

|  |
| --- |
| Solution Architecture Document  DLab project |
|  |

| REVISION HISTORY | | | | | |
| --- | --- | --- | --- | --- | --- |
| Ver. | Description of Change | Author | Date | Approved | |
| Name | Effective Date |
| 1.0 | Initial version of the document | Oleh Hrynyk | 29-Aug-2016 |  | 29-Aug-2016 |
| 1.1 | Priorities changes after conversation with Paul Gesiak | Paul Gesiak | 31-Aug-2016 |  | 31-Aug-2016 |
| 1.2 | Major changes including architecture | Oleh Hrynyk | 13-Sep-2016 |  |  |
| 1.3 | Structure changes, moved AD to deployment section | Kostiantyn Kudriavtsev | 13-Sep-2016 |  |  |
| 1.4 | Added User Scenarios | Dmytro Liaskovskyi | 14-Sep-2016 |  |  |
| 1.5 | Added assumptions and constrains | Kostiantyn Kudriavtsev | 15-Sep-2016 |  |  |
| 1.6 | Added Domain and Data model, more details to Self-service component | Kostiantyn Kudriavtsev | 15-Sep-2016 |  |  |

| Related Artifacts | |
| --- | --- |
| Ref. | Name |
|  |  |
|  |  |

Contents

[DLab project 1](#_Toc461719981)

[1 Executive Summary 4](#_Toc461719982)

[2 Introduction 4](#_Toc461719986)

[Definitions, Acronyms, Abbreviations 4](#_Toc461719987)

[2.1.1 GOALS 4](#_Toc461719988)

[2.1.2 SCOPE 5](#_Toc461719989)

[2.1.3 MAJOR BUSINESS DRIVERS 5](#_Toc461719990)

[3 Requirements 6](#_Toc461719991)

[Stakeholder 6](#_Toc461719992)

[Constraints 6](#_Toc461719993)

[Assumptions 6](#_Toc461719994)

[Functional requirements 6](#_Toc461719995)

[3.1.1 Roles in the system 6](#_Toc461719996)

[3.1.2 Can perform data exploration on exploratory environment by the means of analytical tool UI. Web UI wireframe 7](#_Toc461719997)

[3.1.3 Basic User Scenarios and Workflows 10](#_Toc461719998)

[Non-functional requirements 12](#_Toc461719999)

[3.1.4 Self-service UI 12](#_Toc461720000)

[3.1.5 Private exploratory environment 12](#_Toc461720001)

[3.1.6 Limited resources allocation 12](#_Toc461720002)

[3.1.7 Configurable template 13](#_Toc461720003)

[3.1.8 Configurable identity provider 13](#_Toc461720004)

[3.1.9 Cloud agnostic 13](#_Toc461720005)

[4 Quality Attributes 14](#_Toc461720007)

[5 Baseline Solution architecture 15](#_Toc461720008)

[High-level solution structure 15](#_Toc461720009)

[5.1.1 INITIAL SETUP PACKAGE - OLEH 15](#_Toc461720010)

[5.1.2 SELF-SERVICE NODE – KOSTIA 16](#_Toc461720012)

[5.1.3 EXPLORATORY ENVIRONMENT 17](#_Toc461720025)

[SOLUTION IN AWS CLOUD 18](#_Toc461720026)

[Solution components structure 19](#_Toc461720027)

[5.1.4 SELF-SERVICE UI COMPONENT 19](#_Toc461720028)

[5.1.5 PROVISION SERVICE - VOLODYMYR 20](#_Toc461720034)

[5.1.6 BACKEND 20](#_Toc461720035)

[5.1.7 Authentication and Authorization approach 20](#_Toc461720036)

[Domain model 21](#_Toc461720037)

[Data model 21](#_Toc461720038)

[5.1.8 Management node MongoDB 21](#_Toc461720039)

[5.1.9 Templates on Management node 22](#_Toc461720040)

[High-level deployment approach 23](#_Toc461720041)

[5.1.10 DEPLOYMENT SERVER - OLEH 24](#_Toc461720042)

[6 Technology stack 25](#_Toc461720043)

[Architecture Constraints and Limitations 25](#_Toc461720044)

# Executive Summary

# Introduction

Current project is infrastructure deployment agnostic platform for data analytics, science and machine learning. Platform greatly simplifies and reduces costs for infrastructure preparation and configuration, thus provides with possibility to kick-off project in short-term.

Platform can be deployed on various cloud and hypervisor providers. Currently in scope of MVP AWS platform support shall be implemented, meanwhile Azure support is planned for next release.

Future roadmap is to fully support major public and private cloud providers.

## Definitions, Acronyms, Abbreviations

|  |  |
| --- | --- |
| Abbreviation or Acronym | Definition |
| MVP | Minimum viable product (MVP) is a product with just enough features to gather validated learning about the product and its |
|  |  |



### GOALS

Major goals are to build:

Self-service infrastructure

* Users in charge of provisioning their own environment.
* Minimal support of IT time required

Fail-safe environment

* Sandbox - you can break it
* Built within client’s security perimeter
* Client’s security policies and access controls applied

Exploratory Environment

* Best open source data tools
* Scalable compute
* Data and metadata store
* Training / workshops

Collaborative Data Science Workflow

Collaboration within data science team

Collaboration between data science and engineering team

Fast track from RND to Production

Customizable

Extensible architecture, which allows various components (i.e. analytical toolbox templates, analytical tools) to be added into infrastructure.

### SCOPE

In scope of MVP following features are planned:

Full support of AWS cloud provider

Automated creation/management of AWS infrastructure

Automated provision/management of Exploratory Environment:

Jupyter with Scala, Python and R kernel support

Local Apache Spark computation tool based on EMR libraries

Provision of EMR cluster with Spark

Provision of private storage based on S3 buckets

Implement web-based user interface for Exploratory Environment Management.

### MAJOR BUSINESS DRIVERS

Problem with DATA ACCESS

* No access to production data
* Restricted access to production and raw data
* Data access requires IT assistance

Problem with TOOLS

* Only sql-based tools available on the cluster (RDBMS, hive, impala, etc…)
* Latest tools available only on local machines – difficult cooperation within a team
* Different tools used by data science and engineering teams – difficult cooperation between teams
* Installation of tools is difficult and time consuming (complex security policies, no experience)
* Tools present, but unstable (misconfiguration, insufficient permissions, etc...)

