République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE **E**LECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin d'Etudes de MASTER ACADEMIQUE

Domaine : **Mathématiques et Informatique**
Filière : **Informatique**
Spécialité : **Conduite de projets informatiques(CPI)**

Présenté par
BOULAOUCHE Malika
BOUFATIS Dihia

Thème

Développement d'une interface Web autour du module Horizon d'Openstack (Cloud Computing)

Encadreur : Mr L.Djema
Co-Encadreur : Mr K.Belkacem

Remerciements

D'abord nous remercions le bon Dieu
de nous avoir donné
santé, courage, volonté et foi
pour réaliser ce travail.

Nous tenons à exprimer notre profonde
gratitude à notre promoteur

Enfin, un grand merci à tous ceux qui ont
contribué au bon déroulement
de ce modeste
travail. En particulier nos chers parents,
nos familles
et tous nos amis(es).

Dédicaces

Nous dédions ce modeste travail à nos parents, nos familles, nos camarades

nos proches nos amis(es) et enfin toutes personne ayant contribué au bon accomplissement de notre projet.

SOMMAIRE

Sommaire

Liste des figures :

Liste des tableaux :

Nomenclature :

KVM	Kilobyte (kernel)Virtual Machine
VM	Machines virtuelles.
IBM	International Business Machines.
CISCO	Cisco Systems est une entreprise informatique américaine spécialisée, à l'origine, dans le matériel réseau et depuis 2009 dans les serveurs.
CIGREF	Club informatique des grandes entreprises françaises.
VLAN	Virtual Local Area Network.
CRM	Customer Relationship Management
API	Application Programming Interface

Introduction générale

INTRODUCTION GÉNÉRALE

Face à l'augmentation continuelle des coûts de mise en place et de maintenance des systèmes d'informations, les entreprises externalisent de plus en plus leurs services informatiques en les confiant à des entreprises spécialisées comme les fournisseurs de Cloud. L'intérêt principal de cette stratégie pour les entreprises réside dans le fait qu'elles ne paient que pour les services effectivement consommés.

Le Cloud Computing est aujourd'hui le sujet phare dans le domaine des systèmes d'information et de communication. Après la virtualisation, le Cloud parait être la révélation qui va permettre aux entreprises d'être plus performantes et de gérer le coût des systèmes d'information plus sereinement. Mais suite à cette entrée fracassante nous pouvons tout de même nous demander ce qu'est le Cloud Computing ? C'est pour cela que ce travail de fin d'études de mastère s'intéresse à ce domaine tout nouveau, du moins pour nous.

Le terme Cloud Computing, ou « informatique dans les nuages », est un nouveau modèle informatique qui consiste à proposer les services informatiques sous forme de services à la demande, accessibles de n'importe où, n'importe quand et par n'importe qui. Cette nouvelle technologie permet à des entreprises d'externaliser le stockage de leurs données et de leur fournir une puissance de calcul supplémentaire pour le traitement de grosse quantité d'informations.

L'objectif de ce travail est justement d'approfondir et d'expérimenter nos connaissances sur ce thème de Cloud Computing, puis de faire son état de l'art, en vue de choisir la meilleure solution disponible à l'heure actuelle, de la déployer et l'évaluer. Pour se faire nous avons déployé un Cloud privée de type infrastructure en tant que service.

Ainsi, le présent manuscrit s'articule autour de quatre chapitres :

- Le premier chapitre nous donnons quelques définitions et généralités sur le Cloud,
- Le deuxième chapitre est consacré à la présentation d'Openstack et la description de ses différents composants ,
- Le troisième chapitre détaille la mise en place de la plate forme Openstack et l'architecture de

son installation , ainsi que les outils et les logiciels utilisés.

Enfin dans le dernier chapitre nous montrons l'interface Horizon après le changement du css pour répondre aux besoins de la société TychCloud ,ainsi que le projet (site) Ddjango crée pour cette société. .

Ce travaille se termine par une conclusion générale puis une partie annexe qui présente quelques notions évoquées dans l'étude.

Chapitre I:

Le Cloud Computing

I. Introduction :

Indéniablement, la technologie de l'internet se développe de manière exponentielle depuis sa création. Actuellement, une nouvelle "tendance" à fait son apparition dans le monde des IT (Technologies de l'information et de la communication), il s'agit du Cloud computing.

Cette technologie, s'appuyant sur le WEB 2.0, offre des occasions aux sociétés de réduire les coûts d'exploitation des logiciels par leurs utilisations directement en ligne. Divers fournisseurs comme Google, Amazon, IBM offrent une vaste gamme de services de Cloud Computing.

Cette technologie vient juste d'éclore, elle est au début de son exploitation mais déjà plusieurs acteurs majeurs cités précédemment adoptent leurs propres stratégies de pionnier qui déterminera l'utilisation du Cloud computing des entreprises souhaitant investir.

Beaucoup d'entreprises restent cependant sceptiques sur le cloud computing. La principale raison est l'intégrité et la sécurité des données car les DSI (direction des systèmes d'informations) restent frileuses de penser que leur données critiques sont dans un endroit incontrôlé et souvent inconnu.

Dans ce chapitre nous allons présenter les notions fondamentales du Cloud Computing, ses enjeux, ses évolutions et son utilité ainsi que la technologie qui la constitue et les différents acteurs du secteur.

Nous devons dans un premier temps étudier le Cloud Computing de manière générale (Définitions, Avantages, inconvénients…), dans un second temps nous allons étudier les trois services principaux, sur lesquels le Cloud Computing repose: applicatif, plateforme, infrastructure, qui ont donné naissance aux fameux SaaS/PaaS/IaaS. Et la dernière partie de ce chapitre présente les différents avantages et inconvénient du Cloud Computing, et met l'accent sur l'aspect de la sécurité du Cloud Computing

II. Historique :

Techniquement, le concept de Cloud Computing est loin d'être nouveau, il est même présent depuis des décennies. On en trouve les premières traces dans les années 1960, quand John McCartyaffirmait que cette puissance de traitement informatique serait accessible au public dans le futur. Le terme en lui-même est apparu plus couramment aux alentours de la fin du XXe siècle et il semblerait que Amazon.com soit l'un des premiers à avoir assemblé des data-center et fournit des accès à des clients.

Les entreprises comme IBM et Google ainsi que plusieurs universités ont seulement commencé à s'y intéresser sérieusement aux alentours de 2008, quand le Cloud Computing est devenu un concept à lamode.

Réalisant ce qu'ils pourraient faire de toute cette puissance, de nombreuses compagnies ont ensuite commencé à montrer un certain intérêt, puis à échanger leurs anciennes infrastructures et applications internes contre ce que l'on appelle les « pay per-use service » (services payés à l'utilisation) .

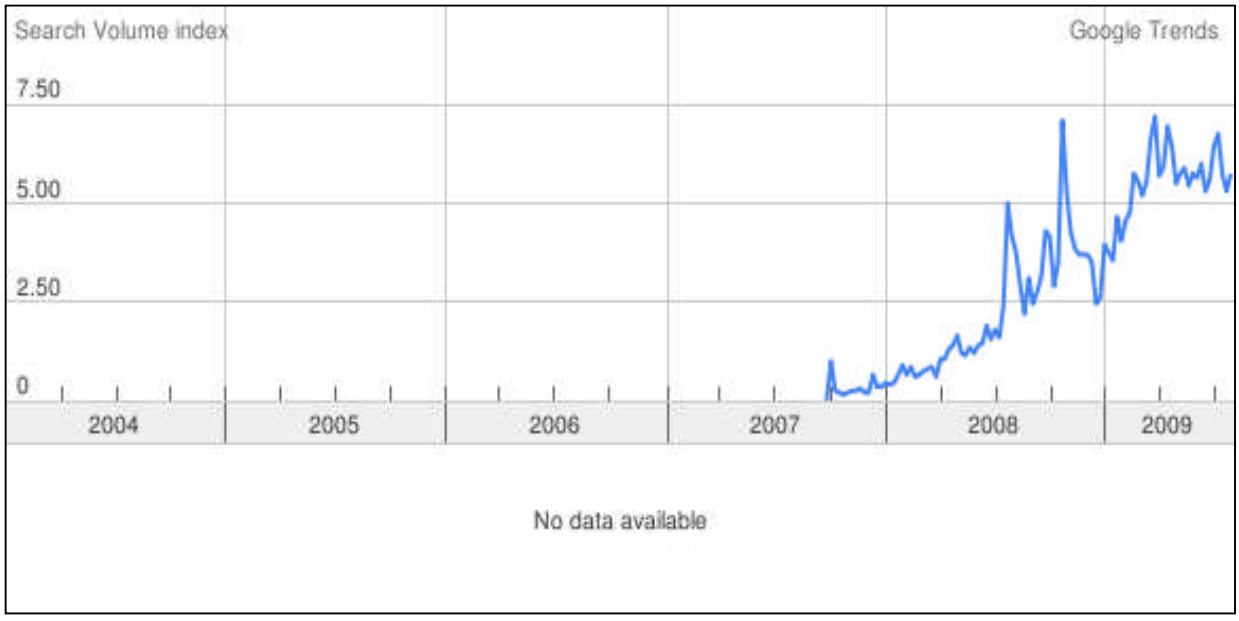


Figure 1. Evolution du Cloud Computing

Auparavant, seuls les superordinateurs permettaient de fournir cette puissance et étaient principalement utilisés par des gouvernements, des militaires, des laboratoires et des universités pour réaliser des calculs aussi complexes que prédire le comportement d'un avion en vol, les

changements climatiques ou la simulation d'explosions nucléaires. Désormais, des entreprises comme Google fournissent des applications qui exploitent le même type de puissance et sont accessibles à tout moment, de n' importe où et par tout un chacun via Internet.

Quelques universités prestigieuses ont également lancé leurs propres programmes de Cloud Computing en fournissant des accès à des maillages de centaines ou milliers de processeurs. Des entreprises comme IBM, ont récemment annoncé leur intention d'utiliser massivement le Cloud Computing à l'avenir. Ces derniers ont dévoilé un système ultra-performant connu sous le nom de « Blue Cloud »qui permettra d'aider les banques et diverses entreprises à distribuer leurs calculs sur un très grand nombre de machines sans posséder d'infrastructure eninterne.

Actuellement les experts sont convaincu que bientôt, nous utiliserons le Cloud Computing de la même manière que nous utilisons l'électricité, c'est à dire en payant uniquement ce que nous consommons sans même nous soucier des aspects techniques nécessaires au bon fonctionnement du système. Le principal facteur de développement restant le fait que toute cette puissance est à tout moment partagée par plusieurs utilisateurs et évite ainsi de perdre du « temps machine »à ne rien faire. Cela devrait également drastiquement réduire les coûts de développement et donc les prix**.[S1]**

III. Définition:

Le Cloud Computing, littéralement l'informatique dans les nuages est un concept qui consiste à déporter sur des serveurs distants des stockages et des traitements informatiques traditionnellement localisés sur des serveurs locaux ou sur le poste de l'utilisateur. Il consiste à proposer des services informatiques sous forme de service à la demande, accessible de n'importe où, n'importe quand et par n'importe qui, grâce à un système d'identification, via un PC et une connexion à Internet. Cette définition est loin d'être simple à comprendre, toutefois l'idée principale à retenir est que le Cloud n'est pas un ensemble de technologies, mais un modèle de fourniture, de gestion et de consommation de services et de ressources informatiques.

Pour Wikipédia, il s'agit : «d'un concept de déportation sur des serveurs distants des traitements informatiques traditionnellement localisés sur le poste client ».

Pour CISCO : « Le Cloud Computing est une plateforme de mutualisation informatique fournissant aux entreprises des services à la demande avec l'illusion d'une infinité de ressources »

.

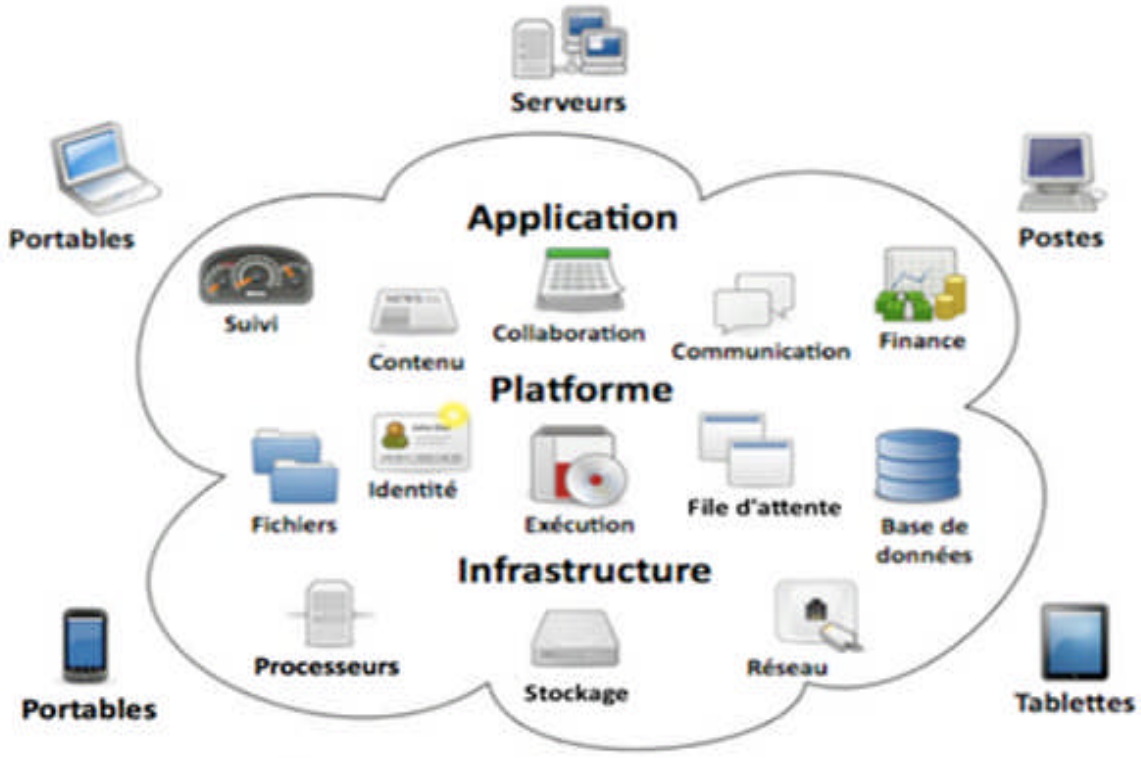


Figure 2. Le nuage

Pour le groupe de travail CIGREF le Cloud Computing est défini par les quatre points suivant :

- ✓ Un Cloud est toujours un espacevirtuel.
- ✓ Contenants des informations qui sontfragmentées.
- ✓ Dont les fragments sont toujours dupliqués et répartis dans cet espace virtuel, lequel peut être sur un ou plusieurs supportsphysiques.
- ✓ Qui possède « une console (programme) de restitution » permettant de reconstituerl'information. **[S2]**

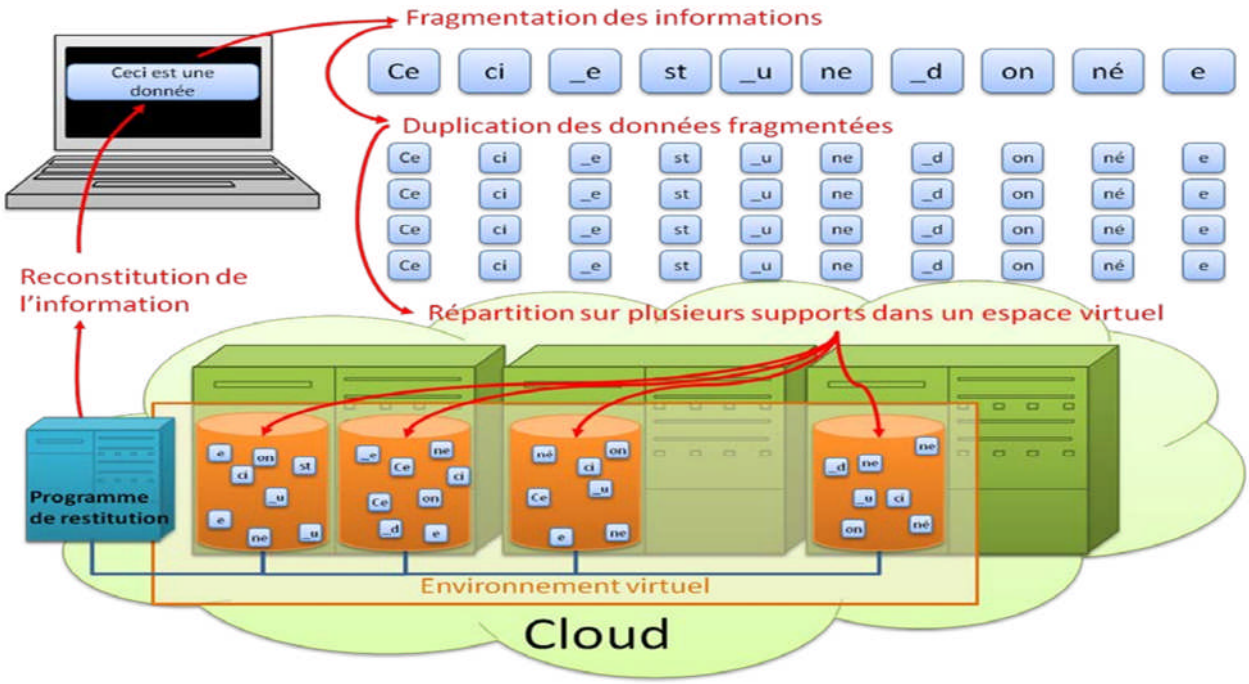


Figure 3. Les quatre points permettant d'identifier un Cloud

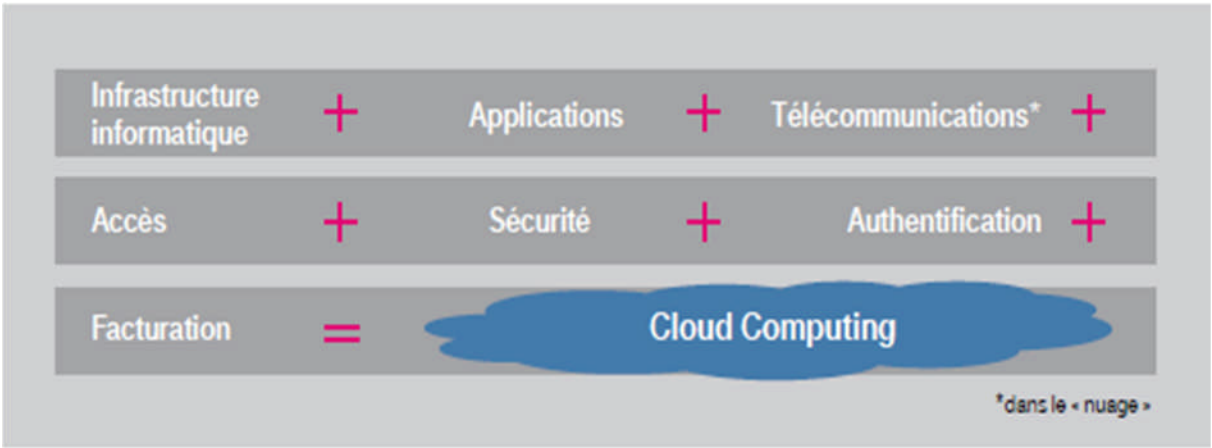


Figure 4. Le Cloud Computing

IV. Les caractéristiques du Cloud computing:

Le modèle Cloud Computing se différencie par les caractéristiques essentielles suivantes :

➢ Un libre service à la demande:

La capacité de stockage et la puissance de calcul sont adaptées automatiquement au besoin d'un consommateur. Ce qui contraste avec la technique classique des hébergeurs où le consommateur doit faire une demande écrite à son fournisseur en vue d'obtenir une augmentation de la capacité - demande dont la prise en compte nécessite évidemment un certain temps. En Cloud computing la demande est automatique et la réponse est immédiate.

