

Department of Veterans Affairs

Mobile Health External Development (MHED)

System Design Document



March 2014

Version 1.3

Revision History

Note: The revision history cycle begins once changes or enhancements are requested after the System Design Document has been baselined.

Date	Version	Description	Author
July 2013	1.0	Initial draft	
July 2013	1.1	Edited Draft	
Sept 2013	1.2	Added SDD for Mobile Adapter and user stories, screen mock ups for MBB, SOC, VAR, and VAAC to appendicies	
March 2014	1.3	Merged the HealthAdapter SDD with the MHED SDD. Updated the Appendix B to include the Addendums from the MHED mobile apps for this Increment.	

Artifact Rationale

The System Design Document (SDD) is a dual-use document that provides the conceptual design as well as the as-built design. This document will be updated as the product is built, to reflect the as-built product. Per the Project Management Accountability System (PMAS) Guide, the SDD with conceptual design is required prior to the Milestone 1 Review. The as-built for each delivery must be incorporated prior to the Milestone 2 Review.

Table of Contents

1. Introduction	4
1.1. Purpose of this document.....	4
1.2. Scope	4
1.3. Relationship to Other Plans	5
1.4. Methodology, Tools, and Techniques.....	5
1.5. Constraining Policies, Directives and Procedures	5
1.6. Constraints	6
1.7. Design Trade-offs	6
1.8. User Characteristics	6
1.9. User Problem Statement	6
2. Background	7
2.1. Overview of the System	7
2.2. Overview of the Business Process	7
2.3. Assumptions	8
2.4. Legacy System Retirement	8
3. Conceptual Design.....	8
3.1. Conceptual Application Design	8
3.1.1. Application Context	8
3.1.2. High-Level Application Design	8
3.1.3. Application Locations	10
3.1.4. Application Users	12
3.2. Conceptual Data Design.....	12
3.2.1. Project Conceptual Data Model	12
3.2.2. Database Information	12
3.2.3. User Interface Data Mapping	12
3.3. Conceptual Infrastructure Design	12
3.3.1. System Criticality and High Availability.....	12
3.3.2. Special Technology	13
3.3.3. Technology Locations.....	13
3.3.4. Conceptual Infrastructure Diagram.....	13
4. System Architecture	15
5. Data Design	20
6. Detailed Design	43
6.1. Software Detailed Design.....	43
6.1.1. Conceptual Design	43
6.1.2. Specific Requirements	47

6.2. Communications Detailed Design	48
7. External Interface Design	49
7.1. Interface Architecture.....	49
8. Human-Machine Interface	49
8.1. Interface Design Rules	65
8.2. Inputs	65
8.3. Outputs	65
8.4. Navigation Hierarchy.....	65
9. System Integrity Controls	65
10. Approval Signatures	67
11. Appendix A – Mobile Adapter SDD	68
12. Appendix B – Mobile Apps	118
12.1.Mobile Blue Button (MBB).....	118
12.2.Summary of Care (SOC)	118
12.3.Veteran Appointment Request (VAR).....	Error! Bookmark not defined.
12.4.VA Appointment Clerk (VAAC)	Error! Bookmark not defined.

1. Introduction

The U.S. Department of Veterans Affairs (VA) continually seeks better ways to serve the needs of VA employees and Veterans, who ask for improved delivery of benefits, services, and information, not only for their own behalf but also for their dependents and annuitants, clinicians, and non-clinical care providers. In meeting these needs, the VA must comply with policies and procedures that protect the integrity and privacy of all Veterans as well as the VA.

One aspect of providing relevant care to Veterans stakeholders is the use of Mobile Applications technology. A large and increasing number of mobile public healthcare applications exist. Mobile Health External Development (MHED) allows for the certification of externally developed mobile applications to be released for use in the VA's information technology enterprise.

VA therefore requires an integrated approach to external mobile applications development that provides standardized processes to meet the following needs:

- Managing the certification of externally developed mobile applications for release to VA users
- Establishing and maintaining the Governance process for externally developed mobile applications
- Ensuring quality development, Assessment and Authorization (A&A), user functionality testing, and release of externally developed mobile applications

Without some specific controls to orchestrate external application development and certification, the volume of externally developed applications could result in duplicity and an inability to ensure that VA standards are met. For externally developed mobile applications to provide the expected benefit supporting Veteran services, the application must be acceptable by the end users, enable improvement in workflow, provide value to, healthcare delivery for the patient and/or staff, and meet VA compliance standards.

1.1. Purpose of this document

The purpose of this document is to describe in sufficient detail how the MHED is to be constructed. Since MHED utilizes the VA Mobile Framework (VAMF) to provide entry of mobile applications into the VA environment, this bulk of the MHED system design describes the VAMF and its components that are available for reuse by all mobile applications. Each mobile application will create a small SDD Addendum that describes how that mobile app uses the VAMF (or not) and the specifics of the design of the mobile app without repeating the design of the VAMF.

1.2. Identification

The MHED project will deploy applications into the OIA VA Mobile Framework system registered with RiskVision as a High system designation. There will be multiple mobile applications deployed through this project and each one will describe its design as an addendum to this SDD.

1.3. Scope

VA Office of Information and Technology (OIT) has developed a process to expeditiously release mobile applications received from outside of OIT development. This requires that OIT must provide testing, certification and release to achieve Enterprise-level certification. MHED is a project that manages the release of products to include the testing and certification of the product, but not the actual development of the product. The scope of the system design is to describe the common operating environment in which the mobile applications will execute.

Table 1: Scope Inclusions

Includes
System design of the VAMF operating environment
Software design of the common services provided to mobile applications hosted within VAMF. These common services are referred to as the Health Adapter on the current wiki sites but in the future will be renamed to apply to all mobile applications, not just health applications

Table 2: Scope Exclusion

Excludes
Software services that are only used by one mobile application and hosted with the VAMF are not considered “common services” and are not discussed here
Software components specific to a particular mobile application and not the overall framework are described in addendums and not in this document

1.4. Relationship to Other Plans

The Mobile Application Program (MAP) defines program-level documentation for all mobile projects and this documentation can be found at: [\[REDACTED\]x/eAD9](#)

The MHED project specific documentation can be found on TSPR at [\[REDACTED\]](#)

1.5. Methodology, Tools, and Techniques

The Atlassian suite of project management tools within the Mobile Application Environment (MAE) will be used to track the progress of each externally developed mobile application to include defect tracking. The processes for changes to the project are described in the MAP Change Management Plan found with the MAP program-level documentation.

Mobile applications will be using the MAE testing environments for the testing and compliance review of the applications. These environments will have the MA software installed and operational to support common services used by the mobile apps. The MAE is located in the Terremark VA Dedicated Cloud and thus is external to the VA.

1.6. Constraining Policies, Directives and Procedures

The MHED project will be responsible to ensure externally developed mobile applications are in compliance will Federal and VA policies, directives, and procedures to include PMAS guidelines.

1.7. Constraints

The MHED project has the following constraints:

- The mobile application governance process is external to MHED; MHED must set limits on the number and complexity of mobile apps released through the project
- Legal/regulatory requirements related to Personal Identifiable Information (PII), HIPAA, Privacy Act, FDA decision to regulate mobile applications, NIST guidelines, and VA policy
- Access to systems and VA services are governed by the VA processes.
- Available resources to provide quality, safety, security, privacy, patient safety, software testing, and governance compliance reviews

1.8. Design Trade-offs

The HealthAdapter is a middle-tier system designed to provide easy-to-consume, resource-oriented services, which are used while developing healthcare applications. The HealthAdapter services include security/content related services (authentication, SSO, context); patient clinical data services (ability to fetch patient's clinical data); provider appointment calendar feeds; and metric/audit recording services (ability to monitor usage and performance).

The origin of the HealthAdapter helps to explain some of its design principles. It was originally developed as a proof-of-concept to make consuming NwHIN services easier, or more specifically, an easier authentication model and easier to consume interface. This proof-of-concept focused on the discovery of patients and the retrieval of documents.

The HealthAdapter has been designed to decouple the service endpoints, business logic, and data sources from each other. This allows for multiple service endpoints, like REST vs SOAP, multiple REST resource format, and allows for multiple data sources, such as VistA, CDW, etc. The concept of decoupling data sources makes it easier to utilize the HealthAdapter for different needs.

1.9. User Characteristics

Users can be categorized in the following buckets:

- Veterans
- Caregivers – proxies on behalf of a veteran
- Staff – clinicians, administrators, or other VA staff that use mobile applications to perform their functions

These users will need access to the VAMF both from within the VA network and outside. Veterans and caregivers will access the systems from the Internet and Staff members may access the systems from inside or outside depending on the application characteristics.

1.10. User Problem Statement

The problems vary by mobile application. Please refer to the business needs and user stories for each mobile application deployed through MHED. The VAMF must support a wide variety of applications both veteran / caregiver focused and VA staff focused.

2. Background

2.1. Overview of the System

The VAMF logical architecture is provided in the figure below. The middle component marked VA Mobile Framework provides the infrastructure for MHED mobile apps to be deployed. Devices can be any mobile device including smartphones, tablets, and laptops. Some applications are targeted towards specific devices, while others are web applications that can be run from any browser.

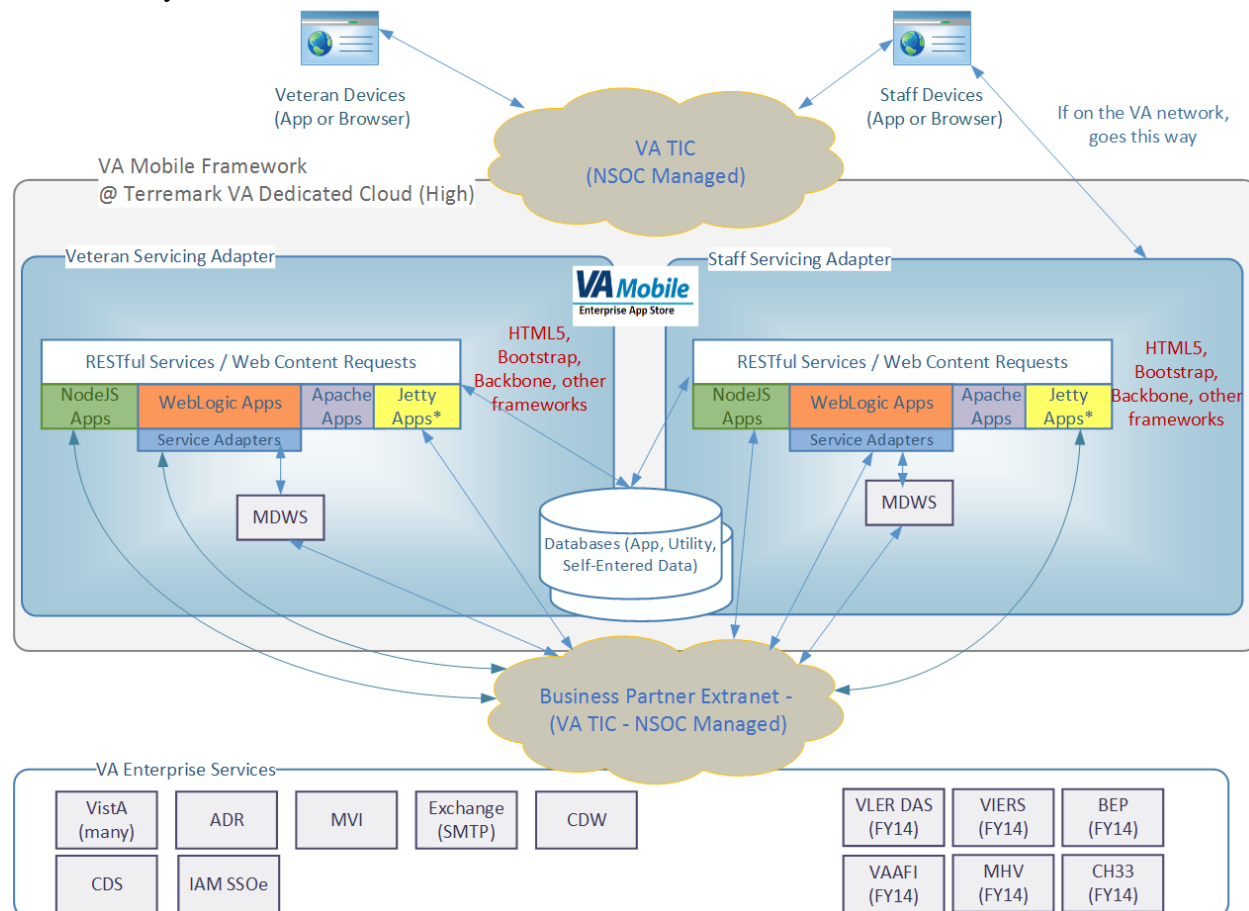


Figure 1: VAMF Logical Architecture

The bottom half of this diagram shows the VA enterprise services used to read and write data from the mobile apps, through the VAMF, and into VA.

2.2. Overview of the Business Process

The business processes are specific to the applications deployed through MHED. Refer to the project wiki sites for more specific information about the business processes and requirements of each. The initial applications are show in the table below with a reference to their project wikis.

Mobile Application	Documentation Link
Burn Pit Registry (web)	x/GwCy
Mobile Blue Button (web)	x/2QFqAQ

Mobile Application	Documentation Link
Summary of Care (web)	x/nAlqAQ
Launchpad (Veteran-web)	x/pwQJAAQ

2.3. Assumptions

- All mobile applications will utilize the VAMF and the mobile applications that generate patient information will store data entered by the veteran on the Patient Generated Database (PGD) datastore
- VA enterprise services will be available for use. Where enterprise services are not available, the mobile application teams will be responsible for building point-to-point interfaces to the desired data systems. The VAMF will not be responsible for providing these services
- The VAMF will be responsible for providing the connection between external Internet facing (or mobile installed) apps into the VA enterprise. This requires opening network connections using the Enterprise System Change Control Board (ESCCB)

2.4. Legacy System Retirement

Not applicable. This does not replace any existing system.

3. Conceptual Design

The MHED conceptual design encompasses the common services provided in the VAMF for the mobile applications.

3.1. Conceptual Application Design

Refer to Figure 1 for the overall context of the system.

3.1.1. Application Context

Table 3: VAMF Objects / Services

Name	Description	Interface Name	Interface System
Apache - Web proxy and static HTML content	This software component within the VAMF provides proxy services for HTTP and HTTPS requests, serves static HTML content, and other web services.	Refer to the individual mobile application addendums	N/A
JEE Container	WebLogic provides a JEE web container for deployment of JEE applications and REST services that provide data to mobile applications	Refer to the individual mobile application addendums	N/A

Name	Description	Interface Name	Interface System
NodeJS	NodeJS applications can be hosted on certain VAMF systems	Refer to the individual mobile application addendums	
Jetty	Jetty/DropWizard technologies can be used to develop self-contained web applications within the VAMF	Refer to the individual mobile application addendums	
MDWS	Medical Domain Web Services (MDWS) is deployed within the VAMF to provide an interface to VA VistA sites	Refer to the MDWS documentation on the Medora.va.gov site or through OSEHRA	VistA

Interfaces External to OIT

Not Applicable. All interfaces are internal to OIT.

Table 4: Interfaces Internal to OIT

Name	Related Object	Input Messages	Output Messages	External Party
VistA	MDWS	Refer to Appendix A for the interfaces	Refer to Appendix A for the interfaces	VistA – all sites
CDW	JEE REST service	Chemistry, Hematology, Microbiology, Problems, Allergies, Medication, Radiology Reports, Appointments (historic), Clinic Note Surgeries, Vitals, Facility Names, Provider Names Requests	Chemistry, Hematology, Microbiology, Problems, Allergies, Medication, Radiology Reports, Appointments (historic), Clinic Note Surgeries, Vitals, Facility Names, Provider Names	CDW
ADR	JEE REST service	Demographics request – REST service call	Demographics – see 11.1.5	Administrative Data Repository (ADR)
MVI	JEE REST service	Patient Correlation request	Patient correlations – where a patient has been seen	Master Veteran Index (MVI)
IAM SSOe	JEE REST service	User Authentication Request	Success / Fail	IAM SSOe
CDS	JEE REST service	Appointments (current) Request	Appointments (current)	CDS to VistA

Name	Related Object	Input Messages	Output Messages	External Party
VIERS	NodeJS application Jetty application	Military History Request Get Claims Appeals	Military Service History Appeal data	VADIR through VIERS TBD
BEP	Jetty application	Find Claims Find Payment Information	Claims data Payment of claims data	BEP – Corporate Database
CH33	Jetty application	Find Chapter 33 claim Find certified enrollments	Claim status Certified Enrollments	CH33
LGY (VAAFI)	Jetty application	Find Loan Information	Loan and payment information	LGY

Table 5: Externally Shared Data Stores

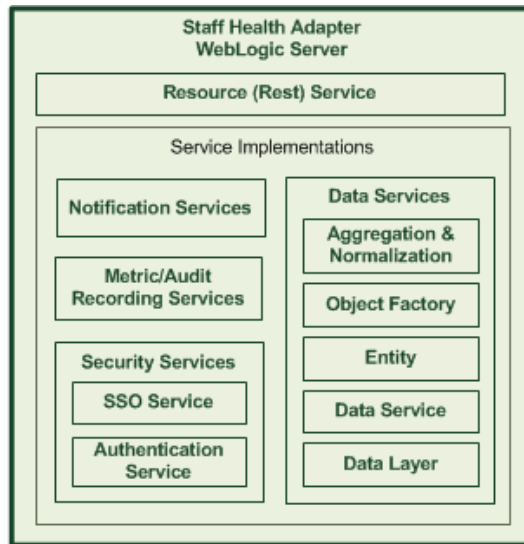
Name	Data Stored	Owner	Access
VLER DAS	Patient Generated Data (PGD) will be stored in the VLER DAS NoSQL database.	Data Access Services	DAS eCRUD web services
Cache	Updates to the health record in VistA are made in some applications through messaging through VA services.	VistA	MDWS provides the services

3.1.2. High-Level Application Design

The **HealthAdapter** has been designed to decouple the service endpoints, business logic, and data sources from each other. This allows for multiple service endpoints, like REST vs SOAP and multiple REST resource formats, and also for multiple data sources, such as Vista, CDW, etc. The concept of decoupling data sources makes it easier to utilize the **HealthAdapter** for different needs.

Clinic-in-hand and the VAI2 2011 efforts will reside in a complex VA architecture, which includes the following: how to authenticate, where to get data, where to store data, etc.

Health Adapter



3.1.3. Application Locations

Mobile applications will be housed in the OIA VA Mobile Framework in the Terremark VA Dedicated Cloud.

Table 6: Application Locations

Application Component	Description	Location at Which Component is Run	Type
Veteran Facing Servers	Veteran facing services are hosted on a set of servers within the VAMF. These servers are a mix of Linux and Windows servers executing a number of web	Terremark Culpeper	Veteran web applications
Staff Facing Servers	Applications that support staff operations are hosted in a separate set of servers within the VAMF. These servers are a mix of Linux and Windows servers executing a number of web. This	Terremark Culpeper	Staff web applications
Database Shared Servers	Both the staff and veteran apps use local storage for utility and application data. This is currently Oracle but migrating to MongoDB under VLER DAS domain	Terremark Culpeper	Database servers

3.1.4. Application Users

Mobile applications will be used by Veterans, Veteran caregivers and clinicians. These applications will improve Veteran health through the empowerment of Veterans and VA staff by using Web and Mobile Technology.

3.2. Conceptual Data Design

3.2.1. Project Conceptual Data Model

The HealthAdapter stores the following data:

- Data created by Veterans gets saved to the VA "Self Entered Database" (SED). This system is external to the HealthAdapter, and thus, from the HealthAdapter's perspective, there is no data-at-rest concern. This Oracle database will be migrated to the DAS PGD database.
- Notifications sent from staff to the Veteran are persisted to the HealthAdapter Repository. Please see the HealthAdapter Repository Data Design Description chapter concerning whether PII, PHI, SPI and/or Sensitive Information is being stored.
- Debug information and raw HTTP are written to log files on the HealthAdapter Server, MDWS Server, and HealthAdapter Load Balancer. This could potentially contain PHI.
- Some data is cached for both the HealthAdapter Server and MDWS Server, but it is only stored to memory on the application server.

3.2.2. Database Information

All the schemas and their descriptions are listed in this section

Schema	Description
AUTHDB	This schema is for the Authorization Server to store OAuth codes and tokens.
EASDB	This schema is for the catalog of mobile apps and the ability of the EAS server to support an Enterprise App Store.
HADB	This schema provides all application support (sending notifications, system metrics, etc.) for Clinic-in-Hand.

3.2.3. User Interface Data Mapping

Please refer to the documentation for each mobile application. The VAMF does not necessarily have a user interface, but each minor application has user interface requirements.

3.3. Conceptual Infrastructure Design

3.3.1. System Criticality and High Availability

The VAMF is built within the Terremark VA Dedicated Cloud at Culpeper, VA in two different enclaves. The development and testing environments are contained within enclave 007 in the general moderate VA cloud at Terremark. This environment is still required to have a 99% availability per the contract. Due to the nature of virtual cloud environments and the backup and

restore capability, individual VMs can be recovered quickly if there is a VM specific issue. For the whole enclave, there is a failover process that is described in the Disaster Recovery Plan with the ATO documentation. The following environments are in the 007 enclave:

- Development
- Test
- Integration
- V&V (future)
- Build and Development Tools

The second enclave is 009 which is a High categorized system and contains PII and PHI. This enclave has a 99.99% availability requirement. There is a backup and cold failover site as described in the VAMF Disaster Recovery Plan at the Terremark backup facility in Miami, FL. The environments hosted in the 009 enclave include:

- Pre-Production
- Production

3.3.2. Special Technology

The system uses standard web technologies for the HealthAdapter and mobile applications. As mobile apps are brought into the environment, any technologies that are not on the TRM are required to have a waiver that must be obtained by the mobile application team.

The VAMF has a waiver to use MongoDB until it migrates to the VLER DAS NoSQL solution.

3.3.3. Technology Locations

3.3.4. Conceptual Infrastructure Diagram

3.3.4.1. Location of Environments

The following figure shows the two enclaves and the networks accessed within each. Note the Business Process Extranet should be labeled Business Partner Extranet (BPE). The traffic that traverses the BPE is controlled by the VA NSOC and ports must be explicitly opened between the VAMF and other VA systems.

