

API Specification Doc

(VGrid)

Version	Date	Author	Description
2.0.0	28 February 2017	Solusi247	

Table of Contents

Methods	3
A. User Management.....	3
1. User.....	3
2. Group.....	9
3. Member	15
4. Role	20
B. Cluster	26
1. Cluster.....	26
2. Cluster Config.....	30
C. Project	37
1. Project.....	37
2. Inventory.....	46
D. Generate Jar	62
1. Generate Jar.....	62
2. Download Jar	63
3. View Jar.....	64
E. Oozie.....	67
1. Submit Job	67
2. Get Job ID.....	68
3. Get Job Info.....	69
4. Get Job History	72
5. Generate XML.....	73

Methods

A. User Management

User Management is used to manage user, group, member, and role on VGrid.

1. User

1.1. Get User by ID

This method is used to get user information.

- Request

Method	URL
GET	{host}:{port}/{apikey}/user/{user_id}
	Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/user/1

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{user_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "user_id": 1, "user_firstname": "Danang", "user_lastname": "Kastowo", "user_email": "danang.kastowo@solusi247.com", "user_apikey": "90867b984d2a5038ee21a190996b900b", "user_ip_address": "192.168.1.80,192.168.1.72,192.168.1.75", "user_create_date": "2016-10-17 00:00:00", "user_update_date": "2017-02-16 10:47:48", "user_is_active": "true", "user_cluster_id": "null" }] }
2	{ "err_code": 2, "err_msg": "User ID is not found" }
2	{

	{ "err_code": 2, "err_msg": "User ID must be numeric" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
5	{ "err_code": 5, "err_msg": "Access denied to view users" }
5	{ "err_code": 4, "err_msg": "IP Address is not registered" }

1.2. Get All User

This method is used to get all user information.

- Request

Method	URL
GET	{host}:{port}/{apikey}/user Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/user

Type	Parameter	Value
HEAD	auth_key	String
	{host}	IP address
URL_PARAMS	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "user_id": 1, "user_firstname": "Danang", "user_lastname": "Kastowo", "user_email": "danang.kastowo@solusi247.com", "user_apikey": "90867b984d2a5038ee21a190996b900b", "user_ip_address": "192.168.1.80,192.168.1.72,192.168.1.75", "user_create_date": "2016-10-17 00:00:00", "user_update_date": "2017-02-16 10:47:48", "user_is_active": "true", "user_cluster_id": "null" }]

	<pre> }, { "user_id": 2, "user_firstname": "anang", "user_lastname": "andrianto", "user_email": "anangandrianto29@solusi247.com", "user_apikey": "608523311f1d8a90e0fc332f8297dabd", "user_ip_address": "192.168.1.75", "user_create_date": "2017-03-08 11:14:29", "user_update_date": "2017-03-20 13:35:11", "user_is_active": "true", "user_cluster_id": "null" }] } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
5	<pre> { "err_code": 5, "err_msg": "Access denied to view users" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>

1.3. Add User

This method is used to add new user.

- Request

Method	URL
POST	<p>host:port/{apikey}/user</p> <p>Example :</p> <p>192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/user</p> <pre> { "user_firstname": "rafida", "user_lastname": "anggraini", "user_email": "rafida.anggraini@solusi247.com", "user_password": "rafida123", "user_ip_address": "192.168.1.75" } </pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer

	{apikey}	String
--	----------	--------

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "user_id": 5, "user_firstname": "rafida", "user_lastname": "anggraini", "user_email": "rafida.anggraini@solusi247.com", "user_apikey": "485bebcc299a94cea3103c7acc0cd142", "user_ip_address": "192.168.1.75", "user_create_date": "2017-03-29 13:51:53", "user_update_date": "null", "user_is_active": "true", "user_cluster_id": "null" }] }
3	{ "err_code": 3, "err_msg": "Access denied" }
2	{ "err_code": 2, "err_msg": "Email is already exist" }
2	{ "err_code": 2, "status": "First name is required" }
3	{ "err_code": 3, "err_msg": "Email is invalid" }
2	{ "err_code": 2, "status": "Password is invalid" }
2	{ "err_code": 2, "status": "IP address is invalid" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

1.4. Update User

This method is used to update user information.

- Request

Method	URL
PUT	host:port/{apikey}/user/{user_id} Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/user/5 { "user_firstname": "rafida", "user_lastname": "anggraini", "user_email": "rafida.anggraini123@solusi247.com", "user_password": "rafida123", "user_ip_address": "192.168.1.75" }

Type	Parameter	Value
HEAD	auth_key	String
	{host}	IP address
	{port}	Integer
	{apikey}	String
	{user_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "user_id": 5, "user_firstname": "rafida", "user_lastname": "anggraini", "user_email": "rafida.anggraini123@solusi247.com", "user_apikey": "485bebcc299a94cea3103c7acc0cd142", "user_ip_address": "192.168.1.75", "user_create_date": "2017-03-29 13:51:53", "user_update_date": "null", "user_is_active": "true", "user_cluster_id": "null" }] }
3	{ "err_code": 3, "err_msg": "Access denied" }
2	{ "err_code": 2,

	"err_msg": "User ID is not found" }
2	{ "err_code": 2, "err_msg": "User ID must be numeric" }
2	{ "err_code": 2, "err_msg": "Email is already exist" }
2	{ "err_code": 2, "status": "First name is required" }
500	{ "err_code": 500, "err_msg": "Body cannot Empty" }
3	{ "err_code": 3, "err_msg": "Email is invalid" }
2	{ "err_code": 2, "status": "Password is invalid" }
2	{ "err_code": 2, "status": "IP address is invalid" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

1.5. Delete User

This method is used to delete user.

- Request

Method	URL
DELETE	{host}:{port}/{apikey}/user/{user_id} Exampel : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/user/5

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer

	{apikey}	String
	{user_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "status": "User has been deleted" }
2	{ "err_code": 2, "err_msg": "User ID must be numeric" }
2	{ "err_code": 2, "err_msg": "User ID is not found" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
3	{ "err_code": 3, "err_msg": "Access denied. Cannot delete this user" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

2. Group

2.1. Get Group by ID

This method is used to get group information.

- Request

Method	URL
GET	host:port/{apikey}/group/{group_id} example : 192.168.1.230:7001/485bebcc299a94cea3103c7acc0cd142/group/1

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{group_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": [{ "group_id": 1, "group_name": "david_group", "group_is_active": "true", "group_status": "false", "group_create_date": "2017-03-08 11:16:05", "group_update_date": "2017-03-20 14:26:32" }] }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Group ID is not found" }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Group ID must be numeric" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

2.2. Get All Group

This method is used to get all group information.

- Request

Method	URL
GET	<pre>{host}:{port}/{apikey}/group</pre> <p>Example :</p> <pre>192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/group</pre>

Type	Parameter	Value
HEAD	auth_key	String
	{host}	IP address
URL_PARAMS	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": [{ "group_id": 1, "group_name": "david_group", "group_is_active": "true", "group_status": "false", "group_create_date": "2017-03-08 11:16:05", "group_update_date": "2017-03-20 14:26:32" }] }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>

2.3. Get Group by User

This method is used to get group information.

- Request

Method	URL
GET	<pre>{host}:{port}/{apikey}/group/{group_id}/user/{user_id}</pre> <p>Example :</p> <pre>192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/group/1/user/2</pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{group_id}	Integer
	{user_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": [</pre>

	<pre>{ "group_id": 1, "group_name": "david_group", "group_is_active": "true", "group_status": "false", "group_create_date": "2017-03-08 11:16:05", "group_update_date": "2017-03-20 14:26:32" }</pre>
1	<pre>{ "err_code": 1, "status": "User ID is not found" }</pre>
2	<pre>{ "err_code": 2, "err_msg": "User ID must be numeric" }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Group ID is not found" }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Group ID must be numeric" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>

2.4. Add Group

This method is used to add new group.

