# Workspace Web Edition Private Edition Guide

5/20/2022

# Table of Contents

# Contents

Find links to all the topics in this guide.

**Related documentation:**

- 
- 

Workspace Web Edition is a service available with the Genesys Multicloud CX private edition offering.

## Overview

Learn more about Workspace Web Edition, its architecture, and how to support high availability and disaster recovery.

- About Workspace Web Edition
- Architecture
- High availability and disaster recovery

## Configure and deploy

Find out how to configure and deploy Workspace Web Edition.

- Before you begin
- Configure Workspace Web Edition
- Provision Workspace Web Edition
- Deploy Workspace Web Edition
- Upgrade, rollback, or uninstall Workspace Web Edition

## Observability

Learn how to monitor Workspace Web Edition with metrics and logging.

- Observability in Workspace Web Edition
- Workspace Web Edition metrics and alerts

- Logging

# About Workspace Web Edition

## Contents

Learn about Workspace Web Edition and how it works in Genesys Multicloud CX private edition.

**Related documentation:**

- 
- 

Workspace Web Edition Agent Desktop enables contact center agents and supervisors to communicate with customers and team members through phone calls, Outbound Campaigns, and Genesys Digital channels, such as chat, email, social media, SMS, WhatsApp, and workitems. Supervisors can use Workspace to monitor and coach their teams.

Workspace Agent Desktop lets contact center agents and supervisors:

- communicate with customers and team members through phone calls and Outbound Campaigns and Genesys Digital channels, including voice, chat, email, social media, SMS, WhatsApp, and workitems
- get help from team members
- meet contact center expectations and personal KPIs
- find and manage contact information
- retrieve work from personal and group workbins
- search for existing interactions
- provide standard responses
- engage in co-browsing your corporate website
- track customer journeys

## Cloud platform support

Workspace Web Edition is supported on the following cloud platforms:

- Google Kubernetes Engine (GKE)
- OpenShift Container Platform (OpenShift)

See the Agent Desktop and Gplus Adapter Release Notes for information about when support was introduced.

# Architecture

## Contents

Learn about Workspace Web Edition architecture.

**Related documentation:**

-
-

## Introduction

The static Workspace Web Edition (WWE) code is bundled in a dedicated NGINX Docker image at build time, ensuring the immutability of the WWE web application content in a Private Edition Kubernetes cluster environment.

WWE is shipped as a Docker image to take advantage of the Multicloud standards such as:

- Pipeline (including Security)
- Repositories
- Shipping logistics

WWE is deployed as part of the Genesys Web Services Kubernetes Services. Client users (agents) use a web browser running the Workspace Agent Desktop to access Workspace Web Edition (WWE), Genesys Web Services (GWS), Genesys Authentication Service (Gauth), and WebRTC containers through the inbound gateway.

In the architecture diagram, the dotted lines from the browser, going through External Ingress and Ingress Controller, to the non-wwe namespace resources represents the connectivity to the back-end services that expose the APIs used by the WWE web application running in the browser. These services are required for WWE to operate. Refer to the following documentation for details about their respective connectivity:

