Maestro

Technology description

May 2023

Version 1.3

CONTENT

1 GENERAL

Maestro application – is a fault-tolerant system for managing distributed hyper-converged virtual infrastructures. The system is based on event-driven architecture leveraging AMQP protocol (Advanced Message Queuing Protocol), powered by RabbitMQ software application for working with message queuing.

The system contains a private agent component, which is based on a cloud abstraction level. Due to this, the differences, derived from specifics of rendering services by various cloud providers, are hidden.

The cloud abstraction layer allows to manage different virtualiser types via the same code:

- OpenStack (Nova-API)
- CloudStack (CloudStack API)
- Huawei (Open API)
- VMware (vCloudDirector)
- VMware (VSphere)

The application is designed to work in a geographically distributed system, under high load and high availability (99.999%) requirements.

The Maestro application architecture scheme is described in Annex A. Architecture Framework.

User Interface examples are presented in Annex B. User Interface.

Design principles

Maestro application is developed based on the **API First** (Application Programming Interface) model.

As part of this concept, on the initial stage of the application development, a certain program interface (API) is created for the system. Then the necessary modules are built on top of the interface. Thus, on top of the Maestro program interface an SDK (Software Development Kit) is provided. The SDK is provided as a unified components integration library.

An extensible system of plug-ins allows Maestro application to add new cloud providers as well as new program interfaces (API) within one month. The system is designed as an open architecture framework which corresponds to OPC principle (Open Closed Principle): open for expanding, closed for changing.

During development, advanced methods as well as the concept of control inversion are used, in which during the construction of the program its parts are called from the shared library. Following this concept greatly simplifies the process of expanding the capabilities of the system.

The software development kit (**SDK**) is implemented based on JSON-RPC remote procedure call protocol.

Business logic

The business logic of the application is built based on microservice architecture, which allows to scale each component of the system individually, depending on the common load of the system.

Innovation

The main innovation of Maestro solution is the ability of the system to be used as SaaS (Software as a Service) based on a serverless architecture built on the AWS Lambda service.

Meanwhile, the same code base is used to build the on-premise version.

2 MAESTRO DEPLOYMENT MODELS

Maestro application can be provided in one of three configurations (deployment models), each of them contains a certain set of functionality for the end user.

- **The Standard Deployment** model allows users to get access to public virtual cloud providers (AWS, Microsoft Azure, Google Cloud Platform). Available functions are:
    - Unified and simply organized reporting for all customer's resources across all public clouds they use
    - A set of analytics tools for all virtual resources under the customer's account
    - Quotas management tool, that allows to set up the monthly expense limits for virtual infrastructures
    - Alerts and notifications that will inform the customer on the significant events on their resources
- **The Professional Deployment model** allows users to get access to public virtual cloud providers (AWS, Microsoft Azure, Google Cloud Platform). Available functions are:
    - All facilities included in the Standard model
    - Virtual servers management
    - Using "Infrastructure as Code" tools - Terraform, AWS CloudFormation
    - Auto configuration
- **The Enterprise model** allows users to get access to public virtual cloud providers (AWS, Microsoft Azure, Google Cloud Platform) and private regions located on OpenStack and VMware platforms**.**
  All functions included in the Professional Deployment model are available, and applicable to both public and private clouds.

3 MAESTRO PROVISIONING MODELS

The Maestro application can be provisioned to the users in one of the following options:

- **SaaS** (Software as a Service). The software is hosted in cloud and is provided to the user by subscription. The user can connect his account to the application and get access to the functionality within the requested deployment model.
- **SaaS + Privat Agent**. The agent is installed in a user's private region (OpenStack or VMware) to enable Maestro control over it. The agent ensures that Maestro, hosted in cloud, is connected to the customer's private cloud. Due to agent settings, Maestro receives only the information, approved by the customer. If the customer also has infrastructures in public clouds, they can be added to Maestro and managed according to the requested deployment model.
- **On Premise**. The Maestro application is installed locally on an isolated instance in the customer's enterprise data center.