- Request

Method	URL
POST	<pre>{host}:{port}/{apikey}/group</pre> <p>Example :</p> <pre>192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/group</pre> <pre>{ "group_name": "Labs247", "group_status": true }</pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "group_id": 4, "group_name": "Labs247", "group_is_active": "true", "group_status": "true", "group_create_date": "2017-03-30 05:00:19", "group_update_date": "null" }] }
2	{ "err_code": 2, "status": "Group name is required" }
2	{ "err_code": 2, "status": "Group status is required" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }

2.5. Update Group

This method is used to update group information.

- Request

Method	URL
PUT	{host}:{port}/{apikey}/group/{group_id} Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/group/1 {

	<pre>"group_name": "LabsSolusi247", "group_status": true }</pre>
--	--

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{group_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": [{ "group_id": 4, "group_name": "LabsSolusi247", "group_is_active": "true", "group_status": "true", "group_create_date": "2017-03-30 05:00:19", "group_update_date": "2017-03-30 05:07:59" }] }</pre>
3	<pre>{ "err_code": 3, "status": "Parameters cannot Empty" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>

2.6. Delete Group

This method is used to delete group.

- Request

Method	URL
DELETE	<pre>{host}:{port}/{apikey}/group/{group_id}</pre> <p>Exampel :</p> <pre>192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/group/4</pre>

Type	Parameter	Value
HEAD	auth_key	String
	{host}	IP address
	{port}	Integer
	{apikey}	String
	{group_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "status": "Group has been deleted" }
1	{ "err_code": 1, "status": "Group ID is not found" }
3	{ "err_code": 3, "err_msg": "Group ID must be numeric" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }

3. Member

3.1. Get Member

This method is used to get member information.

- Request

Method	URL
GET	{host}:{port}/{apikey}/group/{group_id}/member Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/group/1/member

Type	Parameter	Value
HEAD	auth_key	String
	{host}	IP address
	{port}	Integer

	{apikey}	String
	{group_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "member_id": 3, "group_name": "LabsSolusi247", "user_id": 1, "user_firstname": "Danang", "user_lastname": "Kastowo", "user_email": "danang.kastowo@solusi247.com", "member_status": "Active", "member_status_request": "Confirm", "member_role": "Administrator" }] }
3	{ "err_code": 3, "err_msg": "Group ID must be numeric" }
3	{ "err_code": 3, "err_msg": "Group ID is not found" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }

3.2. Add Member

This method is used to add new member to a group.

- Request

Method	URL
POST	{host}:{port}/{apikey}/group example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/member {

	<pre>"user_id": 5, "group_id": 4 }</pre>
--	--

Type	Parameter	Value
HEAD	auth_key	String
	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": { "member_id": 4, "group_name": "LabsSolusi247", "user_id": 5, "user_firstname": "rafida", "user_lastname": "anggraini", "user_email": "rafida.anggraini@solusi247.com", "member_status": "Active", "member_status_request": "confirm", "member_role": "Member" } }</pre>
500	<pre>{ "err_code": 500, "status": "User ID is not found" }</pre>
1	<pre>{ "err_code": 1, "status": "Group id is not found" }</pre>
500	<pre>{ "err_code": 500, "status": "User is already member in this group" }</pre>
3	<pre>{ "err_code": 3, "err_msg": "Access denied" }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Member is already exist" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>

	}
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

3.3. Update Member

This method is used to update member information.

- Request

Method	URL
PUT	host:port/{apikey}/member/{member_id} Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/member/ { "user_id": 5, "group_id":1 }

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{member_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": { "member_id": 4, "group_name": "VGrid", "user_id": 5, "user_firstname": "rafida", "user_lastname": "anggraini", "user_email": "rafida.anggraini@solusi247.com", "member_status": "Active", "member_status_request": "confirm", "member_role": "Member" } }
500	{ "err_code": 500, "status": "User ID is not found" }

1	{ "err_code": 1, "status": "Group id is not found" }
3	{ "err_code": 3, "err_msg": "Member ID must be numeric" }
3	{ "err_code": 3, "err_msg": "Member ID is not found" }
2	{ "err_code": 2, "err_msg": "Member is already exist" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

3.4. Delete Member

This method is used to delete member from a group.

- Request

Method	URL
DELETE	host:port/{apikey}/group/{group_id}/member/{member_id} Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/group/2/member/3

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{group_id}	Integer
	{member_id}	Integer

- Response

Status	Response
0	{ "err_code": 0,

	<pre>"status": "Member has been deleted" }</pre>
3	<pre>{ "err_code": 3, "err_msg": "Member ID must be numeric" }</pre>
3	<pre>{ "err_code": 3, "err_msg": "Member ID is not found" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

4. Role

4.1. Get Role by ID

This method is used to get role information.

- Request

Method	URL
GET	<pre>{host}:{port}/{apikey}/group/{group_id}/member</pre> Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/group/1/member

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{role_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": [{ "role_id": 1, "role_name": "Administrator", "role_create_date": "2017-02-16 10:46:00", </pre>

	<pre> "role_update_date": "null" }] } </pre>
2	<pre> { "err_code": 2, "err_msg": "Role id is not found" } </pre>
3	<pre> { "err_code": 3, "err_msg": "Role ID must be numeric" } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>

4.2. Get All Role

This method is used to get all role information.

- Request

Method	URL
GET	<pre> {host}:{port}/{apikey}/role </pre> <p>Example :</p> <pre> 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/role </pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	<pre> { "err_code": 0, "data": [{ "role_id": 1, "role_name": "Administrator", "role_create_date": "2017-02-16 10:46:00", "role_update_date": "null" }] } </pre>

	}
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

4.3. Add Role

This method is used to add new role.

- Request

Method	URL
POST	{host}:{port}/{apikey}/role Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/role { "role_name": "Member" }

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "role_id": 2, "role_name": "Member", "role_create_date": "2017-03-29 17:17:08", "role_update_date": "null" }] }
2	{ "err_code": 2, "status": "Role name is required" }
500	{

	<pre>"err_code": 500, "err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

4.4. Update Role

This method is used to update role information.

- Request

Method	URL
PUT	<p>host:port/{apikey}/role/{role_id}</p> <p>Example :</p> <p>192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/role/2</p> <pre>{ "role_name": "Customer" }</pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{role_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": [{ "role_id": 2, "role_name": "Customer", "role_create_date": "2017-03-29 17:17:08", "role_update_date": "2017-03-29 17:21:04" }] }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Role name is empty" }</pre>
2	<pre>{ "err_code": 2,</pre>

	<pre>"err_msg": "Role ID is not found" }</pre>
3	<pre>{ "err_code": 3, "err_msg": "Role ID must be numeric" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

4.5. Delete Role

This method is used to delete role.

- Request

Method	URL
DELETE	host:port/{apikey}/role/{role_id}
	Exampel : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/role/2

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{role_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "status": "Role has been deleted" }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Role ID is not found" }</pre>
3	<pre>{ "err_code": 3, "err_msg": "Role ID must be numeric" }</pre>
500	<pre>{ "err_code": 500,</pre>

	<pre>"err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

B. Cluster

Cluster is used to manage cluster and cluster configuration on VGrid.

1. Cluster

1.1. Get All Cluster

This method is used to get all cluster information.

- Request

Method	URL
GET	{host}:{port}/{apikey}/cluster Example : 192.168.1.230:7002/90867b984d2a5038ee21a190996b900b/cluster

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "cluster_id": 1, "cluster_name": "Yava02", "cluster_status": "default", "cluster_create_date": "2017-01-17 00:00:00", "cluster_update_date": "2017-02-16 09:30:42" }] }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

1.1. Get Cluster by ID

This method is used to get cluster information.

- Request

Method	URL
GET	{host}:{port}/{apikey}/cluster/{cluster_id}
	Example : 192.168.1.230:7002/90867b984d2a5038ee21a190996b900b/cluster/1

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{cluster_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "cluster_id": 1, "cluster_name": "Yava02", "cluster_status": "default", "cluster_create_date": "2017-01-17 00:00:00", "cluster_update_date": "2017-02-16 09:30:42" }] }
2	{ "err_code": 2, "err_msg": "Cluster ID must be numeric" }
2	{ "err_code": 2, "err_msg": "Cluster ID is not found" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

1.2. Add Cluster

This method is used to add new cluster.

