FLOWX.AI

PLATFORM DEEP DIVE / Plugins / Plugins setup guides

Contents

- PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Customer management plugin setup
  - Infrastructure Prerequisites:
    - Elastic Search
    - Postgres database
  - Configuration
    - Authorization configuration
    - Datasource configuration
    - Elastic search configuration
    - Kafka configuration
    - Logging
- PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Documents plugin setup / Configuring access rights for Documents
- PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Notification templates plugin setup / Configuring access rights for Notifications
- PLATFORM DEEP DIVE / Plugins / Plugins setup guides / OCR plugin setup
  - Infrastructure Prerequisites:
  - Deployment/Configuration
    - Credentials
    - Kafka configuration
    - Authorization
    - Storage (S3 configuration)
    - Performance
    - Certificates
    - Workers behavior
- PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Reporting setup guide

    - Dependencies
        - Postgres database
        - Reporting plugin helm chart (containing CRON)
        - Superset
    - After installation
        - Datasource configuration
        - Redis configuration
    - Keycloak configuration
        - Extend the Security Manager
        - Configure Superset
- PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Task Manager plugin setup / Configuring access rights for Task management

PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Customer management plugin setup

Infrastructure Prerequisites:

The Customer management plugin is available as a docker image so we need to configure:

Elastic Search

In order to install elasticsearch instance Elastic Cloud on Kubernetes (ECK) can be used.

Use ECK quickstart to deploy CRDs and create elasticsearch instances:

Elasticsearch instance:

```
  apiVersion: elasticsearch.k8s.elastic.co/v1
  kind: Elasticsearch
  metadata:
    name: elasticsearch-flowx
    namespace: elastic-system
  spec:
    version: 7.9.3
    updateStrategy:
      changeBudget:
        maxSurge: 3
        maxUnavailable: 1
    nodeSets:
    # 3 dedicated master nodes
    - name: master
      count: 3
      config:
        node.master: true
        node.data: false
        node.ingest: false
        node.remote_cluster_client: false
        # this allows ES to run on nodes even if their
vm.max_map_count has not been increased, at a performance
cost
        # node.store.allow_mmap: false
      podTemplate:
        spec:
          initContainers:
          - name: sysctl
            securityContext:
              privileged: true
            command: ['sh', '-c', 'sysctl -w
vm.max_map_count=262144']
```

```
          - name: install-plugins
            command:
            - sh
            - -c
            - |
              bin/elasticsearch-plugin install --batch
repository-gcs
          containers:
          - name: elasticsearch
            resources:
              limits:
                memory: 6Gi
                cpu: 2
              requests:
                memory: 2Gi
                cpu: 1
            env:
            - name: ES_JAVA_OPTS
              value: "-Xms2g -Xmx2g"
            - name: READINESS_PROBE_TIMEOUT
              value: "10"
            readinessProbe:
              exec:
                command:
                - bash
                - -c
                - /mnt/elastic-internal/scripts/readiness-
probe-script.sh
              failureThreshold: 3
              initialDelaySeconds: 10
              periodSeconds: 12
              successThreshold: 1
              timeoutSeconds: 12
          affinity:
            podAntiAffinity:
```

```
preferredDuringSchedulingIgnoredDuringExecution:
              - weight: 100
                podAffinityTerm:
                  labelSelector:
                    matchLabels:
                      elasticsearch.k8s.elastic.co/cluster-
name: elasticsearch-flowx
                  topologyKey: kubernetes.io/hostname
      # request 2Gi of persistent data storage for pods in
this topology element
      volumeClaimTemplates:
      - metadata:
          name: elasticsearch-data
        spec:
          accessModes:
          - ReadWriteOnce
          resources:
            requests:
              storage: 5Gi
          storageClassName: standard
    # 3 ingest-data nodes
    - name: ingest-data
      count: 3
      config:
        node.master: false
        node.data: true
        node.ingest: true
        # this allows ES to run on nodes even if their
vm.max_map_count has not been increased, at a performance
cost
        # node.store.allow_mmap: false
      podTemplate:
        spec:
          initContainers:
```

```
          - name: sysctl
            securityContext:
              privileged: true
            command: ['sh', '-c', 'sysctl -w
vm.max_map_count=262144']
          containers:
          - name: elasticsearch
            resources:
              limits:
                memory: 8Gi
                cpu: 2
              requests:
                memory: 4Gi
                cpu: 1
            env:
            - name: ES_JAVA_OPTS
              value: "-Xms2g -Xmx2g"
          affinity:
            podAntiAffinity:

preferredDuringSchedulingIgnoredDuringExecution:
              - weight: 100
                podAffinityTerm:
                  labelSelector:
                    matchLabels:
                      elasticsearch.k8s.elastic.co/cluster-
name: elasticsearch-flowx
                  topologyKey: kubernetes.io/hostname
          # nodeSelector:
          #   diskType: ssd
          #   environment: production
      # request 2Gi of persistent data storage for pods in
this topology element
      volumeClaimTemplates:
      - metadata:
```

```
        name: elasticsearch-data
      spec:
        accessModes:
        - ReadWriteOnce
        resources:
          requests:
            storage: 20Gi
        storageClassName: standard
```

