



The Retail Innovators

SAP Dynamic Pricing by GK

Operation Guide

Version: v4.0.0



COPYRIGHT

© 2022 SAP SE or an SAP affiliate company. All rights reserved. No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company.

1. You may not use the SAP Material for a purpose competitive with SAP or its products unless otherwise clearly permitted by applicable law.
2. You may not use the SAP corporate logo.
3. No use of other SAP trademarks is granted under this section. For information regarding use of SAP trademarks, see <http://www.sap.com/corporate/en/legal/trademark.html>.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Internal Document Information: 1223593622 | 2022-03-02

TABLE OF CONTENTS

1	Introduction	5
2	Support desk Management & Technical Support	5
2.1	Help menu	5
3	Tenant Management	6
3.1	Delete a tenant	6
3.2	Move a tenant	6
4	Starting and stopping the application	6
5	Recalculation of Data	6
5.1	Recalculation of Segment Statistics.....	6
6	Using SSH Keys	7
6.1	Creating a Key	7
6.2	Providing a Key using a Secret.....	7
6.2.1	Docker-compose.....	7
6.2.2	Kubernetes	8
7	Logging and Tracing	9
8	Backup and Recovery	9
8.1	Backup	9
8.2	Recovery	9
8.3	Backup and Restore for PostgreSQL	9
9	Housekeeping	10
10	Monitoring	11
10.1	Business Process Monitoring & Alerting.....	11
10.2	Health check of service	11
11	Filebeat	11
11.1	Configure Filebeat.....	11
11.2	Filebeat integration	12
11.2.1	Docker-compose.....	12
11.2.2	Kubernetes	12
12	Job scheduler and database connection configuration	14
13	Cloud Environment Configuration	15
13.1	Tenant Deployment: One or Multiple Tennants	20
13.1.1	Environment Configuration	20
13.1.1.1	One Tennant	20

13.1.1.2	Multiple Tennants.....	20
13.1.2	Plugin Deployment.....	21
14	Connecting to On Premise Backend	22
14.1	On Premise Backend Connection: Azure.....	22
14.1.1	Technical Architecture	22
15	Trouble Shooting	22
15.1	Database	22
15.1.1	Unclosed database connections on database server side	22
16	Error Codes	23
16.1	Errors - AIR.....	23

1 Introduction

This document describes operational topics for the system administrator to operate the application in a cloud environment:

- Support desk Management & Technical Support
- Tenant management
- Starting and stopping the application
- Logging and tracing
- Backup and recovery
- Housekeeping
- Monitoring
- Error Codes and recommended actions
- etc.

It is also possible to operate the application in on-premise mode. Please contact your partner about this option.

2 Support desk Management & Technical Support

Important information if you contact the SAP support:

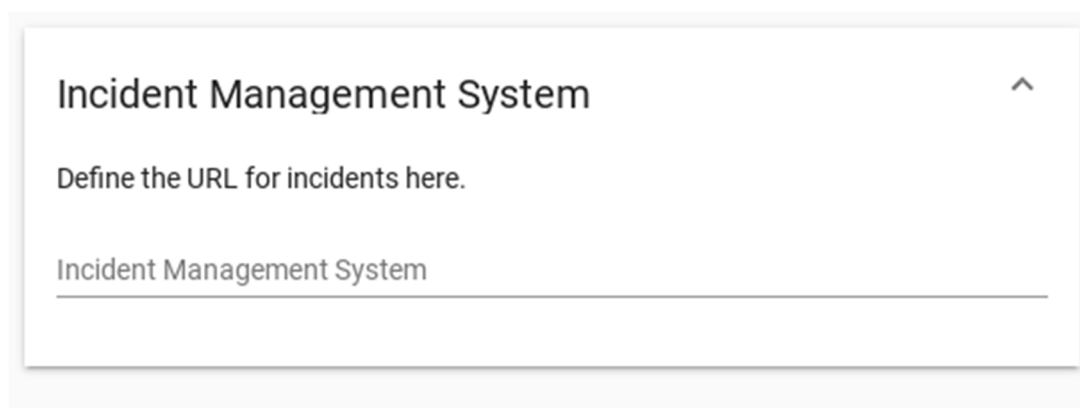
Application Name	SAP Dynamic Pricing by GK
Component Name	XX-PART-GKS-DPG

Incident Management

You can configure a link to a help desk system in "Incident Management System".