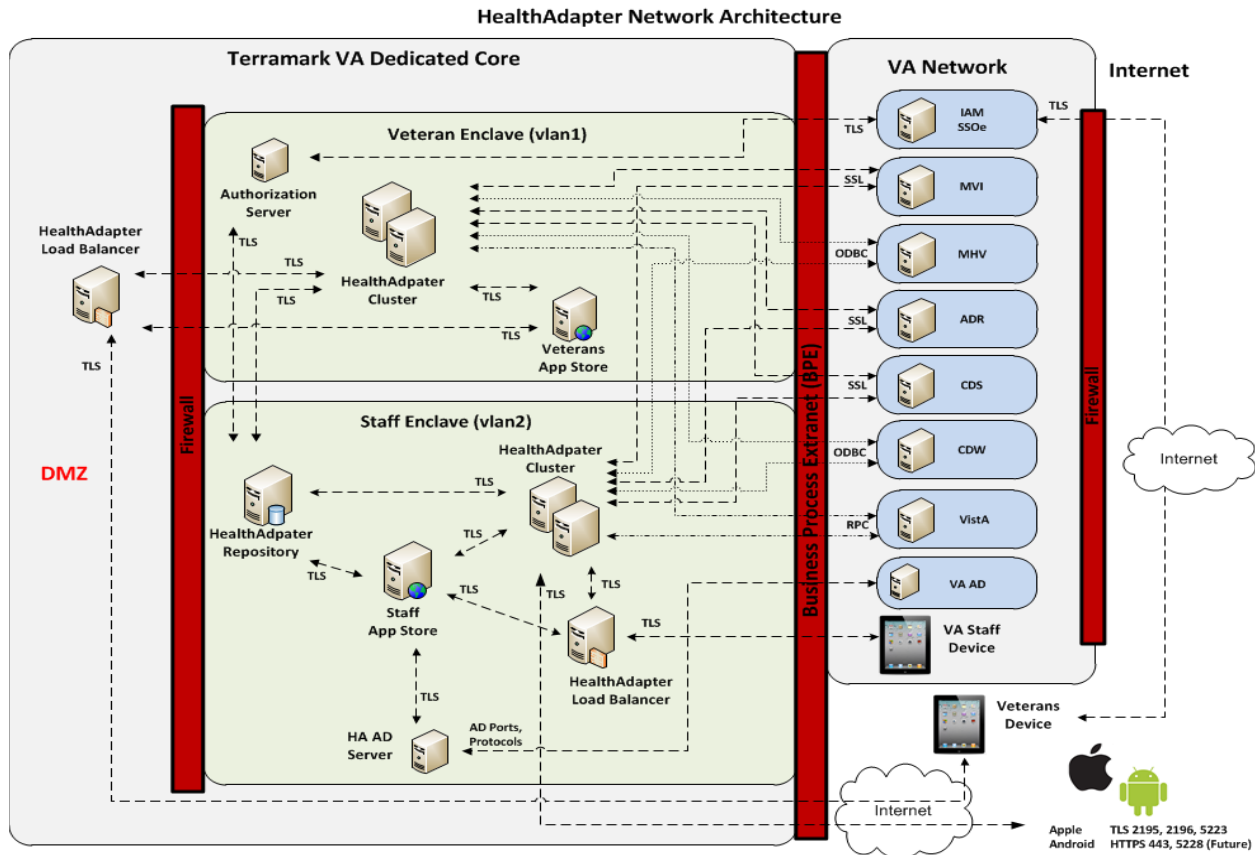


Figure 2: Network Interfaces

3.3.4.2. Conceptual Production String Diagram

The HealthAdapter System may be composed of 1 or many System Instances. Multiple System Instances are not for reasons of performance of scalability but to support various end user/device/environment requirements. The implementation or deployment cited in this documented resulted in multiple instances and for each instance to share application data or application state. This led to a share repository.

While a System Instance only contains a single HealthAdapter Cluster, this "application cluster" can contain 1 or many HealthAdapter nodes to address performance plans. A "HealthAdapter Node" consists of a HealthAdapter application server as well as an IIS/MDWS application server. A collection of these nodes is referred to as the "HealthAdapter Cluster". From a traditional application server perspective, there is also the concept of a cluster of WebLogic application servers and each individual HealthAdapter node represents a single instance of the application (each WebLogic application server runs an instance of the HealthAdapter).

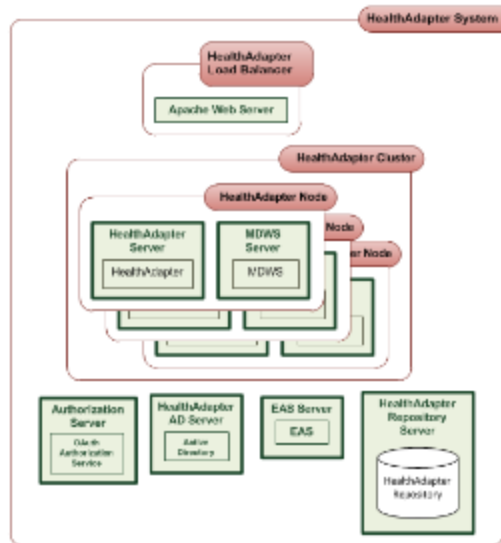


Figure 3: Conceptual Architecture

4. System Architecture

The MHED project will be responsible for ensuring all externally developed mobile applications comply with the PMAS requirements. Please refer to Appendix A.

. All of the following servers that make up the production environment reside at Terremark.

4.1. Hardware Architecture

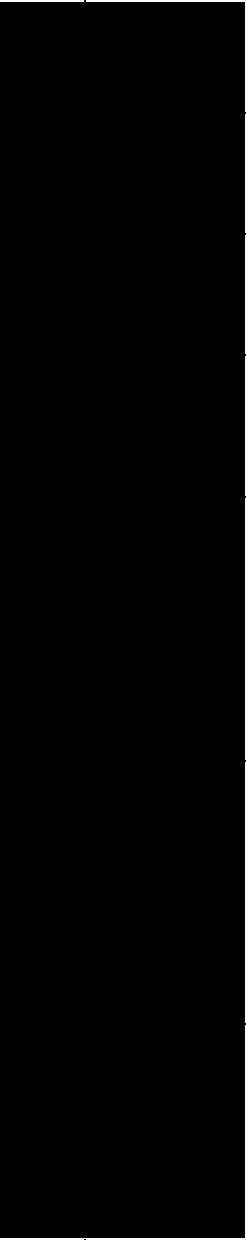
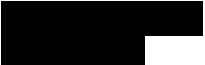
4.1.1. Production Veteran Environment

Server Name	Operating System	Installed components	Purpose	External Connections
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	Used to authenticate the credentials of the Veteran devices	IAM SSOe
	RedHat Enterprise Linux 5.9	Apache Web Server 2.2	HealthAdapter web server/load balancer	Web Address: [REDACTED]
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	Serves as a master server in which controls the HealthAdapter servers within the cluster	
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	Host the Enterprise App Store for the Veterans	
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	This server hosts the HealthAdapter and handles processes dealing with it.	CDS MVI

Server Name	Operating System	Installed components	Purpose	External Connections
			Part of a cluster in which communicates with external VA systems to pull information needing by the application based on user requests.	ADR
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	This server hosts the HealthAdapter and handles processes dealing with it. Part of a cluster in which communicates with external VA systems to pull information needing by the application based on user requests.	CDS MVI ADR
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	This server hosts the HealthAdapter and handles processes dealing with it. Part of a cluster in which communicates with external VA systems to pull information needing by the application based on user requests.	CDS MVI ADR
	Windows Server 2008	.NET Framework v 4.0 Web Service Enhancements (WSE) v 3.0 Internet Information Services Medical Domain Web Services (MDWS) v 2.8.10	Hosts the MDWS environment in which connects to multiple VA resources to pull information to provide to the Veterans. Also communicates with the Health Adapter cluster to organize and deliver the requested information.	CDW MHV VistA
	Windows Server 2008	.NET Framework v 4.0 Web Service Enhancements (WSE) v 3.0 Internet Information Services Medical Domain Web Services (MDWS) v 2.8.10	Hosts the MDWS environment in which connects to multiple VA resources to pull information to provide to the Veterans. Also communicates with the Health Adapter cluster to organize and deliver the requested information.	CDW MHV VistA
	Windows Server 2008	.NET Framework v 4.0 Web Service Enhancements (WSE) v 3.0 Internet Information Services Medical	Hosts the MDWS environment in which connects to multiple VA resources to pull information to provide to the Veterans. Also communicates with the Health Adapter cluster to organize	CDW MHV VistA

Server Name	Operating System	Installed components	Purpose	External Connections
		Domain Web Services (MDWS) v 2.8.10	and deliver the requested information.	

4.1.2. Production Staff Environment

Server Name	Operating System	Installed components	Purpose	External Connections
	RedHat Enterprise Linux 5.9	Apache Web Server 2.2	HealthAdapter web server/load balancer	Web Address: 
	RedHat Enterprise Linux 5.9	Oracle 11G release 2 (11.2.0.3)	Hosts an Oracle database in which the HealthAdapter uses to pull information from.	
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	Serves as a master server in which controls the HealthAdapter servers within the cluster	
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	Hosts the Enterprise App Store for the Staff	
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	This server hosts the HealthAdapter and handles processes dealing with it. Part of a cluster in which communicates with external VA systems to pull information needing by the application based on user requests.	CDS MVI ADR
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	This server hosts the HealthAdapter and handles processes dealing with it. Part of a cluster in which communicates with external VA systems to pull information needing by the application based on user requests.	CDS MVI ADR
	RedHat Enterprise Linux 5.9	WebLogic Server v 10.3.5 JDK v 1.6.0_24	This server hosts the HealthAdapter and handles processes dealing with it. Part of a cluster in which communicates with external VA systems to pull information needing by the application based on user requests.	CDS MVI ADR

Server Name	Operating System	Installed components	Purpose	External Connections
	Windows Server 2008	.NET Framework v 4.0 Web Service Enhancements (WSE) v 3.0 Internet Information Services (IIS) Medical Domain Web Services (MDWS) v 2.8.10	Hosts the MDWS environment in which connects to multiple VA resources to pull information to provide to the Staff. Also communicates with the Health Adapter cluster to organize and deliver the requested information.	CDW MHV VistA
	Windows Server 2008	.NET Framework v 4.0 Web Service Enhancements (WSE) v 3.0 Internet Information Services (IIS) Medical Domain Web Services (MDWS) v 2.8.10	Hosts the MDWS environment in which connects to multiple VA resources to pull information to provide to the Staff. Also communicates with the Health Adapter cluster to organize and deliver the requested information.	CDW MHV VistA
	Windows Server 2008	.NET Framework v 4.0 Web Service Enhancements (WSE) v 3.0 Internet Information Services (IIS) Medical Domain Web Services (MDWS) v 2.8.10	Hosts the MDWS environment in which connects to multiple VA resources to pull information to provide to the Staff. Also communicates with the Health Adapter cluster to organize and deliver the requested information.	CDW MHV VistA

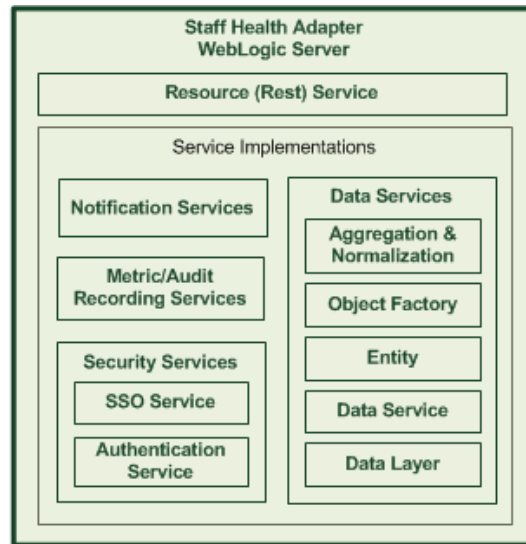
4.1.3. Burn Pit Production Environment

Server Name	Operating System	Installed Components	Purpose	External Connections
	RHEL 6	Apache	Apache Proxy	
	RHEL 6	NodeJS, Nginx,	NodeJS	
	RHEL6	MongoDB	MongoDB	
	RHEL 6	MongoDB	MongoDB	
	RHEL 6	MongoDB, MongoSMS	Arbiter	

4.2. Software Architecture

The following describes the software architecture shown in the figure repeated from above.

Health Adapter



4.2.1. DTO

The Domain Transfer Objects, or abbreviated as DTO's, are a set of java classes that represent the domain/business objects of the system. These classes exist to pass content between the layers of the system, including from the HealthAdapter to the consumer.

4.2.2. Resource Service

From a Resource Oriented Architecture / REST perspective, a resource is the entity that is being exposed and made available as a service. In the HealthAdapter, there are a series of "domain" resource services. These classes define the REST services available from the system, including the paths, HTTP verbs in order to access them, and the representation (format).

4.2.3. Data Links

Within the HealthAdapter, the resources returned from the resource services contain links to other resource services within the system. For example, when fetching a patient in the system, the patient returned contains links to resources services relevant to that patient's health conditions and history, such as that patient's allergies, lab results, etc.

4.2.4. Data Service

The data services in the system are the layer responsible for expose data related services. The data services are generally invoked from the resource services. When invoked, the data services initiate the correct data layer component, translate the patient identifier as needed, and then invoke the data layer. If there are any business rules, these are generally encapsulated in this layer. The data services, like the resources services, are broken out by domain.

4.2.5. Data Layer

The data layer is the layer responsible for the fetching and saving data. For each domain, there is a data layer interface, with an implementation for this data layer interface per source system (VistA, CDW, etc.). There is generally a "mock" data layer implementation used during testing.

4.2.6. Service Registry

The service registry's primary purpose is to provide factory methods for the data layer implementations. The structure of the service registry allows for different data layer implementations to be mapped to different "communities". A community can represent a single logical "data source", such as a Vista instance or it can represent an aggregated logical "data source", such as the VA or the DoD.

5. Data Design

5.1. .DBMS Files

5.1.1. AUTHDB Tables

5.1.1.1. OAUTH_CODE

Column Name	Data Type	Nullable	Description	PK	FK
CODE	varchar(255)	N	Auth Code granted to client	Y	N
AUTHENTICATION	BLOB	Y	Serialized OAuth Authentication object	N	N

5.1.1.2. OAUTH_ACCESS_TOKEN

Column Name	Data Type	Nullable	Description	PK	FK
TOKEN_ID	varchar(255)	N	OAuth Access Token Id	Y	N
TOKEN	BLOB	Y	Serialized OAuth Token Object	N	N
AUTHENTICATION_ID	varchar(255)	Y	Authentication Id	N	N
USER_NAME	varchar(255)	Y	User name of the logged in user	N	N
CLIENT_ID	varchar(255)	Y	Client ID of the application requested access to user resources	N	N
AUTHENTICATION	BLOB	Y	Serialized OAuth Authentication object	N	N

5.1.1.3. OAUTH_REFRESH_TOKEN

Column Name	Data Type	Nullable	Description	PK	FK
TOKEN_ID	varchar(255)	N	OAuth Refresh Token Id	Y	N
TOKEN	BLOB	Y	Serialized OAuth Refresh Token Object	N	N

Column Name	Data Type	Nullable	Description	PK	FK
AUTHENTICATION	BLOB	Y	Serialized OAuth Authentication object	N	N

5.1.1.4. ACCESS_TOKEN_USER

Column Name	Data Type	Nullable	Description	PK	FK
USER_NAME	varchar(255)	Y	User Name of the Logged in User	N	N
ACCESS_TOKEN	varchar(255)	Y	OAuth Access Token issued	N	N
REFRESH_TOKEN	varchar(255)	Y	OAuth Refresh Token issued	N	N

5.1.2. EASDB Tables

5.1.2.1. APPMETADATA

Column Name	Data Type	Nullable	Description	PK	FK
ID	number(10,0)	N	Generated primary key	Y	N
NAME	varchar2(45)	N	Application name	N	N
TAG	varchar2(200)	Y	Application tag	N	N
VERSION	varchar2(20)	N	Application version	N	N
DESCRIPTION	varchar2(200)	Y	Application description	N	N
ICONURL	varchar2(100)	Y	URL to icon file	N	N
POSTDATE	date	Y	Posted date	N	N
DOWNLOADURL	varchar2(100)	Y	URL to application download	N	N
DEVICETYPE	varchar2(200)	N	Compatible device type	N	N
ENTERPRISEAPP	number(1,0)	Y	Whether application is an enterprise application	N	N
IPAFILENAME	varchar2(100)	Y	Name of the ipa (ios application binary) file	N	N
LASTUPDATED	date	N	Last updated date	N	N
UPDATEDBY	varchar2(45)	Y	Username that last updated the record	N	N

Column Name	Data Type	Nullable	Description	PK	FK
IPAFILE	blob	Y	ios application binary	N	N
ICON	blob	Y	Icon binary	N	N
APPSTATUS	varchar2(20)	N	Application status	N	N
CATEGORY	varchar2(20)	N	Application category	N	N
SCHEME	varchar2(45)	Y	ios app scheme	N	N
TYPE	varchar2(45)	Y	Identifier to specify whether it is an app or a web link	N	N

5.1.2.2. APPROLE

Column Name	Data Type	Nullable	Description	PK	FK
ID	number(10,0)	N	Generated primary key	Y	N
APPID	number(10,0)	N	Foreign key reference to application	N	Y
NAME	varchar2(45)	N	Role name	N	N
DESCRIPTION	varchar2(45)	Y	Role description	N	N

5.1.2.3. IPADATA

Column Name	Data Type	Nullable	Description	PK	FK
ID	number(10,0)	N	Generated primary key	Y	N
APPID	number(10,0)	N	Foreign key reference to application	Y	Y
IPA	blob	N	ios application binary	N	N

5.1.2.4. WEBAPPMETADATA

Column Name	Data Type	Nullable	Description	PK	FK
ID	number(10,0)	N	Generated primary key	Y	N
NAME	varchar2(45)	Y	Name of the web link	N	N
TAG	varchar2(200)	Y	Application tag	N	N
DESCRIPTION	varchar2(200)	Y	Web link description	N	N
SCHEME	varchar2(45)	Y	application scheme	N	N
TYPE	varchar2(45)	Y	Identifier to specify whether it is an app	N	N

Column Name	Data Type	Nullable	Description	PK	FK
			or a web link		
LAUNCHPATH	varchar2(200)	Y	Url of the web link	N	N
ICONNAME	varchar2(45)	Y	Icon Name	N	N

5.1.3. HADB Tables

5.1.3.1. APPOINTMENT_METRICS

Used by: Scheduling, Appointment Request

Column Name	Data Type	Nullable	Description	PK	FK
NAME	varchar2(100)	N	Name of metric	YES	N
VALUE	decimal(9,0)	N	Value of metric	N	N

5.1.3.2. APPOINTMENT_REQUEST

Used by: Scheduling, Appointment Request

Column Name	Data Type	Nulla ble	Description	PK	FK
APPOINTMENT_REQUEST_ID	varchar2(32)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	N	System ID of patient requesting appointment. Usually PATIENT_ID means System ID of the user.	N	N
LAST_UPDATED_DATE	date	N	Date appointment request was last updated	N	N
CREATED_DATE	date	N	Date appointment request was created	N	N
DELETED_DATE	date	Y	Date appointment request was deleted	N	N
ACTIVE	number(1,0)	N	Is request active?	N	N
FIRST_NAME	varchar2(100)	N	Patient's first name	N	N
LAST_NAME	varchar2(100)	N	Patient's last name	N	N

Column Name	Data Type	Nulla ble	Description	PK	FK
SECOND_REQUEST	number(1,0)	N	Is request a second attempt?	N	N
APPOINTMENT_DATE	varchar2(50)	Y	Booked date	N	N
APPOINTMENT_TIME	varchar2(50)	Y	Booked time	N	N
OPTION_DATE_1	varchar2(50)	N	First appointment date preference	N	N
OPTION_TIME_1	varchar2(50)	N	First appointment time preference	N	N
OPTION_DATE_2	varchar2(50)	N	Second appointment date preference	N	N
OPTION_TIME_2	varchar2(50)	N	Second appointment time preference	N	N
OPTION_DATE_3	varchar2(50)	N	Third appointment date preference	N	N
OPTION_TIME_3	varchar2(50)	N	Third appointment time preference	N	N
STATUS	varchar2(255)	N	Status of request	N	N
APPOINTMENT_TYPE	varchar2(255)	N	Requested appointment type	N	N
FACILITY_CODE	varchar2(100)	N	Requested facility's site code	N	N
EMAIL	varchar2(255)	N	Patient's email	N	N
PHONE_NUMBER	varchar2(255)	N	Patient's phone number	N	N
PURPOSE_OF_VISIT	varchar2(255)	N	Patient's purpose of visit	N	N
OTHER_PURPOSE_OF_VISIT	varchar2(100)	Y	Other purpose of visit specified by user	N	N
VISIT_TYPE	varchar2(40)	N	Requested visit type	N	N
PROVIDER_ID	varchar2(255)	N	Requested provider's ID	N	N
PROVIDER_NAME	varchar2(255)	N	Requested provider's	N	N

Column Name	Data Type	Nulla ble	Description	PK	FK
			name		
PROVIDER_PERSON_CLASS	varchar2(255)	Y	Requested provider's person class	N	N
PROVIDER_OPTION	varchar2(100)	Y	Additional information for clerk to book best provider	N	N
SECOND_REQUEST_SUBMITTED	number(1,0)	N	If this is a first request, has second request been submitted?	N	N
PARENT_REQUEST_ID	varchar2(32)	Y	If this is a second request, ID of parent request	N	N

5.1.3.3. APPT_REQ_INPROCESS

Used by: [Scheduling, Appointment Request](#)

Column Name	Data Type	Nullable	Description	PK	FK
APPT_REQ_ID	varchar2(32)	N	Appointment request ID that is in process	YES	N
USER_ID	varchar2(255)	N	System ID of user that is processing the request	N	N
FIRST_NAME	varchar2(255)	N	First name of user that is processing the request	N	N
LAST_NAME	varchar2(255)	N	Last name of user that is processing the request	N	N

5.1.3.4. AR_DETAIL_CODE

Used by: [Scheduling, Appointment Request](#)

Column Name	Data Type	Nullable	Description	PK	FK
AR_DETAIL_CODE_ID	varchar2(32)	N	Generated primary key	YES	N
DETAIL_CODE_ID	varchar2(32)	N	ID of associated detail code	N	YES
APPOINTMENT_REQUEST_ID	varchar2(32)	N	ID of associated	N	YES

Column Name	Data Type	Nullable	Description	PK	FK
			appointment request		
CREATED_DATE	varchar2(255)	N	Date detail code was added to appointment request	N	N
USER_ID	varchar2(255)	N	ID of user that associated detail code with appointment request	N	N

5.1.3.5. ASSESSMENT_RESULT

Used by: HealthAssessment, Care4Caregiver, VAPTSDCoach, VAPainCoach

Column Name	Data Type	Nullable	Description	PK	FK
ASSESSMENT_RESULT_ID	varchar2(32)	N	Generated primary key	YES	N
UNIQUE_TITLE	varchar2(255)	N	Title of assessment	N	N
VERSION	varchar2(255)	N	Version of assessment	N	N
AUTHENTICATION_STRATEGY	varchar2(255)	N	Type of authentication	N	N
PATIENT_ID	varchar2(255)	Y	ID of user that took the assessment	N	N
NOTES	varchar2(4000)	Y	Description of the assessment	N	N
DATE_TAKEN	date	Y	Date assessment was taken	N	N
SCORE	number(11,0)	Y	Score of assessment	N	N
RESPONSES	clob	Y	Responses in XML format	N	N
IN_PROGRESS	number(1,0)	Y	Is assessment in progress?	N	N
ASSESSMENT_ID	varchar2(255)	Y	ID of assessment	N	N

Column Name	Data Type	Nullable	Description	PK	FK
			taken		
SCORING_ALGORITHM	varchar2(255)	Y	Scoring algorithm used	N	N
WAITING_PERIOD	decimal(11,0)	N	Waiting period in days	N	N
REPORT	clob	Y	Report in XML format	N	N

5.1.3.6. AUDIT_LOG

Used by: Notifications

Column Name	Data Type	Nullable	Description	PK	FK
AUDIT_ID	varchar2(32)	N	Generated primary key	YES	N
USER_ID	varchar2(255)	Y	System ID of user	N	N
AUDIT_TYPE	varchar2(255)	Y	An optional type associated with the Audit	N	N
AUDIT_SUBTYPE	varchar2(255)	Y	An optional sub-type associated with the Audit	N	N
AUDIT_DATE	date	N	Date the Audit was taken	N	N
DETAILS	clob	Y	Optional details associated with the Audit	N	N

