



TIBCO® Product and Service Inventory

User Guide

*Version 2.1.0
August 2022*



Contents

Contents	2
Product and Service Inventory Overview	6
Uses of Product and Service Inventory	7
Data Managed by Inventory	8
Object Locking	9
Lock Requests	9
Lock Notifications	10
Lock Modes	11
Create With Lock	11
Prevent Concurrent Update	12
Delayed Locking	12
Admin Update When Not Lock Owner Permission	13
Use Cases for Lock Obtained Notification	14
Create Party with Immediate Lock, say Lock 1	14
Update the Party with Delayed Lock, say Lock 2	14
Update the Party to Remove the Lock, say Lock 1	14
Use Cases for Lock Removal Notification	15
Create Party with Immediate Lock	15
Update the Party with Delayed Lock, say Lock 2 with Timeout 240 Seconds	16
Update the Same Party to Remove the Lock, say Lock 2	16
Use Cases for Lock Expired Notification	17
Create Party with Immediate Lock, say Lock 1	17
Update the Party with Delayed Lock	17
Batch Functionality	18

Configuration Values	20
Oracle RAC Configuration	26
Product and Service Inventory User Interface	27
Multitenancy in Product and Service Inventory	27
Basic Access Controls of Product and Service Inventory	27
Logging in to Product and Service Inventory	28
Product and Service Inventory Main Page	29
Searching Items and Parties	30
Logging out of Product and Service Inventory	30
Items	32
Adding a New Item	33
Searching an Item	35
Updating an Item	36
Locking an Existing Item	36
Deleting the Lock of an Item	37
Deleting an Existing Item	37
Parties	38
Adding a New Party	39
Searching a Party	40
Updating a Party	41
Locking a Party	41
Deleting the Lock of a Party	42
Deleting a Party	42
Locks	43
Viewing Locked Items	43
Viewing Locked Parties	43
Bulk Load	44
Bulk Load Process	44
Oracle Bulk Load Setup	46
Bulk Load JSON Configuration	46

Bulk Load Log Files	49
Configurator UI	50
Navigating the Configurator UI	50
Uploading Metadata File	51
Configuring an Application	52
Editing Application Properties	55
Replicating Application Properties	57
Editing Configuration Files	57
Configuration Service	60
Get supported files metadata	60
Save supported files metadata	60
Delete supported files metadata	61
Get Application Properties by Application ID	61
Update Application Properties for Application ID	61
Save Application Properties for Application ID	61
Delete Application Properties for Application ID	62
Replicate Tenant Properties	62
Get list of applications available with Configurator	62
Download Configuration File for Application ID	62
Upload Configuration File for Application ID	63
Configurator notification API	63
Authorization Service	64
Create User	64
Update User	66
Get User	68
Delete User	69
TIBCO Documentation and Support Services	71
Product-Specific Documentation	71

Legal and Third-Party Notices	73
--	-----------

Product and Service Inventory Overview

TIBCO® Product and Service Inventory (Formerly TIBCO® Fulfillment Subscriber Inventory) holds specific information about subscribers and the services or products that the subscribers have purchased, are about to purchase or are being provisioned.

The function of the TIBCO Product and Service Inventory is to maintain a current image of customer-installed products at any given point in time, which is capable of supporting fast concurrent read or write access while ensuring data consistency. TIBCO Product and Service Inventory also provides a rich, user-friendly web interface that can be used to explore and modify the contents of the system safely and securely.

Uses of Product and Service Inventory

You can use Product and Service Inventory to:

- Store data (characteristics) related to the Parties
- Create hierarchical party support for coping with complex organizations, such as an enterprise comprising of many groups or employees
- Create a complex structure of item support to allow storage of complex relationships defined in the Fulfillment Catalog (ProductComprisedOf or ProductDependsOn relationships stored with products)
- Store data (characteristics) related to the Items
- Store related orders for an item so that you can walk through the item history
- Create a relationship between party and item, or, if no party is used, adding a party name to an item for retrieval of subscriber products
- Implement CRUD operations for data management on both party and item
- Search both party and item
- Lock a resource to be protected against other concurrent updates
- Create a batch that allows multiple updates to be grouped, either committed, or rolled back entirely
- Implement access management for user roles and users, so that the access can be restricted
- Implement multi-tenancy for cloud-based deployment and operation for multiple customers

Data Managed by Inventory

The data managed by TIBCO Product and Service Inventory are as follows:

Inventory Managed Data

Party	Stores customer, subscriber, or other party-related information. See the Parties topic for more details.
Item	Stores products, services, or other tangible or intangible entities that have been ordered by a party, and are present as components of that party's image. See the Items topic for more details.

Object Locking

Party and item entities provide the capability for locking and lock management. This is useful in a context where multiple concurrent processes are all working on the same object but it is necessary to control the order of access to the object in a defined sequence.

It is important to note that the Product and Service Inventory locking functionality does not provide any logic or rule-based checking of operations invoked against an object. It only registers a list of locks of interest and publishes the lock notifications in a controlled sequence. If clients do not respect the object locks, there are no hard rules within the Product and Service Inventory system to prevent the clients from accessing or updating the object out of sequence. Failure to respect the locking functionality may violate business requirements and result in an inconsistent state within the Inventory system.

Lock Requests

The Lock requests provide support to the following criteria:

- A key identifying the lock.
- A timeout in seconds:
 - "0", to request an immediate lock, which means the lock request is either successful and grants the lock right away, or returns an error.
 - Non-zero, to request a lock that can be delayed. If the lock cannot be granted right away, it is delayed by the entity. If the entity is released before getting timed out, the lock is granted to the requester. Otherwise, the lock request gets timed out and the request is discarded.
- A priority (higher the value, higher the priority). When a lock is released and there are delayed lock requests, the system grants the lock to the request with a higher priority. If the priority value is the same for many requests then the priority placed earlier gets the lock.

Lock Notifications

TIBCO Product and Service Inventory 2.1.0 supports the following features:

- Sending message notifications using JMS (TIBCO EMS™) messaging system.
- Enabling or disabling the notifications. The configuration for enabling or disabling notifications is in the \$PSI_HOME/seed-data/app-properties/ConfigValues_PSIService.json file. The configuration for using a message notifier, the TIBCO EMS™ is provided in the \$PSI_HOME/seed-data/app-properties/ConfigValues_PSIService.json file. You can also keep the default value none. See the [Configuration Values](#) topic for more details.

The granting of a lock or the expiry of a lock is asynchronous. Client applications must have channels to receive lock notifications. By default, TIBCO Product and Service Inventory uses JMS topics to receive lock notifications. There are three topics per channel. They are:

Topics	Description
com.tibco.inventory.notification.lock.obtained.topic. [tenantId]	Used for messages, which indicate that a lock was granted.
com.tibco.inventory.notification.lock.removed.topic. [tenantId]	Used for messages which indicate that a lock request was removed by the user or an API interaction. <div> <p>Note:</p> <ul style="list-style-type: none"> • The message is not sent when the current lock is "unlocked" on an item or party. It is only meant for removal. • New lock request to acquire a lock on the item or the party gets a lock obtained notification. </div>
com.tibco.inventory.notification.lock.expired.topic. [tenantId]	Used for messages, which indicate that a lock request expired.

Lock Modes

Product and Service Inventory supports two modes of locking. You can configure the modes of locking as per your requirements.

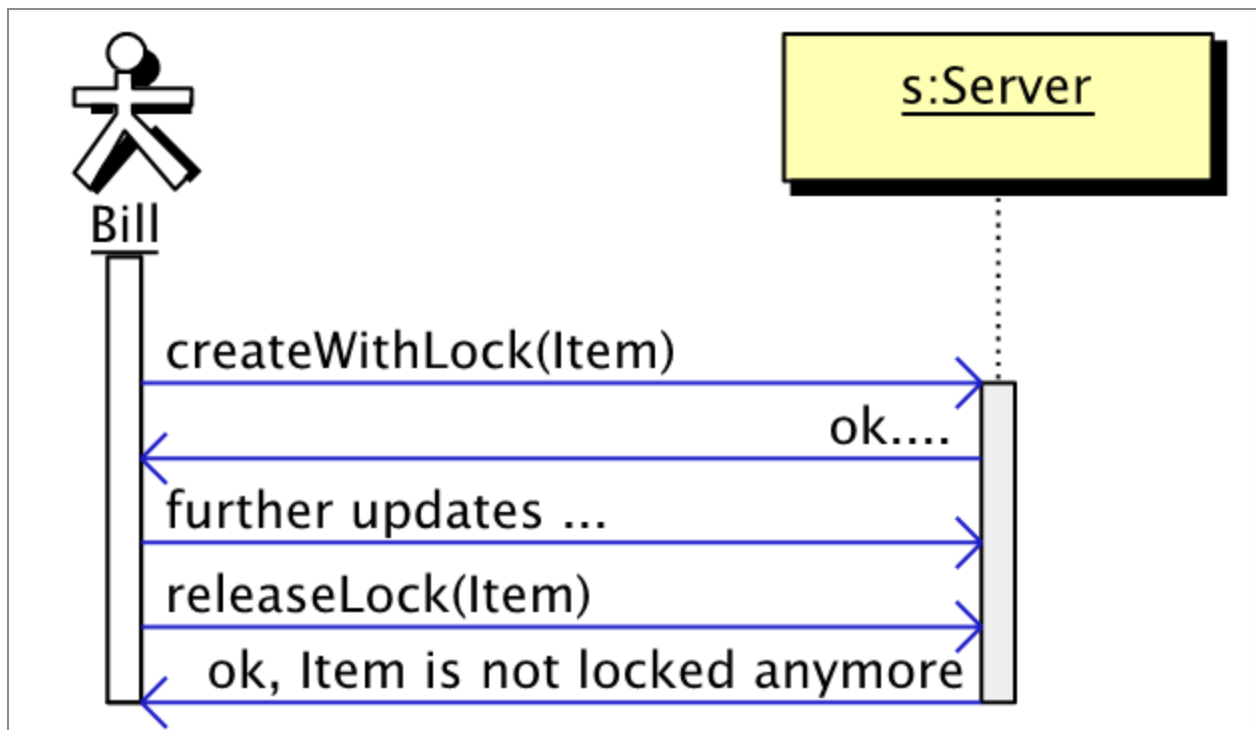
Locks are Advisory	You can always perform an update on an item or a party. You have to check if you need to perform an update or not.
--------------------	--

Locks are Hard	You cannot perform an update if the entity is locked and you were not the one to implement the lock. The default lock mode for all users is hard lock, except the user whose role is admin. The admin user role, by default, has the PERM_UPDATE_WHEN_NOT_LOCK_OWNER permission.
----------------	--

Create With Lock

You can use SOAP API to create an entity with immediate Lock in place.

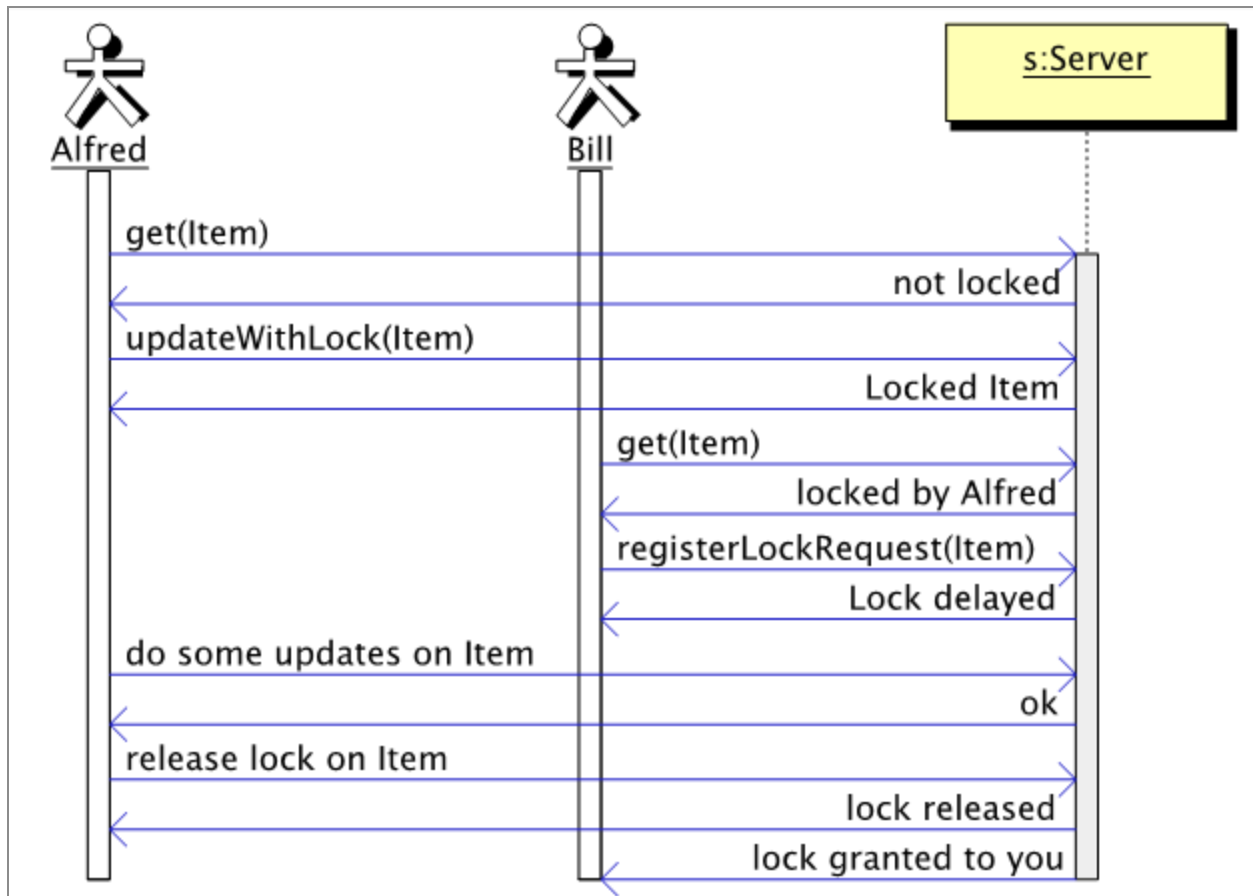
Create With Lock



Prevent Concurrent Update

The following diagram illustrates the concurrent aspect of locks:

Prevent Concurrent Update



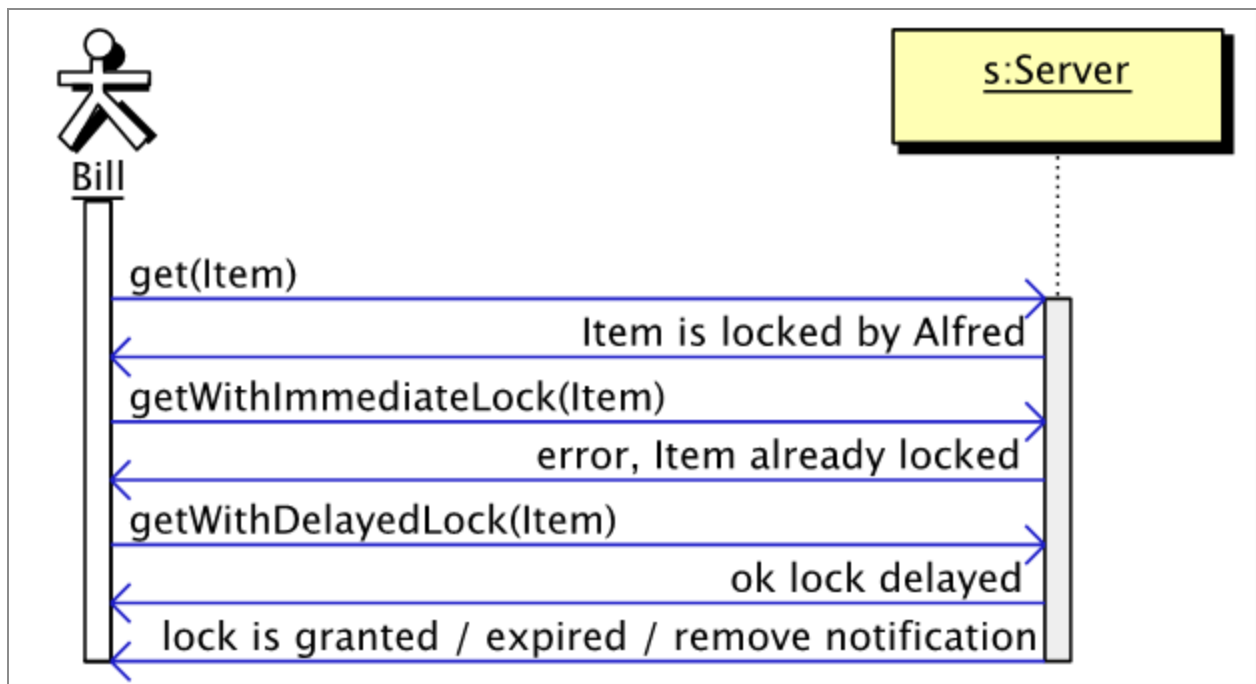
Whether locks are hard or advisory, you must always read an Item before trying to lock the item.

If you request a lock with timeout and the lock gets delayed, the lock is granted, and a notification is published on the JMS topic to indicate the granting of the lock.

Delayed Locking

The following diagram provides additional details related to the immediate or delayed locking mechanism:

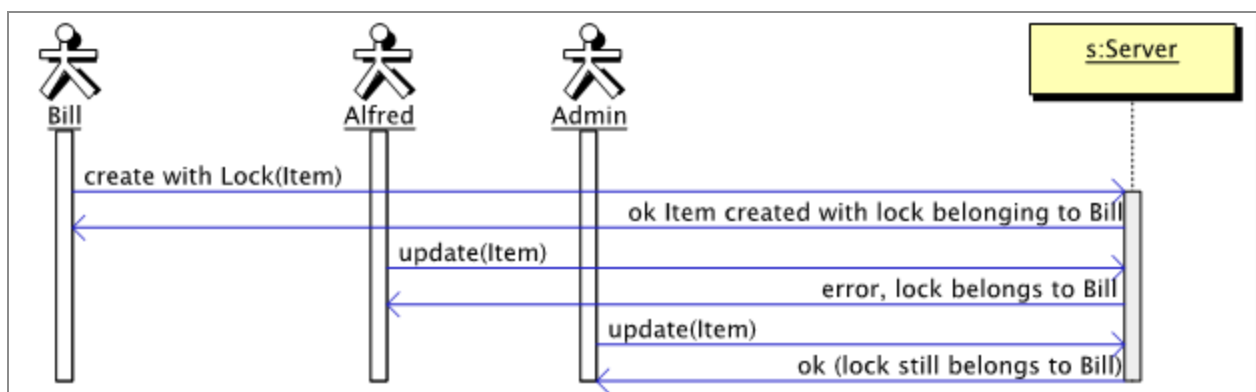
Get with Lock Delayed



Admin Update When Not Lock Owner Permission

If you are a user with administrator privileges you need not own a lock to update the entity.

Admin Update When Not Lock Owner Permission



Use Cases for Lock Obtained Notification

The following use cases describe the lock obtained notification for parties.

Create Party with Immediate Lock, say Lock 1

```
<ser:createPartyRequest businessTransactionId="?" correlationId="?"
sessionId="?">
  <par:entity>
    <com:type>enterprise</com:type>
    <com:status>active</com:status>
    <!--Optional:-->
    <com:id>Party1</com:id>
    <com:lock>Party1_KEY_1</com:lock>
  </par:entity>
</ser:createPartyRequest>
```

Update the Party with Delayed Lock, say Lock 2

```
<ser:updatePartyRequest businessTransactionId="?" correlationId="?"
sessionId="?">
  <par:entity>
    <com:id> Party1</com:id>
    <com:lock>
      <com:key> Party1_KEY_2</com:key>
      <com:priority>10</com:priority>
      <com:timeout>800</com:timeout>
    </com:lock>
  </par:entity>
</ser:updatePartyRequest>
```

Update the Party to Remove the Lock, say Lock 1

```
<ser:updatePartyRequest businessTransactionId="?" correlationId="?"
sessionId="?">
```

```

<!--Optional:-->
<!--1 or more repetitions:-->
<par:entity>
  <com:id>Party1</com:id>
  <com:unlock>
  <com:key> Party1_KEY_1</com:key>
</com:unlock>
</par:entity>
</ser:updatePartyRequest>

```

i Note: As Party1_KEY_1 is removed, Party1_KEY_2 is automatically acquired by the party. You get (Party1_KEY_2) lock obtained notification on topic "com.tibco.inventory.notification.lock.obtained.topic.tibco".

You do not get the (Party1_KEY_1) lock removed notification on a topic as it is explicitly removed.

Use Cases for Lock Removal Notification

The following use cases describe the lock removal notification for parties.

Create Party with Immediate Lock

```

<ser:createPartyRequest businessTransactionId="?" correlationId="?"
  sessionId="?">
  <par:entity>
    <com:type>enterprise</com:type>
    <com:status>active</com:status>
    <!--Optional:-->
    <com:id>Party1</com:id>
    <com:lock>Party1_KEY_1</com:lock>
  </par:entity>
</ser:createPartyRequest>

```

Update the Party with Delayed Lock, say Lock 2 with Timeout 240 Seconds

```
<ser:updatePartyRequest businessTransactionId="?" correlationId="?"
  sessionId="?">
  <par:entity>
    <com:id> Party1</com:id>
    <com:lock>
      <com:key> Party1_KEY_2</com:key>
      <com:priority>10</com:priority>
      <com:timeout>240</com:timeout>
    </com:lock>
  </par:entity>
</ser:updatePartyRequest>
```

Update the Same Party to Remove the Lock, say Lock 2

```
<ser:updatePartyRequest businessTransactionId="?" correlationId="?"
  sessionId="?">
  <!--Optional:-->
  <!--1 or more repetitions:-->
  <par:entity>
    <com:id>Party1</com:id>
    <com:unlock>
      <com:key> Party1_KEY_2</com:key>
    </com:unlock>
  </par:entity>
</ser:updatePartyRequest>
```

i Note: Here, Party1_KEY_2 is removed explicitly which is waiting for the party till its timeout. You get (Party1_KEY_2) lock removed notification on topic "com.tibco.inventory.notification.lock.removed.topic.tibco".

Use Cases for Lock Expired Notification

The following use cases describe the lock expired notification for parties.

Create Party with Immediate Lock, say Lock 1

```
<ser:createPartyRequest businessTransactionId="?" correlationId="?"
sessionId="?">
  <par:entity>
    <com:type>enterprise</com:type>
    <com:status>active</com:status>
    <!--Optional:-->
    <com:id>Party1</com:id>
    <com:lock>Party1_KEY_1</com:lock>
  </par:entity>
</ser:createPartyRequest>
```

Update the Party with Delayed Lock

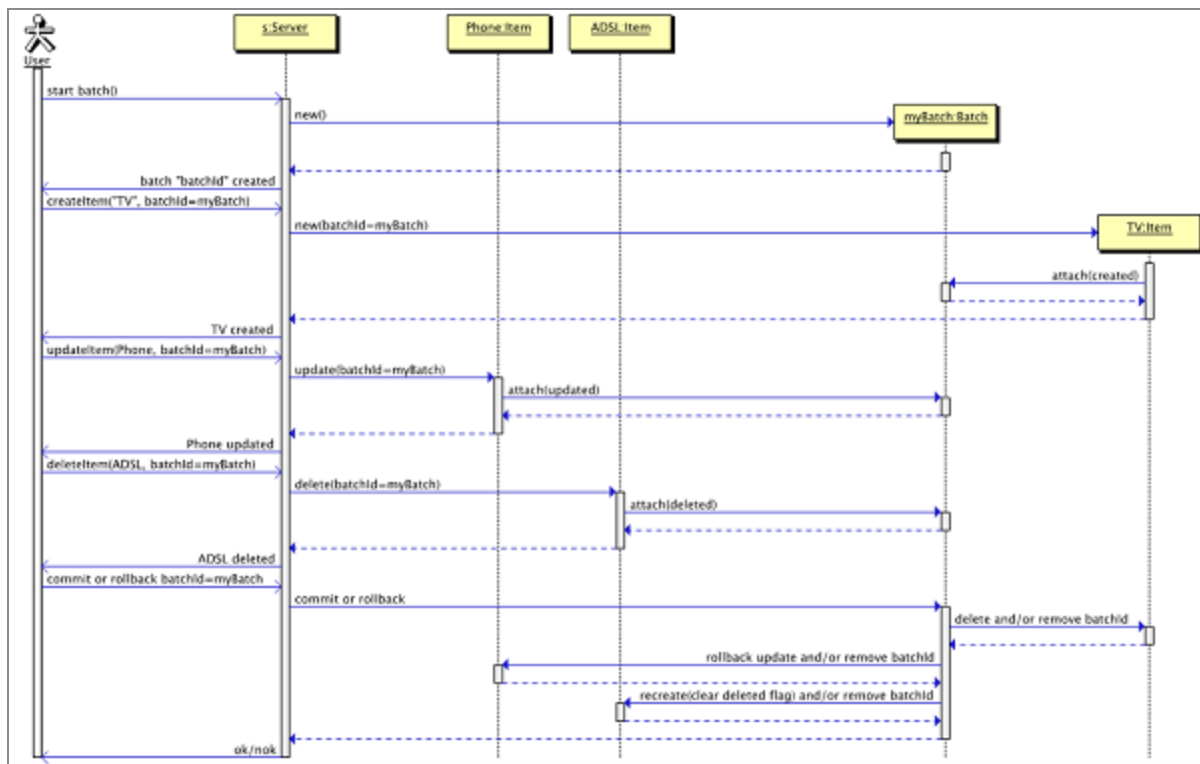
```
<ser:updatePartyRequest businessTransactionId="?" correlationId="?"
sessionId="?">
  <par:entity>
    <com:id> Party1</com:id>
    <com:lock>
      <com:key> Party1_KEY_2</com:key>
      <com:priority>10</com:priority>
      <com:timeout>800</com:timeout>
    </com:lock>
  </par:entity>
</ser:updatePartyRequest>
```

i Note: As Party1_KEY_1 is an immediate lock, and as Party1_KEY_1 is not removed by the owner, Party1_KEY_2 expires automatically after timeout (800 seconds). You get (Party1_KEY_2) lock expired notification on topic "com.tibco.inventory.notification.lock.expired.topic.tibco".

Batch Functionality

The Batch functionality on server side is the ability to group the commands under a common batch ID. The group of commands can be either committed as a whole or rolled back as a whole.

Batch Functionality



The feature works as follows on the SOAP interface:

- Call the StartBatch() to get a new batch with a batch ID.
- Modify (update, delete, or create) entities, sending the batch ID for each request. This attaches the entities to the batch.
- Either call completeBatch() with a commit or rollback parameter.
 - Commit removes the batch ID marker on entities.
 - Rollback removes the batch ID marker on entities and rollback changes on the entities (delete the ones created during the batch, recreate the ones deleted)

during the batch, or just get data back to its original value).

- Both delete the batch.

i **Note:** An entity that is attached to a batch cannot be modified outside of it. Trying to update or delete an entity outside of the batch results in an error.