If configured, the help menu contains a new entry "Report Incident" with the link to the help desk system.

In this dialog the URL to the help desk system can be configured.



Incident Management System

Define the URL for incidents here.

Incident Management System

2.1 Help menu

Open the help menu to click on the link to the help desk system.



3 Tenant Management

3.1 Delete a tenant

If a tenant should be removed for any reason, follow these two steps:

- Remove the tenant via web client
- Remove the linked database from database server

3.2 Move a tenant

A tenant can be moved to another system (other cloud/on-premise hoster, etc.):

- Dump database of source tenant and restore it as new target database anywhere else (see description in chapter "Backup and Recovery").
- Create a new tenant on the new server.
- Make sure that the software version of the application on the target machine matches the one from the source machine.
- Link the new tenant with the restored database.
- Delete tenant from source system (see description above).

4 Starting and stopping the application

Starting and stopping the application means pushing (deploy + start) and deleting (stop and undeploy) the running application container.

5 Recalculation of Data

5.1 Recalculation of Segment Statistics

To recalculate the segment statistics for all segments in a business unit for a time frame, you can use this REST call:

```
GET
http://{HOST}:{PORT}/air/admin/res/{TENANT_ID}/plugins/gui/prudsys/prudsys/pricing/segmentstatisticservices/{BUSINESS_UNIT_ID}/recalculate?start={START_DATE}&end={END_DATE}
```

Date format for start and end is: YYYY-MM-DD e.g.

- The user needs the permission for recalculation of statistics.

6 Using SSH Keys

If you want to use the key based authentication for your sftp connection you will need to provide the private key on the system on which the pricing service is running.

6.1 Creating a Key

If you plan to use an ssh key you first have to create a ssh key in the **PEM format**. You can achieve this by executing the following command.

```
ssh-keygen -t rsa -m PEM
```

This will create a private and a public key. You need to copy the public key which ends with .pub to the server (e.g. SFTP) you want to connect to. The private key needs to be added to the pricing container. Learn more about this in the next chapter.

6.2 Providing a Key using a Secret

6.2.1 Docker-compose

When you're using docker-compose to create your pricing service you can add the directly in the docker-compose.yml. **To add secrets in a docker-compose.yml file the Version has to be at least 3.1.** The following example shows how to add a secret to the docker compose file:

```
version: '3.1'
services:
  pricing:
    image: prudsys/pricing:2.1.0
    container_name: pricing
    ports:
      - "8080:8080"
    environment:
      - MEM_META_KB=2000000
      - MEM_TOTAL_KB=5000000
    secrets:
      - source: sftpsecret
        target: sftpsecrettarget
    volumes:
      - /tmp/pricing:/pricing
secrets:
  sftpsecret:
    file: ./path/to/key/id_rsa
```

Code Block 1 docker-compose.yml

- About pointing to the file: ./ means relative to the docker context.
- When adding secrets to a service make sure that the secrets are listed before the volumes. Otherwise there can occur problems when mounting the secret into the docker. The target will always be mounted into the directory /run/secrets/.

You can also set the uid, gid and mode of the secret.

When your using docker with the version 1.13.1 the default mode is 0000. In newer versions the default mode is 0444. The default value for uid and gid is 0.

```

services:
  pricing:
    ...
    secrets:
      - source: sftpsecret
        target: sftpsecrettarget
        uid: '103'
        gid: '103'
        mode: 0440

```

Code Block 2 secrets.yml

6.2.2 Kubernetes

When using kubernetes you can create a secret by using the following command.

```
kubectl create secret generic key-secret --from-file=./key
```

Code Block 3 kubernetes create secret

You can also create a .yaml file which contains the secret.

```

apiVersion: v1
kind: Secret
metadata:
  name: mysecret
type: Opaque
stringData:
  username: |
    -----BEGIN RSA PRIVATE KEY-----
    ...
    -----END RSA PRIVATE KEY-----

```

Code Block 4 secret.yaml

If you want you can also encode your key with base64.

```

apiVersion: v1
kind: Secret
metadata:
  name: key-secret
type: Opaque
data:
  key: {{ base64 encoded key }}

```

Code Block 5 secret.yaml base64

When you have a secret file you can add it with kubectl apply. Now you're able to mount this secret.

```

spec:
  containers:
    - name: containerName
      volumeMounts:
        - name: keySecret
          mountPath: "/etc/keySecret"
          readOnly: true
      volumes:
        - name: keySecret
          secret:
            secretName: my-key-secret

```

Code Block 6 mount secret

If you need to reference the path to the secret in your configuration (e.g. SFTP XML configuration), **then your path is the mountPath.**

7 Logging and Tracing

In the cloud mode, the application logs all messages to stdout.