➢ Mutualisation de la ressource:

Permet de combiner des ressources hétérogènes (matériel, logiciel, trafic réseau) en vue de servir plusieurs consommateurs à qui les ressources sont automatiquement attribuées[6]. La mutualisation améliore l'évolutivité et l'élasticité et permet d'adapter automatiquement les ressources aux variations de la demande.

➢ Accès aux services par l'utilisateur à la demande:

La mise en œuvre des systèmes est entièrement automatisée et c'est l'utilisateur, au moyen d'une console de commande, qui met en place et gère la configuration à distance.

➢ Ouverture:

les services de Cloud computing sont mis à disposition sur l'Internet, et utilisent des techniques standardisées qui permettent de s'en servir aussi bien avec un ordinateur qu'un téléphone ou une tablette.

➢ Redimensionnement rapide(élasticité):

La mise en ligne d'une nouvelle instance d'un serveur est réalisée en quelques minutes, l'arrêt et le redémarrage en quelques secondes. Toutes ces opérations peuvent s'effectuer automatiquement

par des scripts. Ces mécanismes de gestion permettent de bénéficier pleinement de la facturation à l'usage en adaptant la puissance de calcul au trafic instantané.

- **Facturation à l'usage:**

Il n'y a généralement pas de coût de mise en service (c'est l'utilisateur qui réalise les opérations). La facturation est calculée en fonction de la durée et de la quantité de ressources utilisées. Une unité de traitement stoppée n'est pas facturée.

- **Accès réseau large bande:**

Ces centres de traitement sont généralement raccordés directement sur le backbone Internet pour bénéficier d'une excellente connectivité. Les grands fournisseurs répartissent les centres de traitement sur la planète pour fournir un accès aux systèmes en moins de 50 ms de n'importe quel endroit.

- **Réservoir de ressources (non localisées):**

La plupart de ces centres comportent des dizaines de milliers de serveurs et de moyens de stockage pour permettre des montées en charge rapides. Il est souvent possible de choisir une zone géographique pour mettre les données "près" des utilisateurs**.[S3]**

V. Eléments constructifs du cloud computing:

Les éléments pouvant constitué le système Cloud sont les suivants :

- Les DataCenter
- La virtualisation

V .1-Les Datacenter:

V .1.1-Définition:

Le data center (centre de traitement des données en français) est l'un des éléments nécessaires au traitement et stockage des données numériques. Indispensable à Internet, il a connu un fort développement avec l'essor du Cloud computing. Concrètement, il s'agit d'un lieu physique contenant les serveurs informatiques qui stockent les données numériques et dans lequel les

entreprises peuvent notamment louer un espace de stockage et ainsi éviter la présence de serveurs dans leurs locaux.

V .1.2- Caractéristiques:

Selon la quantité de données à stocker, les caractéristiques d'un Datacenter seront différentes :

- Le type de serveur employé et sa capacité maximale
- Le système de climatisation et de ventilation
- La consommation énergétique globale

a/ Serveur employé : Il existe différents types de serveurs, qui varient dans leur performance de transmission et leur capacité de stockage. Les derniers en place sont les serveurs lames, possédant une capacité de stockage supérieure aux modèles précédents grâce à la mutualisation des composants logiciel dans un châssis, permettant au serveur d'être plus compact.

b/ Température ambiante : La température d'un data center doit être constante et de plus ou moins 20 °C. Elle est assurée par une climatisation classique ainsi que des systèmes de refroidissement par eau dans les villes plus chaudes. Lorsque le local est situé dans une ville où la température moyenne est proche ou en dessous des 20 °C souhaités, un système de récupération d'air froid extérieur pourra être utilisé. Cela permet des économies d'énergie, devenu un nouvel objectif d'évolution (Green Computing). La disposition des serveurs en allées a également été pensée de manière à répartir la chaleur qu'ils dégagent.

c/ Alimentation électrique : L'alimentation électrique peut être assurée par plusieurs circuits différents et complétée par des systèmes de batteries de secours (UPS) ou de générateurs. L'objectif est d'alimenter les systèmes selon le principe du No-Break (zéro coupure). Une défaillance du système électrique aurait pour effet immédiat de rendre impossible l'accès aux données et pourrait même endommager le matériel**.[S4]**

V .2-La virtualisation:

V .2.1- Définition :

Nous voyons que la virtualisation repose sur trois éléments importants :

1. L'abstraction des ressources informatiques ;

2. La répartition des ressources par l'intermédiaire de différents outils, de manière à ce que celles-ci puissent être utilisées par plusieurs environnements virtuels ;

3. La création d'environnements virtuels.

Ces trois concepts fondamentaux nous amènent à donner la définition suivante de la virtualisation :

La virtualisation est un processus qui va permettre de masquer les caractéristiques physiques d'une ressource informatique de manière à simplifier les interactions entre cette ressource et d'autres systèmes, d'autres applications et les utilisateurs. Elle va permettre de percevoir une ressource physique comme plusieurs ressources logiques et, inversément, de percevoir plusieurs ressources physiques comme une seule ressource logique.

V .2.2- Les domaines de la virtualisation :

1/ La virtualisation d'applications:

La virtualisation d'application est une technologie logicielle qui va permettre d'améliorer la portabilité et la compatibilité des applications en les isolant du système d'exploitation sur lequel elles sont exécutées. Elle consiste à encapsuler l'application et son contexte d'exécution système dans un environnement cloisonné. La virtualisation d'application va nécessiter l'ajout d'une couche logicielle supplémentaire entre un programme donné et le système d'exploitation ; son but est d'intercepter toutes les opérations d'accès ou de modification de fichiers ou de la base de registre 1 afin de les rediriger de manière totalement transparente vers une localisation virtuelle (généralement un fichier). Puisque cette opération est trans-parente, l'application n'a pas notion de son état virtuel. Le terme virtualisation d'application est trompeur puisqu'il ne s'agit pas de virtualiser l'application mais plutôt le contexte au sein duquel elle s'exécute (registres du processeur, système de fichiers,...).

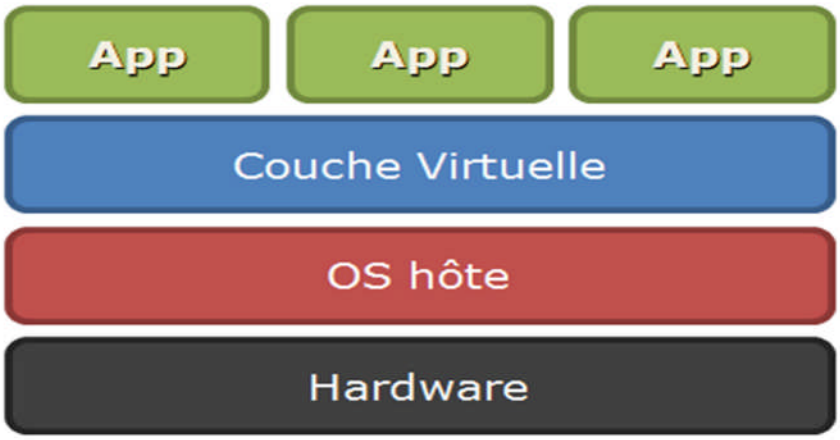


Figure 5. Virtualisation d'applications

La virtualisation d'applications a de nombreux avantages : elle permet d'exécuter des applications qui ont été développées pour d'autres environnements d'exécution (ex ; Wine permet d'exécuter des applications Windows sur une plateforme Linux) ; elle protège le système d'exploitation hôte en s'assurant que l'application virtualisée ne viendra pas interagir avec les fichiers de configuration du système ; elle évite de faire appel à une machine virtuelle qui ++consomme plus de ressources ; elle autorise l'exécution de code incorrect (ex :une application pourrait vouloir écrire un fichier dans un répertoire système dont elle ne possède que les droits en lecture).

2/ La virtualisation de réseaux :

De manière générale, la virtualisation des réseaux consiste à partager une même infrastructure physique (débit des liens, ressources CPU des routeurs,...) au profit de plusieurs réseaux virtuels isolés. Un VLAN (Virtual Local Area Network) est un réseau local regroupant un ensemble de machines de façon logique et non physique. Puisqu'un VLAN est une entité logique, sa création et sa configuration sont réalisées de manière logicielle et non matérielle. On distingue plusieurs types de réseaux virtuels :

- **Les réseaux virtuels de niveau 1:** appelés réseaux virtuels par port (port-based VLAN) : ils définissent un réseau virtuel en fonction des ports de raccordement sur le commutateur (switch). Ainsi, chaque port du commutateur est associé à un réseau virtuel, indépendamment de la machine qui y est physiquement raccordée. Le principal inconvénient d'un VLAN de niveau 1 est sa rigidité : si une station se raccorde physiquement au réseau par l'intermédiaire

d'un autre port du commutateur, alors il est nécessaire de reconfigurer ce commutateur afin de réintégrer la station dans le bon réseau virtuel.

- **Les réseaux virtuels de niveau 2:** appelés réseaux virtuels par adresse MAC (MAC address-based VLAN) : ils consistent à définir un réseau virtuel sur base des adresses MAC des stations. Une adresse MAC est un identifiant unique implémenté dans chaque adaptateur réseau. Ce type de VLAN est beaucoup plus souple que le précédent car il est indépendant de la localisation de la machine.
- **Les réseaux virtuels de niveau 3 :**On distingue principalement deux types de VLAN de niveau 3 :

  - Les réseaux virtuels par adresse de sous-réseau (Network address-based VLAN) : ils déterminent les réseaux virtuels sur base de l'adresse IP source des segments. Ce type de réseau virtuel est très flexible puisque les commutateurs adaptent automatiquement leur configuration lorsqu'une station est déplacée. En revanche, une légère dégradation des performances peut se faire resentir puisque les segments doivent être analysés plus minutieusement.
  - Les réseaux virtuels par protocole (Protocol-based VLAN). Dans ce cas, les réseaux virtuels sont créés sur base des protocoles utilisés (TCP/IP, IPX,...) et les stations sont regroupées en réseaux virtuels suivant le protocole qu'elles utilisent.

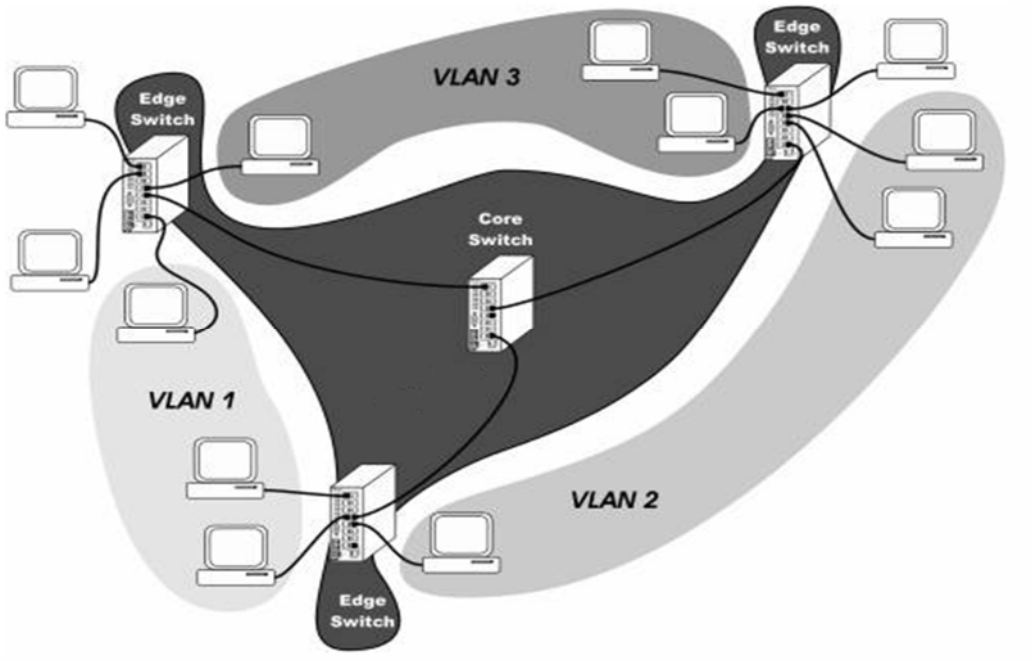


Figure 6. Réseaux virtuels

Les avantages qu'offrent les réseaux virtuels sont les suivants : une réduction du traffic de diffusion (broadcast) puisque celui-ci est à présent contenu au sein de chaque réseau virtuel ; une sécurité accrue puisque l'information est encapsulée dans une couche supplémentaire ; une meilleure flexibilité puisqu'une modification de la structure des réseaux peut être réalisée en modifiant la configuration du commutateur.

3/ La virtualisation de stockage :

Réalité physique de l'espace de stockage. Son but est de faire abstraction des périphériques de stockage utilisés et des interfaces qui leur sont associés (SATA, SCSI,...) afin de limiter l'impact des modifications structurelles de l'architecture de stockage. Ce type de virtualisation fait appel à une application d'administration de volumes logiques (Logical Volume Manager, LVM). Il s'agit d'une couche logicielle qui va permettre de regrouper plusieurs espaces de stockage, appelés volumes physiques, pour ensuite découper cet espace global suivant la demande en partitions virtuelles appelées volumes logiques. Ce processus de virtualisation peut être vu comme une extension du modèle de partitionnement classique des disques dur.

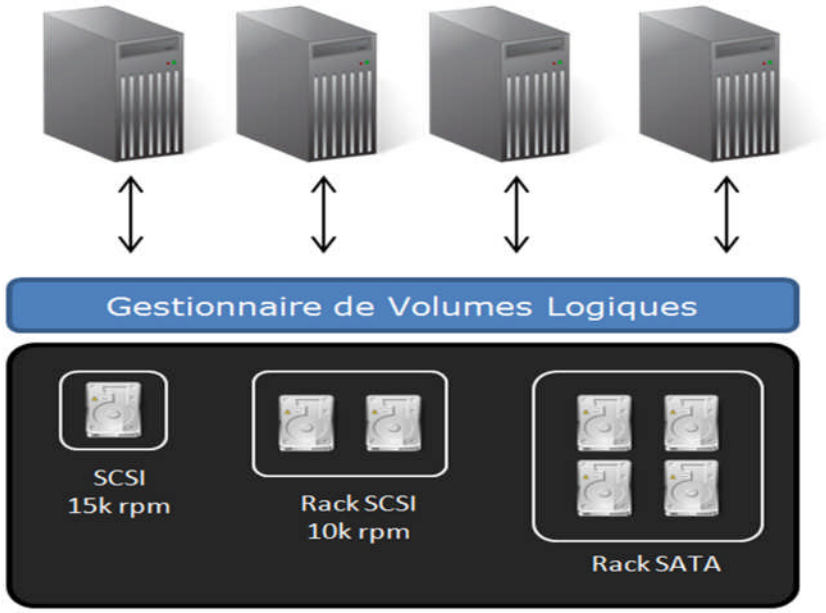


Figure 7. Virtualisation de stockage

La virtualisation de stockage permet :

- d'adjoindre un périphérique de stockage supplémentaire sans interruption des services ;

- de regrouper des unités de disques durs de différentes vitesses, de différentes tailles et de différents constructeurs ;
- de réallouer dynamiquement de l'espace de stockage. Ainsi, un serveur nécessitant un espace de stockage supplémentaire pourra rechercher des ressources non allouées sur le disque logique. Inversément, un serveur nécessitant moins d'espace de stockage pourra libérer cet espace et le rendre disponible pour d'autres serveurs.

4/ La virtualisation de serveurs:

La virtualisation des serveurs consiste à masquer les ressources du serveur, c.-à-d. le nombre et les caractéristiques de chaque machine physique, de chaque processeur et de chaque système d'exploitation pour les utilisateurs de ce serveur. L'administrateur du serveur va utiliser un logiciel grâce auquel il va diviser un serveur physique (constitué ou non de plusieurs machines distinctes) en plusieurs environnements virtuels isolés les uns des autres. Ces environnements isolés sont parfois appelés serveurs privés virtuels, hôtes, instances ou émulations. La virtualisation de serveurs s'inscrit dans une tendance globale qui tend à promouvoir la virtualisation au sein des entreprises en faisant notamment appel à la virtualisation de stockage et à la virtualisation de réseaux. Cette tendance est une composante dans le développement de systèmes autonomes. Un système est dit autonome si il est capable de s'auto-gérer sur base de l'activité qu'il perçoit, sans aucune intervention externe, et en conservant les détails de son implémentation invisibles pour l'utilisateur.

V .2.3- Les avantages de la virtualisation :

Au cours des dernières années, la virtualisation semble s'être imposée comme un élément incontournable au sein des entreprises et ce sont principalement les serveurs qui sont au centre de toutes les attentions. La virtualisation présente les avantages suivants :

- **Une optimisation de l'infrastructure:** La virtualisation permet d'optimiser la charge de travail des serveurs physiques. En effet, il y a quelques années, la relation une application ↔ un serveur était encore largement répandue. Cependant, comme le montre le schéma 5, cette relation introduit un gaspillage important des ressources puisqu'on estime que la charge moyenne d'un serveur se situe entre 5% et 15% . L'idée est alors de récupérer ces ressources disponibles afin d'en faire bénéficier d'autres applications. La virtualisation va apporter une solution efficace : plutôt que de faire tourner une seule application sur le serveur

physique, on va installer sur celui-ci plusieurs serveurs virtuels exécutant chacun une application bien précise, et c'est le logiciel de virtualisation qui se charge de répartir équitablement les ressources entre les différentes instances. De cette manière, comme montre le schéma 6, on optimise le rendement de chacun des serveurs physiques.