5.1.3.7. BEST_TIME_TO_CALL

Used by: Scheduling, Appointment Request

Column Name	Data Type	Nullable	Description	PK	FK
APPOINTMENT_REQUEST_ID	varchar2(32)	N	ID of associated appointment request	YES	N
BEST_TIME	varchar2(40)	N	Time range	YES	N

5.1.3.8. CALENDAR_EVENTS

Used by: Notifications and Reminders

Column Name	Data Type	Nullable	Description	PK	FK
ID	varchar2(32)	N	Primary key	Y	N

Column Name	Data Type	Nullable	Description	PK	FK
PATIENT_ID	varchar2(255)	Y	The patient associated with this calendar event	N	N
TITLE	varchar2(255)	Y	The title for the calendar event	N	N
NOTES	clob	Y	The note for the calendar event	N	N
START_DATE	date	Y	The date for the calendar event	N	N
DURATION	number(11,0)	Y	The duration	N	N
ALERT_DURATION	number(11,0)	Y	The alert duration	N	N

5.1.3.9. COMMUNICATIONS

Used by: Journal

Column Name	Data Type	Nullable	Description	PK	FK
ID	varchar2(32)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	Y	Patient associated with the communications entry	N	N
LOG_TIME	date	Y	The date the communications entry was entered	N	N
CONTACT	varchar2(255)	Y	The name of the person who was contacted	N	N
CONTACT_TYPE	varchar2(255)	Y	How the user contacted the person	N	N
SUBJECT	varchar2(255)	Y	The subject of the communications entry	N	N
NOTE	clob	Y	A note describing the communications entry	N	N

5.1.3.10. DAILY_EVENT

Used by: Journal

Column Name	Data Type	Nullable	Description	PK	FK
ID	varchar2(32)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	Y	Patient associated with the daily event	N	N
TITLE	varchar2(255)	Y	The title of the daily event	N	N

Column Name	Data Type	Nullable	Description	PK	FK
ENTRY_DATE	date	Y	The date the daily event was entered	N	N
NOTE	clob	Y	A note describing the daily event	N	N

5.1.3.11. DATABASECHANGELOG

Not used by an application. Used for database migration tracking

Column Name	Data Type	Nullable	Description	PK	FK
ID	varchar2(63)	N	ID of changeset	YES	N
AUTHOR	varchar2(63)	N	Author of changeset	YES	N
FILENAME	varchar2(200)	N	File changeset is located in	YES	N
DATEEXECUTED	date	N	Date changeset was executed	N	N
ORDEREXECUTED	number(11,0)	N	Order of when changeset was executed	N	N
EXECTYPE	varchar2(10)	N	Result of execution	N	N
MD5SUM	varchar2(35)	Y	MD5 hash of changeset	N	N
DESCRIPTION	varchar2(255)	Y	Description of changeset	N	N
COMMENTS	varchar2(255)	Y	Comments in changeset	N	N
TAG	varchar2(255)	Y	Tag in changeset	N	N
LIQUIBASE	varchar2(20)	Y	Liquibase version	N	N

5.1.3.12. DATABASECHANGELOGLOCK

Not used by an application. Used for database migration tracking

Column Name	Data Type	Nullable	Description	PK	FK
ID	number(11,0)	N	ID of lock	YES	N
LOCKED	number(1,0)	N	Is database locked?	N	N
LOCKGRANTED	date	Y	Date lock was granted	N	N
LOCKEDBY	varchar2(255)	Y	Entity that owns lock	N	N

5.1.3.13. DETAIL_CODE

Used by: Scheduling, Appointment Request

Column Name	Data Type	Nullable	Description	PK	FK
DETAIL_CODE_ID	varchar2(32)	N	Generated primary key	Y	N
PROVIDER_MESSAGE	varchar2(2000)	N	System ID of user	N	N
VETERAN_MESSAGE	varchar2(2000)	N	Date the device was first registered	N	N

5.1.3.14. DEVICE_REGISTRATION

Used by: Notifications

Column Name	Data Type	Nullable	Description	PK	FK
DEVICE_REGISTRATION_ID	varchar2(32)	N	Generated primary key	Y	N
USER_ID	varchar2(255)	N	System ID of user	N	N
REGISTERED_DATE	date	N	Date the device was first registered	N	N
DEVICE_TOKEN	varchar2(100)	N	Device token used for Apple Push Notification Service	N	N
FIRST_NAME	varchar2(255)	Y	First name of user for sending personalized notifications	N	N
LAST_NAME	varchar2(255)	Y	Last name of user	N	N
OPTED_IN_FLAG	number(1,0)	N	Whether device is opted in to receive messages	N	N

5.1.3.15. DIET

Used by: Journal

Column Name	Data Type	Nullable	Description	PK	FK
ID	char(36)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	Y	Patient associated with the diet entry	N	N
MEAL_TYPE	varchar2(50)	Y	The type of meal (breakfast, lunch, dinner, snack)	N	N
CALORIES	varchar2(20)	Y	The number of calories in the diet entry	N	N
FAT	varchar2(20)	Y	The amount of fat in the diet entry	N	N

Column Name	Data Type	Nullable	Description	PK	FK
CARBS	varchar2(20)	Y	The amount of carbohydrates in the diet entry	N	N
PROTEIN	varchar2(20)	Y	The amount of protein in the diet entry	N	N
ENTRY_DATE	date	Y	The date the diet entry was entered	N	N
NOTE	clob	Y	A note describing the diet entry	N	N

5.1.3.16. DOCUMENTS

Used by: Journal

Column Name	Data Type	Nullable	Description	PK	FK
ID	varchar2(32)	N	The primary key	Y	N
ENTRY_ID	varchar2(32)	Y	The entry ID	N	N
PATIENT_ID	varchar2(255)	Y	The patient ID	N	N
TITLE	varchar2(255)	Y	The title of the document	N	N
CLASS_CODE	varchar2(255)	Y	The class code of the document	N	N
MIME_TYPE	varchar2(255)	Y	The mime type of the document	N	N
URI	varchar2(500)	Y	The URI of the document	N	N
DOCUMENT	blob	Y	The document	N	N

5.1.3.17. ENTRIES

Column Name	Data Type	Nullable	Description	PK	FK
ENTRY_ID	varchar2(32)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	Y	Patient associated with the event	N	N
COMPLETED	number(1,0)	N	Flag whether the entry is completed	N	N
DATE_COMPLETED	date	Y	The date the entry was completed	N	N
DATE_CREATED	date	N	The date that the entry was	N	N

5.1.3.18. EVENT

No longer used; will be removed in future releases

Column Name	Data Type	Nullable	Description	PK	FK
EVENT_ID	varchar2(32)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	N	Patient associated with the event	N	N
TYPE	varchar2(255)	Y	Optional Type associated with the record	N	N
SUBTYPE	varchar2(255)	Y	Optional Subtype associated with the record	N	N
EVENT_DATE	date	N	The date that the event occurred	N	N

5.1.3.19. EVENT_OBSERVATION

No longer used; will be removed in future releases

Column Name	Data Type	Nullable	Description	PK	FK
EVENT_OBSERVATION_ID	varchar2(32)	N	Generated Primary Key	YES	N
TITLE	varchar2(255)	Y	An optional Title for the observation	N	N
TYPE	varchar2(255)	Y	Optional Type associated with the record	N	N
NOTE	varchar2(255)	Y	Optional Note field for user input	N	N
OBSERVATION_DATE	date	N	The date that the event occurred	N	N
EVENT_ID	varchar2(32)	N	A FK to the associated Event	N	YES

NOTE = Yes because although this is an optional note field for user input, there are no constraints on the type of information it could contain and because of the context of how it is used, it can contain data that is any combination of PII/PHI/SPI/Sensitive Information

5.1.3.20. EXERCISE

Used by: Journal

Column Name	Data Type	Nullable	Description	PK	FK
ID	varchar2(32)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	Y	Patient associated with the exercise entry	N	N

Column Name	Data Type	Nullable	Description	PK	FK
DURATION	number(11,0)	Y	The length of time the exercise was done, in minutes	N	N
DISTANCE	float(24)	Y	The distance traveled during the exercise, in miles	N	N
INTENSITY	varchar2(255)	Y	The intensity of the exercise (mild, moderate, or strenuous)	N	N
ACTIVITY	varchar2(255)	Y	The type of exercise (walking, running, bicycling, or weight lifting)	N	N
ACTIVITY_DATE	date	Y	The date the exercise occurred	N	N
NOTE	clob	Y	A note describing the exercise	N	N

5.1.3.21. FACILITY

This is no longer used; it will be removed in future releases

Column Name	Data Type	Nullable	Description	PK	FK
NAME	varchar2(100)	N	Name of facility	YES	N
TYPE	varchar2(100)	Y	Type of facility	N	N
FACILITY_CODE	varchar2(100)	N	Facility's site code	N	N
STATE	varchar2(50)	Y	Facility's state	N	N
CITY	varchar2(50)	Y	Facility's city	N	N
ADDRESS	varchar2(100)	Y	Facility's address	N	N
PARENT_SITE_CODE	varchar2(100)	N	Facility's parent site code. Matches its site code if facility is a parent facility.	N	N

5.1.3.22. HEALTH_ADVOCATE_FORM

Used by: HealthAdvocate

Column Name	Data Type	Nulla ble	Description	P K	F K
ID	varchar2(32)	Y	Represents the primary key of the table.	Y	N
STATUS	varchar2(255)	Y	The status of the form	N	N
PATIENT_ID	varchar2(255)	Y	Patient identifier	N	N

Column Name	Data Type	Nullable	Description	PK	FK
REQUEST_DATE	datetime	Y	The date that the form was saved	N	N
HEALTH_ADVOCATE_REQUEST_ID	varchar2(32)	N	reference representing the request in HEALTH_ADVOCATE_REQUEST	N	N
FORM_DATA	blob	Y	The legal form agreed to	N	N

5.1.3.23. HEALTH_ADVOCATE_REQUEST

Used by: HealthAdvocate

Column Name	Data Type	Nullable	Description	PK	FK
HEALTH_ADVOCATE_REQUEST_ID	varchar2(32)	N	Generated Primary Key	YES	N
HEALTH_ADVOCATE_EDIPI	varchar2(255)	N	The VA identifier of the person whom the patient wants to be their health advocate	N	N
PATIENT_EDIPI	varchar2(255)	N	The VA identifier of the patient making the request	N	N
URI_LINK_KEY	varchar2(255)	N	A key for the URL that the health advocate will activate to complete the process (not used yet)	N	N
EMAIL	varchar2(255)	N	The health advocate's email address	N	N
FIRST_NAME	varchar2(255)	N	The health advocate's first name	N	N
LAST_NAME	varchar2(255)	N	The health advocate's last name	N	N
ADDRESS_FIRST_LINE	varchar2(255)	N	The first line of the health advocate's address	N	N
ADDRESS_SECOND_LINE	varchar2(255)	N	The second line of the health advocate's address	N	N

Column Name	Data Type	Nullable	Description	PK	FK
CITY	varchar2(255)	N	The health advocate's city	N	N
STATE	varchar2(20)	N	The health advocate's state	N	N
ZIP_CODE	varchar2(10)	N	The health advocate's zip code	N	N
STATUS	varchar2(30)	Y	The current status of the request	N	N
REQUEST_DATE	date	N	The date the request was initiated	N	N
LAST_ACTION_DATE	date	Y	The date of the request's most recent status change	N	N
PATIENT_NAME	varchar2(255)	N	The name of the patient who made the request	N	N
DRIVERS_LICENSE_NUMBER	varchar2(255)	Y	The health advocate's driver's license number	N	N
DRIVERS_LICENSE_STATE	varchar2(255)	Y	The health advocate's driver's license state of issue	N	N

5.1.3.24. LAUNCHPAD_ITEM

Used by: Veteran Web Launchpad

Column Name	Data Type	Nullable	Description	PK	FK
NAME	varchar2(50)	N	Name of item on launchpad	YES	N
ITEM_MODE	varchar2(50)	N	Is item for veteran or provider?	YES	N
TYPE	varchar2(50)	N	Is item an app or a link	N	N
ITEM_POSITION	number(10,0)	Y	Position of item on launchpad	N	N
DESCRIPTION	varchar2(100)	Y	Description of item	N	N
URL	varchar2(2000)	Y	URL of item	N	N
IMAGE_URL	varchar2(2000)	Y	Image URL	N	N

5.1.3.25. MEDICATION_REFILL_REQUEST

Used by: RxRefill

Column Name	Data Type	Nullable	Description	PK	FK
ID	varchar(32)	N	Generated primary key	Y	N
PATIENT_ID	varchar(255)	N	Id associated with the patient	N	N
MEDICATION_ID	varchar(255)	N	Medication Id associated with the Rx being refilled	N	N
REQUEST_DATE	date	N	Date of the requested refill	N	N

5.1.3.26. METRIC

This is not used by any specific application. It stores general metric information used to evaluate the performance and usage of the system.

Column Name	Data Type	Nullable	Description	PK	FK
METRIC_ID	varchar2(40)	N	Generated primary key	YES	N
METRIC_NAME	varchar2(100)	Y	Method name being executed	N	N
TIMING	number(11,0)	Y	Execution Time	N	N
PARENT_EVENT_ID	varchar2(40)	Y	ID associated with main method that is being executed	N	N
START_DATE	date	Y	Execution Start Time	N	N
END_DATE	date	Y	Execution End Time	N	N
START_TICKS	number(20,0)	Y	Execution Start Ticks	N	N
END_TICKS	number(20,0)	Y	Execution End ticks	N	N
THREAD_ID	varchar2(100)	Y	System Thread Id	N	N
RESULT	varchar2(10)	Y	Execution result, whether it is SUCCESS or FAILURE	N	N
LOG_DATE	varchar2(30)	Y	Date when the metrics are added to database	N	N
LOG_THREAD	varchar2(1000)	Y	System Thread id when metrics are added to the database	N	N

5.1.3.27. METRIC_DATA

This is not used by any specific application. It stores general metric information used to evaluate the performance and usage of the system.

Column Name	Data Type	Nullable	Description	PK	FK
METRIC_ID	varchar2(40)	N	Generated primary key for metric Id	YES	Y
TAG	varchar2(100)	N	Tag to represent metric data	YES	N
VALUE	varchar2(1000)	Y	Value associated with tag	N	N

5.1.3.28. MOOD

Used by: Journal

Column Name	Data Type	Nullable	Description	PK	FK
ID	char(36)	N	Generated primary key	YES	N
PATIENT_ID	varchar2(255)	Y	Patient associated with the mood entry	N	N
MOOD	varchar2(20)	Y	The patient's mood rating from 0-10	N	N
ENTRY_DATE	date	Y	The date the mood rating was entered	N	N
NOTE	clob	Y	A note describing the patient's mood	N	N

5.1.3.29. MYGOALS

Used by: MyStory

Column Name	Data Type	Nullable	Description	PK	FK
MYGOALS_ID	varchar2(32)	N	Generated primary key	YES	N
PATIENT_ID	varchar2(255)	N	Patient associated with the my goals entry	N	N
INVENTORY_TYPE	varchar2(255)	N	The patient's health inventory type (feelings, current and desired state, reflections and priorities)	N	N
QUESTION_KEY	varchar2(255)	N	The patient's health inventory question key, value assigned to each question in Health Inventory	N	N
ANSWER	varchar2(4000)	Y	The patient's answer for the health inventory question	N	N
DATE_UPDATED	date	N	The date, the health Inventory was updated	N	N
ELAPSED_TIME	number(32,0)	Y	The duration, patient took to answer the health Inventory	N	N

Column Name	Data Type	Nullable	Description	PK	FK
			question		

5.1.3.30. NOTIFICATION

Used by: Notifications and Reminders

Column Name	Data Type	Nullable	Description	PK	FK
NOTIFICATION_ID	varchar2(32)	N	Generated primary key	YES	N
USER_ID	varchar2(255)	N	System ID of user	N	N
BODY	varchar2(4000)	Y	Notification message body	N	N
CREATED_DATE	date	N	Created date of message	N	N
ACTIVE	number(1,0)	N	Status of message	N	N
DELETED_DATE	date	Y	Deleted date of message	N	N

BODY = Yes because of the text that a user enters for the notification message. There are no constraints on the type of information it could contain and because of the context of how it is used, it can contain data that is any combination of PII/PHI/SPI/Sensitive Information

5.1.3.31. PAIN_GOALS

Used by: VAPainCoach

Column Name	Data Type	Nullable	Description	PK	FK
PAIN_GOAL_ID	varchar2(32)	N	Generated primary key	YES	N
DATE_STARTED	date	Y	The date the goal entry was created	N	N
NAME	varchar2(255)	N	The name of the goal entry	N	N
GOAL_STATUS	varchar2(255)	N	The current status of the goal	N	N
PERCENT_COMPLETE	varchar2(255)	Y	The current percentage complete for the goal	N	N
GOAL_TYPE	varchar2(255)	N	The type of goal	N	N
DETAILS	varchar2(255)	Y	Details describing the goal and how to achieve it	N	N
PATIENT_ID	varchar2(255)	Y	The ID of the user who created the goal	N	N
NEXT_ACTION_STEP	varchar2(4000)	Y	The description for the next	N	N

Column Name	Data Type	Nullable	Description	PK	FK
			step		
TARGET_DATE	date	Y	The target date for the goal	N	N
DATE_DELETED	date	Y	The date the goal was removed	N	N
DATE_COMPLETED	date	Y	The date that the goal was completed	N	N

5.1.3.32. PAIN_GOAL_ENTRIES

Used by: VAPainCoach

Column Name	Data Type	Nullable	Description	PK	FK
PAIN_GOAL_ENTRY_ID	varchar2(32)	N	Generated primary key	YES	N
ENTRY_DATE	date	Y	The date of the goal progress change	N	N
COMMENTS	varchar2(4000)	N	Comments about the goal progress change	N	N
PERCENT_COMPLETE	varchar2(255)	Y	The current percentage complete for the goal	N	N
PAIN_GOAL_ID	varchar2(32)	N	The ID of the corresponding pain goal in the PAIN_GOALS table	N	Yes
STATUS	varchar2(255)	Y	The status of the goal	N	N

5.1.3.33. PATIENT_METADATA

Used by: Appointment Request

Column Name	Data Type	Nullable	Description	PK	FK
PATIENT_ID	varchar2(255)	N	Patient ID	YES	N
LAST_APPT_REQ_ACCESS_DATE	date	Y	Date patient last accessed appointment request resource	N	N

5.1.3.34. PUBLIC_KEY

Used by: Notifications and Reminders

Column Name	Data Type	Nullable	Description	PK	FK
PUBLIC_KEY	varchar2(255)	N	A randomly generated key associated with the USER_ID	Y	N
USER_ID	varchar2(255)	N	System ID of user	N	N

5.1.3.35. RESPONSES

Column Name	Data Type	Nullable	Description	PK	FK
RESPONSE_ID	varchar2(32)	N	The response ID	Y	N
RESPONSE	vharchar2(255)	N	The response	N	N
QUESTION_ID	varchar2(255)	N	The question ID	N	N
ENTRY_ID	varchar2(32)	N	The entry ID	N	Y

5.1.3.36. USER_FEEDBACK

Used by: Launchpad

Column Name	Data Type	Nullable	Description	PK	FK
USER_FEEDBACK_ID	varchar2(32)	N	Generated primary key	YES	N
RATING	numeric	N	Numeric Rating of an application	N	N
APPT_PROCESSED_TIMELY	varchar2(32)	N	Whether the appointment was processed in a timely manner, pre-set messages	N	N
COMMENTS	clob	Y	Free text comments	N	N
CREATE_DATE	date	N	Date comment was submitted	N	N

5.1.3.37. USER_HISTORY

This is a general table that stores auditing information for many of the SED tables including MYGOALS, ASSESSMENT_RESULT, MOOD, EXERCISE, DIET, APPOINTMENT_REQUEST_MESSAGE, VITAL_ENTRY, DAILY_EVENT, APPOINTMENT_REQUEST, PAIN_GOALS, and MEDICATION_REFILL_REQUEST

Column Name	Data Type	Nullable	Description	PK	FK
ID	int(11)	N	Generated primary key	YES	N
USER_ID	varchar2(255)	Y	System ID of user	N	N

Column Name	Data Type	Nullable	Description	PK	FK
TABLE_NAME	varchar2(255)	Y	Name of table for the event	N	N
ROW_ID	char(36)	Y	The ID of the event	N	N
OPERATION	varchar2(255)	Y	Action performed (insert, update, or delete)	N	N
ACTION_DATE	date	Y	Time stamp for the action	N	N
DATA	clob	Y	XML representation of data being inserted/updated	N	N

5.1.3.38. USER_RIGHTOFACCESS

Used by: All applications. Users must accept a general Right of Access form in order to use any application

Column Name	Data Type	Nullable	Description	PK	FK
USER_ID	varchar2(255)	N	System ID of user	YES	N
ROA_STATE	number(1,0)	N	Right of Access State (Accepted or Rejected)	N	N
ROA_DATE	date	Y	If ROA was accepted, date of acceptance	N	N
ROA_FORM	blob	N	The ROA form	N	N

5.1.3.39. TOOL_TRACKING_RESULT

Used by: VAPTSDCoach

Column Name	Data Type	Nullable	Description	PK	FK
TOOL_TRACKING_RESULT_ID	varchar2(32)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	N	Patient Associated with the vital sign record	N	N
TOOL	varchar2(255)	Y	The tool that is tracked	N	N
FREQUENCY	varchar2(255)	Y	The frequency in which the tool is used	N	N
USEFULNESS	varchar2(255)	Y	The usefulness of the tool	N	N
SUCCESS_RATE	varchar2(255)	Y	The estimated rate of	N	N

Column Name	Data Type	Nullable	Description	PK	FK
			success in which the tool helped with the patient's symptoms.		
DATE_TRACKED	date	N	The date that the event occurred	N	N

5.1.3.40. VITAL_ENTRY

Used by: Journal

Column Name	Data Type	Nullable	Description	PK	FK
VITAL_ENTRY_ID	varchar2(32)	N	Generated Primary Key	YES	N
PATIENT_ID	varchar2(255)	N	Patient Associated with the vital sign record	N	N
SECTION	varchar2(255)	N	Optional Type associated with the record	N	N
QUALIFIER	varchar2(255)	Y	Optional Subtype associated with the record	N	N
NOTES	varchar2(4000)	Y	The entry notes	N	N
ENTRY_DATE	date	Y	The date that the event occurred	N	N

5.1.3.41. VITAL_OBSERVATION

Used by: Journal

Column Name	Data Type	Nullable	Description	PK	FK
VITAL_OBSERVATION_ID	varchar2(32)	N	Generated Primary Key	YES	N
TYPE	varchar2(255)	N	The type of the vital sign	N	N
VALUE	varchar2(255)	Y	The value of the vital sign	N	N
UNITS	varchar2(255)	Y	Units associated with the VALUE field.	N	N
VITAL_ENTRY_ID	date	N	FK to associate an observation with a vital entry	N	YES

5.2. Non-DBMS Files

Other than log files there are no non-DBMS files saved by the HealthAdapter.

6. Detailed Design

Please refer to Appendix A.

6.1. Hardware Detailed Design

As a cloud architecture, the cloud vendor provides the hardware infrastructure. The previous “hardware” section describes the list of virtual machines provided by the cloud Infrastructure as a Service (IaaS).

6.2. Software Detailed Design

6.2.1. Conceptual Design

6.2.1.1. User Interfaces

Refer to each mobile application. The HealthAdapter does not provide user interfaces. Those are the responsibility of the mobile applications.

6.2.1.2. Hardware Interfaces

There are no hardware interfaces in this system.

6.2.1.3. Software Interfaces

Table 4: Interfaces Internal to OIT shows the interfaces with VA systems between the HealthAdapter. Appendix A provides details of the data retrieved and used by the HA services.