Innovative in the On Premise model is the fact that the cloud-based application can be installed in a closed perimeter, without any other cloud services installed. The Maestro application is deployed in private clouds and is focused on protecting the management perimeter and preventing interactions with external cloud service providers. It is achieved due to the correct level of abstractions and a special application construction procedure and the ability to install the system in a closed perimeter without access.

4 USER INTERFACE DESIGN

A **dynamic form construction** protocol was implemented for Maestro based on Angular 8 and Native Script web application development platforms.

The protocol uses meta-elements of the application interface. This allows using component templates and the logic of their interaction on the interface.

Thus, the system allows one to get a web application and a mobile application for iOS and Android based on the source code.

The user interface supports localization. The following languages are currently available:

- English
- Portuguese
- Russian
- Spanish

5 ACCESS CONTROL

For access control, a consolidated unified role-based access control system (RBAC) is used. It is built on group policies, which are dynamically calculated during user authorization according to the information stored in Azure AD.

6 STORING AND DISPLAYING DATA

The Maestro application uses a MongoDB document-oriented database to store data. It allows to store in the original form the information provided by various cloud service providers in various unstructured formats.

An important feature of the MongoDB database is that it allows to store quite large amounts of data.

Information provided by various cloud service providers is displayed in a single unified format (as is) on a specific unified user interface.

ANNEX A. ARCHITECTURE FRAMEWORK

The Maestro application architecture framework is shown on Figure 1:

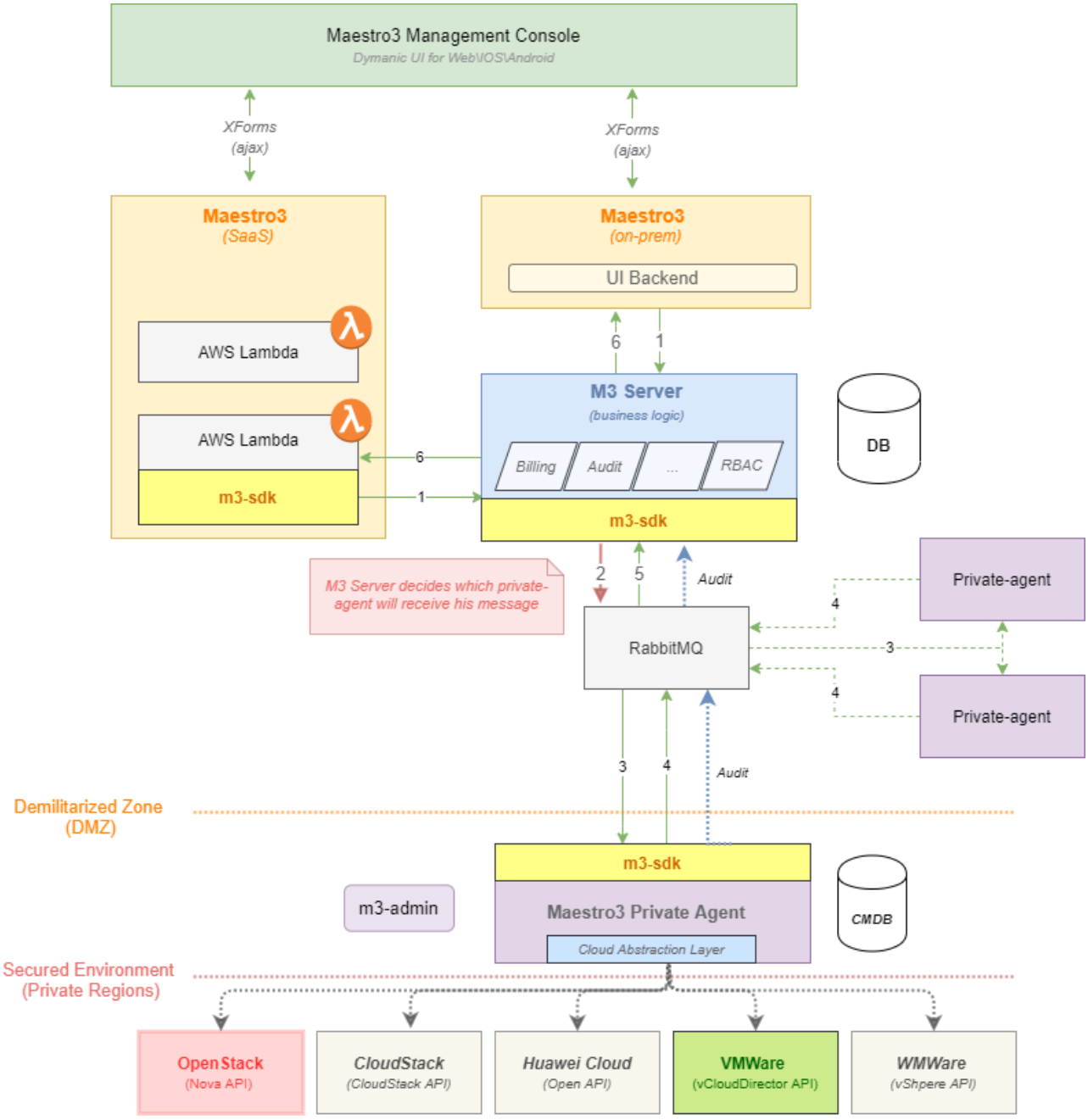

Figure 1 – Maestro architecture framework

All system components are the software manufacturer's in-house development, except for:

- RabbitMQ
- MongoDB (used as a "database" component)
- API of supported public virtual cloud providers and private cloud creation platforms

The main external service that ensures the work of the Maestro application – AWS Lambda.

ANNEX B. USER INTERFACE (BASIC ELEMENTS)

MAESTRO ACTIONS DASHBOARD

Maestro Actions Dashboard allows users to promptly access all main application management tools.

