Source URL: https://help.calibo.com/lazsa/content/configuration/configure%20lazsa%20platform.htm

Platform Setup

After the Calibo team creates your Lazsa Platform subscription and you, as an administrator, perform the initial configuration tasks in the F24H (First 24 Hours) wizard, it is time to set up the platform to make it ready for use for your digital product development teams and other stakeholders.

Configuring Lazsa Platform

Configuration Section	Function
Users, Roles, Teams, and Organization Hierarchy	Manage your organizational hierarchy, users, teams, roles, and permissions.
Lazsa Orchestrator Agents	Configure agents to connect to tools in your environment which the Lazsa Platform cannot access directly.
Cloud Platform, Tools and Technologies	Configure connection details for the tools and technologies in your software engineering environment. Configure your accounts for cloud services, container platforms, development tools, software frameworks, code and artifact repositories, collaboration and communication tools, and more.
Data Pipeline Studio	Configure data crawlers and import data ingestion catalogs.
Settings	Configure platform settings and product-specific settings such as themes, cloud instance configuration, audit logs, custom fields, schedulers, and more.
Standards	Define standard templates and processes for product development.
Maturity Assessment	Manage categories, map questions, and configure other settings for maturity assessment.

Recommended Topics	**What's next?** Cloud Platform Tools & Technologies

---

Source URL: https://help.calibo.com/lazsa/content/configuration/users_roles_and_teams_organization hierarchy.htm

Users, Roles, Teams, and Organization Hierarchy

The Lazsa Platform uses role based access control (RBAC) to monitor and control access to authorized users to various objects and processes in the platform.

The platform provides predefined roles that come with a set of default permissions, or custom roles that you can create based on your requirement. You can assign predefined or custom roles to users based on the tasks that they need to perform in the Platform.

You can create teams to group users working on different products, releases or features and add users to each team.

Organization Hierarchy helps to create a structure and define the hierarchy within the organization. This facilitates the day-to-day operations of the organization.

Before you start creating teams, custom roles or assigning roles to users, we recommend you go through the following information:

Mapping of roles, objects and permissions

What tasks can I perform from Users, Roles, Teams, and Organization Hierarchy?

You can perform several tasks from the different tabs of the Users, Roles, Teams, and Organization Hierarchy workspace.

Users

- Add users to the Platform using one of the following methods:
  - Add Users through Active Directory
  - Add Users Manually
  - Import Users from CSV files
- View the following details about users:
  - User, Platform or Product roles
  - Company, Country
  - Status of login access, Manager
  - Skillsets, Cost Area, Cost, Teams
  - Allocation %, Allocation Period

    - Custom Fields (See Custom Fields Settings)
- Search for a specific user
- Filter users based on role type, company, skillset, cost area, teams, country and so on. You can check **Select All** to select all the options of the filter. Similarly you can uncheck **Select All** to clear all the options of the filter. Once you select filters and apply them they are saved and loaded for each next session until you reset the filters.
- Export user details to an excel sheet
- Manage users by performing the following tasks by clicking the ellipsis (..):
    - Activate or deactivate a user Deactivate or Offboard a User
    - Send email - send an email to a user
    - Add to team - add a user to a team
    - Edit a user
    - Offboard a user
    - View History
    - Delete a user

- Add Users through Active Directory
- Add Users Manually
- Import Users from CSV files

- User, Platform or Product roles
- Company, Country
- Status of login access, Manager
- Skillsets, Cost Area, Cost, Teams
- Allocation %, Allocation Period
- Custom Fields (See Custom Fields Settings)

- Activate or deactivate a user Deactivate or Offboard a User
- Send email - send an email to a user
- Add to team - add a user to a team
- Edit a user
- Offboard a user
- View History
- Delete a user

Roles

- Search for a specific role
- View role details
    - Edit
    - Manage Users
    - Delete
- Export roles to an Excel sheet
- Perform tasks like:
    - Edit users
    - Manage users - add them to a role
    - View users
    - Delete users
- New Role - create a custom role
- Export platform and product roles to an excel sheet
- Configure Table Settings - set the table view by selecting any four columns apart from the first two, which are mandatory.

- Edit
- Manage Users
- Delete

- Edit users
- Manage users - add them to a role
- View users
- Delete users

Teams

- New Team - create a new team
- Export a list of teams to an Excel sheet
- Search for a specific team
- Filter teams based on the organizational structure that is defined.
- Manage tasks like:
    - Edit
    - Manage Members
    - Email Members
    - Delete

- Edit

- Manage Members
- Email Members
- Delete

Organization Hierarchy

- Create a structure for the organization

- Add the groups according to the defined structure

Create a structure for the organization

Add the groups according to the defined structure

Recommended Topics	**What's next?** Create Teams

Source URL: https://help.calibo.com/lazsa/content/configuration/standards.htm

Standards

From the Standards screen you can create different templates to enforce certain best practices and standard processes throughout the product development lifecycle.

Template	Description
Policy Templates	Help enforce certain configuration and settings throughout the product development lifecycle.
Branch Templates	Help define your branching strategy for the code.
Workflow Templates	Help create approval workflows for different phases of the product development lifecycle.

Recommended Topics	**What's next?**Users, Roles, Teams, and Organization Hierarchy

---

Source URL: https://help.calibo.com/lazsa/content/configuration/create_custom_role.htm

Create Custom Role

Why should I create custom roles?

The Lazsa platform provides a set of predefined roles with privileges to perform specific tasks in the platform. Additionally, you can create custom roles and assign specific privileges to such roles, based on your requirement. You can then assign custom roles to different users. Before you create custom roles, you may want to understand how permissions are mapped to roles and objects.

What role do I need to create a custom role?

- **Platform** - these roles are related to the platform.
- **Product** - these roles are specific to a product.

How do I assign custom roles to users?

You are done with creating custom roles as well as assigning them to users. You may now want to create teams.

Recommended Topics	**What's next?** Create Teams

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/import_users_from_csv_files.htm

Import Users from CSV Files

Apart from importing users into the Lazsa Platform using the First 24 Hours wizard, you can also add users from Users, Roles and Teams workspace. You can add users by one of the following methods:

- Add Users Through Active Directory
- Import Users from CSV Files
- Add Users Manually

The **Users** tab shows a list of all the users that are added to the platform. You can filter the users based on the type of role, company, skillset, cost area and so on.

Recommended Topics	**What's next?** Create Custom Role

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/policy_templates_settings.htm

Policy Template Settings

A policy template is a way to define and apply settings for tools, development phases, maturity assessment categories and so on. Based on the settings of the policy template, you can decide the following:

- Categories of maturity assessment
- Number of features that can be added to a product
- Tech stack and tools to be used for product development
- Settings of the deploy phase like pipeline configuration, deployment mode, cloud platform accounts, cluster configuration, machine configuration, and deployment stages.

Once you publish a policy template, you can apply it at the product or feature level.

See Create Product - Additional Details

See Create Feature - Additional Details

How do you create a policy template?

General - **Assessment Categories**	Enable the assessment categories and sub-categories.
General - **Collaboration Tools**	Enable Microsoft Teams.
General - **Portfolio Management**	Enable Rally or Aha based on your preference.
General - **Features**	Enable **Configure maximum number of features** and specify the maximum number of features that can be created under a product.
Define	Enable Agile Management Tools and select the tool of your choice.
Design	• Enable Document Management Tool and select Confluence. • Enable Design Tools.
Develop	Select the frontend, backend, and other tools required for developing a product.
Deploy	Select tools for the following options: • **Deployment Mode** • **Cloud Platform Accounts** • Kubernetes Cluster Configuration • Terraform Configuration • Machine Configuration • **Deployment Stages** - the various deployment stages that can be created for the product.

- Enable Document Management Tool and select Confluence.
- Enable Design Tools.

Select tools for the following options:

- **Deployment Mode**
- **Cloud Platform Accounts**
- Kubernetes Cluster Configuration
- Terraform Configuration
- Machine Configuration
- **Deployment Stages** - the various deployment stages that can be created for the product.

Recommended Topics	**What's next?** Branch Templates Settings

---

Source URL: https://help.calibo.com/lazsa/content/configuration/mapping_of_roles_objects_permissions.htm

Mapping of Roles, Objects, and Permissions

The following table depicts how permissions are mapped to the roles and objects in the Lazsa Platform.

Role	Objects	Permissions
Tenant Admin	User, Role, Team, Identity Provider, Account, Cluster, Tech Stack, Jenkins, Bitbucket, Project, Features, Release Management Plan, VSM, Define/Business Requirements, Define/Ideas, Design, Develop, Deploy, DeployStage, Portfolio, Maturity Assessment	Create, View, Edit, Delete
Project Administrator	Project, Workstream, Release Management Plan, VSM, Define/Business Requirement, Define/Ideas, Design, Develop, Deploy, DeployStage, Portfolio, Maturity Assessment.	Create, View, Edit, Delete
Executive	Dashboard Maturity Assessment	Create, View, Edit, Delete Create, View, Edit, Delete
Config Administrator	Identity Provider, Account, Cluster, Tech Stack, Jenkins, Bitbucket, Maturity Assessment	Create, Edit, View, Delete
Innovator	Project Portfolio Maturity Assessment	View View View
Product Owner	Define Design Develop Deploy DeployStage	Create, View, Edit, Delete View View View View
Architect	Define Design Develop Deploy DeployStage	Create, View, Edit, Delete Create, View, Edit, Delete View View View
Tech Lead	Design Develop Define Deploy Project	Create, View, Edit, Delete Create, View, Edit, Delete View View View
Developer	Design Define Develop Deploy Project	View View Create, View, Edit, Delete View View

Role	Objects	Permissions
DevOps	Design Define Develop Deploy Project DeployStage	View View View Create, View, Edit, Delete View Create, View, Edit, Delete
QA	Design Define Develop Deploy Project DeployStage	View View View Create, View, Edit, Delete View Create, View, Edit, Delete
Business Analyst	Define Design Develop Project	Create, View, Edit, Delete View View View
Project Owner	Project Workstream, Release Management Plan, VSM, Define, Design, Develop, Deploy, DeployStage, Portfolio, Maturity Assessment SignOff Team	View, Edit Create, View, Edit Create, View, Edit View Create, View, Edit

Tenant Admin

User, Role, Team, Identity Provider, Account, Cluster, Tech Stack, Jenkins, Bitbucket, Project, Features, Release Management Plan, VSM, Define/Business Requirements, Define/Ideas, Design, Develop, Deploy, DeployStage, Portfolio, Maturity Assessment

Create, View, Edit, Delete

Project Administrator

Project, Workstream, Release Management Plan, VSM, Define/Business Requirement, Define/Ideas, Design, Develop, Deploy, DeployStage, Portfolio, Maturity Assessment.

Create, View, Edit, Delete

Executive

Dashboard

Maturity Assessment

Create, View, Edit, Delete

Create, View, Edit, Delete

Config Administrator

Identity Provider, Account, Cluster, Tech Stack, Jenkins, Bitbucket, Maturity Assessment

Create, Edit, View, Delete

Innovator

Project

Portfolio

Maturity Assessment

View

View

View

Product Owner

Define

Design

Develop

Deploy

DeployStage

Create, View, Edit, Delete

View

View

View

View

Architect

Define

Design

Develop

Deploy

DeployStage

Create, View, Edit, Delete

Create, View, Edit, Delete

View

View

View

Tech Lead

Design

Develop

Define

Deploy

Project

Create, View, Edit, Delete

Create, View, Edit, Delete

View

View

View

Developer

Design

Define

Develop

Deploy

Project

View

View

Create, View, Edit, Delete

View

View

DevOps

Design

Define

Develop

Deploy

Project

DeployStage

View

View

View

Create, View, Edit, Delete

View

Create, View, Edit, Delete

QA

Design

Define

Develop

Deploy

Project

DeployStage

View

View

View

Create, View, Edit, Delete

View

Create, View, Edit, Delete

Business Analyst

Define

Design

Develop

Project

Create, View, Edit, Delete

View

View

View

Project Owner

Project

Workstream, Release Management Plan, VSM, Define, Design, Develop, Deploy, DeployStage, Portfolio, Maturity Assessment

SignOff

Team

View, Edit

Create, View, Edit

Create, View, Edit

View

Create, View, Edit

Recommended Topics	**What' next?** Create Custom Role

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/add_users_manually.htm

Add Users Manually

Apart from importing users into the Lazsa Platform using the First 24 Hours wizard, you can also add users from the Users, Teams and Roles screen by one of the following methods:

- Add Users through Active Directory
- Import Users from CSV Files

- Add users manually

The **Users** tab shows a list of all the users that are added to the platform. You can filter the users based on the type of role, company, skillset, cost area and so on.

To add users manually

- Enter **First Name**.
- Enter **Last Name**.
- Enter **Email**.
- Enter **Secondary Email** if any. A user who is appointed by a staffing agency may have an email ID of that domain as well.
- Specify the **Company** name if the user is appointed by a staffing agency.
- Select **Country** from the drop-down list.
- Select **Cost Area** - depending on the cost of the resource, cost areas are divided into High Cost Area (HCA), Medium Cost Area (MCA), and Low Cost Area (LCA).
- **Cost (per hour in USD)** - resource cost per hour in US dollars.
- **Skillset** of the resource.
- **Allocation Period** - Specify the **Start Date** and **End Date** of the period for which the resource is available.
- **Purpose** - specify whether the user is being added to administer the platform or to work on products.
- **Custom Fields** - add any additional attributes that you would like to associate with the user.
- **Assign Roles** - assign the required platform roles to the user.

Recommended Topics	**What's next?** Create Teams

Source URL: https://help.calibo.com/lazsa/content/configuration/add_users_through_active_directory.htm

Add Users through Active Directory

Apart from importing users into the Lazsa Platform using the First 24 Hours wizard, you can also add users from the Users, Roles, and Teams screen by one of the following methods:

- Add users through Active Direcrtory
- Import from CSV Files
- Add Manually

The **Users** tab shows a list of all the users that are added to the platform. You can filter the users based on the type of role, company, skillset, cost area and so on.