(Optional) Kibana instance:

```
apiVersion: kibana.k8s.elastic.co/v1
kind: Kibana
metadata:
  name: kibana-flowx
  namespace: elastic-system
spec:
  version: 7.9.3
  count: 1
  elasticsearchRef:
    name: elasticsearch-flowx
    namespace: elastic-system
  config:
     elasticsearch.requestHeadersWhitelist:
     - authorization
  podTemplate:
    spec:
      containers:
      - name: kibana
        resources:
          requests:
            memory: 1Gi
            cpu: 0.5
```

```
            limits:
              memory: 3Gi
              cpu: 2
```

The index used by customer management plugin should be created.

Postgres database

This plugin can work without this database, it will not store the audit data.

Basic Postgres configuration

```
  crmdb:
    existingSecret: {{secretName}}
    metrics:
      enabled: true
      service:
        annotations:
          prometheus.io/port: {{phrometeus port}}
          prometheus.io/scrape: "true"
        type: ClusterIP
      serviceMonitor:
        additionalLabels:
          release: prometheus-operator
        enabled: true
        interval: 30s
        scrapeTimeout: 10s
    persistence:
      enabled: true
      size: 4Gi
    postgresqlDatabase: {{postgres databaseName}}
    postgresqlUsername: {{postgres user}}
    resources:
```

```
        limits:
          cpu: 500m
          memory: 512Mi
        requests:
          cpu: 200m
          memory: 256Mi
      service:
        annotations:
          fabric8.io/expose: "false"
```

Configuration

Authorization configuration

The following variables need to be set in order to connect to the identity management platform:

`SECURITY_OAUTH2_BASE_SERVER_URL`

`SECURITY_OAUTH2_CLIENT_CLIENT_ID`

`SECURITY_OAUTH2_REALM`

Datasource configuration

To store audit for searches this plugins use a postgres database.

The following configuration details need to be added using environment variables:

`SPRING_DATASOURCE_URL`

`SPRING_DATASOURCE_USERNAME`

`SPRING_DATASOURCE_PASSWORD`

You will need to make sure that the user, password, connection link and db name are configured correctly, otherwise you will receive errors at start time.

If you are going to use a database to store the audit, you can use the built-in script to maintain the database schema.

Elastic search configuration

The connection to elastic search cluster is done over https using the elastic search api. To connect to the it you will need to configure the connection details and index use to store customers.

```
elasticsearch:
  ssl: false
  nodes:
    -
      hostname: ${ELASTICSEARCH_HOST}
      port: ${ELASTICSEARCH_PORT}
      scheme: ${ELASTICSEARCH_HTTP_SCHEME}
  user: ${ELASTICSEARCH_USER}
  password: ${ELASTICSEARCH_PASSWORD}
  customer-index: ${ELASTICSEARCH_CUSTOMER_INDEX}
  size: 10
```

Kafka configuration

The following Kafka related configurations can be set by using environment variables:

`SPRING_KAFKA_BOOTSTRAP_SERVERS` - address of the Kafka server

`SPRING_KAFKA_CONSUMER_GROUP_ID` - group of consumers

`KAFKA_CONSUMER_THREADS` - the number of Kafka consumer threads

`KAFKA_AUTH_EXCEPTION_RETRY_INTERVAL` - the interval between retries after `AuthorizationException` is thrown by `KafkaConsumer`

`KAFKA_MESSAGE_MAX_BYTES` - this is the largest size of the message that can be received by the broker from a producer.

