Health Services Development, Security, and Operations Support

Joint Longitudinal Viewer (JLV) 3.14.3.0
Veterans Affairs Enterprise Cloud
Deployment, Installation, Backout, and Rollback Guide (DIBRG)

Department of Veterans Affairs

December 11, 2025

Version 1.0

Booz Allen Hamilton

Contract: REDACTED

Task Order: REDACTED

CLIN: REDACTED

Revision History

Date	Version	Description	Author
12/11/2025	1.0	Submitting the document for approval	Booz Allen Hamilton
11/11/2025	0.1	Initial creation of document from last approved	Booz Allen Hamilton

Artifact Rationale

This document describes the Deployment, Installation, Backout, and Rollback Guide (DIBRG) for Joint Longitudinal Viewer (JLV) releases going into the Department of Veterans Affairs (VA) Enterprise Cloud (VAEC). The Guide includes information about system support, issue tracking, and escalation processes, and it identifies the roles and responsibilities involved in all those activities. Its purpose is to provide clients, stakeholders, and support personnel with a smooth transition to the new product or software, and this artifact should be structured appropriately to reflect the particulars of these procedures at a single or at multiple locations.

Per the Veteran-focused Integrated Process (VIP) Guide, the Deployment, Installation, Backout, and Rollback Guide is required to be completed prior to Critical Decision Point 2 (CD2), with the expectation that it is updated throughout the life cycle of the project for each build as needed.

Table of Contents

1. Introduction 1
   1.1. Purpose 1
   1.2. Dependencies 3
   1.3. Constraints 6
2. Roles and Responsibilities 6
3. Deployment 7
   3.1. Timeline 8
   3.2. Site Readiness Assessment 8
      3.2.1. Deployment Topology (Targeted Architecture) 9
      3.2.2. Site Information (Locations, Deployment Recipients) 10
      3.2.3. Site Preparation 10
   3.3. Resources 10
      3.3.1. Facility Specifics 10
      3.3.2. Hardware 11
      3.3.3. Software 12
      3.3.4. Communications 13
         3.3.4.1. Deployment/Installation/Backout Checklist 13
4. Installation 13
   4.1. Preinstallation and System Requirements 13
   4.2. Platform Installation and Preparation 13
   4.3. Download and Extract Files 15
   4.4. Database (DB) Creation 15
   4.5. Installation Scripts 15
   4.6. Cron Scripts 15
   4.7. Access Requirements and Skills Needed for Installation 15
   4.8. Installation Procedures 15
      4.8.1. Preinstallation Procedures 16
      4.8.2. Installation in ECS Cluster Environments 16
         4.8.2.1. Update JLVQoS Package 16
         4.8.2.2. Update JLVRB Package 16
         4.8.2.3. Update JLV Package 16
         4.8.2.4. Update VistADataService (VDS) Package 17
         4.8.2.5. Update jMeadows Package 17
         4.8.2.6. Steps for JLV Database Updates 17
   4.9. Installation Verification Procedures 17
   4.10. System Configuration 18
   4.11. DB Tuning 18

5. Backout Procedures 18
5.1. Backout Strategy 19
5.2. Backout Considerations 19
5.2.1. Load Testing 19
5.2.2. User Acceptance Testing (UAT) 19
5.3. Backout Criterion 19
5.4. Backout Risks 19
5.5. Authority for Backout 19
5.6. Backout Procedures 19
5.7. Backout Verification Procedures 19
6. Rollback Procedures 20
6.1. Rollback Considerations 20
6.2. Rollback Criterion 20
6.3. Rollback Risks 20
6.4. Authority for Rollback 20
6.5. Rollback Procedures 20
6.5.1. Rollback JLVQoS Package 20
6.5.2. Rollback JLVRB Package 20
6.5.3. Rollback JLV Package 21
6.5.4. Rollback VDS Package 21
6.5.5. Rollback jMeadows Package 21
6.5.6. Rollback the JLV Database 21
6.6. Rollback Verification Procedures 21
A. Acronyms and Abbreviations 22

Table of Figures

Figure 1: JLV Architecture and Components 2
Figure 2: Sample YAML Template 5
Figure 3: JLV Topology 9
Figure 4: JLV ECS and ALB Topology 10
Figure 5: Screenshot of JLV Cluster 14

Table of Tables

Table 1: Project Naming Convention 5
Table 2: Project Roles 6
Table 3: Deployment, Installation, Backout, and Rollback Roles and Responsibilities 7
Table 4: Site Preparation 10
Table 5: Container Production Configuration Specifications (AWS PaaS) 11

Table 6: Software Specifications 12
Table 7: Deployment Installation and Backout Checklist..... 13
Table 8: Implementation Plan Summary 14
Table 9: Acronyms and Abbreviations 22

1. Introduction

Born from a joint Department of Defense (DOD)–Department of Veterans Affairs (VA) venture called JANUS, Joint Longitudinal Viewer (JLV) was directed by the Secretary of the VA and the Secretary of Defense in early 2013 to further support interoperability between the two departments. JLV is a centrally hosted, Java-based web application managed as two similar but distinct products - one tailored for DOD use and another tailored for VA use. Each JLV product is deployed to its respective DOD and VA hosting environments. Although separately hosted, the respective applications use several shared data services. The browser-based, Graphical User Interface (GUI) provides an integrated, read-only view of Electronic Health Record (EHR) data from VA, DOD, and community partners within a single application.

JLV eliminates the need for VA and DOD clinicians to access disparate viewers. The GUI retrieves clinical data from several native data sources and systems, then presents it to the user via widgets, each corresponding to a clinical data domain.

Users can create and personalize tabs, drag, and drop widgets onto tabs, sort data within a widget's columns, set date filters, and expand a widget for a detailed view of patient information. Within each widget, a blue circle indicates VA data; an orange square indicates DOD data; a purple hexagon indicates community partner data; and a green triangle indicates Cerner Millennium Federal Electronic Health Record (FEHR) data.

1.1. Purpose

The Deployment, Installation, Backout, and Rollback Guide (DIBRG) provides a single, common document that defines the ordered, technical steps required to install and deploy the JLV product. Further, it outlines the steps to back out of the installation and roll back to the previously installed version of the product if necessary. The installation process is completed in the VA Enterprise Cloud (VAEC).