Configuration Values

The following configuration values are located in the `$PSI_HOME/seed-data/app-properties/ConfigValues_PSIService.json` file:

Name	Description	Value
logging.config	Logging Configuration File Path	config/logback_psi.xml
datasourceDriverClassName	Pooled Data Source Driver Class Name	oracle.jdbc.driver.OracleDriver (Oracle) or org.postgresql.Driver (Postgres)
psiDsUsername	Pooled Data Source Username	psiusr210
com.tibco.fos.psi.hibernate.default_catalog	Hibernate Default Catalog	psiusr210
psiDsPassword	Pooled Data Source Password	psiusr210
psiDsUrl	Pooled Data	jdbc:oracle:thin:@//localhost:1521/orcl.a pac.tibco.com (Oracle) or

Name	Description	Value
	Source URL	jdbc:postgresql://localhost:5432/psidb?currentSchema=psischema (Postgres)
psiDsInitializeSize	Pooled Data Source Initialize Size	2
psiDsMaxIdle	Pooled Data Source Max Idle	11
psiDsMaxActive	Pooled Data Source Max Active	12
psiDsMaxWait	Pooled Data Source Max Wait	10000
datasourceValidationQuery	Pooled Data Source Validation Query	select 1 from dual or SELECT 1
psiDsTestOnBorrow	Pooled Data Source Test OnBorrow	false

Name	Description	Value
psiDsTestWhileIdle	Pooled Data Source Test WhileIdle	true
psiDsTimeBetweenEvictionRunsMillis	Pooled Data Source Eviction Interval	5000
psiDsMinEvictableIdleTimeMillis	Pooled Data Source Minimum Evictable Idle Time	5000
psiDsNumTestsPerEvictionRun	Pooled Data Source Tests Per Eviction Run	5
hibernateDialect	Hibernate dialect	org.hibernate.dialect.Oracle10gDialect (Oracle) or org.hibernate.dialect.PostgreSQLDialect (Postgres)
hbm.cache.use_second_level_cache	Hibernate Second Level Cache Usage	false

Name	Description	Value
hbm.cache.provider_class	Hibernate Cache Provider Class	org.hibernate.cache.NoCacheProvider
hbm.jdbc.batch_size	Hibernate JDBC Batch size	30
psiDsHibernateShowSql	Hibernate Show SQL	false
db.defaultAutoCommit	Default Automocommit for DB	false
db.rollbackOnReturn	Rollback on Return	false
db.commitOnReturn	Commit on Return	false
expiry.check.delay	Number of milliseconds between 2 expiry checks	60000
watchdog.period	Default Watchdog period	5000
notification.enable	Enable/Disable	false or true

Name	Description	Value
	sable notification generation in locking use cases.	
notification.provider	Notification service provider [none,ems]. It must be set to [ems] if notification generation is enabled	none or ems
emsServerURL	JNDI URL for JMS Service	tibjmsnaming://localhost:7222
emsServerUsername	JNDI Username	admin
emsServerPassword	JNDI Password	admin
jms.targetDestination	JMS Queue to process incoming SOAP API	com.tibco.inventory.soap.api.jms.queue

Name	Description	Value
	requests	
jms.concurrentConsumers	Initial count of consumers that process the incoming SOAP request	5
com.tibco.fos.psi.soapService.jms.maxConcurrentConsumers	Max count of consumers that process the incoming SOAP request	10
bulk.configuration	Bulk load configuration file path (on the inventory server machine).	/tmp/bulkconfig.json

Oracle RAC Configuration

Connecting to oracle RAC (Real Application Cluster) using JDBC with thin driver, the classic url: `jdbc:oracle:thin:@<HOST>:1521:<SID>` does not work.

The correct URL is as follows:

```
jdbc:oracle:thin:@(DESCRIPTION=(LOAD_BALANCE=on)
  (ADDRESS=(PROTOCOL=TCP) (HOST=host1) (PORT=1521))
  (ADDRESS=(PROTOCOL=TCP) (HOST=host2) (PORT=1521))
  (CONNECT_DATA=(SERVICE_NAME=service)))
```

The mentioned URL can be obtained from the `tnsnames.ora` file in the Oracle installation

Product and Service Inventory User Interface

You can access Product and Service Inventory by logging in to the application using the provided credentials.

The homepage or the main page of the application provides access to the entire Product and Service Inventory application.

Multitenancy in Product and Service Inventory

You can use the multitenancy feature of Product and Service Inventory to create a tenancy object, which provides user space per tenant.

You can create a new tenant from authorization service. Use POST method from authorization service to create new user and tenant.

The users within any tenant can be modified using PUT method on the authorization service.

See the Create User, Update User, and Delete User topics in [Authorization Service](#) for details on creation and modification of users in the respective tenant.

Basic Access Controls of Product and Service Inventory

The basic access controls of Product and Service Inventory includes:

Log In	To access Product and Service Inventory system using the provided credentials.
Main	Provides access to all the features provided by the Product and Service Inventory

Page	system.
Search Bar	Search Items and Parties by their name.
Log Out	To exit from the Product and Service Inventory system.

Logging in to Product and Service Inventory

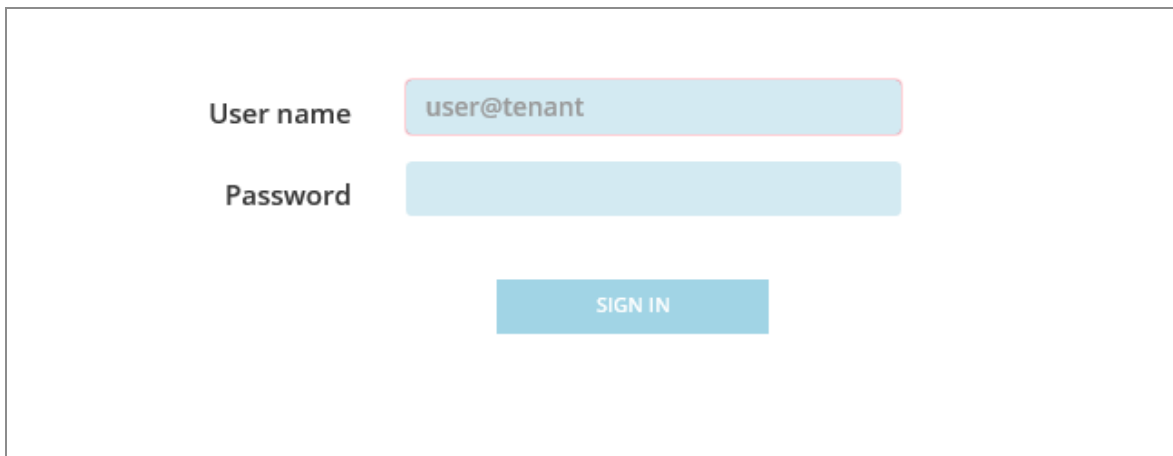
To log in to the Product and Service Inventory application, perform the following steps:

Procedure

1. Enter the following URL format in a browser: `http://<localhost:port_number>`. An example for the URL can be `http://<hostname>:8080`.

The Product and Service Inventory application opens displaying the login page.

Product and Service Inventory Login Page



The screenshot displays a login form with two input fields and a button. The 'User name' field contains the text 'user@tenant'. The 'Password' field is empty. Below the fields is a blue button labeled 'SIGN IN'.

2. Enter the **User name**.

The username format is `username@tenant_name`.

3. Enter the **Password**.

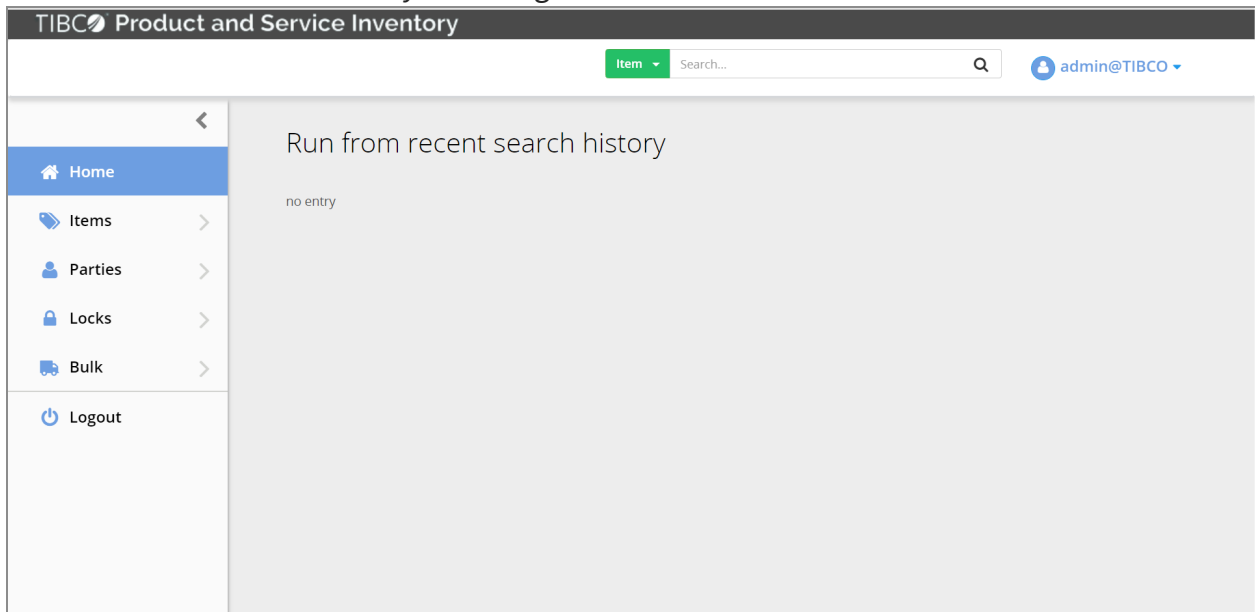
4. Click the **Sign In** button.

Upon successful verification of credentials, a session is established and the control is passed to the Product and Service Inventory main page.

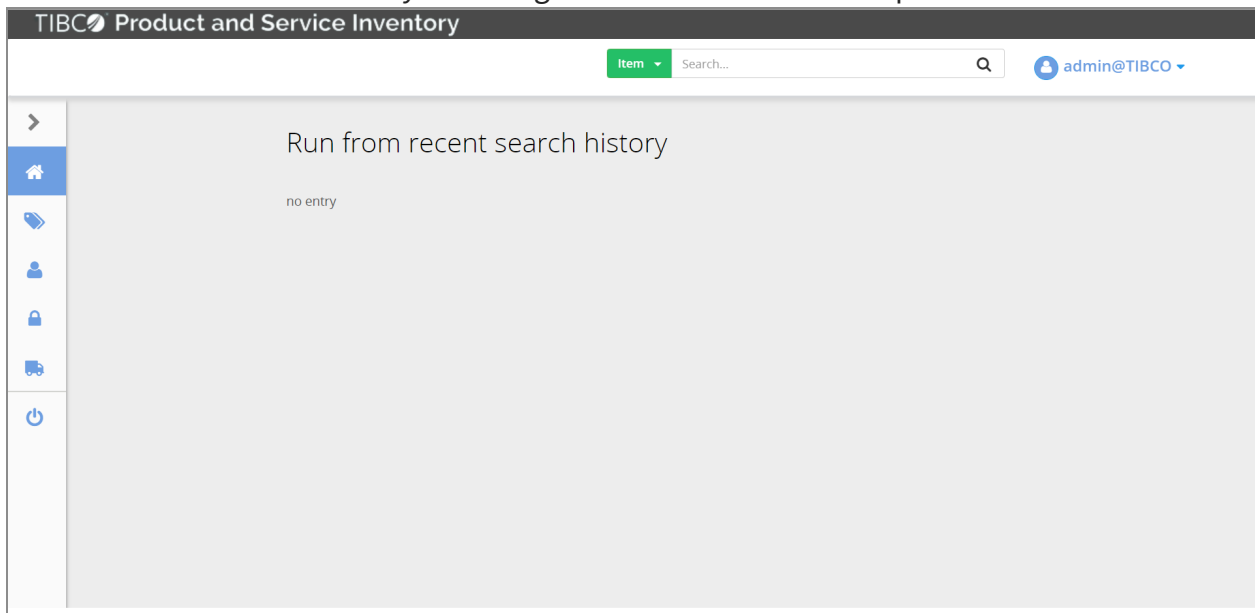
Product and Service Inventory Main Page

The collapsible menu on the left of the Product and Service Inventory main page provides access to all features of the application. The collapsible menu on the left side provides access to:

Product and Service Inventory Main Page



Product and Service Inventory Main Page with Menu Names Collapsed



Home	To pass the control back to the main page
Items	To view and add items
Parties	To view and add parties
Locks	To view the locked items and locked parties
Bulk	To perform Bulk Load operation
Logout	To log out from the Product and Service Inventory application

The top bar of the Product and Service Inventory application provides access to:

Search	To search a particular item or party
Profile Name	Displays the profile that is logged in.

Searching Items and Parties

To search for an item and party, perform the following steps:

Procedure

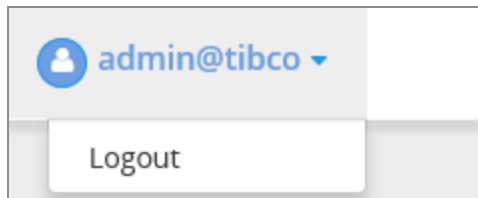
1. Select the value **Item Name** or **Party Name** from the dropdown box.
2. Enter the name of the:
 - item, if the selection is **Item Name**.
 - party if the selection is **Party Name**.
3. Click the search icon or press the Enter key.

The results are displayed on the page.

