limeade Documentation

Release 1.0

Marc Rochow

September 09, 2012

CONTENTS

Willkommen zur Version 1.0 der limeade Dokumentation.

CHAPTER

ONE

EINFÜHRUNG

1.1 Überblick

limeade ist ein Projekt der SkyLime GbR und wurde von Marc Rochow als Bachelorarbeit bei der Hochschule Augsburg weiterentwickelt.

Das Webinterface stellt eine voll funktionstüchtige Webanwendung dar, die speziell für Webhosting Firmen gedacht ist. Es soll das Verwalten und Administrieren beschleunigen und vereinfachen.

1.1.1 Features

Verwalten und Anlegen von

- VHosts
- FTP
- MySQL Datenbanken
- Backups
- Cloud Instanzen
- Domains
- SSL Zertifikaten
- E-mail Adressen, Weiterleitungen und Mailbboxen
- VNC

1.2 Requirements

- Python == 2.7
- Django >= 1.3
- Node.js >= 0.6

1.2.1 weitere Anforderungen

- Celery und django-celery
- pyOpenSSL
- lxml
- Ipy
- libvirt
- RabbitMQ

1.2.2 Integrierte Anwendungen (Django)

- South
- django-uni-form

1.3 Installation

Der Source Code kann von Github geladen werden. Entweder per direkten Download oder über Git.

```
git clone git@github.com:fatality/limeade.git
```

1.3.1 Installation der Anforderungen

Die Installation der Anforderungen wird beispielhaft für **Arch Linux** beschrieben.

Celery und django-celery:

```
pip2 install -U Celery
pip2 install -U django-celery
```

pyOpenSSL:

```
pacman -S python2-pyopenssl
```

lxml:

```
pacman -S python2-lxml
```

Ipy:

```
pacman -S python2-ipy
```

libvirt:

Um libvirt und somit auch KVM benutzen zu können, muss der Computer Virtualisierung unterstützen. Dies lässt sich mit folgendem Befehl testen:

```
grep -E "(vmx|svm)" --color=always /proc/cpuinfo
```

Wenn die Ausgabe korrekt ist und der Computer Virtualisierung unterstützt kann libvirt, KVM und QEMU installiert und eingerichtet werden.

```
pacman -S qemu-kvm libvirt dnsmasq virt-manager
```

Um dnsmasq korrekt einzurichten empfiehlt sich folgende diese Anleitung. Libvirt als Normaluser verwenden zu können ist in dieser Anleitung beschrieben.

libvirt und TCP:

Die Datei */etc/libvirt/libvirtd.conf* öffnen und folgende Stellen ändern:

```
listen_tls = 0
listen_tcp = 1
auth_tcp=none
```

Die Deamon Datei */etc/conf.d/libvirtd* öffnen und den Eintrag *LIBVIRTD_ARGS* in *LIBVIRTD_ARGS="–listen"* ändern.

Als letzter Schritt die QEMU Konfiguration in libvirt (*/etc/libvirt/qemu.conf*) öffnen und *vnc_listen = "0.0.0.0"* eintragen bzw. den Kommentar entfernen.

Als nächstes kann mittels virt-manager eine VM angelegt werden. Die Daten der VM können im Django Admin später eingetragen werden.

RabbitMQ:

In den Arch User Repositories findet sich ein Paket für RabbitMQ welches installiert werden muss.

Anschließend wird der RabbitMQ als root gestartet:

```
rabbitmq-server
```

Läuft der Server, müssen folgende drei Schritte durchgeführt werden:

```
rabbitmqctl add_user limeade EimequuChuap8aa8ohyo
rabbitmqctl add_vhost limeade
rabbitmqctl set_permissions -p limeade limeade ".*" ".*" ".*"
```

Der Server läuft nun und empfängt Nachrichten, leitet diese aber noch nicht weiter. Dazu muss der Deamon limed mittels Celery gestartet werden. In der *settings.py* müssen dazu noch die Angaben für MySQL gemacht werden, damit völlig automatisch Datenbanken erstellt werden können.

1.4 Benutzung

Sollten Node.js, Python, Django und die anderen Abhängigkeiten installiert sein lässt sich die Webanwendung mit Django einrichten. Dazu sollte jedoch eine lokale Konfigurationsdatei erstellt werden. Eine Beispieldatei ist integriert (local_settings.py.example).