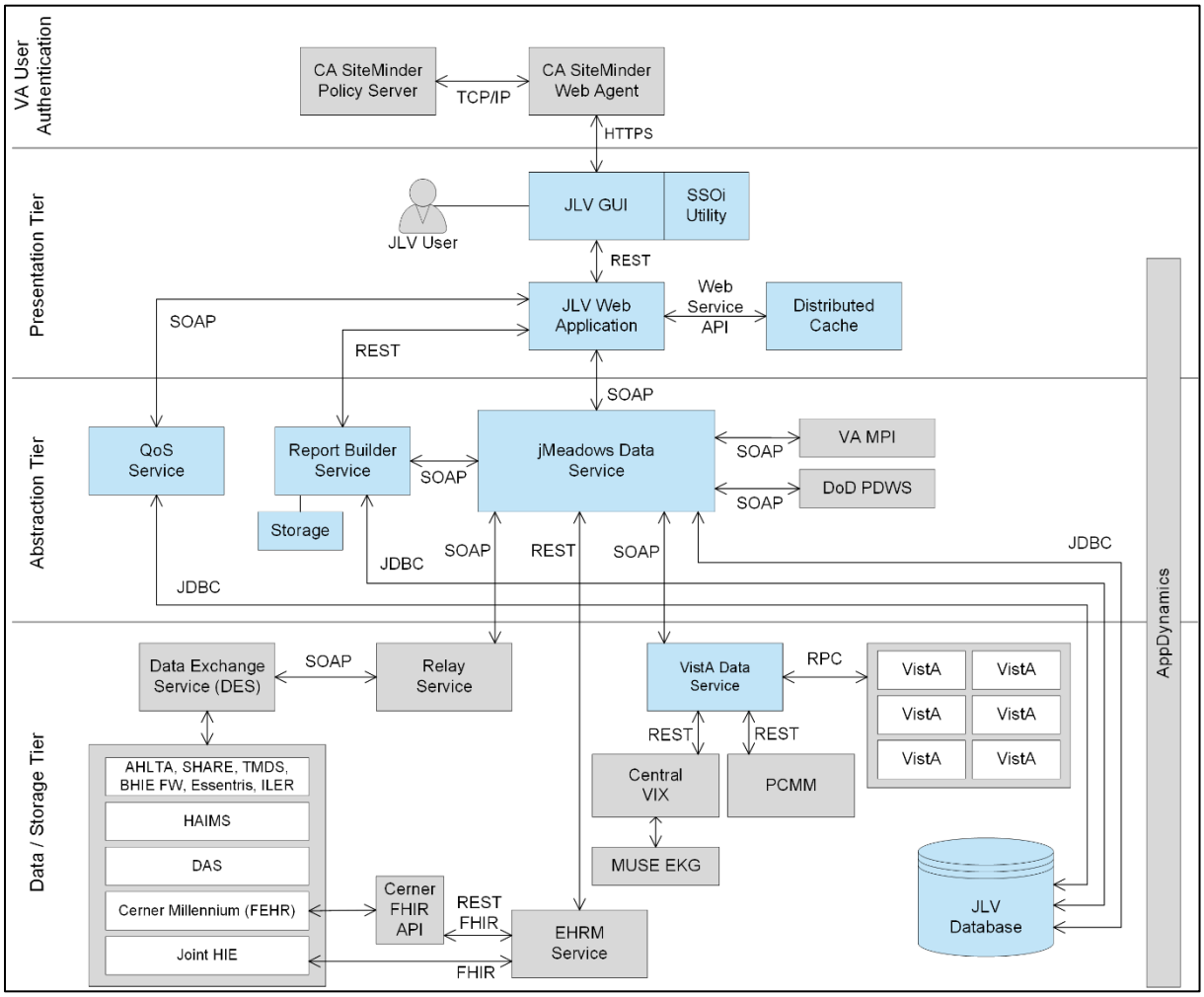
System design specifications and diagrams can be found in the VA JLV Product Repository on GitHub[1]. Please contact the GitHub Administrator to gain access if necessary.

Figure 1 illustrates the three tiers (presentation, abstraction, and data/storage) of the JLV architecture, the location of JLV system components within the tiers, the external systems that JLV communicates with in Enterprise environments, and the communication protocols and authentication methods used. The figure also shows the relationship and processing flow between JLV and the VA Identity and Access Management (IAM) services that provide Single Sign-On capability.

In the diagram, JLV components are shown with a blue background. External systems and services that JLV interfaces with are shown with a gray background.

[1] **NOTE:** Access to the VA JLV Product Repository on GitHub is restricted and must be requested.

Figure 1: JLV Architecture and Components[2]



[2] Active Directory (AD), Application Program Interface (API), Bidirectional Health Information Exchange (BHIE), Central VistA Imaging Exchange (CVIX), Clinical Data Repository (CDR), Computer Associates (CA), Data Access Service (DAS), Data Exchange Service (DES), Electronic Health Record Modernization (EHRM), Fast Healthcare Interoperability Resources (FHIR), Healthcare Artifact and Image Management Solution (HAIMS), Health Information Exchange (HIE) HyperText Transfer Protocol Secure (HTTPS), Internet Protocol (IP), Java Database (DB) Connectivity (JDBC), Lightweight Directory Access Protocol (LDAP), Master Person Index (MPI), Military Health System (MHS), Patient Discovery Web Service (PDWS), Quality of Service (QoS), Remote Procedure Calls (RPCs), Representational State Transfer (REST), Single Sign On Internal (SSOi), Simple Object Access Protocol (SOAP), Theater Medical Data Store (TMDS), Transmission Control Protocol (TCP), Veterans Health Information Systems and Technology Architecture (VistA) Data Service (VDS)

1.2. Dependencies

JLV is dependent on ancillary systems that connect the application to specific data sources. If any of these sources encounter a disruption in data services, the data is not pulled into JLV.

JLV is also dependent on internal VA update requirements, including DB updates, machine image updates, and security patches. If any of the VA Enterprise operational procedures disrupt the normal operation of JLV, the application is not fully functional.

The physical environment is Amazon Web Services (AWS) - GovCloud western region. The VAEC provides security and environmental control over the JLV containers and are restricted by Elevated Privilege (EP) access. Project personnel request EP access by submitting the *JLV and Windows Access Requirements* spreadsheet to the VA project manager (PM) / Contracting Officer Representative (COR) for approval via the Electronic Permission Access System (EPAS). AWS Console access is requested through the VAEC ticketing system. Any delay in granting initial EP access hinders the ability to respond to technical impacts to the containers.

Developers create the code which describe each individual service, the code is saved as a template. Joint Longitudinal Viewer (JLV) in the VAEC uses Docker, which is an industry standard, open-source platform making it possible to create and run containers. Containers isolate application components and their dependencies. Containers also provide virtualization and scalability. Containers bundle together an application's code with all the runtime libraries, system tools and setting needed to support the application. A container is a logical environment created on a computer where an application can run.

The first step in the process is to create a template that describes all the resource properties and the configuration of these needed resources. This template is then built and stored in a repository which can be pulled and ran in all environments from development to production.

Container technology has allowed applications to be decomposed into different smaller services. This technological approach is named Microservices. Microservices are small independent services that communicate over well-defined Application Programming Interfaces (APIs). Services are built as independent components for business capabilities that run each application process as a service and performs a single function. Each component service in a microservices architecture can be developed and scaled without affecting the functioning of other services. Services do not need to share any of their code or implementation with other services. Any communication between individual components between individual components happens via well-defined APIs.