Each action available in the service corresponds to a Kafka event. A separate Kafka topic must be configured for each use-case.

> **CAUTION**
>
> The Engine is listening for messages on topics with names of a certain pattern, make sure to use correct outgoing topic names when configuring the documents plugin.

Needed topics:

`KAFKA_TOPIC_CUSTOMER_SEARCH_IN`

`KAFKA_TOPIC_CUSTOMER_SEARCH_OUT`

> **CAUTION**

> In order to match a request made to the customer management plugin, the engine will have to send the process id on a Kafka header.

Logging

The following environment variables could be set in order to control log levels:

`LOGGING_LEVEL_ROOT` - root spring boot microservice logs

`LOGGING_LEVEL_APP` - app level logs

Was this page helpful?

PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Documents plugin setup / Configuring access rights for Documents

Granular access rights can be configured for restricting access to the Documents plugin component.

The following access authorizations is provided, with the specified access scopes:

1. **Manage-document-templates** - for configuring access for managing document templates

Available scopes:

- import - users are able to import document templates
- read - users are able to view document templates
- edit - users are able to edit document templates
- admin - users are able to publish or delete document templates The Document plugin is preconfigured with the following default users roles for each of the access scopes mentioned above:
- manage-document-templates
    - import:
        - ROLE_DOCUMENT_TEMPLATES_IMPORT
        - ROLE_DOCUMENT_TEMPLATES_EDIT
        - ROLE_DOCUMENT_TEMPLATES_ADMIN
    - read
        - ROLE_DOCUMENT_TEMPLATES_READ
        - ROLE_DOCUMENT_TEMPLATES_IMPORT
        - ROLE_DOCUMENT_TEMPLATES_EDIT
        - ROLE_DOCUMENT_TEMPLATES_ADMIN
    - edit:
        - ROLE_DOCUMENT_TEMPLATES_EDIT
        - ROLE_DOCUMENT_TEMPLATES_ADMIN
    - admin:
        - ROLE_DOCUMENT_TEMPLATES_ADMIN

ROLES

> These roles need to be defined in the chosen identity provider solution.

In case other custom roles are needed, you can configure them using environment variables. More than one role can be set for each access scope.

To configure access for each of the roles above, adapt the following input:

`SECURITY_ACCESSAUTHORIZATIONS_AUTHORIZATIONNAME_SCOPES_SCOPENAME_ROLESALLOWED: NEEDED_ROLE_NAMES`

Possible values for AUTHORIZATIONNAME: MANAGEDOCUMENTTEMPLATES.

Possible values for SCOPENAME: import, read, edit, admin.

For example, if you need to configure role access for read, insert this:

```
SECURITY_ACCESSAUTHORIZATIONS_MANAGEDOCUMENTTEMPLATES_SCOPES_RE
ROLE_NAME_TEST
```

Was this page helpful?

PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Notification templates plugin setup / Configuring access rights for Notifications

Granular access rights can be configured for restricting access to the Notification plugin component.

The following access authorizations are provided, with the specified access scopes:

1. **Manage-notification-templates** - for configuring access for managing notification templates

Available scopes:

- import - users are able to import notification templates
- read - users are able to view notification templates
- edit - users are able to edit notification templates
- admin - users are able to publish or delete notification templates

The Notification plugin is preconfigured with the following default users roles for each of the access scopes mentioned above:

- manage-notification-templates
  - import
    - ROLE_NOTIFICATION_TEMPLATES_IMPORT
    - ROLE_NOTIFICATION_TEMPLATES_EDIT
    - ROLE_NOTIFICATION_TEMPLATES_ADMIN
  - read:
    - ROLE_NOTIFICATION_TEMPLATES_READ
    - ROLE_NOTIFICATION_TEMPLATES_IMPORT
    - ROLE_NOTIFICATION_TEMPLATES_EDIT
    - ROLE_NOTIFICATION_TEMPLATES_ADMIN

- edit:
  - ROLE_NOTIFICATION_TEMPLATES_EDIT"
  - ROLE_NOTIFICATION_TEMPLATES_ADMIN"
- admin:
  - ROLE_NOTIFICATION_TEMPLATES_ADMIN

> **CAUTION**
>
> These roles need to be defined in the chosen identity provider solution.

In case other custom roles are needed, you can configure them using environment variables. More than one role can be set for each access scope.