Problem with SUPPORT

* 1 full time operation engineer required for each 2-3 data scientist (for efficient workflow)
* Data science team restricted to local machines, due to lack of IT support time
* Data science team relies on continuous support of IT ops team in most of daily activities

# Requirements

## Stakeholder

|  |  |  |  |
| --- | --- | --- | --- |
| Stakeholder Role | Stakeholder Name | Key concern | Requirements area |
| Sponsor, Product Owner | Valentin Tsitlik | Scope, costs, timelines | Functional and non-functional |
| Product owner, End user | Paul Gesiak | Scope | Functional and non-functional |
| Software Licensing Officer | Konstantin Boudnik | Legal | Software licensing |
| Account Manager, SA | Oleh Hrynets | Costs, scope, overall architecture, timelines | Functional and non-functional |
| Delivery manager, SA | Kostiantyn Kudriavtsev | Cost, scope, overall architecture, timelines | Functional and non-functional |
| SA | Oleh Hrynyk | Overall architecture |  |
| Project manager | Dmytro Liaskovskyi | Cost, timelines |  |

## Constraints

The following constrains must be taken into account

|  |  |  |
| --- | --- | --- |
| Num. | Constraint | Description |
| C1 | Open source stack | Solution is going to be open sourced and released under Apache 2.0 license, thus only components compatible with this license must be used. Pre-approved licenses includes Apache 2.0 and MIT, all other components must be approved by Software Licensing Officer |
|  |  |  |
|  |  |  |

## Assumptions

|  |  |
| --- | --- |
| Num. | Assumption description |
| A1 | LDAP will be available as identity provider to be used by product for user authentication and authorization |
| A2 | Customer will provide account in cloud with enough permission to support product functionality |
| A3 | Docker is allowed to be used by customer’s policies |
|  |  |

## Functional requirements

### Roles in the system

Following system roles are currently envisioned:

* Administrator / DevOps
  + Performs initial infrastructure setup.
  + Have SSH access to Self-Service node, thus can change Self-Service configuration.
* Data Scientist

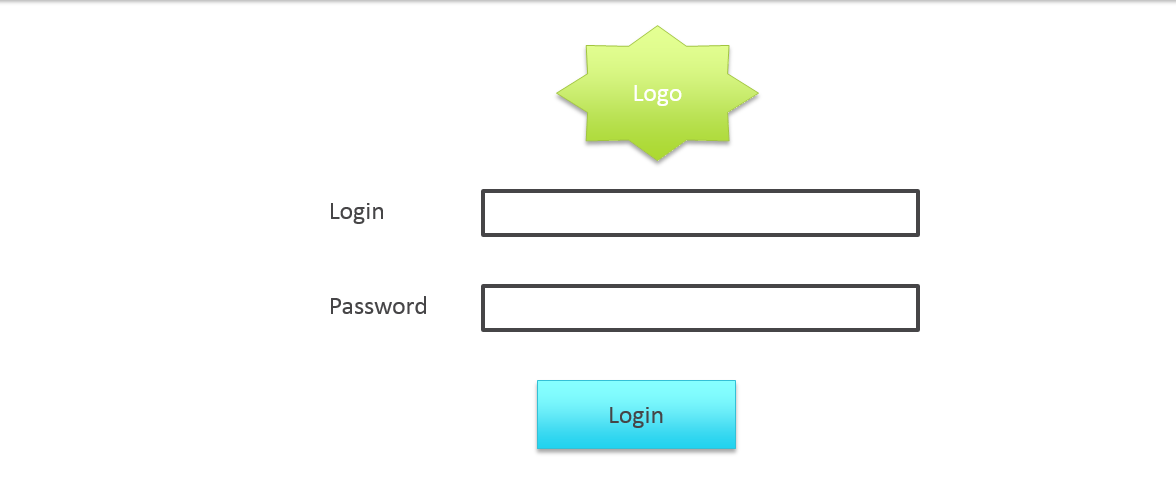
Can create / manage own exploratory environment via Self-Service Web UI.

### Can perform data exploration on exploratory environment by the means of analytical tool UI. Web UI wireframe

Web UI is expected to be used by desktop users on different platform and with different browsers. That’s why an expectation is to get responsible web UI which would provide equal experience for Windows, Mac and Linux users in the following browsers:

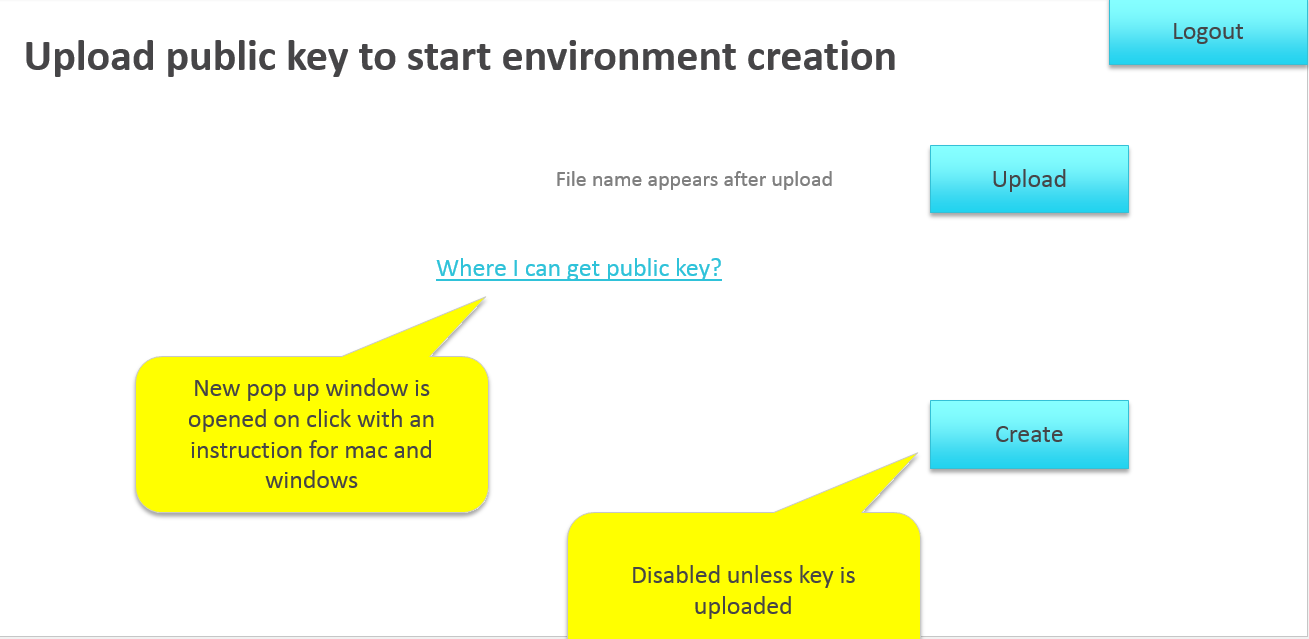
* MS IE 10+
* Google Chrome 42+
* Firefox 35+
* Safari 9.1.2+ only on Mac

The login page is very simple and allows users to be validated against LDAP, look at Figure 1. In case of authentication error, corresponding validation error message should appear above login above Login button.



Figure

When user logs in first time (Figure 2), he/she expected to be asked to provide public SSH key and only after the environment will be created (in this save, bucket on S3, subnet in VPN).



Figure

A button “Create” is inactive until key is loaded.

There is a link “Where I can get public key?” which opens new tab in browser with instruction how to generate public/private SSH keys on Windows and Mac and what exactly expected from user. It must be clear for non-technical users. After clicking on “Create” button “wait until load” message should show up, which will wait backend to perform all preparation steps and redirect user to page Figure 3 List resources when it’s ready.