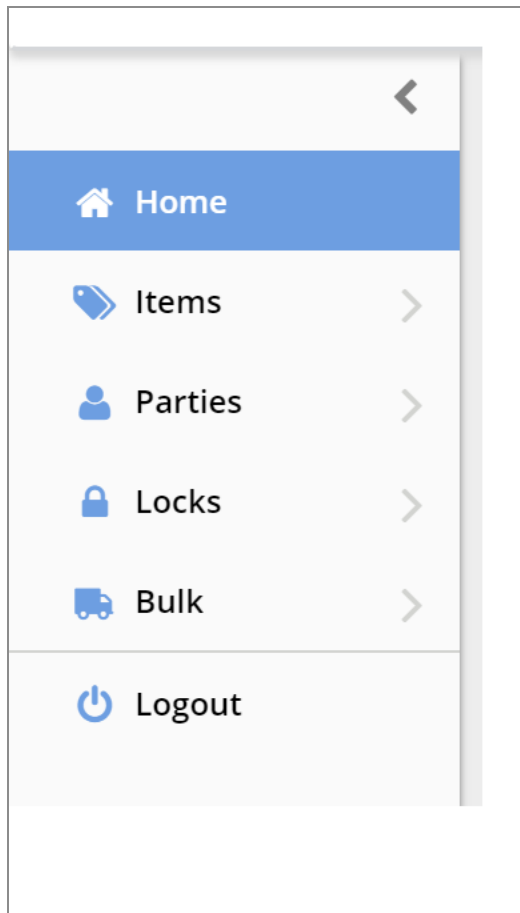
Logging out of Product and Service Inventory

Logging out of Product and Service Inventory ends the session created to access the application. To log out of Inventory, perform the following steps:

Product and Service Inventory Logout from Top Bar



Product and Service Inventory Logout from Menu



Procedure

1. Click the profile name
2. Click **Logout**.

Items

An Item represents any tangible or intangible product or service that has business context within any system that makes use of the Product and Service Inventory in the enterprise. This includes both OSS and BSS layers, as well as systems outside these domains. An item is identified with a unique identifier called an `itemId`. This identifier must be globally unique within the item domain and can be specified externally or generated by the Product and Service Inventory system as required.

Typically an item represents any one of the following types of entities:

- Customer-Facing Service (CFS)
- Resource-Facing Service (RFS)
- Bundle
- Product
- Device
- Service
- Tariff

i Note: The mentioned list is not an exhaustive list of item types. Within the Product and Service Inventory system, an item is a generic object which has a type and subtype classification scheme. Users are free to define their own categories within this scheme, with the one limitation that each item must have a globally unique `itemId`.

Product and Service Inventory also provides the ability to manage a limited set of information that describes an item in the form of characteristics. These characteristics are name-value pairs that describe attributes of the item. Each characteristic is identified with a unique identifier called a `characteristicId`. This identifier must be unique within the scope of any given item instance and can be specified externally or generated by the Product and Service Inventory system as required.

The system also provides the ability to link an item to a list of orders that have interacted with the item. This could be considered an order audit log of changes to the item over its life cycle. Each order is identified with a unique identifier called an `orderId`. This identifier must be unique within the scope of any given item instance and must always be specified externally. This means multiple items can refer to the same order, but the same order may not appear on one item more than once.

Items may have relationships to other items, so the Product and Service Inventory system provides the capability of linking items together using relationships. These relationships do not enforce referential integrity and are keyed on a combination of the relationship type and the primary key of both the parent and child item entities, specifically the `itemId`. These values must be unique for any given item and must be specified externally.

There is also the capability of locking item objects within the Product and Service Inventory system. Each lock is identified with a unique identifier called a key. This identifier must be unique within the scope of any given item instance and must always be specified externally.

Adding a New Item

To add a new item perform the following steps:

Procedure

1. Click the **Items** menu.
2. Click the **+** button.
3. Provide the primary information, which includes:
 - **Item Type** – Some examples for the values are Enterprise, Customer, or Subscriber. This is a mandatory field.
 - **Item Name** – Enter a name for the item.
4. Provide the additional information, which includes:
 - **Item Status** – Some examples for the values are Active, Inactive, Suspended, or Pending Update. This is a mandatory field.
 - **Item Sub Type**
 - **Item ID** – The unique ID for the item. If a value is not provided Product and Service Inventory generates the value.
 - **Item Ref**
 - **Product ID** – This is a mandatory field.
 - **Party ID** – This is a mandatory field.
 - **Product Version**

- **Start Date** – This is a mandatory field.
 - **End Date**
5. Provide the characteristics information (this is optional), by clicking **+ Add** link in the Characteristics Info section. You can add multiple characteristics by clicking **+ Add** link. You can delete unwanted characteristics by clicking **x Remove** link. The characteristics information to be provided are as follows:
- **Name** - This is a mandatory field.
 - **Value** - This is a mandatory field.
 - **ID**
6. Provide the orders information (this is optional), by clicking **+ Add** link in the Orders Info section. You can add multiple order information by clicking **+ Add** link. You can delete unwanted order information by clicking **x Remove** link. The order information to be provided is as follows:
- **Order ID** – This is a mandatory field.
 - **Ref** – This is a mandatory field.
 - **Date** – This is a mandatory field.
 - **Line** – This is a mandatory field.
 - **Line action** – This is a mandatory field.
 - **Action mode**
 - **Plan item ID**
 - **Plan item action**
 - **Comments**
7. Provide the relationships information (this is optional), by clicking **+ Add** link in the Orders Info section. You can add multiple relationship information by clicking **+ Add** link. You can delete unwanted relationship information by clicking **x Remove** link. The relationship information to be provided are as follows:
- **Child** – The target item to relate to. It has to exist in the system. This is a mandatory field.
 - **Forward type** – The name of the forward relationship ProductComprisedOf. This is a mandatory field.

- **Reverse type** – The reverse name of the relationship ProductContainedIn. This is a mandatory field.
8. Click the **Submit** button.

The item is created and the control is passed back to the Inventory Item page. The newly created Item is displayed in the list of Items. To search for the newly created item see [Searching an Item](#) for more details.

Searching an Item

You can search an Item using the common search bar. See Searching Items and Parties for more details. You can use the basic navigation buttons like Previous and Next to locate the desired item. You can also search for an item using the search feature. To search for an item from the Item menu perform the following steps:

Procedure

1. Enter the search query with any or all of the parameters:

- name
- id
- type
- partyId
- status

An example for a query format is as follows: status=<Item Status>> AND name=<<Item Name>> AND type=<<Item Type>>.

2. Click the **Search** button.

The criteria satisfying the search parameters are displayed.

3. Click **Save this Search?** Link to reuse the search query.
4. Enter a relevant name for the query and click the **Save** icon.

The query is saved for reuse.

5. Click **Home** menu.

The saved query name is displayed.

6. Click the query name.

The query details are displayed in the expanded view.

7. Click execute this Item query to execute the saved query again.

The result or results are displayed.

Updating an Item

To update an existing Item, perform the following steps:

Procedure

1. Click the **Items** menu.
2. Locate the desired item that needs to be updated using the **Previous** or **Next** button. You can also search the item using the Search feature. See [Searching an Item](#) for more details.
3. Click the item to be modified.
The Update Item page opens.
4. Make the necessary modifications to the desired fields and click the **Update** button.
The item is updated and the control is passed back to the Inventory Item page.

Locking an Existing Item

Locks can only be added to an existing item. To add a lock to an existing item perform the following steps:

Procedure

1. Click the **Items** menu.
2. Locate the desired item that needs to be locked using the **Previous** or **Next** button. You can also search the item using the Search feature. See [Searching an Item](#) for more details.
3. Click the item to be locked.
The Update Item page opens.
4. Provide values to the following fields:

- **Key** - Unique identifier of a lock on the item.
 - **Priority**
 - **Timeout** - The value entered must be in seconds.
5. Click the **Set** link.
 6. Click the **Update** button.

The item is locked and the control is passed back to the Inventory Item page.

Deleting the Lock of an Item

To delete the lock of an item, perform the following steps:

Procedure

1. Click the **Items** menu.
2. Locate the desired item, whose lock needs to be deleted, using the **Previous** or **Next** button. You can also search the item using the Search feature. See [Searching an Item](#) for more details.
3. Click the item.
The Update Item page opens.
4. Click the **Clear link** in the **List of locks** section.
5. Click the **Update** button.

The lock is deleted and the control is passed back to the Inventory Items page.

Deleting an Existing Item

To delete an existing item, perform the following steps:

Procedure

1. Click the **Items** menu.
2. Locate the desired item that needs to be deleted using the **Previous** or **Next** button. You can also search the item using the **Search** feature. See [Searching an Item](#) for

more details.

3. Click the item to be deleted.

The Update Item page opens.

4. Click the **Delete** button.

The item is deleted and the control is passed back to the Inventory Item page.

Parties

Party represents any user that has business context within any system that makes use of the Inventory in the enterprise.

It includes both OSS and BSS layers, as well as systems outside the domains. A party is identified with a unique identifier called a partyId. This identifier must be globally unique within the party domain and can be specified externally or generated by the Inventory as required. Typically a party is used to represent any one of the following types of entities:

- Customer
- Subscriber
- BAN
- BEN

i Note: The mentioned list is not an exhaustive list of party types. Within the Product and Service Inventory system, a party is a generic object which has a type and subtype classification scheme. Users are free to define their own categories within this scheme, and the only limitation is that each party must have a globally unique partyId.

Parties often exist in hierarchies, so the Product and Service Inventory system provides the capability of linking parties using parent child relationships. These relationships do not enforce referential integrity and are linked solely on the primary key of each party entity, specifically the partyId.

Product and Service Inventory also provides the ability to manage a limited set of information that describes a party in the form of characteristics. These characteristics are name-value pairs which describe attributes of the party. Each characteristic is identified with a unique identifier called a characteristicId. This identifier must be unique within the

scope of any given party instance and can be specified externally or generated by the Inventory as required.

There is also the capability of locking party objects within the Inventory. Each lock is identified with a unique identifier called a key. This identifier must be unique within the scope of any given party instance and must always be specified externally.

i Note: Product and Service Inventory is not a CRM system. Hierarchies and characteristics must be used minimally to support the image information stored only when it is absolutely necessary.

Adding a New Party

To add a new party, perform the following steps:

Procedure

1. Click **Parties**.
2. Click **+**.
3. Provide the primary information, which includes:
 - **Party Type** - The value can be Enterprise, Customer, or Subscriber. This is a mandatory field.
 - **Party Status** - The value can be Active, Inactive, Suspended, or Pending Update. This is a mandatory field.
4. Provide the additional information, which includes:
 - **Party Name**
 - **Party Sub Type**
 - **Party ID**
 - **Party Ref**
 - **Parent Party ID** – This may reference an existing partyID.
 - **Owned By** – If a value is not provided then the value is set to the user ID that created the party.
5. Provide the characteristics information (this is optional), by clicking **+ Add** link in the

Characteristics Info section. You can add multiple characteristics by clicking **+ Add** link. You can delete unwanted characteristics by clicking **x Remove** link. The characteristics information to be provided are as follows:

- **Name** - This is a mandatory field.
- **Value** - This is a mandatory field.
- **ID**

6. Click **Submit**.

The party is created and the control is passed back to the Inventory Party page. The newly created Party is displayed in the list of parties. For more information about searching for a party, see [Searching a Party](#).

Searching a Party

You can search a Party using the common search bar. You can use the basic navigation buttons like **Previous** and **Next** to locate the required item. You can also search for a party using the search feature. To search for a party from the Parties menu perform the following steps:

Procedure

1. Enter the search query with any or all of the parameters:

- name
- id
- type
- status

An example for a query format is as follows: status=<Party Status>> AND name=<<Party Name>> AND type=<<Party Type>>.

2. Click **Search**.

The criteria satisfying the search parameters is displayed.

3. Click **Save this Search?** link to reuse the search query.

4. Enter a relevant name for the query and click **Save**.

The query is saved for reuse.

5. Click **Home** menu.

The saved query name is displayed.

6. Click the query name.

The query details are displayed in the expanded view.

7. Click execute this Party query to execute the saved query again.

The result or results are displayed.

Updating a Party

To update a party, perform the following steps:

Procedure

1. Click **Parties**.
2. Locate the required party that needs to be updated using **Previous** or **Next**. You can also search the item using the Search feature. See [Searching a Party](#) for more details.
3. Click the party to be modified.
The Update Party page opens.
4. Make the necessary modifications to the required fields and click **Update**.
The party is updated and the control is passed back to the Inventory Party page.

Locking a Party

Locks can only be added to existing parties. To add a lock to an existing party perform the following steps:

Procedure

1. Click **Parties**.
2. Locate the required party that needs to be locked using **Previous** or **Next**. You can also search the item using the Search feature. See [Searching a Party](#) for more details.
3. Click the party to be modified.

4. Provide values to the following fields:

- **Key**
- **Priority**
- **Timeout**

5. Click the **Set** link.

6. Click **Update** .

The party is locked and the control is passed back to the Inventory Party page.

Deleting the Lock of a Party

To delete the lock of a party, perform the following steps:

Procedure

1. Click **Parties**.
2. Click the Locked parties link to expand it.
3. Click the required party that is locked.

The Update Party page opens and you can view the lock details.

4. Click the **Clear** link in the **List of locks** section.
5. Click **Update**.

The lock is deleted and the control is passed back to the Inventory Parties page.

Deleting a Party

To delete an existing party, perform the following steps:

Procedure

1. Click **Parties**.
2. Locate the required party that has to be deleted using **Previous** or **Next**. You can also search the party using the Search feature. See [Searching a Party](#) for more details.

3. Click the party to be deleted.

The Update Party page opens.

4. Click **Delete**.

The party is deleted and the control is passed back to the Inventory Party page.

Locks

Party and Item entities have the capability for locking and lock management.