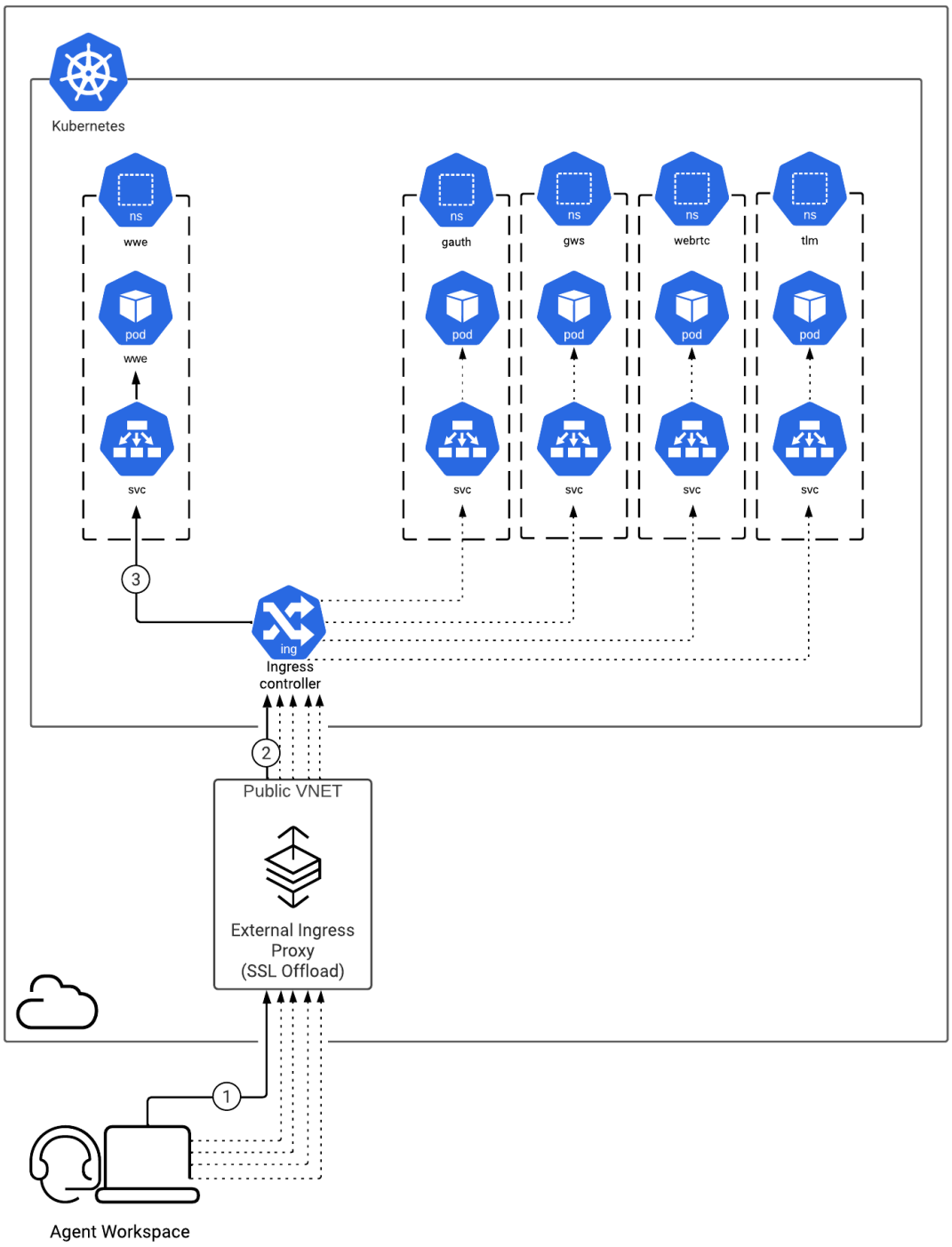
- Genesys Authentication Private Edition Guide
- Genesys Web Services and Applications Private Edition Guide
- WebRTC Private Edition Guide
- Telemetry Service Private Edition Guide

For information about the overall architecture of Genesys Multicloud CX private edition, see the high-level Architecture page.

See also High availability and disaster recovery for information about high availability/disaster recovery architecture.

## Architecture diagram — Connections

The numbers on the connection lines refer to the connection numbers in the table that follows the diagram. The direction of the arrows indicates where the connection is initiated (the source) and where an initiated connection connects to (the destination), from the point of view of Workspace Web Edition as a service in the network.


Kubernetes
ns
wwe
pod
wwe
svc
ns
gauth
pod
svc
ns
gws
pod
svc
ns
webrtc
pod
svc
ns
tlm
pod
svc
3
ing
Ingress
controller
2
Public VNET
External Ingress
Proxy
(SSL Offload)
1
Agent Workspace

## Connections table

The connection numbers refer to the numbers on the connection lines in the diagram. The **Source**, **Destination**, and **Connection Classification** columns in the table relate to the direction of the arrows in the Connections diagram above: The source is where the connection is initiated, and the destination is where an initiated connection connects to, from the point of view of Workspace Web Edition as a service in the network. *Egress* means the Workspace Web Edition service is the source, and *Ingress* means the Workspace Web Edition service is the destination. *Intra-cluster* means the connection is between services in the cluster.

| Connection | Source | Destination | Protocol | Port | Connection Classification | Data that travels on this connection |
|---|---|---|---|---|---|---|
| 1 | Browser | Inbound Gateway | HTTPS | 443 | Ingress | Inbound web traffic |
| 2 | Ingress proxy | Ingress controller | HTTPS | 443 | Intra-cluster | Inbound web traffic |
| 3 | Ingress controller | Workspace Web Edition | HTTPS | 8080 | Intra-cluster | Ingress controller connects to WWE pod |

# High availability and disaster recovery

Find out how this service provides disaster recovery in the event the service goes down.