To configure access for each of the roles above, adapt the following input:

`SECURITY_ACCESSAUTHORIZATIONS_AUTHORIZATIONNAME_SCOPES_SCOPENAME_ROLESALLOWED: NEEDED_ROLE_NAMES`

Possible values for AUTHORIZATIONNAME: `MANAGENOTIFICATIONTEMPLATES`.

Possible values for SCOPENAME: import, read, edit, admin.

For example, if you need to configure role access for read, insert this:

```
SECURITY_ACCESSAUTHORIZATIONS_MANAGENOTIFICATIONTEMPLATES_SCOPE
ROLE_NAME_TEST
```

Was this page helpful?

PLATFORM DEEP DIVE / Plugins / Plugins setup guides / OCR plugin setup

The OCR plugin is a docker image that can be deployed using the following infrastructure prerequisites:

Infrastructure Prerequisites:

- S3 bucket or alternative (for example, minio)
- Kafka cluster

> **IMPORTANT**
>
> Starting with `ocr-plugin 1.X` it no longer requires RabbitMQ.
>
> The following environment from previous releases must be removed in order to use OCR plugin: `CELERY_BROKER_URL`.

Deployment/Configuration

To deploy the OCR plugin, you will need to deploy `ocr-plugin` helm chart with custom values file.

Most important sections are these, but more can be extracted from helm chart.

```
image:
  repository: <repository>/ocr-plugin

applicationSecrets: {}

replicaCount: 2

resources: {}

env: []
```

Credentials

S3 bucket:

```
applicationSecrets:
  enable: true
  envSecretKeyRef:
    STORAGE_S3_ACCESS_KEY: access-key # default empty
    STORAGE_S3_SECRET_KEY: secret-key # default empty
  existingSecret: true
  secretName: ocr-plugin-application-config
```

Kafka configuration

You can override the following environment variables:

Environment Variable	Definition	Default Value

Environment Variable	Definition	Default Value
`ENABLE_KAFKA_SASL`	Indicates whether Kafka SASL authentication is enabled	`False`
`KAFKA_ADDRESS`	The address of the Kafka bootstrap server in the format `<hostname>:<port>`	-
`KAFKA_CONSUME_SCHEDULE`	The interval (in seconds) at which Kafka messages are consumed	`30`
`KAFKA_INPUT_TOPIC`	The Kafka topic from which input messages are consumed	-

Environment Variable	Definition	Default Value
`KAFKA_OCR_CONSUMER_GROUPID`	The consumer group ID for the OCR Kafka consumer	`ocr_group`
`KAFKA_CONSUMER_AUTO_COMMIT`	Determines whether Kafka consumer commits offsets automatically	`True`
`KAFKA_CONSUMER_AUTO_COMMIT_INTERVAL`	The interval (in milliseconds) at which Kafka consumer commits offsets automatically	`1000`

Environment Variable	Definition	Default Value
`KAFKA_CONSUMER_TIMEOUT`	The timeout (in milliseconds) for Kafka consumer operations	`28000`
`KAFKA_CONSUMER_MAX_POLL_INTERVAL`	The maximum interval (in milliseconds) between consecutive polls for Kafka consume	`25000`
`KAFKA_CONSUMER_AUTO_OFFSET_RESET`	The strategy for resetting the offset when no initial offset is available or if the current offset is invalid	`earliest`

Environment Variable	Definition	Default Value
`KAFKA_OUTPUT_TOPIC`	The Kafka topic to which output messages are sent	-

Please note that the default values and examples provided here are for illustrative purposes. Make sure to replace them with the appropriate values based on your Kafka configuration.

> **CAUTION**
>
> When configuring the OCR plugin, make sure to use the correct outgoing topic names that match **the pattern expected by the Engine**, which listens for messages on topics with specific names.

Authorization

You can override the following environment variables:

Environment Variable	Definition	Default Value	E

Environment Variable	Definition	Default Value	E
`OAUTH_CLIENT_ID`	The client ID for OAuth authentication	-	`your_client_i`
`OAUTH_CLIENT_SECRET`	The client secret for OAuth authentication	-	`your_client_s`
`OAUTH_TOKEN_ENDPOINT_URI`	The URI of the token endpoint for OAuth authentication	-	`https://oauth`

Please note that the default values and examples provided here are for illustrative purposes. Make sure to replace them with the appropriate values based on your OAuth authentication configuration.