This is useful in a context where multiple concurrent processes are all working on the same object but it is preferable to control the order of access to the object in a defined sequence.

Viewing Locked Items

To view locked items, perform the following steps:

Procedure

1. Click the **Locks** menu.
2. Click the **Locked items** link to expand it.
3. Click the desired item that is locked.

The Update Item page opens and you can view the lock details.

Viewing Locked Parties

To view locked parties, perform the following steps:

Procedure

1. Click **Locks** menu.
2. Click the **Locked parties** link to expand it.
3. Click the desired party that is locked.

The Update Party page opens and you can view the lock details.

Bulk Load

Bulk Load refers to mass importing of data. You can use Bulk Load to migrate data from an existing system to Product and Service Inventory.

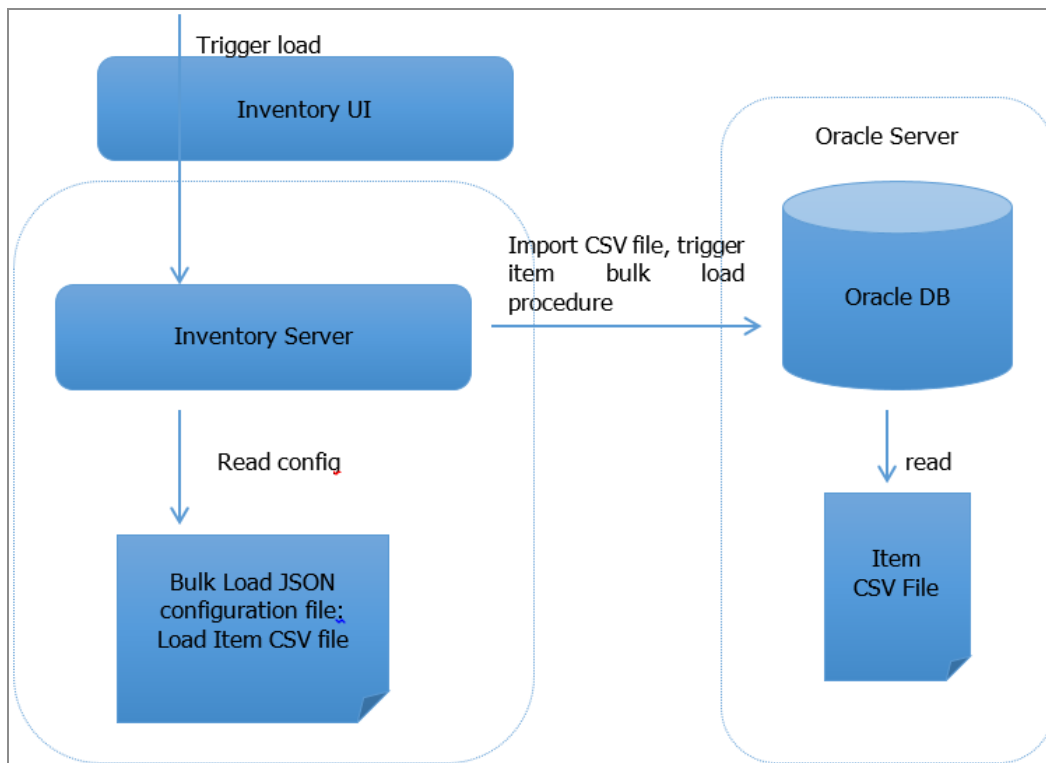
You must ensure that the data is in CSV format as the Oracle database reads the CSV files and treats it as a database table.

The Inventory system automatically applies stored procedures on the data and creates internal items, parties, characteristics, relationships, or orders. See the sections "Bulk Load for PostgreSQL Database" and "Bulk Load for Oracle Database" of the *TIBCO Product and Service Inventory Installation and Configuration Guide* for more information.

Bulk Load Process

The following diagram describes the Bulk Load process:

Bulk Load Flow Diagram



i Note: Bulk Load must be performed on an empty database, as it can cause a duplicate error, if the database contains IDs that are also present in the data loaded from CSV.

To bulk load items into the Product and Service Inventory system, you have to:

- Create a Bulk load JSON configuration file on the inventory server machine.
- Create an Item CSV File on the Oracle server machine that can be accessed from Oracle.
- Trigger a load from the Inventory user interface.

The Inventory Server reads the JSON configuration file. The JSON configuration file contains information about the CSV files that must be loaded. The Inventory Server then triggers the read of CSV file by the Oracle server, and also triggers the conversion of the CSV file data into proper Inventory items. The Logs are generated in the Oracle database and on the oracle server file system. These logs are accessible from the Inventory user interface bulk load pages.

Oracle Bulk Load Setup

You must define a directory accessible by the Oracle database. You need to execute the following command:

```
create or replace directory PSI_BULK_DIR as '[existing directory
absolute path on oracle server]';
grant read, write on directory PSI_BULK_DIR to [inventory oracle user];
```

You need to create the CSV file in the defined directory. The directory and the file must have read or write privileges for the Oracle process user.

Bulk Load JSON Configuration

The Bulk Load JSON configuration drives the bulk load process. You get an indication for various load configurations, location of the CSV files, field names in the CSV files, and so on.

The configuration contains a main section defining the following parameters:

Parameters

oracleDirectoryName	<p>For Oracle database:</p> <p>Logical directory name for oracle. The name is PSI_BULK_DIR.</p> <p>For PostgreSQL database:</p> <p>This is the absolute directory path where the data files containing the data are to be loaded.</p>
timeStampFormat	<p>Timestamp format that is used by the CSV exported data. Timestamps follow the oracle notation. Sample format is "DD-MON-YYYY HH12.MI.SS PM". Example value: "01-JAN-2014 05.40.12 PM"</p>
commitSize	<p>For Oracle database:</p> <p>Frequency of commits when performing creation.</p>

For PostgreSQL database:

Unlike Oracle database, bulk load does not use this parameter. Keep the default.

entities	List of entities to load.
----------	---------------------------

Each entity to load is a configuration object containing:

type	The type of entity to load. Options are:
------	--

- PARTY
- PARTY_CHARACTERISTIC
- ITEM
- ITEM_CHARACTERISTIC
- ITEM_RELATIONSHIP
- ITEM_ORDER
- ITEM_ORDER_COMMENT

dataSourceName	The name of the CSV file relative to the inside of the PSI_BULK_DIR directory
----------------	---

dataSourceColumnList	The names of the columns in the CSV file. The possible values depend on the type of entity to load. Some columns are mandatory, some are optional.
----------------------	--

Entity Type	Mandatory Columns	Optional Columns
PARTY	ID(VARCHAR2), PARTY_TYPE(VARCHAR2), STATUS(VARCHAR2), OWNED_BY (VARCHAR2), CREATED_BY(VARCHAR2), CREATED_ON(TIMESTAMP(6)), VERSION (NUMBER), DELETED(NUMBER) (0 for not deleted, 1 for deleted)	PARTY_REF(VARCHAR2), NAME(VARCHAR2), SUB_TYPE(VARCHAR2), PARENT_PARTY_ID (VARCHAR2), UPDATED_BY(VARCHAR2), UPDATED_ON(TIMESTAMP

Entity Type	Mandatory Columns	Optional Columns
		(6))
PARTY_ CHARACTERISTIC	ID(VARCHAR2), PARTY_ID(VARCHAR2) (the party to relate to), CHARACTERISTIC_NAME (VARCHAR2), CHARACTERISTIC_VALUE (VARCHAR2),	
ITEM	ID(VARCHAR2), PRODUCT_ID(VARCHAR2), ITEM_TYPE(VARCHAR2), STATUS (VARCHAR2), PARTY_ID(VARCHAR2), START_DATE(TIMESTAMP(6)), OWNED_BY (VARCHAR2), CREATED_BY(VARCHAR2), CREATED_ON(TIMESTAMP(6)), VERSION (NUMBER), DELETED(NUMBER) (0 for not deleted, 1 for deleted)	ITEM_REF(VARCHAR2), PRODUCT_VERSION (VARCHAR2), NAME (VARCHAR2), SUB_TYPE (VARCHAR2), END_DATE (TIMESTAMP(6)), UPDATED_BY(VARCHAR2), UPDATED_ON(TIMESTAMP (6))
ITEM_ CHARACTERISTIC	ID(VARCHAR2), ITEM_ID(VARCHAR2) (the item to relate to), CHARACTERISTIC_NAME (VARCHAR2), CHARACTERISTIC_VALUE (VARCHAR2)	
ITEM_ RELATIONSHIP	ID(VARCHAR2), ITEM_ID(VARCHAR2) (the item to relate to), FORWARD_TYPE (VARCHAR2), REVERSE_TYPE(VARCHAR2), CHILD_ITEM_ID(NUMBER) (the child item to relate to),	
ITEM_ORDER	ITEM_ID(VARCHAR2) (the item to relate to), ID(VARCHAR2), ORDER_REF(VARCHAR2), ORDER_DATE(TIMESTAMP(6)), LINE_NUMBER(VARCHAR2), LINE_ACTION (VARCHAR2)	LINE_ACTION_MODE (VARCHAR2), PLAN_ITEM_ID(VARCHAR2), PLAN_ITEM_ACTION(VARCHAR2)
ITEM_ORDER_ COMMENTS	ORDER_ID, COMMENT_DETAIL	

Bulk Load Log Files

The log files for the bulk load are created in the same directory where the bulk load files are placed.

There are four types of log files created for each data input file. The first three files are created by Oracle while creating the external table and are known as the log, bad, and discard files. The last file is created by the procedure which moves the records from the external table to the Oracle table.

Log Files

Log File	<p>For Oracle database:</p> <p>This file is used by Oracle to log information about the process used to create the external table. The log file name has the following format, Log_<EntityName>_<LoadId>.log.</p>
Bad File	<p>For Oracle database:</p> <p>This file contains records that cannot be loaded because of errors. The Bad file name has the following format, Bad_<EntityName>_<LoadId>.bad</p>
Discard File	<p>For Oracle database:</p> <p>The file contains records that fail the condition in the LOAD WHEN clause of the statement used to create the external table. This file is created only if any records are encountered that fail the LOAD WHEN condition. The Discard file name has the following format, Discard_<EntityName>_<LoadId>.discard.</p>
Stored Procedure Logs	<p>For Oracle database: This file is created by the procedures which copy the records from the external table to the actual table. This file displays any errors encountered during the copying process. It also displays the current status of the load. The file name has the following format, Log_SP_<EntityName>_<LoadId>.log.</p> <p>For PostgreSQL database:</p> <p>This file is created by the procedures which copy the records from the external table to the actual table. The file displays the errors encountered during the copying process. It also displays the current status of the load. The file has the following format, Log_SP_<EntityName>_<LoadId>.log.</p>

Configurator UI

With the Configurator UI, you can upload, update, edit, and remove various application properties and configuration values.

Navigating the Configurator UI

Access to the dashboard is controlled through a basic username and password authentication.

To access the Configurator UI from a browser window, perform the following steps:

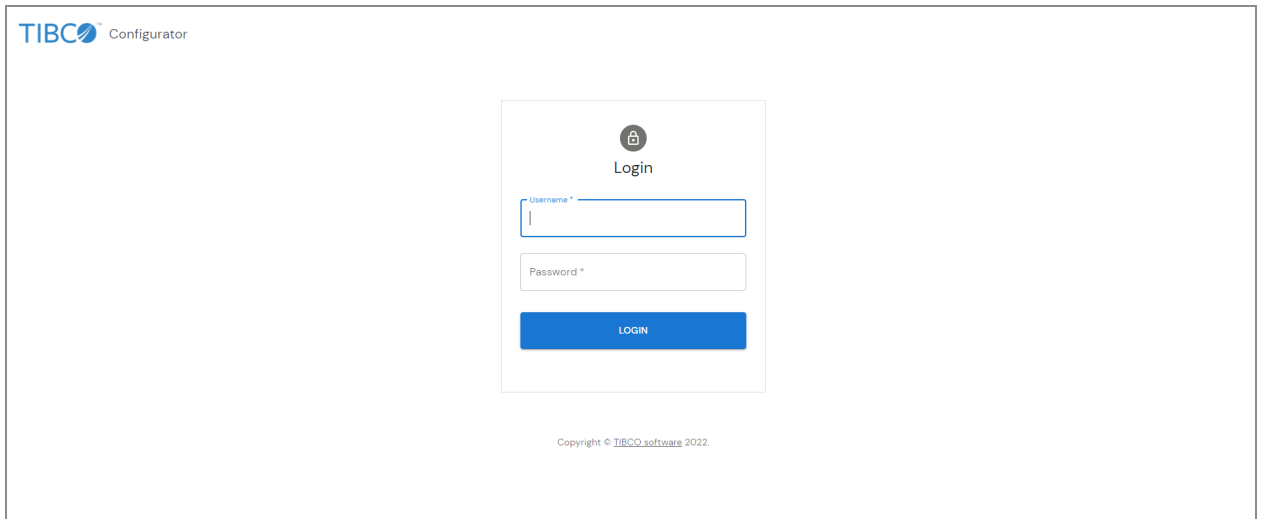
i **Note:** All the latest versions of Google Chrome, Mozilla Firefox, and Internet Explorer are supported by the Configurator UI.

i **Note:** To start the configurator UI, the configurator and authorization service must be up and running. Also, update the `$PSI_HOME/roles/configurator-ui/standalone/config/application.properties` file for `configuratorServiceUrl` and `authorizationServiceTokenEndPoint` values.

Procedure

1. Go to `http://<host>:<port number>` URL to access the **Login** page, where:
 - `host` is the computer where you have started configurator UI service
 - `port` is the port number of the machine where configurator UI listens for the

requests. The default port number is 9104.