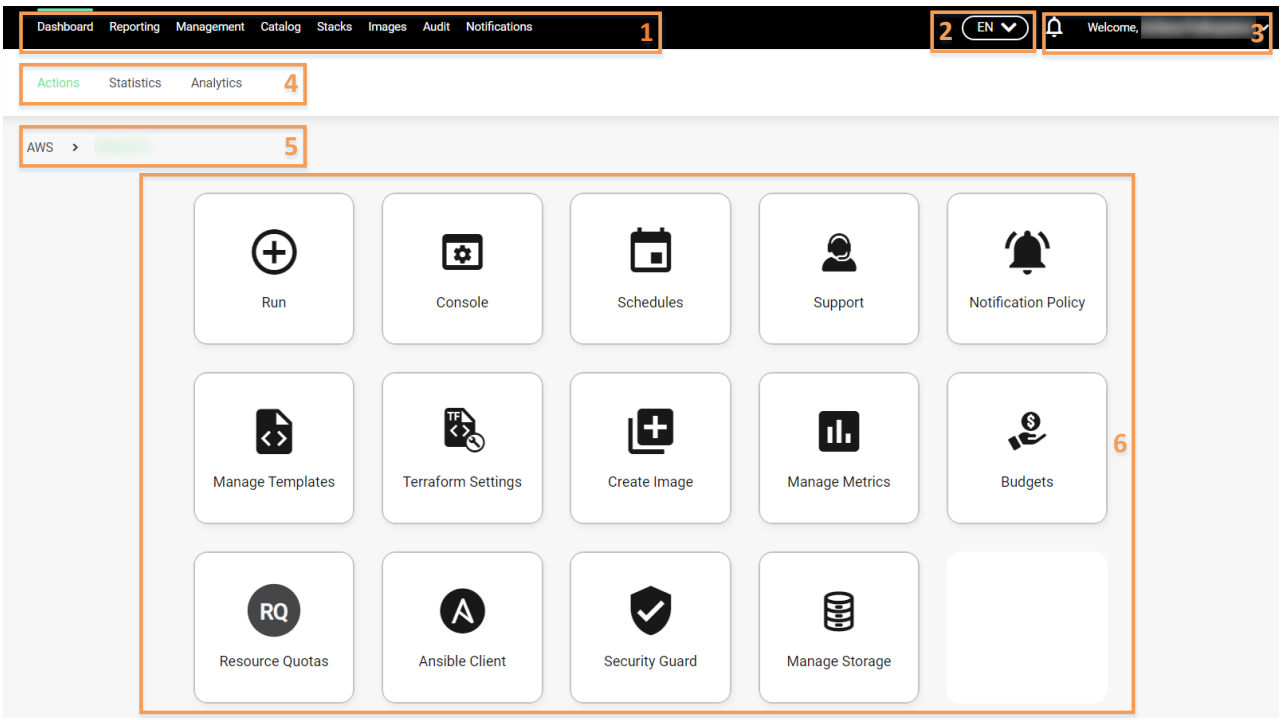

Figure 2 – Maestro Actions Dashboard

The Maestro Actions Dashboard contains the following elements:

1. Menu
2. Language selector
3. User menu
4. Tab sub-menu
5. Breadcrumbs
6. Dashboard controls

MAESTRO ANALYTICS DASHBOARD

Maestro Analytics Dashboard allows users to review the general statistics of the selected tenant. It includes financial, security, optimization points, as well as the possibility to react on deviations and deep dive into specific values detail.

