Inventory API Documentation

Workspace Objects:

URL: <IP>:<Port>/workspaces/

Methods:

- GET (retrieves all workspaces)
- POST (creates a workspace object)

Body:

```
{
    "name": "Workspace Name"
}
```

Response:

```
{
    “workspaces”: [{"name": "Workspace Name"},]
}
```

Status Codes:

201 = Successful POST

200 = Successful GET

400 = Error

URL: <IP>:<Port>/workspaces/<workspace>/

Methods:

- GET (retrieves workspace information)
- DELETE (deletes a workspace object and all objects in that workspace)
- PUT (updates workspace information)

Body:

```
{
    "name": "Workspace Name"
}
```

Infrstructure Objects:

URL: <IP>:<Port>/<workspace>/vms/

Methods:

- GET (retrieves all of user’s virtual machines)
- POST (creates virtual machine object)

Body:

```
{
    "name": "VM Name",
    "description": "VM Description",
    "ip": "10.10.10.3/24",
    "state": "U",
    "area": "Area Name",
    "template": "Template Name",
    "networks": ["Network-01 Name", "Network-02 Name"]
}
```

Response:

```
{
    “vms”: [{"name": “VM Name"},]
}
```

states = [('C_D', 'Creating_Disk'), ('C_N', 'Configuring_Network'), ('U', 'UP'), ('D', 'Down')]

URL: <IP>:<Port>/<workspace>/networks/

Methods:

- GET (retrieves all of user’s networks)
- POST (creates network object)

Body:

```
{
    "name": "Network Name",
    "description": "Network Description",
    "state": "U"
}
```

Response:

```
{
    “networks”: [{"name": “Network Name"},]
}
```

states = [('U', 'Up'), ('C', 'Creating')]

URL: <IP>:<Port>/<workspace>/vms/<vm>/

Methods:

```
- GET (retrieves all information of a virtual machine)
- DELETE (deletes a virtual machine object)
- PUT (updates virtual machine information)
```

Body:

```
{
    "name": "VM Name",
    "description": "VM Description",
    "networks": [{“name”: "Network-01 Name"},],
    "state": "U"
}
```

states = [('C_D', 'Creating_Disk'), ('C_N', 'Configuring_Network'), ('U', 'UP'), ('D', 'Down')]

URL: <IP>:<Port>/<workspace>/networks/<network>

Methods:

```
- GET (retrieves all information of a network)
- DELETE (deletes a network object)
- PUT (updates network information)
```

Body:

```
{
    "name": "Network Name",
    "description": "Network Description",
    "state": "U"
}
```

states = [('U', 'Up'), ('C', 'Creating')]

URL: <IP>:<Port>/<workspace>/routers/

Methods:

- GET (retrieves all of user’s routers)
- POST (creates router object)

Body:

```
{
    "name": "Network Name"
}
```

Response:

```
{
    “routers”: [{"name": “Router Name"},]
}
```

URL: <IP>:<Port>/<workspace>/routers/<router>

Methods:

- GET (retrieves all information of a router)
- DELETE (deletes a router object)
- PUT (updates router information)

Body:

```
{
    "name": "Router Name"
}
```

URL: <IP>:<Port>/<workspace>/routers/<router>/interfaces/

Methods:

- GET (retrieves all of routers’ interfaces)
- POST (creates a router interface)

Body:

```
{
    "network": "Network-01",
    “ip”: “192.168.1.1/24”
}
```

Response:

```
{
    “interfaces”: [{"name": “Interface Name"},]
}
```

URL: <IP>:<Port>/<workspace>/routers/<router>/interfaces/<interface network>

Methods:

```
- GET (retrieves all information of a router interface)
- DELETE (deletes a router interface object)
- PUT (updates router interface information)
```

Body:

```
{
    "ip": "10.10.10.1/24"
}
```

Note:

Manipulating inventory objects (except Workspace and User) must be done via the controller service.

Template Objects:

URL: <IP>:<Port>/templates/

Methods:

```
- GET (retrieves all templates)
- POST (creates a template object)
```

Body:

```
{
    "name": "Template Name",
```

"description": "Template Description",

"os": "Template OS",

"cpu": int,

"ram": int,

"disk": int

}

Response:

{

“templates”: [{"name": “Template Name"},]

}

cpu = 2 (cores) ram = 2 (GiB) disk = 40 (GiB)

URL: <IP>:<Port>/templates/<template>/

Methods:

- GET (retrieves template information)
- DELETE (deletes a template object)
- PUT (updates a template object)

Body:

{

"name": "Template Name",

"description": "Template Description",

"os": "Template OS",

"cpu": int,

"ram": int,

"disk": int

}

cpu = 2 (cores) ram = 2 (GiB) disk = 40 (GiB)

Area Objects:

URL: <IP>:<Port>/areas/

Methods:

- GET (retrieves all areas)
- POST (creates an area object)

Body:

```
{
    "name": "Area Name",
    "subnet": "10.10.10.0/24",
    "next_ip": "10.10.10.2/24"
}
```

URL: <IP>:<Port>/areas/<area>/

Methods:

- GET (retrieves area information)
- DELETE (deletes an area object)
- PUT (updates area information)

Body:

```
{
    "name": "Area Name",
    "subnet": "10.10.10.0/24",
    "next_ip": "10.10.10.2/24"
}
```

Response:

```
{
    “areas”: [{"name": "Area Name"},]
}
```

Address Objects:

URL: <IP>:<Port>/address/<area>/get_ip/

Methods:

- GET (returns a free IP address of the area subnet)

User Objects:

URL: <IP>:<Port>/<workspace>/users/

Methods:

- GET (retrieves all users in a workspace)
- POST (adds a new user to a workspace)

Body:

```
{
    "name": "username",
    "vm_can_add": True,
    "vm_can_edit": False,
    "vm_can_delete": False,
    "network_can_add": True,
    "network_can_edit": False,
    "network_can_delete": False,
    "router_can_add": True,
    "router_can_edit": False,
    "router_can_delete": False,
    "user_can_add": True,
    "user_can_edit": False,
    "user_can_delete": False
}
```

Response:

```
{
    “users”: [{"name": "User Name"},]
}
```

URL: <IP>:<Port>/<workspace>/users/<username>/

Methods:

- GET (retrieves user information)
- DELETE (deletes a user)
- PUT (update user permissions)

Body:


```
{
    "vm_can_add": True,
    "vm_can_edit": False,
    "vm_can_delete": False,
    "network_can_add": True,
    "network_can_edit": False,
    "network_can_delete": False,
    "router_can_add": True,
    "router_can_edit": False,
    "router_can_delete": False,
    "user_can_add": True,
    "user_can_edit": False,
    "user_can_delete": False,
}
```