The log entries are written to the log stream as text log lines or as JSON structure.

In the log stdout stream are different types of log entries from different log targets. They are marked with timestamps, source, log level and type of log message. This log stream can be traced and monitored by other tools (e.g. ELK stack).

Example of a JSON log entry:

```
{
  "logtarget": "logs/serverlog/server.log",
  "timestamp": 1547640347082,
  "date": "2019-01-16 13:05:47",
  "hostname": "test-air",
  "level": "ERROR", "thread": "http-nio-0.0.0.0-8080-exec-2",
  "logger": "com.prudsys.security.login.failed",
  "message": "Login for user [admin] failed!"
}
```

8 Backup and Recovery

8.1 Backup

Because of the stateless behavior of the running application container there is no need to backup anything.

The whole application state (like configuration, master data and user management) is stored in the database.

The data of the database is consistent, because only committed transactions are finally persisted.

■ This is the data which has to be backed up regularly.

To backup the connected database, follow the backup strategies of the used database platform.

8.2 Recovery

Recovery is only necessary for the stored data of the database.

Follow these steps:

1. The first step of recovery is restoring the database to the state before the incident.
2. After that, start a new container instance and connect it to the recovered database.

8.3 Backup and Restore for PostgreSQL

The basic backup strategy in PostgreSQL is to fully dump the database into a backup file.

<https://www.postgresql.org/docs/9.6/backup-dump.html>

This backup file contains the whole database as SQL statements and could be used to restore the database after crashes.

The PostgreSQL database system supports also continuous archiving and Point-in-Time Recovery (PITR).

<https://www.postgresql.org/docs/9.6/continuous-archiving.html>

PostgreSQL could store a "write ahead log" (WAL) which allows a replay of the transactions since last full backup of the database.
 Because of this journal of database transactions, it is possible to restore the database to any point since last backup. This is called "point in time recovery".

9 Housekeeping

The application stores the following data:

- Data import:
 - Records
 - Data in original format
 - Protocol
- Data Export:
 - Protocol
- Optimized prices
- Statistics
- Configuration

The system performs regular cleanups of this data.

It is important to adjust the cleanup intervals to fit the project's needs. Otherwise the storage capacity will be exceeded after a while!

Type of Data	Description	Automatic Cleanup	Manual Cleanup
Optimized Prices	The price calculation results and execution protocols are stored in the database.	Regular background process to perform cleanup. Configuration of interval in business unit configuration.	Delete the data in the web client: Section Computation
Data Export	They create exported data in the export target, which could be the local file system of the execution host. Additionally, the task creates protocol data of the task execution.	Regular background process to perform cleanup. Configuration of interval in business unit configuration.	Delete the data in the web client: Section Computation
Data Import	They create the imported data in the database and protocol data of the task execution and the protocol.	Regular background process to perform cleanup. Configuration of interval in business unit configuration.	Delete the data in the web client: Section Settings - Data import
Statistics	Segment statistic calculations run in a regular manner which is configurable with the setting "Segment statistic interval" in every business unit. Besides the segment statistics, there are also simulation statistics. These statistics are created with every simulation run and are stored in database to be able to view old simulation statistics.	In every business unit there are three settings for statistic cleanups: Maximum age of segment, strategy and simulation statistics can be configured. The cleanup service runs once a day.	Not available
Master and transaction log data	Master data is historized. In case new master data is uploaded, the changed entries are not removed, they are created with a new version. Now, the new version entry is used as current value.	Regular background process to perform cleanup. Configuration of interval in business unit configuration for master and transaction log data.	
Configuration	Configuration is also stored and historized in the database. The historization does not track the timestamp, only the value and the user. <ul style="list-style-type: none"> • Server security management • Plugin security management • Plugin container configurations • Tenant configurations 	Automatically cleaned up.	

10 Monitoring

The following checks are available:

- Health check of service
- Statistics per tenant: Currently not processed tasks
- Statistics per tenant: Currently processed tasks of a node
- Statistics per tenant: Currently processed tasks of a node
- Statistics per tenant: Currently running importers
- Statistics per tenant: Currently running exporters
- Statistics per tenant: Currently running statistics
- Statistics per tenant: Currently running price tasks
- Statistics per tenant: Price computations per day, week, month, year

10.1 Business Process Monitoring & Alerting

The business process alerting is automatically enabled. In case of a failed import, export or pricing service run, this case is reported on the log stream. Thus it can be used to create an alert (e.g. Prometheus).