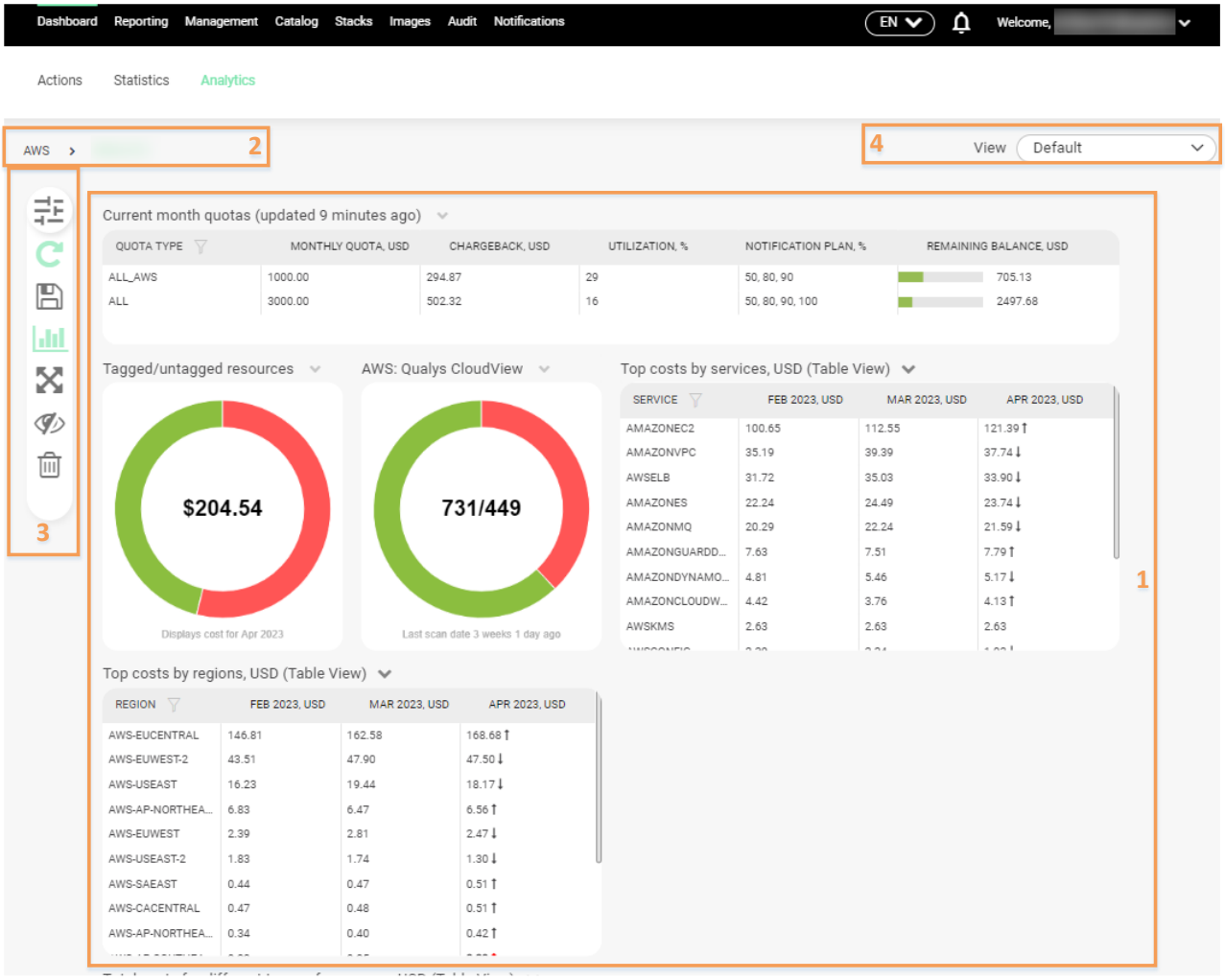

Figure 3 – Maestro Analytics Dashboard

The Maestro Analytics Dashboard contains the following elements:

1. Main Dashboard body
2. Tenant selector
3. Dashboard settings
4. Views selector

REPORTING

The “**Reporting**” tab allows you to view financial reports for selected tenants, filter information about accounts for certain resource groups by tags (tags) and send reports of the selected configuration to the email.