- Request

Method	URL
POST	{host}:{port}/{apikey}/cluster Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/cluster { "cluster_name": "Yava01", "cluster_status": "default" }

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "cluster_id": 2, "cluster_name": "Yava01", "cluster_status": "default", "cluster_create_date": "2017-03-29 17:32:37", "cluster_update_date": "null" }] }
500	{ "err_code": 1, "err_msg": "Cluster name is required" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

1.3. Update Cluster

This method is used to update cluster information.

- Request

Method	URL
PUT	host:port/{apikey}/cluster/{cluster_id} Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/cluster/2 <pre>{ "cluster_name": "Yava011", "cluster_status": "default" }</pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{cluster_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": [{ "cluster_id": 2, "cluster_name": "Yava011", "cluster_status": "default", "cluster_create_date": "2017-03-29 17:32:37", "cluster_update_date": "2017-03-29 17:37:06" }] }</pre>
500	<pre>{ "err_code": 1, "err_msg": "Cluster name is required" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

1.4. Delete Cluster

This method is used to delete cluster.

- Request

Method	URL
DELETE	host:port/{apikey}/cluster/{cluster_id} Example : 192.168.1.230:7001/90867b984d2a5038ee21a190996b900b/cluster/2

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{cluster_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "status": "Cluster has been deleted" }
2	{ "err_code": 2, "err_msg": "Cluster ID must be numeric" }
2	{ "err_code": 2, "err_msg": "Cluster ID is not found" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

2. Cluster Config

2.1. Get All Cluster Config

This method is used to get all cluster configuration.

- Request

Method	URL
GET	{host}:{port}/{apikey}/cluster/{cluster_id}/config Example : 192.168.43.20:7002/90867b984d2a5038ee21a190996b900b/cluster/1/

	onfig
--	-------

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{cluster_id}	Integer

- Response

Status	Response
0	{ "errcode": 0, "data": [{ "cluster_id": 1, "cluster_name": "Yava02", "cluster_status": "default", "config_id": 1, "config_key": "usernameOozie", "config_value": "yava", "config_create_date": "2017-02-01 06:08:04", "config_update_date": "2017-02-17 17:17:56" }] }
2	{ "err_code": 2, "err_msg": "Cluster ID must be numeric" }
3	{ "err_code": 3, "err_msg": "Cluster ID must be numeric" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

2.2. Get Cluster Config

This method is used to get cluster configuration.

- Request

Method	URL
--------	-----

GET	{host}:{port}/{apikey}/cluster/{cluster_id}/config/{config_id} Example : 192.168.43.20:7002/90867b984d2a5038ee21a190996b900b/cluster/1/onfig/1
------------	--

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{cluster_id}	Integer
	{config_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "cluster_id": "1", "cluster_name": "Yava02", "cluster_status": "default", "config_id": 1, "config_key": "usernameOozie", "config_value": "yava", "config_create_date": "2017-02-01 06:08:04", "config_update_date": "2017-02-17 17:17:56" }] }
2	{ "err_code": 2, "err_msg": "Cluster ID must be numeric" }
2	{ "err_code": 2, "err_msg": "Cluster ID is not found" }
6	{ "err_code": 6, "err_msg": "Config ID must be numeric" }
4	{ "err_code": 4, "err_msg": "Configuration is not found in this cluster" }
500	{ "err_code": 500,

	"err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

2.3. Add Cluster Config

This method is used to add new cluster configuration.

- Request

Method	URL
POST	{host}:{port}/{apikey}/cluster/{cluster_id} Example : 192.168.43.20:7002/90867b984d2a5038ee21a190996b900b/cluster/2/ onfig { "config_key": "usernameHdfs", "config_value": "yava" }

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{cluster_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "cluster_id": "2", "cluster_name": "Yava01", "cluster_status": "default", "config_id": 14, "config_key": "usernameHdfs", "config_value": "yava", "config_create_date": "2017-03-29 08:30:05", "config_update_date": "null" }] }
1	{

	{ "err_code": 1, "err_msg": "Configuration key is required" }
1	{ "err_code": 1, "err_msg": "Configuration value is required" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

2.4. Update Cluster Config

This method is used to update cluster configuration.

- Request

Method	URL
PUT	host:port/{apikey}/cluster/{cluster_id}/config/{config_id} Example : 192.168.43.20:7002/90867b984d2a5038ee21a190996b900b/cluster/2/ onfig/14 { "config_key": "usernameHdfs1", "config_value": "hadoop" }

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{cluster_id}	Integer
	{config_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "cluster_id": "2", "cluster_name": "Yava01", }

	<pre> "cluster_status": "default", "config_id": 14, "config_key": "usernameHdfs1", "config_value": "hadoop", "config_create_date": "2017-03-29 08:32:35", "config_update_date": "null" }] } </pre>
2	<pre> { "err_code": 2, "err_msg": "Cluster ID must be numeric" } </pre>
2	<pre> { "err_code": 2, "err_msg": "Cluster ID is not found" } </pre>
6	<pre> { "err_code": 6, "err_msg": "Config ID must be numeric" } </pre>
4	<pre> { "err_code": 4, "err_msg": "Configuration is not found in this cluster" } </pre>
1	<pre> { "err_code": 1, "err_msg": "Configuration key is required" } </pre>
1	<pre> { "err_code": 1, "err_msg": "Configuration value is required" } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>

2.5. Delete Cluster Config

This method is used to delete cluster configuration.

- Request

Method	URL
DELETE	host:port/{apikey}/cluster/{cluster_id}/config/{config_id}
	Example :

	192.168.43.20:7002/90867b984d2a5038ee21a190996b900b/cluster/2/onfig/14
--	--

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{cluster_id}	Integer
	{config_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "status": "Configuration has been deleted" }
2	{ "err_code": 2, "err_msg": "Cluster ID must be numeric" }
2	{ "err_code": 2, "err_msg": "Cluster ID is not found" }
6	{ "err_code": 6, "err_msg": "Config ID must be numeric" }
4	{ "err_code": 4, "err_msg": "Configuration is not found in this cluster" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

C. Project

Project is used to manage project and inventory on VGrid.

1. Project

1.1. Get Project by User

This method is used to get project information based on Project ID and User ID.

- Request

Method	URL
GET	{host}:{port}/{apikey}/project/{project_id}/user/{user_id} Example : 192.168.1.230:7003/90867b984d2a5038ee21a190996b900b/project/4, user/2

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer
	{user_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "project_id": "4", "project_name": "project_SCL", "project_create_date": "2017-03-20 19:16:22", "project_update_date": "2017-03-30 11:13:21", "project_is_share": "true", "user_id": "2" }] }
5	{ "err_code": 5, "err_msg": "Project ID must be numeric" }
2	{ "err_code": 2, "err_msg": "Project is not found" }
5	{ "err_code": 5, "err_msg": "Project is not found" }

	"err_msg": "User ID must be numeric"
1	{ "err_code": 1, "err_msg": "Access denied. Cannot access this project" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

1.2. Get All Project by User

This method is used to get all project information based on User ID.

- Request

Method	URL
GET	{host}:{port}/{apikey}/project/user/{user_id}
	Example : 192.168.1.230:7003/90867b984d2a5038ee21a190996b900b/project/user/2

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{user_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "project_id": "4", "project_name": "project_SCL", "project_create_date": "2017-03-20 19:16:22", "project_update_date": "2017-03-30 11:13:21", "project_is_share": "true", "user_id": "2" }] }

2	{ "err_code": 2, "err_msg": "No project for this user" }
5	{ "err_code": 5, "err_msg": "User ID must be numeric" }
1	{ "err_code": 1, "err_msg": "Access denied. Cannot access this project" }
2	{ "err_code": 2, "err_msg": "No project for this user" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

1.3. Get Project by Group

This method is used to get project information based on Project ID and Group ID.