The output looks like this:

Process Type	Output if a process fails
Import Service	[ALERT][importservice] Import failed: {0} in {1}
Export Service	[ALERT][exportservice] Import failed: {0} in {1}
Pricing Service	[ALERT][pricingservice] Price calculation failed in {0}

10.2 Health check of service

With the following request the service could be checked if it is online and healthy. Healthy means, it is started without errors and can be used by consumers.

```
■ http://<HOST>:<PORT>/air/res/<TENANTID>/event/ping
```

Following return codes are possible:

- HTTP code 200 if the service is online
- HTTP code 503 if the service is offline,
- HTTP code 404 if the service is not available (unknown tenant id)

11 Filebeat

The Filebeat is used to send the logs of the pricing to an elk stack.

Therefor two environment variables must be set:

- OPERATION_MODE_CLOUD_LOG_FORMAT must be set to CONSOLE_JSON
- OPERATION_MODE_CLOUD_LOG_FORMAT_CONSOLE_JSON_LOGID

11.1 Configure Filebeat

You can configure the Filebeat in the filebeat-config.yml file.

```

filebeat.config:
  modules:
    path: ${path.config}/modules.d/*.yaml
    # Reload module configs as they change:
    reload.enabled: false

filebeat.inputs:
- type: docker
  containers.ids:
    - "*"
  processors:
    - add_docker_metadata: ~

output.logstash:
  hosts: [ 'logstash:port' ]
  enabled: true
  ssl.certificate_authorities: [ "/run/secrets/ca.crt" ]

```

Code Block 7 filebeat-config.yml

With the containers.ids setting you can configure a regex which defines of which docker containers the logs should be read.

In output.logstash you can define the logstash to which the logs will be send and the ssl.certificate_authorities for the ssl/tls connection to the logstash.

11.2 Filebeat integration

11.2.1 Docker-compose

Filebeat can be installed by just adding a new service to your docker-compose file.

```

version: '3.1'
services:
  filebeat:
    image: docker.elastic.co/beats/filebeat:7.10.0
    container_name: filebeat
    user: root:root
    command: -e -strict.perms=false
    secrets:
      - source: ca
        target: ca.crt
    volumes:
      - /var/lib/docker/containers:/var/lib/docker/containers:ro
      - /var/run/docker.sock:/var/docker/docker.sock:ro
      - /tmp/filebeat/data:/usr/share/filebeat/data
      - ./filebeat-config.yaml:/usr/share/filebeat/filebeat.yml:ro
    secrets:
      ca:
        file: ./local-ca.crt

```

Code Block 8 docker-compose.yml

The /var/lib/docker/containers directory will be mounted as read only. This mount is necessary so that the filebeat can read the logs of the pricing docker. There is also a mount for the data directory of the filebeat and the filebeat-config.yml which is used to configure the filebeat.

11.2.2 Kubernetes

When using kubernetes it's important to create a DaemonSet to ensure that there is an filebeat pod on every node.

```

apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: filebeat
  labels:
    k8s-app: filebeat
spec:
  selector:
    matchLabels:
      k8s-app: filebeat
  template:
    metadata:
      labels:
        k8s-app: filebeat

    spec:
      containers:
        - name: filebeat
          image: docker.elastic.co/beats/filebeat:7.10.0
          args: [
            "-c", "/etc/filebeat.yml",
            "-e",
          ]
          volumeMounts:
            - name: config
              mountPath: /usr/share/filebeat/filebeat.yml
              readOnly: true
            - name: data
              mountPath: /usr/share/filebeat/data
            - name: varlibdockercontainers
              mountPath: /var/lib/docker/containers
              readOnly: true
      volumes:
        - name: config
          configMap:
            defaultMode: 0600
            name: filebeat-config
        - name: varlibdockercontainers
          hostPath:
            path: /var/lib/docker/containers
        - name: data
          persistentVolumeClaim:
            claimName: filebeat-volumeclaim

---

apiVersion: v1
kind: ConfigMap
metadata:
  name: filebeat-config
  labels:
    k8s-app: filebeat
data:
  filebeat.yml: |-
    filebeat.config:
      filebeat.config:
        modules:
          path: ${path.config}/modules.d/*.yaml
          # Reload module configs as they change:
          reload.enabled: false

    filebeat.inputs:
      - type: docker
        containers.ids:
          - "*"
        processors:
          - add_docker_metadata: ~

    output.logstash:
      hosts: [ 'logstash:port' ]
      enabled: true
      ssl.certificate_authorities: [ "/run/secrets/ca.crt" ]

---

kind: PersistentVolumeClaim
apiVersion: v1