6.2.1.4. Communications Interfaces

Figure 2: Network Interfaces in section 3.3.4 shows the communications paths between the servers and components.

6.2.1.5. Product Features

The mobile applications provide the end user features.

The features of the HealthAdapter architecture can be described by its design patterns.

Based on the development team's desire to utilize Agile project management and engineering practices (such as Scrum and Continuous Integration) as well as experiences with past "heavy" systems, several design principles emerged as they progressed.

6.2.1.5.1. Make the services easy to consume

Over the previous decade, WS* / SOAP web services became the standard for system integration. This was viewed as how to achieve Service Oriented Architecture (SOA). Although SOAP web services and the WS* specification can be very powerful, the complex standards are over-engineered for basic retrieve/save data based services.

In fact, many of the WS* specifications can be redundant when compared with the light-weight HTTP protocol *if the system is not utilizing more complex features of the WS* specs* (such as distributed transactions or SAML).

The complexity of consuming SOAP services is often minimized by leveraging frameworks. For example, in Java there are several standard frameworks including Metro, CXF, and Axis 2. For iOS, frameworks exist, but not at the same level of maturity of those in Java.

The developers have a goal to ensure that the HealthAdapter's mainline services remain as easy to consume as appropriate.

This goal has led to a resource-oriented approach, using REST frameworks to expose the services. It allows for a very simple HTTP consumption using light weight clients including iOS nsURL (simple HTTP calls rather than wrapping payload in SOAP envelopes), curl (easy to execute HTTP GET from command line), browser (easy to navigate the API through a browser), and AB (use Apache Benchmarking tool to perform performance testing through command line).

Also, from this principle, the modeling of the representation of the resources, an example being the XML or JSON returned from the services, is kept as simple as possible, and often as flat as possible. This is a different approach to other models, such as HL7v3, which favor very abstract models that allow for extensibility. The goal in HealthAdapter is to ensure that the representations are simple and easy to parse.

6.2.1.5.2. Keep the build times fast enough to encourage quick feedback

When the build time is fast, it is easy to assume a culture of making small changes, building, and testing. This cycle can be repeated many times in a given day. These quick feedback cycles lead to efficient development effort and it gets changes to be potentially-shippable quickly.

On many software development initiatives, as the project complexity grows, the amount of time that it takes to compile and deploy the software grows as well. This generally moves to a culture of making changes over longer period of time (hours, day, weeks), with long feedback cycles.

This has resulted in keeping the software simple and avoiding complex dependencies and bindings that normally would drive up build, deployment, and test times.

6.2.1.5.3. Keep test coverage high enough to give confidence in making quick changes

When a system has high automated test coverage over the code base, it allows for high quality software that can evolve. The development team can add new functionality quickly and are enabled for constant refactoring - leading to a "clean" code base.

6.2.1.5.4. Keep "data access" separate from the services

In creating the HealthAdapter, it became apparent, that there was a benefit to reuse the same set of services, but to retrieve data from different sources depending on the situation. For example, the initial implementation of the HealthAdapter pulled data from VistA, while the second implementation uses other sources such as CDW. In fact, there have been several proofs-of-concepts showing data pulled from NwHIN, DoD, and Google Health. The complexity of what data source to use is abstracted from the client, allowing application developers to focus on adding business value and not to be concerned with "plumbing".

6.2.1.5.5. HealthAdapter as a Resource-Oriented System

Much discussion exists today about loosely using REST (or RPC-over-HTTP) versus using a true Resource Oriented Architecture strategy. The HealthAdapter has been implemented favoring resource oriented principles to fully take advantage of benefits, such as simplicity, leveraging the transport's security, scalability, and discoverability.

6.2.1.5.6. Self-Discoverable URI's

Consider how a user navigates a web site. Once on a page within the system (either because of a search engine result or an external link), the user navigates to other pages within the system by utilizing hyperlinks on the page. This paradigm, central to the "world wide web" accomplishes two things. First, it limits the amount of external knowledge required in order to use the system - it allows for self-discovery. Second, it allows the state of the application to be maintained based on the URI's utilized, referred to as "[hypermedia as the engine of application state](#)", one of the [REST architectural principles](#).

In general, the HealthAdapter includes the following:

- *Self-Link*: This is the URI of the current resource. This is generally the actual resource that was requested, but could also be a canonical URI to the resource.
- *Related Links*: These are links to other related resources. The primary example of this is on a patient resource, which contains links to other patient clinical data

6.2.1.5.7. Resource Directory

To build on the concept of self-discoverable URI's, how does a consumer determine the URI to perform a patient search? Or how does the consumer discover the health check URI? The HealthAdapter comes with a resource directory that contains links to the root resources. The resource directory is located at this sub-context /rest/public/resource-directory (For example, <http://hosting-server/MobileHealthPlatformWeb/rest/public/resource-directory>).

6.2.1.5.8. Favor stateless calls

In order to allow for scalability, including adding additional servers without concern for session-affinity, the system favors stateless calls.

The primary places where state is maintained is for authentication. However, by maintaining the authentication state in a database that can be shared among multiple server instances, session-affinity concerns can be avoided.

6.2.1.5.9. Content Type Negotiation

There are several instances in the system where a single resource can have multiple representations. For example, a patient's appointments can be returned as XML, JSON, or in internet calendar format (iCal). Because there is a single resource leveraging HTTP as a protocol, each of these representations can be accessed by the same URL and the client specifies the format that it desires in HTTP headers ("Accept").

Where required, either because of testing convenience or because of limitations of certain clients (calendars that subscribe to a URL for its feed typically do not specify the media type), a URL

specific to that content/media type is exposed. For example, the patient appointment ical feed is exposed both as

/patient/{id}/appointments
and

/patient/{id}/appointments.ical
with the latter defaulting to ical as the content type.

6.2.1.5.10. Favor Meaningful URI's

The [HealthAdapter](#) favors use of non-opaque URI's, meaningful URI's that are human readable. Example of a [HealthAdapter](#) URI to access a patient's allergies:

[MobileHealthPlatformWeb/rest/patient/100845/allergies](#) (where 100845 is the patient's identifier)

An example of alternate approach using an opaque URI:

[MobileHealthPlatformWeb/rest/aefqq13](#)

from this example, the [HealthAdapter](#) URI scheme is human-readable. Because URI's are discovered, neither approach affects the applications utilizing the services. However, the non-opaque URI strategy makes it easier during development/testing/debugging.

Although human readable URI's give the impression that it is possible for the client to "self-construct" URI's, this is **not** recommended as it leaves the consumer vulnerable to upgrades (i.e. URI patterns change) or making incorrect assumptions about how to construct the URI. Always "discover" URI's.

6.2.1.5.11. Leverage HTTP protocol for "Methods"

In the [HealthAdapter](#), the system leverages the standard HTTP methods, and utilize in the way that they are intended.

GET - This is generally used to fetch a resource at the URL requested (HTTP GET is used by a browser to fetch a web page)

PUT - This is generally used to update a resource at the URL requested

DELETE - It is generally used to remove a resource at the URL specified

POST - This is generally run an operation, although its name comes from "posting" a new entity to a list

HEAD - This is the same as GET but without the body (gain metadata information on the resource)

Method	Safe	Idempotent
GET	yes	yes
HEAD	yes	yes
PUT	no	yes
DELETE	no	yes
POST	no	no

The proper use of the HTTP verbs should also follow guidelines on being safe and idempotent. Safe operations are those that do not change the state of the resource, while an idempotent operation is an operation that can be run once or multiple times with the same effect.

6.2.1.5.12. Leverage the HTTP Response Codes

Where possible, the HealthAdapter utilizes the standard HTTP error codes to convey causes of failure. The list below are overly-simplified explanations of the status codes as applies to the HealthAdapter.

200 (success): a 200 should only be returned when no error has occurred

401 (unauthorized): user is not authenticated, this generally triggers the authentication process

403 (forbidden): user is authenticated, but does not have permission to access the resource

404 (not found): the resource that was asked for does not exist

500 (server error): general server exception occurred

6.2.1.6. User Characteristics

Please refer to the documentation for the specific mobile applications for user characteristics.

6.2.1.7. Dependencies and Constraints

The system design is constrained by:

- All technologies must be compliant with the VA TRM. For usage of technologies not on the TRM, the VAMF team or the mobile application team must request that the technology be reviewed and added to the TRM.
- The VAMF must implement all applicable High NIST 800-53 controls and all other FISMA requirements of a High categorized system.
- The VAMF must have interfaces to VA enterprise services where available. Point to Point interfaces are discouraged but implemented when enterprise services do not exist yet.
- The VAMF architecture must migrate to the Mobile Application Reference Architecture (MARA) as defined by ASD when it becomes available. A transition plan must be coordinated to execute this migration.

6.2.2. Specific Requirements

6.2.2.1. Database Repository

Please refer to section 5 for a description of the database repositories within the VAMF.

6.2.2.2. System Features

The following table defines the system features of the VAMF based on the MHED Requirements Specification Document (RSD). The individual mobile application features are defined in the mobile application SDD Addendums.

Table 7: VAMF System Features

Feature Section	Feature	Design
Design Constraints	Data Confidentiality	Transport Layer Security (TLS) is used on all transmissions whether XML / JSON objects are contained in the transmission. This protects the data as passed both within the system boundaries and to external systems

Feature Section	Feature	Design
Disaster Recovery	Backup and Recovery of cloud	Terremark and the VAMF sustainment contractor have DR designs that are documented in the VAMF Disaster Recovery Plan
Functional Specifications	Veteran credentials	The requirement is for veterans to log in with DSLogon. The VAMF includes an oAuth service in the veteran string that reaches out to the DSLogon DMDC system through the VAAFI proxy environment in IAM.
Functional Specifications – GUI	Browser support	The VAMF will provide utility services to mobile applications to detect the browser manufacturer and version
Performance Specifications	Concurrent users of 5000	The system has been scaled with multiple WebLogic instances to load balance across servers to meet the specification of 5000 concurrent users. The architecture is scalable to add servers of multiple technologies should the capacity be required.
Reliability Specifications	99.99% Availability	Terremark and the VAMF sustainment contractor have DR designs that are documented in the VAMF Disaster Recovery Plan. Redundant systems exist in the Miami FL system that can be restored and activated to meet this requirement
Integration Specification	Staff credentials	The VA facing string will authenticate healthcare users through VistA credentials. VistA will then authorize the access of the users based on the VistA privileges provided to that user.
Security Specification	Log Data	Log data does not contain PHI/PII and is persisted to the file system behind firewalls and with limited personnel retaining access

6.2.2.3. Design Element Tables

Appendix A contains entry points into the VAMF including the data returned. These entry points are REST services into VAMF. They in turn query VAMF specific data or obtain data from VA enterprise services.

Individual applications that change VistA routines will document those routines and provide them to the VistA Document Library (VDL). The VAMF itself does not include changes to the VistA routines. Appendix A describes the data elements within the VAMF as well as Section 5 for the storage within the VAMF.

HL7 Links will be required for specific mobile applications and those mobile applications will register their HL7 messages with the appropriate HL7 points of contact or administrators.

6.3. Communications Detailed Design

The Terremark VA Dedicated Cloud provides the communication infrastructure. See Section 3.3.4 for the resulting specific to the VAMF.

7. External Interface Design

Please refer to Appendix A.

7.1. Interface Architecture

Figure 4 shows the Integration architecture within the VAMF. The current architecture uses MDWS for integration with VistA data sources through RPC calls. Originally there were plans to use systems such as AVIVA and VistALink, but these plans have been changed. ASD is defining a MARA that includes the use of the iEHR SOA Suite and the VistA Service Assembler (VSA). Transition of the architecture to use VSA will be made once that architecture is defined.

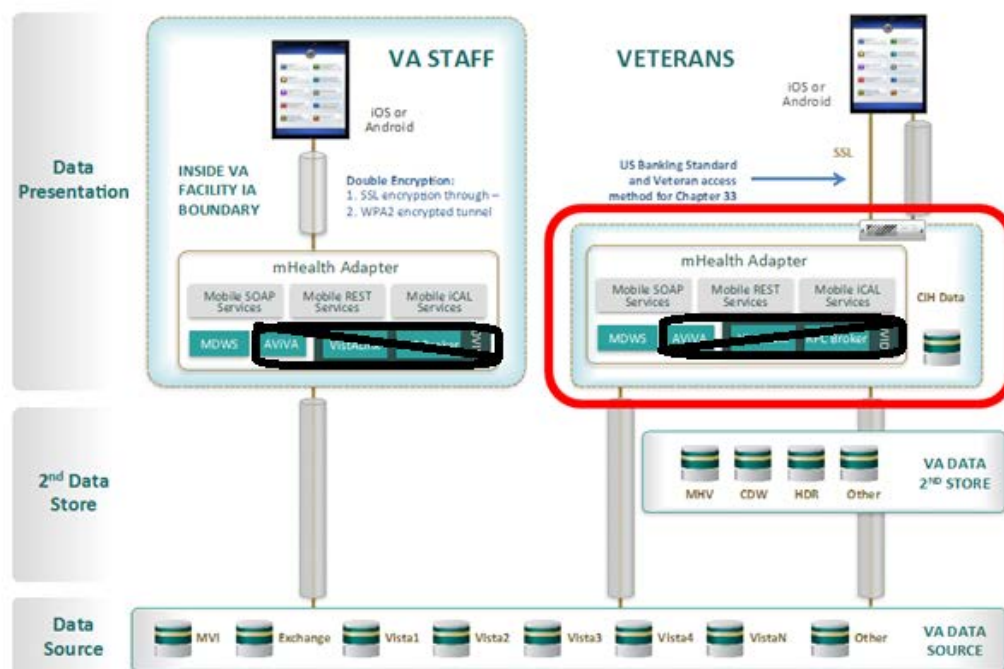


Figure 4: Integration Architecture

7.2. Interface Detailed Design

In the context of the **VA Health Adapter**, authentication is the process by which the adapter verifies that the identity presented to it is valid. This validation is done via its authentication service which uses external VA user repositories. Applications on the client device will connect to the **VA Health Adapter** Authentication Service through a REST call.

7.2.1. Patient Identifier

Patient Identifier has two parts:

- Assigning authority

Patient id given by that authority.

For Clinic-in-Hand, the assigning authority will likely be "ICN" and the patient id will be the patient's ICN. When the data service needs to retrieve data from a specific system.

7.2.2. Transport Layer Security (TLS)

While the HealthAdapter can support SSL, Transport Layer Security (TLS) is preferred protocol used by the HealthAdapter to provide privacy and data integrity of information to other devices/applications. During the handshake between the HealthAdapter and clients, they will negotiate (sending the version of SSL/TLS and cipher suites supported) and use the most secure communication that both support. The VAMF supports TLS 1.0, 1.1, and 1.2.

The primary goal of the TLS protocol is to provide privacy and data integrity between two communicating applications. The protocol is composed of two layers: the TLS Record Protocol and the TLS Handshake Protocol. At the lowest level, layered on top of some reliable transport protocol (e.g., TCP [TCP]), is the TLS Record Protocol. The TLS Record Protocol provides connection security that has two basic properties:

- The connection is private. Symmetric cryptography is used for data encryption (e.g., AES [AES], RC4 [SCH], etc.). The keys for this symmetric encryption are generated uniquely for each connection and are based on a secret negotiated by another protocol (such as the TLS Handshake Protocol). The Record Protocol can also be used without encryption.
- The connection is reliable. Message transport includes a message integrity check using a keyed MAC. Secure hash functions (e.g., SHA-1, etc.) are used for MAC computations. The Record Protocol can operate without a MAC, but is generally only used in this mode while another protocol is using the Record Protocol as a transport for negotiating security parameters.

The TLS Record Protocol is used for encapsulation of various higher- level protocols. One such encapsulated protocol, the TLS Handshake Protocol, allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before the application protocol transmits or receives its first byte of data. The TLS Handshake Protocol provides connection security that has three basic properties:

- The peer's identity can be authenticated using asymmetric, or public key, cryptography (e.g., RSA [RSA], DSA [DSS], etc.). This authentication can be made optional, but is generally required for at least one of the peers.
- The negotiation of a shared secret is secure: the negotiated secret is unavailable to eavesdroppers, and for any authenticated connection the secret cannot be obtained, even by an attacker who can place himself in the middle of the connection.
- The negotiation is reliable: no attacker can modify the negotiation communication without being detected by the parties to the communication.

7.2.3. XML-JSON

The **VA Health Adapter** is able to accept XML or JSON (JavaScript Object Notation) as a method of data transfer to client devices.

Because most JSON-formatted text is also syntactically legal JavaScript code, an easy way for a JavaScript program to parse JSON-formatted data is to use the built-in JavaScript `eval()` function, which was designed to evaluate JavaScript expressions. Rather than using a JSON-specific parser, the JavaScript interpreter itself is used to execute the JSON data to produce native JavaScript objects. However, there are some Unicode characters that are valid in JSON strings but invalid in JavaScript, so additional escaping would be needed before using a JavaScript interpreter. [15]

Unless precautions are taken to validate the data first, the `eval()` technique is subject to security vulnerabilities if the data and the entire JavaScript environment is not within the control of a single trusted source. If the data is itself not trusted, for example, it may be subject to malicious JavaScript code injection attacks. Also, such breaches of trust may create vulnerabilities for data theft, authentication forgery, and other potential misuse of data and resources. Regular expressions can be used to validate the data prior to invoking `eval()`. For example, the RFC that defines JSON ([RFC 4627](#)) suggests using the following code to validate JSON before eval'ing it (the variable 'text' is the input JSON): [16]

Reference

JavaScript Object Notation ([JSON](#))

JSON is a text format for the serialization of structured data. It is derived from the object literals of JavaScript, as defined in the ECMAScript Programming Language Standard, Third Edition [ECMA](#).

JSON can represent four primitive types (strings, numbers, booleans, and null) and two structured types (objects and arrays).

A string is a sequence of zero or more Unicode characters [UNICODE](#).

An object is an unordered collection of zero or more name/value pairs, where a name is a string and a value is a string, number, boolean, null, object, or array.

An array is an ordered sequence of zero or more values.

The terms "object" and "array" come from the conventions of JavaScript.

JSON's design goals were for it to be minimal, portable, textual, and a subset of JavaScript.

Extensible Markup Language ([XML](#))

Extensible Markup Language, abbreviated XML, describes a class of data objects called [XML documents](#) and partially describes the behavior of computer programs which process them. XML is an application profile or restricted form of SGML, the Standard Generalized Markup Language [[ISO 8879](#)]<http://www.w3.org/TR/REC-xml/#ISO8879>. By construction, XML documents are conforming SGML documents.

XML documents are made up of storage units called [entities](#), which contain either parsed or unparsed data. Parsed data is made up of [characters](#), some of which form [character data](#), and some of which form [markup](#). Markup encodes a description of the document's storage layout and logical structure. XML provides a mechanism to impose constraints on the storage layout and logical structure.

7.2.3.1. Supported JSON Data Types

- * Number
- * String
- * Boolean
- * Array
- * Object
- * null

JSON Example

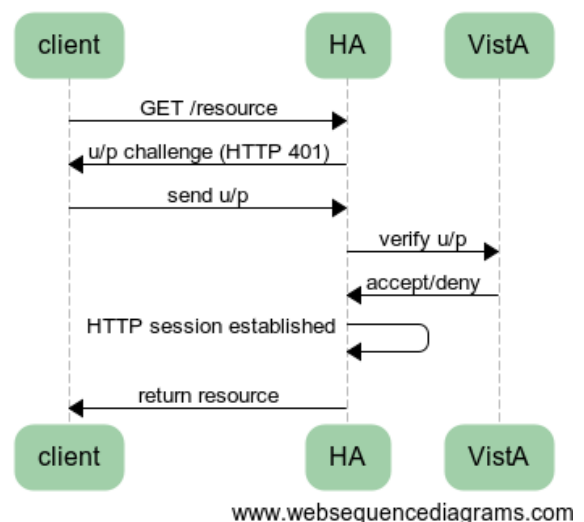
```
<test>
<another tag="Insert JSON Code example"/>
</test>
```

7.2.4. Staff Authentication

When the [HealthAdapter](#) is running in "staff" mode (Staff Environment), the [HealthAdapter](#) protects its services by having users authenticate using their [VistA](#) credentials. The authentication process follows OAuth 2.0 specification.

7.2.4.1. HTTP Basic Security

For VA Innovation Initiative (VAi2) 2010, the [HealthAdapter](#) (then iHealthAdapter) secured services by utilizing [HTTP Basic Security](#). When any client requests data from a secured resource, the client is challenged for a username/password by returning an HTTP status code 401 (unauthorized). The client responds with the username/password. The server verifies the username/password against the authentication store (the authentication adapter allows for substitution of the authentication store; for production this is done against [VistA](#)). If successful, the HTTP session is established and the resource is returned. If the authentication fails, the server prompts the client against (HTTP 401). The basic workflow is diagrammed to the right:



The implementation of this sequence relies on the following actions:

- Upon launch of the client application (in this case, the Patient Viewer application), the user enters the access/verify code into a form. These credentials are used by the application with the HTTP library ([ASI](#)). The application performs a GET on a resource (/user-session), which triggers the sequence above.

- iHealthAdapter uses [Spring Security](#) to manage the server side of the HTTP Basic Security. This ensures that all secure services trigger this workflow. Spring Security allows for a plug-in model. iHealthAdapter has a custom plug-in that works with Spring Security. When a user needs to be authenticated, the plug-in authenticates the user against [VistA](#). If successful, it sets the user info in the session. If it fails, it raises an exception.

For the VAI2 2010 [HealthAdapter](#) (then iHealthAdapter) this capability was sufficient.

7.2.4.2. Need for More Robust Authentication and Authorization

For Clinic-in-Hand, there was a need for sign-on capabilities beyond those provided by HTTP Basic Security. The typical Clinic-in-Hand logon application launch workflow and its related issue are as follows:

- A user logs on to one app on a mobile device.
- A user launches a second app on the mobile device.
- The user's "context" (who he is and the fact that he has been authenticated) is known to the second app, but the username/password are not known.

There are other Clinic-in-Hand logon application launch workflows:

1. A user first launches a "Launch Pad" application, logs on, and then launches an application.
2. A user first launches a "Launch Pad" application, logs on, goes back to "home", and then launches an application directly.
3. A user launches an application (through either one of the scenarios above), then launches a second application from the first.

One iOS approach could be to pass the username/password from one application to another. Although iOS does not support a mechanism to access another application's data, it does support the concept of launching another application and passing information. This is done by invoking a URI, with an application registering its scheme with the device. For example, an application can register that it handles the schema "app1" - the device would defer the URI request of this scheme to this application. This would allow the launch pad to launch the app with the following URI: "app1://username=user&password=pass".

There are some issues with this approach:

- It gives another application full information. (No granular permissions, gives access to username/password directly)
- It does not address workflow #2 (where user accesses child application directly).

Other secondary considerations include the following:

- Create a framework that supports other clients (not just iOS, but includes other HTTP clients such as Android, web applications).
- Create a framework that allows to expand to sharing context in the spirit of CCOW (user, patient, current encounter).
- Create a framework that can limit the data allowed to be accessed by an application (user can choose to allow the allergy checker application to access hit medications and allergies, but not access his mental health records).

7.2.4.3. OAuth

The issues explained in the preceding section are examples of flaws in distributed authentication and authorization implementations in common practice today. In the introduction sections of the [OAuth spec](#), it further defines these issues as follows:

- Third-party applications are required to store the resource-owner's credentials for future use, typically a password in clear-text.
- Servers are required to support password authentication, despite the security weaknesses created by passwords.
- Third-party applications gain overly broad access to the resource-owner's protected resources, leaving resource owners without any ability to restrict duration or access to a limited subset of resources.
- Resource owners cannot revoke access to an individual third-party without revoking access to all third-parties, and must do so by changing their password

These limitations make it difficult to move forward with a distributed authentication model that allows for one system to utilize another system's resources while ensuring compliance with the resource owner's desire of how to expose data.