**Related documentation:**

- 
- 

| Service | High Availability | Disaster Recovery | Where can you host this service? |
|---|---|---|---|
| Workspace Web Edition | N = N (N+1) | Active-spare | Primary or secondary unit |

*This information is under development: Flagged  items aren't yet confirmed or have info coming soon; Checked  items are valid.*

See High Availability information for all services: High availability and disaster recovery

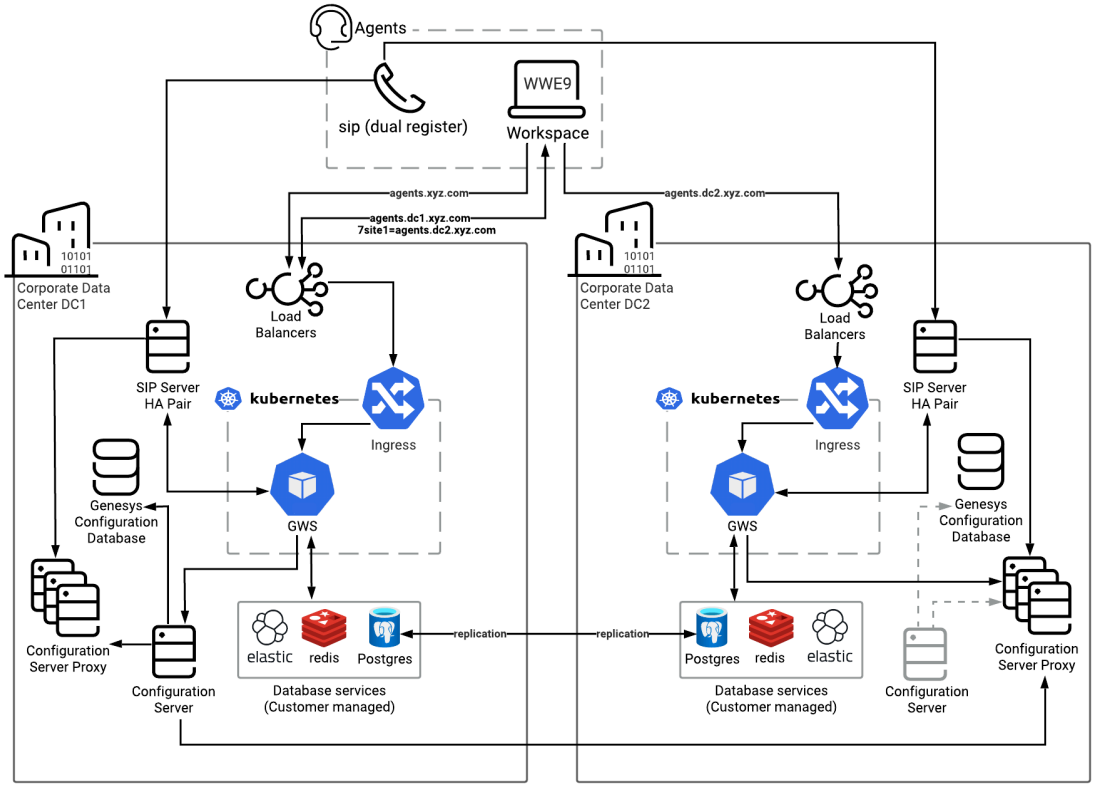
To support High Availability and Disaster Recovery, Workspace Web Edition should be deployed as part of Genesys Web Services in multiple regions.

To maintain business continuity without requiring user manual switching to the backup region, the Workspace Web Edition application (browser-side) automatically fails over to the backup region when it detects a loss of connectivity to the primary region. The application notifies the agent (end user) of the failover operation in progress and might prompt for re-authentication.

Refer to the Architecture topic for information about HLA architecture.

## Smart Failover for Workspace Users

Smart failover support for Workspace users is provided by Genesys Web Services and Applications (GWS).




- SIP phone dual-registers to both data centers.
- SIP HA handles SIP Server failover.
- Various DNs are provisioned in both SIP Servers, so that agents can operate in either SIP Server deployment.
- An agent's browser is set up with a link to the Workspace application. For Smart Failover, DNS resolution is used to return the URL to the local data center's GWS/Workspace app with the address of the backup site specified as the URL parameter.
- CometD connection (through HTTPS) is maintained with the Workspace client.

## Workspace Agent Site Selection

Workspace uses DNS to select and distribute Workspace agents across sites and load balance between nodes.

- The browser uses DNS to resolve the common URL address to the local site. The link to the backup site should also be included in the name resolution:
  `https://agents.xyz.com`
  `https://agents.dc1.xyz.com?site1=agents.dc2.xyz.com`
- The browser fetches the Workspace application from the data center specified in the DNS (DC1).
- Session stickiness persists the link from the browser app to the GWS instance handling the agent's desktop.

If there is an issue with the current site, Workspace senses this (due to the loss of the CometD connection) and reconnects to the DR site URL specified in the parameter (example, `?site1=agents.dc2.xyz.com`).

DNS configuration is an important element of this solution and depends on your load balancer/proxy. It needs to support name resolution to the local site, addition of the siteX URL parameter, and sticky sessions.

## Configuration Server

Configuration Server Proxies are deployed in both data centers to handle requests for configuration information. One data center has a live Configuration Server for making updates. A cold standby pair exists within the other data center.