Adding users through active directory

Recommended Topics	**What's next?** Create Teams

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/custom_fields_settings.htm

Custom Fields Settings

What are custom fields?

Custom fields are additional data fields that you can define for various areas across the Lazsa Platform. Custom fields help you capture specific information that may not be covered by the default fields within the platform. By using custom fields, you can tailor the platform to meet your unique business needs and gather additional data relevant to your processes.

For example, you may want to add a custom field "Region" to capture specific information related to the geographic region or location associated with the product and make it a mandatory field. This means that when adding or editing product details, users must input the relevant region in the "Region" custom field. This information can be useful for products with regional variations, different distribution strategies, or specific market penetration goals.

Defining Global Custom Fields

To define the global custom fields, do the following:

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

- Product Portfolio

- Product

- Release
- Feature
- Team
- Team Member
- Product Milestone
- Feature Milestone
- Users

Product Portfolio

Product

Release

Feature

Team

Team Member

Product Milestone

Feature Milestone

Users

In each area where you want to use custom fields, turn on the **Enable Custom Fields** option.

Field Type	Description
Single Line	A single line field allows users to input and display a single line of text or a short string of characters (no longer than 256 characters). Use this field to capture concise information such as names, titles, email addresses, phone numbers, or other brief descriptions.
Multiline	In this field, users can input multiple lines of text or larger blocks of content. Use a multiline field to accommodate more extensive information such as descriptions, comments, notes, or paragraphs.
Dropdown	A dropdown field provides a predefined list of options to choose from. Users can select one option from the list. Use this field to capture categorical or discrete data, where users need to select a single choice from a set of available options. In the Dropdown Options box, predefine the options. Enter a value and press **Enter** to confirm and to add the next value.
Date Time	A date field allows users to input dates. Users can choose a date from a calendar picker or manually enter it in the DD-MM-YYYY date format. Date fields are commonly used to record event dates, deadlines, milestones, or any other time-related information.
Hyperlink	This allows users to add a clickable link or URL to the data. Hyperlink fields are beneficial for referencing external content, such as websites, documents, or references within the context of the data being managed. You must add a hyperlink in the following syntax: Name || Link For example, My Product Name || https://help.sample.com

A single line field allows users to input and display a single line of text or a short string of characters (no longer than 256 characters). Use this field to capture concise information such as names, titles, email addresses, phone numbers, or other brief descriptions.

In this field, users can input multiple lines of text or larger blocks of content. Use a multiline field to accommodate more extensive information such as descriptions, comments, notes, or paragraphs.

A dropdown field provides a predefined list of options to choose from. Users can select one option from the list. Use this field to capture categorical or discrete data, where users need to select a single choice from a set of available options.

In the Dropdown Options box, predefine the options. Enter a value and press **Enter** to confirm and to add the next value.

This allows users to add a clickable link or URL to the data. Hyperlink fields are beneficial for referencing external content, such as websites, documents, or references within the context of the data being managed.

You must add a hyperlink in the following syntax:

Name || Link

For example, My Product Name || https://help.sample.com

Save your changes.

Thatâs it! The custom fields are now visible in the specified areas on the platform interface.

Recommended Topics	**What's next?** Agile Project Settings

Source URL: https://help.calibo.com/lazsa/content/configuration/general_settings.htm

General Settings

On the General Settings tab, you can decide the product development phases that you want to make available for users at the product level in the platform. You can also decide whether to apply different policy templates to different products in your portfolio, or apply a common policy template across the entire portfolio.

Allow activities on product creation

In this section, enable the phases that you want to include in a product development workflow. This way you can tailor the workflow to align with your project requirements. Here is what each phase means:

Define: This phase typically involves defining business requirements, scoping the project, and outlining the objectives.

Design: This phase focuses on creating the visual and functional aspects of the project, including wireframing, prototyping, and creating the user interface.

Develop: This phase involves writing and implementing code, building features, and ensuring the functionality of the project.

Deploy: This phase involves releasing the project or feature to the production environment, making it accessible to end users.

Policy Templates

A policy template provides a powerful mechanism to define and apply a consistent set of settings for product development within the Lazsa Platform. It serves as a comprehensive blueprint for your project, enabling you to define the tools and technologies to be utilized throughout the product development life cycle.

You can enforce a policy template separately to each product or a common template to the entire portfolio. In this section, choose whether you want to apply a template to a product individually or a common template to the portfolio. If you apply a policy template at the portfolio level, it is applied to all the products associated with the portfolio.

Recommended Topics	**What's next?** Machine Configuration Settings

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/branch_templates_settings.htm

Branch Templates Settings

The Branch Templates tab of the settings page lets you manage branch templates. The page lists created branch templates. You can create a new template.

Related Topics	**What's next?** Add and Configure Technologies

---

Source URL: [https://help.calibo.com/lazsa/content/configuration/dropdown field values.htm](https://help.calibo.com/lazsa/content/configuration/dropdown_field_values.htm)

Dropdown Field Values

In the global settings of the Lazsa Platform, on the **Dropdown Field Values** tab, you can add and manage the values for dropdown lists that appear in different workspaces across the platform. By predefining dropdown values, you provide a standardized list of options for users to choose from. This simplifies their data entry, making it faster and more convenient.

You can also allow users to add new values to dropdown lists in addition to predefined values. This helps them tailor the dropdown lists to their specific needs. Allowing this real-time customization also reduces your administrative overhead as you don't need to handle every customization request.

Defining Dropdown Field Values

To predefine and mange dropdown field values, do the following:

This is how you define and maintain dropdown list options in the Lazsa Platform. Next, you may want to know more about the AIOps data collection schedulers in the platform.

Recommended Topics	**What's next?** Schedulers

Source URL: https://help.calibo.com/lazsa/content/configuration/cloud_platform_tools_and_technologies.htm

Cloud Platform, Tools and Technologies

The Lazsa Platform offers a comprehensive set of infrastructure, tools and technologies that you can use for product development and related activities. You can configure various tools required, based on the type of product you are developing.

Tool/Technology	Description
Cloud Infrastructure for Provisioning	You can either use the platform-managed cloud infrastructure (that is enabled by default), or deactivate it and configure one of the following options: • AWS • Azure
Tech Stacks	Select the type of application you are creating and accordingly select the required tech stacks: • Web App Tech Stack • API Tech Stack • Data Integration Tech Stack • Data Analytics Tech Stack • Database Tech Stack • Data Visualization Tech Stack
DevOps CI/CD Pipeline Configuration	You can either use the platform-managed CI/CD Pipeline which provides Gitlab or you can configure one of the following CI/CD pipelines: • AWS Code Pipeline • Jenkins • GiLlab • Azure Pipeline • Github
Kubernetes Cluster Configuration	Configure Kubernetes Cluster for the following types: • AWS • Azure
Source Code Repository	Select the source code repository from the following options: • GitLab • Bitbucket • GitHub
Collaboration Tools	Select the collaboration tools from the following options: • Microsoft Teams • Zoom • Slack
Agile Planning Tools	Select an agile planning tool from the following options: • Jira • Trello • Custom

Tool/Technology	Description
Document Management Tools	Configure a document management tool provided by the Platform from the following information: • Confluence • Google Docs
Data Stores	Connect to a database from the following options: • Amazon S3 • Snowflake • RDBMS • AWS RDS
Data Integration	• DataBricks • Apache Kafka • NiFi
Data Visualization	• QlikSense
Developer Tools	• Swagger
Artifactory Management Tools	• Amazon ECR
Security Assessment Tools	• SonarQube
Ticketing Tools for Workflow	• ServiceNow

You can either use the platform-managed cloud infrastructure (that is enabled by default), or deactivate it and configure one of the following options:

- AWS
- Azure

Select the type of application you are creating and accordingly select the required tech stacks:

- Web App Tech Stack
- API Tech Stack
- Data Integration Tech Stack
- Data Analytics Tech Stack
- Database Tech Stack
- Data Visualization Tech Stack

You can either use the platform-managed CI/CD Pipeline which provides Gitlab or you can configure one of the following CI/CD pipelines:

- AWS Code Pipeline
- Jenkins
- GiLlab
- Azure Pipeline
- Github

Kubernetes Cluster Configuration

Configure Kubernetes Cluster for the following types:

- AWS
- Azure

Source Code Repository

Select the source code repository from the following options:

- GitLab
- Bitbucket
- GitHub

Select the collaboration tools from the following options:

- Microsoft Teams
- Zoom
- Slack

Select an agile planning tool from the following options:

- Jira
- Trello
- Custom

Document Management Tools

Configure a document management tool provided by the Platform from the following information:

- Confluence
- Google Docs

Connect to a database from the following options:

- Amazon S3
- Snowflake
- RDBMS
- AWS RDS

- DataBricks
- Apache Kafka
- NiFi

Data Visualization

- QlikSense

Developer Tools

- Swagger

Artifactory Management Tools

- Amazon ECR

Security Assessment Tools

- SonarQube

Ticketing Tools for Workflow

- ServiceNow

After you have configured the required tools and technology stacks, you can create a Product Line, add a Product, Releases and Features to the Product. See About Product Line, Product, Release, and Features

Recommended Topics	**What's next?** Product Portfolio or Product Line

---

Source URL: https://help.calibo.com/lazsa/content/configuration/terminologies_settings.htm

Terminology Settings

The Lazsa Platform provides you the flexibility to customize some predefined terms that are visible across the platform interface. This allows you to use terms that resonate with your company's language, internal processes, and industry standards. You can create a more personalized and user-friendly experience for your users, aligning with your organization's terminology.

Customizing Predefined Terms

To customize predefined terms, do the following:

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

Benefits of Customization

The following are a few benefits of customizing predefined terms on the platform interface.

- **Personalization**: Customizing predefined terms allows you to create a more personalized experience for your users, making the platform feel like an extension of your company's language and processes.
- **Improved Understanding**: Following your organization's terminology ensures better communication and understanding among

team members and stakeholders.

- **Adaptation to Industry Standards**: You can adhere to your specific industry jargon or standard terminologies, promoting seamless integration into your workflows.
- **Reduced Learning Curve**: Users won't need to learn new terminologies, reducing the learning curve and making onboarding and training more efficient.

Personalization: Customizing predefined terms allows you to create a more personalized experience for your users, making the platform feel like an extension of your company's language and processes.

Improved Understanding: Following your organization's terminology ensures better communication and understanding among team members and stakeholders.

Adaptation to Industry Standards: You can adhere to your specific industry jargon or standard terminologies, promoting seamless integration into your workflows.

Reduced Learning Curve: Users won't need to learn new terminologies, reducing the learning curve and making onboarding and training more efficient.

After you customize the predefined terms , you may want to configure the settings for custom fields that appear across the Lazsa Platform interface.

Recommneded Topics	**What's next?** Custom Fields Settings

Source URL: https://help.calibo.com/lazsa/content/configuration/settings.htm

Settings

The Settings section enables you to configure and customize various settings that apply globally across all projects and products within the Lazsa Platform. These settings allow you to establish consistent rules, preferences, and configurations, ensuring a streamlined and efficient experience for your organization. You can define the general preferences, choose machine configuration settings for cloud instances, manage security settings, customize theme settings, and define many other options to ensure a seamless self-service experience for users across the platform.

To configure the global settings, do the following:

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

- General Settings
- Machine Configuration Settings
- Security Settings
- Themes Settings
- Audit Logs Settings
- Terminology Settings
- Custom Fields Settings
- Agile Project Settings
- Dropdown Field Values
- Schedulers

Recommended Topics	**What's next?** Account Info

Source URL: https://help.calibo.com/lazsa/content/configuration/workflow_templates_settings.htm

Workflow Templates Settings

You can view the various workflow templates that are available as part of the project. You can search and filter templates. You can also create a new workflow template.

Related Topics	**What's next?** Audit Logs

Source URL: https://help.calibo.com/lazsa/content/configuration/create_teams.htm

Create Teams

Why should I create a team?

Teams help you to group together, the users working on a product in the Lazsa Platform. This helps in managing resources working on a product or assigning permissions to users at the product level efficiently.

What actions can I perform from the Teams tab?

You can perform various actions related to teams from this screen.

Action	Description	
Search team	Search for a specific team using the team name.	
Create new team	Click + **New Team**	
View team details	• Team • Description • Members • Custom fields added to the team • Click the ellipsis (**...**) to the right of the team to perform the following actions: ○ **Edit** the team details. See Edit Team ○ **Manage Members** by adding or removing members ○ **Email Members** - send an email to all the members of the team ○ **Delete** - delete the team ○ **Deactivate** - deactivate a team from the following month. Before you confirm the deactivation, you can click **View Details** to view the list of products that the team is associated and a list of users that are part of the team.	

- Team
- Description
- Members
- Custom fields added to the team
- Click the ellipsis (**...**) to the right of the team to perform the following actions:
  - **Edit** the team details. See Edit Team
  - **Manage Members** by adding or removing members
  - **Email Members** - send an email to all the members of the team
  - **Delete** - delete the team
  - **Deactivate** - deactivate a team from the following month. Before you confirm the deactivation, you can click **View Details** to view the list of products that the team is associated and a list of users that are part of the team.

- **Edit** the team details. See Edit Team
- **Manage Members** by adding or removing members
- **Email Members** - send an email to all the members of the team
- **Delete** - delete the team
- **Deactivate** - deactivate a team from the following month. Before you confirm the deactivation, you can click **View Details** to view the list of products that the team is associated and a list of users that are part of the team.

What role do I need to create a team?

How do I create a team?

- Product Team - select this option to create a team that you want to associate with a product.
- Other - select this option to create a non-functional team, one which is not associated with a product.

- **Team Name**
- **Description**
- **Link the Team with an Agile Board** - when you enable this option, and add this team to a product, the data from the agile board related to the team is fetched and used to calculate the team maturity.
- **Organization Hierarchy**
  - Select the **Domain** or **Group** for the team.
- **Create group for this team in Azure Active Directory** - when you enable this option you can create a user group for this team in Azure Active Directory.
- **Custom Fields** - add any additional parameters that you may want to link with this team.

- Select the **Domain** or **Group** for the team.