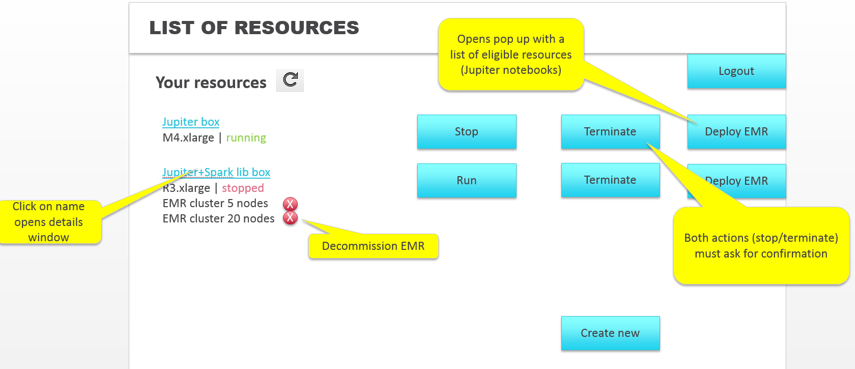


Figure List resources

Also, the page Figure 3 List resources is displayed to user with preconfigured env (subnet, storage, etc., so user is not the first time here and already registered vital components). On this page user can see short description of running resources like resource name, resource size (shape for EC2 instance and number of slave’s X shape of slave for EMR) and status of resource (which can be *running* or *stopped*). There are 2 actions possible on these resources:

* Create new Notebook server.
  + After Data Scientist defines needed template and instance shape following actions are available:
    - Create – creates new notebook server with defined configuration. After user hits Create Button Data Scientist is redirected to List resources page with “creating” status. While “creating” all button related to Notebook server, like Stop, Terminate, Deploy EMR are disabled. They become enabled as soon as status changes to “running”.
    - Cancel button closes the popup.

List of possible statuses of Notebook server provisioning:

* + - Creating – until Notebooks server gets created
    - Running
    - Terminating – after Terminate button is pressed. Resource disappears from the scree as soon as Termination process is completed.
    - Stopping – after Stop button is pressed. Changes to Stopped once done.

If analytical node has attached EMR clusters, it’s showed under the node and each of them can be terminated independently.

At the bottom, there is *Create new* button, which leads to new page or popup with an information required to create a new analytical tool and depictured on Figure 4 Create new analytical tool

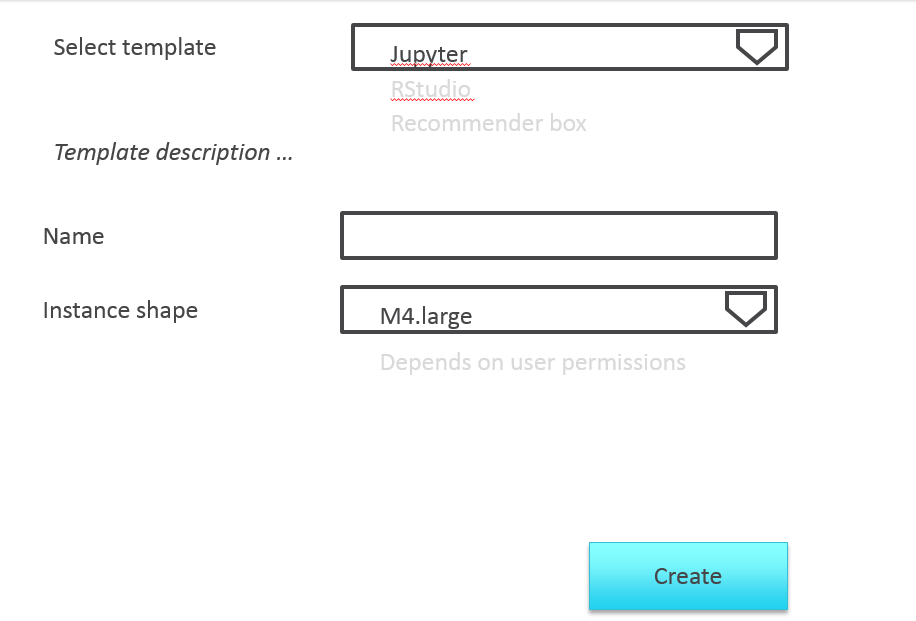


Figure Create new analytical tool

Back to Figure 3 List resources, each analytical node has a button Deploy EMR which allows to create and attached EMR with specific number of nodes to this service as shown in expected popup

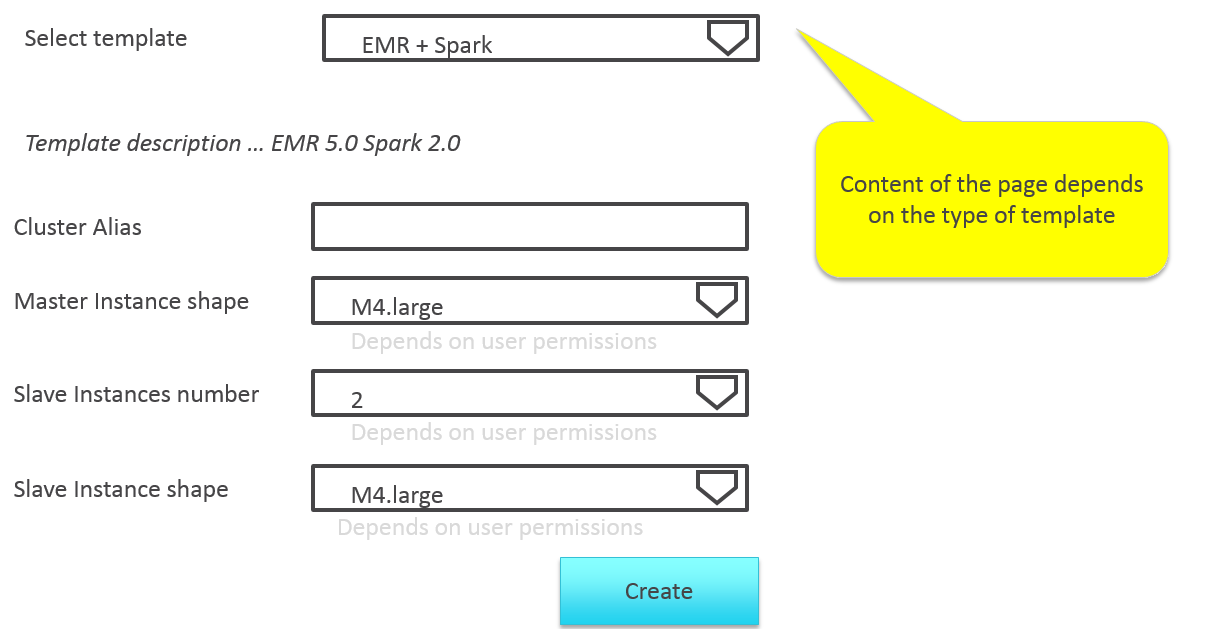
Figure 5 Create computational resources 

Figure Create computational resources

Cluster Alias field should be prefilled with username and Data Scientists adds only alias as suffix.

On the fly in-field validation should prevents from inputting of special characters. List of special chars – TBD..