JLV in VAEC is built on a microservices architecture.

JLV VAEC is built using:

- AWS CodeBuild – continuous integration service that builds source code.
- Docker – containerization platform.

- Nginx- Provides proxy capability.
- AWS Elastic Container Service (ECS) – container orchestration service.
- Elastic Container Registry (ECR) – Docker container registry that stores container images.
- AWS S3 – scalable, reliable, and durable object storage.
- AWS Elastic Load Balancing (ELB) – distributes incoming application traffic across multiple targets in one or more availability zone.
- AWS CloudFormation – infrastructure as code.
- AWS Elastic File Storage (EFS) – serverless file system that automatically shrinks and grows with no need for management or provisioning.
- AWS CloudWatch – monitoring and management services that provide data and actionable insights for AWS.
- AWS Relational Database Service (RDS) – web service designed for assisting with the setup, operation and scaling of a relational database for use in the application.
    - Microsoft SQL server– relational database engine developed by Microsoft.
- AWS Systems Manager Parameter Store - provides secure storage for configuration and secrets data.
- AWS Virtual Private Cloud (VPC) - a logically isolated virtual network in AWS that you define and deploy resources in.
- AWS Certificate Manager (ACM) - a service that lets you create, manage, and deploy SSL and TLS certificates.
- AWS Transit Gateway - a network hub used to interconnect VPCs and on-premises networks
- Apache Tomcat – Java server
- YAML (Yet Another Markup Language) – human friendly data serialization language for all programming languages.
- Java – object-oriented programming language.

Figure 2: Sample YAML Template

```yaml
AWSTemplateFormatVersion: 2010-09-09
Description: Production - ECR Repositories storing JLV docker images

Parameters:
  pEnvName:
    Type: String
    AllowedValues: [ silver, gold, pre-prod, prod ]
    Description: Environment Name - silver, gold, pre-prod, prod

Resources:
  rJLVRepo:
    Type: AWS::ECR::Repository
    Properties:
      ImageScanningConfiguration:
        ScanOnPush: true
      ImageTagMutability: MUTABLE
      RepositoryName: !Sub jlv-web-${pEnvName}
```

Developers create the code which describes each individual service, and the code is saved as a template. These templates are converted into Docker images.

Docker is a technology that provides the tools to build, run, test, and deploy distributed applications that are based on Linux containers. Amazon ECS uses Docker images in task definitions to launch containers as part of tasks in clusters.

The table below shows the naming convention for environments.

Table 1: Project Naming Convention

Project	Service	Environment
JLV	jMeadows	Production
JLV	JLV	Production
JLV	VistA Data Service (VDS)	Production
JLV	Report Builder (JLVRB)	Production
JLV	JLV QOS (Quality of Service)	Production

The JLV team has created two clusters in Production – JLV-Prod-A and JLV-Prod-B. Cluster JLV-Prod-A serves PROD users currently. Our goal in the future is to use both clusters for Blue/Green deployments. The designation of which cluster is Blue, or Green will be determined by which is live at the time.

Until the Blue/Green architecture is fully implemented, we will rely on manual deployments. To manually deploy JLV, each service will have to be built in AWS CodeBuild which will push the new image to AWS Elastic Container Registry (ECR). To deploy the new code, the task needs to be manually stopped which will cause the service to recreate a task with the latest container image from ECR.

When an application is developed and deployed to an AWS ECS Cluster, having two separate environments—blue and green—increases availability and reduces risk. The green environment is the production environment that will handle live traffic. The newest application version will be deployed and tested in the blue environment. After sufficient testing, we then "swap" the Application Load Balancer (ALB) targets between the two environments.

1.3. Constraints

Not applicable to JLV.

2. Roles and Responsibilities

Table 2 and Table 3 list the project and DIBRG roles and responsibilities.

Table 2: Project Roles

Name	Title/Group	Company
REDACTED	JLV Project Manager	VA
REDACTED	Scrum Master	Booz Allen Hamilton
REDACTED	Java Developer Lead	Booz Allen Hamilton
REDACTED	DevOps Engineer	Booz Allen Hamilton
REDACTED	DevOps/Cloud Lead	Booz Allen Hamilton
REDACTED	Developer	Booz Allen Hamilton
REDACTED	Business Analyst (Project Support)	Booz Allen Hamilton
REDACTED	Database Administrator	Booz Allen Hamilton
REDACTED	Project Manager	Booz Allen Hamilton
REDACTED	Information Assurance	Booz Allen Hamilton
REDACTED	Java Developer	Booz Allen Hamilton
REDACTED	Test Engineer	Booz Allen Hamilton
REDACTED	Ops Lead	Booz Allen Hamilton
REDACTED	Ops Engineer	Booz Allen Hamilton
REDACTED	Product Owner (Project Support)	Booz Allen Hamilton
REDACTED	DevOps Engineer	Booz Allen Hamilton
REDACTED	Scheduler (Project Support)	Booz Allen Hamilton
REDACTED	JLV Test Lead	Booz Allen Hamilton
REDACTED	DevOps Engineer	Booz Allen Hamilton
REDACTED	DevOps Engineer	Booz Allen Hamilton
REDACTED	Configuration Manager (Project Support)	Booz Allen Hamilton
REDACTED	Technical Writer	Booz Allen Hamilton

Name	Title/Group	Company
REDACTED	Information Assurance	Booz Allen Hamilton

Please note that references to JLV Support in the table below indicate Booz Allen Hamilton Team Operations and Engineers. See Table 7 for additional details regarding the Phase/Role column of Table 3.

Table 3: Deployment, Installation, Backout, and Rollback Roles and Responsibilities

Team	Phase/Role	Tasks
Enterprise Project Management Office (EPMO)	Approval for Release to Production	Review the Release Readiness Report (RRR) with JLV PM and Health Product Support (HPS) for approval; review and approve the Service Now (SNOW) board entry.
JLV Support	Deployment Coordination	• Plan and schedule deployments (including orchestration with vendors) • Determine and document the roles and responsibilities of those involved in deployments • Test for operational readiness
JLV Support	Installation	• Schedule installations • Ensure that the Authority to Operate (ATO) and certificate authority security documentation is in place • Coordinate training • Complete installations See Section 4.8.2 for the detailed installation process.
JLV Support	Backout	• Confirm the availability of backout instructions and backout strategy. • Identify the criteria that triggers a backout • Complete backout procedure See Section 5 for the detailed backout process.
JLV Support	Rollback	• Confirm the availability of rollback instructions and rollback strategy. • Identify the criteria that triggers a rollback • Complete rollback procedure See Section 6 for the detailed rollback process.
JLV Support	Post-Deployment	Provide software and system support.

3. Deployment

The JLV deployment workflow is as follows.