- Request

Method	URL
GET	{host}:{port}/{apikey}/project/{project_id}/group/{group_id} Example : 192.168.1.230:7003/90867b984d2a5038ee21a190996b900b/project/4, group/1

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer
	{group_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": [{ "project_id": "4", "project_name": "project_SCL", "project_create_date": "2017-03-20 19:16:22", "project_update_date": "2017-03-30 11:13:21", "project_is_share": "true", "group_id": "1" }] }</pre>
5	<pre>{ "err_code": 5, "err_msg": "Project ID must be numeric" }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Project is not found" }</pre>
5	<pre>{ "err_code": 5, "err_msg": "Group ID must be numeric" }</pre>
1	<pre>{ "err_code": 1, "err_msg": "Access denied. Cannot access this project" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

1.4. Get All Project by Group

This method is used to get all project information based on Group ID.

- Request

Method	URL
GET	{host}:{port}/{apikey}/project/group/{group_id} Example : 192.168.1.230:7003/90867b984d2a5038ee21a190996b900b/project/group/1

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{group_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "project_id": "4", "project_name": "project_SCL", "project_create_date": "2017-03-20 19:16:22", "project_update_date": "2017-03-30 11:13:21", "project_is_share": "true", "group_id": "1" }] }
2	{ "err_code": 2, "err_msg": "Project is not found" }
5	{ "err_code": 5, "err_msg": "Group ID must be numeric" }
1	{ "err_code": 1, "err_msg": "Access denied. Cannot access this project" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }

4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>
----------	---

1.5. Add Project

This method is used to add new project.

- Request

Method	URL
GET	<pre>{host}:{port}/{apikey}/project/user/{user_id}</pre> <p>Example :</p> <pre>192.168.1.230:7003/bc4f313607048c0060fcd74ec77edbd0/project/user/2</pre> <pre>{ "project_name": "orm" }</pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{user_id}	Integer

- Response

Status	Response
0	<pre>{ "err_code": 0, "data": { "project_id": 11, "project_name": "orm", "project_create_date": "2017-03-30 12:35:42", "project_update_date": "null", "project_is_share": "false", "user_id": 2, "group_id": "null" } }</pre>
2	<pre>{ "err_code": 2, "status": "Project already exist", "data": [{ "project_id": 11, "project_name": "orm" }] }</pre>

	<pre>] } </pre>
2	<pre> { "err_code": 2, "err_msg": "Project name is required" } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>

1.6. Update Project by User

This method is used to update project information.

- Request

Method	URL
PUT	<pre> {host}:{port}/{apikey}/project/{project_id}/user/{user_id} Example : 192.168.1.230:7003/bc4f313607048c0060fcd74ec77edbd0/project/11/ser/2 { "project_name": "NewORM" } </pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer
	{user_id}	Integer

- Response

Status	Response
0	<pre> { "err_code": 0, "data": { "project_id": 11, "project_name": "NewORM", "project_create_date": "2017-03-30 12:35:42", "project_update_date": "2017-03-30 12:40:38", "project_is_share": 0, </pre>

	<pre> "user_id": 2, "group_id": "null" } </pre>
2	<pre> { "err_code": 2, "err_msg": "Project id is not found" } </pre>
1	<pre> { "err_code": 1, "err_msg": "Access denied. Cannot access this project" } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>

1.7. Update Project Share to Group

This method is used to update project sharing information.

- Request

Method	URL
PUT	<pre> {host}:{port}/{apikey}/project/{project_id}/share/{group_id} Example : 192.168.1.230:7003/bc4f313607048c0060fcd74ec77edbd0/project/11/ hare/2 { "project_is_share": "true" } </pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer
	{group_id}	Integer

- Response

Status	Response
0	<pre> { "err_code": 0, "status": " Project has been shared to group" } </pre>

	}
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }
2	{ "err_code": 2, "err_msg": "Access denied for this project" }

1.8. Delete Project

This method is used to delete project.

- Request

Method	URL
DELETE	{host}:{port}/{apikey}/project/{project_id}/user/{user_id} Example : 192.168.1.230:7003/90867b984d2a5038ee21a190996b900b/project/1/user/2

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer
	{group_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "status": " Project has been shared to group" }
2	{ "err_code": 2, "err_msg": "Access denied. Cannot access this project" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{

	<pre> "err_code": 4, "err_msg": "IP Address is not registered } </pre>
--	--

2. Inventory

2.1. Get Inventory

This method is used to get inventory information.

- Request

Method	URL
GET	<pre> {host}:{port}/{apikey}/project/{project_id}/inventory/{inventory_id} Example : 192.168.1.230:7003/bc4f313607048c0060fcd74ec77edbd0/project/11/ nventory/23 </pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer
	{inventory_id}	Integer

- Response

Status	Response
0	<pre> { "err_code": 0, "data": [{ "project_id": "11", "project_name": "newORM", "inventory_id": 23, "inventory_name": "encaes256", "inventory_path": "/usr/share/vgrid/inventory/anang_andrianto/csd/", "inventory_type": "csd", "inventory_create_date": "2017-03-30 13:48:53", "inventory_update_date": "null", "inventory_package": { "HGrid247": { "HfsSource": { "ID": "1490068619071", "hfsName": "_user_yava_purchases_in", "label": "_user_yava_purchases_in", "type": "HFSSOURCE", "schemeClass": "TextLine", "outputFormat": "hgrid247-####", "enabled": "true", </pre>

```

"maxPrevNode": "1",
"posx": "115",
"posy": "200",
"inputdir": "/user/yava/purchases.in"
},
"Each": [
{
  "ID": "1490068619080",
  "name": "each_1490068619080",
  "type": "EACH",
  "pipeName": "each_1490068619080",
  "maxPrevNode": "1",
  "flowName": "1490068619080",
  "enabled": "true",
  "posx": "230",
  "posy": "200",
  "label": "inputparser_0",
  "inFields": "line",
  "inFieldsType": "String",
  "outFields": "in0.date,in0.cc,in0.total",
  "outFieldsType": "String,String,Double"
},
{
  "ID": "1490068619110",
  "name": "each_1490068619110",
  "type": "EACH",
  "pipeName": "each_1490068619110",
  "maxPrevNode": "1",
  "flowName": "1490068619110",
  "enabled": "true",
  "posx": "345",
  "posy": "200",
  "label": "mapping_1",
  "inFields": "in0.date,in0.cc,in0.total",
  "inFieldsType": "String,String,Double",
  "outFields": "out0.date,out0.Func0,out0.total",
  "outFieldsType": "String,String,Double"
},
{
  "ID": "1490068619126",
  "name": "each_1490068619126",
  "type": "EACH",
  "pipeName": "each_1490068619126",
  "maxPrevNode": "1",
  "flowName": "1490068619126",
  "enabled": "true",
  "posx": "460",
  "posy": "200",
  "label": "outputjoiner_2",
  "inFields": "out0.date,out0.Func0,out0.total",
  "inFieldsType": "String,String,Double",

```

```

        "outFields": "outputfield",
        "outFieldsType": "String"
    }
],
"Transform": [
    {
        "parentID": "1490068619080",
        "ID": "1490068619088",
        "label": "transform_0",
        "enabled": "true",
        "posx": "270",
        "posy": "130",
        "inputCount": "2",
        "outputCount": "2",
        "name": "cascading.solusi247.transform.string.Substring",
        "input": "line",
        "output": "in0.date",
        "parameter": "0,10"
    },
    {
        "parentID": "1490068619080",
        "ID": "1490068619096",
        "label": "transform_1",
        "enabled": "true",
        "posx": "270",
        "posy": "130",
        "inputCount": "2",
        "outputCount": "2",
        "name": "cascading.solusi247.transform.string.Substring",
        "input": "line",
        "output": "in0.cc",
        "parameter": "11,16"
    },
    {
        "parentID": "1490068619080",
        "ID": "1490068619103",
        "label": "transform_2",
        "enabled": "true",
        "posx": "270",
        "posy": "130",
        "inputCount": "2",
        "outputCount": "2",
        "name": "cascading.solusi247.transform.string.Substring",
        "input": "line",
        "output": "in0.total",
        "parameter": "29,7"
    },
    {
        "parentID": "1490068619110",
        "ID": "1490068619118",
        "label": "transform_3",