```

```

metadata:
  name: filebeat-volumeclaim
  labels:
    app: filebeat
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi

```

Code Block 9 filebeat.yml

12 Job scheduler and database connection configuration

The server platform provides a job scheduler framework, which is closely linked with the database connection settings of a tenant.

Type	Name	Unit	Environment parameter	Description
Scheduler	Job count	Number	✓	The job count sets the allowed maximum number of simultaneously runnable jobs.
Scheduler	Job timeout (ms)	ms	✓	Timeout until a abandoned(pending, started, running) job is eligible for rescheduling.
Scheduler	Poll interval	ms	✓	This interval controls the time intervall the job manager reevaluates the existence of new jobs,
Database	Connectionpool size	Number	✓	Maximum number of database connections available within the connection pool. ⚠ The more simultaneously jobs are running, the more connections of the pool are used at the same time. So be sure the pool size fits to the job concurrency.
Database	Validation query	String	✗	SQL query used to check if database connections are alive.
Database	Validation interval	ms	✗	Connection pool interval of checks if connections are alive.
Database	Validation timeout	ms	✗	Allowed time until a connection is closed if not responding.
Database	Receiving buffer size	Number	✗	Size of database receiving buffer in bytes.
Database	Sending buffer size	Number	✗	Size of database sending buffer in bytes.
Database	Transaction buffer size	Number	✗	Size of prepared statement buffer in database in megabytes per connection.

Sizing of Database Connections

Sizing of database Connections per Master Node:

- Base: 10 connections
- Per parallel user request: 2 connections
- Per configured worker slot: 10 connections

Sizing of database Connections per Worker Node:

- Base: 10 connections
- Per configured worker slot: 10 connections

Example: 1 master node, 3 parallel users, 4 slots

- Base: 10
- Users: 3 x 2
- Slots: 10 x 4
- = 56 connections → round up: **60 connections**

Example: 1 master node, 2 worker nodes, 3 parallel users, 4 slots each

- Base: 10 x 3 nodes
- Users: 3 x 2
- Slots: 10 x 4 x 3 nodes

- = 186 connections → round up: **200 connections**

In cloud mode the database connection must be set by environment parameters only!
Nevertheless, this connection is visible in the UI, but should not be changed because it is not persistent over the next container restart.


13 Cloud Environment Configuration

To start the server platform in cloud mode, you need to pass special environment variables.

Parameter	Description	Type	Example values
OPERATION_MODE	Runtime mode of the instance. If not set the server runs in on-premises mode by default.	Enumeration	<ul style="list-style-type: none"> CLOUD => Activate Cloud mode Anything else => On-premises mode
OPERATION_MODE_CLOUD_INSTANCE	Name of the tenant which exists in cloud instance at startup. It's possible to configure more than one tennant: OPERATION_MODE_CLOUD_INSTANCE=tenantID1 tenantID2 tenantID3	String	One tenant: demo Multiple tenants: italy germany france
OPERATION_MODE_CLOUD_INSTANCE_NAME_{tenantID}	Optional name for the instance(s) named in OPERATION_MODE_CLOUD_INSTANCE	String	Displayed name of the tenant
OPERATION_MODE_CLOUD_ADMIN_PASSWORD	Administration password of the default administrator which exists always.	String	123admin123
OPERATION_MODE_CLOUD_LOG_FORMAT	Logging format mode. There are 4 modes. <ul style="list-style-type: none"> CONSOLE_JSON: Structured logging with JSON for easy further logfile analytics with Kibana to stdout CONSOLE_PREMISE: Usual single line logging to stdout CONSOLE_CSV: CSV output to stdout ON_PREMISE: Classic file logging 	Enumeration	<ul style="list-style-type: none"> CONSOLE_JSON CONSOLE_PREMISE CONSOLE_CSV ON_PREMISE
OPERATION_MODE_CLOUD_LOG_FORMAT_CONSOLE_JSON_LOGID	Id to be passed through to the logged records (record.contextMap.log_id). This is a free-text-field. Json- or escaped Content should be avoided as there is additional escaping performed to conform to a json-value-format	String	pricing-123-production
OPERATION_MODE_CLOUD_PLUGIN_REPOSITORY	Folder which contains the plugin ZIP files used during container startup.	String	/opt/prudsys/plugins
OPERATION_MODE_CLOUD_USE_DB_PLATFORMSTORAGE	Store service and security management in database too. If false, those configurations are stored in file system.	Boolean	true, false
OPERATION_MODE_CLOUD_USE_DB_PLATFORMSTORAGE_POOL_SIZE	Optional Maximum size of JDBC connection pool, which holds the database connections available for the platform.	Number	10
OPERATION_MODE_CLOUD_STORAGE_DIR	Optional Configuration of an another storage directory instead of the internal plugin data directory. This allows usage of large additional storages in cloud scenarios, where the main application instance has size limits.	String	/mnt/storage/
OPERATION_MODE_CLOUD_WORKER	Optional Run this node as compute node only. This means it only works on already queued jobs in the database and does not execute scheduled tasks. Compute nodes should not be used as UI or REST API endpoint, because the master node will not see the changes in its caches.	Boolean	true, false
OPERATION_MODE_CLOUD_JOB_CONCURRENCY	Optional Set the maximal number of simultaneous executable tasks. This setting should take in account how many CPU cores the system has got.	Number	4