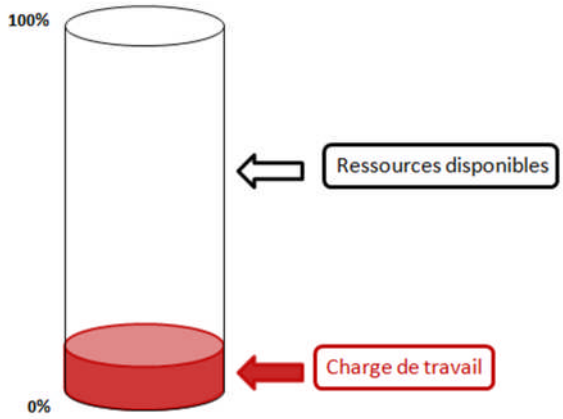


Figure 8. Rendement d'un serveur en absence d'une virtualisation

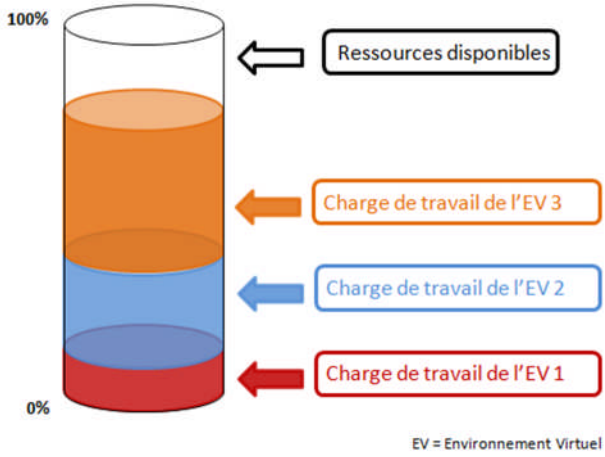
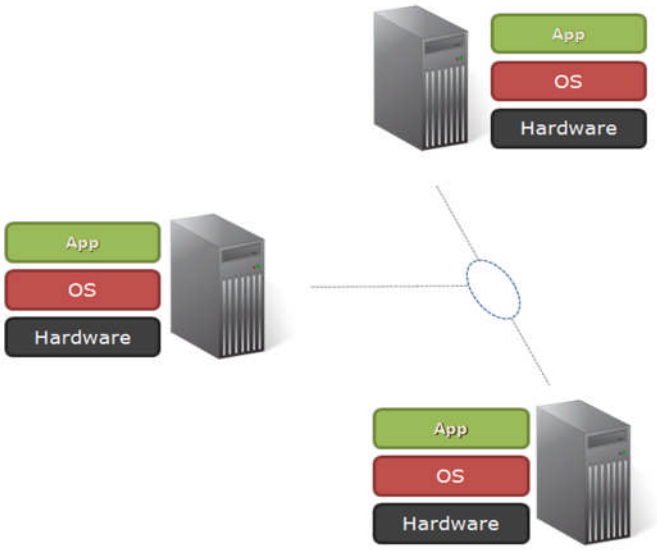


Figure 9. Rendement d'un serveur en présence d'une virtualisation

- **Une réduction de l'infrastructure physique et des économies d'énergies:** Les centres de données sont d'importants consommateurs de ressources. VMware affirme qu'un serveur au repos consomme jusqu'à 30% du pic de consommation électrique et selon Intel, 36,4% de l'énergie est utilisée par les composants physiques des serveurs (processeurs, mémoire,...) tandis que les 63,6% restants sont consommés par les équipements de climatisation. Ces chiffres sont interpellant et la virtualisation de l'infrastructure d'une entreprise doit devenir un de ses projets prioritaires. Puisque plusieurs applications peuvent à présent cohabiter de manière efficace sur un même système, les entreprises vont pouvoir réduire le nombre de machines de leur centre de données en investissant à nouveau dans d'importants mainframes avec lesquels réapparaît le modèle de centralisation de la puissance de calcul. Par conséquent, les entreprises font des économies en terme d'espace mais aussi en terme de frais de ventilation et d'alimentation. La réduction des coûts d'alimentation et de refroidissement s'inscrit également dans une tendance générale à la promotion du respect de l'environnement. Selon Gartner, une entreprise américaine de conseil et de recherche dans le domaine des techniques avancées, l'industrie informatique serait responsable de 2% des émissions mondiales de CO_2.

La virtualisation aurait permis d'économiser 5 milliards de kWh d'énergie électrique (source :Vmware).



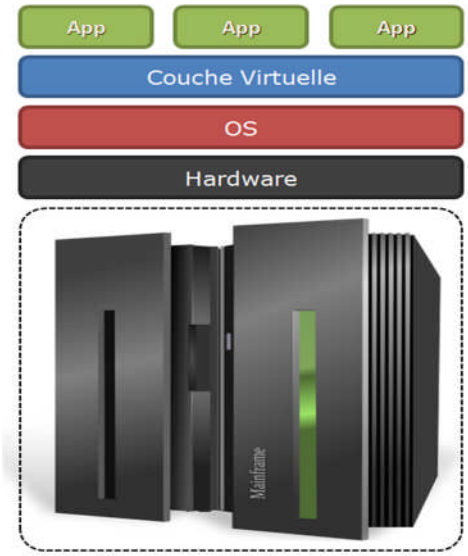


Figure 10. Centre de calcul distribué. **Figure 11. Centre de calcul centralisé.**

- **Une reprise automatique lors des incidents:** La virtualisation permet d'améliorer la prévention et la gestion des pannes ainsi que le plan de reprise de l'activité du système. En effet, les équipements virtuels étant constitués d'un ensemble de fichiers, il est très simple de les sauvegarder. Si un problème survient sur une machine physique, les fichiers qui ont été sauvegardés auparavant pourront rapidement être restaurés sur une autre machine physique, en attendant que la machine virtuelle initiale soit redémarrée.
- **Une optimisation de la sécurité des données:** Par la centralisation des ressources applicatives au sein du centre de données, la virtualisation contribue à sécuriser l'accès et l'usage des données de l'entreprise. Il est en effet beaucoup plus simple de contrôler l'accès aux données lorsqu'elles sont regroupées en un lieu que lorsqu'elles sont réparties sur l'ensemble des sites de l'entreprise, comme c'est le cas dans un modèle d'informatique distribué. Il est néanmoins indispensable de disposer d'un système de synchronisation qui permettra de sauvegarder l'ensemble des informations sur un site extérieur. De cette manière, on limite les risques liés à la destruction du centre de donnée initial.
- **Une facilité de migration:** La virtualisation apporte la possiblité de migrer facilement un environnement virtuel d'une machine physique vers une autre, facilitant ainsi la mise à jour du

centre de donnée ou le remplacement de matériel défectueux. De nombreux outils d'aide à la migration ont été développés. Ils peuvent être regroupés en 3 grandes catégories : P2V (Physical to Virtual), V2P (Virtual to Physical) et V2V (Virtual to Virtual). Les outils P2V sont généralement utilisés dans les projets de virtualisation d'une infrastructure informatique. Ils consistent à convertir les serveurs physiques en serveurs virtuels. Les outils V2P, qui permettent de convertir des serveurs virtuels en serveurs physiques, sont peu demandés mais une telle fonctionnalité rassure les entreprises qui souhaitent utiliser de la virtualisation ; elle leur permet, si elles le désirent, de faire marche arrière. Enfin, les outils V2V, qui permettent de convertir des serveurs virtuels d'un format dans un autre, sont principalement utilisés au sein d'entreprises qui utilisent des gestionnaires de systèmes virtuels différents.

- **Flexibilité et compatibilité:** La virtualisation est un outil efficace en terme de flexibilité et de compatibilité. En effet, elle supprime toute dépendance entre une application donnée et et l'aspect matériel de la machine sur laquelle elle est exécutée. Il devient alors possible d'exécuter sur un système une application qui a été développée à destination d'un autre système.
- **Un cloisonnement:** Le développement d'une application s'accompagne nécessairement de phases de tests au cours desquelles le programmeur s'assure du bon fonctionnement et de la stabilité de son logiciel. Néanmoins, il est parfois risqué d'exécuter une application lorsque l'on n'est pas certain du résultat qui sera produit. Pour réduire les risques liés à des applications peu fiables, la virtualisation peut être utilisée pour créer des environnements isolés et sécurisés qui vont servir de plateformes d'essai. Il devient dès lors possible d'itérer un grand nombre de fois un processus de test sans craindre de déstabiliser la machine physique sur laquelle s'exécute l'application.

V .2.4- Les inconvénients de la virtualisation :

- **Un point de défaillance unique:** Il s'agit probablement du plus gros désavantage de la virtualisation. Puisque plusieurs environnements virtuels s'exécutent sur une unique machine physique, si cette machine tombe en panne, alors les services fournis par les environnements virtuels sont interrompus.
- . **Un recours à des machines puissantes:** La virtualisation permet de réaliser des économies puisque moins de machines physiques sont nécessaires. Néanmoins, les outils de virtualisations sont des applications très gourmandes en ressources et nécessitent des machines puissantes. Il est évidemment possible d'utiliser la virtualisation sur des machines plus modestes, mais un

manque de mémoire ou de capacité CPU peut faire chuter les performances de manière dramatique.

- **Une dégradation des performances:** Bien qu'elle soit implémentée sur des machines puissantes, la virtualisation peut réduire les performances des applications. Suivant le type de virtualisation envisagé, cette perte de performances peut ou non être significative. Ce problème est d'autant plus embarassant qu'il est difficile d'estimer à l'avance l'impact qu'aura la virtualisation sur une application donnée. Dans certains cas, la dégradation des performances est telle qu'il est préférable de conserver l'application dans un environnement physique.
- **Une complexité accrue de l'analyse d'erreurs:** La virtualisation d'un serveur implique des changements importants dans l'infrastructure du système. La couche de virtualisation vient s'ajouter aux autres et apporte potentiellement avec elle un ensemble de nouveaux problèmes. La difficulté de cette couche additionnelle réside dans le fait que, si quelque chose ne fonctionne pas correctement, la localisation du problème peut demander des efforts considérables.
- **Une impossibilité de virtualisation:** Bien que, dans la plupart des cas, il n'est pas possible de prédire si une application se comportera normalement une fois virtualisée, il existe des applications qui sont connues pour rencontrer des difficultés lors de leur virtualisation. Les bases de données sont un exemple de telles applications. En effet, les bases de données ont recours à de nombreux accès disques et le délai d'accès supplémentaire introduit par la virtualisation peut dans certain cas rendre l'application inutilisable.
- **Un déploiement massif de serveurs:** Il ne s'agit pas réellement d'un désavantage de la virtualisation en tant que tel, mais d'un désavantage de la facilité de mise en place d'un nouvel environnement virtuel. En effet, certains administrateurs estiment qu'il est plus sécurisant de déployer un serveur dans un nouvel environnement virtuel, isolé des autres, plutôt qu'au sein d'un système d'exploitation qui exécute simultanément d'autres applications. Cependant, l'administration et les coûts d'une telle opération ajoutent une charge supplémentaire qu'il est difficile d'estimer à l'avance.
- **Un problème de standardisation:** La virtualisation est une technologie largement répandue mais elle ne fait l'objet d'aucun protocole de normalisation. Il n'existe par conséquent aucun outil standard aux différentes formes de virtualisation

V .3-La Plateforme collaborative:

Une plate-forme de travail collaboratif est un espace de travail virtuel. C'est un site qui centralise tous les outils liés à la conduite d'un projet et les met à disposition des acteurs.

L'objectif du travail collaboratif est de faciliter et d'optimiser la communication entre les individus dans le cadre du travail ou d'une tâche.

Les plates-formes collaboratives intègrent généralement les éléments suivants :

- Des outilsinformatiques.
- Des guides ou méthodes de travail en groupe, pour améliorer la communication, la production, lacoordination.
- Un service demessagerie.
- Un système de partage des ressources et desfichiers.
- Des outils de type forum, pages dediscussions.
- Un trombinoscope, ou annuaire des profils desutilisateurs.
- Des groupes, par projet ou parthématique.
- Un calendrier.**[1]**

VI. Les modèles de déploiement du Cloud Computing:

Le Cloud Computing est une solution qui fournit un espace dans lequel il est possible de placer virtuellement des infrastructures serveur ou réseau, des plateformes de développement ou d'exécution etc., il existe différentes typologies du Cloud :

VI .1 Le Cloud Public :

- L'infrastructure est mise à la disposition du grand public ou d'un grand groupe d'entreprises via Internet.
- Cette infrastructure est gérée par un prestataire externe.
- Les ressources peuvent être partagées entre plusieurs entités clientes.

Exemples: de clouds publics: Amazon Elastic Compute Cloud (EC2), Sun Cloud, IBM's Blue Cloud, Google AppEngine and Windows Azure Services Platform.

V I.2 Le Cloud privé (interne) :

- L'infrastructure est exploitée par l'entreprise.
- Elle peut être hébergée par l'organisation ou par un tiers.
- Dans le cas d'un fournisseur tiers, l'infrastructure est accessible via des réseaux sécurisés de type VPN.

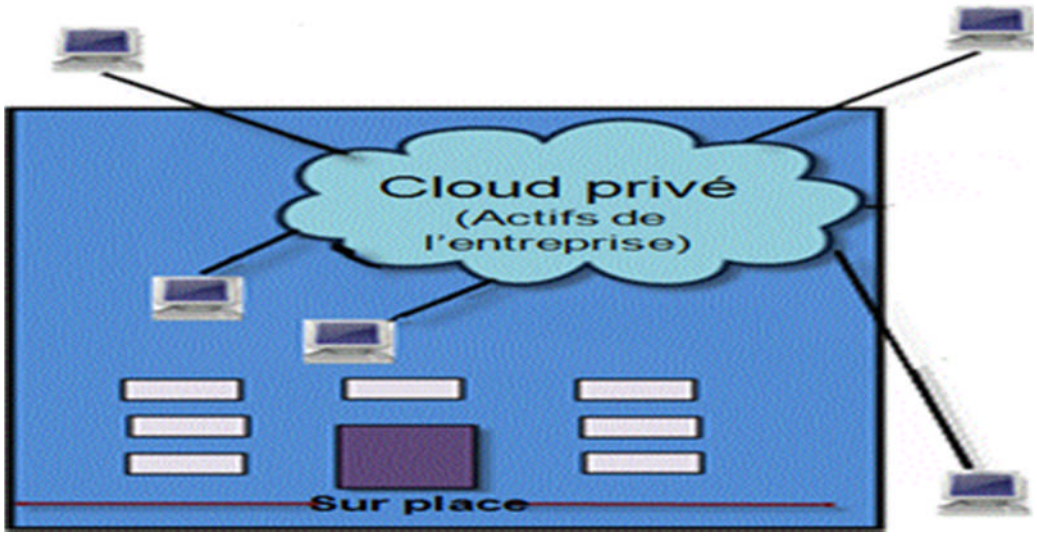


Figure 12. Le Cloud Privé

VI .3 Le Cloud Hybride :

- L'infrastructure est composée de deux ou plusieurs nuages (public ou privé).
- Ces nuages demeurent des entités indépendantes.
- Ils sont toutefois liés par l'entremise de technologies standards ou propriétaires.
- Objectif: permettre la portabilité des données et des applications entre les nuages.

Exemple: IBM avait conclu un partenariat avec Juniper Networks.

Cette association a permit à Big Blue de déployer son offre de Cloud hybride. Ainsi les entreprises qui utilisent ce service peuvent faire basculer, par un simple glisser déposer, des applications hébergées dans un nuage privé interne vers un nuage public sécurisé**.[S5]**

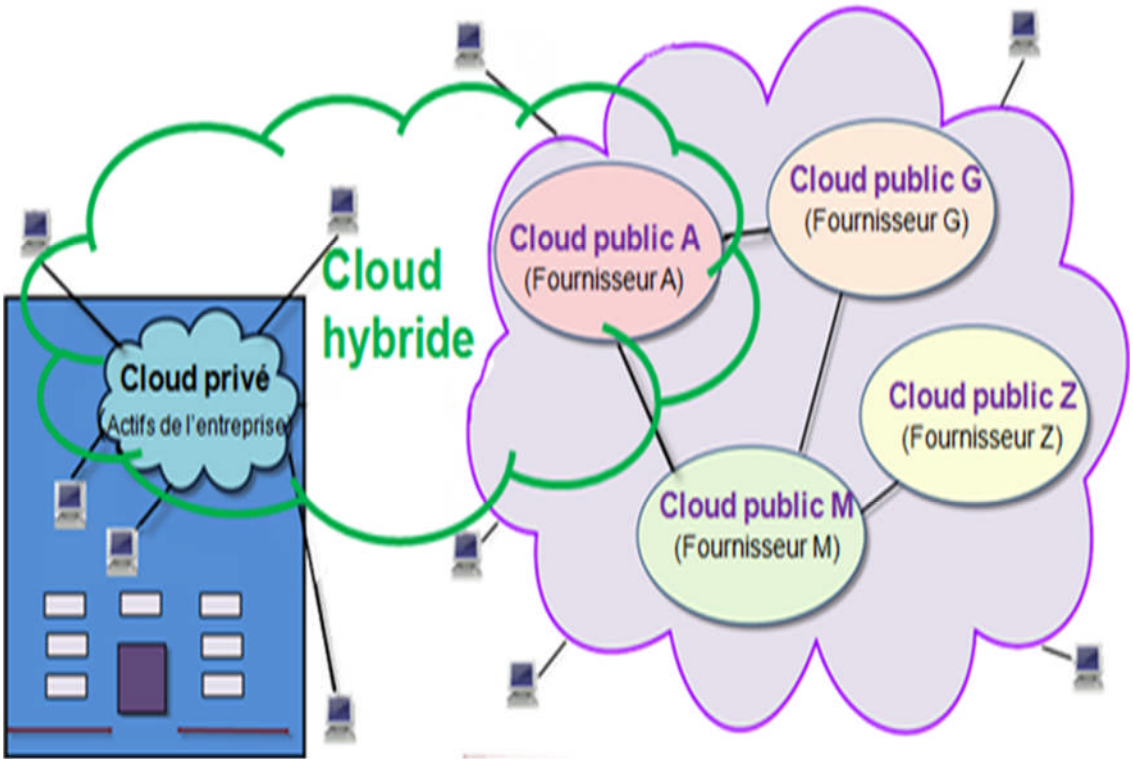


Figure 13. Le Cloud Hybride

VII. Les niveaux de services du Cloud Computing:

VII .1 SaaS (Logiciel-service/Software as a Service) :

Son concept consiste à proposer un abonnement à un logiciel plutôt que l'achat d'une licence. On oublie donc le modèle client/serveur et aucune application n'est installée sur l'ordinateur, elles sont directement utilisables via le navigateur Web.
L'utilisation reste transparente pour les utilisateurs, qui ne se soucient ni de la plateforme, ni du matériel, qui sont mutualisés avec d'autres entreprises.