```



```
"enabled": "true",
"posx": "270",
"posy": "130",
"inputCount": "2",
"outputCount": "2",
"name": "EncryptAes256",
"input": "in0.cc",
"output": "out0.Func0",
"parameter": "myPassword123"
},
{
  "parentID": "1490068619126",
  "ID": "1490068619133",
  "label": "transform_4",
  "enabled": "true",
  "posx": "270",
  "posy": "130",
  "inputCount": "2",
  "outputCount": "2",
  "name": "FieldJoiner",
  "input": "out0.date,out0.Func0,out0.total",
  "output": "outputfield",
  "parameter": "|"
}
],
"Hfs2Pipe": {
  "fromID": "1490068619071",
  "toID": "1490068619080"
},
"Identity": [
  {
    "parentID": "1490068619110",
    "fromField": "in0.date",
    "toField": "out0.date"
  },
  {
    "parentID": "1490068619110",
    "fromField": "in0.total",
    "toField": "out0.total"
  }
],
"Pipe2Pipe": [
  {
    "fromID": "1490068619080",
    "toID": "1490068619110",
    "color": "black"
  },
  {
    "fromID": "1490068619110",
    "toID": "1490068619126",
    "color": "black"
  }
]
```

	<pre> }], "HfsSink": { "ID": "1490068619141", "hfsName": "_user_yava_purchases_encrypted", "label": "_user_yava_purchases_encrypted", "type": "HFSSINK", "enabled": "true", "posx": "575", "posy": "200", "outputFormat": "hgrid247-#####", "fileType": "Hfs", "outputdir": "/user/yava/purchases.encrypted", "incomingFields": "outputfield" }, "Pipe2Hfs": { "fromID": "1490068619126", "toID": "1490068619141" } } } } } }] } </pre>
5	<pre> { "err_code": 5, "err_msg": "Project ID must be numeric" } </pre>
3	<pre> { "err_code": 3, "err_msg": "Project Id not found" } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>
2	<pre> { "err_code": 2, "err_msg": "Project is not share for this user" } </pre>

2.2. Download Inventory

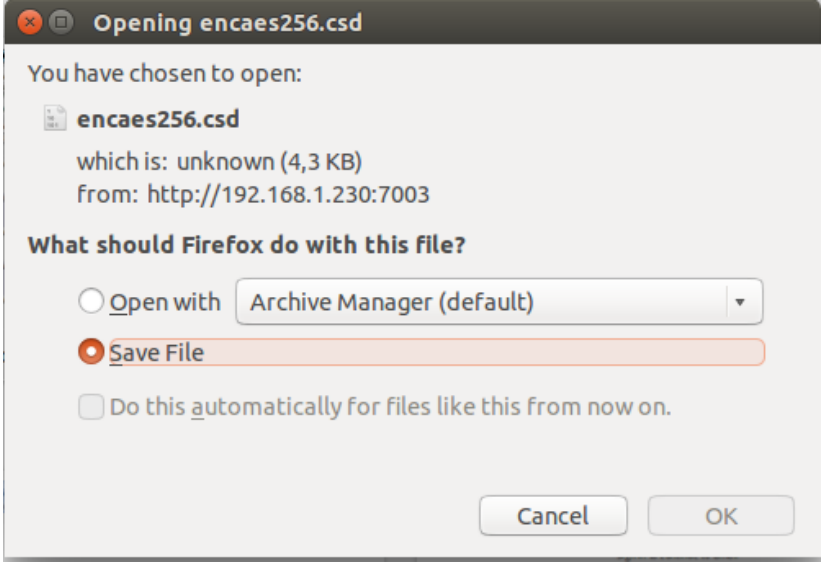
This method is used to download inventory.

- Request

Method	URL
GET	{host}:{port}/{apikey}/download_inventory/{inventory_id}/project/{project_id} Example : 192.168.1.230:7003/bc4f313607048c0060fcd74ec77edbd0/download_inventory/23/project/11

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{inventory_id}	Integer
	{project_id}	Integer

- Response

Status	Response
0	
2	{ "err_code": 2, "err_msg": "Inventory is not found" }
5	{ "err_code": 5, "err_msg": "Project ID must be numeric" }
3	{ "err_code": 3, "err_msg": "Project Id not found" }

	}
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }
2	{ "err_code": 2, "err_msg": "Project is not share for this user" }

2.3. Add Inventory

This method is used to add new inventory to a project.

- Request

Method	URL
POST	<p>{host}:{port}/{apikey}/project/{project_id}/inventory</p> <p>Example :</p> <p>192.168.1.230:7003/bc4f313607048c0060fcd74ec77edbd0/project/11/inventory</p> <pre>{ "inventory_name": "encaes256", "inventory_type": "csd", "inventory_package": { "HGrid247": { "HfsSource": { "ID": "1490068619071", "hfsName": "_user_yava_purchases_in", "label": "_user_yava_purchases_in", "type": "HFSSOURCE", "schemeClass": "TextLine", "outputFormat": "hgrid247-#####", "enabled": "true", "maxPrevNode": "1", "posx": "115", "posy": "200", "inputdir": "/user/yava/purchases.in" }, "Each": [{ "ID": "1490068619080", "name": "each_1490068619080", "type": "EACH", "pipeName": "each_1490068619080", "maxPrevNode": "1",</pre>

```

"flowName": "1490068619080",
"enabled": "true",
"posx": "230",
"posy": "200",
"label": "inputparser_0",
"inFields": "line",
"inFieldsType": "String",
"outFields": "in0.date,in0.cc,in0.total",
"outFieldsType": "String,String,Double"
},
{
  "ID": "1490068619110",
  "name": "each_1490068619110",
  "type": "EACH",
  "pipeName": "each_1490068619110",
  "maxPrevNode": "1",
  "flowName": "1490068619110",
  "enabled": "true",
  "posx": "345",
  "posy": "200",
  "label": "mapping_1",
  "inFields": "in0.date,in0.cc,in0.total",
  "inFieldsType": "String,String,Double",
  "outFields": "out0.date,out0.Func0,out0.total",
  "outFieldsType": "String,String,Double"
},
{
  "ID": "1490068619126",
  "name": "each_1490068619126",
  "type": "EACH",
  "pipeName": "each_1490068619126",
  "maxPrevNode": "1",
  "flowName": "1490068619126",
  "enabled": "true",
  "posx": "460",
  "posy": "200",
  "label": "outputjoiner_2",
  "inFields": "out0.date,out0.Func0,out0.total",
  "inFieldsType": "String,String,Double",
  "outFields": "outputfield",
  "outFieldsType": "String"
}
],
"Transform": [
  {
    "parentID": "1490068619080",
    "ID": "1490068619088",
    "label": "transform_0",
    "enabled": "true",
    "posx": "270",
    "posy": "130",

```

```

"inputCount": "2",
"outputCount": "2",
"name": "cascading.solusi247.transform.string.Substring",
"input": "line",
"output": "in0.date",
"parameter": "0,10"
},
{
  "parentID": "1490068619080",
  "ID": "1490068619096",
  "label": "transform_1",
  "enabled": "true",
  "posx": "270",
  "posy": "130",
  "inputCount": "2",
  "outputCount": "2",
  "name": "cascading.solusi247.transform.string.Substring",
  "input": "line",
  "output": "in0.cc",
  "parameter": "11,16"
},
{
  "parentID": "1490068619080",
  "ID": "1490068619103",
  "label": "transform_2",
  "enabled": "true",
  "posx": "270",
  "posy": "130",
  "inputCount": "2",
  "outputCount": "2",
  "name": "cascading.solusi247.transform.string.Substring",
  "input": "line",
  "output": "in0.total",
  "parameter": "29,7"
},
{
  "parentID": "1490068619110",
  "ID": "1490068619118",
  "label": "transform_3",
  "enabled": "true",
  "posx": "270",
  "posy": "130",
  "inputCount": "2",
  "outputCount": "2",
  "name": "EncryptAes256",
  "input": "in0.cc",
  "output": "out0.Func0",
  "parameter": "myPassword123"
},
{
  "parentID": "1490068619126",