1. Once EPMO approval is complete, the JLV Support team schedules the deployment.
2. JLV Support completes a SNOW Change Order (CHG).
3. The DevOps team will create a Docker container image with the latest application code and configurations using AWS CodeBuild.
4. The container image is tagged and pushed to AWS Elastic Container Registry (ECR).

5. The AWS Elastic Container Service (ECS) Task Definition is updated with the latest ECR image version.
6. After the ECS Service is updated to deploy the latest Task Definition, the ECS service will create the respective containers using the task definition.
7. The new image will have been tested in the lower environment (Pre-Prod); however, smoke testing is conducted in Production during maintenance hours.
8. Once the deployment is complete, Production testing is verified by the JLV Support team; please see Access Requirements and Skills Needed for Installation for additional information.
9. If there are any issues, we can revert the change by updating the services with the previous task definition and image located in ECR.

3.1. Timeline

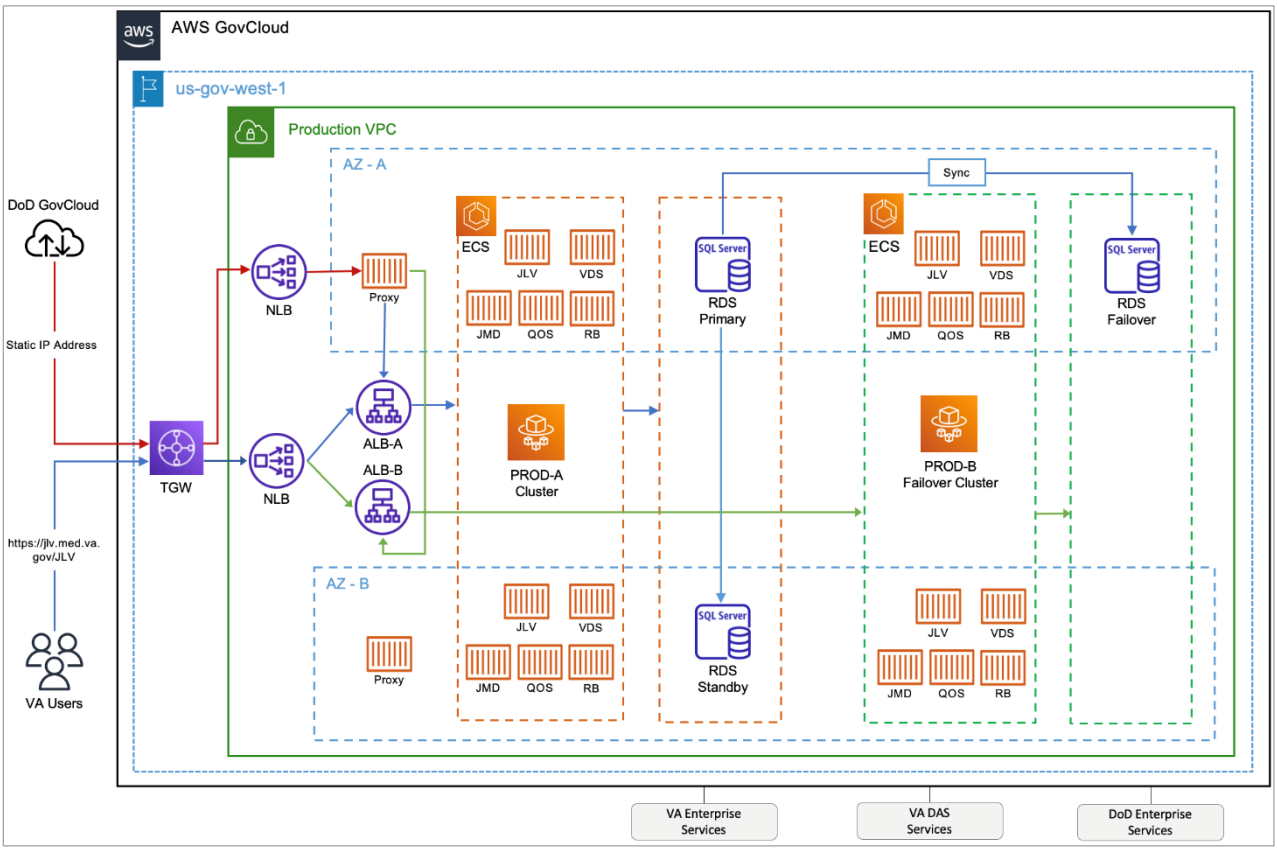
The deployment and installation have a duration of 8 hours.

3.2. Site Readiness Assessment

JLV is a Production, enterprise-wide application hosted in VAEC. All site readiness assessments are completed by JLV Operations.

3.2.1. Deployment Topology (Targeted Architecture)

Figure 3: JLV Topology[3]



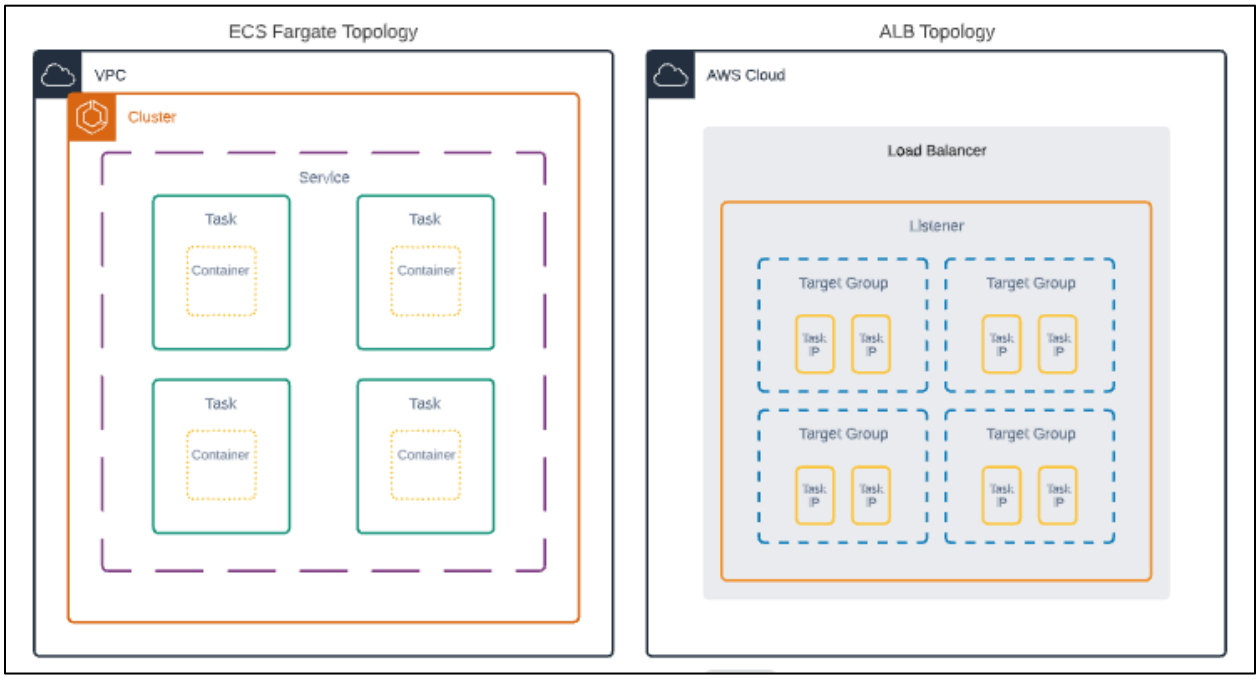
JLV is hosted on AWS with our application containerized using Docker images. The Docker images are built on AWS CodeBuild and pushed to Elastic Container Registry (ECR) image repository. The application is running in AWS Elastic Container Service (ECS) which pulls in new Docker images for each tier from ECR. ECS abstracts the need to maintain virtual servers thus making it a serverless architecture. Each tier of JLV will function as an ECS which will orchestrate task/container placement and direct traffic from the Application Load Balancer (ALB). A Task runs one or more containers and gets its instructions from its Task Definition.