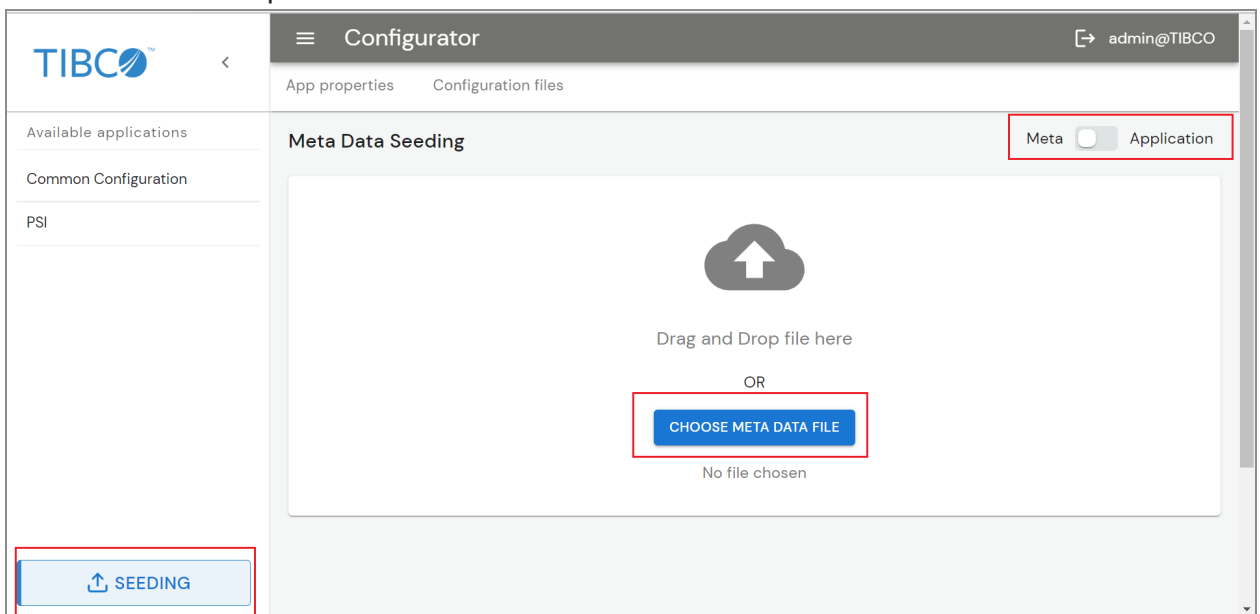
The image shows the TIBCO Configurator login interface. At the top left is the TIBCO logo and the word "Configurator". In the center is a login box with a lock icon and the word "Login". Below this are two input fields: "Username *" and "Password *". A blue "LOGIN" button is at the bottom of the login box. At the bottom center of the page, there is a small copyright notice: "Copyright © TIBCO software 2022."

2. Enter the credentials in the **Username** and **Password** fields and then click **LOGIN** to sign in.
The Configurator UI dashboard opens.

Uploading Metadata File

Procedure

1. Click **SEEDING** to upload the metadata file.

The image shows the TIBCO Configurator dashboard. On the left is a sidebar with the TIBCO logo and a menu with items: "Available applications", "Common Configuration", and "PSI". The main area has a header "Configurator" with a user icon and "admin@TIBCO". Below the header are tabs for "App properties" and "Configuration files". The "Configuration files" tab is active, showing a "Meta Data Seeding" section. In the top right of this section are two radio buttons: "Meta" (selected) and "Application". The main content area has a large cloud icon with an upward arrow, the text "Drag and Drop file here", and "OR" below it. A blue button labeled "CHOOSE META DATA FILE" is highlighted with a red box. Below the button is the text "No file chosen". In the bottom left corner of the dashboard, a button labeled "SEEDING" with an upward arrow icon is also highlighted with a red box.

2. Switch the radio button to select **Meta** on the top right-hand side.
3. To upload the \$PSI_HOME/seed-data/application_metadata.json file, use the drag-and-drop function or click **CHOOSE META DATA FILE**.

Configuring an Application

When you log in for the first time, an empty dashboard opens without any applications. It means the app_properties and configuration tables are blank in the database.

Before you begin

Before configuring an application, you must upload the metadata file.

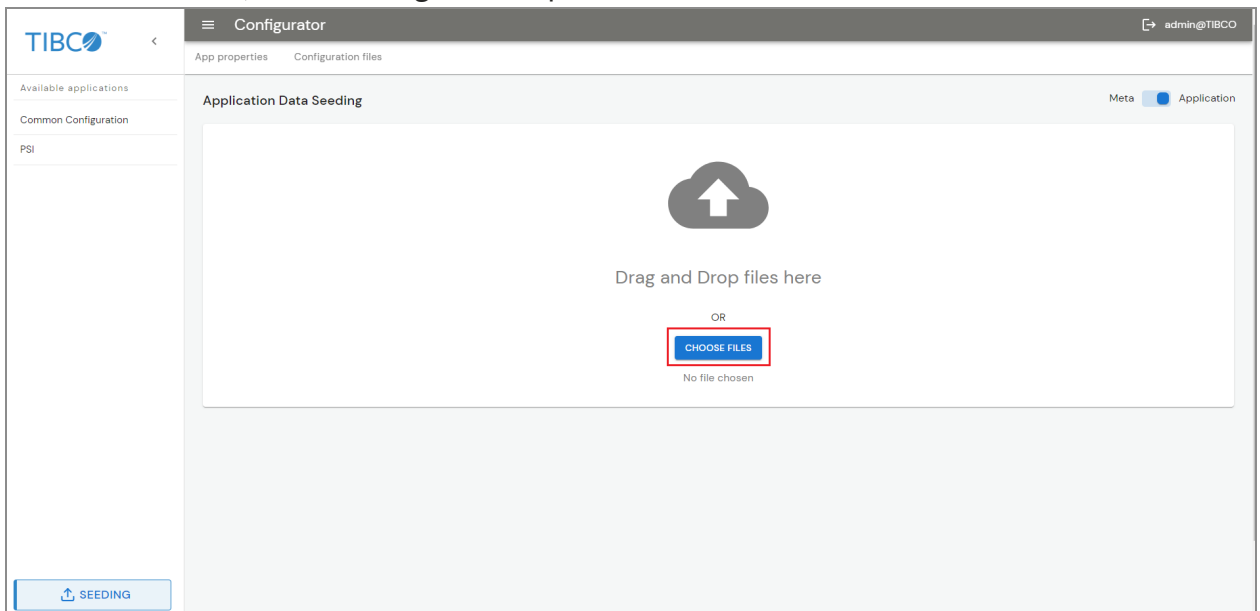
Procedure

1. Click **SEEDING** to upload the user's data.

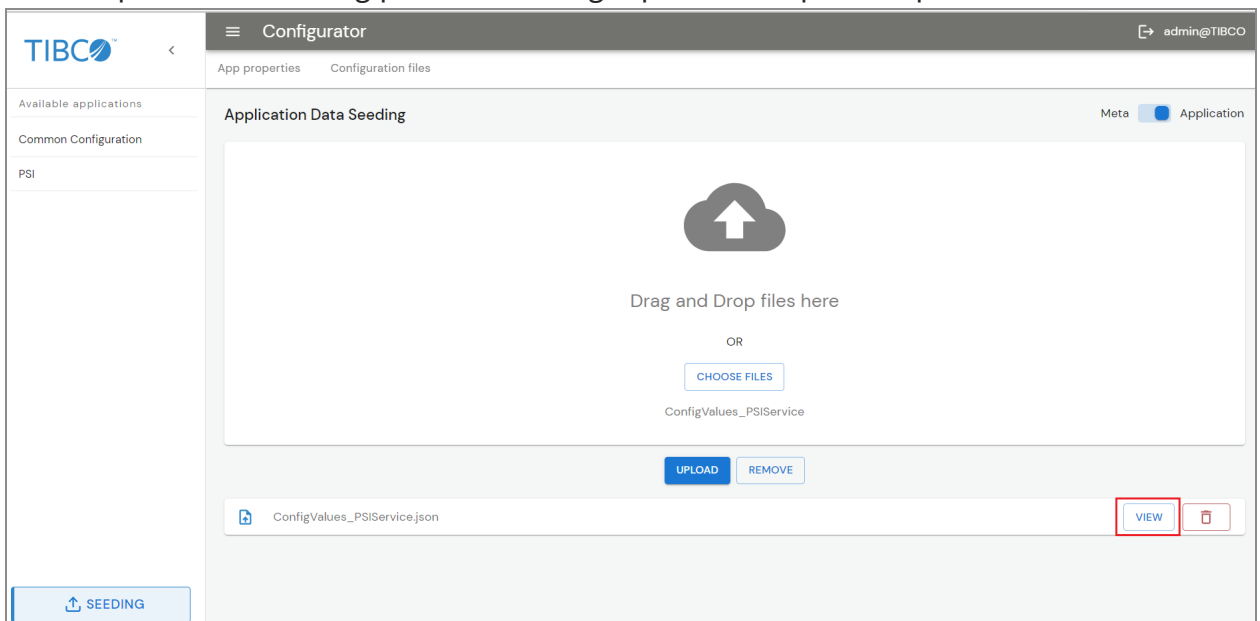
The screenshot shows the TIBCO Configurator UI. The top navigation bar includes the TIBCO logo, a menu icon, the title 'Configurator', and a user profile 'admin@TIBCO'. Below the navigation bar, there are tabs for 'App properties' and 'Configuration files'. The main content area is divided into a left sidebar and a main panel. The sidebar has a section 'Available applications' with a sub-section 'Common Configuration' and a list of categories: 'PSI', 'Authorization Server Configuration Properties Used for Swagger UI', 'General Configurations', 'Resource and Service Level Metrics', and 'Security Configurations'. The main panel displays the 'Common Configuration' section, which includes a search bar, a table of configuration properties, and an 'EDIT' button. The table has columns for 'Property name', 'Value', and 'Description'. The 'SEEDING' button is located in the bottom left corner of the main panel, highlighted with a red box.

Property name	Value	Description
authentication.token.signing.key	ENC(nSaOk6lmjPPN8ZA5SO68pQ==)	Authentication Token Signing Key
authorization.client.id	order-management-client	Authorized Client ID
authorization.client.secret	ENC(ggsmFvh5HBbeSDlj+I5YOrP4qvOrJvEm)	Authorized Client Secret
authorizationServiceTokenEndPoint	http://10.69.92.110:9091	Authorization Server OAuth URL
useAccessTokenFromProperty	false	Use Static Access-Token

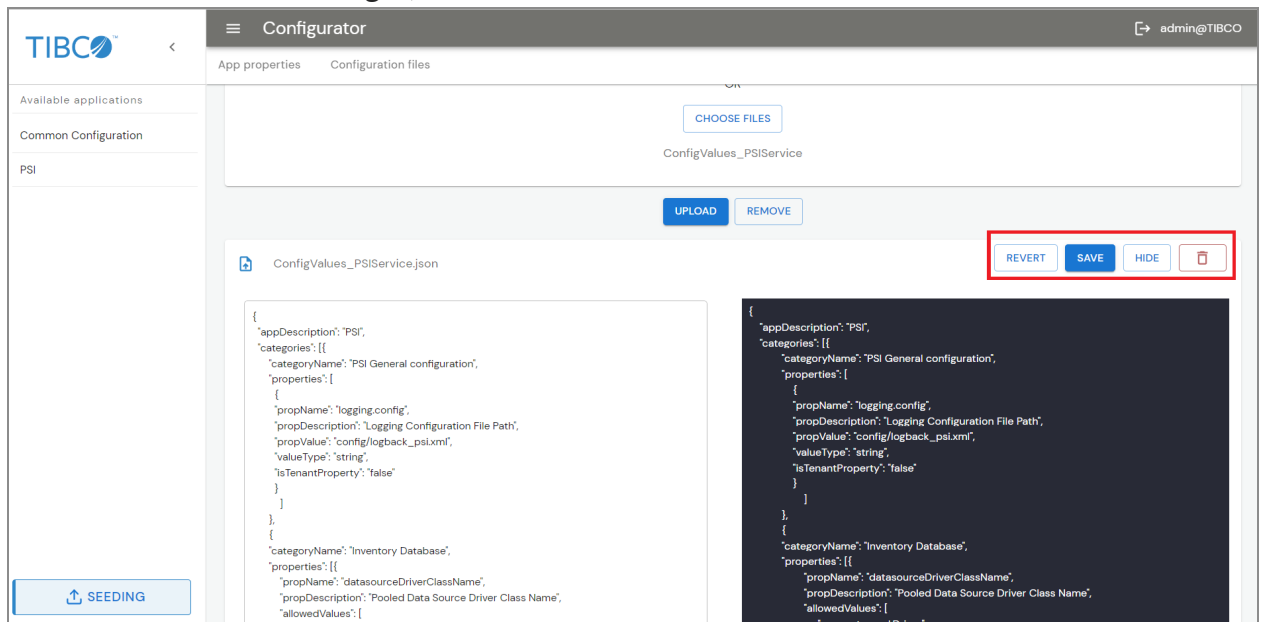
2. To select the files, use the drag-and-drop function or click **CHOOSE FILES**.



3. To view an application property file, click **VIEW**. The file opens in text editor mode. The left pane is the editing pane and the right pane is the preview pane.