Le Saas remplace l'ASP, aussi appelé fournisseur d'applications hébergées ou FAH, ou application service provider en anglais ou ASP, qui est une entreprise qui fournit des logiciels ou des services informatiques à ses clients au travers d'un réseau.

Deux principales différences avec l'ASP traditionnel sont qu'une simple interface web est utilisée côté client dans tous les cas (pas de client lourd), et que le SaaS propose une seule instance de logiciel qui évolue indépendamment des clients.

Avec l'arrivé du Haut débit, les logiciel en mode SaaS deviennent utilisables sans problèmes.

- **Avantage :**plus d'installation, plus de mise à jour (elles sont continues chez le fournisseur), plus de migration de données etc. Paiement à l'usage. Test de nouveaux logiciels avec facilité.
- **Inconvénients :** limitation par définition au logiciel proposé. Pas de contrôle sur le stockage et la sécurisation des données associées au logiciel. Réactivité des applications Web pas toujours idéale.

Les cibles sont les utilisateurs finaux. On retrouvera des entreprises comme Sales Force ou Diva (projet lancé par une équipe de l'école d'ingénieur EPITECH).

VII .2 PaaS (Platforme-service/Platform as a Service) :

Il s'agit des plateformes du nuage, regroupant principalement les serveurs mutualisés et leurs systèmes d'exploitation. En plus de pouvoir délivrer des logiciels en mode SaaS, le PaaS dispose d'environnements spécialisés au développement comprenant les langages, les outils et les modules nécessaires.

L'avantage est que ces environnements sont hébergés par un prestataire basé à l'extérieur de l'entreprise ce qui permet de ne disposer d'aucune infrastructure et de personnel de maintenance et donc de pouvoir se consacrer au développement.

- **Avantage:** le déploiement est automatisé, pas de logiciel supplémentaire à acheter ou à installer.
- **Inconvénient :** limitation à une ou deux technologies (ex. : Python ou Java pour Google AppEngine, .NET pour Microsoft Azure, propriétaire pour force.com). Pas de contrôle des machines virtuelles sousjacentes. Convient uniquement aux applications Web.

Les cibles: sont les développeurs. **Google App Engine** est le principal acteur proposant ce genre d'infrastructures.

VII .3 IaaS (Infrastructure -service/Infrastructur as a Service) :

Infrastructure as a service est caractérisée par l'utilisation d'une application partagée. Il s'agit de la mise à disposition, à la demande, de ressources d'infrastructures dont la plus grande partie est localisée à distance dans des Datacenters.

L'IaaS permet l'accès aux serveurs et à leurs configurations pour les administrateurs de l'entreprise.

Le client a la possibilité de louer des clusters, de la mémoire ou du stockage de données. Le coût est directement lié au taux d'occupation. Une analogie peut être faîte avec le mode d'utilisation des industries des commodités (électricité, eau, gaz) ou des Télécommunications.

- **Avantage :** grande flexibilité, contrôle total des systèmes (administration à distance par SSH ou Remote Desktop, RDP), qui permet d'installer tout type de logiciel métier.
- **Inconvénient :** besoin d'administrateurs système comme pour les solutions de serveurs classiques sur site.

Les cibles :sont les responsables d'infrastructures informatiques. Amazon EC2 est le principal qui propose ce genre d'infrastructures. Eucalyptus est un exemple d'infrastructure

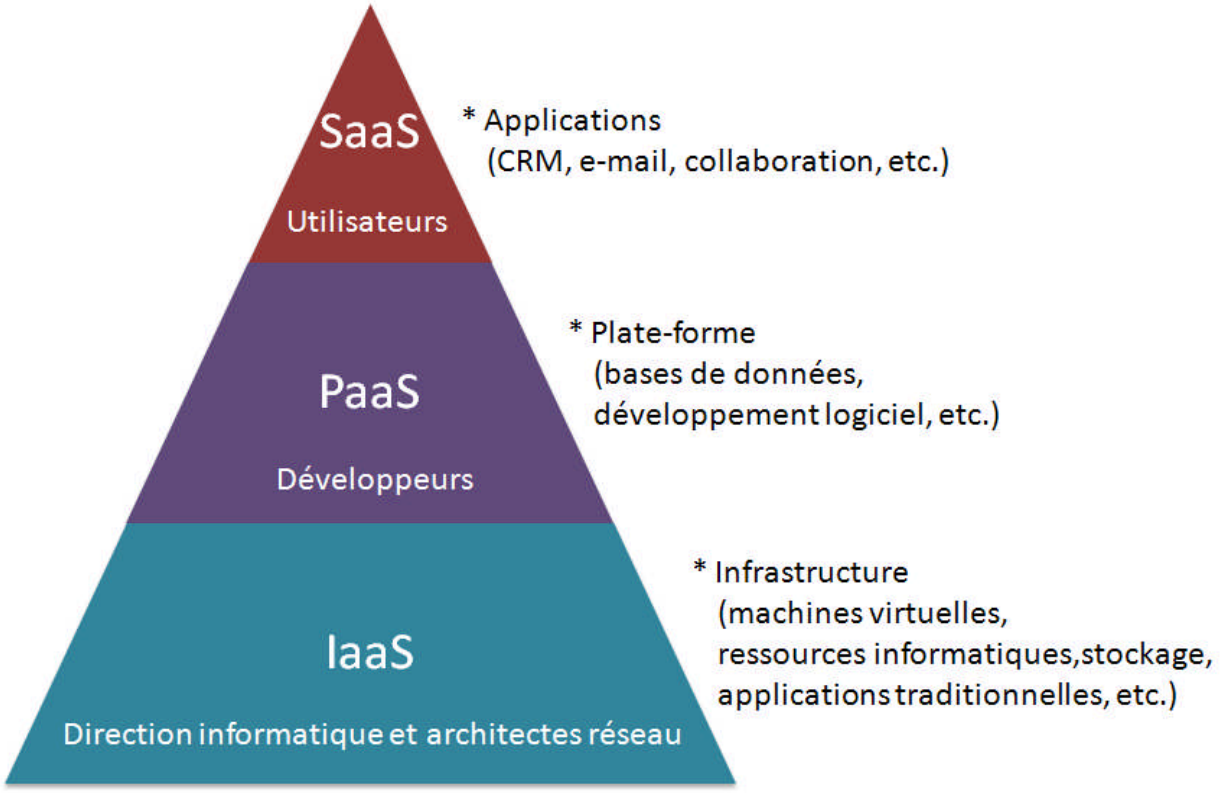


Figure 14. Les niveaux du service du Cloud Computing

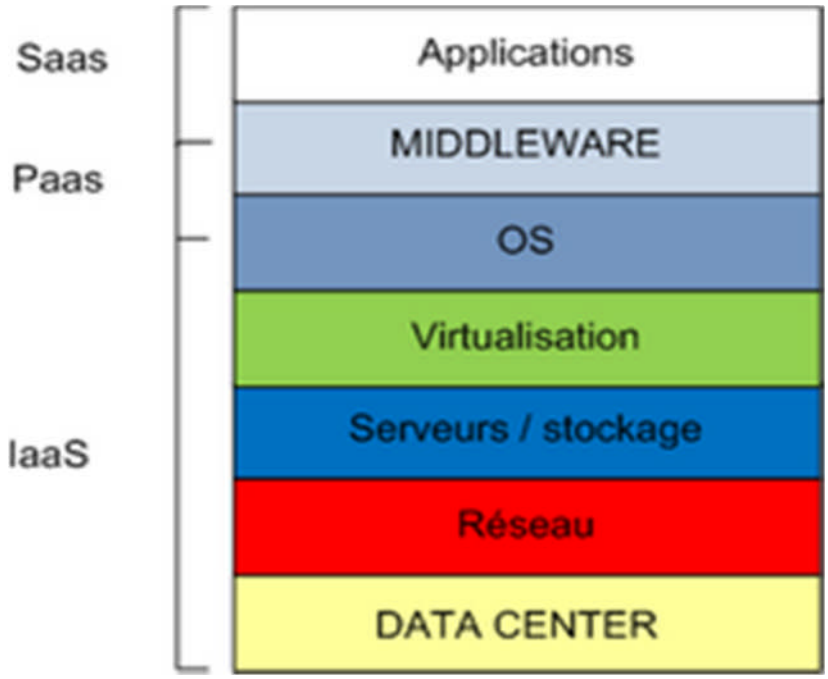


Figure 15. Le positionnement de chaque niveau de service

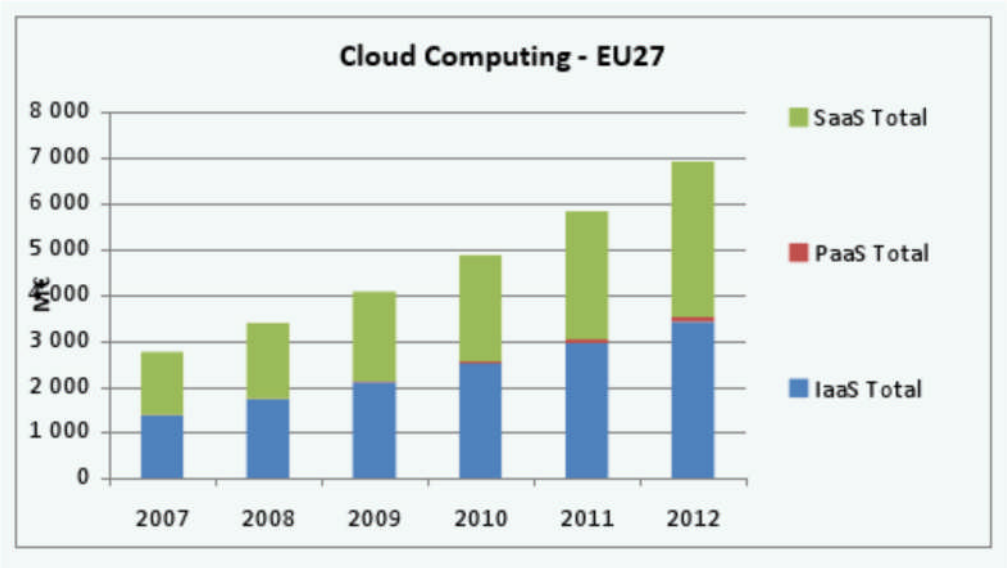


Figure 16. L'IaaS c'est plus de 50% du marché du Cloud dans le monde

D'autres services sont également disponible:

- **DaaS (Data as a Service):**

Correspond à la mise à disposition de données délocalisées quelque part sur le réseau. Ces données sont principalement consommées par ce que l'on appelle des **mushups**.

- **CaaS (Communication as a Service):**

Correspond à la fourniture de solutions de communication substituant aux matériels et serveurs locaux(PABX,ACD,SVI...)des ressources partagées sur Internet.

- **Naas(Network as a service):**

Correspond à la fourniture de services de réseaux, suivant le concept de software defined networking(SD.N).

- **STaaS(Storage as a Service):**

Correspond au stockage de fichiers chez des prestataires externes, qui les hébergent pour le compte de leurs clients. Des services grand public, tels que Microsoft OneDrive, SugarSync et Box.net, proposent ce type de stockage , le plus souvent à des fins de sauvegarde ou de partage de fichiers.

VIII. Les principaux fournisseurs du Cloud Computing:

VIII . 1 Amazon Web Services (IaaS) :

- **EC2 :** **E**lastic **C**ompute **C**loud, ou comment démarrer/arrêter un nombre quelconque de machines virtuelles Linux ou Windows, en quelques minutes, par Web Services, à Dublin, en Virginie, en Californie, à Singapour et à Tokyo, pour consommer de la ressource Informatique à la demande, en payant à l'heure consommée, sans investissementminimal
- **EBS :** **E**lastic **B**lock **S**torage, ou comment créer et attacher à ces VMs des volumes de taille quelconque (quelques GB à plusieursTB).
- **CloudWatch :** monitoring des instances EC2 (% CPU, % RAM utilisésetc.)
- **SimpleDB :** Base de données dans le Cloud, scalablé et sansschéma.
- **CloudFront :** CDN (Content Delivery Network), réplication géographique de données à proximité des clients Web pour une expérience utilisateur plusréactive.
- **SQS: S**imple **Q**ueue **S**ervice, messagerie applicative par queue, pour une communication fiable entre applications, avec récupérationd'erreur.
- **RDS : R**elational **D**atabase **S**ervice, base de données MySQL ou Oracle scalable dans le Cloud (avecschéma).

VIII .2 Amazon Web Services (PaaS) :

- **S3 : S**imple **S**torage **S**ervice, stockage scalable et accessible directement par REST (pas

besoind'EC2).

VIII . 3 Microsoft Azure (PaaS) :

- **Azure Compute :** hébergement scalable d'applications ASP.NET ou batches .NET / natifs.
- **Azure Storage :** stockage scalable de blobs dans le Cloud, accessible directement par REST.
- **SQL Azure:** base de données relationnelle (SQL Server) scalable dans leCloud.
- **Service Bus:** ESB dans le Cloud.
- **Access Control:** Fédération d'identité pour SSO dans le Cloud.

VIII . 4 Google App Engine(PaaS) :

- **Cloud Hosting:** Hébergement scalable d'application python ou java , avec stockage et base de données plate BigTable.
- **Forece.Com (PaaS):** Hébergementd'application orientées CRM
- **SalseForce.Com (SaaS):** Consommation à la demande de logiciels Web CRM.

VIII . 5 Google Apps (SaaS) : Version professionnelle de Gmail et Google Apps **.**

IX. Les avantages et les inconvénients du Cloud computing:

IX .1 Les avantages :

- **Un usage simplifié**

Le Cloud Computing simplifie les usages en permettant de s'affranchir des contraintes de l'outil informatique traditionnel (installation et mise à jour des logiciels, espace de stockage, portabilité des données…). il offre aussi plus d'élasticité et d'agilité car il permet d'accéder plus rapidement à des ressources IT (serveur, stockage ou bande passante) via un simple portail web et donc sans investir dans des équipements matériels supplémentaires. La mise à disposition est donc immédiate.

De plus, l'utilisateur n'a pas d'infrastructure à gérer, c'est au fournisseur Cloud de maintenir le matériel serveur, le stockage, les réseaux. L'entreprise peut donc se concentrer avant tout sur son métier, son activité et son savoir-faire.

- **Une réduction des coûts**

Pourquoi le Cloud Computing est-il économiquement attrayant ? Il permet de démarrer une activité professionnelle sans avoir à investir dans d'une infrastructure IT très coûteuse en

interne. Et pour les entreprises qui disposent déjà de leur propre infrastructure mais qui ont des besoins IT additionnels comme pour gérer des pics d'activités, le Cloud Computing est le moyen le plus économique d'y répondre. En effet, l'entreprise ajuste, , son infrastructure à ses besoins et en augmentant ou diminuant les ressources disponibles. En souscrivant à des offres de Cloud Computing, l'utilisateur paie uniquement ce qu'il consomme. Enfin, l'utilisateur n'a pas non plus à supporter les coûts liés à la maintenance et le renouvellement des équipements. Avec le Cloud Computing, l'entreprise utilisatrice réduit considérablement ses investissements IT (coûts liés au Capex) et optimise ses coûts de fonctionnement et d'exploitation (Opex).

- **Une haute disponibilité du service**

Contrairement aux idées reçues, le Cloud Computing permet de garantir les accès et la disponibilité des services, ce qui est important aujourd'hui car les entreprises emploient des salariés de plus en plus nomades. Les collaborateurs doivent donc pouvoir accéder à toutes leurs applications et à leurs données sans interruption de service. Les contrats Cloud Computing sont donc très importants pour les clients car ils permettent d'avoir des réponses précises sur les garanties offertes par les fournisseurs sur les SLA (niveaux de continuité de service). La disponibilité de service offerte par un fournisseur de service Cloud doit se situer entre 98 et 99,99% incluant les temps d'arrêt des serveurs pour maintenance ou pour des interruptions inattendues.

- **La sécurité : les garanties du fournisseur**

Le Cloud Computing apporte plus de sécurité. Et pour cause, faute de temps, de compétences et de budget, les entreprises sont de moins en moins capables d'assurer pleinement la sécurité de leur propre système d'information. En revanche, le Cloud Computing garantit cette sécurité en possédant de biens meilleurs dispositifs et services de sécurité (réplication des données, plan de reprise d'activité, cyberdéfense, etc) avec des mises à jour et des audits réguliers.. Le lieu de stockage des données est une question récurrente des entreprises utilisatrices, elles sont très méfiantes à l'égard du Patriot Act. En effet, rappelons que la loi américaine autorise aux organismes gouvernementaux d'avoir un accès quasi illimité aux informations appartenant aux entreprises. Enfin les contrats prévoyant des clauses de réversibilité avec restitution de l'intégralité des données pour les entreprises sont aujourd'hui prises en compte par les fournisseurs.

IX .2 Les inconvénients :

- L'utilisation des réseaux publics, dans le cas du Cloud public, entraîne des risques liés à la sécurité du Cloud. En effet, la connexion entre les postes et les serveurs applicatifs passe par le réseau internet, et expose à des risques supplémentaires de cyber attaques, et de violation de confidentialité. Le risque existe pour les particuliers, mais aussi pour les grandes et moyennes entreprises, qui ont depuis longtemps protégé leurs serveurs et leurs applications des attaques venues de l'extérieur grâce à des réseaux internes cloisonnés.

- Les entreprises perdent la maîtrise de l'implantation de leurs données. De ce fait, les interfaces inter-applicatives (qui peuvent être volumineuses) deviennent beaucoup plus complexes à mettre en œuvre que sur une architecture hébergée en interne.
- Tout comme les logiciels installés localement, les services de *cloud computing* sont utilisables pour lancer des attaques (craquage de mots de passe, déni de service).En 2009, par exemple, un cheval de Troie a utilisé illégalement un service du *cloud* public d'Amazon pour infecter des ordinateurs.

- Des questions juridiques peuvent se poser, notamment par l'absence de localisation précise des données du cloud computing. Les lois en vigueur s'appliquent, mais pour quel serveur, quel data center, et surtout quel pays ? **[S6]**

X. Le Cloud Computing et la sécurité :

La sécurité et la conformité émergent systématiquement, comme les principales préoccupations des responsables informatiques lorsqu'il est question de Cloud Computing, des préoccupations encore plus accentuées lorsqu'il s'agit de Cloud public .

La sécurité permet de garantir la confidentialité, l'intégrité, l'authenticité et la disponibilité des informations.