Each application runs on top of Tomcat 8.5 which connects to the SQL database (DB) hosted on Relational Database Service (RDS). The JLV web tier task runs two (2) containers, one for the SiteMinder Identity Access Management connection, and the other for the main application that uses Apache Httpd as a proxy to Tomcat. The Report Builder tier uses Amazon Elastic File Storage (EFS) to store report files. All tiers use AppDynamics and Amazon CloudWatch for logging and monitoring. Logs are also shipped to the VAEC's Central Logging Solution to be viewed in Kibana.

There is a Network Load Balancer (NLB) setup for the DOD JLV connection which accepts traffic and forwards to the ALB.

[3] Note that all inbound and outbound traffic to the JLV Virtual Private Cloud (VPC) will pass through an AWS Transit Gateway, (which is not shown on the figure).

Figure 4: JLV ECS and ALB Topology



3.2.2. Site Information (Locations, Deployment Recipients)

The host site for JLV is VAEC located in the AWS west region.

3.2.3. Site Preparation

All site preparation is completed by JLV Operations team.

JLV application container images are built using the latest Amazon managed, Amazon:Linux2 base image, which contains the latest security patches/packages during every new image build.

Table 4 describes the preparation required by the site(s) prior to deployment.

Table 4: Site Preparation

Site	Problem/Change Needed	Features to Adapt/Modify to New Product	Actions/Steps	Owner
VAEC	Pull the latest Amazon: Linux 2 container image	Non-identifiable	Implement/Verify	JLV OPS

3.3. Resources

Descriptions of the hardware, software, facilities, and documentation are detailed in the following subsections.

3.3.1. Facility Specifics

VAEC is a cloud-based General Support System (GSS), hosted on AWS GovCloud; FedRAMP package #: REDACTED. As per AWS's FedRAMP authorization, AWS is responsible for all

Disaster Recovery Plan (DRP) activities within this environment. AWS's DRP can be found in FedRAMP package #: REDACTED [4]. The AWS is designed utilizing separate data centers (availability zones - AZs), and geographically separated regions. Currently AWS has two regions located in Iowa and Virginia.

The VAEC AWS GovCloud High is a GSS that provides a secure application and hosting environment for VA applications, content, and utilities. These applications and services are used to deliver content to an audience made up of employees, Veterans, contractors, partners across all VA medical centers and component facilities, Federal government, and the general public. Content and applications are provided by Veterans Benefits Administration (VBA), Veterans Health Administration (VHA), National Cemetery Administration (NCA), and VA-level support offices. VAEC provides the following services: Content delivery, Application Hosting and Management Services.

3.3.2. Hardware

The VAEC infrastructure is hosted by AWS GovCloud, a cloud service provider. The AWS GovCloud platform is used to provide a variety of hosting environments to suit a variety of needs. AWS GovCloud can support applications categorized up to "High" as rated in accordance with Federal Information Processing Standard (FIPS) 199. VA applications available to the public are hosted in AWS GovCloud.

A dedicated private data link (AWS Direct Connect) provides all connectivity for VA resources communicating to the environment. Virtual Private Clouds (VPCs) wrap the applications within AWS GovCloud to encapsulate network access. Access from the applications to VA internal resources such as Identity, Credential, and Access Management (ICAM) and Active Directory (AD) Services are conducted over the encrypted private data link to the VA Network.

AWS is in two (2) regions with three (3) Availability Zones in each region designed to allow United States (U.S.) government agencies, contractors, and customers to move sensitive workloads into the cloud for addressing specific regulatory and compliance requirements. AWS GovCloud does not manage logical access controls within the VAEC system boundary. VAEC offers the same level of security as other VA physical technology centers and supports existing VA security controls and certification requirements such as FISMA, Health Insurance Portability and Accountability Act of 1996 (HIPAA), Health Information Technology for Economic and Clinical Health (HITECH), SAS-70, ISO 27001, FIPS 140-2 compliant end points, and PCI DS

Table 5 describes the hardware specifications required at each site prior to deployment. Please see Table 3 for details about the party or parties responsible for preparing the site to meet the hardware specifications.

Table 5: Container Production Configuration Specifications (AWS PaaS)

Required Hardware	Model	Configuration	Manufacturer	Number of Containers
JLVRB Container	Amazon Linux 2	vCPU 2 RAM 16 GB Storage 11.52 GiB	Virtual	Automatically Scaled

[4] The link to the AWS DRP will need to be requested directly from FedRAMP.

Required Hardware	Model	Configuration	Manufacturer	Number of Containers
jMeadows Container	Amazon Linux 2	vCPU 4 RAM 16 GB	Virtual	Automatically Scaled
VistADataService Container	Amazon Linux 2	vCPU 4 RAM 16 GB	Virtual	Automatically Scaled
JLVQoS Container	Amazon Linux 2	vCPU 2 RAM 8 GB	Virtual	One, with automatic failover.
JLV Web Container (main)	Amazon Linux 2	vCPU 2 RAM 20 GB	Virtual	Automatically Scaled
Single Sign on Internal (SSOi) Container (sidecar)	Amazon Linux 2	vCPU 2 RAM 8 GB	Virtual	Automatically Scaled
RDS Instance	SQL Server Enterprise Edition 15.00.4236.7.v1	vCPU 16 RAM 64 GB Storage 2 TB	Virtual	One, with automatic failover.

3.3.3. Software

Table 6 describes the software specifications required at each site prior to deployment. Please see Table 3 for details about the party or parties responsible for preparing the site to meet the software specifications.