Click on each resource listed on Figure 3 List resources must show details about box and instruction how to connect to this node, as shown on Figure 6 Details

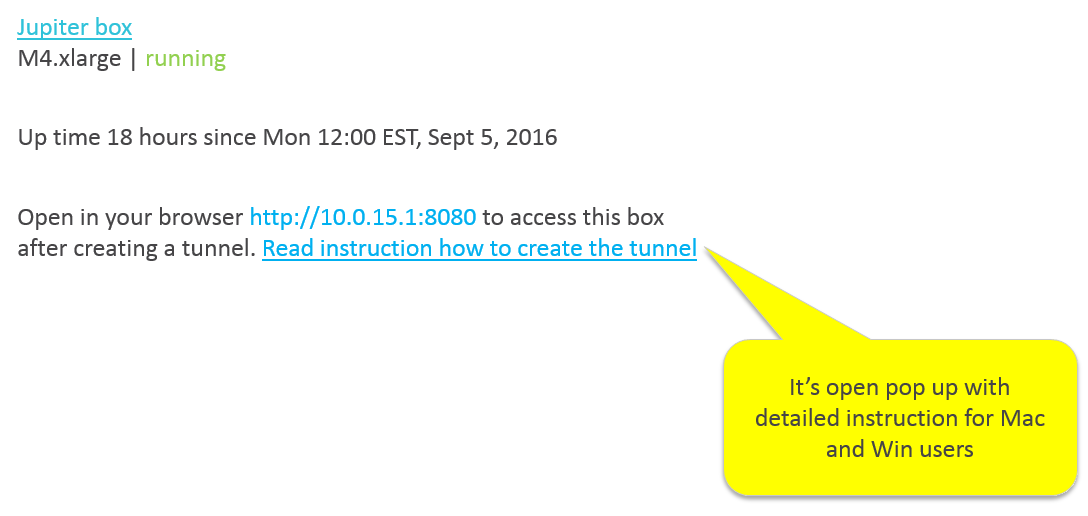


Figure Details

### Basic User Scenarios and Workflows

1. Data Scientist logs into application

As a Data Scientist I would like to login to application via WEB UI Login screen using my corporate LDAP credentials.

If I enter wrong username and/or password and hit Login - then validation message shows up between password field and Login button: “Wrong username and/or password”

If I enter proper credentials and hit Login – then I should be logged into application and automatically redirected to “Upload public key button” or “List of resources” screen.

If I haven’t upload my public key before – then I should be redirected to “Upload my public key screen”.

If I have already uploaded my public key successfully – then after Login I should be redirected to “List of resources screen”

1. Data Scientist uploads his public key

If I haven’t uploaded my public key before – then after Login I need to be redirected to “Upload public key” screen

If I have been redirected to “Upload public key screen” I should be able to perform following actions from it: Logout, upload a key, get acquainted with instructions, Create initial infrastructure.

If I click on Upload button – then a standard OS popup shows up asking to select a public key.

If I upload a corrupted public key, or not a key at all – validation error should show up and key should not be uploaded. Create button should remain disabled.

If I upload a proper public key – then name of the key should be displayed and Create button should become enabled.

1. Data Scientist follows the instructions for creation of public key for Win, Linux and Mac OS’s

If I don’t have public key available – then I should be able to create a pair of private + public keys following the instruction described in “Where can I get public key?” tab.

Edge Case: Data Scientist has multiple combinations of private and public keys and doesn’t know which private key related to which public key. Instructions should provide a clear guidance of how to determine which private key belongs to which public key.

1. Data Scientist creates initial infrastructure

If I successfully uploaded my public key and its name is showing up on the left of Upload button – then Create button at the bottom of “Upload public key” screen is activated.

If I click on Create button – then corresponding scripts for provisioning of initial infrastructure are called on the back-end and loading icon appears with the message “Provisioning of initial infrastructure is in progress. Please wait…”

If provisioning of initial infrastructure is completed – then corresponding instance of Edge node, subnet have been created.

If any errors occurred during provisioning of initial infrastructure – exception should be caught/logged and shown to Data Scientist in same in meaningful way (brief stack trace + expand button to see additional details).

If provisioning of initial infrastructure is completed – then I am redirected to “List resources page”

1. Data Scientists creates new analytical tool

If redirected to “List of resources page” – then only Logout and Create button should be available for Data Scientist in case no analytical tools have been created before, or if analytical tools created before have been terminated.

If redirected to “List of resources page” – then following actions should be available for Data Scientist if at least one analytical tool has been created and is not terminated: Run/Stop (depending on current status), Terminate and Deploy EMR, Template description (text), Create new and Logout button.

If I hit on “Create new” button – then a modal dialog (Figure 4 Create new analytical tool) box shows on top of “List Resources page”. Then I am able to select a template for analytical tool, Name and Instance Shape.

If I don’t have enough permissions – then I can only select a shape which I am allowed to.

If new analytical tools are being added – then they should automatically show up in the dropdown “Select template”

If a filled in name violates any of validation rules (TBD) – then a red border should appear over Name field and a tooltip with validation rule should appear near Name field.

If any of validation rules are violated – then Create button is disabled on “Create new analytical tool popup”.

If I have properly chosen all configuration for analytical tool – then Create button becomes activated and hitting on it should take me to “List resources page” with a record for new analytical tool on it and status “creating”.

If I hit on Refresh button and analytical tool is already up-and-running – then status should be changed to “running”

1. Data Scientist stops Notebook server

If there is an analytical tool already created and running – then Stop button label substitutes Run button. Terminate button and Deploy EMR are active.

If I hit on Stop button – then confirmation message shows up: “Notebook server [Name\_of \_analytical\_tool] will be stopped and all connected EMR instances will be terminated. Do you want to proceed?” with Yes and No action buttons.

If I select Yes (meaning Stop) – then status should be changed to “stopping”, “Deploy EMR” button should become disabled. As soon as Notebook server is stopped – status is changed to “stopped”, “Run”, “Terminate” buttons are active, “Deploy EMR” button is disabled.

1. Data Scientist terminates Notebook server

If there is an analytical tool already created and is in “stopped” or “running” state – then “Terminate” button should be active.

If I hit on “Terminate” button – then confirmation message shows up: “Notebook server [Name\_of \_analytical\_tool] and all connected EMR instances [names\_of\_all\_EMR instances] will be terminated. Do you want to proceed?” with Yes and No action buttons.

If I select “Yes” (meaning Terminate) – then status should be changed to “terminating”, “Deploy EMR” button should become disabled. As soon as Notebook server is terminated – corresponding analytical tool record gets deleted from “List resources” page, corresponding Analytical tool instance/process gets removed, all related EMR instances are removed as well.

1. Data Scientist creates computational resources

If

1. Data Scientist wants to see details of the created resources
2. Data Scientist wants to connect to his analytical tool Web UI be the means of provided instructions

## Non-functional requirements

### Self-service UI

There are two options: build (1) simple Web UI under MVC architecture where web application returns server-side generate web page or build (2) SPA backed by a set of RESTful web service.

The benefits of (2) is an implemented REST API which can be used as integration point with 3rd party systems. At the same time, there is the risk that such kind of application is significantly more difficult to implement and more expensive to support.