During a data center failure, the Configuration Server Proxies in the remaining data center handle requests. Configuration updates are not possible. If needed, the cold standby Configuration Server can be turned on in the remaining data center, assuming that configuration database replication is enabled between sites. If the cold standby Configuration Server is turned on, the Configuration Server Proxies need to re-establish a connection with the new operational Configuration Server. Expect performance impacts from switching over.

## Failover Scenarios

### Voice Channel

If the SIP phone loses connection to SIP Servers in one data center, it re-connects to SIP Servers in the other data center, based on SIP/dual registration protocol. Existing voice calls might be lost.

GWS monitors the connection to the SIP Server session. The connection drops when the SIP Server goes down. GWS notifies an application in the agent's browser about the Voice channel unavailability with the `ServiceStateChanged UNAVAILABLE` event.

### Workspace

Workspace triggers a disaster recovery failover to another site based on the following triggers:

- DN unregistration (DNStateChanged Inactive event received)
- Voice channel unavailability (ServiceStateChanged UNAVAILABLE event received)
- CometD real-time channel loss
- Workspace initialization issues at login
- Registration failure for Softphone
- Registration failure for WebRTC

The Workspace application needs to authenticate in order to log in to the backup site. The single sign-on (SSO) integration automatically logs in to the backup site without requiring the agent to enter the password.

## Data Center

If the entire data center fails, both the voice channel and the Workspace application fail. This impacts both CometD connections to the Workspace client, forcing it to reconnect to the backup site. The SIP endpoint also switches over to the other site.

# Ports

## Contents

**Related documentation:**

- 
- 

## Ports and protocols for Workspace Web Edition

| Included Service | Protocol | Port | Type of data | Comment |
|---|---|---|---|---|
| Workspace Web Edition | HTTP | 80 | Web static content | Kubernetes service port (default) |
| Workspace Web Edition | HTTP | 8080 | Web static content | Container port (default) |

# Before you begin

## Contents

Find out what to do before deploying Workspace Web Edition.

**Related documentation:**

- 
- 

## Limitations and assumptions

There are no limitations or assumptions related to the deployment.

## Download the Helm charts

The Workspace Web Edition Helm charts are included in the Genesys Web Services (GWS) Helm charts. You can access them when you download the GWS Helm charts from JFrog using your credentials.

See Helm charts and containers for Genesys Web Services and Applications for the Helm chart version you must download for your release.

For information about downloading Genesys Helm charts from JFrog Edge, refer to this article: Downloading your Genesys Multicloud CX containers.

## Third-party prerequisites

Not applicable

## Storage requirements

There are no specific storage requirements for Workspace Web Edition.

## Network requirements

Network requirements include:

- Required properties for ingress:

    - Cookies usage: None
    - Header requirements - client IP & redirect,  passthrough: None
    - Session stickiness: None
    - Allowlisting - optional: None
    - TLS for ingress - optional (you can enable or disable TLS on the connection): Though annotation like any UI or API in the solution
- Cross-region bandwidth: N/A
- External connections from the Kubernetes cluster to other systems: N/A
- WAF Rules (specific only for services handling internet traffic): N/A
- Pod Security Policy: N/A
- High-Availability/Disaster Recovery: Refer to High availability and disaster recovery
- TLS/SSL Certificate configurations: No specific requirements

## Browser requirements

You can use any of the supported browsers to run Workspace Agent Desktop on the client side.

## Genesys dependencies

### Mandatory Dependencies

The following services must be deployed and running before deploying the WWE service. For more information, refer to Order of services deployment.

- Genesys Authentication Service:
    - A redirect must be configured in Auth/Environment to allow an agent to login from the WWE URL. The redirect should be configured in the Auth onboarding script, according to the DNS assigned to the WWE service.
- GWS services:
    - The CORS rules for WWE URLs must be configured in GWS. This should be configured in the GWS onboarding script, according to the DNS assigned to the WWE service.
    - The GWS API URL should be specified at the WWE deployment time as part of the Helm values.
- TLM service:
    - The CORS rules for the domain where WWE is declared must be configured in Telemetry Service (TLM). For example: genesysengage.com

### Optional Dependencies

Depending on the deployed architecture, the following services must be deployed and running before deploying the WWE service:

- WebRTC Service: To allow WebRTC in the browser
- Telemetry Service: To allow browser observability (metrics and logs)

### Miscellaneous desktop-side optional dependencies

The following software must or might be deployed on agent workstations to allow agents to leverage the WWE service:

- **Mandatory**: A browser referenced in the supported browser list.
- **Optional**: Genesys Softphone: a SIP or WebRTC softphone to handle the voice channel of agents.

## GDPR support

Workspace Web Edition does not have specific GDPR support.