1.4.1 Django

Minimal sollte eine Datenbank angelegt werden und folgender Schritt durchgeführt werden:

```
cd web/limeade
python2 manage.py syncdb --migrate
```

Dies erstellt alle Tabellen in der Datenbank und zugleich auch einen Benutzer mit vollen Adminrechten. Starten lässt sich die Anwendung anschließend lokal mit

```
python2 manage.py runserver
```

Die Webanwendung läuft nun unter http://127.0.0.1:8000/system/ und kann im Admin mit Daten gefüttert werden (http://127.0.0.1:8000/admin/).

1.4.2 Node.js Proxy

Der Node.js Proxy wird im *proxy*-Verzeichnis ebenfalls ausgeführt:

```
node index.js
```

1.4.3 RabbitMQ und Celery

Zum Abschluss muss RabbitMQ und der Deamon im *limed*-Verzeichnis gestaret werden:

```
rabbitmq-server
celeryd
```

CHAPTER

TWO

API DOKUMENTATION

2.1 System

2.1.1 Models

Models for limeade system

class `limeade.system.models.`**`Contract`**`(`**args*, ***kwargs*`)`

Creates a contract for a person with a product.

Parameters

- **person** – the person
- **product** – the product

Example

```
>>> from limeade.system.models import Contract
>>> contract = Contract.objects.get(pk=1)
>>> contract
<Contract: Testuser / Testproduct>
```

`has_add_permission``(`*request*`)`

Returns true if user has permission to add a contract.

`has_change_permission``(`*request*, *obj*`)`

Returns true if user has permission to change the contract.

`has_delete_permission``(`*request*, *obj*`)`

Returns true if user has permission to delete the contract.

`queryset``(`*request*`)`

Returns a filtered queryset by request.

class `limeade.system.models.`**`Domain`**`(`**args*, ***kwargs*`)`

Model for a domain, which belongs to a contract.

Parameters

- **contract** – the contract
- **name** – the name of the domain

Example

```
>>> from limeade.system.models import Domain
>>> domain = Domain.objects.get(pk=1)
>>> domain
<Domain: testdomain.com>
```

Note: name of domain must be unique

owner()
Returns the owner of the domain.

class `limeade.system.models.`**Person**(**args*, ***kwargs*)
Creates a user profile for any registered person.

Parameters

- **user** – the user stored in django.auth.models.User
- **company** – if available, the company the user belongs to
- **address** – default address of user
- **parent** – persons can have own customers (e.g. reseller > customer)

Example

```
>>> from limeade.system.models import Person
>>> person = Person.objects.get(pk=1)
>>> person
<Person: Test User (Test Company)>
```

first_name()
Returns the first name of the user.

last_name()
Returns the last name of the user.

system_user_home()
Returns home directory of the user.

system_user_id()
Returns the user id.

system_user_name()
Returns unicode representation of the username.

username()
Returns the username of the user.

class `limeade.system.models.`**Product**(**args*, ***kwargs*)
Defines a product which belongs to a person.

Parameters

- **name** – name of the product
- **personalized** – true if its, othwerwise false
- **owner** – the person who owns this product

Example

```
>>> from limeade.system.models import Product
>>> product = Product.objects.get(pk=1)
>>> product
<Product: Testproduct>
```

has_add_permission(*request*)
Returns true if user has permission to add.

has_change_permission(*request*, *obj*)
Returns true if user has permission to change.

has_delete_permission(*request*, *obj*)
Returns true if user has permission to delete.

queryset(*request*)
Returns a filtered queryset.

limeade.system.models.**create_user_profile**(*sender*, *instance*, *created*, ***kwargs*)
This function creates a user profile for Django's get_profile() method, after the user is registered. This works with signals.

2.1.2 Views

Views for limeade system

limeade.system.views.**account**(*request*, **args*, ***kwargs*)
Display basic account information.

Parameters **request** – the request object

Returns the account template

limeade.system.views.**contract_add**(*request*, **args*, ***kwargs*)
Add a new contract for a customer.

Parameters

- **request** – the request object
- **slug** – the id of the customer

Returns template for adding a contract

limeade.system.views.**contract_customize**(*request*, **args*, ***kwargs*)
Personalize a plan for a customer.