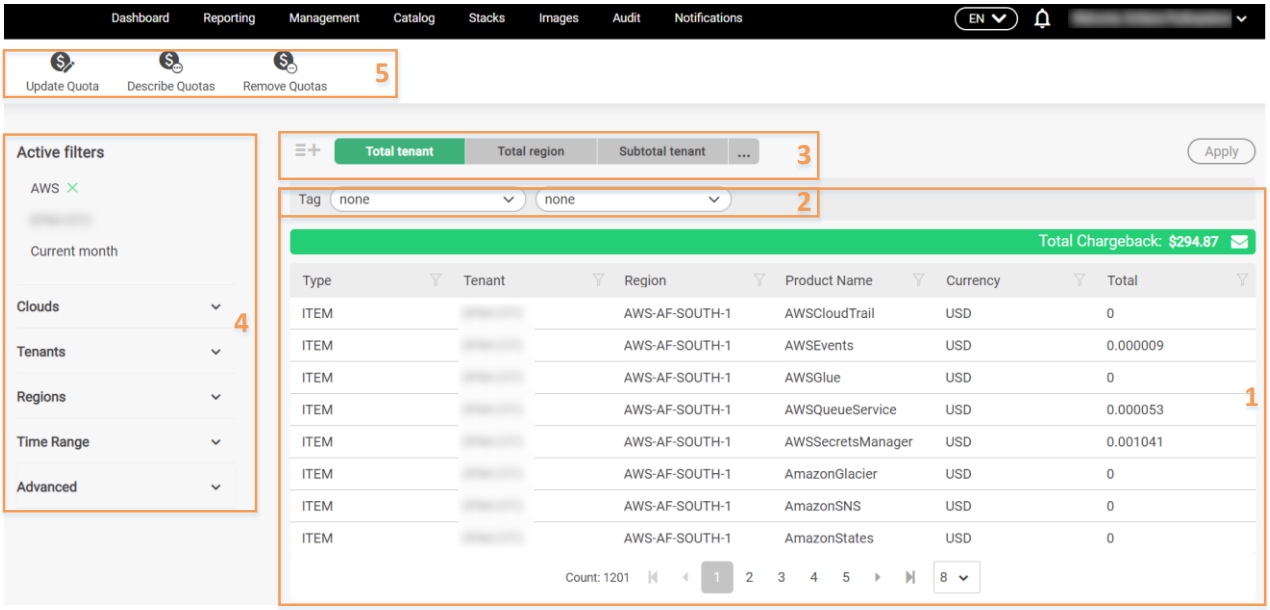

Figure 4 – “Reporting” page

The “Reporting” page contains the following elements:

1. The report for the specified period
2. Tag filter
3. Report scope
4. Report settings
5. Reporting sub-menu

NOTIFICATIONS

The “**Notifications**” tab contains all messages sent by the Maestro application to the current user.

The menu of the page allows you to adjust the filtering by the date of sending notifications and their content.

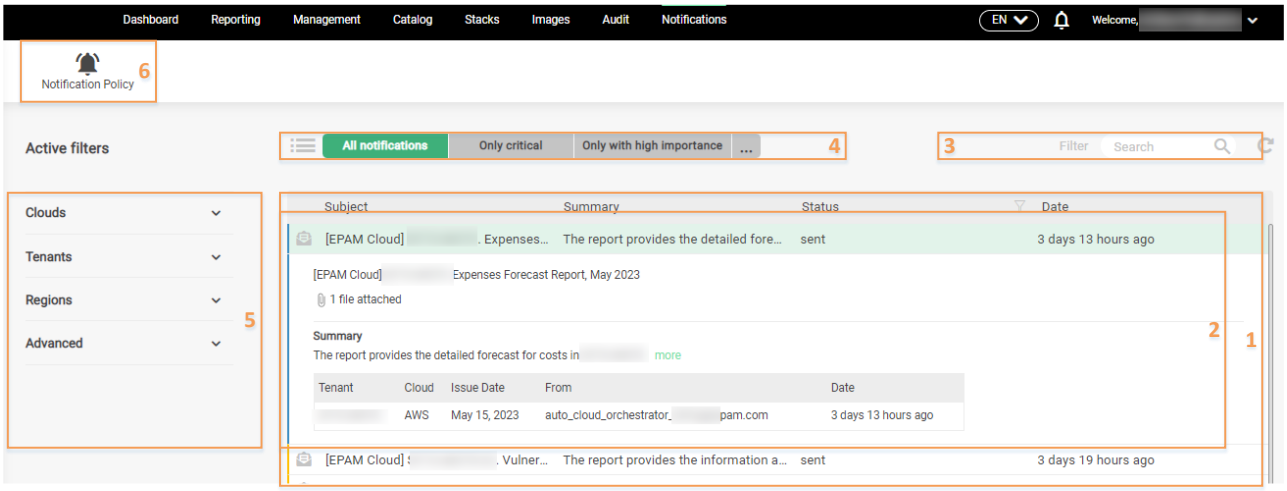

Figure 5 – “Notifications” tab

The “Notifications” tab contains the following elements:

1. The list of received notifications
2. The preview of a selected notification
3. Search in notifications
4. Notification types selector
5. Notification filters by details
6. Notifications policy wizard in the Notifications tab sub-menu

WIZARD

Maestro allows users to manage cloud resources with a set of Wizards that provide a unified set of tools for all supported providers.