Table 6: Software Specifications

Required Software	Make	Version	Manufacturer	Other
SQL Server Enterprise Edition on AWS RDS	N/A	15.00.4236.7.v1	Microsoft	N/A
Amazon Linux	N/A	2	Amazon	N/A
Apache	N/A	2.4	Apache	N/A
SiteMinder	N/A	12.51	CA Technologies	N/A
Tomcat	N/A	8.5	Apache	N/A

More information about SSOi server installation can be found in the *CA SiteMinder Apache Web Agent Install & Configuration Guide,* maintained by Identity and Access Management (IAM).

3.3.4. Communications

JLV Operations team performs JLV installation and deployment activities in the virtualized environments at VAEC utilizing the release-ready package. When possible, the installation is performed during off-hours to minimize the impact on users. We will use a 60-minute window for downtime to manually deploy and test.

An overview of typical steps and/or communication during the implementation process is as follows:

1. The Configuration Manager submits a JLV release (RLSE) notification via SNOW/CHG Order
2. Plan the system upgrade and change notifications:
   a. Notify the JLV PMs and the OIT PM/COR
3. Perform the installation/deployment:
4. The Scrum Master notifies the stakeholders and Product team that systems are updated.

3.3.4.1. Deployment/Installation/Backout Checklist

Table 7 captures the coordination effort and documents the day/time/individual when each activity (deploy, install, back-out) is completed for a project.

Table 7: Deployment Installation and Backout Checklist

Activity	Day	Time	Completed By
Deployment	Decision by VA OIT	Deployment dependent on a planned maintenance ticket	JLV Development Operations
Installation	Deployments staged	Coordinated with JLV OPS and JLV DevOps	JLV Development Operations
Backout	As needed	As needed, with a time estimate to be communicated to stakeholders when determined	JLV Development Operations

4. Installation

4.1. Preinstallation and System Requirements

Please see the Hardware and Software sections for information regarding preinstallation system requirements.

4.2. Platform Installation and Preparation

JLV is being installed within the JLV VAEC Virtual Private Clouds (VPC) using the ECS managed service. The JLV services are stored as container images in ECR.

Docker technology is used to create file images. Docker is a set of "platform as a service" (PAAS) products that use Operating System (OS)-level virtualization to deliver software in packages called containers.

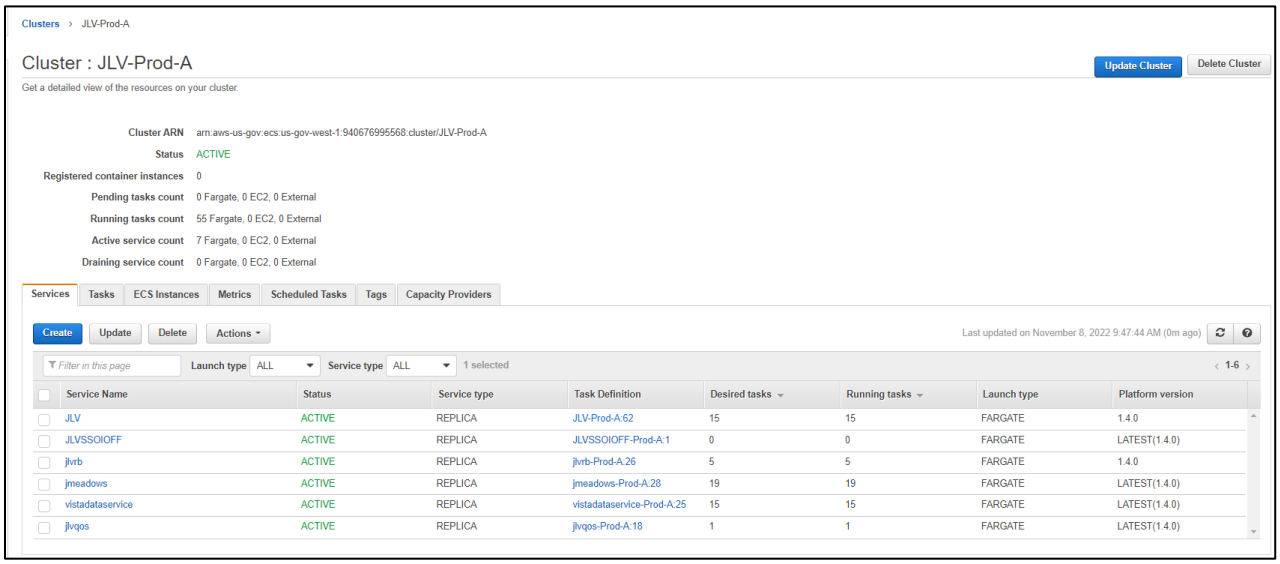
When a Docker file is built, the configuration instructions are executed, and the artifacts are installed into the image. This activity creates the container image.

Docker images are packaged up and sent to AWS ECR. When changes are made to individual services, a new Docker image is created, and the new image is sent to ECR, where a new container is created to replace the outdated container.

Table 8: Implementation Plan Summary

Considerations	**Associated Details**	
What systems are affected?	**Component:**	**Deployed to:**
	JLV (Web/SSOi)	VAEC Environment
	JLV Report Builder	VAEC Environment
	jMeadows Data Service	VAEC Environment
	VDS	VAEC Environment
	JLV QoS	VAEC Environment
Who is impacted by the change?	JLV users	
What is the estimated timeframe for restoring service?	Estimated down time is 60 minutes for deployment and testing.	
What pre-implementation work is required?	Build Docker images using AWS CodeBuild.	

Figure 5: Screenshot of JLV Cluster



JLV ECS Cluster is a grouping of containers which together form the JLV application.

- There is a setting in the cluster which describes the number of containers that are needed at any one time.

    - Any time one of the containers crashes, the ECS service will automatically create a new container to ensure the desired number of containers is maintained.
- Each container is described by the task definition associated with each specific container.
- Task Definitions are stored in ECS.
- ECR stores and manages container images.

4.3. Download and Extract Files

The application code is pulled into the Docker file from GitHub and built during the container build process. Configuration files and certificates are pulled into the Docker file from a Nexus artifact repository located on a server in the AWS JLV domain.

4.4. Database (DB) Creation

The JLV DB was migrated to AWS Relational Database Service (RDS). It uses an SQL Server Enterprise Edition DB, to store user profile information, audit records, and medical standard translation and mapping reference tables.

System design specifications and diagrams can be found in the VA JLV Product Repository on GitHub.

4.5. Installation Scripts

This does not apply to JLV. There are only Docker files which are deployment templates for creating images. Docker will install any software, configurations, and artifacts needed.

4.6. Cron Scripts

Cron scripts are Linux-based events. There are currently no cron scripts run for JLV.

The singular, regularly scheduled event is the hourly server health check, a Windows-based event.

4.7. Access Requirements and Skills Needed for Installation

AWS Console or Command Line Interface (CLI) access is required for installation activities. JLV System Engineers have been granted access, and they are designated to access the *jump servers* for deployment, maintenance, and backout activities. A jump server, jump host or jump box is a system on a network used to access and manage devices in a separate security zone. This document assumes the installer has knowledge and experience with the AWS cloud platform, Docker containers, MS SQL Server, Linux, Apache, and Tomcat, in addition to a general understanding of the web-based applications and familiarity with networking and basic troubleshooting, such as Telnet and ping.