4. To revert or save the changes, click **REVERT** or **SAVE**.



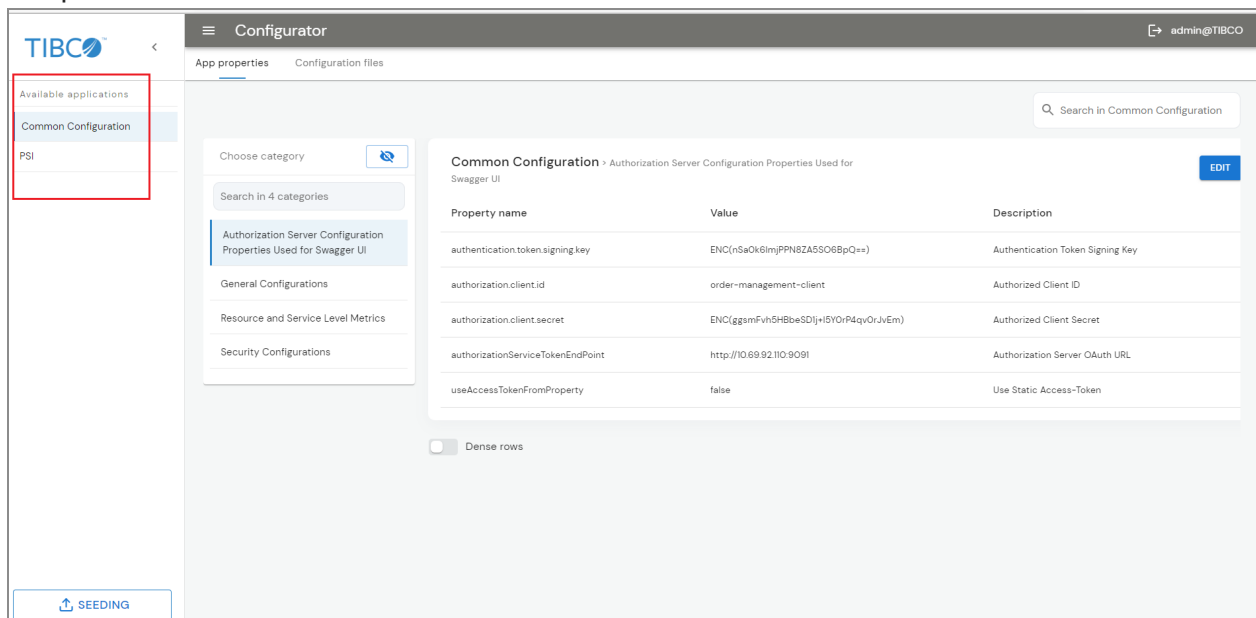
5. To hide or delete the files, click **HIDE** or **DELETE**.

6. Finally, click **UPLOAD** to upload the files.

You can also use the **UPLOAD ALL** or **CLEAR ALL** option to upload or remove all the files at once.

Result

Once the files are uploaded successfully, the available applications are displayed on the left pane.

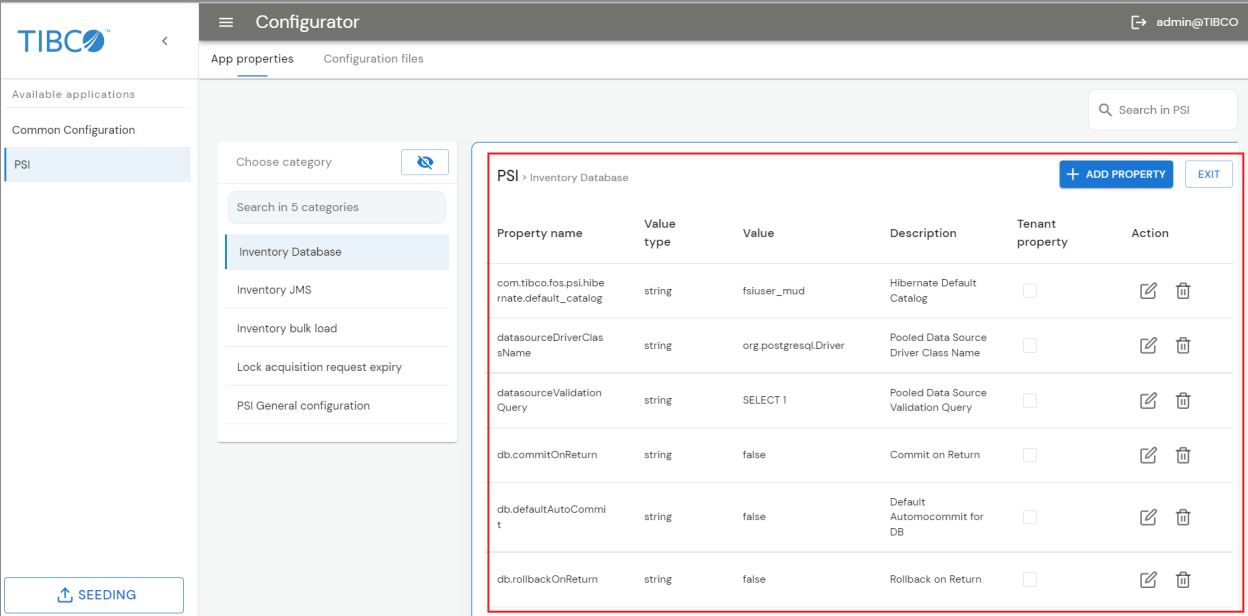


Note: If the files are uploaded from the Configurator swagger, run the refresh management function on the Configurator server's endpoint (http://<host>:<configurator_port>/management/refresh) to reflect the properties in the Configurator UI. Once this is done, log in again to the configurator UI.

Editing Application Properties

Procedure

1. Select the required application from the left pane, choose an App properties category row, and click **EDIT**.




The screenshot shows the TIBCO Configurator UI. On the left, the 'PSI' category is selected under 'Common Configuration'. The main area shows the 'Inventory Database' category selected. The 'App properties' tab is active, and a table of properties is displayed. The table is highlighted with a red border.

Property name	Value type	Value	Description	Tenant property	Action
com.tibco.fos.psi.hibernate.default_catalog	string	fsuser_mud	Hibernate Default Catalog	<input type="checkbox"/>	
datasourceDriverClassName	string	org.postgresql.Driver	Pooled Data Source Driver Class Name	<input type="checkbox"/>	
datasourceValidationQuery	string	SELECT 1	Pooled Data Source Validation Query	<input type="checkbox"/>	
db.commitOnReturn	string	false	Commit on Return	<input type="checkbox"/>	
db.defaultAutoCommit	string	false	Default Automocommit for DB	<input type="checkbox"/>	
db.rollbackOnReturn	string	false	Rollback on Return	<input type="checkbox"/>	

The application properties file for that category opens in an editing pane.

2. To add new properties, click **ADD PROPERTY**, fill the details in the **Property name**, **Value**, and **Description** fields and select a **Value type** from the drop-down option.

Then click the save icon .

If the property is tenant-specific, you can select the **Tenant property** check box.

The screenshot shows the TIBCO Configurator UI. On the left, the 'PSI' application is selected under 'Common Configuration'. The main area displays the 'Inventory Database' configuration. A table lists properties with columns: Property name, Value type, Value, Description, Tenant property, and Action. The 'Tenant property' checkbox for the first row is highlighted with a red box. The 'ADD PROPERTY' button is also highlighted with a red box.

Property name	Value type	Value	Description	Tenant property	Action
com.tibco.fos.psl.hibernate.default_catalog	String	fsuser_mud	Hibernate Default Catalog	<input type="checkbox"/>	
datasourceDriverClassName	string	org.postgresql.Driver	Pooled Data Source Driver Class Name	<input type="checkbox"/>	
datasourceValidationQuery	string	SELECT 1	Pooled Data Source Validation Query	<input type="checkbox"/>	
db.commitOnReturn	string	false	Commit on Return	<input type="checkbox"/>	
db.defaultAutoCommit	string	false	Default Automocommit for DB	<input type="checkbox"/>	

3. Click the edit icon or delete icon to edit or delete application properties.

The screenshot shows the TIBCO Configurator UI. On the left, the 'PSI' application is selected under 'Common Configuration'. The main area displays the 'Inventory Database' configuration. A table lists properties with their values and descriptions. The 'edit icon' and 'delete icon' in the 'Action' column for the second row are highlighted with a red box.

Property name	Value type	Value	Description	Tenant property	Action
com.tibco.fos.psl.hibernate.default_catalog	String	fsuser_mud	Hibernate Default Catalog	<input type="checkbox"/>	
datasourceDriverClassName	string	org.postgresql.Driver	Pooled Data Source Driver Class Name	<input type="checkbox"/>	
datasourceValidationQuery	string	SELECT 1	Pooled Data Source Validation Query	<input type="checkbox"/>	
db.commitOnReturn	string	false	Commit on Return	<input type="checkbox"/>	
db.defaultAutoCommit	string	false	Default Automocommit for DB	<input type="checkbox"/>	
db.rollbackOnReturn	string	false	Rollback on Return	<input type="checkbox"/>	

You can restore the previous property value by clicking on the restore icon .

Note: You can only edit the value and value type of a property.

4. Once editing is completed, click **CANCEL**.

Replicating Application Properties

Procedure

1. Log in to the Configurator UI as a non-default tenant.
2. Select the required application from the left pane, choose an App properties category row, and click **EDIT**.
3. In the **Tenant Replication** window, enter the **Source TenantID** and click **REPLICATE**.

The screenshot shows the TIBCO Configurator UI. On the left, under 'Available applications', the 'PSI' category is selected. Below it, the 'Tenant Replication' window is open, showing a 'Source TenantID' input field and a 'REPLICATE' button. The main area displays the 'Inventory Database' properties for the 'PSI' application. The properties are listed in a table with columns for Property name, Value, and Description.

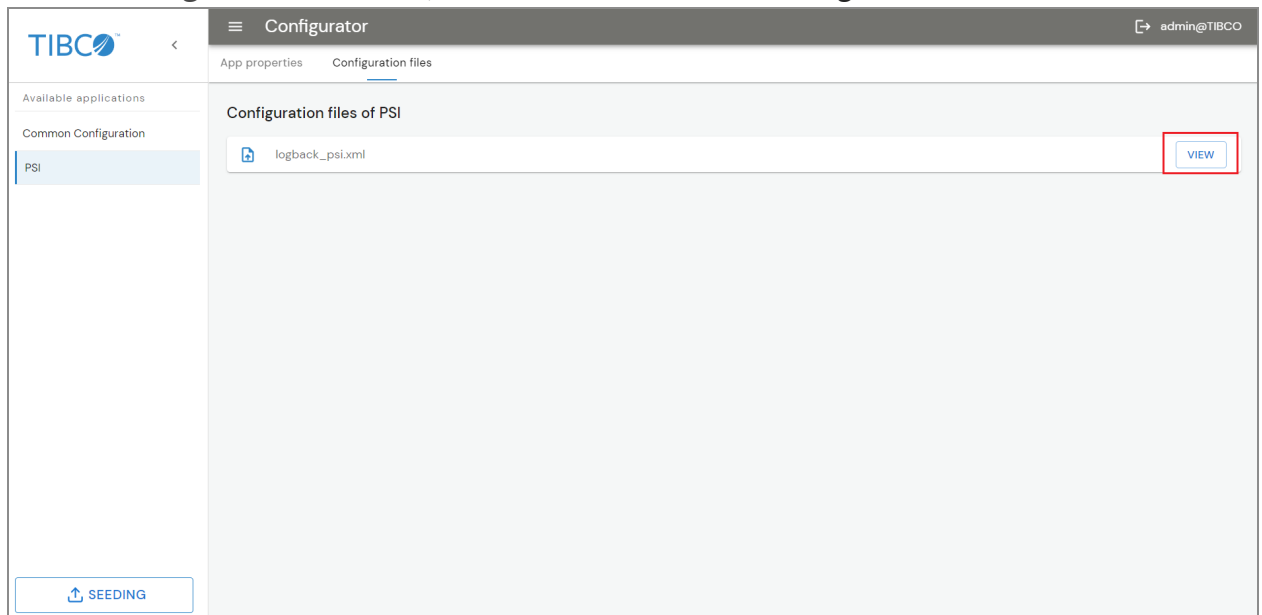
Property name	Value	Description
com.tibco.fos.psi.hibernate.default_catalog	fsluser_mud	Hibernate Default Catalog
datasourceDriverClassName	org.postgresql.Driver	Pooled Data Source Driver Class Name
datasourceValidationQuery	SELECT 1	Pooled Data Source Validation Query
db.commitOnReturn	false	Commit on Return
db.defaultAutoCommit	false	Default Automocommit for DB
db.rollbackOnReturn	false	Rollback on Return
hbm.cache.provider_class	org.hibernate.cache.NoCacheProvider	Hibernate Cache Provider Class
hbm.cache.use_second_level_cache	false	Hibernate Second Level Cache Usage
hbm.jdbc.batch_size	30	Hibernate JDBC Batch size

Note: When you log in with the default tenant or any other tenant with no data on the database, the data seeding option is enabled. When you log in with a non-default tenant and the database is not empty, the tenant replicate option is visible.