OPERATION_MODE_CLOUD_JOB_LIFE_SIGN_TIMEOUT	Optional Timeout until a abandoned(pending, started, running) job is eligible for rescheduling.	Number ms	120000
OPERATION_MODE_CLOUD_JOB_POLL_INTERVALL	Optional Time intervall for the job manager to evaluate the existence of new jobs,	Number ms	1000
OPERATION_MODE_CLOUD_POOL_SIZE	Optional Maximum size of JDBC connection pool, which holds the database connections available for the application. Every job which works with the database needs one connection from pool at least. That means, if more simultaneous jobs are allowed, a bigger connection pool is needed.	Number	10

MEMORY_CONFIGURATION_MODE	<p>Optional Select memory configuration mode:</p> <ul style="list-style-type: none"> • PERCENT (Default if not set): <ul style="list-style-type: none"> ○ Use "MEM_JAVA_PERCENT" of "MEM_TOTAL_KB" for java heap. ○ The remaining memory is available for operation system and other services. • ADAPTIVE: <ul style="list-style-type: none"> ○ Use a total of memory "MEM_TOTAL_KB" and split that into 3 parts. Java heap, Java meta space and a reserved part for operation system and other things. ○ Java heap is calculated with the following formula: Java heap = Total - Reserved - (Meta*1.5) ○ Meta gets additional 50% because of unpredictable sizing in Java memory management. 	Enumeration	<ul style="list-style-type: none"> • PERCENT <ul style="list-style-type: none"> ○ 80% of 5000000 KB (5GB) as Java heap • ADAPTIVE <ul style="list-style-type: none"> ○ Allocate 5000000 KB (5GB) <ul style="list-style-type: none"> ▪
---------------------------	--	-------------	---

MEM_JAVA_PERCENT	Used in mode: PERCENT Optional (Default: 70) Percentage of memory in container used for Java VM heap (Xmx and Xms).	Number	70
MEM_TOTAL_KB	Used in mode: PERCENT and ADAPTIVE Optional (Default: System memory size) Set the memory of the docker container in kb.	Number	1024000
MEM_META_KB	Used in mode: ADAPTIVE Optional (Default: 256000) Set the size of java meta space with additional buffer of +50% inside of total memory.	Number	256000
MEM_RESERVED_KB	Used in mode: ADAPTIVE Optional (Default: 150000) Set the size of reserved memory inside of total memory	Number	150000
OPERATION_MODE_CLOUD_AUTHENTICATION_ENABLED_BASIC_AUTH	Optional Allow HTTP basic authentication additionally to the default digest authentication.  Attention: Should be used in secured networks only!	Boolean	true, false
OPERATION_MODE_CLOUD_ALTERNATE_DB	Optional Use a dedicated database for platform storage.	String	db2
Database connection (e.g. for Azure)			
DBHOST	Database host where all the data is stored. For one or multiple tenants	String	mydb.com DBHOST_tenantID1=mydb.com DBHOST_tenantID2=mydb.com
DBUSER	Database user For one or multiple tenants	String	dbuser DBUSER_tenantID1=dbuser DBUSER_tenantID2=dbuser