- **Team Type** - if you have selected the option - Other, select the type of team that you want to create.
- **Team Name**
- **Description**
- **Custom Fields** - add any additional parameters that you may want to link with this team.

Now that you have created a team, you can add additional users to the team.

How do I add users to a team?

In the Team Members screen, select a member from the drop down list and a role to assign to the user.

Recommended Topics	**What's next?**Create Custom Role

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/machine_configuration_settings.htm

Machine Configuration Settings

The Machine Configuration Settings screen helps you control the infrastructure used for deploying your applications or technology stack. With these settings, you can manage and optimize infrastructure costs by limiting the machine configurations and the number of cloud instances and load balancers that users can provision and use for their deployments.

The configuration options that you enable here will be available for selection to developers when they add cloud instances in the Deploy phase of feature development.

Note:

If you enforce a policy template, the machine configuration options enabled in the policy template supersede the options that you enable on this screen.

Note:

If you enforce a policy template, the machine configuration options enabled in the policy template supersede the options that you enable on this screen.

To enable the required machine configuration settings, do the following.

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

Enable or disable the predefined instance configuration options based on your application's needs and performance requirements. The following lighter and hence, cost-efficient configurations are available. They vary in terms of their memory (RAM) and the number of virtual Central Processing Units (vCPUs).

- Large - 8 GB RAM 2 vCPU
- Medium - 4 GB RAM 2 vCPU
- Small - 2GB RAM 2 vCPU

Custom Configuration

In this section, we have predefined some heavy configuration options for you. You can add and enable these additional options as per your development requirements. You may require these instance types for various reasons, depending on the nature of your applications, development tasks, and performance requirements. The following are some common scenarios where developers might opt for such instances:

- For building resource-intensive applications
- To meet high traffic and scalability requirements
- For big data processing tasks
- In testing and staging environments that closely mimic production settings
- In virtualization and containerization to ensure optimal performance for running multiple instances or containers on a single host

For building resource-intensive applications

To meet high traffic and scalability requirements

For big data processing tasks

In testing and staging environments that closely mimic production settings

In virtualization and containerization to ensure optimal performance for running multiple instances or containers on a single host

Note:

While heavy configuration instances provide significant computational power, they are cost-intensive, so you should carefully assess your specific requirements and workload demands to strike the right balance between performance and cost-effectiveness.

Note:

While heavy configuration instances provide significant computational power, they are cost-intensive, so you should carefully assess your specific requirements and workload demands to strike the right balance between performance and cost-effectiveness.

Enable this option and define the maximum number of cloud instances that can be provisioned across products being developed from within the Lazsa Platform. By setting a maximum limit on the number of cloud instances, you can control and manage infrastructure costs more effectively. This prevents users from unintentionally provisioning excessive instances, which could lead to unexpected expenses.

This is a setting used in your Docker deployments from within the Lazsa Platform. If you enable this option, users can choose between manual and automatic load balancer creation at a stage level in the Deploy phase of feature development.

With automatic load balancer creation, users can offload the load balancer setup tasks to the Lazsa Platform, making it easier for users to deploy scalable and highly available applications.

Thus, the machine configuration settings enable you to optimize your infrastructure usage, adhere to budget constraints, and make efficient decisions when deploying your tech stack or applications from within the Lazsa Platform. With these settings, you can strike a balance between performance and cost-effectiveness.

Recommended Topics	**What's next?** Security Settings

Source URL: https://help.calibo.com/lazsa/content/configuration/agile_project_settings.htm

Agile Project Settings

In the context of Agile project management, an Agile project key is a unique identifier or code assigned to a specific Agile project. It serves as a reference or identifier for the project within an Agile project management system or tool.

In this section, you can choose whether you want to associate an Agile project key individually at the product level, or add a common agile project key to your entire portfolio.

If you decide to associate an Agile key with a product, you see a dropdown list of Agile project keys while adding the product details. Linking an Agile project key to a product enhances organization, traceability, and collaboration. It helps in alignment with Agile methodologies and streamlined project management.

While Agile project keys are commonly used to identify and manage individual projects, they can also be linked with higher-level structures such as a product portfolio. If you decide to associate an Agile key with a portfolio, you see a dropdown list of Agile project keys on the Agile tab while adding the portfolio details. By associating an Agile project key to a product portfolio, you establish a consistent identifier that represents the entire product line within your Agile project management system or tool. This can be beneficial when managing multiple products under a unified product line, as it provides a clear reference point for the entire line.

The available keys that you see in the dropdown list at the product or portfolio level are fetched from your default Agile planning tool that you configure on the Cloud Platforms, Tools & Technologies screen.

Recommended Topics	**What's next?** Dropdown Field Values

Source URL: https://help.calibo.com/lazsa/content/configuration/schedulers.htm

Schedulers

Scheduling enables you to automate the execution of various tasks across the Lazsa Platform. Currently, schedulers are available for the AIOps data collection tasks.

Configuring Scheduler Options for AIOps Data Collection

You can configure schedules for AIOps data collection from various tools running in your Lazsa environment. This data is further processed and used for graphical representation in the Monitoring dashboards. Currently, the following AIOps data collection schedulers are available in the Lazsa Platform:

Scheduler	What it does
Agile Tools Scheduler	Collects data from your agile planning tools like Jira.
Security Assessment Scheduler	Collects data from the security assessment monitoring tools such as SonarQube and Qualys.

Scheduler	What it does
Source Code Scheduler	Collects data related to code commits and code merges from the source code repositories in your Lazsa development environment.

Security Assessment Scheduler

Collects data from the security assessment monitoring tools such as SonarQube and Qualys.

Collects data related to code commits and code merges from the source code repositories in your Lazsa development environment.

To change the default settings of these schedulers and to customize them as per your requirements, do the following:

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

Reviewing Scheduler Details

On the Schedulers screen, for each scheduler type, you can view the following details:

Column	Description
Frequency	In this column, you can view the frequency at which the scheduler is set to run. The frequency could be the default setting or a custom setting that you edit as per your requirements. The timestamp indicates time in Coordinated Universal Time (UTC).
Last Run Status	This column indicates the outcome of the last scheduled run. You can view the "Completed" or the "Failed" status depending on whether the data collection completed without errors or encountered any issues during its last run.
Last Modified By	This column shows who last edited the scheduler settings and when. This is the user with administrative privileges. These details help track the individual responsible for any modifications made to the scheduler settings.
Next Scheduled Run	In this column, you can view the scheduled date and time of the next run for each scheduler. Here the timestamp is adjusted to reflect the time zone settings of your local browser.

This is how you can schedule data collection from various tools running in your Lazsa environment.

Recommended Topics	**What's next?**Audit Logs

Source URL: https://help.calibo.com/lazsa/content/configuration/themes_settings.htm

Themes Settings

Using **Themes**, you can customize the look and feel of the Lazsa Platform user interface to suit your company branding. With Themes settings, you can create a cohesive and personalized interface for your users. Here's how you can customize the Themes settings:

Choose your preferred color from the color palette. Based on your selection, call-to-action (CTA) buttons throughout the platform will be displayed in your preferred color, providing a consistent and branded user experience.

Switch between dark mode and light mode for the side navigation bar. Dark mode provides a sleek and modern look, while light mode offers a clean and traditional appearance. You can choose the mode that suits your preferences and enhances the visual appeal of your platform.

To further personalize your platform, you can upload two types of logos: the logo mark and the main logo. The logo mark appears in the collapsed sidebar. The main logo appears in the open sidebar, representing your brand throughout the platform. By uploading your logos, you can reinforce your company's identity and create a cohesive branding experience.

The preview pane allows you to visualize the final output of your theme settings before you apply and save them. This way, you can make informed decisions about your customization choices and ensure the desired visual impact.

By leveraging the Themes settings in the Lazsa Platform, you can create a visually appealing and branded user interface that aligns with your company's identity.

Related Topics	**What's next?** Audit Logs Settings

Related Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/account_info.htm

Account Info

You create and consume various cloud resources from within the throughout your product development life cycle. You are charged by your cloud service providers for the compute, containers, and services you use. The cloud resources and services you are charged for typically include the cloud instances you spin up, Kubernetes clusters you configure, load balancers you create within Kubernetes or OpenShift clusters, and technologies you deploy within cloud instances or virtual machines on Kubernetes clusters or OpenShift platform. It is important to keep track of and optimize these costs.

The Account Info screen provides you with a comprehensive view of costs incurred for cloud resources that are associated with products and portfolios created in the Lazsa Platform. On this screen, you can view the total cost for all the products in your Lazsa account and also view costs incurred at the feature and stage level per product in the platform.

To view the costs incurred for your products and features developed within the Lazsa Platform, do the following:

On the Account Info screen, all your products developed in the Lazsa Platform are listed along with the details such as the name of the creator of the product, date when the product was created in the platform, names of designated product owners, and the cost (in USD) associated with the product.

On this screen, you can do the following:

Generate Cost

In the upper right corner below the Help icon, click **Generate Cost** to calculate the real-time cost of all your products in the Lazsa Platform. It provides up-to-the-minute cost data for the last 12 months. The cost is displayed in USD.

Include or Exclude Products with No Technologies Configured

Turn on the **Include products having no technologies / tools configured** toggle switch to view the products in which no technologies or tools are configured. Visibility into such products helps in better project management and monitoring. It also helps you determine which projects lack necessary technology configurations. This can help in better resource allocation. If such products are no longer in use, you can delete them.

Sort Products by Text

In the **Product Name** column, you can sort product names alphabetically in ascending or descending order.

Sort Products by Date

In the **Created On** column, you can sort your products by dates (oldest to newest and newest to oldest).

View Cost at Feature or Stage Level

To view the cost incurred at the feature and stage level in a product, click **View Details** in the extreme right beside the ellipsis (...) in a product row.

On the cost details screen for the selected product, the name of the product, creator of the product, date when the product was created in the platform, and names of product owners are displayed.

- **Go to Product**: To access the product, click **Go to Product** in the upper right corner.
- **Generate Cost**: To view the real-time cost of the product, click **Generate Cost** in the upper right corner. It provides up-to-the-minute cost data for the last 12 months. The cost is displayed in USD.
- **View Cost Per Development Stage Per Feature**

  Below the total cost of the product, the features in the product are listed. Click each feature tab to view its constituent development stages. In each stage, you can view the details such as the machine configuration of the cloud instance used for feature development, technologies added, and the cost incurred per cloud instance in a stage.
- **Manage Cloud Instance Cost**

  Within a development stage, if you find that the expenses incurred by running instances are exceeding your allocated budget, you can stop a particular running instance or all running instances. You can start an instance again whenever you want.

  - To stop a single running instance within a stage, click the ellipsis (...) in the Actions column in the instance row.
  - To stop all the instances running within a stage, click the ellipsis (...) in front of the stage name.

  This enables you to manage your expenses while aligning them with your defined development budgets. By suspending instances, you can effectively curtail ongoing costs and ensure efficient financial management throughout your software development process.

  The instance status is dynamically updated at the feature stage level to reflect whether the instance is running or stopped.

Go to Product: To access the product, click **Go to Product** in the upper right corner.

Generate Cost: To view the real-time cost of the product, click **Generate Cost** in the upper right corner. It provides up-to-the-minute cost data for the last 12 months. The cost is displayed in USD.

View Cost Per Development Stage Per Feature

Below the total cost of the product, the features in the product are listed. Click each feature tab to view its constituent development stages. In each stage, you can view the details such as the machine configuration of the cloud instance used for feature development, technologies added, and the cost incurred per cloud instance in a stage.

Manage Cloud Instance Cost

Within a development stage, if you find that the expenses incurred by running instances are exceeding your allocated budget, you can stop a particular running instance or all running instances. You can start an instance again whenever you want.

- To stop a single running instance within a stage, click the ellipsis (...) in the Actions column in the instance row.
- To stop all the instances running within a stage, click the ellipsis (...) in front of the stage name.

To stop a single running instance within a stage, click the ellipsis (...) in the Actions column in the instance row.

To stop all the instances running within a stage, click the ellipsis (...) in front of the stage name.

This enables you to manage your expenses while aligning them with your defined development budgets. By suspending instances, you can effectively curtail ongoing costs and ensure efficient financial management throughout your software development process.

The instance status is dynamically updated at the feature stage level to reflect whether the instance is running or stopped.

Thus, by using the Account Info screen, you can stay on top of your cloud resource expenses and optimize your software development financial management. This helps you ensure cost-efficiency throughout your development process.

Recommended Topics	**What's next?** Platform Setup

Recommended Topics

What's next? Platform Setup

Source URL: https://help.calibo.com/lazsa/content/configuration/develop.htm

Develop

In the develop phase, you add the technologies that you want to use to develop an application and configure them. You also create a branch template which decides the branching strategy for checking in your code. After you add and configure the technologies, you are ready to start coding. Once coding is done, you deploy your technology. The diagram below shows the high-level tasks and the sequence in which you must perform the tasks in the Develop phase:

- Configure Branch Template - click this option to configure the branching strategy for your product. For more information, see Create Branch Template.
- Data Pipeline Studio - click this option if you are creating a Data & Analytics product and want to create a data pipeline for your product. For more information, see Data Pipeline Studio.
- **+ New Technologies** - click this option to add the required technologies. Select the type of application from the tab and select the appropriate front-end, back-end and testing tools. For more information, see Add and Configure Technologies. Once you have added and configured the technologies you can start coding.

Recommended Topics	**What's next?** Create Branch Template

Source URL: https://help.calibo.com/lazsa/content/configuration/agile_planning_tools.htm

Configure Connection Details of Agile Planning Tools

Agile planning tools enable you to manage agile product delivery at scale. These tools provide several features for backlog management, project planning and tracking, release management and roadmapping, and bug tracking, among others. These features boost adaptability, effective collaboration and communication and help you achieve early and continuous delivery of valuable software ensuring customer satisfaction.

Currently, the Lazsa Platform supports Jira in the Agile Planning Tools category. For the Lazsa Platform to pull data from Jira, you must provide the connection details of your active Jira subscription in the Lazsa Platform.

Note:

The Jira user account that you configure must have the Administrator Jira global permission.

Note:

The Jira user account that you configure must have the Administrator Jira global permission.