Certaines questions légitimes reviennent sans cesse :

- Mes données sont-elles sures dans le Cloud?
- Où sont stockées mes données?
- Qui va avoir accès à mes données?s
- Aurais-je accès à mes données à n'importe quel moment?
- Que deviendrons mes données s'il y a interruption du service?

La mise sur pied d'une solution de Cloud Computing comporte des problèmes de sécurité

inhérents à la solution elle-même. Le fait de centraliser toutes les informations sur un site pose un grand nombre de problèmes. On peut citer comme problème potentiel :

- Une possible interruption massive duservice.
- Une cible de choix pour leshackers

➢ Interface et API nonsécurisé

➢ Ce point de vulnérabilité du Cloud Computing fait l'objet depuis quelques années l'objet de recherches avancées. Il a été créé un organisme chargé de mettre sur pied des normes en matière de sécurité dans le Cloud Computing. Cet organisme s'appelle CSA (Cloud Security Alliance). Du travail de cet organisme, il en est ressorti certaines techniques utilisées de nos jours pour améliorer la sécurité du Cloud Computing. Parmi ces techniques on peut citer :

- La multi-location : cette technique permet de créer des instances d'une même donnée sur plusieurs sites différents. Elle permet une récupération facile en cas dedésastre.
- Le chiffrement : le chiffrement de l'accès à l'interface de contrôle, le chiffrement des données dans leCloud.
- L'isolation des machines virtuelles.

La sécurité absolue n'existe pas, donc le problème de sécurité reste le plus souvent un problème de confiance entre le fournisseur de service et le consommateur de service. Cette confiance se traduit par la signature d'un contrat nommé SLA (Service Level Agreement). Ce contrat précise les taux de disponibilité du service. En règle générale, et pour la plupart des fournisseurs, ce taux est supérieur à 99 %**.[S7]**

XI. Conclusion:

le Cloud revient à simplifier la vie de l'utilisateur (ou de l'administrateur des machines) en lui évitant l'installation locale répétée des logiciels professionnels ou de la configuration de chaque poste individuel.

Les utilisateurs ou les entreprises ne sont ainsi plus directement gérants de leurs serveurs informatiques mais accèdent de manière évolutive à de nombreux services en ligne sans avoir à appréhender l infrastructure sous-jacente, souvent complexe et identifiée de manière métaphorique à un nuage. Ce nuage cache la réalité physique des serveurs distants, qui disposent d'une puissance de calcul, d'une capacité de stockage.

Chapitre II:

Présentation d'Openstack

. Introduction :

Le Cloud Computing représente un nouveau défi dans le monde informatique. Plusieurs solutions sont proposées : des solutions propriétaires et des solutions open sources. Dans ce chapitre, nous allons présenter une des solutions Cloud existante qui s'agit d'openstack, de manière non exhaustive, sa définition, son architecture et ses composants .

II. Historique :

Il existe de différentes solutions implémentant le cloud computing. OPENSTACK représente aujourd'hui la solution open source vedette .

OpenStack est à l'origine le regroupement de deux projets. Un premier projet nommé Nebula, initié en 2008 et développé par la **NASA** afin de répondre à leurs besoins en termes de calculs sur les quantités considérables de données qu'ils récupèrent. Nebula est donc l'ancêtre de Nova et traite de l'aspect compute. Au même moment, **Rackspace** travaille dans un même esprit (APIs

génériques, …) sur leur système de stockage orienté objet nommé **Swift** (commercialisé par Rackspace sous le nom Cloud Files) . Le deuxième projet, que Rackspace a rendu open source. Etant donné l'alignement technique des deux projets, La NASA et Rackspace ont associé leurs projets dans un projet open source commun nommé OpenStack.

Ces deux sociétés ont ensuite été rejointes par Cloud.com, Citrix Systems, Dell, Cloudkick, Intel, et Cisco pour le développement d'OpenStack. Cette plate-forme est intégrée dans les distributions UbuntuServer.

La dernière version en date du produit est Grizzly, successeur de Folsom (ayant succédé lui-même à Essex). La prochaine version attendue est Havana.)

III. Définition :

OpenStack est un ensemble de logiciels libre (open source) permettant de déployer des infrastructures de Cloud Computing (infrastructure en tant que service) développées en Python et

permettant de déployer un Cloud de type **IaaS** avec des fonctionnalités similaires aux offres bien connues du marché comme celle de Amazon .

Il s'installe sur un système d'exploitation libre comme Ubuntu ou Debian et se configure entièrement en ligne de commande. C'est un système robuste et qui a fait ses preuves auprès des professionnels du domaine

Il possède une architecture modulaire composée de plusieurs projets adressant des fonctionnalités complémentaires permettantla construction de Cloud privé et public de type IaaS et de contrôler de grands bassins de calcul, de stockage, et les ressources de réseautage à travers

un centre de données , tout géré par un tableau de bord qui donne aux administrateurs de contrôler tout en permettant à leurs utilisateurs à des ressources d'approvisionnement par le biais d'une interface web .

OpenStack joue le rôle d'une couche de management de Cloud qui assure la communication entre la couche physique ou se trouve des serveurs physiques occupés par des hyperviseurs différents (Vmware ESX, Citrix Xen, KVM, qemu...) et la couche applicative (Applications, utilisateurs, administrateurs…)

Le projet est porté par la Fondation OpenStack, une organisation non-commerciale qui a pour but de promouvoir le projet OpenStack ainsi que de protéger et d'aider les développeurs et toute la communauté OpenStack. **[S8]**

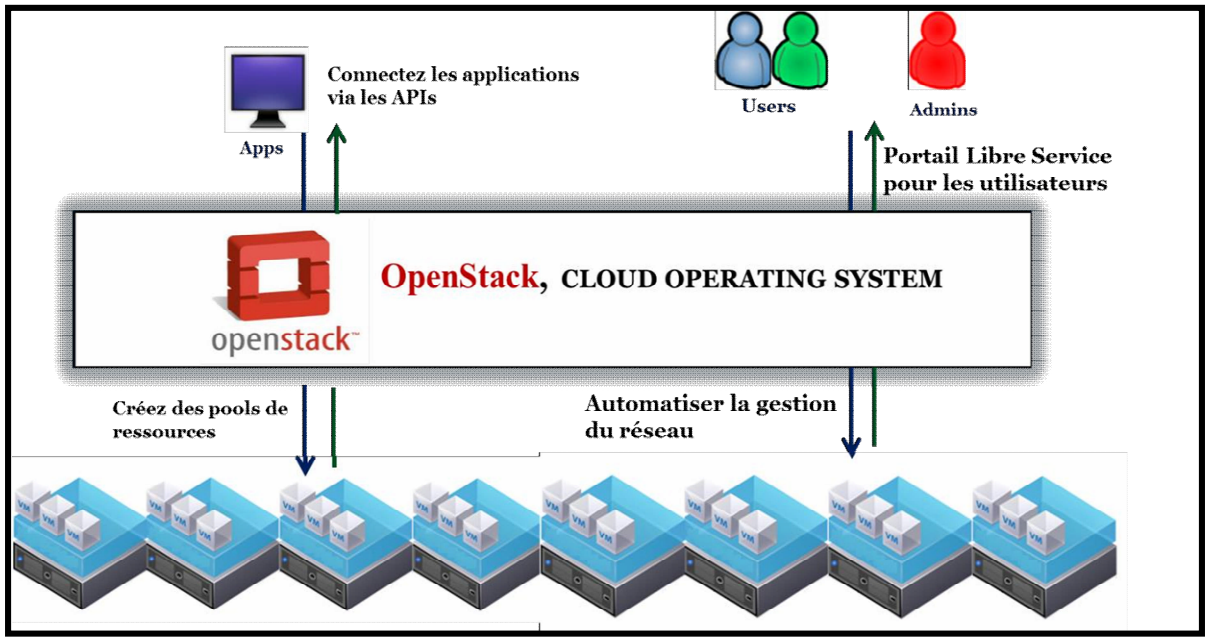


Figure 17. Le rôle d'openstack

OpenStack a libéré plusieurs versions comme indiqué dans le tableau ci- dessous:

Nom	Date de sortie	Nom de composant inclus
Austin	21 Octobre 2010	Nova, Swift
Bexar	3 Fevrier 2011	Nova, Swift, glance
Cactus	15 Avril 2011	Nova, Swift, glance
Diablo	22 Septembre 2011	Nova, Swift, glance
Essex	5 Avril 2012	Nova, Swift, glance, horizon, Keystone
Folsom	27 Septembre 2012	Nova,Swift, glance, horizon, Keystone,quantum, cinder
Grizzly	4 Avril 2013	Nova, Swift, glance, horizon, Keystone,quantum, cinder Keystone, quantum cinder

Tableau 1. Les versions d'openstack

IV. Les différents composants d'openstack :

OpenStack possède une architecture qui comprend de nombreux composants qui s'agit d'une série de logiciels et de projets au code source libre:

IV.1 Nova (calcul):

Développé à l'origine par la NASA, c'est le cœur d'OpenStack. Il s'occupe principalement de la gestion des hyperviseurs (ordonnanceur et gestion des machines virtuelles) par l'intermédiaire de la libvirt ou directement par des API . Aujourd'hui l'hyperviseur le mieux supporté reste KVM, mais nova fonctionne aussi avec Xen, ESX, et Hyper-V voire avec des gestionnaires de conteneur comme Docker. Il contrôle aussi les ressources (CPU, RAM, réseaux et stockages)

L'architecture de la brique de Nova est conçue pour évoluer horizontalement en rajoutant du matériel. D'ailleurs Nova fonctionne avec du matériel non spécialisé ce qui permet de réutiliser des serveurs existants par exemple.

Letableauci-dessous va nous permettre de comprendrel'architectureNova Compute et les rôles de chaque composant :

Le composant Nova	**Le rôle**
API	✓ Cœur deNova ✓ Fonction Principale : Cloud Controller avec le servicenova-api. ✓ Compatible avec l'API AmazonEC2 ✓ Ecoute sur le port 8773 pour EC2 API et 8774 pour OpenStack API ✓ Initialise la plupart desactivités
Scheduler	✓ Renforce certaines fonctionnalités (ex :quotas) ✓ Principe simple : il prend une demande d'instance de machine virtuelle et détermine où (quel « Compute server ») doit-elle êtreexécutée. ✓ Fonctionnement par algorithmes pour assurer un fonctionnementoptimal. ✓ 3 choix d'ordonnancement: - Simple : tente de trouver l'hôte le moins « chargé» - Chance (celui par défaut) : choisit un hôte disponible au hasard depuis sa « Service Table» - Zone : Prend un hôte au hasard depuis une zone « disponible»
Nova Compute	✓ Créé et termine les instances de machinesvirtuelles ✓ Reçoit et exécute des actions visant à mettre à jour les états des VM dans la base dedonnées ✓ Supporte plusieurs API : KVM, Xen, Citrix, VMware,Hyper-V, …

Nova Volume	✓ Gère la création, l'attachement et le détachement de volumes persistants. ✓ Compatible avec AoE, iSCSI (dont Solaris ZFS), Sheepdog, RBD, LeftHand(HP).

Nova Network	✓ Configure les interfacesbridge ✓ Adapte les règles de pare-feu(Iptables) ✓ 2 types d'adresse IP pour une instance: - Adresse fixe :privée - Adresse provisoire :publique ✓ 3 gestionnaires de réseaux: - Flat : adresse fixe attachée à l'interfacebridge - Flat DHCP : adressage dynamique pour chaque interface bridge - Support des VLAN : chaque projet dispose de sa plage d'adresses IP accessibles viaVLAN.
Queue	✓ Point de passage obligé pour les instructions échangées entre les services ✓ Différents types de files d'attente de messages pour faciliter la communication : Topics, Fanout,Host…
Data Base	✓ Enregistre la configuration et les états en temps réels pour une infrastructure Cloud : types d'instances disponibles, instances en cours d'utilisation, réseaux disponibles,projets, ✓ Supporte la plupart des SGBD : MySQL,PostgreSQL

Tableau 2: Les composants de Nova

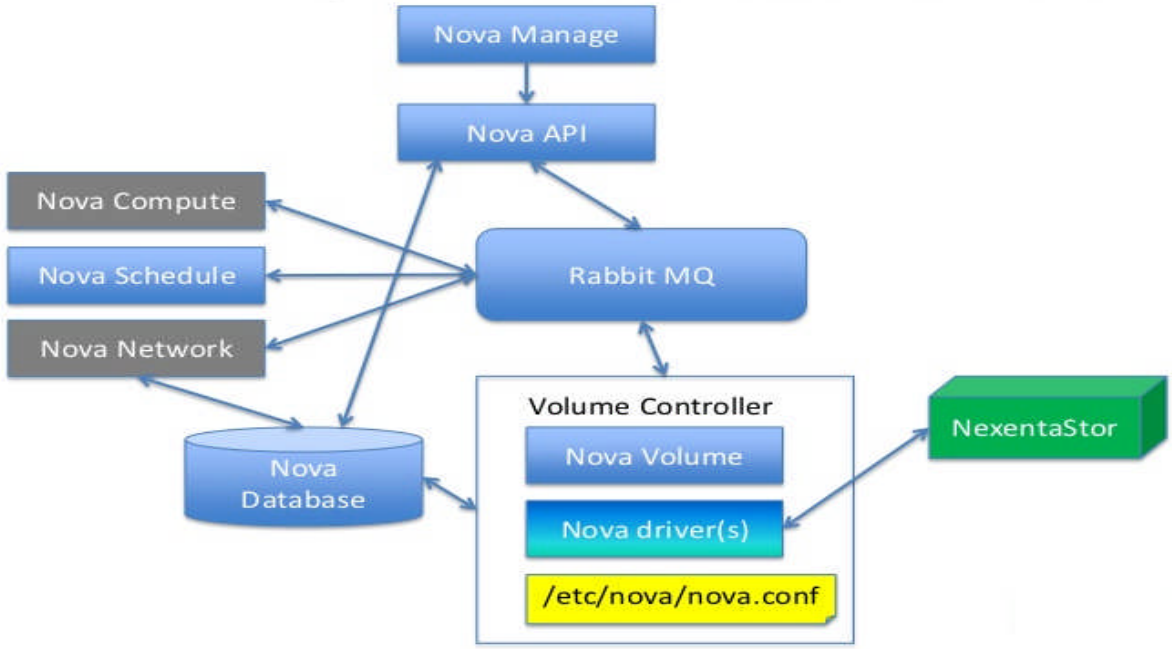


Figure 18. Architecture de Nova

IV.2 Swift (project storage):

Le stockage objet d'OpenStack s'appelle Swift. C'est un système de stockage de données redondant et évolutif. Les fichiers sont écrits sur de multiples disques durs répartis sur plusieurs serveurs dans un Datacenter. Il s'assure de la réplication et de l'intégrité des données au sein du cluster. Le Cluster Swift évolue horizontalement en rajoutant simplement de nouveaux serveurs. Si un serveur ou un disque dur tombe en panne, Swift réplique son contenu depuis des nœuds actifs du cluster dans des emplacements nouveaux. Puisque toute la logique de Swift est applicative, elle permet l'utilisation de matériel peu couteux et non spécialisé.

En aout 2009, c'est Rackspace qui a commencé le développement de Swift, en remplacement de leur ancien produit nommé Cloud Files. Aujourd'hui c'est la société SwiftStack qui mène le développement de Swift avec la communauté.

L'Architecture de Swift est composée par les éléments suivants:

- **ProxyServer**
  - ✓ Partie visible (public) deSwift,
  - ✓ Détermine le nœud de stockageapproprié,
  - ✓ Coordonne lesréponses.
- **L'anneau (Ring)**
  - ✓ Lie les requêtes au nœud destockage,
  - ✓ Gère les zones dedisponibilité,
  - ✓ Extensible sans affecter les autresentités.
- **Serveurs de stockage (Comptes etContainers):**
  - ✓ Base de donnéesSQLite,
  - ✓ Les groupes de containers et les objets sont contenus dans lescomptes,
  - ✓ Schéma simple : table pour les listes et table pour lesmétadatas.
- **Serveurs de stockage (Objet):**
  - ✓ stocker lesfichiers,
  - ✓ Fichiers nommés avec un marqueur detemps.
- **Serveurs deconsistance**
  - ✓ Réplicats,
  - ✓ Gestion des mises àjour,
  - ✓ Gestion desaudits.

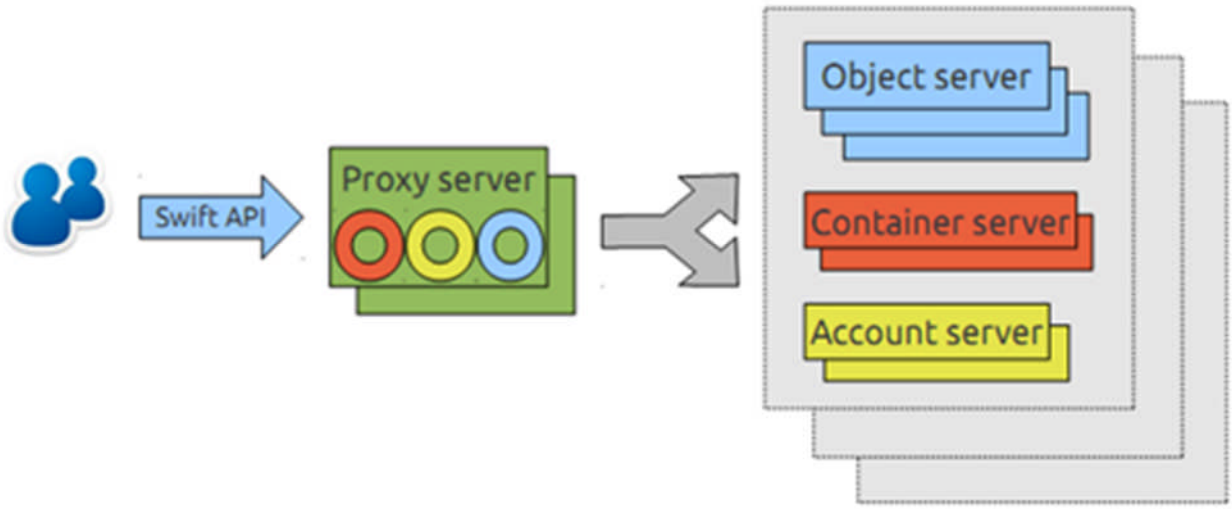


Figure 19. Architecture de Swift

IV.3 Glance (service d'image):

Glance a été extrait rapidement de Nova pour en faire un composant à entière.

Imaging Service fournit les services de stockages, de découvertes, d'enregistrements et de distributions pour les images disques de machines virtuelles. Il fournit également une API compatible REST permettant d'effectuer des requêtes pour obtenir des informations sur les images hébergées par les différents magasins de stockages .