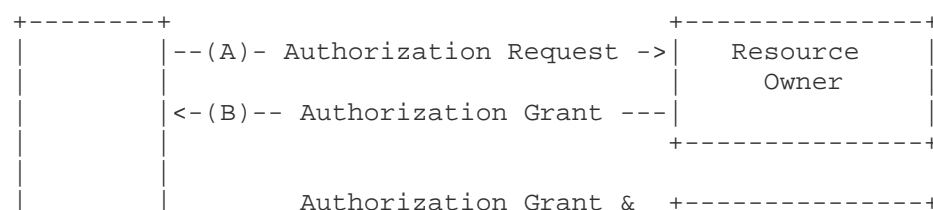
Facebook is often used as an example of need here: Many third-party applications seek to use Facebook as an authentication source; Facebook seeks to allow other applications to use their data; Facebook users want to limit what data can be accessed by a specific third-party application from their profile.

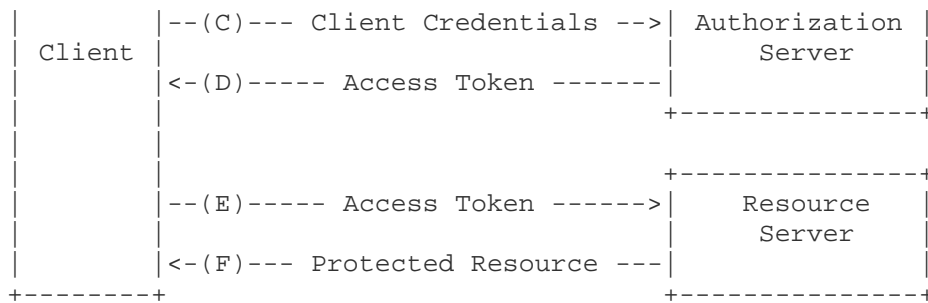
These limitations have driven the formation of the [OAuth](#) protocol (sponsored by Yahoo!, Facebook, and Microsoft). The emerging [OAuth 2.0](#) standard addresses these issues and allows for both distributed authentication and fine-grain access control of resources.

[OAuth](#) defines four main roles:

- Resource Owner: this is the entity that can grant access to a protected resource. (Example: Facebook users who grant/deny access to a portion of their profile to a specific application; patients who grant/deny access to a portion of their healthcare record to a specific application)
- Resource Server: the server that exposes services/resources. (Example: Facebook's API; [HealthAdapter](#))
- Client: application that desires to utilize the resources on behalf of a resource owner. (Example: Third-party Facebook application; Clinic-in-Hand application)
- Authorization Server: the server that brokers the grant between the client/resource server/resource owners.

A very generic depiction of the interaction is defined here:





This can be stated as follows:

- Client needs to get an authorization grant
- Client uses the authorization grant to obtain an access token
- Client uses access token to access the protected resource.

Authorization Grant

Consider the authorization grant as the credential used to get an access token. The process to get the authorization grant follows one of four "styles" (defined as the authorization grant type):

Grant Type	Description	Benefit	Disadvantage
Authorization Code	The client uses an intermediary between the client and resource-owner (user)	Resource-owner credentials are NOT shared with the client (secure) Access token is never shared with the intermediary	Requires complex interaction with the user agent
Implicit	The client is granted an access token without intermediary Simplified for client-side browser applications	More responsive, "faster"	Security implications of exposing the token with the resource owner or other applications
Resource Owner Password Credentials	Resource owner enters credentials into application	Simplified workflow	Requires high trust of client application. (Our application doesn't lend itself well to sharing of context/single sign-on without fully sharing credentials.)
Client Credentials	Rather than resource owner entering credentials, application sends its "system" credentials		Requires that the application is fully trusted, regardless of the user

Access Token

This is the credential used to access the protected resource. It is obtained by using an authorization grant. An access token is good for a specific scope (access to a patient record or access to patient allergies) and duration (expires after certain period).

The concept of an access token separates the process of authentication of the user's credentials from the access of the resource, limiting the exposure of the user's credentials. This abstraction also allow the resource server to not have to care about different ways to authenticate the user (username/password, Common Access Card [CAC], retina scan, etc.).

Choosing a Grant Type

In deciding on the authorization grant type, the specification calls out two types of client types:

- "Confidential" clients have the ability to securely store their client credentials used to authenticate the client application (not the resource owner/user). This is typically a server-side application, not a client-side application.
- "Public" clients are those that do not have means to secure their client credentials.

The specification considers a native application (such as an iOS application) to be "public". I read this to be true because a mechanism to secure the applications username/password would be by securing the source code. Decompiling or access source code is not a means to high degree of confidence of security.

The authorization code grant type is optimized for confidential clients; the implicit grant type is optimized for public clients. However, the specification further explains some considerations for native applications. Summarized:

- Native applications can use either authorization code grant or implicit grant type, they do not simply rely on the client authentication from a security perspective.
- Choose your user-agent carefully. An external user-agent is good for single sign-on; it is secure because users never share their credentials with application. An embedded user-agent is good for style and user experience.

Based on this input, we are implementing our mobile applications as a "public app" using "authorization code" as the grant type.

Authorization Code

First, the client recognizes that it needs access to a protected resource. This either can be built-in knowledge or a redirect based on a resource challenge.

The client redirects the resource owner to a user-agent (such as a browser), including the URI of the authorization server and a redirect URI to be used to return control to the client application upon successful grant. Other data passed includes application identifier and information on the resource desired to be accessed.

Note on the redirect URI: This can either be passed or declared in an out-of-spec registration process (i.e., app1 redirect url = "app1://" or "http://app1.com").

The user-agent prompts the resource owner to authenticate, typically via an HTML page from the authorization server rendered in the browser/user-agent. The user-agent submits the credentials to the authorization server and, if successful, returns an authorization code. This is

done via appending the authorization code to the redirect URL for the client and returning an HTTP redirect to the user-agent.

The client then invokes the authorization server, providing its authorization code to obtain an access token and client-authentication. (In the case of the native application, the client is not able to prove its identity.)

The token can then be used on subsequent steps.

Other

The standard is defined here: <http://tools.ietf.org/pdf/draft-ietf-oauth-v2-12.pdf>

A simplified explanation as used by Facebook is defined

here: <https://developers.facebook.com/docs/authentication/>

7.2.4.4. Single Sign-on for HealthAdapter Applications

The applications being built as part of the Clinic-in-Hand effort will implement single sign-on utilizing OAuth using the "Authorization Code" grant type workflow.

The server side components will leverage the Spring Security framework (Spring Security implements HTTP Basic Authentication as well as OAuth). The client-side components will likely be bundled into a reusable library to be shared across Clinic-in-Hand applications and potentially reused by other vendors building mobile applications.

This explanation will be geared towards iPad application, but the workflow is intended to be the same/similar on other client platforms.

Prior to the application being deployed, the application is registered with the HealthAdapter, providing an application identifier, an application "weak-password" (come up with better name here), and an application logon redirect URI.

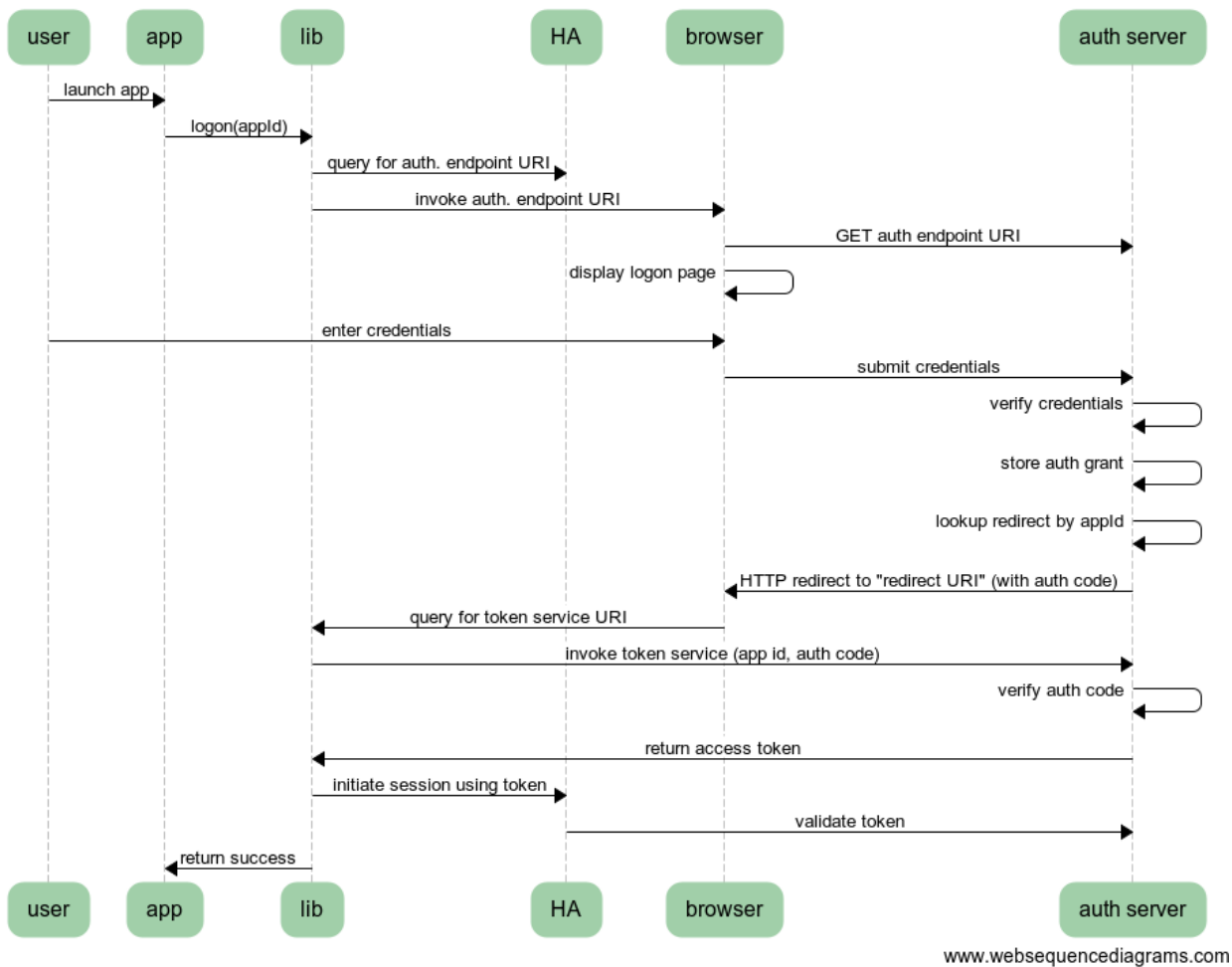
The workflow starts by the user launching an application, either from the "Launch Pad" or by launching the native application directly.

The application uses the HealthAdapter Library to initiate a logon. The library queries the HealthAdapter (HA for diagram below) to obtain the resource URI associated with authorization endpoint (i.e., [REDACTED]/authorization). The client library then launches the URI (with parameters including the application identifier and redirect URI), triggering the device to launch the browser to this URI/URL. This presents the user with a logon screen. The user then enters his credentials, which are verified against the patient authentication store (i.e., DS Logon). If successful, the server returns an HTTP redirect to the redirect URI provided (or registered for this application ID) plus the authorization code. This triggers the browser to relaunch the client, including the authorization code.

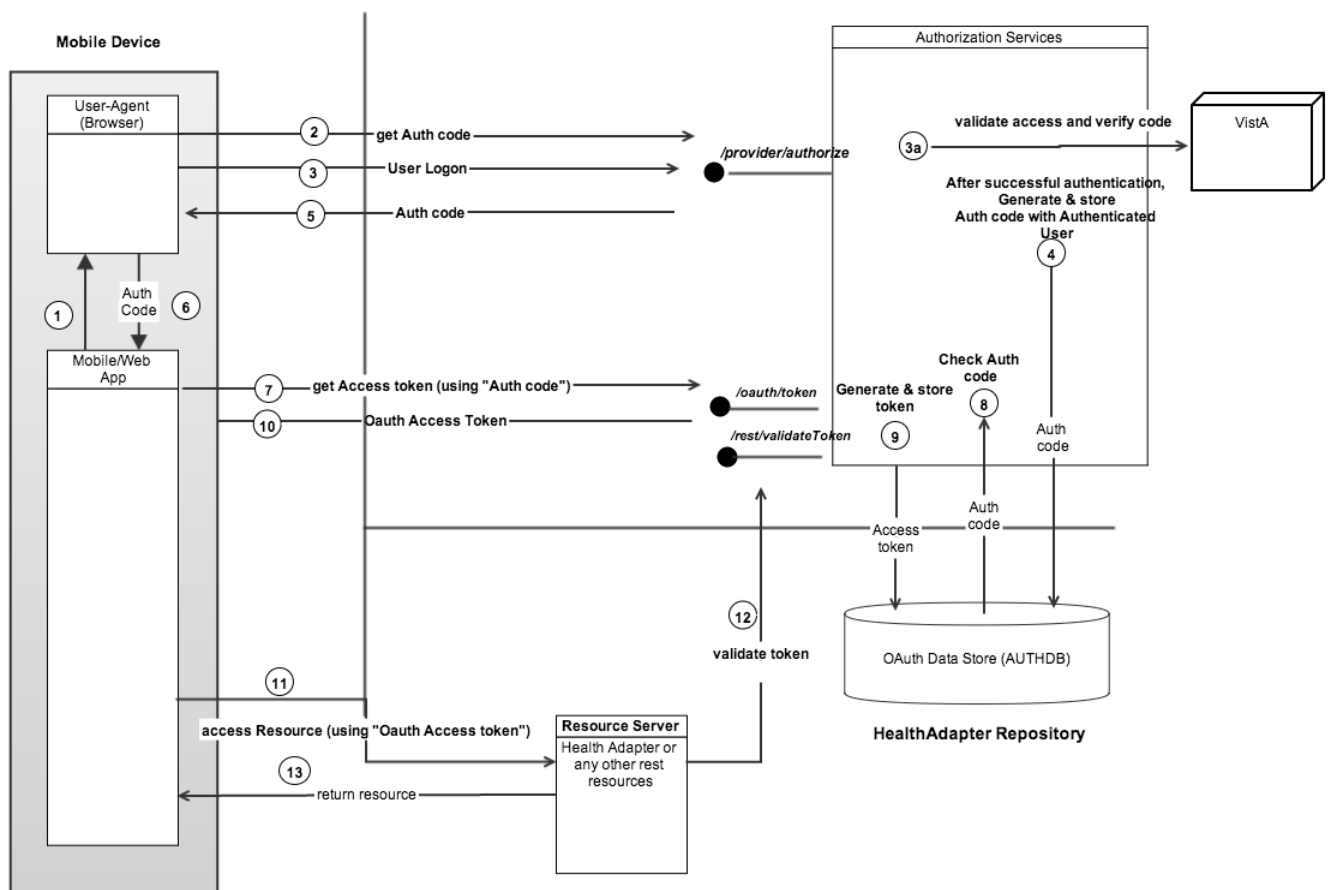
(Option: Having the client provide the redirect URI is convenient for development as it does not require registration, but it should be disabled during production mode to ensure that you are not trying to impersonate another application.)

Once relaunched, the client library looks up the authorization server's token endpoint and then invokes this, including the authorization code and client-credentials. (The client credentials are not to be considered as what makes this secure). The result is the access token.

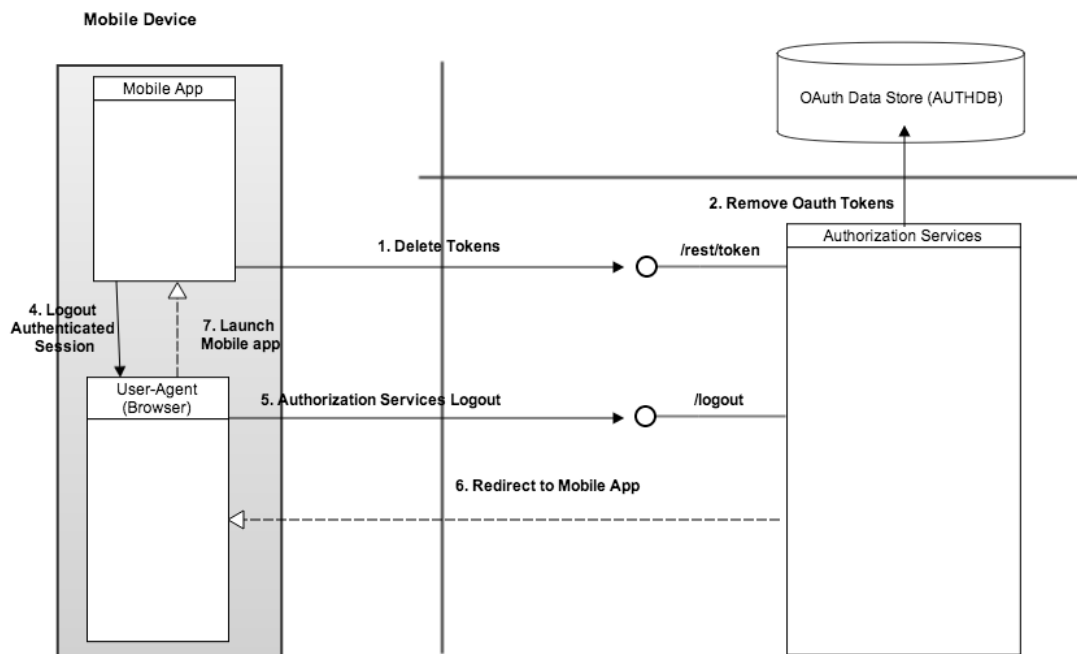
The access token is then used to access the protected resource.



7.2.4.4.1. Staff iOS App Logon Authentication and OAuth Authorization Workflow



7.2.4.5. Staff Logout



7.2.4.6. Retrieving Data from VistA

During staff login, MDWS session id is stored as part of the user object along with VistA security keys. MDWS session id will be used in subsequent VistA data retrieval calls through MDWS. This mechanism uses the user connection instead of System account to connect to VistA. This helps with security as VistA options and keys will be applied per user and also with performance as there is no need to connect to VistA every time when data is retrieved.

7.2.5. Veteran Authentication

When running in "Veteran" mode (Veteran Environment), the HealthAdapter protects its services by having users authenticate using their [DS Logon](#) credentials. Authenticating and identifying the user is accomplished using [DS Logon](#) as an authentication provider and by using IAM's SSOe service offering to establish the session between the user and the [HealthAdapter](#).

7.2.5.1. IAM SSOe

IAM is the VA organization chartered for "Identity and Access Management". This group is responsible for identifying Veterans, tracking their "preferences", and protecting access to Veteran resources.

In "Veteran" mode, the [HealthAdapter](#) utilizes IAM's SSOe services. Although this single sign-on service has been designed for Veteran-facing web applications, this service will also be integrated for use with the [HealthAdapter's](#) mobile applications.

IAM SSOe supports multiple authentication providers, but for "Veteran" mode will use [DS Logon](#) as the authentication provider/user credentials. If in the future other authentication provider support is required, the IAM emerging offering for Enterprise Authentication Gateway can be utilized to abstract the authentication provider.

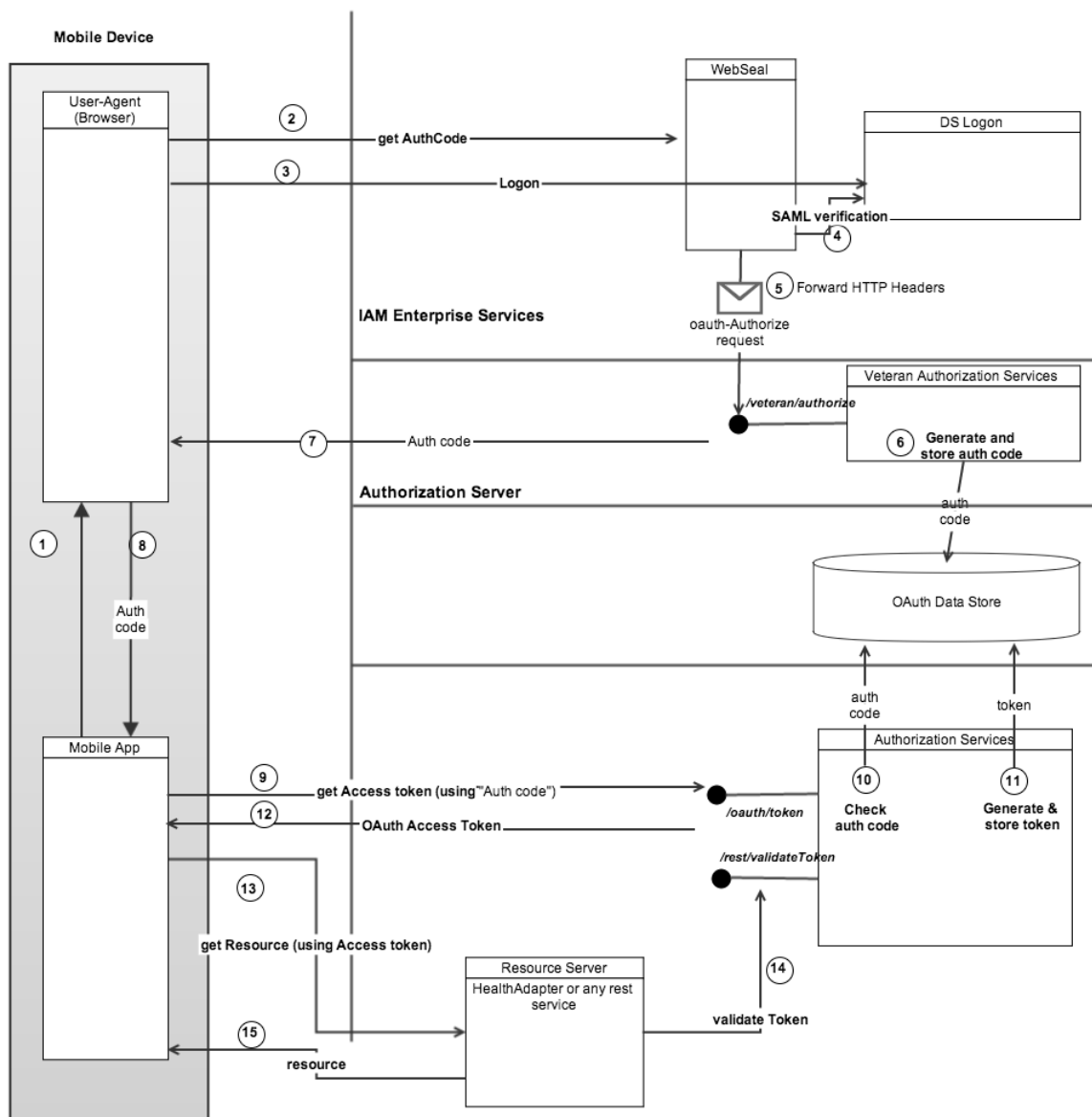
One of the challenges in this integration is that IAM SSOe is designed for web applications. This allows for an assumption that to provide single sign-on, all user traffic comes from a single client--a single browser on the user's device. All traffic flows from the user's browser to the IAM [WebSeal](#), acting as a reverse proxy, to the VA web applications. When there are multiple client applications that need to share in this authentication content, as is the case with using native mobile applications, this approach does not work "out of the box".