Parameters

- **request** – the request object
- **slug** – the id of the customer
- **contract_id** – the id of the contract

Returns edit form template

limeade.system.views.**contract_delete**(*request*, **args*, ***kwargs*)
Remove a contract.

Parameters

- **request** – the request object
- **slug** – the id of the customer

- **contract_id** – the id of the contract

Returns redirect to customer view

`limeade.system.views.`**`customer_add`**(*request*, **args*, ***kwargs*)

Form to add a new customer.

Parameters **request** – the request object

Returns the add a customer template

`limeade.system.views.`**`customer_delete`**(*request*, **args*, ***kwargs*)

Remove a customer.

Parameters

- **request** – the request object
- **slug** – the id of the customer

Returns redirect to customer list

`limeade.system.views.`**`customer_edit`**(*request*, **args*, ***kwargs*)

Edit details of a customer.

Parameters

- **request** – the request object
- **slug** – the id of the customer

Returns a edit form template

`limeade.system.views.`**`customer_list`**(*request*, **args*, ***kwargs*)

Show a list of customers.

Parameters **request** – the request object

Returns the customer template

`limeade.system.views.`**`customer_manage`**(*request*, **args*, ***kwargs*)

Switch the current user to the customer. This allowes one to interact with the interface as if one was logged into the customers account.

Parameters

- **request** – the request object
- **slug** – the id of the customer

Returns redirect to ressources

`limeade.system.views.`**`customer_manage_return`**(*request*, **args*, ***kwargs*)

Return to the original user.

Parameters **request** – the request object

Returns redirect to ressources

`limeade.system.views.`**`customer_view`**(*request*, **args*, ***kwargs*)

View details of a customer.

Parameters

- **request** – the request object
- **slug** – the id of the customer

Returns the customer detail template

`limeade.system.views.`**`domain_add`**(*request*, *slug*, *contract_id*)
Add a new domain.

Parameters

- **request** – the request object
- **slug** – the id of the customer
- **contract_id** – the id of the contract

Returns a edit form template

`limeade.system.views.`**`domain_delete`**(*request*, *slug*, *contract_id*, *domain_id*)
Remove a domain.

Parameters

- **request** – the request object
- **slug** – the id of the customer
- **contract_id** – the id of the contract
- **domain_id** – the id of the domain

Returns redirect to customer view

`limeade.system.views.`**`product_add`**(*request*, **args*, ***kwargs*)
Create a new product. A product is a combination of different limitsets.

Parameters **request** – the request object

Returns a edit form template

`limeade.system.views.`**`product_delete`**(*request*, **args*, ***kwargs*)
Remove a product.

Parameters

- **request** – the request object
- **slug** – the id of the product

Returns a edit form template

`limeade.system.views.`**`product_edit`**(*request*, **args*, ***kwargs*)
Edit a product.

Parameters

- **request** – the request object
- **slug** – the id of the product
- **next** – redirect to view

Returns a edit form template

`limeade.system.views.`**`product_list`**(*request*, **args*, ***kwargs*)
Show a list of products.

Parameters **request** – the request object

Returns a list of products

`limeade.system.views.`**`ressources`**(*request*, **args*, ***kwargs*)
Display the aggregated ressources available to the current user. This depends on the products a user has subscribed to.

Parameters **request** – the request object

Returns the ressource template

2.2 Web

2.2.1 Models

Models for limeade web

class `limeade.web.models.`**`DefaultVHost`**`(`**args*, ***kwargs*`)`
Saves the default VHost.

Parameters

- **doamin** – the foreign key to the domain
- **vhost** – the foreign key to the vhost

class `limeade.web.models.`**`HTTPRedirect`**`(`**args*, ***kwargs*`)`
Creates http redirects

Parameters

- **name** – the name of the redirect
- **doamin** – the associated domain
- **to** – the redirect to

class `limeade.web.models.`**`Limitset`**`(`**args*, ***kwargs*`)`
Maximum limit available.

Parameters

- **products** – the foreign key to the product
- **vhosts** – maximum vhosts
- **redirects** – maximum redirects
- **webspace** – maximum webspace
- **cputime** – maximum cputime

static **`utilization`**`(`*user*, *ressource*`)`
Filters specific ressources.

class `limeade.web.models.`**`PoolIP`**`(`**args*, ***kwargs*`)`
The IP address pool.