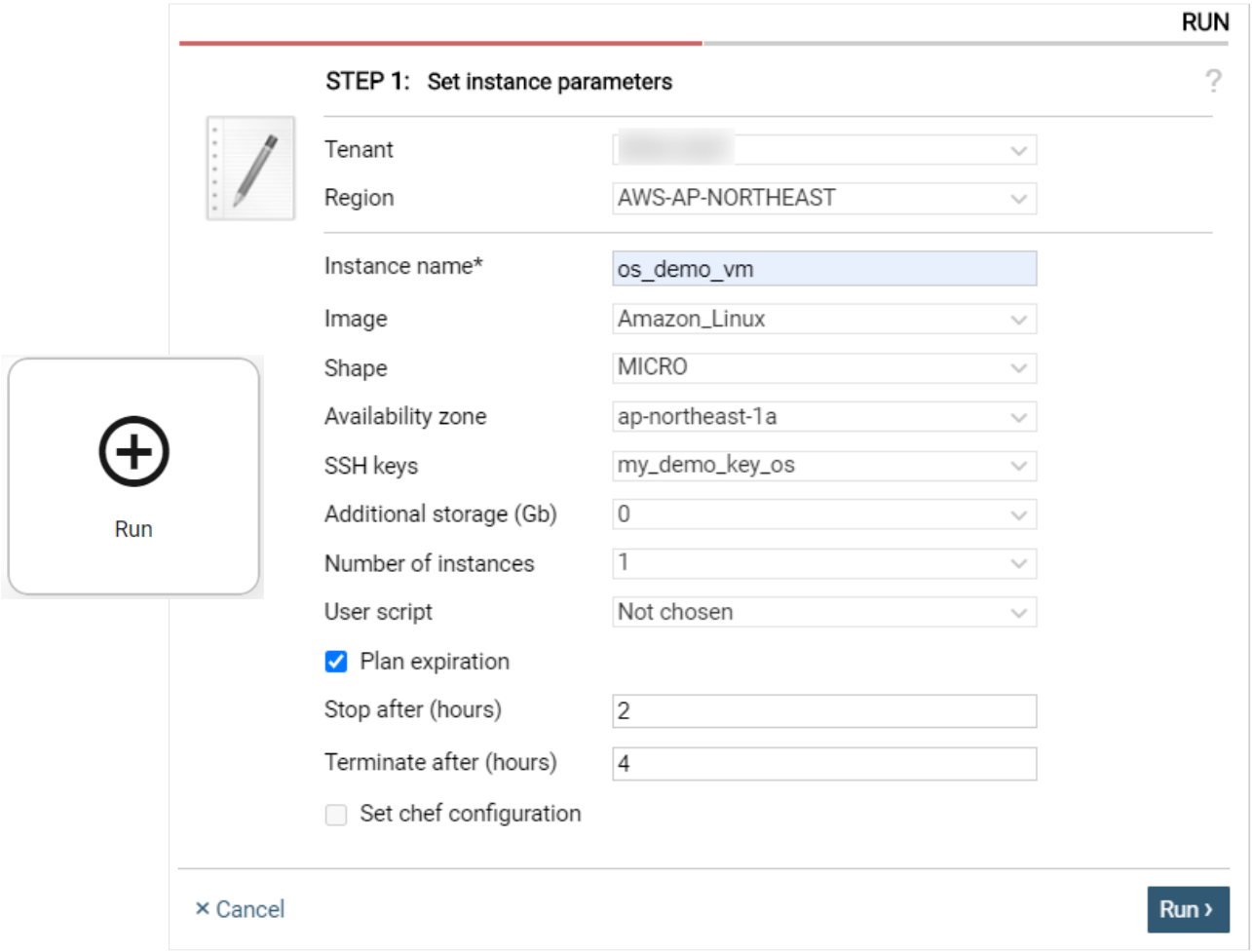

Figure 6 – Wizard for running new virtual instances

VERSION HISTORY

Version	Date	Summary
1.3	18 May, 2023	General details updated, screenshots updated
1.2	12 August, 2022	Minor details updated, screenshots updated
1.1	1 June, 2021	Screenshots updated
1.0	10 June, 2020	First published