A second challenge is that [DS Logon](#), by policy, does not allow for credentials to be proxied through another system. That is, it would not be appropriate for users to enter their credentials into the mobile application directly, nor for the credentials to be sent to the HealthAdapter and then forwarded to [DS Logon](#).

The approach laid out in this document augments the IAM SSOe offerings while ensuring that a mobile app and the [HealthAdapter](#) are never given direct access to the credentials.

7.2.5.2. Workflow

At a high level, the integration with SSOe, mobile applications, and the HealthAdapter work as shown in the following diagram:



Once the app has been launched, the user then clicks "logon". This causes the app to perform an "openURL" on the "OAuth-authorize" resource, which launches the mobile browser (*not* an embedded browser) acting in the OAuth role as the user-agent [#1 in diagram]. The browser attempts to perform an HTTP GET [#2 in diagram] to the Authorization Server via WebSEAL.

Because the browser is not authenticated, WebSEAL returns a HTTP 302, redirecting the browser to the DS Logon Authentication Page [#3 in diagram].

The user enters credentials into the DS Logon Authentication Page and submits. Once successful, WebSeal forwards or redirects [#5 in diagram] the OAuth-authorize request to the Authorization Server (this includes information as to the identity of the user such as the user's EDIPI). The HealthAdapter's OAuth-Authorization Endpoint (on the Authorization Server) trusts the call from WebSEAL (because of the user of mutual TLS authentication).

The Authorization Server asks the user to approve the app to access their clinical resources [#5 in diagram] and once the user gives approval, the Auth server generates and stores auth-code [#6 in diagram] and redirects the browser to the HealthAdapter Server [#7 in diagram]. What is returned is a redirect (launchpad:///authcode=XXX) and auth code [#8 in diagram]. This causes the device to launch the mobile app which then performs an HTTP POST against the token resource [#9 in diagram], passing in the authcode. After the HealthAdapter validates the auth code against the OAuth Data store [#10 in diagram], the HealthAdapter then generates and returns a token to the app [#11,12 in diagram]. The HealthAdapter also stores the information related to the token (user, expiration) in memory.

As the mobile app (like LaunchPad) accesses protected resources on the HealthAdapter (such as fetch of the veteran's allergies), the token is passed as an HTTP header (HTTP Authorization Header) [#13]. The HealthAdapter invokes a service to validate the token. Since the token must be validated for all calls, it is likely that token validation would be cached to prevent network service invocations. The HealthAdapter requires the ability to translate the token to the corresponding user as the HealthAdapter acts a Policy Enforcement Point, ensuring that the resource being accessed is allowed by this user.

Note, changes from the original proposal

- the user-agent is the mobile browser, not an embedded browser within the app
- in #2 the browser attempts to access a resource via WebSEAL, and that because the browser is not authenticated, the browser is redirected to DS Logon Authentication page
- (Logoff: mobile app invokes service on OAuth provider, which invalidates the token. Ideally, there is an HTTP call that can be made which will logoff the browser-WebSEAL connection AND redirect back to the mobile app).

When a second application needs to logon, it will attempt to access the "Authorization Endpoint" (via WebSEAL). Because there is an existing session between the mobile device and WebSEAL, the browser is able to access the authorization endpoint without prompting the user to authenticate a second time.

7.2.5.3. Authentication Level

During the authentication process, a check is made to ensure that the user has the [appropriate authentication level](#). The different levels are described below.

DS Logon Account Type	Level	Allow HealthAdapter Access
Basic Account	1	No
Premium Account	2	Yes

7.2.5.4. Authorization for HealthAdapter to Access Patient Data on Patient's Behalf

In order for the [HealthAdapter](#) to access patient data, permission must be granted to the mobile application used by the Veteran. The approach will be to surface an authorization form to the user and to ask the user to accept the terms of agreement that will include language for the

system to access the Veteran's data and for secure messaging. This form is referred to as the "Right of Access" form. Ideally, the storage of the form can be in a repository that allows the form to be accessed by the VA enterprise. That is, we do not want to create a permanent silo of information. It is ideal that the Right of Access agreement be structured such that the acceptance covers all applications, not just a specific application or just the Clinic-in-Hand applications. This should lower the impediment for future initiatives to gain access to Veteran data.

The initial implementation will be to store this user preference in the [HealthAdapter Repository](#). At a future time, this will be migrated to an IAM service offering.

7.2.5.5. Patient Identifiers

At the conclusion of the authentication process, users will be known by their identifier associated with their [DS Logon](#) account, which is their DoD identifier ([EDIPi](#)).

In order to access patient data, it will be necessary to take a given patient identifier and discover the patient identifier used in various systems/contexts. For example, to access the [CDW](#), the patient's [ICN](#) is used, while when accessing [VistA](#), the patient's local IEN is used.

The direction from IAM is to use the authoritative source's ([MVI](#)) native identifier. The [MVI](#) services are WS*-style web services, offering services to "translate" a patient identifier (such as the DoD [EDIPi](#)) to another context using the [getCorrId\(\)](#) service.

Note from [MVI](#) on [ICN](#): Integrated Control Number ([ICN](#)) is an enterprise identifier assigned by the [MVI](#) that uniquely identifies a person. It is composed of 17 digits (10-digit sequence + 'V' delimiter + 6-digit checksum). In the near future (with non-Patient), [MVI](#) shall be updating this definition to include the Encryption key - additional 6 digits.

The call to [MVI](#) is expected to be "expensive". Making this call each time the [HealthAdapter](#) is to make a call to retrieve data would have a negative performance impact on the user's application. Further, the call to [MVI](#) is to be done as part of the user's SSO session, which only exists during authentication. Therefore, the call to [MVI](#) is to be done during the authentication process, and the result will be stored in a cache repository associated with the user session token. Once the session expires (either because of explicit logoff or because of time), the cache is deleted.

7.2.5.6. Logout

When user logs out of Mobile app, Health Adapter will clean up the OAuth tokens, http session and it will logout from SSOE session by redirecting the logout request to the SSOE/VAAFI logout url. After successful logout, user request will be redirected to the Launchpad from SSOE/VAAFI.

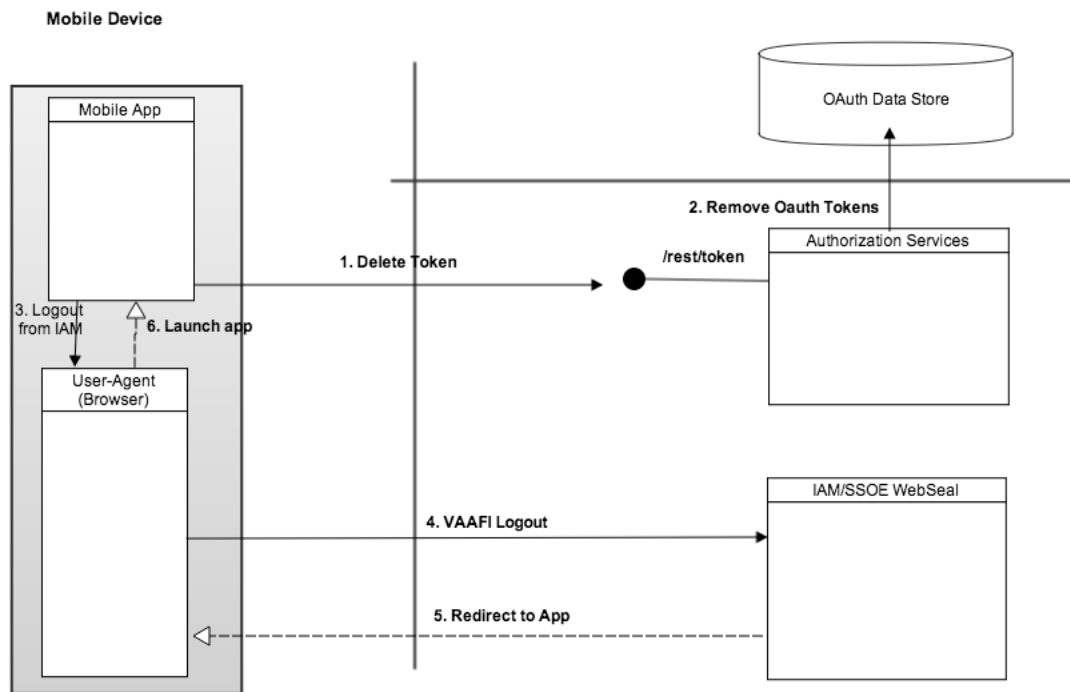


Figure 5: Logout Flow

8. Human-Machine Interface

The VAMF does not include user interfaces within it. The individual mobile apps provide the presentation layer. Refer to the SDD Addendums for each application to see the technologies and workflows of the individual applications.

8.1. Interface Design Rules

Please refer to the mobile application SDD Addendums.

8.2. Inputs

Please refer to the mobile application SDD Addendums.

8.3. Outputs

Please refer to the mobile application SDD Addendums.

8.4. Navigation Hierarchy

Please refer to the mobile application SDD Addendums.

9. System Integrity Controls

Please refer to Appendix A.

10. Approval Signatures

The signature below is an acknowledgement that the signatory understands the purpose and content of this document.

Signed: _____

Integrated Project Team Chair

Date

Signed: _____

Business Sponsor

Date

Signed: _____

IT Program Manager

Date

Signed: _____

Project Manager

Date

Signed: _____

Enterprise Architecture

Date

Signed: _____

Service Delivery and Engineering

Date

11. Appendix A – Mobile Adapter SDD

11.1. Sources of Data

The following decision table shows which external system will be used when fetching a certain data type. For example, if the demographic data is going to a Veteran, then data will be retrieved from [CDW](#).

Data Type	Veteran mode (Longitudinal)	Staff mode (Operational)	Staff mode (Longitudinal)
Patient Correlation	MVI	NA	MVI
Demographics	ADR / MVI	Local VistA (via MDWS)	ADR / MVI
Chemistry	CDW (via MDWS)	Local VistA (via MDWS)	CDW
Hematology	CDW (via MDWS)	Local VistA (via MDWS)	CDW
Microbiology	CDW (via MDWS)	Local VistA (via MDWS)	CDW
Problems	CDW (via MDWS)	Local VistA (via MDWS)	CDW
Allergies	CDW (via MDWS)	Local VistA (via MDWS)	CDW (via MDWS)
Medications	CDW (via MDWS)	Local VistA (via MDWS)	CDW (note that CDW only has outpatient meds)
Radiology Reports	CDW (raw) (via MDWS)	Local VistA (via MDWS)	VistA**
Provider Notes	Vista Shadows (via MDWS)	Local VistA (via MDWS)	VistA**
Veteran Self-entered Data	HealthAdapter Repository	HealthAdapter Repository	HealthAdapter Repository
Appointments (historic) Also known as Encounters	CDW (via MDWS)	Local VistA	CDW
Clinic Note	CDW (via MDWS)	Local VistA	CDW

Appointments (future)	VistA (via Pathways/CDS)	Local VistA	VistA*****
RxRefill	Vista*** (via MDWS)	NA	NA
SecureMessage	MHV (via MDWS)	MHV (via MDWS)	NA
Surgeries	CDW (via MDWS)	Local Vista (File 130)	CDW (via MDWS)
Vitals	CDW (via MDWS)	VistA (via MDWS)	CDW (via MDWS)
Facility Names (for Appt Request App)	CDW (via MDWS)	CDW (via MDWS)	N/A
Provider Names (for Appt Request App)	CDW (via MDWS)	CDW (Via MDWS)	N/A

Key

[MDWS](#)* will be used for all data services.

[VistA](#)** will be used until [CDW](#) updates become more timely

VistA*** --VistA communicates with CMOP for placing orders

"*****" Change data capture will be used to provide near real-time updates of appointment changes.

11.2. Additional data domains to be added

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Notes	Vista/TIU	Write	create / save a note for a given patient	<ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and selects a patients 3. ... accesses the note writing feature 4. ... creates a note associated 5. ... associates the note with an "encounter" 6. ... signs note 	ability to link new note to "encounter"

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Consults Orders	Vista	Read / Write	fetch by patient across site fetch by status at single site fetch by facility at single site	Use Case 1 1. Provider/clerk selects facility and authenticates 2. ... searches for and selects a patient 3. ... accesses the patient's consult orders through a tab 4. ... sees all a patient's open and pending consult orders 5. ... selects a consult order and sees its associated details Use Case 2 1. Clerk selects facility and authenticates 2. ... selects the consults tab 3. ... accesses the facility consult orders 4. ... selects a consult order and sees its associated details Use Case 3 1. Patient authenticates with DS Logon 2. ... accesses their consult orders through a tab 3. ... sees all their open and pending consult orders 4. ... selects a consult order and sees its associated details	No existing RPC to pull consult order by provider.

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Patient Search	Vista	Read	fetch by clinic/ward/provider at single site fetch list of wards	Use Case 1 1. Provider selects facility and authenticates 2. ... searches for patients by clinic 3. ... sees a list of resulting patients Use Case 2 1. Provider selects facility and authenticates 2. ... searches for patients by ward 3. ... sees a list of resulting patients Use Case 3 1. Provider selects facility and authenticates 2. ... searches for patients by clinic 3. ... sees a list of resulting patients Note: This is enhancement to our existing patient search to allow to find inpatients based on location and attending provider	Ability to search by ward
Immunization	Vista	Write	record new immunization	Mass immunization campaign in lobby of VAMC	Recording immunization in Vista

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Health Factors	Vista	Write	create / save health factors record "health factors" collected during immunization documentation and wound care assessments	Use Case 1 1. Provider selects facility and authenticates 2. ... searches for a patient 3. ... sees a list of resulting patients 4. ... selects a patient 5. ... administers a vaccination and documents associated data 6. ... saves the record Use Case 2 1. Provider selects facility and authenticates 2. ... searches for a patient 3. ... sees a list of resulting patients 4. ... selects a patient 5. ... starts a new pressure ulcer review 6. ... records data into a pressure ulcer form 7. ... saves the record	Immunization and Wound Care
Scheduling: retrieve appointment slots	Vista	Read	fetch slots for a given facility, clinic, provider, specialty for a single site	1. Clerk selects facility and authenticates 2. ... selects the availability tab 3. ... selects a timeframe to search by 4. ... enters a clinic, provider, or specialty to filter by 5. ... sees the booked and available appointments for that clinic, according to the filter set	MDWS has a service - getAppointmentDetails which can retrieve appointment details for a clinic. However, this has not been tested.

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Scheduling: appointment wait list	Vista	Read	fetch records on wait list by patient, provider, clinic for a single site	<ol style="list-style-type: none"> 1. Clerk selects facility and authenticates 2. ... selects the wait list tab 3. ... selects a timeframe to search by 4. ... selects a clinic, patient, or provider to filter the wait list by 5. ... sees the items from the wait lists, according to the filter set 	Scheduling
Scheduling: appointment wait list	Vista	Write	Update wait list (EWL, NEAR, Recall)	<ol style="list-style-type: none"> 1. Clerk selects facility and authenticates 2. ... selects the wait list tab 3. ... selects a timeframe to search by 4. ... enters a clinic, patient, or provider to filter the wait list by 5. ... sees the items from the wait lists, according to the filter set 6. ... selects an item from the wait list 7. ... books an appointment 8. ... updates the wait list accordingly 	Scheduling
Scheduling: book appointments	Vista	Write	Create new appointment	<ol style="list-style-type: none"> 1. Clerk selects facility and authenticates 2. ... searches for a patient 3. ... sees a list of resulting patients 4. ... selects a patient 5. ... starts the booking process 6. ... searches for availability 7. ... sees available time slots 8. ... books an appointment for the patient 	Scheduling

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Scheduling: cancel appointment	Vista	Write	Cancel appointment	<ol style="list-style-type: none"> 1. Clerk selects facility and authenticates 2. ... searches for a patient 3. ... selects a patient 4. ... sees a list of a patient's future appointments 5. ... selects a future appointment 6. ... cancels the appointment 	Scheduling

Others

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Notes	Vista/TIU	Read	fetch by patient at one site	<p>Use Case 1</p> <ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and selects a patients 3. ... accesses the patient's notes through a tab 4. ... sees a list of note metadata for the given patient, allowing the user to determine the type of note (progress notes vs discharge notes vs consults etc.). 5. ... selects a single note in the list and view the body of the note <p>Use Case 2</p> <ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and selects a patients 3. ... accesses the patient's notes through a tab 4. ... sees a list of notes 	n/a

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Discharge Summaries	Vista/TIU	Read	fetch by patient and/or inpatient episode at single site	<p>Use Case 1</p> <ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and select a patient 3. ... accesses the patient's discharge summary through a tab/filter 4. ... sees all discharge summaries 5. ... selects a summary and sees its associated details <p>Use Case 2</p> <ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and select a patient 3. ... accesses the patient's inpatient encounters through a tab/filter 4. ... sees a discharge summary(ies) for the inpatient encounter selected 5. ... selects a summary and sees its associated details 	n/a
Radiology Reports	Vista	Read	fetch by patient and/or radiology result at single site	<ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and selects a patient 3. ... accesses the patient's radiology reports through a tab 4. ... sees all a patient's radiology reports 5. ... selects a report and sees its associated details 	n/a

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Operative Reports	Vista	Read	fetch by patient at single site	<ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and selects a patient 3. ... accesses the patient's operative reports through a tab 4. ... sees all a patient's operative reports 5. ... selects a report and sees its associated details <p>Note: Purpose is to review a surgeon's notes related to an operation. Mentioned that this may be data from the surgery package, and not just a note</p>	n/a
Procedure Reports	Vista	Read	fetch by patient at single site	<ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and selects a patient 3. ... accesses the patient's procedure reports through a tab 4. ... sees all a patient's procedure reports 5. ... selects a report and sees its associated details 	n/a

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Patient Search	Vista	Read	fetch by clinic/ward/provider at single site	<p>Use Case 1</p> <ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for patients by clinic 3. ... sees a list of resulting patients <p>Use Case 2</p> <ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for patients by ward 3. ... sees a list of resulting patients <p>Use Case 3</p> <ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for patients by clinic 3. ... sees a list of resulting patients <p>Note: This is enhancement to our existing patient search to allow to find inpatients based on location and attending provider</p>	n/a
Immunizations	Vista	Read	fetch by patient across facilities	<ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for a patient 3. ... sees a list of resulting patients 4. ... selects a patient 5. ... sees a list of historical flu and pneumococcal vaccinations for the patient 	n/a

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
Imaging	Vista / Vista Imaging	Read	fetch by patient	<p>Outstanding question to ASD/PD is if we should use Vista Imaging services directly or use a level of abstraction</p> <ol style="list-style-type: none"> 1. Provider selects facility and authenticates 2. ... searches for and selects a patient 3. ... accesses the patient's images' metadata through a tab 4. ... selects a image to review 5. ... sees a non-diagnostic quality image 	
<i>Lab Order Entry</i>	<i>Vista</i>	<i>Write</i>	<i>Retrieve lists to support placing an order (such as list of orderables) + place an order</i>	Allow user to place an order from a mobile device	Order Entry
<i>Consult Order Entry</i>	<i>Vista</i>	<i>Write</i>	<i>Retrieve lists to support placing an order (such as list of orderables) + place an order</i>	Allow user to place an order from a mobile device	Order Entry
<i>Medication Order Entry</i>	<i>Vista</i>	<i>Write</i>	<i>Retrieve lists to support placing an order (such as list of orderables) + place an order</i>	Allow user to place an order from a mobile device	Order Entry

Data Type	Source	Read/Write	Operations	Example use case	Functionality at Risk
<i>Radiology Order Entry</i>	<i>Vista</i>	<i>Write</i>	<i>Retrieve lists to support placing an order (such as list of orderables) + place an order</i>	Allow user to place an order from a mobile device	Order Entry

11.3. Interfaces

11.3.1. Admissions

Description: Retrieve all admissions for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/admissions

Parameters: none

Response: XML or JSON object

Field	Description
admission.admittingProviderName	Name of the Admitting Provider
admission.facilityName	Name of the facility where the patient was admitted
admission.dischargeDate	Date that the patient was discharged. MM/DD/YYYY DD:HH:SS
admission.admissionDate	Date that the patient was admitted MM/DD/YYYY DD:HH:SS
dataIdentifier.uniqueId	Unique identifier for record in source system
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example:

Example

```
<ns2:admissions size="1">
```

```
<ns3:link rel="self"
```


[REDACTED]

</ns2:admission>

11.3.2. Allergies

Description: Retrieve all allergies for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/allergies

Parameters: none

Response: XML or JSON allergy object

Field	Description
allergy.active	Identifies whether an allergy is active
allergy.substance	Name of the allergy
allergy.category	The category name of the allergy
allergy.reaction	The reaction that the patient has to this allergen (substance).
allergy.severity	The severity rating of the allergy
allergy.sourceSystem	Name of the source system for the allergy
dataIdentifier.uniqueId	Unique identifier for record in source system
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

<ns2:allergies size="1">

<ns3:link rel="self"

```

h
ID26/allergies"/>
<ns2:allergy active="true" substance="Penicillin" category="Drug Allergy"
reaction="Hypotension" severity="Severe" sourceSystem="VA">
<dataIdentifier uniqueId="PATID26.1"/>
<patientIdentifier localPatientId="PATID26"/>
</ns2:allergy>

```

11.3.3. Appointments – Past

Description: Retrieve all past appointments for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/appointments/past

Parameters: none

Response: XML or JSON object

Field	Description
appointment.status	Indicates the state of the appointment
appointment.reasonForVisit	Reason for the appointment
appointment.appointmentEndDate	Appointment End Time/Date MM/DD/YYYY HH:MM:SS
appointment.appointmentStartDate	Appointment Start Time/Date MM/DD/YYYY HH:MM:SS
appointment.facilityName	Name of the facility the appointment is at
appointment.clinicType	The type of clinic the appointment is in
appointment.clinicName	The name of the clinic the appointment is at
appointment.clinicId	Clinic identifier for record in source system
appointment.providerName	The name of the provider that appointment is made for
appointment.providerId	Provider identifier for record in source system
dataidentifier.uniqueId	Unique identifier for record in source system
dataidentifier.localPatientId	
patientIdentifier.	Patient Identifier *

Links:
self-link only

Example

```
<ns2:appointments size="4">
<ns3:link rel="self"
href="
<ns2:appointment status="kept" reasonForVisit="No Notes" appointmentEndDate="01/16/2012
11:30:00" appointmentStartDate="01/16/2012 11:00:00" facilityName="DCVAMC"
clinicType="Primary Care" clinicName="Dr. Smith Clinic" clinicId="s" patientSSN=""
providerName="Provider One" providerId="provider1">
<ns3:link rel="self"
href="htt
<dataIdentifier uniqueId="5"/>
<patientIdentifier assigningAuthority="default" localPatientId="D123401"/>
</ns2:appointment>
```

11.3.4. ContactLogs

Description: Retrieve all contact logs for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/contactLogs

Parameters: none

Response: XML or JSON object

Field	Description
contactLog.logTime	Date and time contact was entered MM/DD/YYYY HH:MM:SSS
contactLog.patientId	
contactLog.note	
contactLog.contact	
contactLog.task	
contactLog.activity	
dataIdentifier.uniqueId	Unique identifier for record in source system

patientIdentifier.	Patient Identifier *
--------------------	--------------------------------------

Links:

self-link only

Example

```
<ns2:contactLogs size="1">
```

```
<ns3:link rel="self"
```

```
[REDACTED]
```

```
[REDACTED]01/27/2012 15:45:00" patientId="D123401" note="" contact="Peter"
task="Medication" activity="Phone">
```

```
<ns3:link rel="self"
```

```
href="http:[REDACTED]"
```

```
[REDACTED]
```

```
<patientIdentifier localPatientId="D123401"/>
```

```
</ns2:contactLog>
```

11.3.5. Demographics

Description: Retrieve demographics for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/demographics

Parameters: none

Response: XML or JSON object

Field	Description
demographics.phoneNumberWork	Patients Work Phone Number
demographics.phoneNumberHome	Patients Home Phone Number
demographics.phoneNumberMobile	Patients Mobile Phone Number
demographics.phoneNumberPager	Patients Pager Phone Number
nextOfKin.relationship	Relationship of next of kin to patient

nextOfKin.phoneNumber	Phone number of next of kin
nextOfKin.name	The name of the patients next of kin
address.formattedDisplayAddress1	Line 1 of the address, formatted to include multiple street address parts as a single line. For example, an address with streetAddressLine1="123 St" and streetAddressLine2="Apt A" becomes formattedDisplayAddress1="123 St, Apt A"
address.formattedDisplayAddress2	Line 2 of the address, typically be the city, state, and zip code.
address.zip	Patients zip code
address.state	Patients state
address.city	Patients city
address.streetAddressLine1	Line 1 of the patients Address
address.streetAddressLine2	Line 2 of the patients Address

Links:

self-link only

Example

```
<ns2:demographics phoneNumberWork="555-555-5555" phoneNumberHome="222-222-2222"
phoneNumberMobile="666-666-6666" phoneNumberPager="888-888-8888">
```

```
<ns3:link rel="self"
```

```
href="
```

_"/>

```
<ns2:nextOfKin relationship="Father" phoneNumber="703-555-2323" name="Smith, Father"/>
```

```
<ns2:address formattedDisplayAddress2="Chantilly, VA 20151"
```

```
formattedDisplayAddress1="L30 MAIN STREET, APT #27" zip="20151" state="VA"
```

```
city="Chantilly" streetAddressLine2="APT #27" streetAddressLine1="L30 MAIN STREET"/>
```

```
</ns2:demographics>
```

11.3.6. Diet

1 Description: Retrieve all Diet information on a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/diet

Parameters: none

Response: XML or JSON object

Field	Description
dietEntry.protein	
dietEntry.carbs	
dietEntry.fat	
dietEntry.calories	
dietEntry.entryDate	Date the entry was made into the system
dietEntry.notes	
dietEntry.mealType	
dataIdentifier.uniqueId	Unique identifier for record in source system
patientIdentifier.	Patient Identifier *

Links:

self-link only

11.3.7. Lab Micro Results

Description: Need better description.