Parameters

- **ip** – the IP
- **region** – the foreign key to the region

Example

```
>>> from limeade.web.models import PoolIP
>>> ip = PoolIP.objects.get(pk=1)
>>> ip
<Person: 127.0.0.1>
```

class `limeade.web.models.`**`SSLCert`**(**args*, ***kwargs*)
Creates SSL Certifications

Parameters

- **owner** – the user
- **serial** – the ssl cert
- **valid_not_before** – date of valid
- **valid_not_after** – date of valid
- **subject** – topic of cert
- **cn** – cert number
- **issuer** – the person
- **cert** – the cert
- **key** – key of the cert
- **ca** – the ca of the cert
- **ip** – the foreign key to the ip

`set_cert`(*cert*, *key*, *ca*)
Saves the certification.

class `limeade.web.models.`**`VHost`**(**args*, ***kwargs*)
Creates a vhost.

Parameters

- **name** – the name of the Vhost
- **domain** – foreign key to the domain
- **style** – php or python vhost
- **cert** – the ssl cert

Example

```
>>> from limeade.web.models import VHost
>>> vhost = VHost.objects.get(pk=1)
>>> vhost
<Person: Test VHost.testdomain.de>
```

`limeade.web.models.`**`get_vhosts`**(*user*)
Returns all vhosts that belongs to one user.

2.2.2 Views

Views for limeade web

`limeade.web.views.`**`poolip_add`**(*request*, **args*, ***kwargs*)
Adds an IP address.

Parameters **request** – the request object

Returns an edit form template

`limeade.web.views.`**`poolip_list`**`(`*request*, **args*, ***kwargs*`)`

Lists IP addresses.

Parameters **request** – the request object

Returns ip address pool template

`limeade.web.views.`**`redirect_add`**`(`*request*, **args*, ***kwargs*`)`

Create a new HTTP redirect.

Parameters **request** – the request object

Returns an edit form template

`limeade.web.views.`**`redirect_delete`**`(`*request*, **args*, ***kwargs*`)`

Remove a HTTP Redirect.

Parameters

- **request** – the request object
- **slug** – the id of the redirect

Returns redirect to http redirects list

`limeade.web.views.`**`redirect_edit`**`(`*request*, **args*, ***kwargs*`)`

Edit a HTTP redirect.

Parameters

- **request** – the request object
- **slug** – the id of the redirect

Returns an edit form template

`limeade.web.views.`**`redirect_list`**`(`*request*, **args*, ***kwargs*`)`

List HTTP redirects.

Parameters **request** – the request object

Returns list of http redirects

`limeade.web.views.`**`sslcert_add`**`(`*request*, **args*, ***kwargs*`)`

Add a new SSL certificate.

Parameters **request** – the request object

Returns an edit form template

`limeade.web.views.`**`sslcert_delete`**`(`*request*, **args*, ***kwargs*`)`

Remove a SSL certificate.

Parameters

- **request** – the request object
- **slug** – the id of the ssl cert

Returns redirects to ssl cert list

`limeade.web.views.`**`sslcert_list`**`(`*request*, **args*, ***kwargs*`)`

List a users SSL certificates.

Parameters **request** – the request object

Returns a list of ssl certificates

`limeade.web.views.`**`vhost_add`**`(`*request*, **args*, ***kwargs*`)`
Create a new web-vhost.

Parameters request – the request object

Returns an edit form template

`limeade.web.views.`**`vhost_catchall_delete`**`(`*request*, **args*, ***kwargs*`)`
Disable catch-all for this domain.

Parameters

- **request** – the request object
- **slug** – the id of the vhost

Returns redirects to vhost list

`limeade.web.views.`**`vhost_catchall_set`**`(`*request*, **args*, ***kwargs*`)`
Set this vhost as catch-all for the domain.

Parameters

- **request** – the request object
- **slug** – the id of the vhost

Returns redirects to vhost list

`limeade.web.views.`**`vhost_delete`**`(`*request*, **args*, ***kwargs*`)`
Remove a web-vhost.

Parameters

- **request** – the request object
- **slug** – the id of the vhost

Returns redirects to vhost list

`limeade.web.views.`**`vhost_edit`**`(`*request*, **args*, ***kwargs*`)`
Edit details of a web-vhost.

Parameters

- **request** – the request object
- **slug** – the id of the vhost

Returns an edit form template

`limeade.web.views.`**`vhost_list`**`(`*request*, **args*, ***kwargs*`)`
List of web-vhosts.