# Configure Workspace Web Edition

## Contents

Learn how to configure Workspace Web Edition.

**Related documentation:**

- 
- 

## Override Helm chart values

You can override values in the Helm charts to configure Workspace Web Edition (WWE) by using the `--set` flag or by creating the values.yaml file. For more information about how to override Helm chart values, see Overriding Helm chart values.

If you want to use arbitrary UIDs in your OpenShift deployment, you must override the **securityContext** settings in the **values.yaml** file, so that no user or group IDs are specified. For details, see the Configure security section, below.

Refer to the **Parameters** table for a full list of overridable values for WWE.

Parameters

| Parameter | Description | Default | Valid Values |
|---|---|---|---|
| wwe.image.registry | docker registry | pureengage-docker-staging.jfrog.io | A valid registry |
| wwe.image.repository | docker repository | gws | Valid repository in the registry |
| wwe.image.tag | WWE image version | 9.0.000.93 | Valid version of the WWE image |
| wwe.image.pullPolicy | Image pull policy | IfNotPresent | Kubernetes pullpolicy: IfNotPresent/Always |
| wwe.image.imagePullSecrets | Image pull secrets | [] | Array of pull secrets to use to pull from registry |
| wwe.service.enabled | Create k8s service | true | Boolean |
| wwe.service.type | k8s service type | ClusterIP | Kubernetes Service Type value |
| wwe.service.port | k8s service port | 80 | Valid port number |
| wwe.ingress.enabled | enabling ingress | false | Boolean |
| wwe.ingress.annotations | ingress annotations | {} | Key/value map of annotations |
| wwe.ingress.hosts | host configurations | {} | Valid host kubernetes Ingress mapping |
| wwe.ingress.tls | TLS based security enabling | {} | Valid tls kubernetes Ingress mapping |

| Parameter | Description | Default | Valid Values |
|---|---|---|---|
| wwe.resources.limits.cpu | Maximum amount of CPU K8s allocates for container | "4" | Number of milicpu. See k8s doc |
| wwe.resources.limits.memory | Maximum amount of Memory K8s allocates for container | 6Gi | Bytes allocation |
| wwe.resources.requests.cpu | Guaranteed CPU allocation for container | 500m | Number |
| wwe.resources.requests.memory | Guaranteed Memory allocation for container | 4Gi | Bytes allocation |
| wwe.deployment.strategy | k8s deployment strategy | {} | Kubernetes deployment strategy map |
| wwe.livenessProbe.httpGet.path | liveness prob path | /index.html | Valid internal pass |
| wwe.livenessProbe.httpGet.port | liveness prob port to use | http | http |
| wwe.livenessProbe.initialDelaySeconds | liveness prob startup delay | 10 | Number |
| wwe.livenessProbe.periodSeconds | liveness prob check interval | 5 | Number |
| wwe.livenessProbe.failureThreshold | liveness prob failure count | 3 | Number |
| wwe.livenessProbe.timeoutSeconds | liveness prob timeout seconds | 5 | Number |
| wwe.readinessProbe.httpGet.path | readiness prob path | /index.html | Valid internal pass |
| wwe.readinessProbe.httpGet.port | readiness prob port to use | http | http |
| wwe.readinessProbe.initialDelaySeconds | readiness prob startup delay in seconds | 10 | Number |
| wwe.readinessProbe.periodSeconds | readiness prob check interval in seconds | 5 | Number |
| wwe.readinessProbe.failureThreshold | readiness prob failure count | 3 | Number |
| wwe.readinessProbe.timeoutSeconds | readiness prob timeout seconds | 5 | Number |
| wwe.priorityClassName | k8s priority class name | '' | Valid priority class |
| wwe.affinity | pod affinity | {} | Kubernetes Pod affinity map |
| wwe.nodeselector | k8s node selector map | {} | Kubernetes node selector map |
| wwe.tolerations | pod toleration | [] | Kubernetes pod tolerations list |
| wwe.autoscaling.enabled | activate auto scaling | true | Boolean |
| wwe.autoscaling.targetCPUUtilizationPercentage | CPU percentage autoscaling trigger | 40 | Number |

| Parameter | Description | Default | Valid Values |
|---|---|---|---|
| wwe.autoscaling.targetMemoryUtilizationPercentage | Memory perceptage autoscaling trigger | 80 | Number |
| wwe.context.envs.optimizedConfig | Activate WWE optimized config | false | Boolean |
| wwe.context.envs.gwsUrl | Url of GWS API | `` | Valid GWS API url |
| grafanaDashboard.enabled | Deploy the grafana Dashboard | false | Boolean |
| securityContext | Pod security context | {runAsNonRoot:true,runAsUser:500,runAsGroup:500,fsGroup:500} | A Security Context map |

## Create the values.yaml file

From the following sample file, create the **values.yaml** file with appropriate overrides for a sample deployment.