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

(After you save connection details for at least one Jira subscription, you see the **Modify** button here.)

On the **Agile Planning Tools** screen, click the **Jira** tile to configure the connection details of your active Jira accounts.

Configuration Name: Give a local name to your configuration. Your Jira connection details are saved by this name in the Lazsa Platform.

URL: Provide your Jira host URL. For example, https://testonly.atlassian.net.

Jira project creation required: If you enable this option, a new Jira project is created for a product or a portfolio depending on your preferences in Configuration > Settings > Agile Project Settings.

To provide your Jira account credentials, do one of the following:

- **Connect using Lazsa Orchestrator Agent**:

  Enable this option to resolve the user credentials for Jira within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

  Depending on your configured cloud service provider account (AWS or Microsoft Azure), you see the AWS Secrets Manager or Azure Key Vault as the secrets management tool configured for the Lazsa Orchestrator Agent.

  For AWS Secrets Manager, provide the Secret Name, Username Key, and API Token Key for your Jira account credentials.

  For Azure Key Vault, provide the Vault Name, Username Secret, and API Token Secret for your Jira account credentials.

- **Select an Identity Security Provider**

  In the **Select an Identity Security Provider** section, do one of the following:

  - Select **Lazsa** and type your Jira account username and API token. In this case, the user credentials are securely stored in the Lazsa-managed secrets store.
  - For AWS cloud environment, select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the API Token Key for the Lazsa Platform to retrieve the secrets for your Jira account.
  - For Azure cloud environment, select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and API Token Secret for the Lazsa Platform to retrieve the credential values.

Connect using Lazsa Orchestrator Agent:

Enable this option to resolve the user credentials for Jira within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

Depending on your configured cloud service provider account (AWS or Microsoft Azure), you see the AWS Secrets Manager or Azure Key Vault as the secrets management tool configured for the Lazsa Orchestrator Agent.

For AWS Secrets Manager, provide the Secret Name, Username Key, and API Token Key for your Jira account credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, and API Token Secret for your Jira account credentials.

Select an Identity Security Provider

In the **Select an Identity Security Provider** section, do one of the following:

- Select **Lazsa** and type your Jira account username and API token. In this case, the user credentials are securely stored in the Lazsa-managed secrets store.
- For AWS cloud environment, select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the API Token Key for the Lazsa Platform to retrieve the secrets for your Jira account.
- For Azure cloud environment, select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen

are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and API Token Secret for the Lazsa Platform to retrieve the credential values.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check if you can connect to the configured Jira account successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen.

That's it! You have configured the connection details of your agile planning tool account. Now, you may want to add other tools required to manage your product development life cycle.

Recommended Topics	**What's next?** Document Management Tools

What's next? Document Management Tools

Source URL: https://help.calibo.com/lazsa/content/configuration/configure_secrets_management_tools.htm

Configure Connection Details of Secrets Management Tools

As a security best practice, you store sensitive data such as database credentials, application credentials, authentication tokens, API keys, and other secrets in a secrets management tool. To access the tools and technologies in your cloud environments from within the Lazsa Platform, the platform must have authenticated access to your secrets management tool. You must provide the connection details of your secrets management tool in the Lazsa Platform and assign the read-only permissions to the Lazsa Platform in your secrets management tools.

Currently , the Lazsa Platform supports the following secrets management tools:

- AWS Secrets Manager
- Azure Key Vault

AWS Secrets Manager

Azure Key Vault

To provide the connection details of secrets management tools in the Lazsa Platform, perform these steps:

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

(After you save connection details for at least one secrets management tool, you see the **Modify** button here.)

On the **Vault Configuration** screen, click the **AWS Secrets Manager** tile or the **Azure Key Vault** tile to configure the connection properties of your active accounts for these tools.

To save the connection properties of your AWS Secrets Manager account, provide the following details:

Field	Description
Name	Give a name to your configuration. Your AWS Secrets Manager connection details are saved by this name in the Lazsa Platform.
Description	Provide a description of your configuration. When you save multiple connection details in the Lazsa Platform, a brief description always helps you identify the saved connection details easily.
Region	AWS Region that specifies where your AWS Secrets Manager resources are managed.
Master AWS Account	Calibo's Master AWS Account ID is auto-populated. You need to mention this ID in the IAM role policy that you create to allow the Lazsa Platform to access your AWS Secrets Manager. If you use the CFT for IAM role policy provided by Calibo, this ID is already mentioned in the template.
External ID	This is the unique identifier generated by Calibo. You need to mention this ID in the IAM role policy that you create to allow the Lazsa Platform to access your AWS Secrets Manager. If you use the CFT for IAM role policy provided by Calibo, this ID is already mentioned in the template.
Cross Account Role ARN	After you create an IAM role and attach a policy to establish a trusted relationship between your AWS account and Calibo's account, you can provide the ARN here.
Download CFT	Download this CloudFormation Template provided by Calibo. This template creates an IAM role and the required policy to allow the Lazsa Platform to access your AWS Secrets Manager.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check if you can connect to the configured AWS Secrets Manager account successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen.

To save the connection properties of your Azure Key Vault subscription, provide the following details :

Field	Description
Name	Give a name to your configuration. Your Azure Key Vault connection details are saved by this name in the Lazsa Platform.
Description	Provide a description of your configuration. When you save multiple connection details in the Lazsa Platform, a brief description always helps you identify the saved connection details easily.
Subscription ID	A GUID that uniquely identifies your subscription to use Azure services.
Tenant ID	Provide the Azure Active Directory tenant ID that is used for authenticating requests to the key vault.
Client ID	This is the unique Application (client) ID assigned to your app by Azure AD when the app was registered. To find your Application (Client) ID in your Azure subscription, go to Azure AD > Enterprise applications > Application ID.
Client Secret	This is the secret string that your application uses to authenticate itself while requesting a token from Azure Key Vault.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check if you can connect to the configured Azure Key Vault subscription successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen.

Related Topics	**What's next?** Configure Source Code Repository Connection Details

Related Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/add_configure_tech_stacks.htm

Add and Configure Technologies

To kickstart the development of your cloud-native application, you must first add the necessary technologies in the Develop phase of a feature of your product.

The available technology options on this screen are predefined by your Administrator. If your desired technology is not in the list, reach out to your Administrator. The choice of technologies depends on the type of the cloud-native application you are building.

Adding and Configuring Technologies

Sign in to the Lazsa Platform, click **Products** in the left navigation pane.

On the **Products** screen, choose your desired product and the feature within the product, where you want to add technologies.

In the phases of the feature, click **Develop**.

On the Develop screen, click **+ New Technologies**. If you haven't created a branch template yet, you'll be prompted to do so before adding a technology.

To create a branch template as per your branching strategy, click **Configure Branch Template**. See Create Branch Template.)

If you choose to do it later, click **Proceed**.

On the **Add Technologies** screen, the **All** tab contains a consolidated list of all supported technologies. Alternatively, you can go to specific tabs such as API, Web App, Databases, Data Integration, Data Analytics, Data Visualization, and Data Quality, among others, and choose technologies based on your application type.

Select the required technologies and click **Add**.

Here's a list of all the supported technologies in the Lazsa Platform. The availability of these technologies may vary, depending on the technologies selected by your Administrator during the initial configuration of the platform.

- **Back-End Technologies**
  - Core Java 17 - Gradle
  - Core Java - Gradle
  - Core Java 17 - Maven
  - Core Java - Maven

    - Django
    - FastAPI
    - Java18 with Spring Boot- Gradle
    - Java17 with Spring Boot - Gradle
    - Java 15 with Spring Boot - Gradle
    - Java with Spring Boot - Gradle
    - Java18 with GraphQL Spring Boot- Gradle
    - Java17 with GraphQL Spring Boot-Gradle
    - Java with GraphQL Spring Boot - Gradle
    - Java18 with GraphQL Spring Boot - Maven
    - Java17 with GraphQL Spring Boot- Maven
    - Java with GraphQL Spring Boot - Maven
    - Java18 with Spring Boot- Maven
    - Java17 with Spring Boot- Maven
    - Java 15 with Spring Boot - Maven
    - Java with Spring Boot - Maven
    - Node.js with Express
    - Python-3.11.5
    - Python-3.9.1
    - Spark QL
- **Testing Tools**
    - Rest Assured 3.0

- Core Java 17 - Gradle
- Core Java - Gradle
- Core Java 17 - Maven
- Core Java - Maven
- Django
- FastAPI
- Java18 with Spring Boot- Gradle
- Java17 with Spring Boot - Gradle
- Java 15 with Spring Boot - Gradle
- Java with Spring Boot - Gradle
- Java18 with GraphQL Spring Boot- Gradle
- Java17 with GraphQL Spring Boot-Gradle
- Java with GraphQL Spring Boot - Gradle
- Java18 with GraphQL Spring Boot - Maven
- Java17 with GraphQL Spring Boot- Maven
- Java with GraphQL Spring Boot - Maven
- Java18 with Spring Boot- Maven
- Java17 with Spring Boot- Maven
- Java 15 with Spring Boot - Maven
- Java with Spring Boot - Maven
- Node.js with Express
- Python-3.11.5
- Python-3.9.1
- Spark QL

- Rest Assured 3.0

- **Front-End Technologies**
    - Angular 14.0
    - Angular 13.2.4
    - Angular 12.0
    - Angular 11.0
    - Angular 7.0
    - Angular 5.0
    - Next.js 13
    - Next.js 12
    - Node.js 18.14 without Express
    - Node.js without Express
    - Nuxt - 3.2.2
    - Nuxt - 2.x
    - React 18.2.0
    - React 17.0.2
    - React 16.0
    - React 18.2.0 without TypeScript
    - React 17.0.2 without TypeScript
    - React 16.0 without TypeScript
    - React 18.2.0 without TypeScript - Yarn
    - React 17.0.2 without TypeScript - Yarn
    - React 18.2.0 with TypeScript - Yarn
    - React 17.0.2 with TypeScript - Yarn

    - Vue.js
- **Back-End Technologies**
    - NestJS 9
    - NestJS 7
    - SonarQube
- **Testing Tools**
    - Cucumber
    - Selenium

- Angular 14.0
- Angular 13.2.4
- Angular 12.0
- Angular 11.0
- Angular 7.0
- Angular 5.0
- Next.js 13
- Next.js 12
- Node.js 18.14 without Express
- Node.js without Express
- Nuxt - 3.2.2
- Nuxt - 2.x
- React 18.2.0
- React 17.0.2
- React 16.0
- React 18.2.0 without TypeScript
- React 17.0.2 without TypeScript
- React 16.0 without TypeScript
- React 18.2.0 without TypeScript - Yarn
- React 17.0.2 without TypeScript - Yarn
- React 18.2.0 with TypeScript - Yarn
- React 17.0.2 with TypeScript - Yarn
- Vue.js

- NestJS 9
- NestJS 7
- SonarQube

- Cucumber
- Selenium

- Amazon S3
- Amazon Aurora
- Amazon RDS for MySQL
- Amazon RDS for PostgreSQL
- Amazon RDS for Oracle
- Amazon RDS for SQL Server
- Snowflake
- Azure Database for PostgreSQL
- Azure Database for MySQL
- Azure SQL Database
- Azure Data Lake
- MySQL 5.7
- H2
- DBPedia
- PostgreSQL
- DocStore

- Debezium
- Snowflake Bulk Ingest
- Snowflake Stream Ingest
- Databricks
- Azure Data Factory
- AWS Glue
- Amazon AppFlow
- Talend
- Pentaho

- AWS SageMaker
- Drools
- Amazon Redshift
- Lazsa MLOps
- Lazsa Visualizer
- MLFlow

- Python with JupyterLab
- Python with JupyterLab - 3.0.7
- RStudio Connect
- RStudio Server
- Splunk
- Apache Flink
- Apache Hive

- Metabase
- Qlik Sense
- Tableau
- JReport
- Grafana
- Kibana

- Lazsa Data Deduplication (for Databricks)
- Lazsa Data Analyzer (for Databricks)
- Lazsa Data Analyzer (for Snowflake)
- Lazsa Data Profiler (for Databricks)
- Lazsa Data Profiler (for Snowflake)
- Lazsa Issue Resolver (for Databricks)
- Lazsa Issue Resolver (for Snowflake)
- OpenRefine

- Snowflake
- Lazsa Action

- Data Reconciliation (for Databricks)

Data Reconciliation (for Databricks)

For each technology, specify a local title and a repository name. The technology instance will be listed in the platform by the name you provide. For each technology instance you add, a separate repository is created in the Configure Source Code Repository Connection Details tool configured in the Lazsa Platform by your Administrator.

- **Start Coding** - Click this option to start coding in the Visual Studio Code source code editor.
- **Source Code** - Click this option to access your configured source code repository.
- **CI/CD pipeline** - Click this option to go to the CI/CD pipeline in the Dev stage of the Deploy phase. This option is enabled after you deploy the technology.
- **Live URL** - Click this to access the live URL of your deployed application. This option is enabled after you deploy the technology.

Now that your technologies are added and configured, you are ready to start coding! After your coding is done, you may want to deploy the technology.

Recommended Topics	**What's next?** Deploy

Recommended Topics

What's next? Deploy

Source URL: https://help.calibo.com/lazsa/content/configuration/edit_team.htm

Edit Team

You can edit details of the team and team members as well as view the history of the selected team using the Edit option.

What are the different tasks that I can perform using the Edit option?

You can send an email to all current team members and download the list of team members to a CSV file. Apart from this you can also perform the following tasks:

Team Details	Edit the following details related to the team: • Description • Link the team with an agile board • Associate a team with the defined organization hierarchy: ◦ Group ◦ Domain ◦ Product Line ◦ Product • Custom Fields
Members	• Search team members • Add member ◦ Select the member and assign a role to add a new member. ◦ Set the allocation period. ◦ Add any required custom fields. • Expand the sections for Past, Current, and Future members. Click the ellipsis (...) and **edit** or **remove** a member.
History	View details of updates done to the team like adding members, removing members and so on. Click **View Members** to know the current composition of the team.

