		✓	✓

		budget information			
	ii. Non-Full Supply (/Additional drugs and Medical Supplies)	Allows user to add additional non-full supply products to the requisition, specify quantity requested, and reason, with optional display of budget information		✓	✓
	b) Approve	Allows supervisor (or appropriate personnel) to review and approve submitted requisitions		✓	✓
	c) Convert to Order	Allows approved requisitions to be converted to orders		✓	✓
	d) View	Allows user to view requisitions based on selected facility, program, and date filters		✓	✓
8	**Orders**			✓	✓
	a) View Orders	Allows user to view list of orders, order status, and download as CSV file		✓	✓
	i. Filters	Allows users to determine orders appearing in order list based on program, schedule, year,		**X**	✓

		period, and supply depot			
	b) Manage POD	Allows user to update proof of delivery information for each order, to change order status to received		✓	✓
	i. Filters	Allows users to identify the program and facility (or all facilities) for which they want to view or edit		X	✓
Allocation (Push) Supply Process					
9	**Distribution**			✓	Under development
	a) Manage	Allows the initiation of a supply distribution to a specified delivery zone, for a specified program and period. Load amounts can also be viewed. Data from initiated distributions will be cached on the device (laptop or tablet) for offline access to the "Record Data" function		✓	Under development
	b) Record Data	Allows user to record logistics and program data for all facilities included in an initiated distribution.		✓	Under development

		This feature is available offline			
	c) Sync	Allows data collected offline to be synchronized with the LMIS online system		✓	Under development
10	**Vaccine**			**X**	Under development
	a) Create/Submit Report			**X**	Under development
	b) Distributions			Feature 9	Under development
	1) Manage	Allows the initiation of a supply distribution to a specified delivery zone, for a specified program and period. Load amounts can also be viewed. Data from initiated distributions will be cached on the device (laptop or tablet) for offline access to the "Record Data" function		Feature 9	Under development
	2) Record Data	Allows user to record logistics and program data for all facilities included in an initiated distribution. This feature is available offline		Feature 9	Under development
	3) Sync	Allows data collected offline to be		Feature 9	Under development

		synchronized with the LMIS online system			
	c) Vaccine Report	Allows viewing of logistics and program vaccine data by selected facility		X	Under development
	d) Vaccine Inventory			X	Under development
	1) Receive	Allows recording of vaccine delivery and product data for creation of receipts		X	Under development
	2) Distribute	Allows creation of distribution and recording of data		X	Under development
Equipment Management					
11	**Equipment**			X	✓
	a) Equipment Inventory	Allows management of equipment inventory based on facility and program		X	✓
	b) Maintenance Requests	Displays existing Maintenance requests and allows updating of existing requests		X	✓
	c) Service Contracts	Allows recording of service contract information		X	✓
Administration					
12	**Reports**			✓	✓
	General	Allows user to select a report for download/view in PDF, XLS,		✓	X

		CSV, or HTML formats. New reports can also be added through upload			
	a) Admin	Hard coded administrative reports available for view or download in PDF or Excel format		X	✓
	i. Facility List	List of facilities and basic information. Can be filtered by program, geographic zone, facility type, and status. Mailing labels can be printed for the selected facilities		X	✓
	ii. User Summary	Displays proportion of users in various functions. Can be filtered by supervisory node, role, and program		X	✓
	b) Consumption	Hard coded consumption reports available for view or download in PDF or Excel format		X	✓
	i. Aggregate Consumption	Displays consumption information by program, schedule, year, and period. Can be filtered by geographic		X	✓

		zone, product category, and product			
	ii. Consumption Average by Product	Displays consumption of products by period type and date range. Can be filtered by requisition group, zone, facility type, facility, product category, and product(s)		X	✓
	iii. District Consumption Comparison	Displays consumption of products by district based on program, schedule, year, and period. Can be filtered by geographic zone, product category, and product		X	✓
	iv. Export to PipeLine	Creates CSV or Excel export file based on program, schedule, year, and period		X	✓
	c) Equipment Reports	Hard coded equipment reports available for view or download in PDF or Excel format		X	✓
	i. Lab Equipment List	Lists lab equipment based on operational status by program. Can be filtered by		X	✓

		geographic zone, facility type, facility, equipment type, and label equipment			
	ii. Lab Equipments by Funding Source	Lists lab equipment based on funding source by program. Can be filtered by geographic zone, donor, facility type, facility, equipment type, and label equipment		X	✓
	iii. Lab Equipments by Location	Displays map with markers of lab equipment based on locations and operational status by program. Can be filtered by geographic zone, facility type, facility, equipment type, and label equipment. Information about service contract status also available		X	✓
	d) Order Fulfillment	Hard coded order fulfillment reports available for view or download in PDF or Excel format		X	✓
	i. District Order Compilation	Displays order compilation by		X	✓

		district based on program, year, and period. Can be filtered by schedule and geographic zone			
	ii. Order Fill Rate Report Summary	Displays chart of order fill rates by program, schedule, year, and period. Can be filtered by geographic zone and facility type		X	✓
	iii. Order Fill Rate Report by Facility	Displays chart of order and item fill rates by program, schedule, year, period, and facility. Can be filtered by geographic zone, facility type, product category, and product		X	✓
	iv. Order Report	Creates voucher for medical supplies		X	✓
	v. Report and Requisition Feedback	Lists requisition and inventory information for products based on program, schedule, year, and period. Can be filtered by order type, geographic zone, facility type, facility, and product		X	✓
	vi. Seasonality/Rationing	Displays		X	✓