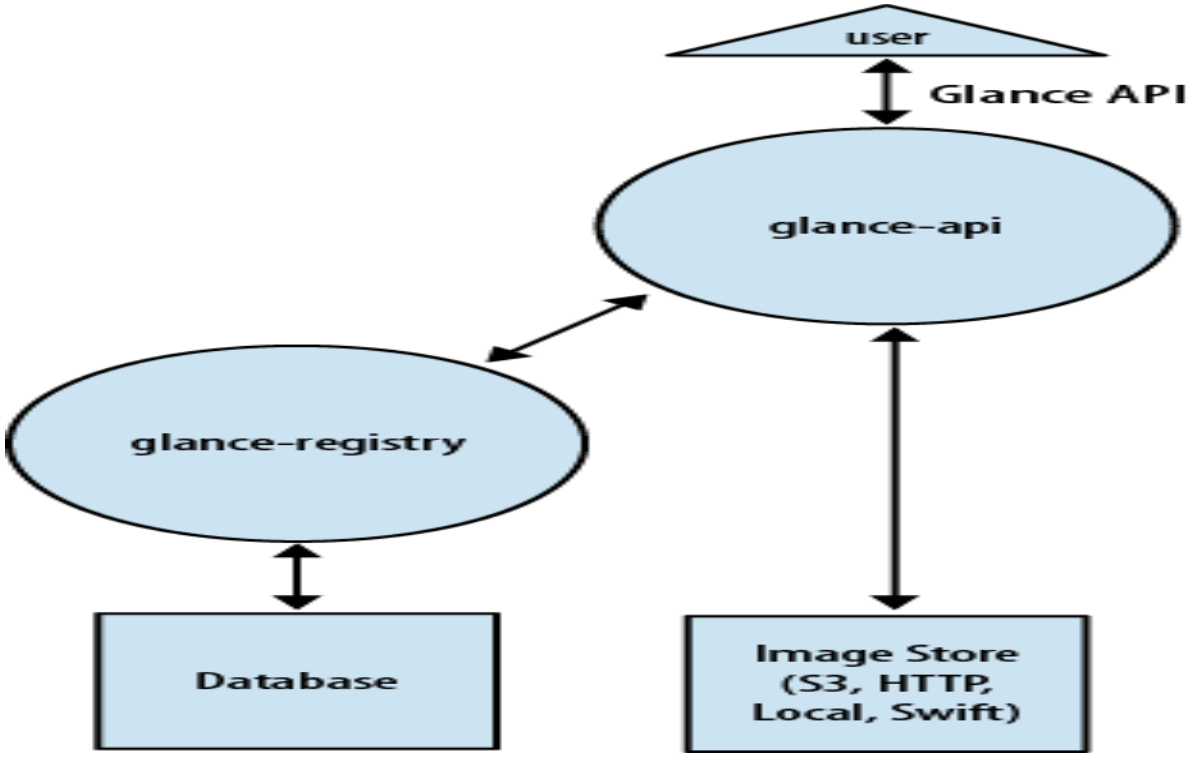


Figure20. Architecture de Glance

IV.4 Horizon :

Horizon est la mise en œuvre canonique du tableau de bord de Openstack, qui fournit une interface utilisateur basée sur le Web pour Openstack services, y compris Nova , Swift , Keystone, etc.

IV.5 Quantum :

permet d'offrir une gestion des réseaux à la demande à l'intérieur de son Cloud. Le service permet aux utilisateurs de créer des réseaux à la demande et d'y attacher des machines virtuelles. Quantum a une architecture ouverte grâce à des plugins permettant de supporter différents fournisseurs de réseau ou des technologies réseaux différentes.

IV.6 Cinder :

permet d'offrir des disques persistants pour les machines virtuelles. Ce service était inclus dans Nova à l'origine (sous le nom nova-volume) dans les versions précédente deOpenStack.[]

V. Conclusion :

Dans ce chapitre on a présenté Openstack ,son architecture d'installation ainsi que ses différents composants en présentant leurs architectures.

Dans le dernier chapitre on va présenter le travail effectué lors de ce mémoire dédié à la société TycheCloud.

Chapitre III:

Mise en place d'Openstack

I. Introduction :

Dans ce chapitre, nous allons mettre en place notre plate-forme OpenStack en présentant son architecture d'installation, les différents cas d'utilisation du système avec des diagrammes et les outils logiciels et matériels ainsi que toutes les étapes et la démarche à suivre pour installer les différents composants d'OpenStack.

II. La mise en place de Openstack :

II. 1 Architecture d'installation :

Selon la documentation Openstack il y a plusieurs architectures possibles. La figuresuivante montre ces derniers.

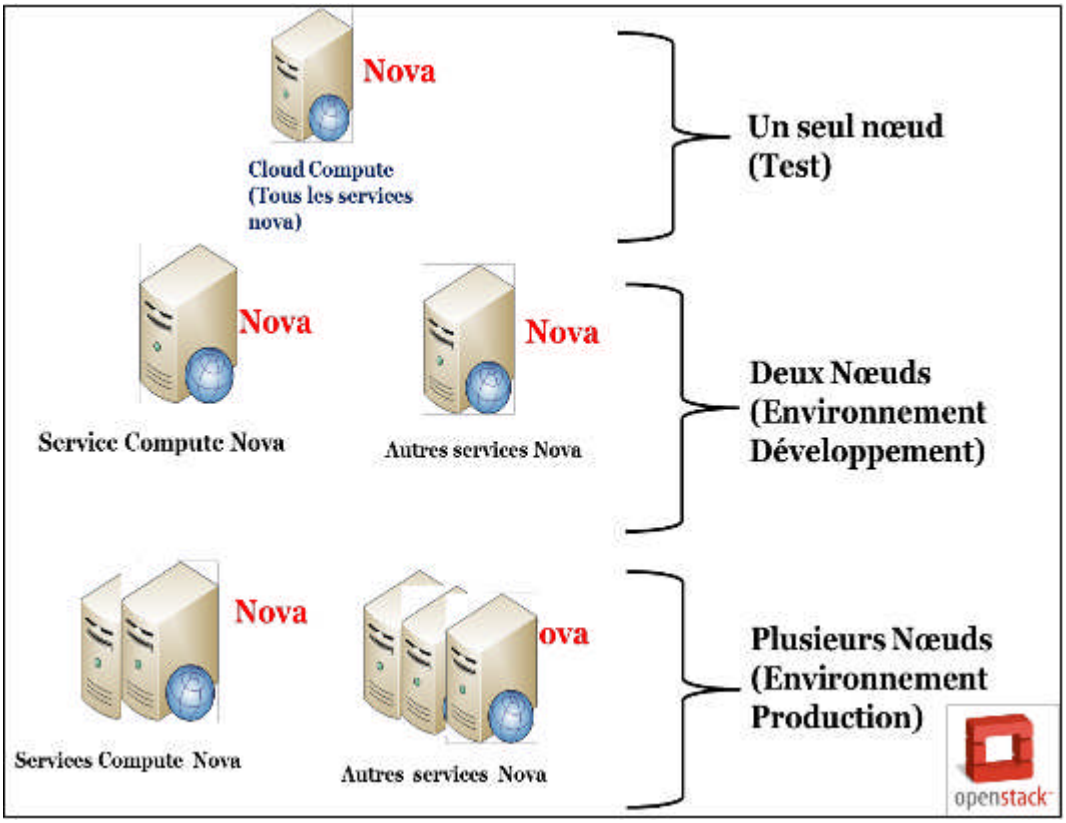


Figure 21. Les différentes architectures d'installations possibles

Les configurations mononoeuds sont utiles pour tester OpenStack en tant que produit et pour se familiariser avec ses fonctionnalités ce qui est notre cas. En revanche, une configuration à un seul noeud ne convient pas dans un environnement de production. Pour cet environnement, l'installation et la configuration d'OpenStack doit être sur plusieurs systèmes ou noeuds.

C'est pour cela que nous avons installer Openstack sur un seul nœud dans le quelle tous lesservices ainsi que toutes les instances sont hébergés au sein du même serveur.

II.2 Utilisateurs du système :

Afin de représenter les differentes fonctionnalités et utilisations possible du système, nous allons présenter une série de diagramme, en l'occurence un diagramme de cas d'utilisation et des diagrammes de sequence.

Il existe deux types d'utilisateurs ,l'administarteur du système et les utilisateurs.

II.2.1. L'administrateur :

L'administrateur est toute personne physique ayant reçu les droits d'administration. Généralement, lors de l'installation, on configure les droits du premier administrateur.

Un administrateur peut :

- Ajouter de nouveaux administrateurs.
- Supprimer des administrateurs.
- Ajouter de nouveaux utilisateurs.
- Créer un projet.
- Créer de nouvelles machines virtuelles.
- Gérer et créer un réseau.

Chaque utilisateur possède un login et un mot de passe unique, modifiable à volonté par le concerné.

II.2.2. L'utilisateur :

L'utilisateur est toute personne physique de l'entreprise ayant reçu un compte d'accès. A ce titre, il peut :

- Stocker des données dans la limite de ses possibilités.
- Instancier des machines virtuelles.

II.2.3. diagrammes

- **Diagramme de cas d'utilisation**

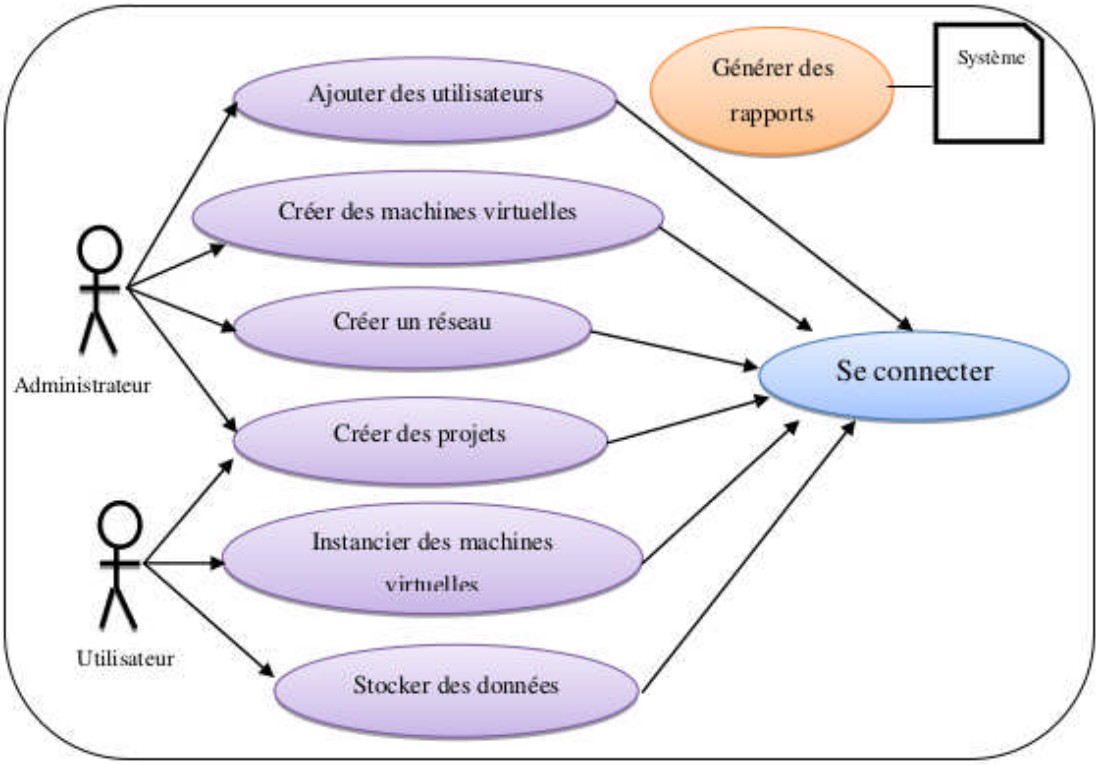


Figure 22. Diagramme de cas d'utilisation

➢ **Diagrammes de séquence**

- Diagramme de séquence du cas d'utilisation « Connexion »

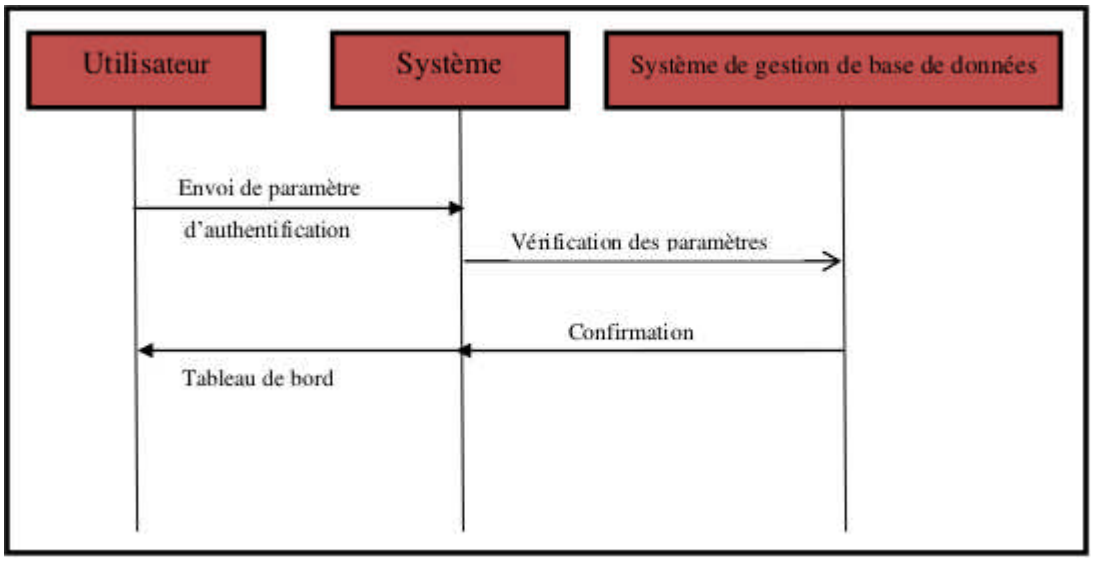


Figure 23: Diagramme de séquence"connexion"

- Diagramme de séquence du cas d'utilisation « Création d'une machine virtuelle »

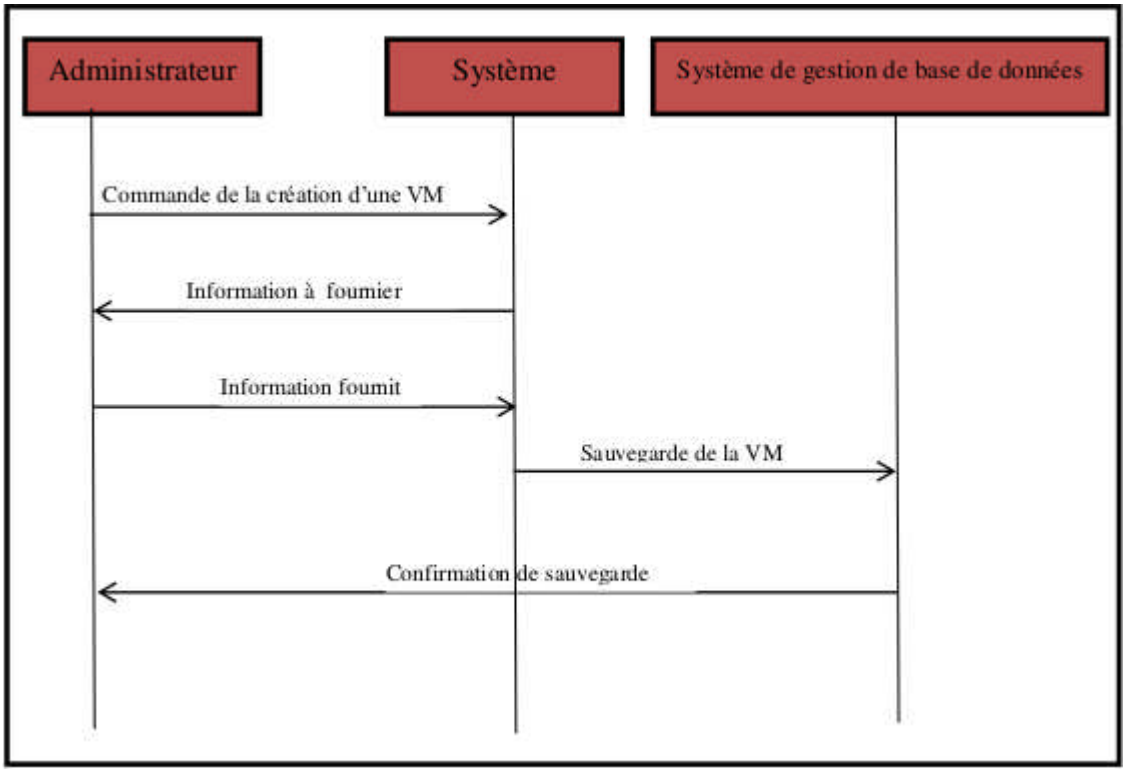


Figure 24. Diagramme de séquence "Création d'une machine virtuelle".

- Diagramme de séquence du cas d'utilisation « Stocker des données »

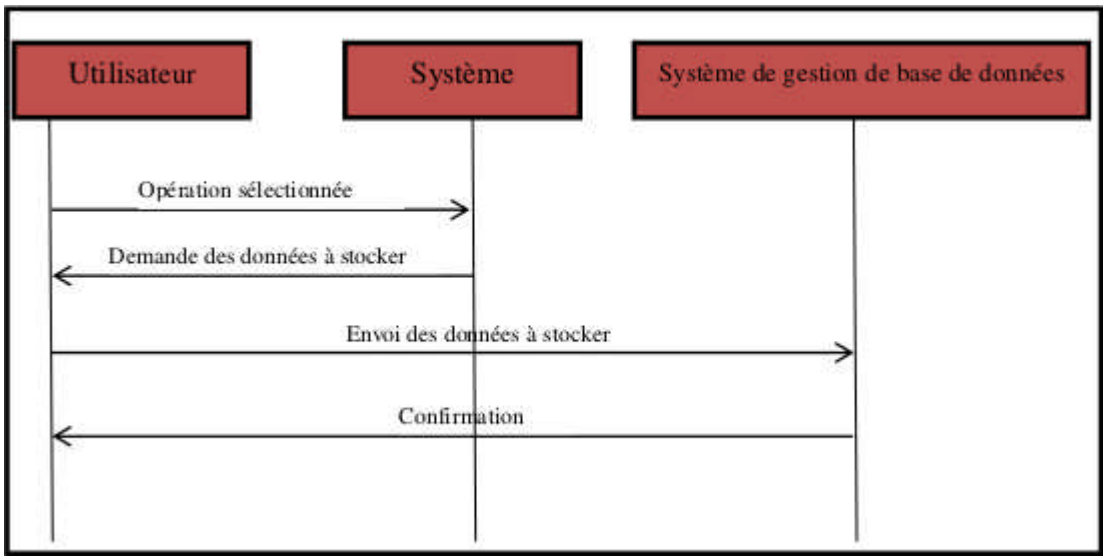


Figure 25. Diagramme de séquence "Stocker des données".