Parameters request – the request object

Returns list of web vhosts

2.3 MySQL

2.3.1 Models

Models for limeade mysql

class `limeade.mysql.models.`**`Limitset`**`(`**args*, ***kwargs*`)`
Saves maximum available databases.

Parameters

- **product** – the foreign key to the product
- **dbs** – the maxmimum databases

2.3.2 Views

Views for limeade mysql

`limeade.mysql.views.`**`db_add`**`(`*request*, **args*, ***kwargs*`)`
Form to add a new Database. Each database has a respective user with its own credentials.

Parameters **request** – the request object

Returns an edit form template

`limeade.mysql.views.`**`db_delete`**`(`*request*, **args*, ***kwargs*`)`
Drop a database.

Parameters

- **request** – the request object
- **slug** – the id of the database

Returns redirect to the database list

`limeade.mysql.views.`**`db_edit`**`(`*request*, **args*, ***kwargs*`)`
Form to set a new password for the database.

Parameters

- **request** – the request object
- **slug** – the id of the database

Returns an edit form template

`limeade.mysql.views.`**`db_list`**`(`*request*, **args*, ***kwargs*`)`
Show a list of the users databases.

Parameters **request** – the request object

Returns a list of databases

Raises TimeoutError

2.4 Mail

2.4.1 Models

Models for limeade mail

class `limeade.mail.models.`**`Account`**`(`**args*, ***kwargs*`)`
Creates a new mail account

Parameters

- **name** – the name of the account

- **domain** – the foreign key to the domain
- **password** – the password of the account

set_password(*password*)
Salt and hashes the password.

class limeade.mail.models.**Limitset**(**args*, ***kwargs*)
Creates a limitset.

Parameters

- **product** – the foreign key to the product
- **accounts** – the maximum number of mail accounts
- **redirects** – the maximum number of mail redirects
- **mailspace** – the maximum number of space

static utilization(*user*, *ressource*)
Returns the correct ressource.

class limeade.mail.models.**Redirect**(**args*, ***kwargs*)
Creates a new mail redirect

Parameters

- **name** – the name of the redirect
- **domain** – the foreign key to the domain
- **to** – redirect to

2.4.2 Views

Views for limeade mail

limeade.mail.views.**account_add**(*request*, **args*, ***kwargs*)
Add a new mail account.

Parameters **request** – the request object

Returns an edit form template

limeade.mail.views.**account_delete**(*request*, **args*, ***kwargs*)
Remove a mail account.

Parameters

- **request** – the request object
- **slug** – the id of the account

Returns redirects to mail account list

limeade.mail.views.**account_edit**(*request*, **args*, ***kwargs*)
Set a new password for an email account.

Parameters

- **request** – the request object
- **slug** – the id of the account

Returns an edit form template

`limeade.mail.views.`**`account_list`**`(`*request*, **args*, ***kwargs*`)`
List all mail accounts of a user.

Parameters **request** – the request object

Returns a list of mail accounts

`limeade.mail.views.`**`redirect_add`**`(`*request*, **args*, ***kwargs*`)`
Create a new mail redirect.

Parameters **request** – the request object

Returns an edit form template

`limeade.mail.views.`**`redirect_delete`**`(`*request*, **args*, ***kwargs*`)`
Remove a mail redirect.

Parameters

- **request** – the request object
- **slug** – the id of the redirect

Returns redirects to a list of mail redirects

`limeade.mail.views.`**`redirect_edit`**`(`*request*, **args*, ***kwargs*`)`
Edit a mail redirect.

Parameters

- **request** – the request object
- **slug** – the id of the redirect

Returns an edit form template

`limeade.mail.views.`**`redirect_list`**`(`*request*, **args*, ***kwargs*`)`
List all mail redirects.

Parameters **request** – the request object

Returns a list of mail redirects

2.5 FTP

2.5.1 Models

Models for limeade ftp

class `limeade.ftp.models.`**`Account`**`(`**args*, ***kwargs*`)`
Creates an ftp account.

Parameters

- **name** – the name of the account
- **password** – the password
- **vhost** – the foreign key to the vhost

`save``(`***kwargs*`)`
Saves the ftp account.

set_password(*password*)
Salt and hashes the password.

class limeade.ftp.models.**Limitset**(**args*, ***kwargs*)
Saves the maximum number of ftp accounts.