**Note**: **ingress** should be enabled and set with an appropriate hostname and the value for **gwsUrl** must be set with the external GWS URL:

```
context:
  envs:
    optimizedConfig: false
    gwsUrl: 'https://'
```

For example:

```
namespace: wwe
nameOverride: ""
fullnameOverride: ""

securityContext:
  runAsNonRoot: true
  runAsUser: 500
  runAsGroup: 500
  fsGroup: 500

podLabels: {}
podAnnotations: {}

wwe:
  image:
    registry:
    repository: gws
    name: gws-ui-workspace
    tag:
    pullPolicy: IfNotPresent
    imagePullSecrets: []
  service:
    enabled: true
    type: ClusterIP
    port: 80
  ingress:
    enabled: true
```

```
  hosts:
    # Example
     - host: wwe.apps.vce-c0.eps.genesys.com
       paths:
         - path: '/'
           port: 443
  annotations: {}
  #   Example
  #   cert-manager.io/cluster-issuer: letsencrypt-prod-nginx
  #   nginx.ingress.kubernetes.io/ssl-redirect: "false"
  #   kubernetes.io/ingress.class: nginx01-internal
  #   nginx.ingress.kubernetes.io/proxy-body-size: "0"
  tls:
    # Example
     - secretName: ""
       hosts:
         - wwe.apps.vce-c0.eps.genesys.com
serviceName: wwe
deployment:
  type: Deployment
  replicaCount: 3
  minReplicas: 1
  maxReplicas: 10
  strategy: {}
annotations: {}
livenessProbe:
  httpGet:
    path: /index.html
    port: http
  initialDelaySeconds: 10
  periodSeconds: 5
  failureThreshold: 3
  timeoutSeconds: 5
readinessProbe:
  httpGet:
    path: /index.html
    port: http
  initialDelaySeconds: 10
  periodSeconds: 5
  failureThreshold: 3
  timeoutSeconds: 5
context:
  envs:
    optimizedConfig: false
    gwsUrl: 'https://'
resources:
  requests:
    cpu: 500m
    memory: 2Gi
  limits:
    cpu: "1"
    memory: 6Gi
priorityClassName:
affinity: {}
nodeSelector:
  genesysengage.com/nodepool:
tolerations: []
labels: {}
autoscaling:
  enabled: true
  targetCPUUtilizationPercentage: 40
  targetMemoryUtilizationPercentage: 80
```

## Configure Kubernetes

There is no specific Kubernetes configuration required for Workspace Web Edition.

## Configure security

The security context settings define the privilege and access control settings for pods and containers.

By default, the user and group IDs are set in the **values.yaml** file as `500:500:500`, meaning the **genesys** user.

```
securityContext:
  runAsNonRoot: true
  runAsUser: 500
  runAsGroup: 500
  fsGroup: 500
```

### Arbitrary UIDs in OpenShift

If you want to use arbitrary UIDs in your OpenShift deployment, you must override the **securityContext** settings in the **values.yaml** file, so that you do not define any specific IDs.

```
podSecurityContext:
  runAsNonRoot: true
  runAsUser: null
  runAsGroup: 0
  fsGroup: null

securityContext:
  runAsNonRoot: true
  runAsUser: null
  runAsGroup: 0
```

# Provision Workspace Web Edition

## Contents

- Administrator

Learn how to provision Workspace Web Edition.

**Related documentation:**

- 
- 

## Workspace Web Edition configuration in Agent Setup

Use Agent Setup to configure your contact center, provision agents, and set up Workspace Web Edition.

## Tenant provisioning

All Workspace Web Edition functionality are provisioned using Agent Setup, including the creation of Agents.

# Deploy Workspace Web Edition

## Contents

Learn how to deploy Workspace Web Edition (WWE) into a private edition environment.

**Related documentation:**

- 
- 

## Assumptions

- The instructions on this page assume you are deploying the service in a service-specific namespace or OpenShift project, named in accordance with the requirements on Creating namespaces. If you are using a single namespace for all private edition services, replace the namespace element in the commands on this page with the name of your single namespace or project.
- Similarly, the configuration and environment setup instructions assume you need to create namespace-specific (in other words, service-specific) secrets. If you are using a single namespace for all private edition services, you might not need to create separate secrets for each service, depending on your credentials management requirements. However, if you do create service-specific secrets in a single namespace, be sure to avoid naming conflicts.

> **Important**
> Make sure to review Before you begin for the full list of prerequisites required to deploy Workspace Web Edition.