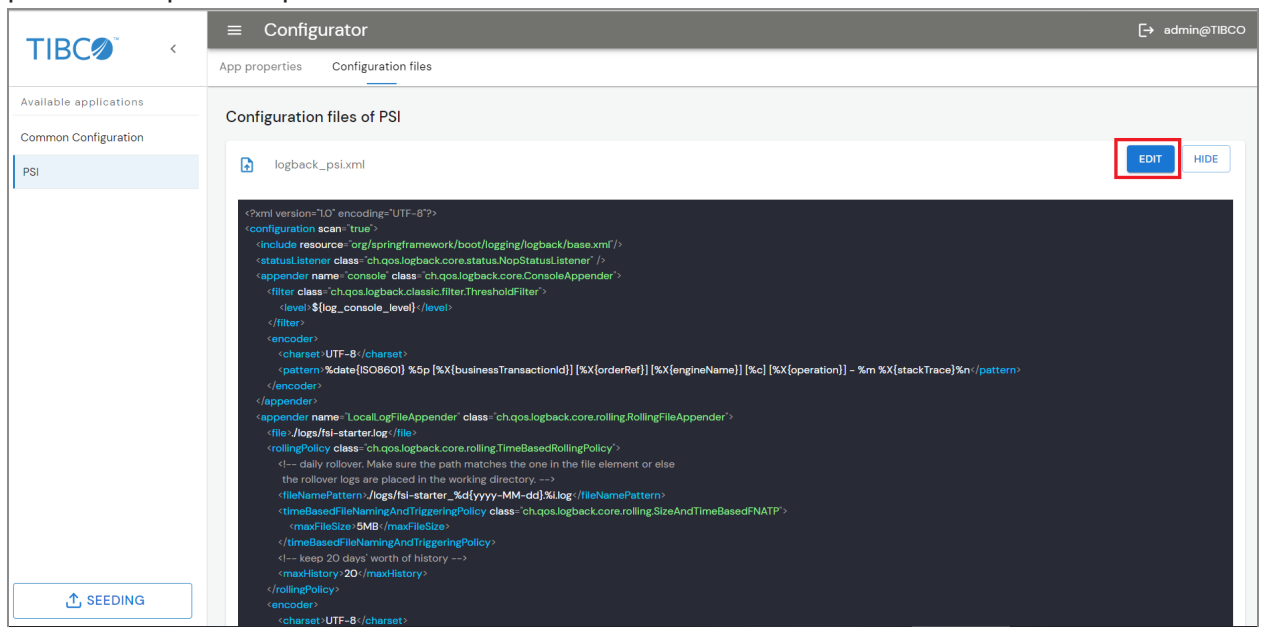
Editing Configuration Files

Procedure

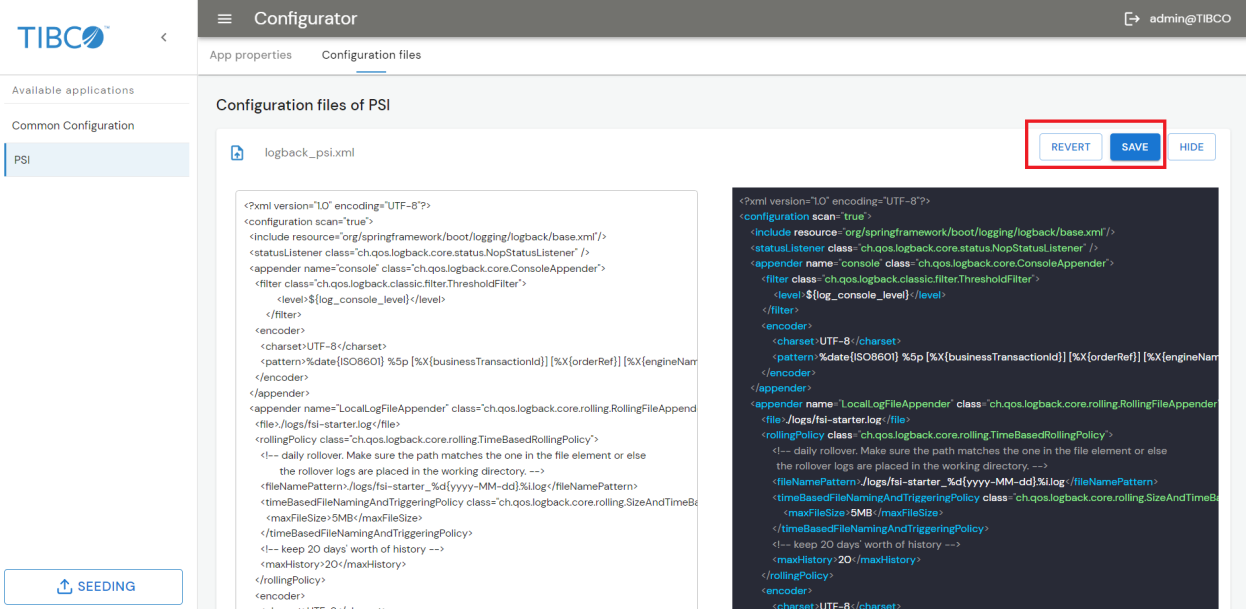
1. On the **Configuration files** tab, click **SEEDING** to add a configuration file.



2. To view a configuration file, click **VIEW** next to the respective configuration file.
3. To edit a configuration file, click **EDIT**.
The file opens in text editor mode. The left pane is the editing pane and the right pane is the preview pane.



4. To discard or save the changes, click **REVERT** or **SAVE**.



The screenshot displays the TIBCO Configurator UI. On the left, a sidebar shows 'Available applications' and 'Common Configuration' sections. The 'Common Configuration' section is expanded, showing 'PSI' as the selected application. The main area, titled 'Configuration files of PSI', lists several configuration files. The file 'logback_psi.xml' is selected, and its content is displayed in a split view. The left pane shows the original XML content, and the right pane shows the modified content. At the top right of the right pane, there are three buttons: 'REVERT', 'SAVE', and 'HIDE'. The 'SAVE' button is highlighted with a red box. The XML content in the right pane shows changes to the logging configuration, including the addition of a 'LocalLogFileAppender' and a 'TimeBasedRollingPolicy'.

Configuration Service

This REST service is used for various configurations of files such as get, add, update, and delete.

- [Get supported files metadata](#)
- [Save supported files metadata](#)
- [Delete supported files metadata](#)
- [Get Application Properties by Application ID](#)
- [Update Application Properties for Application ID](#)
- [Save Application Properties for Application ID](#)
- [Delete Application Properties for Application ID](#)
- [Replicate Tenant Properties](#)
- [Get list of applications available with Configurator](#)
- [Download Configuration File for Application ID](#)
- [Upload Configuration File for Application ID](#)
- [Configurator notification API](#)

Get supported files metadata

Method: HTTP GET

Endpoint: *http://<host_address>:<port_number>/v1/configuration/configFilesMetadata*

Parameters: No parameters

Save supported files metadata

Method: HTTP POST

Endpoint: *http://<host_address>:<port_number>/v1/configuration/configFilesMetadata*

Parameters: applicationId, applicationDescription, appProperties, and configurationFiles in the request body.

Delete supported files metadata

Method: HTTP DELETE

Endpoint: *http://<host_address>:<port_number>/v1/configuration/configFilesMetadata/{applicationId}*

Parameters: applicationid (Select the value from the drop-down list)

Get Application Properties by Application ID

Method: HTTP GET

Endpoint: *http://<host_address>:<port_number>/v1/configuration/{applicationid}*

Parameters: applicationid (Select the value from the drop-down list)

Update Application Properties for Application ID

Method: HTTP PUT

Endpoint: *http://<host_address>:<port_number>/v1/configuration/{applicationid}*

Parameters: applicationid (Select the value from the drop-down list)

Save Application Properties for Application ID

Method: HTTP POST

Endpoint: *http://<host_address>:<port_number>/v1/configuration/{applicationid}*

Parameters: applicationid (Select the value from the drop-down list)

Delete Application Properties for Application ID

Method: HTTP DELETE

Endpoint: *http://<host_address>:<port_number>/v1/configuration/{applicationid}*

Parameters: applicationid (Select value from the drop-down)

Replicate Tenant Properties

This API is used to replicate the tenant-specific properties through the REST service. Previously, properties could only be replicated through the Configurator UI.

Method: HTTP POST

Endpoint: *http://<host_address>:<port_number>/v1/configuration/replicateTenantProperties/{sourceTenantId}*

Parameters: sourceTenantId



Note: If the secure API is enabled, you need to create the required tenant by using the authorization token.

Get list of applications available with Configurator

Method: HTTP GET

Endpoint: *http://<host_address>:<port_number>/v1/configuration/availableApplications*

Parameters: No parameters

Download Configuration File for Application ID

Method: HTTP GET

Endpoint: *http://<host_address>:<port_number>/v1/configuration/configFile/{applicationId}*

Parameters: applicationid (Select value from the drop-down)

Upload Configuration File for Application ID

This API is used to upload config files into the configuration table in the admin database through the REST service.

Method: HTTP POST

Endpoint: *http://<host_address>:<port_number>/v1/configuration/configFile/{applicationId}*

Select the applicationid from the drop-down list, click **Choose File** to browse, select the file that you want to upload, and click **Execute**.



Note: When you upload a file that is already present in the database, the older file is replaced.

Configurator notification API

This API generates notification for server side events whenever there is a change (add, update, or delete) in application properties for any application.

Method: HTTP GET

Endpoint: *http://<host_address>:<port_number>/v1/configurator/events*

Parameters: No parameters

Authorization Service

Token-based authentication is implemented in TIBCO Product and Service Inventory to ensure secure access to TIBCO Product and Service Inventory Server REST APIs, and to support multitenancy. The authentication service in TIBCO Product and Service Inventory uses JSON WebToken(JWT) to validate user credentials (user name, password, and tenantID).

The following functions are covered under the Authorization Service:

- [Create User](#)
- [Update User](#)
- [Get User](#)
- [Delete User](#)

Create User

This request is used to create a new user.

Method: HTTP POST method

Endpoint:*http://<host_address>:<port_number>/v1/user*

Create User Parameters

Parameter	Cardinality	Description
X-API-AppId	Mandatory	The application ID is used to get user details. The default ID is auth.
X-API-Key	Mandatory	This key is used to get user

Parameter		Cardinality	Description
			details. The default ID is auth.
userInfo (Body)	enabled	Mandatory	The value can be "true" or "false". If you set the value as "true", then the user is accessible through the Configurator and the "false" value disables the user.
	password	Mandatory	The password to be used for the user.
	tenantId	Mandatory	This is the TENANT value as stored in the users' table in the database. If the tenantId is not present in the database, then a new TENANT is created.
	userName	Mandatory	It specifies the user name to be created or modified.
	userRoles	Mandatory	It assigns the role to the user. The valid role values are ROLE_ADMIN, ROLE_USER, ROLE_PARTY, ROLE_ITEM

Example for the Create User request:

```
{
  "user": [
```

```

{
  "enabled": true, "password": "testpassword", "tenantId": "testTenant", "userName":
  "testuser", "userRoles": [
    "ROLE_ADMIN"
  ]
}

```

Update User

This request is used to update an existing one.

Method: HTTP PUT method

Endpoint: *http://<host_address>:<port_number>/v1/user*

Update User Parameters

Parameter	Cardinality	Description
X-API-AppId	Mandatory	<p>The application ID is used to get user details.</p> <p>The default ID is auth.</p>
X-API-Key	Mandatory	This key is used to get user

Parameter		Cardinality	Description
			details. The default ID is auth.
userInfo (Body)	enabled	Mandatory	The value can be "true" or "false". If you set the value as "true", then the user is accessible through the Configurator and the "false" value disables the user.
	password	Mandatory	The password to be used for the user.
	tenantId	Mandatory	This is the TENANT value as stored in the users' table in the database. If the tenantId is not present in the database, then a new TENANT is created.
	userName	Mandatory	It specifies the user name to be created or modified.
	userRoles	Mandatory	It assigns the role to the user. The valid role values are ROLE_ADMIN, ROLE_USER, ROLE_PARTY, ROLE_ITEM

Example for the Update User request:

```
{
  "user": [
```

```

{
  "enabled": true, "password": "testpassword", "tenantId": "testTenant", "userName":
  "testuser", "userRoles": [
    "ROLE_ADMIN"
  ]
}

```

Get User

This request is used to get the details of the existing user.

Method: HTTP GET method

Endpoint: *http://<host_address>:<port_number>/v1/user*

Get User Parameters

Parameter	Cardinality	Description
X-API-AppId	Mandatory	The applicationID is used to get the user details. The default ID is auth.
X-API-Key	Mandatory	This key is used to get the user details. The default ID is auth.

Parameter	Cardinality	Description
tenantId	Mandatory	This is the TENANT value as stored in the users' table in the database.
userId	Mandatory	This is the userId value as stored in the users' table in the database.

Delete User

This request is used to delete the existing user.

Method:HTTP DELETE method

Endpoint:*http://<host_address>:<port_number>/v1/user*

Delete User Parameters

Parameter		Cardinality	Description
X-API-AppId		Mandatory	The applicationID is used to get the user details. The default ID is auth.
X-API-Key		Mandatory	This key is used to get the user details. The default ID is auth.
userInfo (Body)	tenantId	Mandatory	Tenant value stored in the users' table in the database.
	userName	Mandatory	It specifies the user name to be deleted.

Example: Delete User request

```
[
```

```
[
```

```
{
```

```
  "userName":  
  "string",
```

```
  "tenantId":  
  "string"
```

```
}
```

```
]
```

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the [TIBCO Product Documentation](#) website, mainly in HTML and PDF formats.

The [TIBCO Product Documentation](#) website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation for this product is available on the [TIBCO® Product and Service Inventory](#) documentation page:

- *TIBCO® Product and Service Inventory Release Notes*
- *TIBCO® Product and Service Inventory Installation and Configuration*
- *TIBCO® Product and Service Inventory User Guide*
- *TIBCO® Product and Service Inventory REST Services Guide*
- *TIBCO® Product and Service Inventory Web Services Guide*

How to Contact TIBCO Support

Get an overview of [TIBCO Support](#). You can contact TIBCO Support in the following ways:

- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the [TIBCO Support](#) website.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to [TIBCO Support](#) website. If you do not have a user name, you can request one by clicking **Register** on

the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the [TIBCO Ideas Portal](#). For a free registration, go to [TIBCO Community](#).

Legal and Third-Party Notices

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, the TIBCO logo, and the TIBCO O logo are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the readme file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

This and other products of TIBCO Software Inc. may be covered by registered patents. Please refer to TIBCO's Virtual Patent Marking document (<https://www.tibco.com/patents>) for details.

Copyright © 2015-2022. TIBCO Software Inc. All Rights Reserved.