Parameters

- **product** – the foreign key to the product
- **accounts** – maximum number of ftp accounts

static utilization(*user*, *ressource*)
Returns the correct ressource.

2.5.2 Views

Views for limeade ftp

limeade.ftp.views.**account_add**(*request*, **args*, ***kwargs*)
Add a new FTP account.

Parameters **request** – the request object

Returns an edit form template

limeade.ftp.views.**account_delete**(*request*, **args*, ***kwargs*)
Remove an ftp account.

Parameters

- **request** – the request object
- **slug** – the id of the ftp account

Returns redirects to a list of ftp accounts

limeade.ftp.views.**account_edit**(*request*, **args*, ***kwargs*)
Change an ftp accounts password.

Parameters

- **request** – the request object
- **slug** – the id of the ftp account

Returns an edit form template

limeade.ftp.views.**account_list**(*request*, **args*, ***kwargs*)
Show a list of FTP accounts.

Parameters **request** – the request object

Returns a list of ftp accounts

2.6 Cluster

2.6.1 Models

Models for limeade cluster

class `limeade.cluster.models.`**`Region`**`(`**args*, ***kwargs*`)`
Creates a cluster region.

Parameters **name** – name of the region

class `limeade.cluster.models.`**`Server`**`(`**args*, ***kwargs*`)`
Creates a cluster server.

Parameters

- **hostname** – name of the region
- **ip** – the ip of the server
- **region** – foreign key to cluster region
- **services** – foreign key to cluster service
- **enabled** – indicates if this server is enabled

class `limeade.cluster.models.`**`Service`**`(`**args*, ***kwargs*`)`
Creates a cluster service.

Parameters **name** – name of the service

2.6.2 Views

Views for limeade cluster

`limeade.cluster.views.`**`region_add`**`(`*request*, **args*, ***kwargs*`)`
Add a new region to the cluster.

Parameters **request** – the request object

Returns an edit form template

`limeade.cluster.views.`**`region_delete`**`(`*request*, **args*, ***kwargs*`)`
Remove a region from the cluster

Parameters

- **request** – the request object
- **slug** – the id of the region

Returns redirect to list of regions

`limeade.cluster.views.`**`region_edit`**`(`*request*, **args*, ***kwargs*`)`
Edit a region.

Parameters

- **request** – the request object
- **slug** – the id of the region

Returns an edit form template

`limeade.cluster.views.`**`region_list`**`(`*request*, **args*, ***kwargs*`)`
Show a list of regions.

Parameters **request** – the request object

Returns a list of regions

`limeade.cluster.views.`**`server_add`**`(`*request*, **args*, ***kwargs*`)`
Add a new server to the cluster.

Parameters **request** – the request object

Returns an edit form template

`limeade.cluster.views.`**`server_delete`**(*request*, **args*, ***kwargs*)
Remove a server from the cluster.

Parameters

- **request** – the request object
- **slug** – the id of the server

Returns redirect to server list

`limeade.cluster.views.`**`server_disable`**(*request*, **args*, ***kwargs*)
Put a server into maintenance.

Parameters

- **request** – the request object
- **slug** – the id of the server

Returns redirect to server list

`limeade.cluster.views.`**`server_edit`**(*request*, **args*, ***kwargs*)
Edit a server.

Parameters

- **request** – the request object
- **slug** – the id of the server

Returns an edit form template

`limeade.cluster.views.`**`server_enable`**(*request*, **args*, ***kwargs*)
Enable a server after a maintenance is over.

Parameters

- **request** – the request object
- **slug** – the id of the server

Returns redirect to server list

`limeade.cluster.views.`**`server_list`**(*request*, **args*, ***kwargs*)
Show a list of servers.

Parameters **request** – the request object

Returns a list of servers

2.7 Cloud

2.7.1 Models

class `limeade.cloud.models.`**`Instance`**(**args*, ***kwargs*)
Creates a cloud instance

Parameters

- **hostname** – the hostname of the instance

- **sshkeys** – foreign key to the ssh keys
- **owner** – the owner of the instance
- **domain** – domain of the instance
- **node** – foreign key to the cloud node
- **active** – indicates if the instance is active
- **mac_addr** – the mac address of the instance

`generate_mac_addr()`
Creates a mac address.