Both approaches are expected to be validate and the best one picked after checking available resources and cost/benefit estimation.

### Private exploratory environment

Each exploratory environment is created per user and inaccessible for other users inside cloud.

### Limited resources allocation

It must be a possibility to limit user in a number of resources available for him. The following resources must be considered:

* Shape for analytical toolbox
* Number of slaves in EMR cluster
* Shape for master and slave nodes in EMR cluster

### Configurable template

Templates must be reachable for customer’s devops. They must be able to make required changes in existing templates or adding new templates to the system on their own.

### Configurable identity provider

LDAP was selected as primary integration point to be implemented. The crutial point is to enable integration with corporate LDAP without a need to recompile solution. The example of potential issue with LDAP integration: customer’s LDAP has attributes “first name” and “last name” when user’s domain model has only “full name” attribute; nevertheless “first name” and “last name” must be mapped to “full name”. Mentioned mapping must be done without project recompilation without development team involvement, but only by means of customer.

### Cloud agnostic

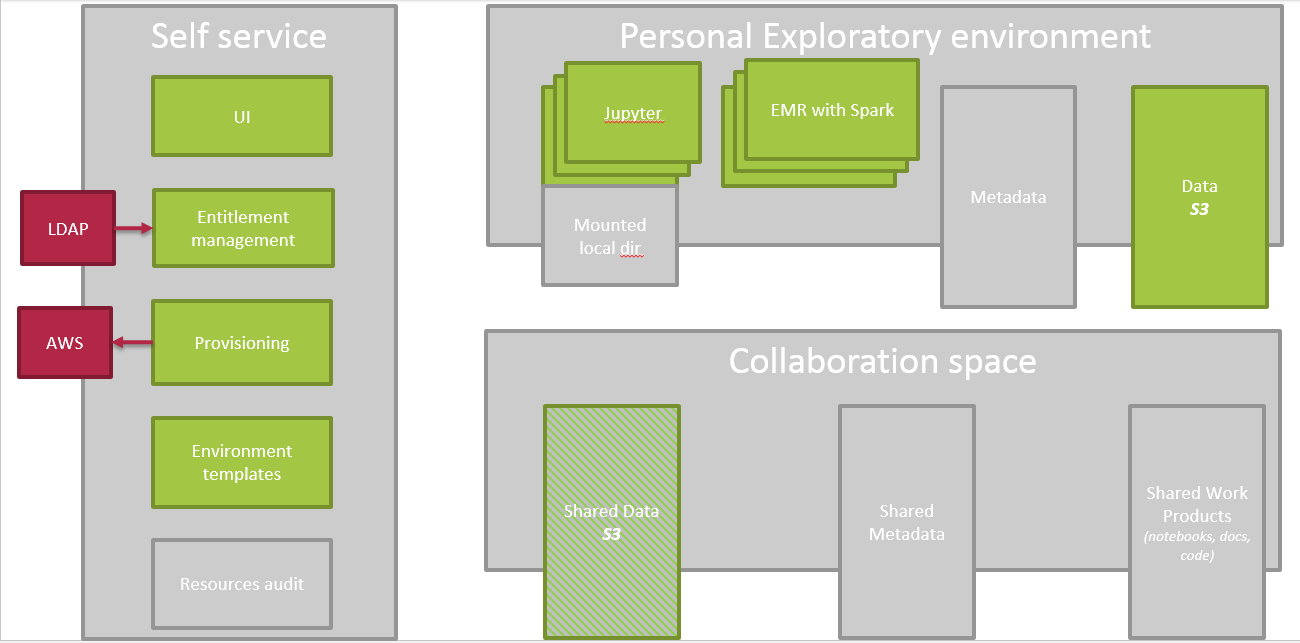
Solution shall be modular and flexible enough to be extended for support of new infrastructure provider without a need to significantly refactor core components of the system.

# Quality Attributes

|  |  |
| --- | --- |
| Attribute | Description |
| Maintainability | The system shall be able to undergo changes with minimal effort. |
| Interoperability | Despite the fact that the system is created to support AWS cloud at first place, these interfaces must be designed to be abstract to be replaced with other cloud providers in future. |
| Security | Security is the one of crucial drivers of the system. It must protect assets and prevent unauthorized access to them. |
| Usability | The system is self-explained for end users and must require minimal training or amount of documentation. |

# Baseline Solution architecture

## High-level solution structure



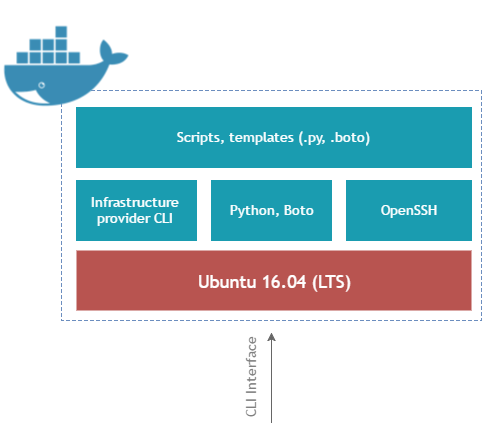
Solution was split in 3 logical areas which of them has own responsibility area and contains several logically independent components.

Self-service provides a set of components to enhance data scientist experience and decrease involvement of system administrators into data scientist lifecycle. It contains components like friendly UI, entitlement management integrated with corporate LDAP interface, provisioning on AWS and configurable environment templates.

Personal exploratory environment is created per data scientist and contains data storage (S3), a number of analytical tools such as Jupyter notebooks and EMR clusters used to provide scalable computational resources.

Collaboration space is currently out of scope and only shared data can be potentially used by data scientists.

### INITIAL SETUP PACKAGE - OLEH



|  |  |
| --- | --- |
| **Description** | Initial setup package is prepared docker container which is stored in EPAM docker registry. |
| **Technology Stack** | Initial setup package shall be based on Ubuntu 16.04 image with following additional components installed:   * Infrastructure provider CLI – in order to reduce additional dependency installation and simplify configuration and communication with Infrastructure Resource Manager (AWS SLI or Azure SLI for example). * Python, Boto shall be installed in order to run .py, .boto scripts. * Scripts and Templates for infrastructure setup. |
| **Related components** | [List related components with a short description of the relation nature] |
| **Covered functional requirements** | The main functionality of initial setup package is:  Perform initial infrastructure configuration.  Simplify deployment and setup scenarios. Initial setup package shall configure following components in the cloud: **- VOLODYMYR**   * Cloud resource group. (For AWS Tagging shall be used to mark all components as resource group) * Cloud vnet. * Create management node subnet. * Create and configure Self-Service node. (For AWS subnet for self-service node, ec2 instance for self-service node * Create and configure Collaboration space. (For AWS S3 bucket with pre-configured security policies)   Security group and profiles (For AWS iam profile) |
| **Notes** | [Put any additional specific notes here] |