4.8. Installation Procedures

The subsections below detail pre-installation, and installation procedures.

Note: Estimated down time is 60 minutes for deployment and testing.

4.8.1. Preinstallation Procedures

Record the JLV software version number to be installed 3.14.3.0 as well as the software version number of the previous installation, 3.14.3.0.

If the database changes are needed, follow the steps below in section 4.8.2.7.

Once the preinstallation activities are complete, the installation of JLV system components begins in the primary environment.

4.8.2. Installation in ECS Cluster Environments

Complete these installation steps and validate the installation according to the steps in Installation Verification Procedures. See the Deployment workflow for the detailed process.

4.8.2.1. Update JLVQoS Package

1. Login to AWS CodeBuild in the Console.
2. Navigate to Build projects and select “jlvqos-prod”
3. Select Start build with overrides"
4. Go to the Environment variables override section and expand the details.
5. Update the "IMAGE_TAG" variable with the proper tag, 3.14.3.0, and verify other environment variables.
6. Select "Start build".
7. Go to AWS ECS in the Console and select “Clusters”.
8. Select the JLV-Prod-A cluster
9. Select the JLV-QoS task
10. Stop the JLV-QoS task – AWS will automatically start a new task with the latest version.

4.8.2.2. Update JLVRB Package

1. Login to AWS CodeBuild in the Console.
2. Navigate to Build projects and select “reportbuilder-prod”
3. Select Start build with overrides"
4. Go to the Environment variables override section and expand the details.
5. Update the "IMAGE_TAG" variable with the proper tag, 3.14.3.0 and verify other environment variables.
6. Select "Start build".
7. Go to CloudFormation in the Console and select “project-ecs-jlv-rb-prod”.
8. Select “Update” then “Use current template”.
9. Change pBuildTag to the IMAGE_TAG variable that was created in step 5.
10. Select “Update Stack”
11. Wait for the old task definition to drain or stop the process.

4.8.2.3. Update JLV Package

1. Login to AWS CodeBuild in the Console.
2. Navigate to Build projects and select “jlv-web-prod”
3. Select Start build with overrides"
4. Go to the Environment variables override section and expand the details.

5. Update the "IMAGE_TAG" variable with the proper tag, 3.14.3.0, and verify other environment variables.
6. Select "Start build".
7. Go to CloudFormation in the Console and select “project-ecs-web-prod”.
8. Select “Update” then “Use current template”.
9. Change pBuildTag to the IMAGE_TAG variable that was created in step 5.
10. Select “Update Stack”
11. Wait for the old task definition to drain or stop the process.

4.8.2.4. Update VistADataService (VDS) Package

1. Login to AWS CodeBuild in the Console.
2. Navigate to Build projects and select “vds-prod”
3. Select Start build with overrides"
4. Go to the Environment variables override section and expand the details.
5. Update the "IMAGE_TAG" variable with the proper tag, 3.14.3.0, and verify other environment variables.
6. Select "Start build".
7. Go to CloudFormation in the Console and select “project-ecs-vistadataservice-prod”.
8. Select “Update” then “Use current template”.
9. Change pBuildTag to the IMAGE_TAG variable that was created in step 5.
10. Select “Update Stack”
11. Wait for the old task definition to drain or stop the process.

4.8.2.5. Update jMeadows Package

1. Login to AWS CodeBuild in the Console.
2. Navigate to Build projects and select “jmeadows-prod”
3. Select Start build with overrides"
4. Go to the Environment variables override section and expand the details.
5. Update the "IMAGE_TAG" variable with the proper tag, 3.14.3.0, and verify other environment variables.
6. Select "Start build".
7. Go to CloudFormation in the Console and select “project-ecs-jmeadows-prod”.
8. Select “Update” then “Use current template”.
9. Change pBuildTag to the IMAGE_TAG variable that was created in step 5.
10. Select “Update Stack”
11. Wait for the old task definition to drain or stop the process.

4.8.2.6. Steps for JLV Database Updates

1. There are no updates for this release.

4.9. Installation Verification Procedures

After completing the installation processes detailed in Installation Procedures the Operations Team performs a manual smoke test. Use the steps below to test each module as an end user to validate the installation, deployment, and functionality of all JLV applications and services.

1. Validate that JLV is running
    a. Access the JLV application web page using the following URL: REDACTED
2. Validate that VA Personal Identification Verification (PIV) and Personal Identification Number (PIN) are accepted by SSOi
3. Validate that the system status appears on the JLV **Login** page
    a. **Expected Result:** The system status should show a circular, green icon with a white checkmark
4. Validate the ability to log in with VA credentials
5. Validate that VA data displays within the JLV widgets, using test patients REDACTED and REDACTED
6. Validate that FEHR data displays within the JLV widgets
7. Validate that community partner data displays within the JLV widgets
8. Validate that VA terminology mapping occurs
    a. **Expected Result:** VA terminology is properly mapped in the JLV widgets
9. Validate that DOD terminology mapping occurs
    a. **Expected Result:** DOD terminology is properly mapped in the JLV widgets
10. Validate that QoS is running by verifying that QoS is writing updates to the DB in the QoS_LOGS table
    a. Run the following command in SSMS on the active MSSQL server: select top 100 * from jlv.dbo.QOS_LOGS and order by date desc
        i. **Expected Result:** The select top 100* results will be displayed and indicate a time stamp of within 5 minutes of the time the query was run
        ii. If the top rows do not show, double check the installation steps
11. Validate that Report Builder is running
    a. **Expected Result:** The user can add items to the Report Builder and generate a printable Portable Document Format (PDF) file
12. Once all verification steps are complete, JLV users are rerouted back to the primary Production operating environment

4.10. System Configuration

Table 5 describes the server configurations for JLV Enterprise Production infrastructure hosted at the VAEC.

4.11. DB Tuning

JLV Engineering/Development ensure DB indexing for each release. The DB schema and the performance-based jobs are validated by JLV Database Administrators (DBAs) for the deployment of each release.

5. Backout Procedures

Both backout and rollback procedures are performed consistently for each JLV service to return to the last known good operational state of the software and platform settings.

5.1. Backout Strategy

If a backout/rollback is required for one of the JLV services deployed to Prod ECS cluster, the JLV team will run the CloudFormation template, ecs-service-fargate.yml, setting the "IMAGE_TAG" variable to the previous 3.14.2.0 image version.

5.2. Backout Considerations

The following subsections detail the considerations for backing out of the current installation of JLV.

5.2.1. Load Testing

Load testing is coordinated with the VA Enterprise Testing Service (ETS) team, however there is no load testing planned for this release.