`save(**kwargs)`
Saves the instance.

class `limeade.cloud.models.Limitset(*args, **kwargs)`
Maxmimum limit for an cloud instance.

Parameters

- **product** – foreign key to the product
- **cpu_cores** – max number of cpu cores
- **memory** – max number of memory
- **storage** – max number of storage

static `utilization(user, ressource)`
Returns the correct ressource.

class `limeade.cloud.models.Node(*args, **kwargs)`
Creates a new node in the cloud.

Parameters

- **name** – the name of the node
- **uri** – the unique resource identifier of the node

class `limeade.cloud.models.SSHKey(*args, **kwargs)`
Creates a SSH Key for the cloud instance.

Parameters

- **comment** – comment for this key
- **key** – the ssh key
- **owner** – the owner of this key

2.7.2 Views

Views for limeade cloud API calls

`limeade.cloud.views.api.instance_activate(request)`
Called when an instance is done provisioning.

Parameters **request** – the request object

Requires two GET Parameters:

- site_api_key: for security reasons

•instance: ID of the instance to activate

Returns a http response with data

`limeade.cloud.views.api.`**`instance_info`**`(`*request*, *slug*`)`
Returns information an instance can use to configure itself.

Parameters

- **request** – the request object
- **slug** – the mac address of the instance

It includes:

•the hostname

•SSH Keys

Returns a http response with data

Views for limeade cloud Instances

`limeade.cloud.views.instance.`**`instance_add`**`(`*request*, **args*, ***kwargs*`)`
View for adding a new instance.

Parameters **request** – the request object

Returns an edit form template

`limeade.cloud.views.instance.`**`instance_delete`**`(`*request*, **args*, ***kwargs*`)`
Delete a instance.

Parameters

- **request** – the request object
- **slug** – the id of the instance

Returns redirect to a list of cloud instances

`limeade.cloud.views.instance.`**`instance_list`**`(`*request*, **args*, ***kwargs*`)`
Show a list of the current users instances.

Parameters **request** – the request object

Returns list of instances

`limeade.cloud.views.instance.`**`instance_restart`**`(`*request*, **args*, ***kwargs*`)`
Restart a instance.

Parameters

- **request** – the request object
- **slug** – the id of the instance

Returns redirect to a list of cloud instances

`limeade.cloud.views.instance.`**`instance_start`**`(`*request*, **args*, ***kwargs*`)`
Start a instance.

Parameters

- **request** – the request object
- **slug** – the id of the instance

Returns redirect to a list of cloud instances

`limeade.cloud.views.instance.`**`instance_stop`**(*request*, **args*, ***kwargs*)
Stop a instance.

Parameters

- **request** – the request object
- **slug** – the id of the instance

Returns redirect to a list of cloud instances

Views for limeade cloud SSH Keys

`limeade.cloud.views.ssh.`**`sshkey_add`**(*request*, **args*, ***kwargs*)
Form to add a new SSH Key.

Parameters **request** – the request object

Returns an edit form template

`limeade.cloud.views.ssh.`**`sshkey_delete`**(*request*, **args*, ***kwargs*)
Delete a SSH Key.

Parameters

- **request** – the request object
- **slug** – the id of the ssh key

Returns redirect to a list of ssh keys

`limeade.cloud.views.ssh.`**`sshkey_list`**(*request*, **args*, ***kwargs*)
Show a list of the Users SSH Keys.

Parameters **request** – the request object

Returns a list of ssh keys

Views for limeade cloud VNC

`limeade.cloud.views.vnc.`**`instance_vnc`**(*request*, **args*, ***kwargs*)
Lets view the authenticated user a VNC console.

Parameters

- **request** – the request object
- **slug** – the virtual machine id

Returns the vnc template

Raises DoesNotExist

`limeade.cloud.views.vnc.`**`instance_vnc_auth`**(*request*, *slug*, *token*)
API call for websocket to tcp proxy to get the settings.

Parameters

- **request** – the request object
- **slug** – the virtual machine id
- **token** – the session id from the requesting user

Returns dictionary containing proxy settings

Raises ObjectDoesNotExist

CHAPTER

THREE

INDEX UND TABELLEN

Benötigten Informationen nicht gefunden? Versuche es im Index oder probiere die Suchfunktion:

- *genindex*
- *search*

PYTHON MODULE INDEX

INDEX