DBPASS	Password of user For one or multiple tenants	String	mySecretPwd DBPASS_tenantID1=jkfadsjlfjksdf DBPASS_tenantID2=ierjleafldf
DBNAME	Name of the database to connect with. For one or multiple tenants	String	dbinstance1 DBNAME_tenantID1=dbinstance1 DBNAME_tenantID2=dbinstance2
DBPORT	Port of the database host For one or multiple tenants	Number	5432 DBPORT_tenantID1=5432 DBPORT_tenantID2=5432

13.1 Tenant Deployment: One or Multiple Tenants

13.1.1 Environment Configuration

13.1.1.1 One Tenant

1. Tenant ID
OPERATION_MODE_CLOUD_INSTANCE=tenant1

Default for the name is the tenant ID. Optionally to set a custom name for your tenant:
OPERATION_MODE_CLOUD_INSTANCE_NAME_tenant1=My Custom Name

2. AIR and Application Database settings. This database is used both for AIR and storing application data, separated by different DB schemes:

DBNAME=
DBHOST=
DBUSER=
DBPASS=
DBPORT=

If you want to use a separate DB for AIR storage, then set:
OPERATION_MODE_CLOUD_ALTERNATE_DB

3. Database Connections (see chapter "Job scheduler and database connection configuration"for details)
OPERATION_MODE_CLOUD_USE_DB_PLATFORMSTORAGE_POOL_SIZE=10
OPERATION_MODE_CLOUD_POOL_SIZE=20
4. Memory Settings
MEM_TOTAL_KBMEM_META_KB

13.1.1.2 Multiple Tenants

1. Tenant ID
OPERATION_MODE_CLOUD_INSTANCE=tenant1 tenant2

OPERATION_MODE_CLOUD_INSTANCE_NAME_tenant1=My custom name 1
OPERATION_MODE_CLOUD_INSTANCE_NAME_tenant2=My custom name 2

1. AIR Database settings:

OPERATION_MODE_CLOUD_ALTERNATE_DB=platformDB
DBHOST=
DBUSER=
DBPASS=
DBPORT=

2. Tenannt Database settings, if all tenants use the same database server:

```
DBNAME_tenant1=  
DBNAME_tenant2=
```

You might want to use separate database servers for each tenant, then configure:

```
DBHOST_tenant1=  
DBUSER_tenant1=  
DBPASS_tenant1=  
DBPORT_tenant1=  
DBHOST_tenant2=  
DBUSER_tenant2=  
DBPASS_tenant2=  
DBPORT_tenant2=
```

3. Database Connections (see chapter "Job scheduler and database connection configuration" for details)
OPERATION_MODE_CLOUD_USE_DB_PLATFORMSTORAGE_POOL_SIZE=20
OPERATION_MODE_CLOUD_POOL_SIZE=40

4. Memory Settings

You need to take into account the tenant count!

- a. Meta Space: 256000 KB * tenant count
MEM_META_KB=512000
- b. Total Memory: Depends on your concurrent slots and your data volume for each computation
MEM_TOTAL_KB=14000000

13.1.2 Plugin Deployment

There is a folder /opt/prudsys/plugins (see environment variables to change that) that contains the plugins for auto deployment.

The resolution order is:

1. Check for a tenant if there is a dedicated sub folder, e.g. for tenant1: /opt/prudsys/plugins/tenant1
2. If the folder exists, those plugins are deployed for this tenant. Even if the folder is empty.
3. If the folder DOES NOT exist, then fallback to the main folder: /opt/prudsys/plugins

■ There is no merging of plugins between the dedicated tenant folder and the fallback folder.

Compute Node Setup


Because of the limitations of productive cloud runtime containers it is necessary to scale over multiple nodes. (see Architecture Guide: Scaling Options)

Adding computation nodes to an existing system is quite easy!

Just connect the additional AIR runtime node to the same database as the already running nodes.

That's it! The nodes synchronize them self over the database, which provides a list of workable jobs. Every compute node takes jobs from this list and marks them as in progress, so the current in progress jobs would not be taken from another compute node.