### SELF-SERVICE NODE – KOSTIA

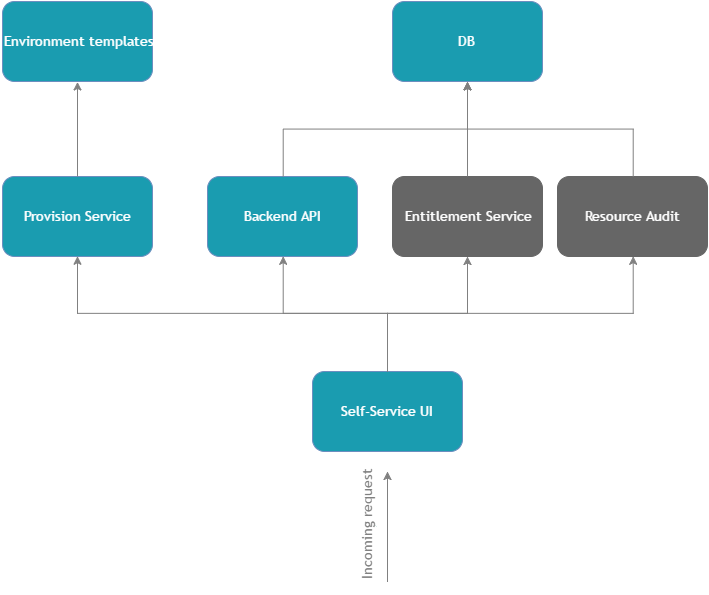


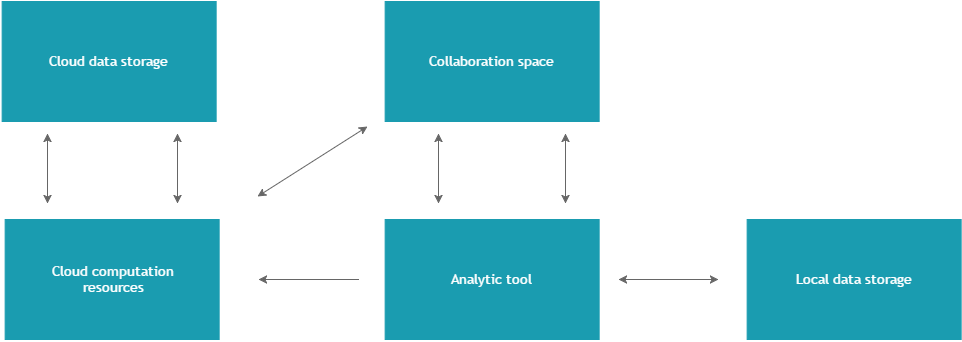
Figure 7All components presented on self-service node

NOTE: Components marked as gray are out-of-scope of MVP.

Self-service component (UI and backend) and related MongoDB instance are run on bare OS, not in Docker, in order to simplify a work with templates (which a packed into Docker containers) and further replacement or customization of templates.

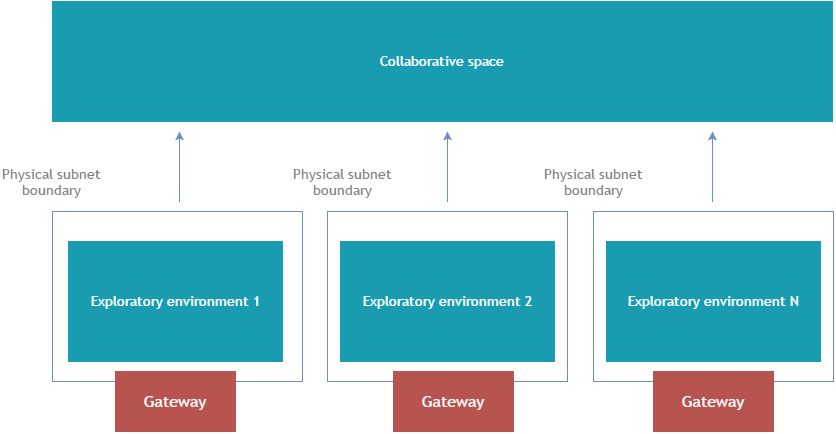
|  |  |
| --- | --- |
| **Description** | Self-service node major components:   * Self-service node web UI. * Self-service node DB. * Self-service node provisioning service. * Self-service node entitlement manager. * Environment templates. * Resource audit. |
| **Technology Stack** |  |
| **Related components** | [List related components with a short description of the relation nature] |
| **Covered functional requirements** | Self-service node shall serve as management component responsible for:   * Exploratory environment configuration and management (CRUD). * Collaboration space configuration. * User mapping to resource policy. |
| **Notes** | [Put any additional specific notes here] |