```

```

    "ID": "1490068619133",
    "label": "transform_4",
    "enabled": "true",
    "posx": "270",
    "posy": "130",
    "inputCount": "2",
    "outputCount": "2",
    "name": "FieldJoiner",
    "input": "out0.date,out0.Func0,out0.total",
    "output": "outputfield",
    "parameter": "|"
  }
],
"Hfs2Pipe": {
  "fromID": "1490068619071",
  "toID": "1490068619080"
},
"Identity": [
  {
    "parentID": "1490068619110",
    "fromField": "in0.date",
    "toField": "out0.date"
  },
  {
    "parentID": "1490068619110",
    "fromField": "in0.total",
    "toField": "out0.total"
  }
],
"Pipe2Pipe": [
  {
    "fromID": "1490068619080",
    "toID": "1490068619110",
    "color": "black"
  },
  {
    "fromID": "1490068619110",
    "toID": "1490068619126",
    "color": "black"
  }
],
"HfsSink": {
  "ID": "1490068619141",
  "hfsName": "_user_yava_purchases_encrypted",
  "label": "_user_yava_purchases_encrypted",
  "type": "HFSSINK",
  "enabled": "true",
  "posx": "575",
  "posy": "200",
  "outputFormat": "hgrid247-#####",
  "fileType": "Hfs",

```

	<pre> "outputdir": "/user/yava/purchases.encrypted", "incomingFields": "outputfield" }, "Pipe2Hfs": { "fromID": "1490068619126", "toID": "1490068619141" } } } } </pre>
--	---

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer

- Response

Status	Response
0	<pre> { "err_code": 0, "data": [{ "project_id": "11", "project_name": "newORM", "inventory_id": 21, "inventory_name": "encaes256", "inventory_path": "/usr/share/vgrid/inventory/anang_andrianto/csd/", "inventory_type": "csd", "inventory_create_date": "2017-03-30 13:45:30", "inventory_update_date": "null", "inventory_package": { "HGrid247": { "HfsSource": { "ID": "1490068619071", "hfsName": "_user_yava_purchases_in", "label": "_user_yava_purchases_in", "type": "HFSSOURCE", "schemeClass": "TextLine", "outputFormat": "hgrid247-#####", "enabled": "true", "maxPrevNode": "1", "posx": "115", "posy": "200", "inputdir": "/user/yava/purchases.in" }, },], "Each": [</pre>


```

{
  "ID": "1490068619080",
  "name": "each_1490068619080",
  "type": "EACH",
  "pipeName": "each_1490068619080",
  "maxPrevNode": "1",
  "flowName": "1490068619080",
  "enabled": "true",
  "posx": "230",
  "posy": "200",
  "label": "inputparser_0",
  "inFields": "line",
  "inFieldsType": "String",
  "outFields": "in0.date,in0.cc,in0.total",
  "outFieldsType": "String,String,Double"
},
{
  "ID": "1490068619110",
  "name": "each_1490068619110",
  "type": "EACH",
  "pipeName": "each_1490068619110",
  "maxPrevNode": "1",
  "flowName": "1490068619110",
  "enabled": "true",
  "posx": "345",
  "posy": "200",
  "label": "mapping_1",
  "inFields": "in0.date,in0.cc,in0.total",
  "inFieldsType": "String,String,Double",
  "outFields": "out0.date,out0.Func0,out0.total",
  "outFieldsType": "String,String,Double"
},
{
  "ID": "1490068619126",
  "name": "each_1490068619126",
  "type": "EACH",
  "pipeName": "each_1490068619126",
  "maxPrevNode": "1",
  "flowName": "1490068619126",
  "enabled": "true",
  "posx": "460",
  "posy": "200",
  "label": "outputjoiner_2",
  "inFields": "out0.date,out0.Func0,out0.total",
  "inFieldsType": "String,String,Double",
  "outFields": "outputfield",
  "outFieldsType": "String"
}
},
"Transform": [
{

```

```

"parentID": "1490068619080",
"ID": "1490068619088",
"label": "transform_0",
"enabled": "true",
"posx": "270",
"posy": "130",
"inputCount": "2",
"outputCount": "2",
"name": "cascading.solusi247.transform.string.Substring",
"input": "line",
"output": "in0.date",
"parameter": "0,10"
},
{
"parentID": "1490068619080",
"ID": "1490068619096",
"label": "transform_1",
"enabled": "true",
"posx": "270",
"posy": "130",
"inputCount": "2",
"outputCount": "2",
"name": "cascading.solusi247.transform.string.Substring",
"input": "line",
"output": "in0.cc",
"parameter": "11,16"
},
{
"parentID": "1490068619080",
"ID": "1490068619103",
"label": "transform_2",
"enabled": "true",
"posx": "270",
"posy": "130",
"inputCount": "2",
"outputCount": "2",
"name": "cascading.solusi247.transform.string.Substring",
"input": "line",
"output": "in0.total",
"parameter": "29,7"
},
{
"parentID": "1490068619110",
"ID": "1490068619118",
"label": "transform_3",
"enabled": "true",
"posx": "270",
"posy": "130",
"inputCount": "2",
"outputCount": "2",
"name": "EncryptAes256",

```

```

    "input": "in0.cc",
    "output": "out0.Func0",
    "parameter": "myPassword123"
  },
  {
    "parentID": "1490068619126",
    "ID": "1490068619133",
    "label": "transform_4",
    "enabled": "true",
    "posx": "270",
    "posy": "130",
    "inputCount": "2",
    "outputCount": "2",
    "name": "FieldJoiner",
    "input": "out0.date,out0.Func0,out0.total",
    "output": "outputfield",
    "parameter": "|"
  }
],
"Hfs2Pipe": {
  "fromID": "1490068619071",
  "toID": "1490068619080"
},
"Identity": [
  {
    "parentID": "1490068619110",
    "fromField": "in0.date",
    "toField": "out0.date"
  },
  {
    "parentID": "1490068619110",
    "fromField": "in0.total",
    "toField": "out0.total"
  }
],
"Pipe2Pipe": [
  {
    "fromID": "1490068619080",
    "toID": "1490068619110",
    "color": "black"
  },
  {
    "fromID": "1490068619110",
    "toID": "1490068619126",
    "color": "black"
  }
],
"HfsSink": {
  "ID": "1490068619141",
  "hfsName": "_user_yava_purchases_encrypted",
  "label": "_user_yava_purchases_encrypted",

```

	<pre> "type": "HFSSINK", "enabled": "true", "posx": "575", "posy": "200", "outputFormat": "hgrid247-#####", "fileType": "Hfs", "outputdir": "/user/yava/purchases.encrypted", "incomingFields": "outputfield" }, "Pipe2Hfs": { "fromID": "1490068619126", "toID": "1490068619141" } } } }] } </pre>
2	<pre> { "err_code": 2, "err_msg": "Project is not share for this user" } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>

2.4. Delete Inventory

This method is used to delete inventory from a project.

- Request

Method	URL
DELETE	<pre> {host}:{port}/{apikey}/project/{project_id}/inventory/{inventory_id} </pre> <p>Example :</p> <pre> 192.168.1.230:7003/bc4f313607048c0060fcd74ec77edbd0/project/11/ nventory/23 </pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{inventory_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "status": "Inventory has been deleted" }
2	{ "err_code": 2, "err_msg": "Access denied. Cannot access this project" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

D. Generate Jar

Generate Jar is used to generate, download, and view a jar on VGrid.

1. Generate Jar

This method is used to generate jar.

- Request

Method	URL
GET	{host}:{port}/{apikey}/{apikey}/generate_jar/{project_id}/{inventory_id}
	Example : 192.168.1.230:7004/90867b984d2a5038ee21a190996b900b/generate_jar/7/2

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer
	{inventory_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": [{ "compile_id": 42, "compile_name": "newORM", "compile_path": "/user/yava/vgrid/jar/newORM/", "compile_type": "jar", "compile_create_date": "2017-03-30 14:24:58", "compile_update_date": "null", "compile_project_id": 11, "compile_inventory_id": 22 }] }
1	{ "err_code": 1, "err_msg": "Project id is not found" }
26	{ "err_code": 26, "err_msg": "Project or inventory is not found" }
28	{

	<pre>"err_code": 28, "err_msg": "Inventory id must be numeric" }</pre>
29	<pre>{ "err_code": 29, "err_msg": "Project id must be numeric" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

2. Download Jar

This method is used to download jar.