## Prerequisites for GKE

### Secret configuration for pulling image

### Log in to the cluster

Use the following command to log in to the cluster from the deployment host:

```
kubectl login --token  --server
```

### Connect to the cluster using Cloud SDK

Use the following command to connect to the cluster from the deployment host:

```
gcloud container clusters get-credentials  --zone  --project
```

### Create the secret for accessing the jfrog registry

Use the following command to create the secret:

```
kubectl create secret docker-registry mycred
--docker-server=pureengage-docker-staging.jfrog.io
--docker-username=
--docker-password=
--docker-email= -n wwe
```

## Environment Preparation for GKE

### Download the Helm charts

1. Download the WWE Helm charts from following repository: https://pureengage.jfrog.io/ui/repos/tree/General/helm-staging%2Fwwe
2. Create the **override_values.yaml** with appropriate overrides from the following sample file for a sample deployment:

   ```
   context:
    envs:
    optimizedConfig: false
    gwsUrl: 'https://'
   ```

3. Enable and set Ingress with the appropriate hostname.
4. Set the value for **gwsUrl** applying the external gws url.

## WWE installation on GKE

### Log in to GKE cluster

Use the following command to connect to the GKE cluster using Cloud SDK from the deployment host:

```
gcloud container clusters get-credentials  --zone  --project
```

### Create Namespace for WWE

Use the following command to create a new namespace for WWE:

```
kubectl create namespace wwe
```

### Render the templates

To verify whether resources are getting created without issue, execute the following command to render templates without installing:

```
helm template --debug wwe ./wwe-nginx-9.0.5.tgz -f override_values.yaml -n wwe
```

Review the displayed Kubernetes descriptors. The values are generated from Helm templates and are based on settings from the **values.yaml** and **values-test.yaml** files. Ensure that no errors are displayed. Later, you will apply this configuration to your Kubernetes cluster.

### Deploy WWE

Use the following command to deploy WWE:

```
helm install wwe ./wwe-nginx-9.0.5.tgz -f override_values.yaml -n wwe
```

This process takes several minutes. Wait until all objects are created and allocated, and the Kubernetes descriptors applied to the environment appear.

### Verify the installation

Use the following command to check the installed Helm release:

```
helm list –all-namespaces
```

Use the following command to check the WWE objects created by Helm:

```
kubectl get all -n wwe
```

Verify that you can now access WWE at the following URL:

http://wwe.

## Provisioning WWE Ingress on GKE

### Create or download the wwe-ingress.yaml file

Use the following example template to create the **wwe-ingress.yaml** Ingress file for WWE. In this example template, the namespace is set specifically to **wwe**. Adjust the values needed for your deployment.

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: wwe-ingress
  namespace: wwe
  annotations:
    # add an annotation indicating the issuer to use.
    cert-manager.io/cluster-issuer: "selfsigned-cluster-issuer"
    # Custom annotations for NGINX Ingress Controller
    kubernetes.io/ingress.class: "nginx"
    nginx.ingress.kubernetes.io/ssl-redirect: "false"
    nginx.ingress.kubernetes.io/use-regex: "true"
spec:
  rules:
  - host: wwe.test.dev
```

```
    http:
      paths:
        - path: /.*
          backend:
            serviceName:  wwe-wwe-nginx
            servicePort: 80
  tls:
  - hosts:
    - wwe.test.dev
    secretName: wwe-ingress-cert
```

### Apply the yaml file to your namespace

Use the following command to apply the yaml file to your namespace:

```
kubectl apply -f wwe-ingress.yaml -n wwe
```

## Deploy in Kubernetes on OpenShift

### Log in to the cluster

Use the following command to log in to the cluster from the deployment host :

```
kubectl login --token  --server
```

### Select your WWE Project

Use the following command to select the default WWE project that was created as a prerequisite:

```
kubectl project wwe
```

### Render the templates

To ensure that resources are created correctly, you can render the templates for debugging purposes without installing them. Use the following command to render the templates:

```
helm template --debug -f values.yaml wwe-helm wwe-nginx/
```

Kubernetes descriptors are displayed. The values are generated from Helm templates, based on settings from **values.yaml** and **values-test.yaml**. Ensure that no errors are displayed. This configuration is applied to your Kubernetes cluster.

### Deploy WWE

Use the following command to deploy WWE:

```
helm install --debug --namespace wwe --create-namespace -f values.yaml wwe-helm wwe-nginx
```

This process takes several minutes. Wait until all objects are created and allocated, and the Kubernetes descriptors applied to the environment appear.

### Check the deployment

Use the following command to check the installed Helm release:

```
helm list –all-namespaces
```

Use the following command to check the status of the WWE project:

```
kubectl status
```

Use the following command to check the WWE objects created by Helm:

```
kubectl get all -n wwe
```

### Expose the WWE service

Make WWE accessible from outside the cluster, using the standard HTTP port. Use the following command to expose the WWE service on OpenShift:

```
oc create route edge --service=wwe-helm-wwe-nginx --hostname=
```

## Validate the deployment on OpenShift

Use the following command to verify that the new route is created in the WWE project on OpenShift:

```
oc get route -n wwe
```

The result should show details similar to the following:

```
NAME   HOST/PORT                          PATH   SERVICES             PORT   TERMINATION
WILDCARD
wwe    wwe.apps.vce-c0.eps.genesys.com          wwe-helm-wwe-nginx   http   edge/Allow    None
```

where is the host name generated by Kubernetes.

Verify that you can access WWE at the following URL:

http://wwe.

# Upgrade, rollback, or uninstall Workspace Web Edition

## Contents

Learn how to upgrade, rollback or uninstall Workspace Web Edition

**Related documentation:**

- 
- 

## Upgrade Workspace Web Edition

To upgrade Workspace Web Edition, edit the **wwe.image.tag** option in the Helm Value file, then redeploy the chart according to standard rules.

## Rollback Workspace Web Edition

To rollback Workspace Web Edition, edit the **wwe.image.tag** option in the Helm Value file, then redeploy the chart according to standard rules.

## Uninstall Workspace Web Edition

To uninstall Workspace Web Edition, use the standard procedure for any service.

# Observability in Workspace Web Edition

## Contents

Learn about the logs, metrics, and alerts you should monitor for Workspace Web Edition.

**Related documentation:**

- 
- 

# Monitoring

Private edition services expose metrics that can be scraped by Prometheus, to support monitoring operations and alerting.

- As described on Monitoring overview and approach, you can use a tool like Grafana to create dashboards that query the Prometheus metrics to visualize operational status.
- As described on Customizing Alertmanager configuration, you can configure Alertmanager to send notifications to notification providers such as PagerDuty, to notify you when an alert is triggered because a metric has exceeded a defined threshold.

The services expose a number of Genesys-defined and third-party metrics. The metrics that are defined in third-party software used by private edition services are available for you to use as long as the third-party provider still supports them. For descriptions of available Workspace Web Edition metrics, see:

- Workspace Web Edition metrics

See also System metrics.

The WWE container does not expose any Prometheus endpoints. The container is based on an NGINX HTTP server structured to serve a set of static files that comprise the WWE Web Application.

## Enable monitoring

Monitoring of the WWE container is available through the standard monitoring capabilities of a Container running in a Kubernetes Pod through cAdvisor.

| Service | CRD or annotations? | Port | Endpoint/ Selector | Metrics update interval |
|---|---|---|---|---|
| Workspace Web Edition | n/a | n/a | n/a | n/a |

## Configure metrics

The metrics that are exposed by the WWE service are available by default. No further configuration is required to define or expose these metrics. You cannot define your own custom metrics.

The Metrics page linked to above shows some of the metrics the WWE service exposes. You can also query Prometheus directly or via a dashboard to see all the metrics available from the WWE service.

### What do Workspace Web Edition metrics monitor?

All the metrics documented on the Metrics page are standard Kubernetes metrics as delivered by cAdvisor. These can apply to any similar pod. The WWE container does not expose any Prometheus endpoints that would expose service-specific metrics.

## Alerting

No alerts are defined for Workspace Web Edition.

## Logging

Refer to Logging.

# Workspace Web Edition metrics and alerts

Find the metrics WWE exposes and the alerts defined for WWE.

**Related documentation:**

- 

## Contents

| Service | CRD or annotations? | Port | Endpoint/Selector | Metrics update interval |
|---|---|---|---|---|
| Workspace Web Edition | n/a | n/a | n/a | n/a |
| | All the WWE Service metrics are standard Kubernetes metrics as delivered by a standard Kubernetes metrics service. | | | |

## Metrics

All the WWE Service metrics are standard Kubernetes metrics as delivered by a standard Kubernetes metrics service. These can apply to any similar pod. The WWE container does not expose any Prometheus endpoint that would expose service-defined metrics.

You can query Prometheus directly to see all the available system metrics exposed by WWE. The following metrics are likely to be particularly useful:

- container_cpu_usage_seconds_total
- container_fs_reads_bytes_total
- container_memory_working_set_bytes
- container_network_receive_bytes_total
- container_network_transmit_bytes_total
- kube_pod_container_status_ready
- kube_pod_container_status_restarts_total

For information about standard system metrics to use to monitor services, see System metrics.

For information about standard metrics to monitor Genesys Web services, see [[GWS/Current/GWSPEGuide/GWSMetrics#Metrics|]].

## Alerts

No alerts are defined for Workspace Web Edition.

# Logging

Learn how to store logs for Workspace Web Edition.

**Related documentation:**

- 
- 

The Workspace Web Edition service is based on NGINX instances which are configured to log to **stdout**.