III. L'installation d 'Openstack :

L'installation de Openstack a été conduite sous Ubuntu LTS server 16.04 64 bits.

Nous avons fait l'installation de ce système d'exploitation sur notre machine physique pour des problèmes de RAM et de bugs après avoir essayer son installation virtuellement sur VirtualBox.

Il existe plusieurs méthodes pour installer Openstack, dont :

- DevStack.
- installation via des scripts.
- Depuis les packages.

Selon les besoin de l'organisme TycheCloud nous avons utiliser Devstack pour déployer OpenStack sur notre machine.

III.1. Devstack :

Devstack est un projet d'installation de Openstack à base de scripts bash maintenu par une communauté de développeurs. Il permet de construire des environnements complets destinés au développement, aux tests. Il n'est absolument pas destiné à une mise en production. Il permet d'installer Openstack sur en All-in-One (sur un seul nœud) ou en Multi-Nodes (sur plusiers noeuds).

Selon la version de Devstack qu'on a installer , les composants de Openstack qui ce sont installer sont :

- Nova nommé compute, c'est un peu le système central de OpenStack. Ce composant gère notamment le ou les hyperviseurs de virtualisation : KVM (par défaut), Xen, QEMU, ESX...
- Cinder pour le stockage dit type bloc, en clair où seront les disques des machines virtuelles (File System classiques, SAN, NAS, iSCSI,...)
- Keystone pour la gestion des autorisations et des droits.
- Glance fournit les services de stockages, de découvertes, d'enregistrements et de distributions pour les images disques de machines virtuelles.

A ceci se rajoute une interface d'administration en mode Web nommée Horizon. Il existe d'autres composants, mais ça ne concerne pas notre travaille.

III.2. Création du serveur Ubuntu

- Nous avons créer un serveur avec les paramètres suivants :
- Nom : Ubuntu 16.04
- Type de système d'exploitation : Linux.
- Version : Ubuntu 16.04 LTS Server 64bits.
- Mémoire (RAM) : 3.8 Go.
- Disque : 245.9 Go.

III.3. Étapes d'installation de Openstack

Les étapes de l'installation sont comme suit sur le terminal de Ubuntu :

1. Crée un utilisateur stack :

sudo useradd -g stack -s /bin/bash -d /opt/stack -m stack
puis lui ajouter un mot de passe pour plus de securité avec la commande
sudo passwd stack

2. Ajouter les droits d'accés pour l'utilisateur stack :

vi /etc/sudoers et en ajoute la ligne (stack ALL=(ALL:ALL) NOPASSWD: ALL)

3. Installer le Git :

sudo apt-get install git –y

4. Télécharger Devstack :

git clone git://github.com/openstack-dev/devstack.git

5. Ajouter les droits d'accès aux deux répertoires stack et devstack :

sudo chown -R stack:stack /opt/stack
sudo chown -R stack:stack /home/nom du répertoire/devstack

6. Crée le fichier local.conf :

vi /home/nom du répertoire/devstack local.conf
puis ajouter ça

```
[[local|localrc]]
ADMIN_PASSWORD=password
DATABASE_PASSWORD=$ADMIN_PASSWORD
RABBIT_PASSWORD=$ADMIN_PASSWORD
SERVICE_PASSWORD=$ADMIN_PASSWORD
#FIXED_RANGE=172.31.1.0/24
#FLOATING_RANGE=192.168.20.0/25
#HOST_IP=10.3.4.5
```

7. Executer la commande ./stack.sh en tant que l'utilisateur stack :

cd /home/le nom du répertoire/devstack
su stack (puis entrer le mot de passe crée pour l'utilisateur stack)
./stack.sh

Voilà aprés l'installation commence et beaucoup de choses s'affichent à l'écran,car il est entrain d'installer tous les composants de Openstack cité dans II.3.1.Selon le débit Internet l'installation est plus ou moins longue (compter 40-60 minutes avec une ADSL classique).
La fin de notre installation est signalée ainsi **:[S9]**

Figure 26. Fin d'installation de Openstack.

IV. Conclusion:

Dans ce chapitre, on a présenté les outils logiciels et matériels ainsi que toutes les étapes et les démarches à suivre pour installer les différents composants d'OpenStack.

Bien que l'installation semble facile à première vue, mais on a beaucoup cherché avant de la finaliser avec succès, car à chaque fois un problème survenait, qu'il fallait résoudre pour passer à l'étape suivante. En plus du fait que certaines informations ne sont pas évidentes à trouver, pour mener à bien cette laborieuse installation et configuration.

Chapitre VI:

Réalisation& implémentation

I. Introduction

Dans ce chapitre nous allons présenter le travaille qu'en a réaliser selon les besoin de l'organisme de TycheCloud. Nous allons expliqué et définir les différents outils qu'en a utiliser pour pouvoir modifier le tableau de bord de Openstack et pour pouvoir réaliser le projet qui permet d'ajouter des utilisateur de Openstack. Pour cela nous allons parler du framwork Django.

II. Le Framwork Django

Le module Horizon de Openstack a été développer avec le framwork Django, pour cela nous avons étudier ce framework pour pouvoir faire des modification sur les templates et le css de ce tableau de bord.

II.1. Définition :

Django est un framework Pythondestiné au web, ce qui veut dire qu'il s'agit d'un ensemble d'outils et de bibliothèques coordonnées écrites en python. Il apporte les bases communes à la majorité des programmes ou des sites web, ce qui permet de simplifie le travail d'un développeur et lui éviter de réécrire plusieurs fois une même fonctionnalité. Ce n'est pas le seul dans sa catégorie, nous pouvons compter d'autres frameworks Python du même genre comme web2py, TurboGears, Tornado ou encore Flask. Il a cependant le mérite d'être le plus exhaustif, d'automatiser un bon nombre de choses et de disposer d'une très grande communauté.

II.2. Le fonctionnement de Django:

Lorsque nous parlons de frameworks qui fournissent une interface graphique à l'utilisateur (soit une page web soit l'interface d'une application graphique classique), nous parlons souvent de l'architecture **MVC : Modèle-Vue-Contrôleur**. L'architecture utilisée par Django diffère légèrement de l'architecture MVC classique. En effet, la « magie » de Django réside dans le fait qu'il **gère lui-même la partie contrôleur** (gestion des requêtes du client, des droits sur les actions…). Ainsi, nous parlons plutôt de framework utilisant l'architecture **MVT** : **M**odèle-**V**ue-**T**emplate.

A) L'architecture MVC :

Il s'agit d'un modèle distinguant plusieurs rôles précis d'une application, qui doivent être accomplis. Comme son nom l'indique, l'architecture **M**odèle-**V**ue-**C**ontrôleur est composé de trois entités distinctes, chacune ayant son propre rôle à remplir.

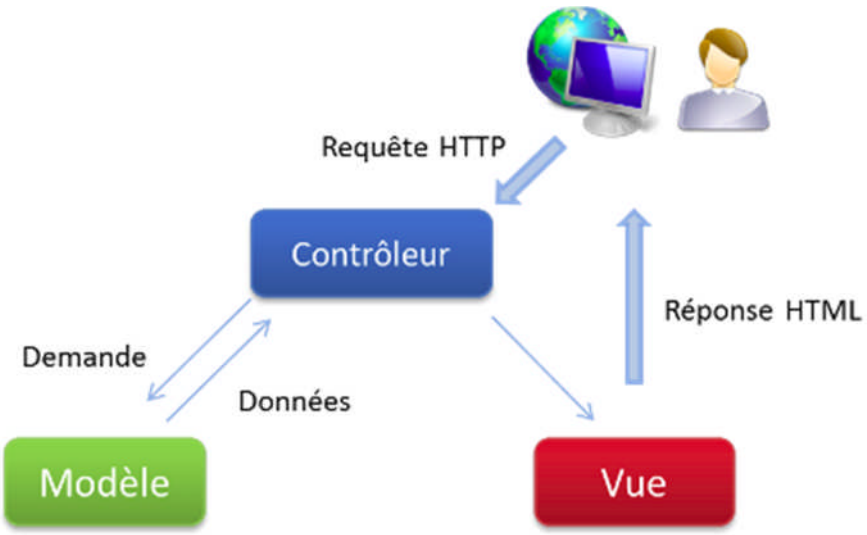


Figure 27.Schéma de l'architecture MVC.

Tout d'abord, **le modèle** représente une information enregistrée quelque part, le plus souvent dans une base de données. Il permet d'accéder à l'information, de la modifier, d'en ajouter une nouvelle, de vérifier que celle-ci correspond bien aux critères (on parle d'intégrité de l'information), de la mettre à jour, etc. Il s'agit d'une interface supplémentaire entre votre code et la base de données, mais qui simplifie grandement les choses, comme nous le verrons par la suite.

Ensuite **la vue** qui est, comme son nom l'indique, la ***visualisation de l'information***. C'est la seule chose que l'utilisateur peut voir. Non seulement elle sert à présenter une donnée, mais elle permet aussi de ***recueillir une éventuelle action*** de l'utilisateur (un clic sur un lien, ou la soumission d'un formulaire par exemple). Typiquement, un exemple de vue est une page web, ni plus, ni moins.

Finalement, **le contrôleur** ***prend en charge tous les événements de l'utilisateur*** (accès à une page, soumission d'un formulaire, etc.). Il se charge, en fonction de la requête de l'utilisateur, de récupérer les données voulues dans les modèles. Après un éventuel traitement sur ces

données, il transmet ces données à la vue, afin qu'elle s'occupe de les afficher. Lors de l'appel d'une page, c'est le contrôleur qui est chargé en premier, afin de savoir ce qu'il est nécessaire d'afficher.

B) L'architecture MVT :

Cette architecture reprend les définitions de modèle et de vue que nous avons vues, et en introduit une nouvelle : le template. Un template est un fichier HTML. Il sera récupéré par la vue et envoyé au visiteur ; cependant, avant d'être envoyé, il sera analysé et exécuté par le framework, comme s'il s'agissait d'un fichier avec du code. Django fournit un moteur de templates très utile qui permet, dans le code HTML, d'afficher des variables, d'utiliser des structures conditionnelles (***if/else***) ou encore des boucles (***for***), etc.

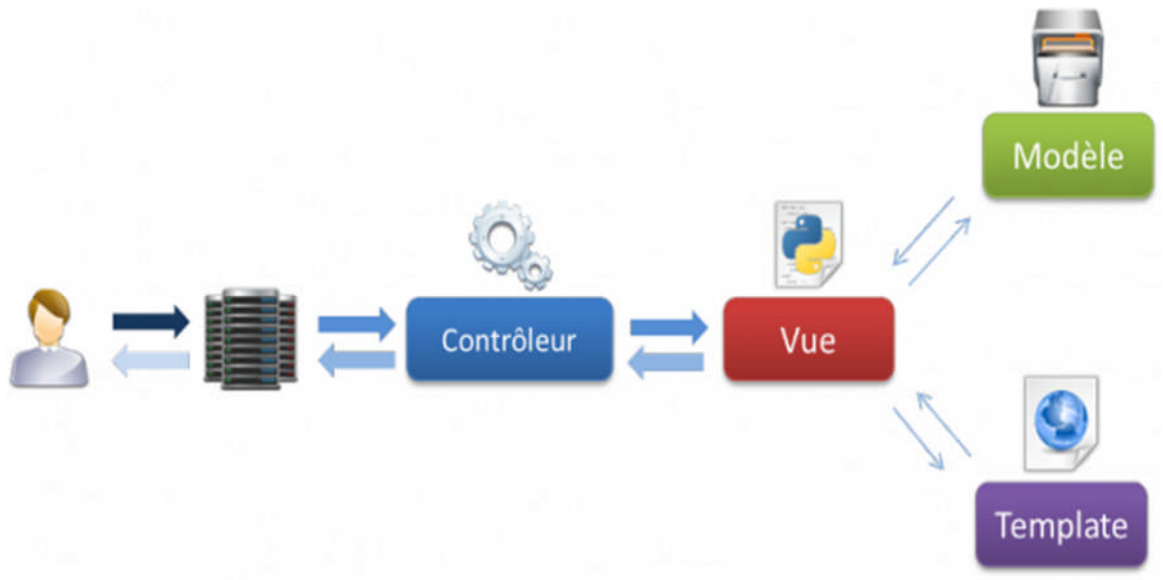


Figure 28.Schéma d'exécution d'une requête.

Concrètement, lorsque l'internaute appelle une page de votre site réalisé avec Django, le framework se charge, via les règles de routage URL définies, d'exécuter la vue correspondante. Cette dernière récupère les données des modèles et génère un rendu HTML à partir du template et de ces données. Une fois la page générée, l'appel fait chemin arrière, et le serveur renvoie le résultat au navigateur de l'internaute.

On distingue les quatre parties qu'un développeur doit gérer :

- Le routage des requêtes, en fonction de l'URL ;
- La représentation des données dans l'application, avec leur gestion (ajout, édition, suppression…), c'est-à-dire les modèles ;
- L'affichage de ces données et de toute autre information au format HTML, c'est-à-dire les templates ;
- Enfin le lien entre les deux derniers points : la vue qui récupère les données et génère le template selon celles-ci**.[S10]**

II.3. Projets et applications:

En plus de l'architecture MVT, Django introduit le développement d'un site sous forme de projet. Chaque site web conçu avec Django est considéré comme un projet, composé de plusieurs applications. Une application consiste en un dossier contenant plusieurs fichiers de code, chacun étant relatif à une tâche du modèle MVT que nous avons vu. En effet, chaque bloc du site web est isolé dans un dossier avec ses vues, ses modèles et ses schémas d'URL.

Ce principe de séparation du projet en plusieurs applications possède deux avantages principaux :

- Le code est beaucoup plus structuré. Les modèles et templates d'une application ne seront que rarement ou jamais utilisés dans une autre, nous gardons donc une séparation nette entre les différentes applications, ce qui évite de s'emmêler les pinceaux !
- Une application correctement conçue pourra être réutilisée dans d'autres projets par un simple copier/coller (ou un import), comme le montre la figure suivante.

III. L'interface de Horizon :

Rappelons que le tableau de bord (Dashboard) de Openstack a été développer avec Django, alors après avoir étudier le fonctionnement et les différents composants du framework Django, nous avons été en mesure de commencer les modifications sur le Dashboard de Openstack. Pour pouvoir apercevoir l'arborescence d'un projet Django sur le terminale de notre machine on a installé le packet tree avec la commande « sudo apt-get install tree **».** Pour les templates on a modifié les lignes html soit pour changer le logo de Openstack par celui de TycheCloud ou bien

pour allégé les pages trop encombré pas les écritures, en ce qui concerne le css, on a changé quelques caractéristiques des pages du Dashboard comme la couleur, le style d'écriture, les bordures et le style du side-menu. Voici les templates et les fichiers css qu'on a modifié :

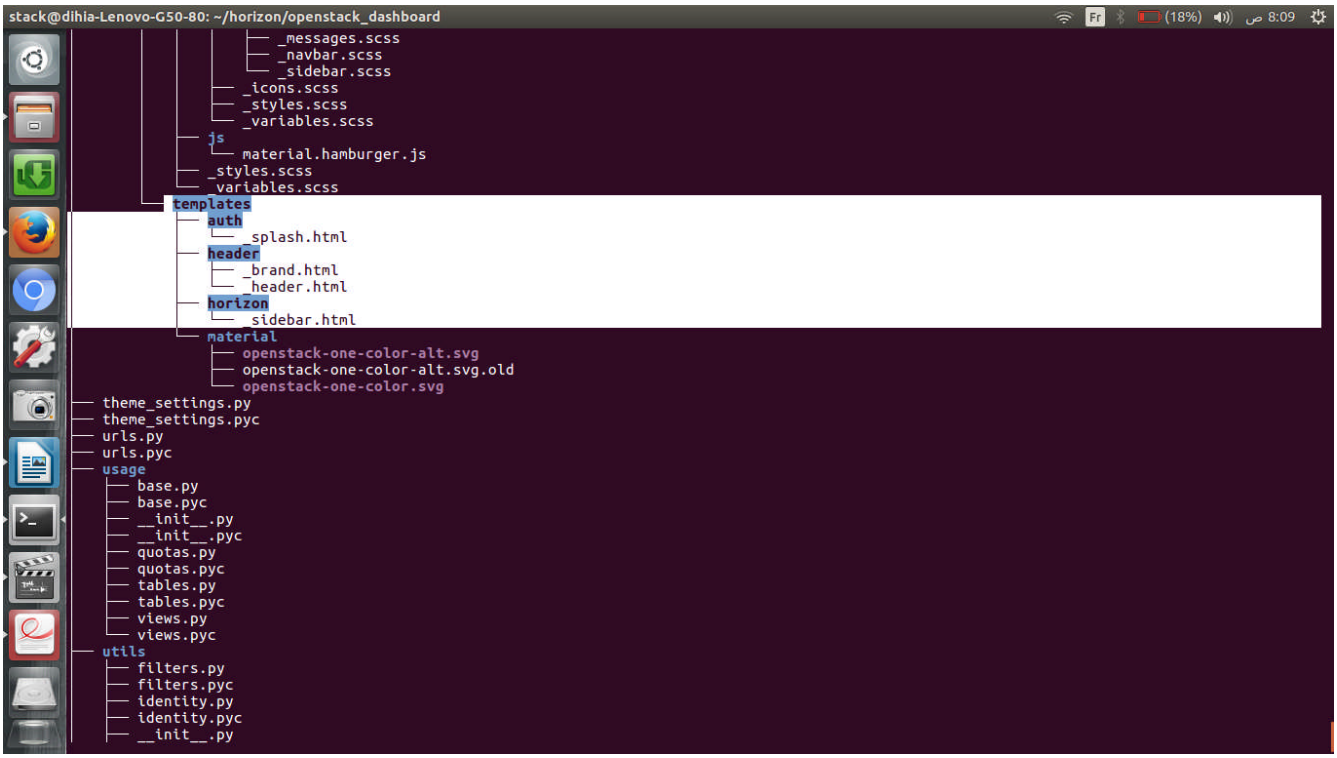


Figure 29.Arborescence des templates modifiées.

Sur cette figure en voit les templates qu'on a modifié.