Edit the following details related to the team:

- Description
- Link the team with an agile board
- Associate a team with the defined organization hierarchy:
  - Group
  - Domain
  - Product Line
  - Product
- Custom Fields

Description

Link the team with an agile board

Associate a team with the defined organization hierarchy:

- Group
- Domain
- Product Line
- Product

Group

Domain

Product Line

Product

Custom Fields

- Search team members
- Add member
    - Select the member and assign a role to add a new member.
    - Set the allocation period.
    - Add any required custom fields.
- Expand the sections for Past, Current, and Future members. Click the ellipsis (...) and **edit** or **remove** a member.

Search team members

Add member

- Select the member and assign a role to add a new member.
- Set the allocation period.
- Add any required custom fields.

Select the member and assign a role to add a new member.

Set the allocation period.

Add any required custom fields.

Expand the sections for Past, Current, and Future members. Click the ellipsis (...) and **edit** or **remove** a member.

View details of updates done to the team like adding members, removing members and so on.

Click **View Members** to know the current composition of the team.

Recommended Topics	**What's next?**Create Teams

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/document_management_tools.htm

Document Management Tools

Corporate wiki tool is a collaborative authoring system generally used as an enterprise knowledge base. It acts as a central database for authorized users to add, edit, store, and manage content related to company processes, projects, products, services, customer updates, and more.

Currently, the Lazsa Platform supports Confluence Cloud in the Document Management Tools category. You can link your existing Confluence wikispaces with the products you create in the Lazsa Platform or create new wikispaces from within the platform. For the Lazsa Platform to pull data from Confluence Cloud , you must provide the connection details of your active Atlassian Confluence Cloud subscription in the Lazsa Platform.

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

(After you save connection details for at least one Confluence subscription, you see the **Modify** button here.)

On the **Document Management Tools** screen, click the **Confluence** tile to configure the connection details of your active Atlassian Confluence Cloud account.

Configuration Name: Give a local name to your configuration. Your Confluence connection details are saved by this name in the Lazsa Platform.

URL: Enter your Confluence Cloud host URL. For example, https://example.confluence.com.

Allow creation of new space: If you enable this option, you can create a new wikispace in Confluence while creating or editing product details from within the Lazsa Platform.

To provide your Confluence Cloud account credentials, do one of the following:

- **Connect using Lazsa Orchestrator Agent**:

  Enable this option to resolve your Confluence Cloud credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

Depending on your configured cloud service provider account (AWS or Microsoft Azure), you see the AWS Secrets Manager or Azure Key Vault as the secrets management tool configured for the Lazsa Orchestrator Agent.

For AWS Secrets Manager, provide the Secret Name, Username Key, and API Token Key for your Confluence Cloud account credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, and API Token Secret for your Confluence Cloud account credentials.

- **Select an Identity Security Provider**

  In the **Select an Identity Security Provider** section, do one of the following:

  - Select **Lazsa** as the Identity Security Provider and type your Confluence Cloud username and API token. In this case, the user credentials are securely stored in the Lazsa-managed secrets store.
  - For AWS cloud environment, select **AWS Secrets Manager** as the Identity Security Provider. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the API Token Key for the Lazsa Platform to retrieve the secrets for your Confluence Cloud account.
  - For Azure cloud environment, select **Azure Key Vault** as the Identity Security Provider. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and API Token Secret for the Lazsa Platform to retrieve the credential values.

Connect using Lazsa Orchestrator Agent:

Enable this option to resolve your Confluence Cloud credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

Depending on your configured cloud service provider account (AWS or Microsoft Azure), you see the AWS Secrets Manager or Azure Key Vault as the secrets management tool configured for the Lazsa Orchestrator Agent.

For AWS Secrets Manager, provide the Secret Name, Username Key, and API Token Key for your Confluence Cloud account credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, and API Token Secret for your Confluence Cloud account credentials.

Select an Identity Security Provider

In the **Select an Identity Security Provider** section, do one of the following:

- Select **Lazsa** as the Identity Security Provider and type your Confluence Cloud username and API token. In this case, the user credentials are securely stored in the Lazsa-managed secrets store.
- For AWS cloud environment, select **AWS Secrets Manager** as the Identity Security Provider. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the API Token Key for the Lazsa Platform to retrieve the secrets for your Confluence Cloud account.
- For Azure cloud environment, select **Azure Key Vault** as the Identity Security Provider. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and API Token Secret for the Lazsa Platform to retrieve the credential values.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check if you can connect to the configured Confluence Cloud account successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen. While you can save and activate the connection details of as many Confluence Cloud accounts as you want, you must set only one of them as the default account. You can create new Confluence wikispaces and pages in or select the existing ones from your default account from within the Lazsa Platform. When you create a policy template for a product, you can select any configured Confluence account (including the non-default accounts) and associate it with your product.

With this, you have successfully configured the connection details of your Confluence Cloud account. Now, you may want to add other tools required to manage your product development life cycle.

Recommended Topics	**What's next?** Databases and Data Warehouses

Source URL: https://help.calibo.com/lazsa/content/configuration/configure lazsa platform.htm

Platform Setup

After the Calibo team creates your Lazsa Platform subscription and you, as an administrator, perform the initial configuration tasks in the F24H (First 24 Hours) wizard, it is time to set up the platform to make it ready for use for your digital product development teams and other stakeholders.

Configuring Lazsa Platform

Configuration Section	Function
Users, Roles, Teams, and Organization Hierarchy	Manage your organizational hierarchy, users, teams, roles, and permissions.
Lazsa Orchestrator Agents	Configure agents to connect to tools in your environment which the Lazsa Platform cannot access directly.
Cloud Platform, Tools and Technologies	Configure connection details for the tools and technologies in your software engineering environment. Configure your accounts for cloud services, container platforms, development tools, software frameworks, code and artifact repositories, collaboration and communication tools, and more.
Data Pipeline Studio	Configure data crawlers and import data ingestion catalogs.
Settings	Configure platform settings and product-specific settings such as themes, cloud instance configuration, audit logs, custom fields, schedulers, and more.
Standards	Define standard templates and processes for product development.
Maturity Assessment	Manage categories, map questions, and configure other settings for maturity assessment.

Recommended Topics	**What's next?** Cloud Platform Tools & Technologies

---

Source URL: https://help.calibo.com/lazsa/content/configuration/Data_Analytics_Platform.htm

---

Source URL: https://help.calibo.com/lazsa/content/configuration/Create_Branch_Template.htm

---

Source URL: https://help.calibo.com/lazsa/content/configuration/security_settings.htm

Security Settings

This section contains settings to ensure the overall security of the Lazsa Platform.

Currently, in this section, you can enable or disable the **Allow Two-Factor/Multi-Factor Authentication** option.

If you enable this option, you can activate the **Two-factor Authentication (2FA)** option in Configuration > Platform Setup > Security & SSO. You can then implement the OTP-based two-factor authentication.

With two-factor or multi-factor authentication, users are required to provide additional authentication information (beyond SSO credentials) to verify their identity. This makes it harder for unauthorized or malicious users to gain access to your account or data even if they could obtain user password. Thus, 2FA or MFA adds an extra layer of security for the users of the Lazsa Platform, reducing the risk of unauthorized access and protecting your users' accounts and data from various types of attacks.

Related Topics	**What's next?** Themes Settings

Related Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/source_code_repositories.htm

Configure Source Code Repository Connection Details

In today's software development landscape, effective source code management is crucial for collaborative and streamlined development processes. Source code repositories serve as central repositories to store, version, and manage the source code of software projects. Integrating these repositories with your development tools and workflows can significantly enhance productivity, code quality, and team collaboration.

You can access your code base hosted in your source code repositories from within the Lazsa Platform. In the CI/CD View in the Deploy phase, you can also review the commit history and track the progression of your code base. To establish a connection to your source code repository, the Lazsa Platform needs the connection details of the repository.

Currently, the Lazsa Platform supports the following source code repositories:

- GitLab
- Bitbucket Server
- GitHub
- Bitbucket Cloud

GitLab

Bitbucket Server

GitHub

Bitbucket Cloud

To provide the connection details of your active accounts of these source code repositories, follow these steps:

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

(After you save connection details for at least one source code repository account, you see the **Modify** button here.)

On the **Source Code Repository** screen, click the source code repository of which you want to configure the connection details.

To save the connection properties of your GitLab account, do the following:

Configuration Name: Give a local name to your configuration. Your GitLab connection details are saved by this name in the Lazsa Platform.

GitLab URL: Provide your GitLab server URL.

To provide your GitLab account credentials, do one of the following:

- **Connect using Lazsa Orchestrator Agent**:

  If you enable this option, GitLab account credentials are resolved and the communication with GitLab server happens within your private network via Lazsa Orchestrator Agent.

  Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

  For AWS Secrets Manager, provide the Secret Name, Username Key, and Private Token Key for your GitLab account credentials.

  For Azure Key Vault, provide the Vault Name, Username Secret, and Private Token Secret for your GitLab account credentials.

- **Select an Identity Security Provider**

  If you don't use the Lazsa Orchestrator Agent, you can directly provide the credentials in the configuration, or retrieve them from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

  - Select **Lazsa** and type your GitLab account Username and Private Token. In this case, credentials are securely stored in the Lazsa-managed secrets store.
  - Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Private Token Key for the Lazsa Platform to retrieve the secrets.
  - Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Private Token Secret for the Lazsa Platform to retrieve the credential values.

Connect using Lazsa Orchestrator Agent:

If you enable this option, GitLab account credentials are resolved and the communication with GitLab server happens within your private network via Lazsa Orchestrator Agent.

Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

For AWS Secrets Manager, provide the Secret Name, Username Key, and Private Token Key for your GitLab account credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, and Private Token Secret for your GitLab account credentials.

Select an Identity Security Provider

If you don't use the Lazsa Orchestrator Agent, you can directly provide the credentials in the configuration, or retrieve them from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

- Select **Lazsa** and type your GitLab account Username and Private Token. In this case, credentials are securely stored in the Lazsa-managed secrets store.
- Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Private Token Key for the Lazsa Platform to retrieve the secrets.
- Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Private Token Secret for the Lazsa Platform to retrieve the credential values.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check whether you can connect to the configured GitLab server successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen.

To save the connection properties of your Bitbucket Server account, do the following:

Configuration Name: Give a local name to your configuration. Your Bitbucket Server connection details are saved by this name in the Lazsa Platform.

Base URL: This is the root URL or address used to access the Bitbucket Server instance.

To provide your Bitbucket Server credentials, do one of the following:

- **Connect using Lazsa Orchestrator Agent**:

  If you enable this option, Bitbucket Server credentials are resolved and the communication with Bitbucket Server instance happens within your private network via Lazsa Orchestrator Agent.

  Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

  For AWS Secrets Manager, provide the Secret Name, Username Key, Password Key, and Token Key for your Bitbucket Server credentials.

  For Azure Key Vault, provide the Vault Name, Username Secret, Password Secret, and Token Secret for your Bitbucket Server credentials.

- **Select an Identity Security Provider**

  If you don't use the Lazsa Orchestrator Agent, you can directly provide the credentials in the configuration, or retrieve them from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

  - Select **Lazsa** and type your Bitbucket Server account Username, Password, and HTTP Access Token. In this case, credentials are securely stored in the Lazsa-managed secrets store.
  - Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, Password Key, and Token Key for the Lazsa Platform to retrieve the secrets.
  - Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, Password Secret, and Token Secret for the Lazsa Platform to retrieve the credential values.

  ## Note:

  If your HTTP access token expires, renew the token in Bitbucket Server and update the renewed token in your Bitbucket Server configuration in the Lazsa Platform.

Connect using Lazsa Orchestrator Agent:

If you enable this option, Bitbucket Server credentials are resolved and the communication with Bitbucket Server instance happens within your private network via Lazsa Orchestrator Agent.

Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

For AWS Secrets Manager, provide the Secret Name, Username Key, Password Key, and Token Key for your Bitbucket Server credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, Password Secret, and Token Secret for your Bitbucket Server credentials.

Select an Identity Security Provider

If you don't use the Lazsa Orchestrator Agent, you can directly provide the credentials in the configuration, or retrieve them from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

- Select **Lazsa** and type your Bitbucket Server account Username, Password, and HTTP Access Token. In this case, credentials are securely stored in the Lazsa-managed secrets store.
- Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, Password Key, and Token Key for the Lazsa Platform to retrieve the secrets.
- Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, Password Secret, and Token Secret for the Lazsa Platform to retrieve the credential values.

Note:

If your HTTP access token expires, renew the token in Bitbucket Server and update the renewed token in your Bitbucket Server configuration in the Lazsa Platform.

Note:

If your HTTP access token expires, renew the token in Bitbucket Server and update the renewed token in your Bitbucket Server configuration in the Lazsa Platform.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check whether you can connect to the configured Bitbucket Server instance successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen.

To save the connection properties of your GitHub account, do the following:

Configuration Name: Give a local name to your configuration. Your GitHub connection details are saved by this name in the Lazsa Platform.

GitHub URL: This is the root URL or address used to access the GitHub instance.

To provide your GitHub credentials, do one of the following:

- **Connect using Lazsa Orchestrator Agent**:

  If you enable this option, Bitbucket Server credentials are resolved and the communication with Bitbucket Server instance happens within your private network via Lazsa Orchestrator Agent.

  Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

  For AWS Secrets Manager, provide the Secret Name, Username Key, and Password Key, and API Token Key for your Bitbucket Server credentials.

  For Azure Key Vault, provide the Vault Name, Username Secret, Password Secret, and API Token Secret for your Bitbucket Server credentials.

- **Select an Identity Security Provider**

  If you don't use the Lazsa Orchestrator Agent, you can directly provide the credentials in the configuration, or retrieve them from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

    - Select **Lazsa** and type your GitHub account Username and Private Token. In this case, credentials are securely stored in the Lazsa-managed secrets store.
    - Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, Password Key, and Token Key for the Lazsa Platform to retrieve the secrets.
    - Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Connect using Lazsa Orchestrator Agent:

If you enable this option, Bitbucket Server credentials are resolved and the communication with Bitbucket Server instance happens within your private network via Lazsa Orchestrator Agent.

Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

For AWS Secrets Manager, provide the Secret Name, Username Key, and Password Key, and API Token Key for your Bitbucket Server credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, Password Secret, and API Token Secret for your Bitbucket Server credentials.

Select an Identity Security Provider

If you don't use the Lazsa Orchestrator Agent, you can directly provide the credentials in the configuration, or retrieve them from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

- Select **Lazsa** and type your GitHub account Username and Private Token. In this case, credentials are securely stored in the Lazsa-managed secrets store.
- Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, Password Key, and Token Key for the Lazsa Platform to retrieve the secrets.
- Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check whether you can connect to the configured Bitbucket Server instance successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen.

To save the connection properties of your Bitbucket Cloud account, do the following:

Configuration Name: Give a local name to your configuration. Your Bitbucket Cloud connection details are saved by this name in the Lazsa Platform.

Base URL: This is the root URL or the primary address used to access the Bitbucket Cloud platform. For example, https://bitbucket.org/myorganization/myrepository

To provide your Bitbucket Cloud credentials, do one of the following:

- **Connect using Lazsa Orchestrator Agent**:

  If you enable this option, Bitbucket Cloud credentials are resolved and the communication with Bitbucket Cloud instance happens within your private network via Lazsa Orchestrator Agent.

  Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

  For AWS Secrets Manager, provide the Secret Name, Username Key, and Password Key for your Bitbucket Cloud credentials.

  For Azure Key Vault, provide the Vault Name, Username Secret, and Password Secret for your Bitbucket Cloud credentials.

- **Select an Identity Security Provider**

  If you don't use the Lazsa Orchestrator Agent, you can directly provide the credentials in the configuration, or retrieve them from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

  - Select **Lazsa** and type your Bitbucket Cloud account Username and Password. In this case, credentials are securely stored in the Lazsa-managed secrets store.
  - Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and Password Key for the Lazsa Platform to retrieve the secrets.
  - Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Connect using Lazsa Orchestrator Agent:

If you enable this option, Bitbucket Cloud credentials are resolved and the communication with Bitbucket Cloud instance happens within your private network via Lazsa Orchestrator Agent.

Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

For AWS Secrets Manager, provide the Secret Name, Username Key, and Password Key for your Bitbucket Cloud credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, and Password Secret for your Bitbucket Cloud credentials.

Select an Identity Security Provider

If you don't use the Lazsa Orchestrator Agent, you can directly provide the credentials in the configuration, or retrieve them from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

- Select **Lazsa** and type your Bitbucket Cloud account Username and Password. In this case, credentials are securely stored in the Lazsa-managed secrets store.
- Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and Password Key for the Lazsa Platform to retrieve the secrets.
- Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Provide User Mapping: In accordance with GDPR regulations, username endpoints and username fields are no longer available in the Bitbucket Cloud REST APIs. Hence, mapping of Lazsa user accounts to Bitbucket Cloud user accounts through API calls is not possible. To map Lazsa user accounts to Bitbucket Cloud user accounts, do one of the following:

- Export Bitbucket Cloud users as a CSV file from the Atlassian Admin Portal and upload the file to the drop zone.

- Download the sample CSV file template available below the drop zone, update user details, and upload the file.

Export Bitbucket Cloud users as a CSV file from the Atlassian Admin Portal and upload the file to the drop zone.

Download the sample CSV file template available below the drop zone, update user details, and upload the file.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check whether you can connect to the configured Bitbucket Cloud instance successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen.

With this, you have completed the configuration of Source Code Repository! You may want to configure the other tools required for product development.

Recommended Topics	**What's next?** Collaboration Tools

What's next? Collaboration Tools

Source URL: https://help.calibo.com/lazsa/content/configuration/configuring_rdbms_amazon_redshift.htm

Configuring RDBMS - Amazon Redshift

Amazon Redshift is a fully managed cloud data warehousing solution that makes it simple and cost-effective for handling huge volumns of data.

After you save the connection details for Amazon Redshift, you can start using it as a data source in your data pipelines.

The Lazsa Platform offers various options for retrieving database credentials to establish a secure connection. You can either directly provide the credentials within the connection details, where they are securely stored in the Lazsa-managed secret manager. Alternatively, you can choose to retrieve credentials programmatically from your designated secrets management tool.

To configure the connection details of Amazon Redshift, do the following:

In the list of available database and data warehouse options, click .

In the **Details** section, provide the following details:

Field	Description
Name	Give a unique name to your Amazon Redshift configuration. This name is used to save and identify your specific Amazon Redshift connection details within the Lazsa Platform.
Description	Provide a brief description that helps you identify the purpose or context of this Amazon Redshift configuration.

In the **Configuration** section, provide the following information:

Field	Description
Select RDBMS Subtype	Select Amazon Redshift from the dropdown list.
Host	Specify the host or IP address of the server where Amazon Redshift is running.
Port	Enter the port number on which Amazon Redshift is listening for connections.
Database Name	Provide the name of a specific database within Amazon Redshift that you want to connect to.

Select Amazon Redshift from the dropdown list.

Depending on how you want to retrieve the credentials to connect to Amazon Redshift, do one of the following:

Field	Description
Connect using Lazsa Orchestrator Agent	Enable this option to resolve your Amazon Redshift credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform. Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents. • If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Amazon Redshift credentials. • If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your Amazon Redshift credentials.
Select Secret Manager	• Select **Lazsa** and type your Amazon Redshift username and password. In this case, the user credentials are securely stored in the Lazsa-managed secrets store. • Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for Amazon Redshift. • Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Enable this option to resolve your Amazon Redshift credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents.

- If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Amazon Redshift credentials.

- If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your Amazon Redshift credentials.

If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Amazon Redshift credentials.

If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your Amazon Redshift credentials.

- Select **Lazsa** and type your Amazon Redshift username and password.

  In this case, the user credentials are securely stored in the Lazsa-managed secrets store.

- Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for Amazon Redshift.
- Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Click **Save Configuration**. The configured connection details, you can see the configuration listed on the **Databases and Data Warehouses** screen.

Sample Connection Details Configured for Amazon Redshift:

Recommended Topics	**What's next?** Data Visualization

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/configuring_rdbms_oracle.htm

Configuring RDBMS - Oracle

Oracle is a robust relational database management system (RDBMS) that is a fully configurable and scalable enterprise database solution that uses a relational model for information management. Renowned for its scalability and comprehensive features,Oracle is widely used for diverse data-driven applications.

After you save the connection details for Oracle, you can start using it as a data source in your data pipelines.

The Lazsa Platform offers various options for retrieving database credentials to establish a secure connection. You can either directly provide the credentials within the connection details, where they are securely stored in the Lazsa-managed secret manager. Alternatively, you can choose to retrieve credentials programmatically from your designated secrets management tool.

To configure the connection details of Oracle, do the following:

In the **Details** section, provide the following details:

Field	Description
Name	Give a unique name to your Oracle configuration. This name is used to save and identify your specific Oracle connection details within the Lazsa Platform.
Description	Provide a brief description that helps you identify the purpose or context of this Oracle configuration.

In the **Configuration** section, provide the following information:

Field	Description
Select RDBMS Subtype	Select Oracle from the dropdown list.
Host	Specify the host or IP address of the server where Oracle is running.
Port	Enter the port number on which Oracle is listening for connections.
Database Name	Provide the name of a specific database within Oracle that you want to connect to.

Select Oracle from the dropdown list.

Depending on how you want to retrieve the credentials to connect to Oracle, do one of the following:

Field	Description

Field	Description
Connect using Lazsa Orchestrator Agent	Enable this option to resolve your Oracle credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform. Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents. • If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Oracle credentials. • If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store yourOracle credentials.
Select Secret Manager	• Select **Lazsa** and type your Oracle username and password. In this case, the user credentials are securely stored in the Lazsa-managed secrets store. • Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for Oracle. • Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Enable this option to resolve your Oracle credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents.

- If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Oracle credentials.

- If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store yourOracle credentials.

If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Oracle credentials.

If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store yourOracle credentials.

- Select **Lazsa** and type your Oracle username and password. In this case, the user credentials are securely stored in the Lazsa-managed secrets store.

- Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for Oracle.
- Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Click **Save Configuration**. The configured connection details, you can see the configuration listed on the **Databases and Data Warehouses** screen.

Sample Connection Details Configured for Oracle:

Recommended Topics	**What's next?** Data Visualization

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/configuring_rdbms_postgresql.htm

Configuring RDBMS - PostgreSQL

PostgreSQL is a powerful open-source relational database management system known for its extensibility and compliance with SQL standards. It offers advanced features, robust performance, and strong support for complex data types.

After you save the connection details for your PostgreSQL RDBMS, you can start using it as a data source in your data pipelines.

The Lazsa Platform offers various options for retrieving database credentials to establish a secure connection. You can either directly provide the credentials within the connection details, where they are securely stored in the Lazsa-managed secret manager. Alternatively, you can choose to retrieve credentials programmatically from your designated secrets management tool.

To configure the connection details of your PostgreSQL RDBMS, do the following:

Note:

The user that you configure must have the read-only access to your RDBMS.

Note:

The user that you configure must have the read-only access to your RDBMS.

In the list of available database and data warehouse options, click .

In the **Details** section, provide the following details:

Field	Description
Name	Give a unique name to your PostgreSQL configuration. This name is used to save and identify your specific Microsoft SQL Server connection details within the Lazsa Platform.
Description	Provide a brief description that helps you identify the purpose or context of this PostgreSQL configuration.

In the **Configuration** section, provide the following information:

Field	Description
Select RDBMS Subtype	Select **PostgreSQL** from the dropdown list.
Host	Specify the host or IP address of the server where your PostgreSQL RDBMS is running.
Port	Enter the port number on which your PostgreSQL RDBMS is listening for connections.
Database Name	Provide the name of a specific database within your PostgreSQL RDBMS that you want to connect to.

Select **PostgreSQL** from the dropdown list.

Depending on how you want to retrieve the credentials to connect to your PostgreSQL RDBMS, do one of the following:

Field	Description
Connect using Lazsa Orchestrator Agent	Enable this option to resolve your PostgreSQL credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform. Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents. • If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your PostgreSQL RDBMS credentials. • If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your PostgreSQL RDBMS credentials.
Select Secret Manager	• Select **Lazsa** and type your PostgreSQL username and password. In this case, the user credentials are securely stored in the Lazsa-managed secrets store. • Select **AWS Secrets Manager**. In the **Secret Management** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for your PostgreSQL RDBMS. • Select **Azure Key Vault**. In the **Secret Management** dropdown list, the Azure Key Vault configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Enable this option to resolve your PostgreSQL credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents.

- If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your PostgreSQL RDBMS credentials.
- If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your PostgreSQL RDBMS credentials.

If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your PostgreSQL RDBMS credentials.

If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your PostgreSQL RDBMS credentials.

- Select **Lazsa** and type your PostgreSQL username and password.

  In this case, the user credentials are securely stored in the Lazsa-managed secrets store.

- Select **AWS Secrets Manager**. In the **Secret Management** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for your PostgreSQL RDBMS.
- Select **Azure Key Vault**. In the **Secret Management** dropdown list, the Azure Key Vault configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Click **Save Configuration**. The configured connection details, you can see the configuration listed on the **Databases and Data Warehouses** screen.

Sample Connection Details Configured for PostgreSQL:

Recommended Topics	**What's next?** Data Visualization

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/databases_datawarehouses.htm

Databases and Data Warehouses

Databases and data warehouses play a crucial role in modern data-driven application development. They serve as organized repositories for storing, managing, and retrieving vast amounts of structured and unstructured data. They provide a structured framework for applications to efficiently interact with and retrieve information.

Databases ensure data integrity, security, and consistency, enabling developers to build robust and scalable applications. Data warehouses, on the other hand, specialize in aggregating and analyzing large datasets, facilitating informed decision-making through powerful analytics and business intelligence. Together, they form the backbone of data-centric applications, supporting functionalities ranging from transaction processing to advanced analytics and insights generation.

The Lazsa Platform is a unified interface to seamlessly connect to your databases and data warehouses. As an administrator, all you need to do is, configure the connection details of your databases and data warehouses, test the connection, and done! Your data engineers or citizen developers can then start integrating data from the configured databases and warehouses into Lazsa data pipelines. Saving the connection details is a one-time activity. Of course, you can always come back and edit your saved configurations. You can also manage access to your saved configurations.

Currently, the Lazsa Platform supports the following databases and data warehouses.

- Azure RDS
- Amazon RDS
- Amazon S3

- Amazon Appflow
- Azure Data Lake
- FTP
- Kafka Streams
- Amazon Kinesis Data Streams
- RDBMS

Azure RDS

Amazon RDS

Amazon S3

Amazon Appflow

Azure Data Lake

FTP

Kafka Streams

Amazon Kinesis Data Streams

RDBMS

Recommended Topics	**What's next?** Data Visualization

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/configuring_rdbms_mysql.htm

Configuring RDBMS - MySQL

MySQL is an open-source relational database management system (RDBMS) known for its reliability, ease of use, and widespread adoption. It supports structured query language (SQL) and is commonly used for web applications and various data-driven projects..

After you save the connection details for your MySQL RDBMS , you can start using it as a data source in your data pipelines.

The Lazsa Platform offers various options for retrieving database credentials to establish a secure connection. You can either directly provide the credentials within the connection details, where they are securely stored in the Lazsa-managed secret manager. Alternatively, you can choose to retrieve credentials programmatically from your designated secrets management tool.

To configure the connection details of your MySQL RDBMS, do the following:

Note:

The user that you configure must have the read-only access to your RDBMS.

Note:

The user that you configure must have the read-only access to your RDBMS.

In the list of available database and data warehouse options, click .

In the **Details** section, provide the following details:

Field	Description
Name	Give a unique name to your MySQL configuration. This name is used to save and identify your specific MySQL connection details within the Lazsa Platform.
Description	Provide a brief description that helps you identify the purpose or context of this MySQL configuration.

In the **Configuration** section, provide the following information:

Field	Description
Select RDBMS Subtype	Select **MySQL** from the dropdown list.
Host	Specify the host or IP address of the server where your MySQL RDBMS is running.