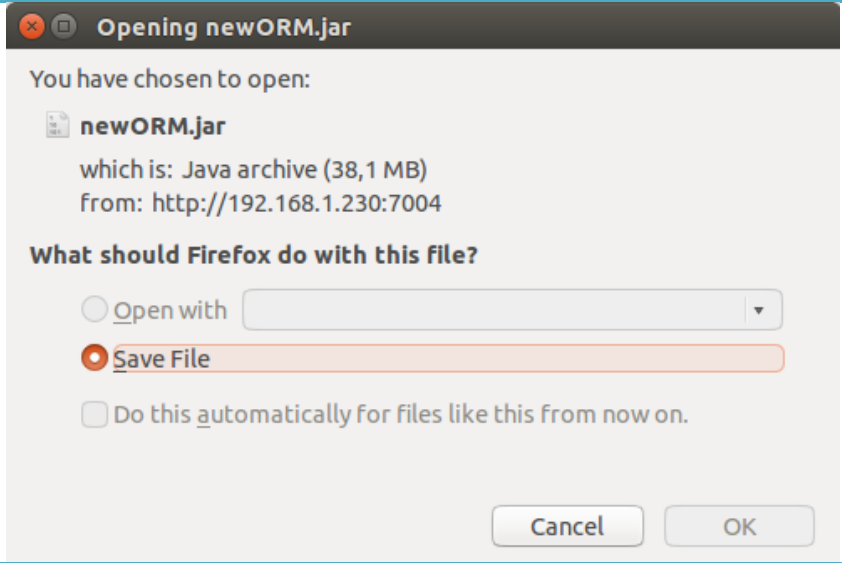
- Request

Method	URL
GET	<pre>{host}:{port}/{apikey}/download_inventory/{inventory_id}/project/{project_id}</pre> <p>Example :</p> <pre>192.168.1.230:7003/bc4f313607048c0060fcd74ec77edbd0/download_inventory/17/project/10</pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{inventory_id}	Integer

- Response

Status	Response
--------	----------

0	
1	<pre>{ "err_code": 1, "err_msg": "Compile id must be numeric" }</pre>
2	<pre>{ "err_code": 2, "err_msg": "Compile ID is not found" }</pre>
1	<pre>{ "err_code": 1, "err_msg": "Project id is not found" }</pre>
1	<pre>{ "err_code": 1, "err_msg": "Project ID must be numeric" }</pre>
500	<pre>{ "err_code": 500, "err_msg": "Wrong apikey" }</pre>
4	<pre>{ "err_code": 4, "err_msg": "IP Address is not registered" }</pre>

3. View Jar

This method is used to view jar.

- Request

Method	URL
GET	{host}:{port}/{apikey}/view_jar/{compile_id}/project/{project_id} Example : 192.168.1.230:7004/bc4f313607048c0060fcd74ec77edbd0/view_jar/42

	project/11
--	------------

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{compile_id}	Integer
	{project_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": { "project_id": 11, "project_name": "newORM", "compile_id": 42, "compile_name": "newORM", "compile_path": "/user/yava/vgrid/jar/newORM/", "compile_type": "jar", "compile_create_date": "2017-03-30 14:24:58", "compile_update_date": "null", "compile_inventory_id": 22 } }
1	{ "err_code": 1, "err_msg": "Compile id must be numeric" }
2	{ "err_code": 2, "err_msg": "Compile ID is not found" }
1	{ "err_code": 1, "err_msg": "Project id is not found" }
1	{ "err_code": 1, "err_msg": "Project ID must be numeric" }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

	}
--	---

E. Oozie

Oozie is used to manage Oozie Job on VGrid.

1. Submit Job

This method is used to submit Oozie Job.

- Request

Method	URL
POST	<code>{host}:{port}/{apikey}/submit_job</code> Example : <code>http://192.168.1.230:7005/bc4f313607048c0060fcd74ec77edbd0/submit_job</code> { "projectId": "11", "jarName": "/user/yava/vgrid/jar/newORM//newORM.jar", "jobName": "SCL_encaes256", "className": "newORM.encaes256" }

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	{ "err_code": 0, "data": { "oozie_job_id": "0000024-170322094651185-oozie-oozi-W", "job_status": "RUNNING" } }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

- Oozie Web Console

[Documentation](#)

Oozie Web Console									
Workflow Jobs Coordinator Jobs Bundle Jobs SLA System Info Instrumentation Settings									
All Jobs Active Jobs Done Jobs Custom Filter Server version [4.2.0]									
Job Id	Name	Status	Run	User	Group	Created	Started	Last Modified	Ended
1 0000024-170322094651185-oozie-oozi-W	SCL_encaes256	SUCCEEDED	0	yava		Thu, 30 Mar 2017 15:10:12 WIB	Thu, 30 Mar 2017 15:10:12 WIB	Thu, 30 Mar 2017 15:11:16 WIB	Thu, 30 Mar 2017 15:11:16 WIB

- Cluster Hadoop Application

Show 20 entries											Search:			
ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking				
application_1490148801695_0180	yava	[9097867CD5A94429838878F62D2C585B/5E03DA28E6074E4FB4DA5780939C1BE9] HGrid247-encaes256(1/1) .../java/purchases.encrypted	MAPREDUCE	default	Thu Mar 30 15:10:40 +0700 2017	Thu Mar 30 15:11:01 +0700 2017	FINISHED	SUCCEEDED		History				
application_1490148801695_0179	yava	oozie:launcher:T=java:W=SCL_encaes256:A=hgrid:ID=0000024-170322094651185-oozie-oozi-W	MAPREDUCE	level1	Thu Mar 30 15:10:19 +0700 2017	Thu Mar 30 15:11:13 +0700 2017	FINISHED	SUCCEEDED		History				

- Namenode Hadoop Information

Hadoop

Overview

Datanodes

Snapshot

Startup Progress

Utilities

Browse Directory

/user/java/purchases.encrypted

Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	yava	hadoop	0 B	30/3/2017 15.10.57	2	128 MB	_SUCCESS
-rw-r--r--	yava	hadoop	185 B	30/3/2017 15.10.56	2	128 MB	hgrid247-00000
-rw-r--r--	yava	hadoop	124 B	30/3/2017 15.10.56	2	128 MB	hgrid247-00001

Hadoop, 2015.

2. Get Job ID

This method is used to get Oozie Job ID.

- Request

Method	URL
GET	{host}:{port}/{apikey}/get_job_id Example : http://192.168.1.230:7005/bc4f313607048c0060fcd74ec77edbd0/get_ob_id

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
--------	----------

0	{ { "oozie_job_id": "0000024-170322094651185-oozie-oozi-W" } }
500	{ "err_code": 500, "err_msg": "Wrong apikey" }
4	{ "err_code": 4, "err_msg": "IP Address is not registered" }

3. Get Job Info

This method is used to get Oozie Job Information.