### EXPLORATORY ENVIRONMENT

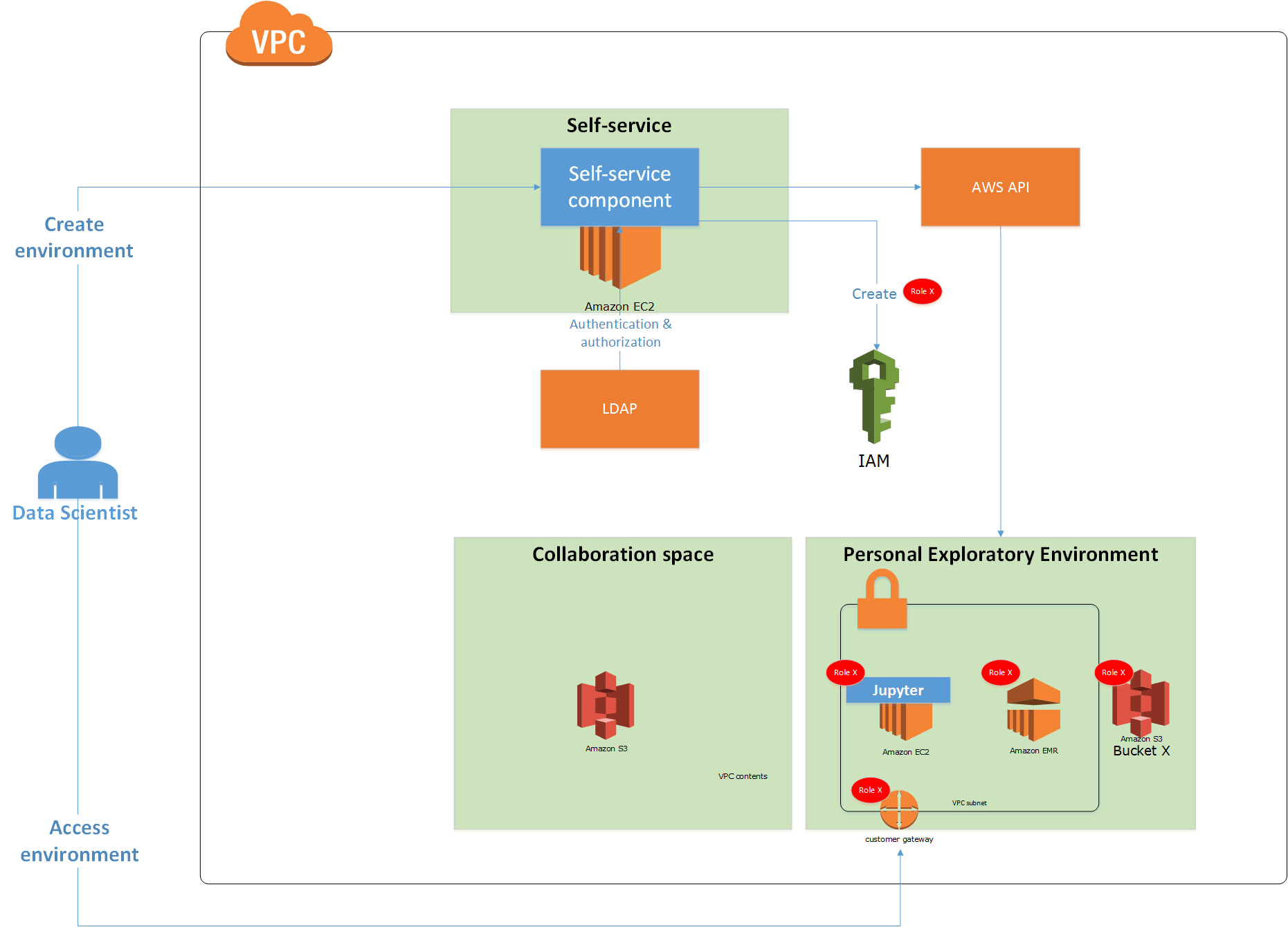


|  |  |
| --- | --- |
| **Description** | Exploratory environment consists of following components: |
| **Technology Stack** | * Analytic tool - according to template * Cloud computation resource – cloud-based Spark cluster provided by Amazon (EMR) * Data storage – per each exploratory environment separate storage resources shall be created. |
| **Related components** | [List related components with a short description of the relation nature] |
| **Covered functional requirements** |  |
|  |  |
| **Notes** | [Put any additional specific notes here] |

Each exploratory environment shall be isolated and put in separate subnet which will not be accessible from other exploratory environments, however access to collaborative space shall be available.



## SOLUTION IN AWS CLOUD



SOLUTION IN MS AZURE CLOUD

## Solution components structure

### SELF-SERVICE UI COMPONENT

|  |  |  |
| --- | --- | --- |
|  | Fluid design SPA /w REST | Server-generated web |
| Flexibility to modify | Full control over markup and styling. | Limited control, harder to implement and attach nice views. |
| Scalability / Portability | Due to service design much easier to scale and re-use functionality. | Tightly coupled design. |
| Maintainability | Separated UI, granular backend make system easier to test, support and troubleshoot. | More efforts required for system support and testing. |
| Future | Industry standard. | This approach is no longer treated as industry standard. |
| Costs | About 2x-3x higher cost for UI implementation than server generated web approach. |  |

### PROVISION SERVICE - VOLODYMYR

Service responsible for infrastructure management:

* Exploratory environment CRUD.
* Collaborative space management.
* Manage infrastructure templates.

Provision service provides API for these operations.

Provision service shall be implemented in technology selected in Self-Service UI and run Boto/Python scripts as well environment templates.

There are three type of containers:

Initial environment - edge node, bucket (data bucket + management bucket), role with policy)

Analytical tool – one machine with applied existing subnet and role/policy

Computational template (EMR) – EMR, reuses existing subnet, roles/policy

The following table illustrates API for each type of template where (+) means that method is implemented and (-) means method is not implemented:

|  |  |  |  |
| --- | --- | --- | --- |
| API method | Initial environment | Analytical tool | Computational template |
| Create | + | + | + |
| Get Status | + | + | + |
| Get Metadata | + | + | + |
| Stop | - | + | - |
| Run | - | + | - |
| Terminate | + | + | + |

Each template is called through command line interface from self-service component and interface between docker and component is defined via JSON formatted input/output.

VOLODYMYR: please provide JSON structure here

### BACKEND

Backend for self-service UI, shall perform various operations as well as communication with database.

### Authentication and Authorization approach

For MVP scope only support of LDAP is planned to be implemented. The goal is to achieve flexible and pluggable solution which allows to replace different identity provider with a minimal effort. LDAP is the primary goal for MVP and the following solution is proposed to solve compatibility between customer LDAP schema and solution entities:

Enterprise

AD

with LDAP interface

Request

LDAP integration

Mapping component

Self-service backend

Self-service UI

Figure LDAP integration

The idea of mapping component is to provide flexible approach to map attributes from customer’s LDAP to solution own entities. For example, each customer can have different attribute which specify user’s group. The mapping component is implement as Python file (in fact, Jython) which is executed in JVM and perform a mapping of LDAP attributes to general model. Mentioned Python file can be easy edited and replaced without a need to restart server.

The example of mapping is depictured on the diagram Figure 8 Mapping using Python

**Java LDAP connector**

*Java class*

(&(givenName=John)(l=Dali))

**LDAP**

email

given name

last name

**MAPPING**

*Python code*

email => email

given name + last name => full name

**USER**

*Java bean*

email

full name

Figure Mapping using Python

For database technology MongoDB is recommended as it supports large variety of frameworks and technologies as is document database, also it fits well for storing json configuraiton.

## Domain model

User has the following attributes:

* Unique id (it can be common Name from AD/LDAP)
* Full name
* User email

## Data model

[**Description:** This section describes the approach to data storage and data models. List data storage components of any nature, relational and non-relational, with data model diagrams for each storage.

**Section Type:** Highly recommended (if applicable)]

### Management node MongoDB

Management node DB shall contain following information:

* Exploratory environment information like who and what kind of resources created, released, etc
* Resources allocation policies like group A allowed to have no more than 10 nodes

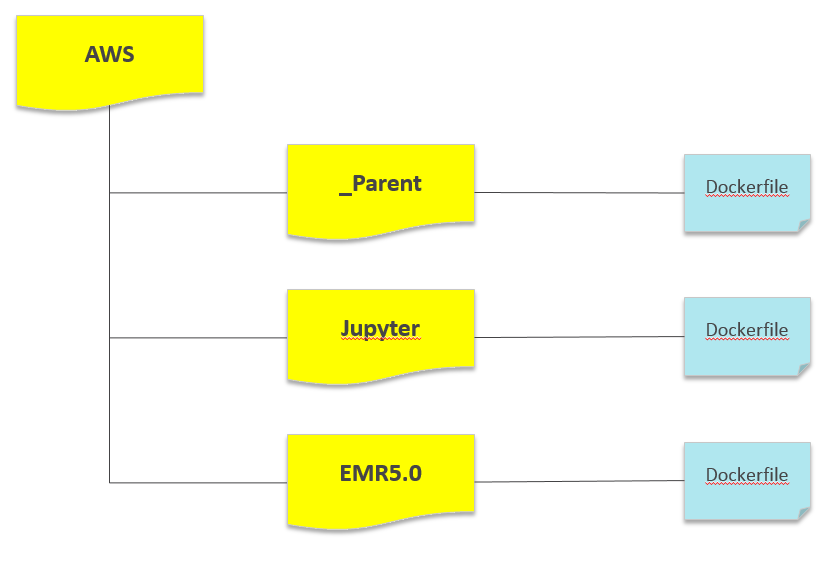
Each action about any interaction of data scientist with provisioning service must be recorded. The following format is proposed: user ID, user name, user email, timestamp, actionCode, details about action. Where “details about action” is a complex object which contains action specific information to be defined later. The list of supported actionCodes is the following: createNode, startNode, stopNode, terminateNode, createEMR, terminateEMR.