Storage (S3 configuration)

You can override the following environment variables:

Environment Variable	Definition	Default Value	

Environment Variable	Definition	Default Value	
`STORAGE_S3_HOST`	The host address of the S3 storage service	-	`minio: west-1`
`STORAGE_S3_SECURE_CONNECTION`	Indicates whether to use a secure connection (HTTPS) for S3 storage	`False`	
`STORAGE_S3_LOCATION`	The location of the S3 storage service	-	`eu-wes`

Environment Variable	Definition	Default Value	
`STORAGE_S3_OCR_SCANS_BUCKET`	The name of the S3 bucket for storing OCR scans	-	`pdf-sc`
`STORAGE_S3_OCR_SIGNATURE_BUCKET`	The name of the S3 bucket for storing OCR signatures	-	`extrac`
`STORAGE_S3_OCR_SIGNATURE_FILENAME`	The filename pattern for extracted OCR signatures	-	`extrac`

Environment Variable	Definition	Default Value	
`STORAGE_S3_SECRET_KEY`	The secret key for connecting to the S3 storage service	-	

Please note that the default values and examples provided here are for illustrative purposes. Make sure to replace them with the appropriate values based on your S3 storage configuration.

Performance

Environment Variable	Definition	Default Value
`ENABLE_PERFORMANCE_PAYLOAD`	When set to true, the response payload will contain performance metrics related to various stages of the process.	`true`

Example

```
"perf": {
  "total_time": 998,
  "split": {
    "get_file": 248,
    "extract_images": 172,
    "extract_barcodes": 37,
    "extract_signatures": 238,
    "minio_signature_save": 301
  }
}
```

Certificates

You can override the following environment variables:

- `REQUESTS_CA_BUNDLE` - the path to the certificate bundle file used for secure requests

Workers behavior

You can override the following environment variables:

Environment Variable	Definition	Default Value
`OCR_WORKER_COUNT`	Number of workers	`5`

Environment Variable	Definition	Default Value
`OCR_WORK_QUEUE_TIMEOUT`	If no activity has occurred for a certain number of seconds, an attempt will be made to refresh the workers	`10`

INFO

If no worker is released after `OCR_WORK_QUEUE_TIMEOUT` seconds, the application will verify whether any workers have become unresponsive and need to be restarted.

If none of the workers have died, it means they are likely blocked in some process. In this case, the application will terminate all the workers and shut down itself, hoping that the container will be restarted.

Was this page helpful?

PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Reporting setup guide

The reporting plugin is available a docker image, and it has the following dependencies:

Dependencies

- a reporting PostgreSQL instance
- reporting-plugin helm chart - containing cronJob which performs the following actions:
  - reads from FLOWX.AI Engine db
  - writes in the FLOWX.AI Reporting plugin db
- Superset:
  - a Superset PostgreSQL db
  - a Redis instance for caching
  - exposes the UI through an ingress -> host needed

Postgres database

Basic Postgres configuration:

```
postgresql:
  enabled: true
  postgresqlUsername: {{userName}}
  postgresqlPassword: ""
  postgresqlDatabase: "reporting"
  existingSecret: {{scretName}}
  persistence:
    enabled: true
    storageClass: standard-rwo
    size: 5Gi
  resources:
    limits:
      cpu: 1000m
      memory: 1024Mi
    requests:
```

```
      memory: 256Mi
      cpu: 100m
  metrics:
    enabled: true
    serviceMonitor:
      enabled: false
    prometheusRule:
      enabled: false
  primary:
    nodeSelector:
      preemptible: "false"
```

Reporting plugin helm chart (containing CRON)

reporting-plugin helm.yaml

```
sync:
  cronjob:
    image:
      repository: {{env}}/reporting-plugin

    schedule: "*/5 * * * *"

    extraEnvVarsMultipleSecretsCustomKeys:
      - name: process-engine-application-config
        secrets:
          ENGINE_DATABASE_PASSWORD: {{db paswword}}
        secrets:
          REPORTING_DATABASE_PASSWORD: {{db password}}

    env:
      ENGINE_DATABASE_USER: {{engine db user}}
```

```
        ENGINE_DATABASE_URL: {{engine db URL}}
        ENGINE_DATABASE_NAME: {{engine db name}}

        REPORTING_DATABASE_USER: {{reporting db user}}
        REPORTING_DATABASE_URL: {{reporting db URL}}
        REPORTING_DATABASE_NAME: {{reporting db name}}
```

Superset

» Superset configuration

» Superset documentation

After installation

- datasource URL -> FLOWX.AI Reporting database
- Datasets
- Dashboards

Datasource configuration

To store data related to document templates and documents the service uses a Postgres database.