- Request

Method	URL
GET	{host}:{port}/{apikey}/get_job_info/{oozie_job_id} Example : http://192.168.1.230:7005/bc4f313607048c0060fcd74ec77edbd0/get_job_info/0000024-170322094651185-oozie-oozi-W

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{oozie_job_id}	Integer

- Response

Status	Response
0	{ "err_code": 0, "data": { "appPath": "hdfs://192.168.1.230:8020/user/yava/jobfile/workflow/newORM/workflow.xml", "acl": null, "status": "SUCCEEDED", "createdTime": "Thu, 30 Mar 2017 08:10:12 GMT", "conf": "<configuration>\r\n <property>\r\n <name>user.name</name>\r\n <value>yava</value>\r\n </property>\r\n <property>\r\n <name>oozie.use.system.libpath</name>\r\n <value>>true</value>\r\n </property>\r\n <property>\r\n <name>oozie.libpath</name>\r\n"

```

<value>hdfs://192.168.1.230:8020/user/java/oozie/hgrid/lib</value>\n
</property>\n\n <property>\n\n
<name>mapreduce.job.user.name</name>\n\n
<value>java</value>\n\n </property>\n\n <property>\n\n
<name>queueName</name>\n\n <value>level1</value>\n\n
</property>\n\n <property>\n\n <name>nameNode</name>\n\n
<value>hdfs://192.168.1.230:8020</value>\n\n </property>\n\n
<property>\n\n <name>jobTracker</name>\n\n
<value>192.168.1.231:8050</value>\n\n </property>\n\n
<property>\n\n <name>oozie.wf.application.path</name>\n\n
<value>hdfs://192.168.1.230:8020/user/java/jobfile/workflow/newORM/workflow.xml</value>\n\n </property>\n\n</configuration>",
  "lastModTime": "Thu, 30 Mar 2017 08:11:16 GMT",
  "run": 0,
  "endTime": "Thu, 30 Mar 2017 08:11:16 GMT",
  "externalId": null,
  "appName": "SCL_encaes256",
  "id": "0000024-170322094651185-oozie-oozi-W",
  "startTime": "Thu, 30 Mar 2017 08:10:12 GMT",
  "parentId": null,
  "toString": "Workflow id[0000024-170322094651185-oozie-oozi-W]
status[SUCCEEDED]",
  "group": null,
  "consoleUrl":
"http://java231.solusi247.com:11000/oozie?job=0000024-170322094651185-oozie-oozi-W",
  "user": "java",
  "actions": [
    {
      "errorMessage": null,
      "status": "OK",
      "stats": null,
      "data": null,
      "transition": "hgrid",
      "externalStatus": "OK",
      "cred": "null",
      "conf": "",
      "type": ":START:",
      "endTime": "Thu, 30 Mar 2017 08:10:12 GMT",
      "externalId": "-",
      "id": "0000024-170322094651185-oozie-oozi-W@:start:",
      "startTime": "Thu, 30 Mar 2017 08:10:12 GMT",
      "userRetryCount": 0,
      "externalChildIDs": null,
      "name": ":start:",
      "errorCode": null,
      "trackerUri": "-",
      "retries": 0,
      "userRetryInterval": 10,
      "toString": "Action name[:start:] status[OK]",
      "consoleUrl": "-",

```

```

"userRetryMax": 0
},
{
  "errorMessage": null,
  "status": "OK",
  "stats": null,
  "data": null,
  "transition": "end",
  "externalStatus": "SUCCEEDED",
  "cred": "null",
  "conf": "<java xmlns=\"uri:oozie:workflow:0.5\">\r\n <job-
tracker>192.168.1.231:8050</job-tracker>\r\n <name-
node>hdfs://192.168.1.230:8020</name-node>\r\n
<configuration>\r\n  <property>\r\n
<name>mapred.job.queue.name</name>\r\n
<value>level1</value>\r\n  </property>\r\n </configuration>\r\n
<main-class>newORM.encaes256</main-class>\r\n
<file>hdfs://192.168.1.230:8020/user/java/vgrid/jar/newORM/newOR
M.jar</file>\r\n <capture-output />\r\n</java>",
  "type": "java",
  "endTime": "Thu, 30 Mar 2017 08:11:16 GMT",
  "externalId": "job_1490148801695_0179",
  "id": "0000024-170322094651185-oozie-oozi-W@hgrid",
  "startTime": "Thu, 30 Mar 2017 08:10:12 GMT",
  "userRetryCount": 0,
  "externalChildIds": null,
  "name": "hgrid",
  "errorCode": null,
  "trackerUri": "192.168.1.231:8050",
  "retries": 0,
  "userRetryInterval": 10,
  "toString": "Action name[hgrid] status[OK]",
  "consoleUrl":
"http://cassandra04.solusi247.com:8088/proxy/application_149014880
1695_0179/",
  "userRetryMax": 0
},
{
  "errorMessage": null,
  "status": "OK",
  "stats": null,
  "data": null,
  "transition": null,
  "externalStatus": "OK",
  "cred": "null",
  "conf": "",
  "type": ":END:",
  "endTime": "Thu, 30 Mar 2017 08:11:16 GMT",
  "externalId": "-",
  "id": "0000024-170322094651185-oozie-oozi-W@end",
  "startTime": "Thu, 30 Mar 2017 08:11:16 GMT",

```

	<pre> "userRetryCount": 0, "externalChildIDs": null, "name": "end", "errorCode": null, "trackerUri": "-", "retries": 0, "userRetryInterval": 10, "toString": "Action name[end] status[OK]", "consoleUrl": "-", "userRetryMax": 0 }] } } </pre>
7	<pre> { "err_code": 7, "err_msg": "Job Id is not found" } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>

4. Get Job History

This method is used to get Oozie Job History.

- Request

Method	URL
GET	<pre> {host}:{port}/{apikey}/get_job_history/project/{project_id} </pre> <p>Example :</p> <pre> http://192.168.1.230:7005/bc4f313607048c0060fcd74ec77edbd0/get_job_history/project/11 </pre>

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String
	{project_id}	Integer

- Response

Status	Response
0	{

	<pre> "err_code": 0, "data": [{ "job_id": "0000024-170322094651185-oozie-oozi-W", "job_name": "SCL_encaes256", "job_status": "RUNNING", "job_create_date": "2017-03-30 15:10:12", "job_update_date": "2017-03-30 15:10:12", "job_workflow": "<workflow-app xmlns=\\"uri:oozie:workflow:0.5\\" name=\\"SCL_encaes256\\">\n <start to=\\"hgrid\\"/>\n <action name=\\"hgrid\\">\n <java>\n <job-tracker>\${jobTracker}</job- tracker>\n <name-node>\${nameNode}</name-node>\n <configurat", "project_id": 11, "user_id": 2 }] } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>

5. Generate XML

This method is used to generate XML.

- Request

Method	URL
GET	{host}:{port}/{apikey}/ generate_xml Example : http://192.168.1.230:7005/bc4f313607048c0060fcd74ec77edbd0/gene ate_xml

Type	Parameter	Value
HEAD	auth_key	String
URL_PARAMS	{host}	IP address
	{port}	Integer
	{apikey}	String

- Response

Status	Response
0	<pre> { "err_code": 0, </pre>

	<pre> "data": { "xml": "<?xml version='1.0'?>\n<configuration>\n <property>\n <name>user.name</name>\n <value>yava</value>\n </property>\n <property>\n <name>nameNode</name>\n <value>hdfs://192.168.1.230:8020</value>\n </property>\n <property>\n <name>jobTracker</name>\n <value>192.168.1.231:8050</value>\n </property>\n <property>\n <name>queueName</name>\n <value>level1</value>\n </property>\n <property>\n <name>oozie.use.system.libpath</name>\n <value>true</value>\n </property>\n <property>\n <name>oozie.libpath</name>\n <value>\${nameNode}/user/yava/oozie/hgrid/lib</value>\n </property>\n <property>\n <name>oozie.wf.application.path</name>\n <value>\${nameNode}/user/yava/jobfile/workflow/newORM/workflow. ml</value>\n </property>\n</configuration>", "workflow": "<workflow-app xmlns='uri:oozie:workflow:0.5' name='SCL_encaes256'>\n <start to='hgrid'/>\n <action name='hgrid'>\n <java>\n <job-tracker>\${jobTracker}</job tracker>\n <name-node>\${nameNode}</name-node>\n <configuration>\n <property>\n <name>mapred.job.queue.name</name>\n <value>level1</value>\n </property>\n </configuration>\n <main-class>newORM.encaes256</main- class>\n <file>\${nameNode}/user/yava/vgrid/jar/newORM/newORM.jar</file>\n <capture-output/>\n </java>\n <ok to='end'/>\n <error to='fail'/>\n </action>\n <kill name='fail'>\n <message>Workflow failed, error message[\${wf:errorMessage(wf:lastErrorNode())}]</message>\n </kill>\n <end name='end'/>\n</workflow-app>" } } </pre>
500	<pre> { "err_code": 500, "err_msg": "Wrong apikey" } </pre>
4	<pre> { "err_code": 4, "err_msg": "IP Address is not registered" } </pre>