The following format shall be used to describe resource limitation for users: groupName, type, limitation. Where “limitation” is a complex object which contains type-specific limitations, the following four types are supported:

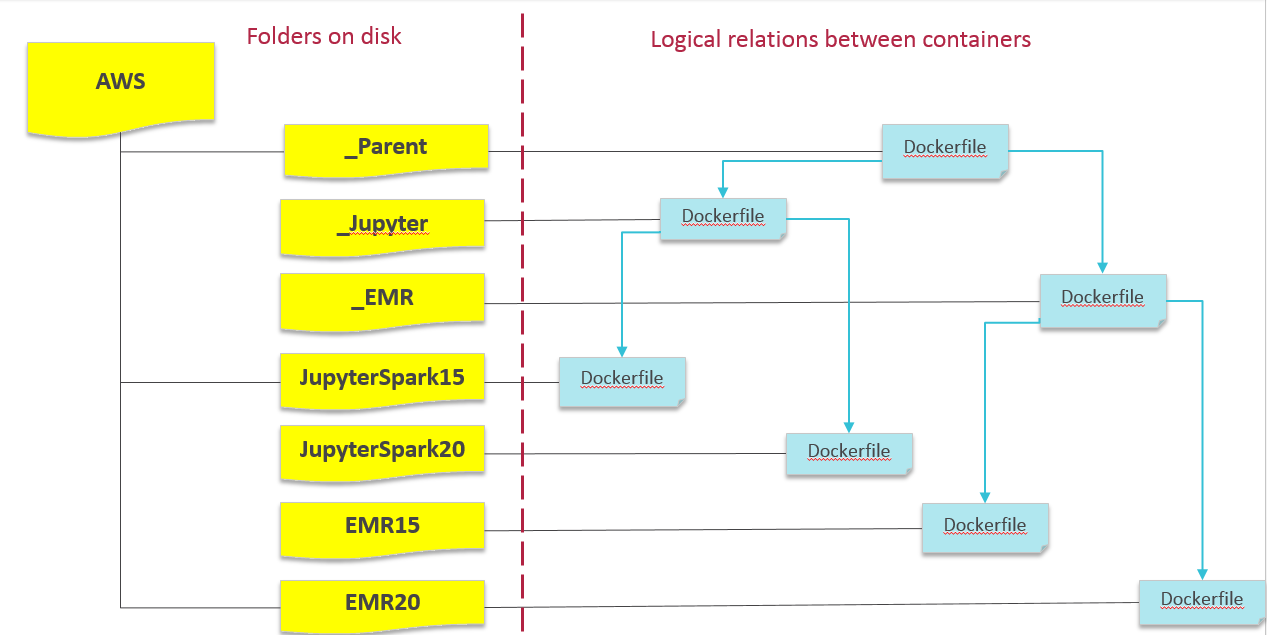
|  |  |
| --- | --- |
| Type | Info to record |
| Shape | Array of allowed shapes for assigned group |
| EmrSize | Integer, which is the maximum number of slaves in EMR cluster |
| EmrMasterShape | Array of allowed shapes for master node in EMR cluster |
| EmrSlaveShape | Array of allowed shapes for slave node in EMR cluster |

### Templates on Management node

Each template is stored in own Dockerfile and implements the same interface described in the section 5.1.5. These files are stored on filesystem in the following hierarchy, where root is pointed by configuration property. The example is show below, more templates are expected in real system:



Container name which starts with underscore must be ignored while calculating the list of available containers. This type of containers play utility role and don’t affect list of containers available to data scientist to spin off environment. Docker containers can form hierarchy tree when features of parent containers are inherited by children and the most obvious example is the following: parent AWS container is the parent container for all other containers. Consider the following example:



In this example, the following template are available to data scientists:

Jupyter notebook with Spark 1.5

Jupyter notebook with Spark 2.0

EMR with Spark 1.5

EMR with Spark 2.0

However, it’s easy to notice that both Jupyter templates share a lot of common components like Jupyter itself. It makes sense to create parent Dockerfile which contains Jupyter installation instructions and inherited by specific templates with different Spark integration. As parent Jupyter is described in folder that starts with underscore, it would be ignored.

All list of available templates is read during self-service boot and refreshed only after reboot operation. This process includes reading project structure and for each visible template (doesn’t start with underscore) it calls and get metadata (see section 5.1.5).

### 

## High-level deployment approach

### DEPLOYMENT SERVER - OLEH

|  |  |
| --- | --- |
| **Description** | In order to minimize efforts required for initial infrastructure setup and resolve issues related to docker and OS differences (Linux, Windows, MacOS) an pre-configured image shall be prepared.  Additionally, python/boto script shall be implemented that creates machine from scratch. |
| **Technology Stack** | Following software shall be installed in deployment server:   * Ubuntu 16.04 LTS. * OpenSSH. * Git, additionally access to EPAM Git repository shall be configured. * Docker, additionally access to EPAM docker registry shall be configured. |
| **Related components** | [List related components with a short description of the relation nature] |
| **Covered functional requirements** | [List covered functional requirements] |
| **Notes** | [Put any additional specific notes here] |

# Technology stack

|  |  |  |
| --- | --- | --- |
| Technology | Usage | Description |
| Ubuntu 16.04 x64 LTS | OS image and version used as base image / OS for Edge, Self-Service, Exploratory Environment Nodes, Docker base image | One of the leading open-source OS platforms. |
| Docker | Containerization and isolation platform used as container for Initial Setup Package | Open-source web-based platform for data-analytics. |
| MongoDB | Database for Self-Service node. |  |
| Jupyter | Used in exploratory environment as workplace for data scientist. | Open-source web-based platform for data-analytics. |
| Apache Spark | Data processing engine used in Exploratory environment. | Open-source Big Data batch in-memory data processing platform. |
| Python, Boto | Environment preparation scripts. |  |
| Python, SciPy | Data analytics, visualization, interface with Spark | Python ecosystem of open-source packages for mathematics, science and Statistics |
| R | Data analytics, visualization, interface with Spark | Open-source languageand environment for statistical computing and graphics |
| ReactJS, ImmutableObjectJS, Dispatcher | Front-end libraries for SPA development. | One of the leading Javascript frameworks for client-side UI implementation based on Flux architecture |
| jQuery, requirejs | Helper frameworks for Javascript. |  |
| Bootstrap | Helper framework for front-end development and styling. | One of the most popular frameworks for UI development. |
| Java 8, dropwizard.io, NodeJS, Sails.js | Backend frameworks for Self-Service UI |  |
| Jetty | HTTP server based on JVM |  |

## Architecture Constraints and Limitations

[**Description:** This section describes all the constraints and limitations that the proposed solution architecture has.

The Customer must be made aware of them, so you must communicate them explicitly and clearly.

Section Type: Optional]