The following configuration details need to be added using environment variables:

`SPRING_DATASOURCE_URL`

`SPRING_DATASOURCE_USERNAME`

`SPRING_DATASOURCE_PASSWORD`

You will need to make sure that the user, password, connection link and db name are configured correctly, otherwise you will receive errors at start time.

The datasource is configured automatically via a liquibase script inside the service. All updates will include migration scripts.

> **INFO**
>
> Database schema is managed by a liquibase script that will create, manage and migrate future versions.

Redis configuration

The following values should be set with the corresponding Redis-related values:

`SPRING_REDIS_HOST`

`SPRING_REDIS_PORT`

Keycloak configuration

To enable a different user authentication than the regular one (database), you need to override the `AUTH_TYPE` parameter in your `superset` .yml file.

It would look something like this:

`AUTH_TYPE: AUTH_OID`

You will also need to provide a reference to your `openid-connect` realm:

`OIDC_OPENID_REALM: 'flowx'`

With this configuration, the login page changes to a prompt where the user can select the desired OpenID provider (in our case keycloak)

Extend the Security Manager

Firstly, you will want to make sure that flask stops using `flask-openid` and starts using `flask-oidc` instead.

To do so, you will need to create your own security manager that configures `flask-oidc` as its authentication provider.

```
extraSecrets:
  keycloak_security_manager.py: |
    from flask_appbuilder.security.manager import AUTH_OID
    from superset.security import SupersetSecurityManager
    from flask_oidc import OpenIDConnect
```

To enable OpenID in Superset, you would previously have had to set the authentication type to `AUTH_OID`.

The security manager still executes all the behavior of the super class, but overrides the OID attribute with the `OpenIDConnect` object.

Further, it replaces the default OpenID authentication view with a custom one:

```
    from flask_appbuilder.security.views import AuthOIDView
    from flask_login import login_user
    from urllib.parse import quote
    from flask_appbuilder.views import expose
    from flask import request, redirect

    class AuthOIDCView(AuthOIDView):
        @expose('/login/', methods=['GET', 'POST'])
        def login(self, flag=True):
            sm = self.appbuilder.sm
            oidc = sm.oid
            superset_roles = ["Admin", "Alpha", "Gamma",
"Public", "granter", "sql_lab"]
            default_role = "Admin"
            @self.appbuilder.sm.oid.require_login
            def handle_login():
                user =
sm.auth_user_oid(oidc.user_getfield('email'))
                if user is None:
                    info =
oidc.user_getinfo(['preferred_username', 'given_name',
'family_name', 'email', 'roles'])
                    roles = [role for role in superset_roles
if role in info.get('roles', [])]
                    roles += [default_role, ] if not roles
else []
                    user =
sm.add_user(info.get('preferred_username'),
info.get('given_name', ''), info.get('family_name', ''),
                                            info.get('email'),
[sm.find_role(role) for role in roles])
                login_user(user, remember=False)
```

```
                return
redirect(self.appbuilder.get_url_for_index)
            return handle_login()
        @expose('/logout/', methods=['GET', 'POST'])
        def logout(self):
            oidc = self.appbuilder.sm.oid
            oidc.logout()
            super(AuthOIDCView, self).logout()
            redirect_url = request.url_root.strip('/')
            # redirect_url = request.url_root.strip('/') +
self.appbuilder.get_url_for_login
            return redirect(
                oidc.client_secrets.get('issuer') +
'/protocol/openid-connect/logout?redirect_uri=' +
quote(redirect_url))
```

On authentication, the user is redirected back to Superset.