5.2.2. User Acceptance Testing (UAT)

UAT results were not available at the time of this writing. When all testing cycles (including UAT) are complete, the data is made available in the VA JLV Product Repository in GitHub.

5.3. Backout Criterion

The criterion for backing out of the current installation is that JLV does not operate as intended when tested by VA and partner testers and the JLV Support team.

5.4. Backout Risks

The risks for executing the backout are minimal because a backout is performed during planned downtime when user interruption is minimized. Once the restored system is online and validated, users may access the system again.

If a backout is initiated later in the deployment window, restoration time may exceed the downtime planned for deployment. This risk is mitigated by scheduling deployments for weekends and other times when expected usage levels are low.

5.5. Authority for Backout

If a backout is necessary, approval for the backout comes from the VA Project Manager REDACTED.

5.6. Backout Procedures

Because backout and rollback are performed consecutively, the backout and rollback procedures are combined in Rollback Procedures.

5.7. Backout Verification Procedures

See Installation Verification Procedures.

6. Rollback Procedures

If a rollback is required for one of the JLV services deployed to Prod ECS cluster, the JLV team will run the CloudFormation template, ecs-service-fargate.yml, setting the "IMAGE_TAG" variable to the previous 3.14.2.0 image version.

6.1. Rollback Considerations

The consideration for performing a rollback is that the JLV application does not operate as intended when tested by the JLV Support team.

6.2. Rollback Criterion

The criterion for performing a rollback is that the JLV application does not operate as intended when tested by the JLV Support team.

6.3. Rollback Risks

The risks for executing a rollback are minimal because the system will revert to the last working image/configuration.

6.4. Authority for Rollback

If a rollback is necessary, approval for the rollback comes from the VA Project Manager REDACTED.

6.5. Rollback Procedures

AWS ECR will maintain versions of previous releases. After the initial rollout of the JLV Cloud instance, a rollback would involve restarting a previous version, 3.14.2.0 of the JLV application.

If it is determined that a rollback is necessary, task definitions with the previous image version will be deployed to ECS.

6.5.1. Rollback JLVQoS Package

1. Go to CloudFormation in the Console and select “project-ecs-jlvqos-prod”.
2. Select “Update” then “Use current template”.
3. Change pBuildTag to the previous version variable, 3.14.2.0
4. Select “Update Stack”
5. Wait for the old task definition to drain or stop the process.

6.5.2. Rollback JLVRB Package

1. Go to CloudFormation in the Console and select “project-ecs-jlvrb-prod”.
2. Select “Update” then “Use current template”.
3. Change pBuildTag to the previous version variable, 3.14.2.0
4. Select “Update Stack”
5. Wait for the old task definition to drain or stop the process.

6.5.3. Rollback JLV Package

1. Go to CloudFormation in the Console and select “project-ecs-jlv-prod”.
2. Select “Update” then “Use current template”.
3. Change pBuildTag to the previous version variable, 3.14.2.0
4. Select “Update Stack”
5. Wait for the old task definition to drain or stop the process.

6.5.4. Rollback VDS Package

1. Go to CloudFormation in the Console and select “project-ecs-vistadataservice-prod”.
2. Select “Update” then “Use current template”.
3. Change pBuildTag to the previous version variable, 3.14.2.0
4. Select “Update Stack”
5. Wait for the old task definition to drain or stop the process.

6.5.5. Rollback jMeadows Package

1. Go to CloudFormation in the Console and select “project-ecs-jmeadows-prod”.
2. Select “Update” then “Use current template”.
3. Change pBuildTag to the previous version variable, 3.14.2.0
4. Select “Update Stack”
5. Wait for the old task definition to drain or stop the process.

6.5.6. Rollback the JLV Database

1. There are no updates for this release.

6.6. Rollback Verification Procedures

After completing the rollback procedures, perform the validation steps in Installation Verification Procedures.

A. Acronyms and Abbreviations

Table 9 lists the acronyms and abbreviations are used throughout this document.

Table 9: Acronyms and Abbreviations

Acronym	Definition
AD	Active Directory
ALB	Application Load Balancer
API	Application Programming Interface
ATO	Authority to Operate
AWS	Amazon Web Services
BHIE	Bidirectional Health Information Exchange
CA	Computer Associates
CD2	Critical Decision Point 2
CDR	Clinical Data Repository
CM	Change Management
CHDR	Clinical/Health Data Repository
CHG	Change Order
COR	Contracting Officer's Representative
CVIX	Central VistA Imaging Exchange
DAS	Data Access Service
DB	Database
DBA	Database Administrator
DES	Data Exchange Service
DIBRG	Deployment, Installation, Backout, and Rollback Guide
DRP	Disaster Recovery Plan
DOD	Department of Defense
ECR	Elastic Container Registry
ECS	Elastic Container Service
EHR	Electronic Health Record
EHRM	Electronic Health Record Modernization
EP	Elevated Privilege
EPAS	Electronic Permission Access System
EPMO	Enterprise Program Management Office
FEHR	Federal Electronic Health Record
FHIR	Fast Healthcare Interoperability Resources
FIPS	Federal Information Processing Standard
GB	Gigabyte
GUI	Graphical User Interface
HAIMS	Healthcare Artifact and Image Management Solution
HIE	Health Information Exchange
HITECH	Health Information Technology for Economic and Clinical Health
HTTPS	Hypertext Transfer Protocol Secure
IAM	Identity and Access Management

Acronym	Definition
IP	Internet Protocol
IT	Information Technology
JDBC	Java Database Connectivity
JLV	Joint Longitudinal Viewer
LDAP	Lightweight Directory Access Protocol
MHS	Military Health System
MS	Microsoft
MPI	Master Person Index
NCA	National Cemetery Administration
OIT	Office of Information and Technology
OS	Operating System
PAAS	Platform As A Service
PDWS	Patient Discovery Web Service
PDF	Portable Document Format
PIN	Personal Identification Number
PIV	Personal Identification Verification
PM	Project Manager
POC	Point of Contact
POM	Production Operations Manual
QoS	Quality of Service
RAM	Random Access Memory
RDS	Relational Database Service
REST	Representational State Transfer
RPC	Remote Procedure Calls
RRR	Release Readiness Report
SOAP	Simple Object Access Protocol
SNOW	Service Now
SSMS	SQL Server Management Studio
SSOi	Single Sign on Internal
SQL	Structured Query Language
TCP	Transmission Control Protocol
TMDS	Theater Medical Data Store
UAT	User Acceptance Testing
URL	Universal Resource Locator
U.S.	United States
VA	Department of Veterans Affairs
VAEC	VA Enterprise Cloud
VBA	Veterans Benefits Administration
VDS	VistA Data Service
VHA	Veterans Health Administration
VIP	Veteran-Focused Integration Process
VistA	Veterans Information Systems and Technology Architecture

Acronym	Definition
VM	Virtual Machine
YAML	Yet Another Markup Language