	Adjustments	information on seasonality or rationing adjustments made by program. Can be filtered by geographic zone, product category, and product.			
	e) Regimen Report	Hard coded regimen reports available for view or download in PDF or Excel format		X	✓
	i. Aggregate Regimen	Displays aggregate information of patients on, to initiate, or stopped a treatment regimen by program, schedule, year, and period. Can be filtered by geographic zone, category, regimen		X	✓
	ii. Regimen Summary	Displays distribution of regimen patients by facility based on program, schedule, year, and period		X	✓
	iii. Regimen Distribution by District	Displays distribution of regimen patients by district based on program, schedule, year, and period		X	✓

	f) Report Status	Hard coded report status reports available for view or download in PDF or Excel format		X	✓
	i. Non Reporting Facilities	Displays chart and list of reporting statistics for facilities based on program, schedule, year, and period. Can be filtered by geographic zone		X	✓
	ii. Reporting Rate	Displays map with color-indication of reporting rate status of districts based on program, schedule, year, and period. Filterable by indicator		X	✓
	iii. Timeliness	Displays information on timeliness of reporting by district based on program, schedule, year, and period, and filterable by geographic zone		X	✓
	g) Stock Keeping	Hard coded stock reports available for view or download in PDF or Excel format		X	✓
	i. Adjustment Summary	List of adjustments		X	✓

		applied for each product based on program, schedule, year, and period, and filterable			
	ii. Stocked Out	List of products stocked out at facilities based on program, schedule, year, and period, and filterable		X	✓
	iii. Stock Imbalance by Facility	List of stock imbalance at facilities based on program, schedule, year, and period, and filterable		X	✓
	iv. Summary Report	List of product stock overall based on program, schedule, year, and period, and filterable		X	✓
	v. Supply Status by Facility	Supply status of facilities based on program, schedule, year, and period, and filterable		X	✓
	vi. Supply Status by Location	Map of product stock by program, district, and period		X	✓
13	**ILS Gateway**	Allows login to LISGateway account		X	✓
14	**Dashboards**			X	✓
	a) Summary	Provides summary information regarding alerts, stock outs,		X	✓

		emergency requisitions, and reporting performance. Filterable by program, schedule, year, period, geographic zone, and product			
	b) Stock Efficiency	Stock efficiency information including alerts and stock outs. Filterable by program, schedule, year, period, status, geographic zone, and product		X	✓
	c) Order Turnaround	Order turnaround information including alerts and stock outs. Filterable by program, schedule, year, period, status, geographic zone, and product		X	✓
	d) Stock Outs	Stock out information. Filterable by program, schedule, year, period, status, geographic zone, and product		X	✓
	e) Notification	Allows user to send e-mail or SMS notification message for various occurrences to		X	✓

		selected facilities			
	f) RnR Status Summary	Summary of RnR information for selected program, schedule, year, period, and geographic zone.		X	✓
15	**Additional Links**			X	✓
	a) About			X	Not Configured
	i. About Us			X	Not Configured
	ii. About eLMIS			X	Not Configured
	iii. About eHealth			X	Not Configured
	iv. About OpenLMIS			X	Not Configured
	b) Distribution System			X	Not Configured
	i. Central Warehouses			X	Not Configured
	ii. Zonal Warehouses			X	Not Configured
	iii. Regions	List of regions		X	✓
	iv. Districts	List of districts		X	✓
	c) Program Commodities			X	Not Configured
	i. ARV Commodities			X	Not Configured
	ii. HIW Commodities			X	Not Configured
	iii. ILS			X	Not Configured
	iv. TB			X	Not Configured
	d) Order Processing			X	Not Configured
	i. RnR Submission			X	Not Configured
	ii. Order Fulfillment			X	Not Configured
	iii. Delivery			X	Not Configured
	iv. Proof-Of-Delivery			X	Not Configured

	e) Reports			X	Not Configured
	i. Stock Status Summary			X	Not Configured
	ii. Consumption Trend			X	Not Configured
	iii. Dashboard			X	Not Configured
	iv. GIS Maps			X	Not Configured
16)	**User Profile**	View and edit basic user information such as contact info, roles, and facilities		X	✓
17)	**Help**	Allows user to select and view available links for help topics		X	✓