Configure Superset

Finally, we need to add some parameters to the superset .yml file:

```
    '''
    ------------------------KEYCLOACK ------------------
---------
    '''
    curr  =  os.path.abspath(os.getcwd())
    AUTH_TYPE = AUTH_OID
    OIDC_CLIENT_SECRETS =  curr +
'/pythonpath/client_secret.json'
    OIDC_ID_TOKEN_COOKIE_SECURE = True
    OIDC_REQUIRE_VERIFIED_EMAIL = True
    OIDC_OPENID_REALM: 'flowx'
```

```
    OIDC_INTROSPECTION_AUTH_METHOD: 'client_secret_post'
    CUSTOM_SECURITY_MANAGER = OIDCSecurityManager
    AUTH_USER_REGISTRATION = False
    AUTH_USER_REGISTRATION_ROLE = 'Admin'
    OVERWRITE_REDIRECT_URI = 'https://{{
.Values.flowx.ingress.reporting }}/oidc_callback'
    '''
    ------------------------------------------------------------
------
    '''
```

Was this page helpful?

PLATFORM DEEP DIVE / Plugins / Plugins setup guides / Task Manager plugin setup / Configuring access rights for Task management

Granular access rights can be configured for restricting access to the Task management plugin component.

Two different access authorizations are provided, each with specified access scopes:

1. **manage-tasks** - for configuring access for viewing the tasks lists

Available scopes:

- read - users are able to view tasks

2. **manage-hooks** - for configuring access for managing hooks

Available scopes:

- import - users are able to import hooks
- read - users are able to view hooks
- edit - users are able to edit hooks
- admin - users are able to delete hooks

3. **manage-process-allocation-settings** - for configuring access for managing process allocation settings

Available scopes:

- import - users are able to import allocation rules
- read - users are able to read/export allocation rules
- edit - users are able to edit access - create/edit allocation rules
- admin - users are able to delete allocation rules

4. **manage-out-of-office-users** - for configuring access for managing out-of-office users

Available scopes:

- read - users are able to view out-of-office records
- edit - users are able to create and edit out-of-office records
- admin - users are able to delete out-of-office records

The Task management plugin is preconfigured with the following default users roles for each of the access scopes mentioned above:

- manage-tasks
  - read:
    - ROLE_TASK_MANAGER_TASKS_READ
- manage-hooks
  - import:
    - ROLE_TASK_MANAGER_HOOKS_IMPORT
    - ROLE_TASK_MANAGER_HOOKS_EDIT
    - ROLE_TASK_MANAGER_HOOKS_ADMIN
  - read:
    - ROLE_TASK_MANAGER_HOOKS_READ
    - ROLE_TASK_MANAGER_HOOKS_IMPORT
    - ROLE_TASK_MANAGER_HOOKS_EDIT
    - ROLE_TASK_MANAGER_HOOKS_ADMIN
  - edit:
    - ROLE_TASK_MANAGER_HOOKS_EDIT
    - ROLE_TASK_MANAGER_HOOKS_ADMIN
  - admin:
    - ROLE_TASK_MANAGER_HOOKS_ADMIN
- manage-process-allocation-settings
  - import:
    - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_IMPORT
    - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_EDIT

        - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_ADMIN
    - read:
        - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_READ
        - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_IMPORT
        - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_EDIT
        - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_ADMIN
    - edit:
        - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_EDIT
        - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_ADMIN
    - admin:
        - ROLE_TASK_MANAGER_PROCESS_ALLOCATION_SETTINGS_ADMIN
- manage-out-of-office-users
    - read:
        - ROLE_TASK_MANAGER_OOO_READ
        - ROLE_TASK_MANAGER_OOO_EDIT
        - ROLE_TASK_MANAGER_OOO_ADMIN
    - edit:
        - ROLE_TASK_MANAGER_OOO_EDIT
        - ROLE_TASK_MANAGER_OOO_ADMIN

- admin:
  - ROLE_TASK_MANAGER_OOO_ADMIN

> **CAUTION**
>
> These roles need to be defined in the chosen identity provider solution.

In case other custom roles are needed, you can configure them using environment variables. More than one role can be set for each access scope.

To configure access for each of the roles above, adapt the following input:

`SECURITY_ACCESSAUTHORIZATIONS_AUTHORIZATIONNAME_SCOPES_SCOPENAME_ROLESALLOWED: NEEDED_ROLE_NAMES`

Possible values for AUTHORIZATIONNAME: MANAGETASKS, MANAGEHOOKS.

Possible values for SCOPENAME: import, read, edit, admin.

For example, if you need to configure role access for read, insert this:

```
SECURITY_ACCESSAUTHORIZATIONS_MANAGEHOOKS_SCOPES_READ_ROLESALL
ROLE_NAME_TEST
```

Was this page helpful?