You should see the current working nodes in Settings → Nodes and Tasks:

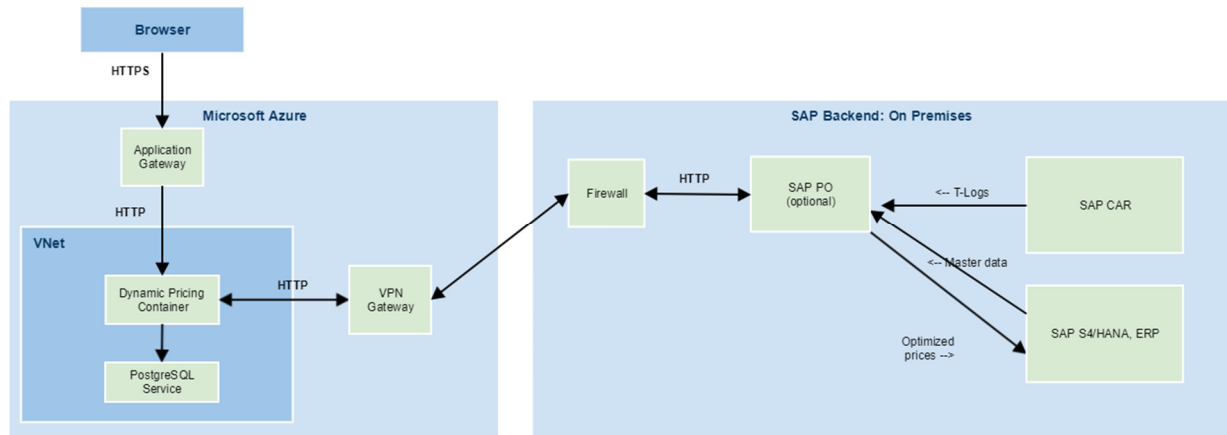


The screenshot shows a web interface with a sidebar on the left containing 'Overview', 'Nodes', and 'Pending Tasks'. The main area displays a table of 'Compute Node' entries. The first entry has ID '00000000000000000000' and status 'Ready'. The second entry has ID '00000000000000000000' and status 'Ready'. The third entry has ID '00000000000000000000' and status 'Ready'.

14 Connecting to On Premise Backend

14.1 On Premise Backend Connection: Azure

14.1.1 Technical Architecture



15 Trouble Shooting

15.1 Database

15.1.1 Unclosed database connections on database server side

Problem:

If the application is terminated without a correct shutdown, it is possible that database connections and locks are kept by the database server side. Those connections and locks must be closed and released by the database server.

Depending on the database default settings, this could last 2 hours or more.

Until the connections are not closed or the locks are not released, they block resources in the set of possible server connections and much worse they lock data inside the database!

Solution:

It is not possible to handle this from application side, because database server connections are not under our management.

But you can:

- Avoid hard application kills
- Set a shorter connection cleanup interval in the database
















16 Error Codes

16.1 Errors - AIR

Startup errors are critical errors preventing the application deployment.

There are errors and warnings, errors leading to a failed server startup.

Warnings do not prevent the server from starting, errors do.

Error Code	Error  , Warning 	Description	Recommended Actions
900001		The operation mode CLOUD requires configured credentials!	Check configuration of environment variables for database authentication
900002		HashFunction not available.	Wrong installation
900003		The operation mode CLOUD requires a configured instanceid!	Check setting of environment variable OPERATION_MODE_CLOUD_INSTANCE
900004		No PluginRepo configured.	Wrong installation
900005		Cannot deploy plugins and bundles at the same time!	Wrong installation
900006		Unable to read plugin: <PATH>	Wrong installation
900007		Dependency-Validation-Error for Plugin: <PLUGIN NAME>	Check that all necessary plugins are available in the environment variable OPERATION_MODE_CLOUD_PLUGIN_REPOSITORY
900008		Unable to calculate pluginId for: <PATH>	Wrong installation
900009		TargetDir for redeployment does not exist for: <PATH>	Wrong installation
900010		Processing of Plugins failed <EXCEPTION>	Wrong installation
900011		Configured PluginRepo is not accessible!	Check existence and permissions of path in the environment variable OPERATION_MODE_CLOUD_PLUGIN_REPOSITORY
900012		Autodeployment of Bundles is not supported	Wrong installation
900013		LogConfiguration failed.	Wrong configuration, see environment variables

CONTACT

GK Software SE
Waldstraße 7
08261 Schöneck
Germany

T +49 (0) 3 74 64 84 – 0

F +49 (0) 3 74 64 84 – 15

documentation@gk-software.com

www.gk-software.com