Field	Description
Port	Enter the port number on which your MySQL is listening for connections.
Database Name	Provide the name of a specific database within your MySQL RDBMS that you want to connect to.

Select **MySQL** from the dropdown list.

Depending on how you want to retrieve the credentials to connect to your MySQL RDBMS, do one of the following:

Field	Description
Connect using Lazsa Orchestrator Agent	Enable this option to resolve your MySQL credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform. Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents. • If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your MySQL credentials. • If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your MySQL credentials.
Select Secret Manager	• Select **Lazsa** and type your MySQL username and password. In this case, the user credentials are securely stored in the Lazsa-managed secrets store. • Select **AWS Secrets Manager**. In the **Secret Management** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for your MySQL RDBMS. • Select **Azure Key Vault**. In the **Secret Management** dropdown list, the Azure Key Vault configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Enable this option to resolve your MySQL credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents.

- If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your MySQL credentials.
- If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your MySQL credentials.

If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your MySQL credentials.

If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your MySQL credentials.

- Select **Lazsa** and type your MySQL username and password.

  In this case, the user credentials are securely stored in the Lazsa-managed secrets store.

- Select **AWS Secrets Manager**. In the **Secret Management** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for your MySQL RDBMS.

- Select **Azure Key Vault**. In the **Secret Management** dropdown list, the Azure Key Vault configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the

configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Click **Save Configuration**. The configured connection details, you can see the configuration listed on the **Databases and Data Warehouses** screen.

Sample Connection Details Configured for MySQL:

Recommended Topics	**What's next?** Data Visualization

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/data_visualization.htm

Data Visualization

You can add tools required for Data Visualization along with other required tools to develop a Data Analytics application.

- Name
- Description
- Qlik Sense URL
- Qlik Sense API Token
- Secure connection details with password

Now that your Data Visualization tech stack is added, you can add the other required tools.

Recommended Topics	**What's new?** Configure Technologies and Testing Tools

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/configure_technologies_testing_tools.htm

Configure Technologies and Testing Tools

In the process of digital and data product development, developers require specific front-end and back-end technologies along with testing tools to build robust and innovative solutions. The Lazsa Platform supports a comprehensive set of best-in-breed modern tools and technologies for developers. You can pick the technology stack that suits your cloud-native application development requirements. Based on your selection on the Technologies & Testing Tools screen, the technology stack is available to developers in the Develop phase in the Lazsa Platform. The technology stack is updated on a regular basis to make the latest technologies available.

To select the desired technology stack, follow these steps:

Tools and Technologies Supported in Lazsa

- **Back-End Technologies**
  - Core Java 17 - Gradle
  - Core Java - Gradle
  - Core Java 17 - Maven
  - Core Java - Maven
  - Django
  - FastAPI
  - Java18 with Spring Boot- Gradle
  - Java17 with Spring Boot - Gradle
  - Java 15 with Spring Boot - Gradle
  - Java with Spring Boot - Gradle
  - Java18 with GraphQL Spring Boot- Gradle
  - Java17 with GraphQL Spring Boot-Gradle
  - Java with GraphQL Spring Boot - Gradle
  - Java18 with GraphQL Spring Boot - Maven
  - Java17 with GraphQL Spring Boot- Maven
  - Java with GraphQL Spring Boot - Maven
  - Java18 with Spring Boot- Maven
  - Java17 with Spring Boot- Maven
  - Java 15 with Spring Boot - Maven

    - Java with Spring Boot - Maven
    - Node.js with Express
    - Python-3.11.5
    - Python-3.9.1
    - Spark QL
- **Testing Tools**
    - Rest Assured 3.0

- Core Java 17 - Gradle
- Core Java - Gradle
- Core Java 17 - Maven
- Core Java - Maven
- Django
- FastAPI
- Java18 with Spring Boot- Gradle
- Java17 with Spring Boot - Gradle
- Java 15 with Spring Boot - Gradle
- Java with Spring Boot - Gradle
- Java18 with GraphQL Spring Boot- Gradle
- Java17 with GraphQL Spring Boot-Gradle
- Java with GraphQL Spring Boot - Gradle
- Java18 with GraphQL Spring Boot - Maven
- Java17 with GraphQL Spring Boot- Maven
- Java with GraphQL Spring Boot - Maven
- Java18 with Spring Boot- Maven
- Java17 with Spring Boot- Maven
- Java 15 with Spring Boot - Maven
- Java with Spring Boot - Maven
- Node.js with Express
- Python-3.11.5
- Python-3.9.1
- Spark QL

- Rest Assured 3.0

- **Front-End Technologies**
    - Angular 14.0
    - Angular 13.2.4
    - Angular 12.0
    - Angular 11.0
    - Angular 7.0
    - Angular 5.0
    - Next.js 13
    - Next.js 12
    - Node.js 18.14 without Express
    - Node.js without Express
    - Nuxt - 3.2.2
    - Nuxt - 2.x
    - React 18.2.0
    - React 17.0.2
    - React 16.0
    - React 18.2.0 without TypeScript
    - React 17.0.2 without TypeScript
    - React 16.0 without TypeScript
    - React 18.2.0 without TypeScript - Yarn
    - React 17.0.2 without TypeScript - Yarn
    - React 18.2.0 with TypeScript - Yarn
    - React 17.0.2 with TypeScript - Yarn
    - Vue.js
- **Back-End Technologies**
    - NestJS 9
    - NestJS 7
    - SonarQube
- **Testing Tools**
    - Cucumber
    - Selenium

- Angular 14.0
- Angular 13.2.4
- Angular 12.0
- Angular 11.0
- Angular 7.0
- Angular 5.0

- Next.js 13
- Next.js 12
- Node.js 18.14 without Express
- Node.js without Express
- Nuxt - 3.2.2
- Nuxt - 2.x
- React 18.2.0
- React 17.0.2
- React 16.0
- React 18.2.0 without TypeScript
- React 17.0.2 without TypeScript
- React 16.0 without TypeScript
- React 18.2.0 without TypeScript - Yarn
- React 17.0.2 without TypeScript - Yarn
- React 18.2.0 with TypeScript - Yarn
- React 17.0.2 with TypeScript - Yarn
- Vue.js

- NestJS 9
- NestJS 7
- SonarQube

- Cucumber
- Selenium

- Amazon S3
- Amazon Aurora
- Amazon RDS for MySQL
- Amazon RDS for PostgreSQL
- Amazon RDS for Oracle
- Amazon RDS for SQL Server
- Snowflake
- Azure Database for PostgreSQL
- Azure Database for MySQL
- Azure SQL Database
- Azure Data Lake
- MySQL 5.7
- H2
- DBPedia
- PostgreSQL
- DocStore

- Debezium
- Snowflake Bulk Ingest
- Snowflake Stream Ingest
- Databricks
- Azure Data Factory
- AWS Glue
- Amazon AppFlow
- Talend
- Pentaho

- AWS SageMaker
- Drools
- Amazon Redshift
- Lazsa MLOps
- Lazsa Visualizer
- MLFlow
- Python with JupyterLab
- Python with JupyterLab - 3.0.7
- RStudio Connect
- RStudio Server
- Splunk
- Apache Flink
- Apache Hive

- Metabase
- Qlik Sense
- Tableau
- JReport
- Grafana
- Kibana

- Lazsa Data Deduplication (for Databricks)
- Lazsa Data Analyzer (for Databricks)
- Lazsa Data Analyzer (for Snowflake)
- Lazsa Data Profiler (for Databricks)
- Lazsa Data Profiler (for Snowflake)
- Lazsa Issue Resolver (for Databricks)
- Lazsa Issue Resolver (for Snowflake)
- OpenRefine

- Snowflake
- Lazsa Action

- Data Reconciliation (for Databricks)

Data Reconciliation (for Databricks)

Now that you have selected the tools and technologies, you may want to add other tools required to manage your product development life cycle.

Recommended Topics	**What's next?** DevOps CI/CD Pipeline Configuration

Recommended Topics

What's next? DevOps CI/CD Pipeline Configuration

Source URL: https://help.calibo.com/lazsa/content/configuration/collaboration_tools.htm

Collaboration Tools

You can either configure a collaboration tool of your choice or use the tool managed by the Lazsa Platform.

Configuring a collaboration tool

- Configuration Name
- App ID
- Client ID
- Client Secret
- Username
- Password
- Enable Secure connection details with password to secure your configuration details

This completes the configuration of your collaboration tool! Now you may want to add other tools required to manage your product development life cycle.

Recommended Topics	**What's next?** Cloud Platform, Tools and Technologies

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/configuring_rdbms_microsoft_sql_server.htm

Configuring RDBMS - Microsoft SQL Server

Microsoft SQL Server is a robust relational database management system (RDBMS) that provides efficient storage, retrieval, and management of structured data. Renowned for its scalability and comprehensive features, MS SQL Server is widely used for diverse data-driven applications.

After you save your MS SQL Server connection details in the Lazsa Platform, you can start using it as a data source in your data pipelines.

The Lazsa Platform offers various options for retrieving your database credentials to establish a secure connection. You can either directly provide the credentials within the connection details, where they are securely stored in the Lazsa-managed secret manager. Alternatively, you can choose to retrieve credentials programmatically from your designated secrets management tool.

To configure the connection details of your Microsoft SQL Server, do the following:

Note:

The user that you configure must have the read-only access to your RDBMS.

Note:

The user that you configure must have the read-only access to your RDBMS.

In the list of available database and data warehouse options, click .

In the **Details** section, provide the following details:

Field	Description
Name	Give a unique name to your Microsoft SQL Server configuration. This name is used to save and identify your specific Microsoft SQL Server connection details within the Lazsa Platform.
Description	Provide a brief description that helps you identify the purpose or context of this Microsoft SQL Server configuration.

In the **Configuration** section, provide the following information:

Field	Description
Select RDBMS Subtype	Select Microsoft SQL Server from the dropdown list.
Host	Specify the host or IP address of the server where your Microsoft SQL Server is running.
Port	Enter the port number on which your Microsoft SQL Server is listening for connections.
Database Name	Provide the name of a specific database within your Microsoft SQL Server that you want to connect to.

Select Microsoft SQL Server from the dropdown list.

Depending on how you want to retrieve the credentials to connect to your Microsoft SQL Server, do one of the following:

Field	Description
Connect using Lazsa Orchestrator Agent	Enable this option to resolve your Microsoft SQL Server credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform. Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents. • If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Microsoft SQL Server credentials. • If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your Microsoft SQL Server credentials.
Select Secret Manager	• Select **Lazsa** and type your Microsoft SQL Server username and password. In this case, the user credentials are securely stored in the Lazsa-managed secrets store. • Select **AWS Secrets Manager**. In the **Secret Management** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for your Microsoft SQL Server. • Select **Azure Key Vault**. In the **Secret Management** dropdown list, the Azure Key Vault configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Enable this option to resolve your Microsoft SQL Server credentials within your private network via Lazsa Orchestrator Agent without sharing them with the Lazsa Platform.

Select the Lazsa Orchestrator Agent that you want to use from the list of your configured agents.

- If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Microsoft SQL Server credentials.

- If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your Microsoft SQL Server credentials.

If you select an agent installed in an Amazon EKS cluster, the secrets management tool AWS Secrets Manager is auto-selected. Provide the name and the key of the secret where you store your Microsoft SQL Server credentials.

If you select an agent installed in an Azure AKS cluster, the secrets management tool Azure Key Vault is auto-selected. Provide the Vault Name and the name of the secret where you store your Microsoft SQL Server credentials.

- Select **Lazsa** and type your Microsoft SQL Server username and password.

  In this case, the user credentials are securely stored in the Lazsa-managed secrets store.

- Select **AWS Secrets Manager**. In the **Secret Management** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password Key for the Lazsa Platform to retrieve the secrets for your Microsoft SQL Server.
- Select **Azure Key Vault**. In the **Secret Management** dropdown list, the Azure Key Vault configurations that you save and activate in the Secret Management section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password Secret for the Lazsa Platform to retrieve the credential values.

Click **Save Configuration**. The configured connection details, you can see the configuration listed on the **Databases and Data Warehouses** screen.

Sample Connection Details Configured for Microsoft SQL Server:

Recommended Topics	**What's next?** Data Visualization

Recommended Topics

Source URL: https://help.calibo.com/lazsa/content/configuration/devops_cicd_pipeline_configuration.htm

DevOps CI/CD Pipeline Configuration

DevOps CI/CD pipeline tools refer to software tools and platforms that facilitate the implementation and management of Continuous Integration (CI) and Continuous Deployment (CD) practices within DevOps workflows. These tools are designed to automate and streamline the process of building, testing, and deploying software applications, enabling teams to deliver high-quality software more efficiently and reliably. They offer a range of features and functionalities such as source code management, build automation, testing and quality assurance, deployment automation, continuous monitoring, and orchestration and workflow management, among others.

Currently, the Lazsa Platform supports Jenkins in the DevOps CI/CD Pipeline Configuration category. To automate the continuous building, testing, and deployment of your apps through Jenkins from within the Lazsa Platform, you must provide the connection details of your active Jenkins user account as explained in the following steps:

Sign in to the Lazsa Platform and click **Configuration** in the left navigation pane.

(After you save connection details for at least one Jenkins user account, you see the **Modify** button here.)

On the **DevOps CI/CD Pipeline Configuration** screen, click the **Jenkins** tile to configure the connection details of your active Jenkins user account.

Configuration Name: Give a local name to your configuration. Your Jenkins server connection details are saved by this name in the Lazsa Platform.

Jenkins URL: Provide your Jenkins server URL.

To provide your Jenkins account credentials, do one of the following:

- **Connect using Lazsa Orchestrator Agent**:

  If you enable this option, your Jenkins account credentials are resolved and communication with your Jenkins server happens within your private network via Lazsa Orchestrator Agent.

  Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

For AWS Secrets Manager, provide the Secret Name, Username Key, and Password or Token Key for your Jenkins account credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, and Password or Token Secret for your Jenkins account credentials.