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/lab-microbiology/results

Parameters: none

Response: XML or JSON object

Field	Description
labResult.displayDescription	Descriptive name of the lab result
labResult.displayName	That name that will be displayed for the result
labResult.specimenName	Name of the specimen tested
labResult.specimenId	ID of the specimen
labResult.orderId	ID of the order
labResult.labtype	The abnormality indicator given by the fulfilment system, typically "H" for high, "L" for low, and "" for normal or unknown abnormality.

labResult.testId	ID of the test
labResult.accessionNumber	The accession number refers to the identifier given by the fulfillment system to refer to this result.
labResult.abnormalIndicatorTriFlag	Indicator to specify that result is considered abnormal. "TRUE" if abnormal, "FALSE" if normal, and "UNKNOWN" if unknown abnormality.
labResult.abnormalIndicatorFlag	Indicator to specify that result is considered abnormal. "TRUE" if abnormal, "FALSE" if unknown or normal abnormality.
labResult.indicator	The abnormality indicator given by the fulfillment system, typically "H" for high, "L" for low, and "" for normal or unknown abnormality.
labResult.referenceLow	Reference range to indicate the lower value of normal range. For example, if a test is consider normal if the value is 100-200, the reference low is "100".
labResult.referenceHigh	Reference range to indicate the upper value of normal range. For example, if a test is consider normal if the value is 100-200, the reference high is "200".
labResult.valueUnits	The units the the lab value is given.
labResult.value	The value of the lab result.
labResult.resultedDate	Date of the test result
labResult.testname	The name of the lab test
labResult.sourceSystem	Name of the source system for this record
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:labResults size="1">
```

```
<ns3:link rel="self"
```

```

_><ns2:labResult displayDescription="AFB C&S"
displayName="AFB C&S (BLOOD)" specimenName="BLOOD" specimenId="70" orderId="1-
100" labtype="M" testId="AFB C&S" accessionNumber="microAccession1"
abnormalIndicatorTriFlag="unknown" abnormalIndicatorFlag="false" indicator=""
referenceLow="" referenceHigh="" valueUnits="" value="this is a sample small report"

```

```

resultedDate="09/01/2011 00:00:00" testname="AFB C&S" sourceSystem="VA">
<patientIdentifier localPatientId="PATID26"/>
</ns2:labResult>

```

11.3.8. Lab Micro Tests

Description: Retrieve all laboratory microbiology tests and their results for a patient.

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/lab-microbiology/tests

Parameters: none

Response: XML or JSON object

Field	Description
labTest.testname	The name of the lab test.
labTest.displayName	That name that will be displayed for the test
labTest.specimenName	Name of the specimen tested
labTest.specimenId	ID of the specimen
labTest.id	ID of the micro test
link.title="patient-lab-results-for-test"	This link points to a resource that contains the results that this patient has for this lab test.
dataidentifier.uniqueId	Unique identifier for record in source system
labResult.displayDescription	
labResult.displayName	That name that will be displayed for the micro test
labResult.specimenName	Name of the specimen used to collect the sample to perform this test
labResult.specimenId	ID of the specimen
labResult.orderId	ID of the order
labResult.labtype	Refers to the type of lab as specified in the source system. Common values are "M" for microbiology and "C" for chemistry. This data is likely unnecessary for this interface as the type of labs returned will be specified by this resource. For example, all labs returned for microbiology results will have type of "M".
labResult.testId	ID of the test

labResult.labResult	Refers to the most recent lab result(s) for this test for this patient.
---------------------	---

Links:

self-link only

Example:

```
<ns2:labTests size="2">
  <ns3:link rel="self" href="http://ihealth3.agilexhealth.com:8080/MobileHealthPlatformWeb/rest/patient/default/PATID26/lab-microbiology/tests"/>
  <ns3:labTest testname="AFB C&S (BLOOD)" displayName="AFB C&S (BLOOD)" testDescription="AFB C&S" specimenName="BLOOD" specimenId="70" id="AFB C&S">
    <ns3:link title="patient-lab-results-forest" rel="related" href="http://ihealth3.agilexhealth.com:8080/MobileHealthPlatformWeb/rest/patient/default/PATID26/lab-microbiology/results/test-id=AFB+C%26S&specimenId=70"/>
    <labResult displayDescription="AFB C&S" displayName="AFB C&S (BLOOD)" specimenName="BLOOD" specimenId="70" orderId="1-100" labtype="M" testId="AFB C&S" accessionNumber="microAccession1"
      abnormalIndicatorTriFlag="unknown" abnormalIndicatorFlag="false" indicator="" referenceLow="" referenceHigh="" valueUnits="" value="this is a sample small report" resultDate="09/01/2011 00:00:00" testname="AFB C&S"
      sourceSystem="VA">
      <patientIdentifier localPatientId="PATID26"/>
    </labResult>
    <labResult displayDescription="AFB C&S" displayName="AFB C&S (BLOOD)" specimenName="BLOOD" specimenId="70" orderId="1-102" labtype="M" testId="AFB C&S" accessionNumber="microAccession3"
      abnormalIndicatorTriFlag="unknown" abnormalIndicatorFlag="false" indicator="" referenceLow="" referenceHigh="" valueUnits="" value="---- MICROBIOLOGY ---- Accession: MI 11 2 Received: xxx Collection sample: CSF Collection
      date: xxx Site Specimen: CEREBROSPINAL FLUID Provider: PROVIDER.ONE Comment on specimen: PRELIM Test(s) ordered: AFB CULTURE & SMEAR completed:xxx * MYCOBACTERIOLOGY PRELIMINARY REPORT =>xxx TECH
      CODE: 10958 Direct Acid Fast Stain: Negative Quantity: 10 Mycobacterium: MYCOBACTERIUM XENOPI Quantity: 65 Comment: prelim ETH 60 RIF 3 STREPTOMYCIN 10.06 ISONIAZID 0.2 4 Mycobacteriology Remark(s): PRELIM "
      resultDate="08/01/2011 00:00:00" testname="AFB C&S" sourceSystem="VA">
      <patientIdentifier localPatientId="PATID26"/>
    </labResult>
    <labResult displayDescription="AFB C&S" displayName="AFB C&S (BLOOD)" specimenName="BLOOD" specimenId="70" orderId="1-101" labtype="M" testId="AFB C&S" accessionNumber="microAccession2"
      abnormalIndicatorTriFlag="unknown" abnormalIndicatorFlag="false" indicator="" referenceLow="" referenceHigh="" valueUnits="" value="---- MICROBIOLOGY ---- Accession: MI 11 2 Received: xxx Collection sample: CSF Collection
      date: xxx Site Specimen: CEREBROSPINAL FLUID Provider: PROVIDER.ONE Comment on specimen: PRELIM Test(s) ordered: AFB CULTURE & SMEAR completed:xxx * MYCOBACTERIOLOGY PRELIMINARY REPORT =>xxx TECH
      CODE: 10958 Direct Acid Fast Stain: Negative Quantity: 10 Mycobacterium: MYCOBACTERIUM XENOPI Quantity: 65 Comment: prelim ETH 60 RIF 3 STREPTOMYCIN 10.06 ISONIAZID 0.2 4 Mycobacteriology Remark(s): PRELIM "
      resultDate="07/01/2011 00:00:00" testname="AFB C&S" sourceSystem="VA">
      <patientIdentifier localPatientId="PATID26"/>
    </labResult>
    <labResult displayDescription="AFB C&S" displayName="AFB C&S (BLOOD)" specimenName="BLOOD" specimenId="70" orderId="1-103" labtype="M" testId="AFB C&S" accessionNumber="microAccession4"
      abnormalIndicatorTriFlag="unknown" abnormalIndicatorFlag="false" indicator="" referenceLow="" referenceHigh="" valueUnits="" value="---- MICROBIOLOGY ---- Accession: MI 11 2 Received: xxx Collection sample: CSF Collection
      date: xxx Site Specimen: CEREBROSPINAL FLUID Provider: PROVIDER.ONE Comment on specimen: PRELIM Test(s) ordered: AFB CULTURE & SMEAR completed:xxx * MYCOBACTERIOLOGY PRELIMINARY REPORT =>xxx TECH
      CODE: 10958 Direct Acid Fast Stain: Negative Quantity: 10 Mycobacterium: MYCOBACTERIUM XENOPI Quantity: 65 Comment: prelim ETH 60 RIF 3 STREPTOMYCIN 10.06 ISONIAZID 0.2 4 Mycobacteriology Remark(s): PRELIM "
      resultDate="01/01/2009 00:00:00" testname="AFB C&S" sourceSystem="VA">
      <patientIdentifier localPatientId="PATID26"/>
    </labResult>
  </ns3:labTest>
  <ns2:labTest testname="C&S (BLOOD)" displayName="C&S (BLOOD)" testDescription="C&S" specimenName="BLOOD" specimenId="70" id="C&S"></ns2:labTest>
</ns2:labTests>
```

11.3.9. Lab Micro Tests by Group

Description: Retrieve all laboratory microbiology tests in Groups.

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/lab-microbiology/groups

Parameters: none

Response: XML or JSON object

Field	Description
labTestGroup.name	Name of the group of tests. Typically will be a standard collection of tests, such as "CHEM7".

labTest	A test that belongs to this group.
---------	------------------------------------

Links:

self-link only

11.3.10. Lab Results

Description: Retrieve lab results for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/lab/results

Parameters: none

Response: XML or JSON object

Field	Description
labResult.specimenName	Name of the specimen tested
labResult.specimenId	ID of the specimen
labResult.orderId	ID of the order
labResult.labtype	The abnormality indicator given by the fulfilment system, typically "H" for high, "L" for low, and "" for normal or unknown abnormality.
labResult.testId	ID of the test
labResult.accessionNumber	The accession number refers to the identifier given by the fulfillment system to refer to this result.
labResult.abnormalIndicatorTriFlag	Indicator to specify that result is considered abnormal. "TRUE" if abnormal, "FALSE" if normal, and "UNKNOWN" if unknown abnormality.
labResult.abnormalIndicatorFlag	Indicator to specify that result is considered abnormal. "TRUE" if abnormal, "FALSE" if unknown or normal abnormality.
labResult.indicator	The abnormality indicator given by the fulfilment system, typically "H" for high, "L" for low, and "" for normal or unknown abnormality.
labResult.referenceLow	Reference range to indicate the lower value of normal range. For example, if a test is consider normal if the value is 100-200, the reference low is "100".
labResult.referenceHigh	Reference range to indicate the upper value of normal range. For example, if a test is consider normal if the value is 100-200, the reference high is "200".
labResult.valueUnits	The units the the lab value is given.
labResult.value	The value of the lab result.

labResult.resultedDate	Date of the test result
labResult.testname	The name of the lab test.
labResult.sourceSystem	Name of the source system for this record
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:labResults size="1">
```



```

:labResult displayDescription="oddValue" specimenName="BLOOD" specimenId="70"
orderId="oddValue" labtype="C" testId="9991" accessionNumber="1"
abnormalIndicatorTriFlag="false" abnormalIndicatorFlag="false" indicator=""
referenceLow="10" referenceHigh="1" valueUnits="K/mm3" value="99"
resultedDate="09/02/2011 00:00:00" testname="oddValue" sourceSystem="VA">

```

```
<patientIdentifier localPatientId="PATID26"/>
```

```
</ns2:labResult>
```

11.3.11. Lab Tests

Description: Retrieve lab tests and results for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/lab/tests

Parameters: none

Response: XML or JSON object

Field	Description
labTest.testname	Name of the test
labTest.displayName	That name that will be displayed for the test
labTest.specimenName	Name of the specimen tested
labTest.specimenId	ID of the specimen
labTest.id	ID of the test
link.title="patient-lab-results-for-test"	This link points to a resource that contains the results that this patient has for this lab test.

Field	Description
labResult.specimenName	Name of the specimen tested
labResult.specimenId	ID of the specimen
labResult.orderId	ID of the order
labResult.labtype	The abnormality indicator given by the fulfillment system, typically "H" for high, "L" for low, and "" for normal or unknown abnormality.
labResult.testId	ID of the test
labResult.accessionNumber	The accession number refers to the identifier given by the fulfillment system to refer to this result.
labResult.abnormalIndicatorTriFlag	Indicator to specify that result is considered abnormal. "TRUE" if abnormal, "FALSE" if normal, and "UNKNOWN" if unknown abnormality.
labResult.abnormalIndicatorFlag	Indicator to specify that result is considered abnormal. "TRUE" if abnormal, "FALSE" if unknown or normal abnormality.
labResult.indicator	The abnormality indicator given by the fulfillment system, typically "H" for high, "L" for low, and "" for normal or unknown abnormality.
labResult.referenceLow	Reference range to indicate the lower value of normal range. For example, if a test is consider normal if the value is 100-200, the reference low is "100".
labResult.referenceHigh	Reference range to indicate the upper value of normal range. For example, if a test is consider normal if the value is 100-200, the reference high is "200".
labResult.valueUnits	The units the the lab value is given.
labResult.value	The value of the lab result.
labResult.resultedDate	Date of the test result
labResult.testname	The name of the lab test.
labResult.sourceSystem	Name of the source system for this record
patientIdentifier.localPatientId	Identifier for patient from local VistA

Links:
self-link only

Example

```
<ns2:labTests size="1">
```



```
  :labTest testname="K+ (SERUM)" displayName="K+ (SERUM)"  
  testDescription="Potassium" specimenName="SERUM" specimenId="72">
```



```
    lab/results?test-id=5543&specimenId=72"/>
```

```
  <dataIdentifier uniqueId="5543"/>
```

```
  <ns2:labResult displayDescription="Potassium" specimenName="SERUM" specimenId="72"  
    orderId="172" labtype="C" testId="5543" accessionNumber="1"  
    abnormalIndicatorTriFlag="false" abnormalIndicatorFlag="false" indicator=""  
    referenceLow="3.5" referenceHigh="5" valueUnits="mmol/L" value="3.8"  
    resultedDate="05/23/2011 00:00:00" testname="K+" sourceSystem="VA"><patientIdentifier  
    localPatientId="PATID26"/>
```

```
</ns2:labResult>
```

```
</ns2:labTest>
```

11.3.12. Lab Tests by Group

[Go to start of metadata](#)

Description: Retrieve Lab tests in groups for a patient. Test returned also contain results.

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/lab/groups

Parameters: none

Response: XML or JSON allergy object

Field	Description
labTestGroup.name	
labTest.testname	Name of the test
labTest.displayName	That name that will be displayed for the test
labTest.specimenName	Name of the specimen tested

Field	Description
me	
labTest.specimenId	ID of the specimen
labTest.id	ID of the test
link.title="patient-lab-results-for-test"	This link points to a resource that contains the results that this patient has for this lab test.
labResult.specimenName	Name of the specimen tested
labResult.specimenId	ID of the specimen
labResult.orderId	ID of the order
labResult.labtype	The abnormality indicator given by the fulfillment system, typically "H" for high, "L" for low, and "" for normal or unknown abnormality.
labResult.testId	ID of the test
labResult.accessionNumber	The accession number refers to the identifier given by the fulfillment system to refer to this result.
labResult.abnormalIndicatorTriFlag	Indicator to specify that result is considered abnormal. "TRUE" if abnormal, "FALSE" if normal, and "UNKNOWN" if unknown abnormality.
labResult.abnormalIndicatorFlag	Indicator to specify that result is considered abnormal. "TRUE" if abnormal, "FALSE" if unknown or normal abnormality.
labResult.indicator	The abnormality indicator given by the fulfillment system, typically "H" for high, "L" for low, and "" for normal or unknown abnormality.
labResult.referenceLow	Reference range to indicate the lower value of normal range. For example, if a test is consider normal if the value is 100-200, the reference low is "100".
labResult.referenceHigh	Reference range to indicate the upper value of normal range. For example, if a test is consider normal if the value is 100-200, the reference high is "200".
labResult.valueUnits	The units the the lab value is given.
labResult.value	The value of the lab result.
labResult.resultedDate	Date of the test result

Field	Description
ate	
labResult.testname	The name of the lab test.
labResult.sourceSystem	Name of the source system for this record

Links:

self-link only

Example

```

<ns2:labTestGroups size="3">
  <ns3:link rel="self"
  href="lab/groups"/>
  <ns2:labTestGroup name="CBC PROFILE">
    <ns2:labTest testname="WBC (BLOOD)" displayName="WBC (BLOOD)"
    testDescription="White Blood Cells" specimenName="BLOOD" specimenId="70">
      <ns3:link title="patient-lab-results-for-test" rel="related"
      >
    </ns2:labTest>
  </ns2:labTestGroup>
  <dataIdentifier uniqueId="5721"/>
  <ns2:labResult displayDescription="White Blood Cells" specimenName="BLOOD"
  specimenId="70" orderId="18" labtype="C" testId="5721" accessionNumber="1"
  abnormalIndicatorTriFlag="false" abnormalIndicatorFlag="false" indicator=""
  referenceLow="4.5" referenceHigh="15" valueUnits="K/CMM" value="6"
  resultedDate="05/23/2011 00:00:00" testname="WBC" sourceSystem="VA">
    <patientIdentifier localPatientId="PATID26"/>
  </ns2:labResult>
</ns2:labTest>
<ns2:labTest testname="RBC (BLOOD)" displayName="RBC (BLOOD)" testDescription="Red
Blood Cells" specimenName="BLOOD" specimenId="70"></ns2:labTest>
<ns2:labTest testname="HGB (BLOOD)" displayName="HGB (BLOOD)"
testDescription="Hemoglobin" specimenName="BLOOD" specimenId="70"></ns2:labTest>
<ns2:labTest testname="HCT (BLOOD)" displayName="HCT (BLOOD)"
testDescription="Hematocrit" specimenName="BLOOD" specimenId="70"></ns2:labTest>
<ns2:labTest testname="PLT CNT (BLOOD)" displayName="PLT CNT (BLOOD)"
testDescription="Platelet Count" specimenName="BLOOD" specimenId="70"></ns2:labTest>

```

</ns2:labTestGroup>

11.3.13. Medications

Description: Retrieve all medications for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/medications

Parameters: none

Response: XML or JSON allergy object

Field	Description
medication.daysSupply	Number of days that a single dispense covers
medication.quantity	Number of units of the medication included in a single dispense of the order.
medication.refills	Number of refills for the order.
medication.lastFilledDate	Date that the medication order was last filled
medication.sig	The "sig" (instructions) for the order.
medication.status	Indicates if the order is active, inactive, on hold, cancelled, etc.
medication.medicationSource	Refers to if this medication is inpatient, outpatient, or other.
medication.prescriptionId	
medication.orderNumber	Unique identifier given by the ordering system.
medication.active	Flag to indicate if the medication is active
medication.endDate	Date that the patient stopped taking the medication as reported by the source system. If the medication is active, this will be blank.
medication.startDate	Date that the patient began taking the medication, usually the date that the medication was first filled.
medication.drugName	Name of the medication
medication.sourceSystem	Name of the source system for this record*

medication.refills	Number of refills for the order.
medication.lastFilledDate	Date that the medication order was last filled
medication.sig	The "sig" (instructions) for the order.
medication.status	Indicates if the order is active, inactive, on hold, cancelled, etc.
medication.medicationSource	Refers to if this medication is inpatient, outpatient, or other.
medication.prescriptionId	
medication.orderNumber	Unique identifier given by the ordering system.
medication.active	Flag to indicate if the medication is active
medication.endDate	Date that the patient stopped taking the medication as reported by the source system. If the medication is active, this will be blank.
medication.startDate	Date that the patient began taking the medication, usually the date that the medication was first filled.
medication.drugName	Name of the medication
medication.sourceSystem	Name of the source system for this record*
link.title="patient-medication-detail"	Link to the details of the medication, including the text report of the order history.
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:medications size="1">
```

```
<ns3:link rel="self"
```

```
ications/active"/>
```

```
<ns2:medication daysSupply="30" quantity="30" refills="3" lastFilledDate="Jun 07, 2011"
sig="TAKE ONE TABLET BY MOUTH EVERY MORNING" status="ACTIVE"
medicationSource="OUTPATIENT" prescriptionId="34R;O" orderNumber="1058"
active="true" endDate="06/07/2012 00:00:00" startDate="03/10/2010 00:00:00"
drugName="RABEPRAZOLE NA 20MG EC TAB" sourceSystem="VA">
```

```
<ns3:link rel="self"
```

[REDACTED]

[REDACTED]

[REDACTED] = "PATID26"/>

</ns2:medication>

11.3.15. Medications Inpatient

Go to start of metadata

Description: Retrieve all inpatient medications for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/medications/inpatient

Parameters: none

Response: XML or JSON allergy object

Field	Description
medication.daysSupply	Number of days that a single dispense covers
medication.quantity	Number of units of the medication included in a single dispense of the order.
medication.refills	Number of refills for the order.
medication.lastFilledDate	Date that the medication order was last filled
medication.sig	The "sig" (instructions) for the order.
medication.status	Indicates if the order is active, inactive, on hold, cancelled, etc.
medication.medicationSource	Refers to if this medication is inpatient, outpatient, or other.
medication.prescriptionId	
medication.orderNumber	Unique identifier given by the ordering system.
medication.active	Flag to indicate if the medication is active

Field	Description
medication.endDate	Date that the patient stopped taking the medication as reported by the source system. If the medication is active, this will be blank.
medication.startDate	Date that the patient began taking the medication, usually the date that the medication was first filled.
medication.drugName	Name of the medication
medication.sourceSystem	Name of the source system for this record*
link.title="patient-medication-detail"	Link to the details of the medication, including the text report of the order history.
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:medications size="1">
<ns3:link rel="self"
href="h
medications/inpatient"/>
<ns2:medication sig="GIVE 50 ml every day" status="ACTIVE"
medicationSource="INPATIENT" orderNumber="4" active="true" endDate=""
startDate="03/20/2011 00:00:00" drugName="my iv" sourceSystem="VA">
<ns3:link rel="self"
medication/4"/>
<ns3:link title="patient-medication-detail" rel="related"
href="medication/4/detail"/>
<patientIdentifier localPatientId="D123401"/>
</ns2:medication>
```

11.3.16. Medications Inpatient Current

Description: Retrieve all inpatient medications for a patient that are current/not expired.