- **Select an Identity Security Provider**

  If you don't use the Lazsa Orchestrator Agent, you can directly provide your Jenkins username and password or authentication token in the configuration, or retrieve it programmatically from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

  - Select **Lazsa** and type your Jenkins account username and password or API token. In this case, your Jenkins user credentials are securely stored in the Lazsa-managed secrets store.
  - Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password or Token Key for the Lazsa Platform to retrieve the secrets for your Jenkins account.
  - Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password or Token Secret for the Lazsa Platform to retrieve the credential values.

Connect using Lazsa Orchestrator Agent:

If you enable this option, your Jenkins account credentials are resolved and communication with your Jenkins server happens within your private network via Lazsa Orchestrator Agent.

Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

For AWS Secrets Manager, provide the Secret Name, Username Key, and Password or Token Key for your Jenkins account credentials.

For Azure Key Vault, provide the Vault Name, Username Secret, and Password or Token Secret for your Jenkins account credentials.

Select an Identity Security Provider

If you don't use the Lazsa Orchestrator Agent, you can directly provide your Jenkins username and password or authentication token in the configuration, or retrieve it programmatically from a secrets management tool of your choice (such as AWS Secrets Manager or Azure Key Vault). Do one of the following:

- Select **Lazsa** and type your Jenkins account username and password or API token. In this case, your Jenkins user credentials are securely stored in the Lazsa-managed secrets store.
- Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Secret Name, Username Key, and the Password or Token Key for the Lazsa Platform to retrieve the secrets for your Jenkins account.
- Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the Vault Name, Username Secret, and Password or Token Secret for the Lazsa Platform to retrieve the credential values.

Add Jenkins jobs to run from Lazsa

You can trigger your existing Jenkins job builds, pass parameters, and retrieve job status and results from within the Lazsa Platform. All you need to do is, enable the **Add Jenkins jobs to run from Lazsa** option and add your Jenkins job details and the required parameters in this section. You can then run your configured Jenkins jobs from the Deploy phase of your product development cycle.

To make your Jenkins job management easier and for reusability, you can store the values of your job parameters in the Lazsa Platform. However, if you mark a parameter as sensitive, its value is not stored in the platform. You must enter the values of sensitive parameters when you trigger a job from the Deploy phase.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check if you can connect to the configured Jenkins server successfully.

After you save and activate the configured connection details, you can see them listed on the Cloud Platform, Tools & Technologies screen.

With this, your Jenkins connection details are successfully configured in the Lazsa Platform. Now, you may want to configure the connection details of other tools involved in your PDLC.

Recommended Topics	**What's next?** Configure Kubernetes Cluster Connection Details

Source URL: https://help.calibo.com/lazsa/content/configuration/kubernetes_cluster_configuration.htm

Configure Kubernetes Cluster Connection Details

You can deploy your containerized applications on a Kubernetes cluster from within the Lazsa Platform. Here we assume that you already have a running Kubernetes cluster. To access this cluster from within the Lazsa Platform, you must configure the cluster connection details as described in the following steps:

(After you save your first cluster connection details, you see the **Modify** button here.)

- **Configuration Name**: Give a name to your configuration. Your Kubernetes cluster connection details are saved by this name in the Lazsa Platform.
- **Description**: Provide a description of your configuration. When you save multiple connection details in the Lazsa Platform, a brief description always helps you identify the saved connection details easily.
- Use one of the following options to provide authentication details of your Kubernetes cluster:
  Use Token

  With this option, you can use the authentication token to authenticate to your Kubernetes cluster. Perform these steps to enter the cluster details and use the token options.

  1. Do one of the following:

     - **Fetch cluster connection properties from configured cloud account**

       1. To fetch the details (such as name, URL, and certificate authority data) of a Kubernetes cluster running in your cloud account that you have configured in the Lazsa Platform, enable the **Use from a service provider** option.
       2. Select the cloud service provider: AWS or Microsoft Azure.
       3. Depending on your selection in the previous step, the AWS or Azure cloud accounts that you have configured in the Lazsa Platform are available for selection in the **Cloud Account** list.
       4. Names of Kubernetes clusters that you have configured in the selected cloud account are available for selection in the **Cluster Name** list.
       5. After you select the cluster name, the URL to connect to this cluster, and its certificate authority data are auto-populated.

     - **Enter connection properties manually**

       To add connection details of a Kubernetes cluster other than the ones running on the cloud accounts configured in the Lazsa Platform, keep the **Use from a service provider** option disabled, and manually enter the Cluster Name, URL, and the Certificate Authority data for the cluster.

  2. Do one of the following:

     - To resolve the token within your private network via Lazsa Orchestrator Agent without sharing it with the Lazsa Platform, enable the **Connect using Lazsa Orchestrator Agent** option.

       Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

       For AWS Secrets Manager, provide the name and the key of the secret where you store the authentication token.

       For Azure Key Vault, provide the vault name and the name of the secret where you have stored the token.

     - In the **Select an Identity Security Provider** section, do one of the following:

       - Select **Lazsa** and in the **Token** field, provide the authentication token. In this case, the token is securely stored in the Lazsa-managed secrets store.
       - Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the name and the key of the secret where you have stored the token.
       - Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the vault name and the name of the secret where you have stored the token.

  Upload Kubeconfig File

  Kubeconfig file is a YAML file that contains the Kubernetes cluster details, certificate authority data, and the secret token to authenticate the cluster. You can use a Kubeconfig file to establish a connection with your running Kubernetes cluster from within the Lazsa Platform. To connect with your Kubernetes cluster by using the Kubeconfig file, do the following:

1. Provide the name of your running Kubernetes cluster that you want to access from within the Lazsa Platform.

2. Depending on where you want to store the Kubeconfig file, do one of the following:

   - **Connect using** Lazsa Orchestrator Agent

     If you enable this option, the details in the Kubeconfig file are resolved and the communication with the cluster happens within your private network via Lazsa Orchestrator Agent.

     Currently, we support AWS Secrets Manager as the default secrets management tool to store your Kubeconfig file. Provide the name of the secret in AWS Secrets Manager where you store the Kubeconfig file details.

     Note:

     Currently, Azure Key Vault is not supported for the **Upload Kubeconfig File** option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

   - **Select an Identity Security Provider**

     If you don't use the Lazsa Orchestrator Agent, you can directly upload the Kubeconfig file in the configuration, or retrieve it from AWS Secrets Manager. Do one of the following:

     - Select **Lazsa** and upload the Kubeconfig file in the drop zone. In this case, the Kubeconfig file is securely stored in the Lazsa-managed secrets store.
     - Select **AWS Secrets Manager** as the Identity Security Provider. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. In the **Secret Name** field, provide the name of the secret where you have stored the Kubeconfig file as plain text.

       Note:

       Currently, Azure Key Vault is not supported for the Upload Kubeconfig File option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

- To password-protect your Kubernetes cluster connection details, enable the **Secure configuration details with a password** option, enter a password, and then retype it to confirm.

  This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

- Click **Test Connection** to check if you can connect to the configured Kubernetes cluster successfully.

- After you save and activate the configured connection details, you can see them listed on the **Cloud Platform, Tools & Technologies** screen.

With this option, you can use the authentication token to authenticate to your Kubernetes cluster. Perform these steps to enter the cluster details and use the token options.

Do one of the following:

- **Fetch cluster connection properties from configured cloud account**

  1. To fetch the details (such as name, URL, and certificate authority data) of a Kubernetes cluster running in your cloud account that you have configured in the Lazsa Platform, enable the **Use from a service provider** option.

  2. Select the cloud service provider: AWS or Microsoft Azure.

  3. Depending on your selection in the previous step, the AWS or Azure cloud accounts that you have configured in the Lazsa Platform are available for selection in the **Cloud Account** list.

  4. Names of Kubernetes clusters that you have configured in the selected cloud account are available for selection in the **Cluster Name** list.

  5. After you select the cluster name, the URL to connect to this cluster, and its certificate authority data are auto-populated.

- **Enter connection properties manually**

  To add connection details of a Kubernetes cluster other than the ones running on the cloud accounts configured in the Lazsa Platform, keep the **Use from a service provider** option disabled, and manually enter the Cluster Name, URL, and the Certificate Authority data for the cluster.

Fetch cluster connection properties from configured cloud account

To fetch the details (such as name, URL, and certificate authority data) of a Kubernetes cluster running in your cloud account that you have configured in the Lazsa Platform, enable the **Use from a service provider** option.

Select the cloud service provider: AWS or Microsoft Azure.

Depending on your selection in the previous step, the AWS or Azure cloud accounts that you have configured in the Lazsa Platform are available for selection in the **Cloud Account** list.

Names of Kubernetes clusters that you have configured in the selected cloud account are available for selection in the **Cluster Name** list.

After you select the cluster name, the URL to connect to this cluster, and its certificate authority data are auto-populated.

Enter connection properties manually

To add connection details of a Kubernetes cluster other than the ones running on the cloud accounts configured in the Lazsa Platform, keep the **Use from a service provider** option disabled, and manually enter the Cluster Name, URL, and the Certificate Authority data for the cluster.

Do one of the following:

- To resolve the token within your private network via Lazsa Orchestrator Agent without sharing it with the Lazsa Platform, enable the **Connect using Lazsa Orchestrator Agent** option.

  Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

  For AWS Secrets Manager, provide the name and the key of the secret where you store the authentication token.

  For Azure Key Vault, provide the vault name and the name of the secret where you have stored the token.

- In the **Select an Identity Security Provider** section, do one of the following:

  - Select **Lazsa** and in the **Token** field, provide the authentication token. In this case, the token is securely stored in the Lazsa-managed secrets store.
  - Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the name and the key of the secret where you have stored the token.
  - Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the vault name and the name of the secret where you have stored the token.

To resolve the token within your private network via Lazsa Orchestrator Agent without sharing it with the Lazsa Platform, enable the **Connect using Lazsa Orchestrator Agent** option.

Depending on the secrets management tool that you configure for the Lazsa Orchestrator Agent, you see the options for AWS Secrets Manager or Azure Key Vault.

For AWS Secrets Manager, provide the name and the key of the secret where you store the authentication token.

For Azure Key Vault, provide the vault name and the name of the secret where you have stored the token.

In the **Select an Identity Security Provider** section, do one of the following:

- Select **Lazsa** and in the **Token** field, provide the authentication token. In this case, the token is securely stored in the Lazsa-managed secrets store.
- Select **AWS Secrets Manager**. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the name and the key of the secret where you have stored the token.
- Select **Azure Key Vault**. In the **Vault Configuration** dropdown list, the Azure Key Vault configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. Provide the vault name and the name of the secret where you have stored the token.

Kubeconfig file is a YAML file that contains the Kubernetes cluster details, certificate authority data, and the secret token to authenticate the cluster. You can use a Kubeconfig file to establish a connection with your running Kubernetes cluster from within the Lazsa Platform. To connect with your Kubernetes cluster by using the Kubeconfig file, do the following:

Provide the name of your running Kubernetes cluster that you want to access from within the Lazsa Platform.

Depending on where you want to store the Kubeconfig file, do one of the following:

- **Connect using** Lazsa Orchestrator Agent

  If you enable this option, the details in the Kubeconfig file are resolved and the communication with the cluster happens within your private network via Lazsa Orchestrator Agent.

  Currently, we support AWS Secrets Manager as the default secrets management tool to store your Kubeconfig file. Provide the name of the secret in AWS Secrets Manager where you store the Kubeconfig file details.

  ### Note:

  Currently, Azure Key Vault is not supported for the **Upload Kubeconfig File** option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

- **Select an Identity Security Provider**

  If you don't use the Lazsa Orchestrator Agent, you can directly upload the Kubeconfig file in the configuration, or retrieve it from AWS Secrets Manager. Do one of the following:

  - Select **Lazsa** and upload the Kubeconfig file in the drop zone. In this case, the Kubeconfig file is securely stored in the Lazsa-managed secrets store.
  - Select **AWS Secrets Manager** as the Identity Security Provider. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. In the **Secret Name** field, provide the name of the secret where you have stored the Kubeconfig file as plain text.

    ### Note:

    Currently, Azure Key Vault is not supported for the Upload Kubeconfig File option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

Connect using Lazsa Orchestrator Agent

If you enable this option, the details in the Kubeconfig file are resolved and the communication with the cluster happens within your private network via Lazsa Orchestrator Agent.

Currently, we support AWS Secrets Manager as the default secrets management tool to store your Kubeconfig file. Provide the name of the secret in AWS Secrets Manager where you store the Kubeconfig file details.

Note:

Currently, Azure Key Vault is not supported for the **Upload Kubeconfig File** option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

Note:

Currently, Azure Key Vault is not supported for the **Upload Kubeconfig File** option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

Select an Identity Security Provider

If you don't use the Lazsa Orchestrator Agent, you can directly upload the Kubeconfig file in the configuration, or retrieve it from AWS Secrets Manager. Do one of the following:

- Select **Lazsa** and upload the Kubeconfig file in the drop zone. In this case, the Kubeconfig file is securely stored in the Lazsa-managed secrets store.
- Select **AWS Secrets Manager** as the Identity Security Provider. In the **Vault Configuration** dropdown list, the AWS Secrets Manager configurations that you save and activate in the Vault Configuration section on the Cloud Platform, Tools & Technologies screen are listed for selection. Select the configuration of your choice. In the **Secret Name** field, provide the name of the secret where you have stored the Kubeconfig file as plain text.

  ### Note:

  Currently, Azure Key Vault is not supported for the Upload Kubeconfig File option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

Note:

Currently, Azure Key Vault is not supported for the Upload Kubeconfig File option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

Note:

Currently, Azure Key Vault is not supported for the Upload Kubeconfig File option. If you use Azure Key Vault, go ahead with the **Use Token** option instead.

This is optional but recommended. When you share the connection details with multiple users, password protection helps you ensure authorized access to the connection details.

Click **Test Connection** to check if you can connect to the configured Kubernetes cluster successfully.

After you save and activate the configured connection details, you can see them listed on the **Cloud Platform, Tools & Technologies** screen.

With this, you are all set to connect with your Kubernetes cluster from within the Lazsa Platform. You can select this cluster when you deploy your tech stack. You may want to configure the next tool required for your product development.

Recommended Topics	**What's next?** Configure Source Code Repository Connection Details

Recommended Topics