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/medications/inpatient/current

Parameters: none

Response: XML or JSON allergy object

Field	Description
medication.sig	The "sig" (instructions) for the order.
medication.status	Indicates if the order is active, inactive, on hold, cancelled, etc.
medication.medicationSource	Refers to if this medication is inpatient, outpatient, or other.
medication.orderNumber	Unique identifier given by the ordering system.
medication.active	Flag to indicate if the medication is active
medication.endDate	Date that the patient stopped taking the medication as reported by the source system. If the medication is active, this will be blank.
medication.startDate	Date that the patient began taking the medication, usually the date that the medication was first filled.
medication.drugName	Name of the medication
medication.sourceSystem	Name of the source system for this record*
link.title="patient-medication-detail"	Link to the details of the medication, including the text report of the order history.
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:medications size="1">
```

```
<ns3:link rel="self"
```

```
  [REDACTED]  
  [REDACTED] atient/current"/>
```

```
<ns2:medication sig="GIVE 50 ml every day" status="ACTIVE"  
  medicationSource="INPATIENT" orderNumber="4" active="true" endDate=""  
  startDate="03/20/2011 00:00:00" drugName="my iv" sourceSystem="VA">
```

```
<ns3:link rel="self"
```

```
  [REDACTED]  
  [REDACTED] ion/4"/>
```

```
<ns3:link title="patient-medication-detail" rel="related"
```

[dication/4/detail"/>](#)

<patientIdentifier localPatientId="D123401"/>

</ns2:medication>

11.3.17. Medications Outpatient

Description: Retrieve all outpatient medications for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/medications/outpatient

Parameters: none

Response: XML or JSON allergy object

Field	Description
medication.daysSupply	Number of days that a single dispense covers
medication.quantity	Number of units of the medication included in a single dispense of the order.
medication.refills	Number of refills for the order.
medication.lastFilledDate	Date that the medication order was last filled
medication.sig	The "sig" (instructions) for the order.
medication.status	Indicates if the order is active, inactive, on hold, cancelled, etc.
medication.medicationSource	Refers to if this medication is inpatient, outpatient, or other.
medication.prescriptionId	
medication.orderNumber	Unique identifier given by the ordering system.
medication.active	Flag to indicate if the medication is active
medication.endDate	Date that the patient stopped taking the medication as reported by the source system. If the medication is active, this will be blank.
medication.startDate	Date that the patient began taking the medication, usually the date that the medication was first filled.
medication.drugName	Name of the medication
medication.sourceSystem	Name of the source system for this record*
link.title="patient-	Link to the details of the medication, including the text report of the

Field	Description
medication-detail"	order history.
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:medications size="1">
```

```
<ns3:link rel="self"
```

```
ns2:medication pharmacy="CVS" category="Rx" medicationSource="OUTPATIENT"
orderNumber="1" active="true" endDate="" startDate="01/01/2008 00:00:00"
drugName="AMITRIPTYLINE HCL 50 MG TAB" sourceSystem="VA">
```

```
<ns3:link rel="self"
```

```
medication/1"/>
```

```
<ns3:link title="patient-medication-detail" rel="related"
```

```
href
```

```
medication/1/detail"/>
```

```
<patientIdentifier localPatientId="D123401"/>
```

```
</ns2:medication>
```

11.3.18. Medications Outpatient Current

Description: Retrieve all outpatient medications for a patient that are current/not expired.

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/medications/outpatient/current

Parameters: none

Response: XML or JSON allergy object

Field	Description
medication.daysSupply	Number of days that a single dispense covers
medication.quantity	Number of units of the medication included in a single dispense of the order.
medication.refills	Number of refills for the order.
medication.lastFilledDate	Date that the medication order was last filled
medication.sig	The "sig" (instructions) for the order.

Field	Description
medication.status	Indicates if the order is active, inactive, on hold, cancelled, etc.
medication.medicationSource	Refers to if this medication is inpatient, outpatient, or other.
medication.prescriptionId	
medication.orderNumber	Unique identifier given by the ordering system.
medication.active	Flag to indicate if the medication is active
medication.endDate	Date that the patient stopped taking the medication as reported by the source system. If the medication is active, this will be blank.
medication.startDate	Date that the patient began taking the medication, usually the date that the medication was first filled.
medication.drugName	Name of the medication
medication.sourceSystem	Name of the source system for this record*
link.title="patient-medication-detail"	Link to the details of the medication, including the text report of the order history.
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:medications size="1">
```

```
<ns3:link rel="self"
```

```
[REDACTED]
```

```
[REDACTED] pharmacy="CVS" category="Rx" medicationSource="OUTPATIENT"
orderNumber="1" active="true" endDate="" startDate="01/01/2008 00:00:00"
drugName="AMITRIPTYLINE HCL 50 MG TAB" sourceSystem="VA">
```

```
<ns3:link rel="self"
```

```
[REDACTED]
dication/1"/>
```

```
<ns3:link title="patient-medication-detail" rel="related"
```

```
[REDACTED]
medication/1/detail"/>
```

```
<patientIdentifier localPatientId="D123401"/>
```


</ns2:medication>

11.3.19. Patient Mood

Description: Retrieve all mood vitals for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/mood

Parameters: none

Response: XML or JSON allergy object

Field	Description
moodEvents.value	
moodEvents.date	
dataIdentifier.uniqueId	Unique identifier for record in source system
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:moods size="1">
```

```
<ns3:link rel="self"
```

```
h
```

```
mood"/>
```

```
<ns2:moodEvents value="02" date="01/01/2015 00:00:00">
```

```
<ns3:link rel="self"
```

```
_/>
```

```
<dataIdentifier uniqueId="45959187-5815-4c1c-a458-a07ee2beea37"/>
```

```
<patientIdentifier localPatientId="D123401"/>
```

```
</ns2:moodEvents>
```

11.3.20. Problems

Description: Retrieve all problems for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/problems/active

Parameters: none

Response: XML or JSON allergy object

patientIdentifier.	Patient Identifier *
--------------------	--------------------------------------

Links:

self-link only

Example

```
<ns2:problems size="1">
```

```
<ns3:link rel="self"
```



```
<ns2:problem active="true" onsetDate="01/01/2010" description="Hypertension"
sourceSystem="mock">
```

```
<dataIdentifier uniqueId="PATID26.1"/>
```

```
<patientIdentifier localPatientId="PATID26"/>
```

```
</ns2:problem>
```

11.3.22. Problems Outpatient

Description: Retrieve all outpatient problems for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/problems/outpatient/active

Parameters: none

Response: XML or JSON allergy object

Field	Description
problem.active	Identifies whether the problem is active
problem.onsetDate	
problem.description	The description of the problem
medication.sourceSystem	Name of the source system for this record*
dataIdentifier.uniqueId	Unique identifier for record in source system
patientIdentifier.	Patient Identifier *

Links:

self-link only

Example

```
<ns2:problems size="1">
```

```
<ns3:link rel="self"
```

```
href="
```

[problems/outpatient/active"/>](#)

```
<ns2:problem active="true" onsetDate="01/01/2010" description="Hypertension"
sourceSystem="mock">
```

```
<dataIdentifier uniqueId="PATID26.1"/>
```

```
<patientIdentifier localPatientId="PATID26"/>
```

```
</ns2:problem>
```

11.3.23. Radiology Tests

Description: Retrieve all radiology tests for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/radiology/tests

Parameters: none

Response: XML or JSON allergy object

Field	Description
radiologyTest.testName	Name of the test.
radiologyTest.shortName	Abbreviated version of the test name.
dataIdentifier.uniqueId	Unique identifier for record in source system
radiologyResults	List of radiology results for this test for this patient.

Links:

self-link only

Example:

```

<ns2:radiologyTests size="5">
  <ns3:link rel="self" href="http://ihealth3.agilexhealth.com/MobileHealthPlatformWeb/rest/patient/default/D123401/radiology/tests"/>
  <ns2:radiologyTest testName="Abdomen X-Ray" shortName="Abdomen (Xray)">
    <dataIdentifier uniqueId="abdomenxray"/>
    <ns2:radiologyResults interpretation="Normal Chest" facilityName="CAVAMC" resultedDate="08/12/2006 00:00:00" testName="Abdomen X-Ray" testShortName="Abdomen (Xray)" testId="abdomenxray">
      <ns3:link rel="self" href="http://ihealth3.agilexhealth.com/MobileHealthPlatformWeb/rest/patient/default/D123401/radiology/results/8"/>
      <dataIdentifier uniqueId="8"/>
      <patientIdentifier assigningAuthority="default" localPatientId="D123401"/>
    </report>
    Detailed Report ECHOGRAM ABDOMEN LTD Exam Date: APR 08, 2008@09:00 Req Phys: RADIOLOGIST,ONE Pat Loc: ULTRASOUND (Req'd Loc) Img Loc: SBK ULTRASOUND Service: Unknown (Case 45 WAITING ) ECHOGRAM ABDOMEN LTD (US Detailed) CPT: 76705 Reason for Study: SEE CLINICAL HISTORY Clinical History: Stomach Pain Report Status: Verified Date Reported: APR 08, 2008 Date Verified: APR 30, 2010 Verifier E-Sig: ES,ONE RADIOLOGIST Report: TECHNIQUE: Longitudinal and transverse real-time sonographic images of the kidneys and urinary bladder are obtained. FINDINGS: The right kidney measures 10.2 cm in length and the left kidney measures 10 cm in length. The kidneys are normal in size, shape, contour and position. The cortices are normal in thickness and echogenicity. There is no evidence of hydronephrosis. Within the mid pole of the right kidney a 5.6 x 5.2 cm (L, AP) cyst is seen. A smaller cyst is demonstrated within the left mid pole that measures 1.8 x 1.6 cm (AP, W). Scanning through the pelvis demonstrates a partially distended bladder with anechoic urine. The bladder grossly appears normal. Impression: 1. BILATERAL MID POLE KIDNEY CYST. Primary Diagnostic Code: ABNORMALITY, ATTN: NEEDED Primary Interpreting Staff: ONE RADIOLOGIST, Staff Physician (Verifier) /SDR Facility: CAMP MASTER
    </report>
    <ns2:radiologyResult>
  </ns2:radiologyTest>
  <ns2:radiologyResults interpretation="Normal Chest" facilityName="SDVAMC" resultedDate="07/01/2000 00:00:00" testName="Abdomen X-Ray" testShortName="Abdomen (Xray)" testId="abdomenxray">
    <ns3:link rel="self" href="http://ihealth3.agilexhealth.com/MobileHealthPlatformWeb/rest/patient/default/D123401/radiology/results/7"/>
    <dataIdentifier uniqueId="7"/>
    <patientIdentifier assigningAuthority="default" localPatientId="D123401"/>
  </report>
  Detailed Report ECHOGRAM ABDOMEN LTD Exam Date: APR 08, 2008@09:00 Req Phys: RADIOLOGIST,ONE Pat Loc: ULTRASOUND (Req'd Loc) Img Loc: SBK ULTRASOUND Service: Unknown (Case 45 WAITING ) ECHOGRAM ABDOMEN LTD (US Detailed) CPT: 76705 Reason for Study: SEE CLINICAL HISTORY Clinical History: Stomach Pain Report Status: Verified Date Reported: APR 08, 2008 Date Verified: APR 30, 2010 Verifier E-Sig: ES,ONE RADIOLOGIST Report: TECHNIQUE: Longitudinal and transverse real-time sonographic images of the kidneys and urinary bladder are obtained. FINDINGS: The right kidney measures 10.2 cm in length and the left kidney measures 10 cm in length. The kidneys are normal in size, shape, contour and position. The cortices are normal in thickness and echogenicity. There is no evidence of hydronephrosis. Within the mid pole of the right kidney a 5.6 x 5.2 cm (L, AP) cyst is seen. A smaller cyst is demonstrated within the left mid pole that measures 1.8 x 1.6 cm (AP, W). Scanning through the pelvis demonstrates a partially distended bladder with anechoic urine. The bladder grossly appears normal. Impression: 1. BILATERAL MID POLE KIDNEY CYST. Primary Diagnostic Code: ABNORMALITY, ATTN: NEEDED Primary Interpreting Staff: ONE RADIOLOGIST, Staff Physician (Verifier) /SDR Facility: CAMP MASTER
  </report>
  <ns2:radiologyResult>
</ns2:radiologyTest>
+ <ns2:radiologyTest testName="Chest CT Scan" shortName="Chest (CT Scan)"></ns2:radiologyTest>
+ <ns2:radiologyTest testName="Chest MRI" shortName="Chest (MRI)"></ns2:radiologyTest>
+ <ns2:radiologyTest testName="Chest X-Ray PA and Lateral" shortName="Chest (Xray)"></ns2:radiologyTest>
+ <ns2:radiologyTest testName="Pelvis CT Scan" shortName="Pelvis (CT Scan)"></ns2:radiologyTest>
</ns2:radiologyTest>

```

11.3.24. Surgeries

Description: Retrieve all surgeries for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/surgeries

Parameters: none

Response: XML or JSON allergy object

Field	Description
surgery.procedureDate	
surgery.facility	
surgery.procedure	
dataIdentifier.uniqueId	Unique identifier for record in source system

Links:

self-link only

Example

```
<ns2:surgeries size="1">
```

```
<ns3:link rel="self"
```

[geries"/>](#)

```
<ns2:surgery procedureDate="01/01/2011" facility="VA" procedure="Tonsillectomy">  
<dataIdentifier uniqueId="1"/>  
</ns2:surgery>
```

11.3.25. Vitals

Description: Retrieve all vitals for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/problems/vitals

Parameters: none

Response: XML or JSON allergy object

Field	Description
vitalEntry.qualifier	
vitalEntry.entryDate	Date the entry was made into the system.
vitalEntry.notes	
vitalEntry.section	
dataIdentifier.uniqueId	Unique identifier for record in source system.
patientIdentifier.	Patient Identifier *
vitalObservations.valueUnits	
vitalObservations.value	
vitalObservations.type	
dataIdentifier.uniqueId	Unique identifier for record in source system.

Links:

self-link only

Example

```
<ns2:vitalEntries size="1">
```

[REDACTED]

[vitals"/>](#)

```
<ns2:vitalEntry qualifier="Standing" entryDate="01/30/2012 11:16:00" notes="" section="BP">
```

[REDACTED]

```

vitals/id/9547"/>
<dataIdentifier uniqueId="9547"/>
<patientIdentifier localPatientId="PATID26"/>
<vitalObservations valueUnits="mm\[Hg\]" value="88" type="8480-6">
<dataIdentifier uniqueId="14868"/>
</vitalObservations>
<vitalObservations valueUnits="mm\[Hg\]" value="77" type="8462-4">
<dataIdentifier uniqueId="14869"/>
</vitalObservations>
<vitalObservations valueUnits="bpm" value="66" type="8867-4">
<dataIdentifier uniqueId="14870"/>
</vitalObservations>
</ns2:vitalEntry>

```

11.3.26. Vitals – Blood Pressure

Description: Retrieve all bloodpressure vitals for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/vitals/bloodpressure

Parameters: none

Response: XML or JSON allergy object

Field	Description
vitalEntry.qualifier	
vitalEntry.entryDate	Date the entry was made into the system
vitalEntry.notes	
vitalEntry.section	
dataIdentifier.uniqueId	Unique identifier for record in source system.
patientIdentifier.	Patient Identifier *
vitalObservations.valueUnits	
vitalObservations.value	
vitalObservations.type	
dataIdentifier.uniqueId	Unique identifier for record in source system.

Example

11.3.27. Vitals – Height

Response: XML or JSON allergy object

Field	Description
vitalEntry.entryDate	Date the entry was made into the system.
vitalEntry.notes	
vitalEntry.section	

Field	Description
dataIdentifier.uniqueId	Unique identifier for record in source system.
patientIdentifier.	Patient Identifier *
vitalObservations.valueUnits	
vitalObservations.value	
vitalObservations.type	
dataIdentifier.uniqueId	Unique identifier for record in source system.

Links:

self-link only

Example

```
<ns2:vitalEntries size="1">
```

```
<ns3:link rel="self"
```

```
[REDACTED]
```

```
<ns2:vitalEntry entryDate="02/29/2012 09:28:00" notes="TEXT removed" section="height">
```

```
<ns3:link rel="self"
```

```
[REDACTED]
```

```
<dataIdentifier uniqueId="11497"/>
```

```
<patientIdentifier localPatientId="D123401"/>
```

```
<vitalObservations valueUnits="[in_us]" value="70" type="8302-2">
```

```
<dataIdentifier uniqueId="17148"/>
```

```
</vitalObservations>
```

```
</ns2:vitalEntry>
```

11.3.28. Vitals – Pain

Description: Retrieve all pain vitals for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/problems/vitals/pain

Parameters: none

Response: XML or JSON allergy object

Field	Description
-------	-------------

vitalEntry.qualifier	
vitalEntry.entryDate	Date the entry was made into the system.
vitalEntry.notes	
vitalEntry.section	
dataIdentifier.uniqueId	Unique identifier for record in source system.
patientIdentifier.	Patient Identifier *
vitalObservations.valueUnits	
vitalObservations.value	
vitalObservations.type	
dataIdentifier.uniqueId	Unique identifier for record in source system.

Links:

self-link only

Example

```
<ns2:vitalEntries size="1">
```

```
<ns3:link rel="self"
```

```
[REDACTED]
```

```
<ns2:vitalEntry qualifier="0" entryDate="01/03/2012 15:15:00" notes="" section="pain">
```

```
<ns3:link rel="self"
```

```
[REDACTED]
```

```
<dataIdentifier uniqueId="2221"/>
```

```
<patientIdentifier localPatientId="PATID26"/>
```

```
<vitalObservations valueUnits="none" value="2" type="pain">
```

```
<dataIdentifier uniqueId="5461"/>
```

```
</vitalObservations>
```

```
</ns2:vitalEntry>
```

```
</ns2:vitalEntries>
```

11.3.29. Vitals – Provider Entered

Description: Retrieve vital information entered by a Provider

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/vitals/providerentered

Parameters: none

Response: XML or JSON allergy object

Field	Description
vitalEntry.qualifier	
vitalEntry.entryDate	Date the entry was made into the system.
vitalEntry.notes	
vitalEntry.section	
dataIdentifier.uniqueId	Unique identifier for record in source system.
patientIdentifier.	Patient Identifier *
vitalObservations.valueUnits	
vitalObservations.value	
vitalObservations.type	

Links:

self-link only

Example

```
<ns2:vitalEntries size="1">
  <ns3:link rel="self"
    [redacted]
    [redacted]="Sitting" entryDate="01/16/2012 15:00:00" notes="Test note"
    section="BP">
    <ns3:link rel="self"
      href=[redacted]
      [redacted]
      [redacted]="A15"/>
  <patientIdentifier localPatientId="D123401"/>
  <vitalObservations valueUnits="BPM" value="75" type="8867-4"/>
  <vitalObservations valueUnits="mm(hg)" value="135" type="8480-6"/>
  <vitalObservations valueUnits="mm(hg)" value="86" type="8462-4"/>
```

</ns2:vitalEntry>

11.3.30. Vitals – Respirations

Description: Retrieve all respiration vitals for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/problems/vitals/respiration

Parameters: none

Response: XML or JSON allergy object

Field	Description
vitalEntry.size	
vitalEntry.entryDate	Date the entry was made into the system.
vitalEntry.notes	
vitalEntry.section	
dataIdentifier.uniqueId	Unique identifier for record in source system.
patientIdentifier.	Patient Identifier *
vitalObservations.valueUnits	
vitalObservations.value	
vitalObservations.type	
dataIdentifier.uniqueId	Unique identifier for record in source system.

Links:

self-link only

Example

<ns2:vitalEntries size="1">

<ns3:link rel="self"

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED] = "2076"/>

```

<patientIdentifier localPatientId="PATID26"/>
<vitalObservations valueUnits="/minute" value="40" type="9279-1">
<dataIdentifier uniqueId="5265"/>
</vitalObservations>
</ns2:vitalEntry>
</ns2:vitalEntries>

```

11.3.31. Vitals – Temperature

Description: Retrieve all tempature vitals for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/problems/vitals/temperature

Parameters: none

Response: XML or JSON allergy object

Field	Description
vitalEntry.qualifier	
vitalEntry.entryDate	Date the entry was made into the system.
vitalEntry.notes	
vitalEntry.section	
dataIdentifier.uniqueId	Unique identifier for record in source system.
patientIdentifier.	Patient Identifier *
vitalObservations.valueUnits	
vitalObservations.value	
vitalObservations.type	
dataIdentifier.uniqueId	Unique identifier for record in source system.

Links:

self-link only

Example

```

<ns2:vitalEntries size="1">
<ns3:link rel="self"

```

[vitals?section=temperature"/>](#)

```
<ns2:vitalEntry qualifier="Oral" entryDate="12/30/2011 16:24:00" notes="completely normal"
section="temperature">
```



```
<dataIdentifier uniqueId="2031"/>
<patientIdentifier localPatientId="PATID26"/>
<vitalObservations valueUnits="[degF]" value="98.6" type="8310-5">
<dataIdentifier uniqueId="5216"/>
</vitalObservations>
</ns2:vitalEntry>
```

11.3.32. Vitals – Weight

Description: Retrieve generic weight vital for a patient

URL: /MobileHealthPlatformWeb/rest/patient/default/(id)/problems/vitals/weight

Parameters: none

Response: XML or JSON allergy object

Field	Description
vitalEntry.entryDate	Date the entry was made into the system.
vitalEntry.notes	
vitalEntry.section	
dataIdentifier.uniqueId	Unique identifier for record in source system.
patientIdentifier.	Patient Identifier *
vitalObservations.valueUnits	
vitalObservations.value	
vitalObservations.type	
dataIdentifier.uniqueId	Unique identifier for record in source system.

Links:

self-link only

Example

```

<ns2:vitalEntries size="1">
<ns3:link rel="self"
[REDACTED]
>
<ns2:vitalEntry entryDate="01/01/2012 10:47:00" notes="Ggggg" section="weight">
<ns3:link rel="self"
h [REDACTED]
[REDACTED]
<dataIdentifier uniqueId="10943"/>
<patientIdentifier localPatientId="PATID26"/>
<vitalObservations valueUnits="[lb_av]" value="999" type="3141-9">
<dataIdentifier uniqueId="16507"/>
</vitalObservations>
</ns2:vitalEntry>

```

12. Appendix B – Mobile Apps

12.1. Launchpad



LP-Web-SDD 12 Feb
2014.pdf

12.2. Mobile Blue Button (MBB)



MBB SDD
Addendum.pdf

12.3. Summary of Care (SOC)



SOC-web-SDD
14Feb 2014.pdf

12.4. Burn Pit Registry



BPR-Architectureand
Design-280314-1422-