- **Splash.html :** Pour modifier le logo de Openstack par celui de Tychecloud qui se trouve sur le formulaire d'authentification vers le Dashboard.
- **Brand.html et _header.html :** Pour modifier le logo de Openstack par celui de Tychecloud qui ce trouve sur les toutes les pages du Dashboard.
- **Sidebar.html :** Pour modifier le html de la barre menu du Dashboard.

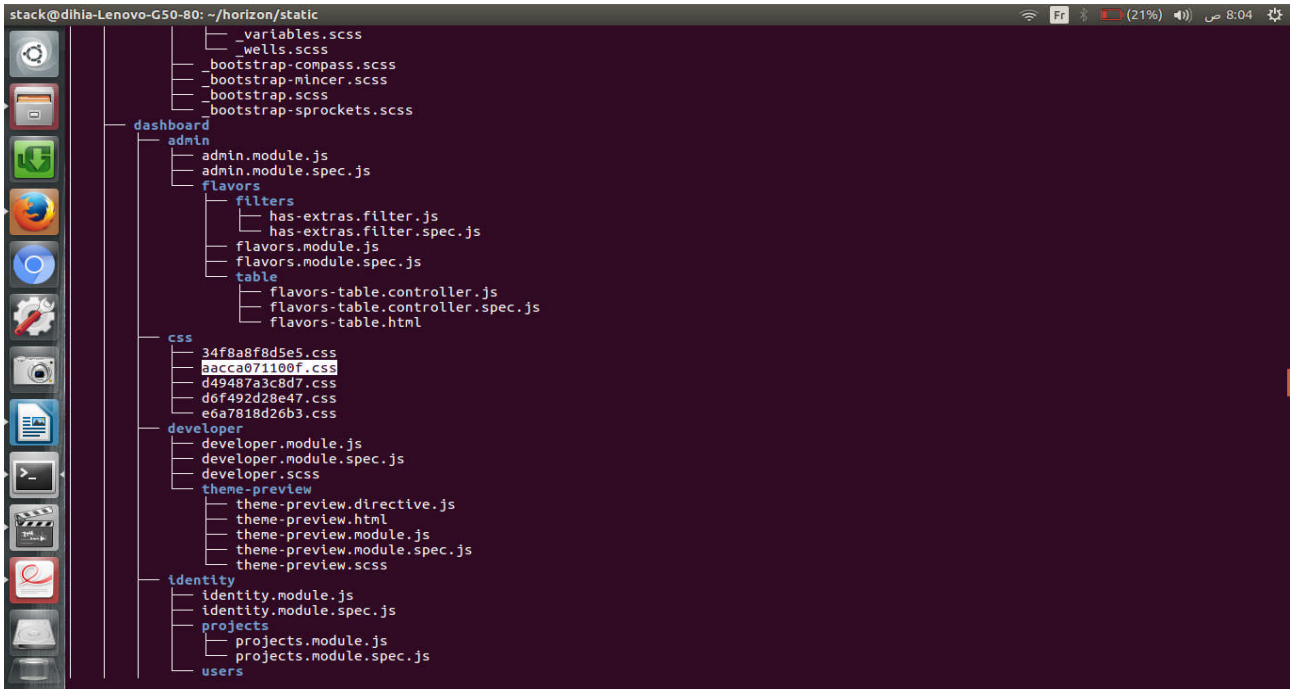


Figure 30.Arborescence du fichier css modifié.

Voici quelques captures d'écran de Horizon (Dashboard de Openstack) avant les modifications et après les modifications :

- Le formulaire d'authentification avant les modifications :

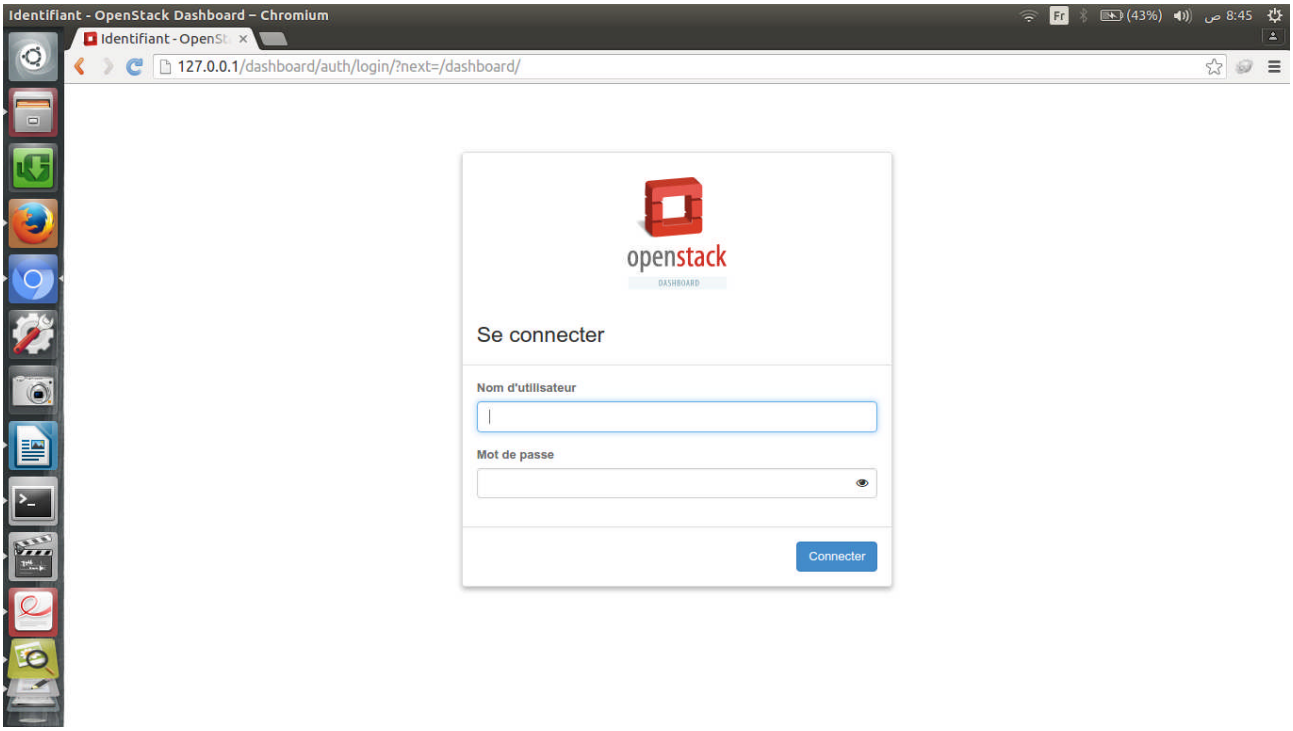


Figure 31. Formulaire d'authentification avant.

- Le formulaire d'authentification après les modifications :

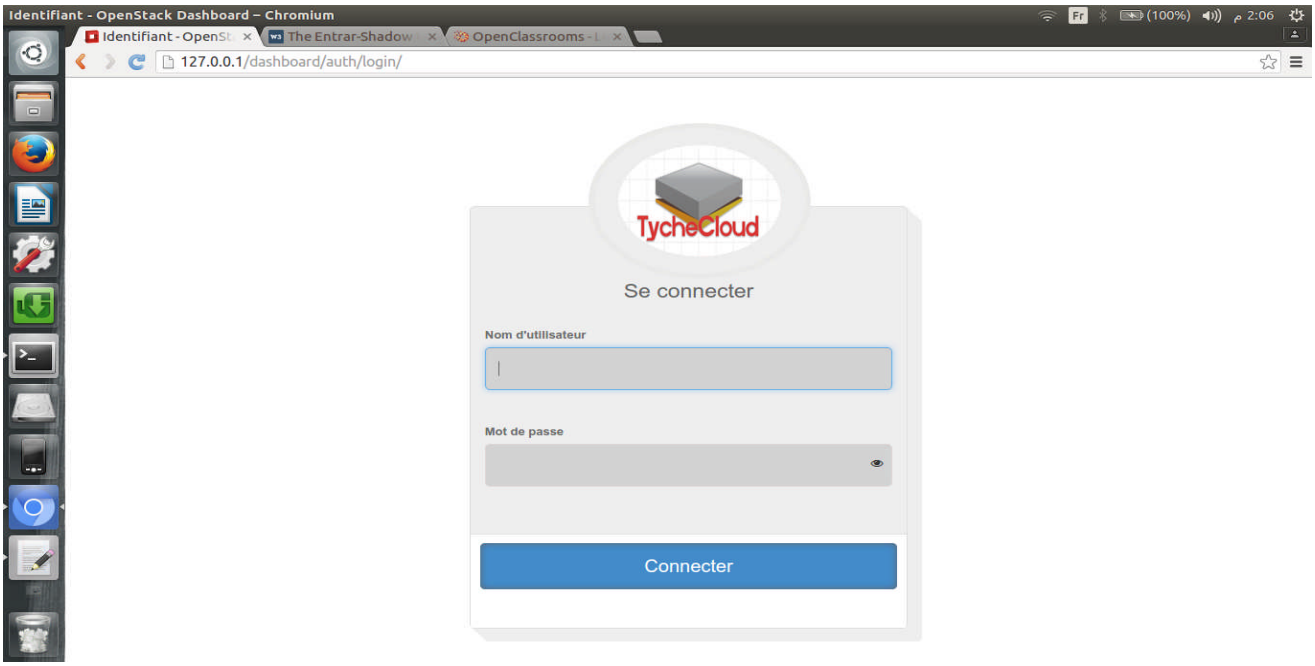


Figure 32.Formulaire d'authentification après.

- L'interface du Dashboard « Identité / Projet » avant les modifications :

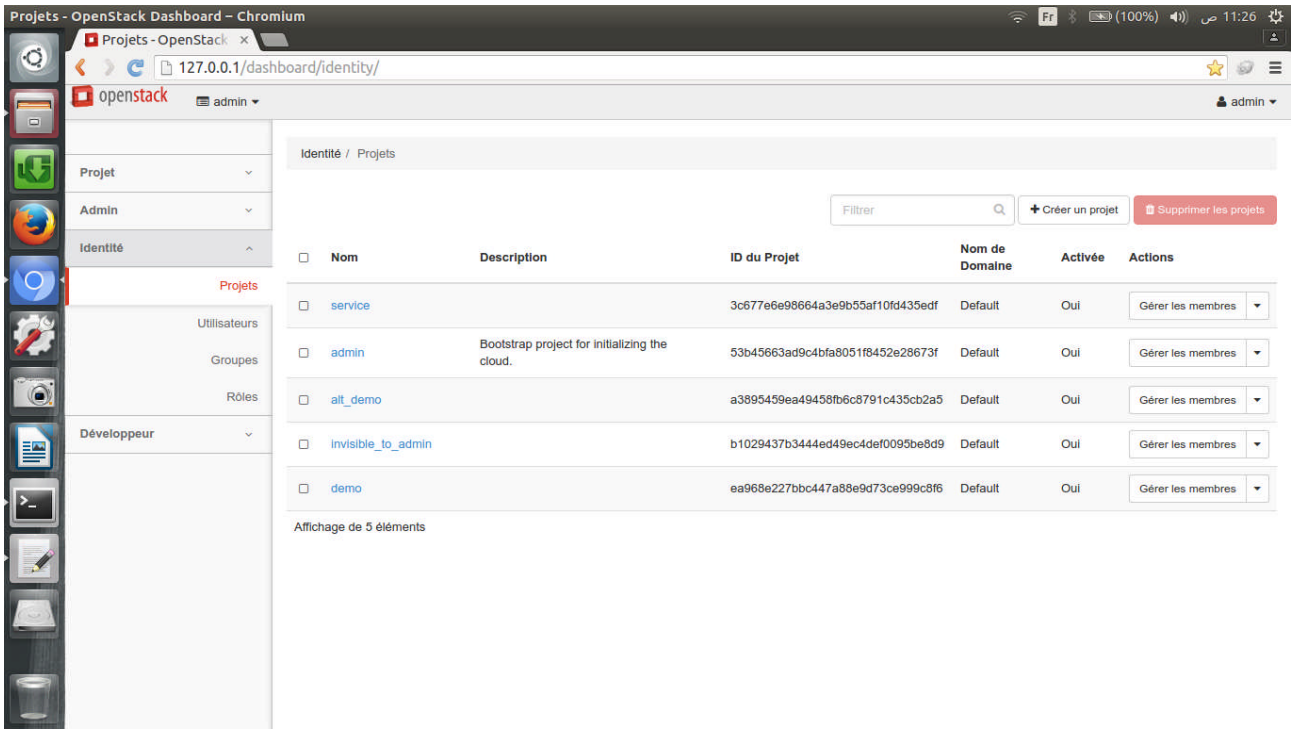


Figure 33. Page de Dashboard "Identité / Projet" avant.

- L'interface du Dashboard « Identité / Projet » après les modifications :

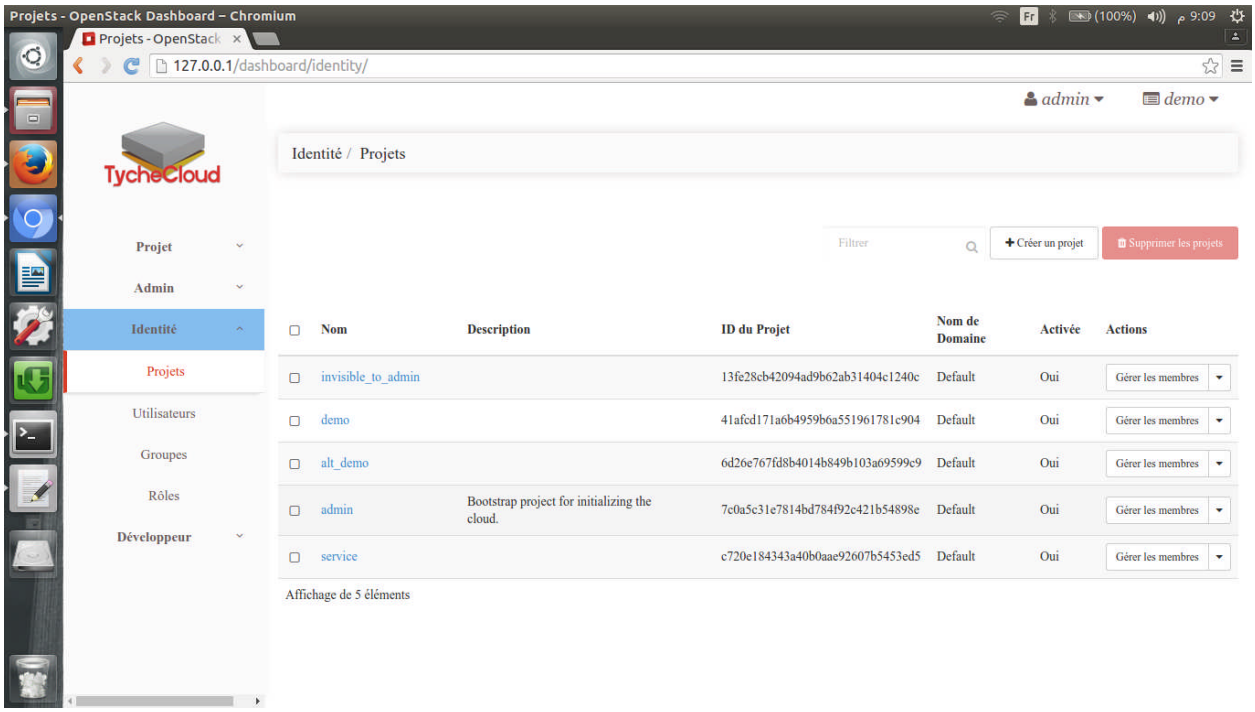


Figure 34. Page du D.ashboard "Identité / Projet" après.

- L'interface du Dashboard « Admin / Systeme / Vue d'ensemble » avant les modifications :

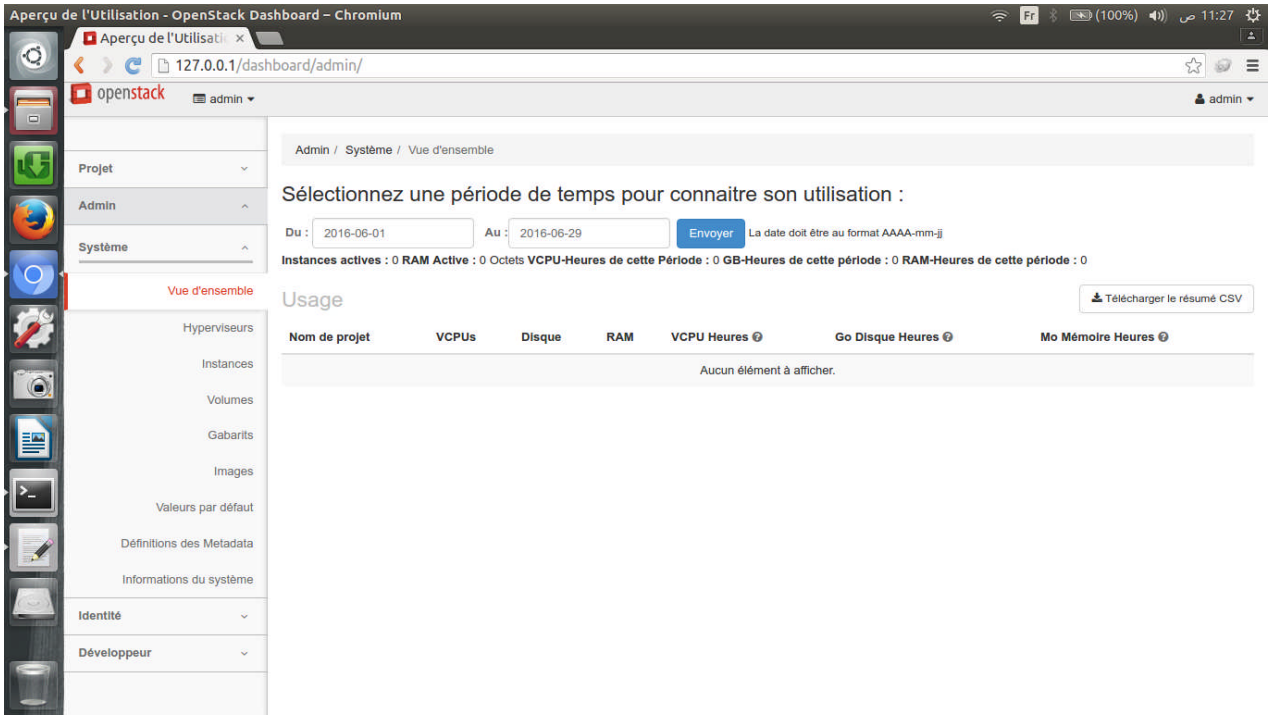


Figure 35. Page du Dashboard"Admin/SystemeVue d'ensemble" avant.

- L'interface du Dashboard « Admin / Systeme / Vue d'ensemble » après les modifications :

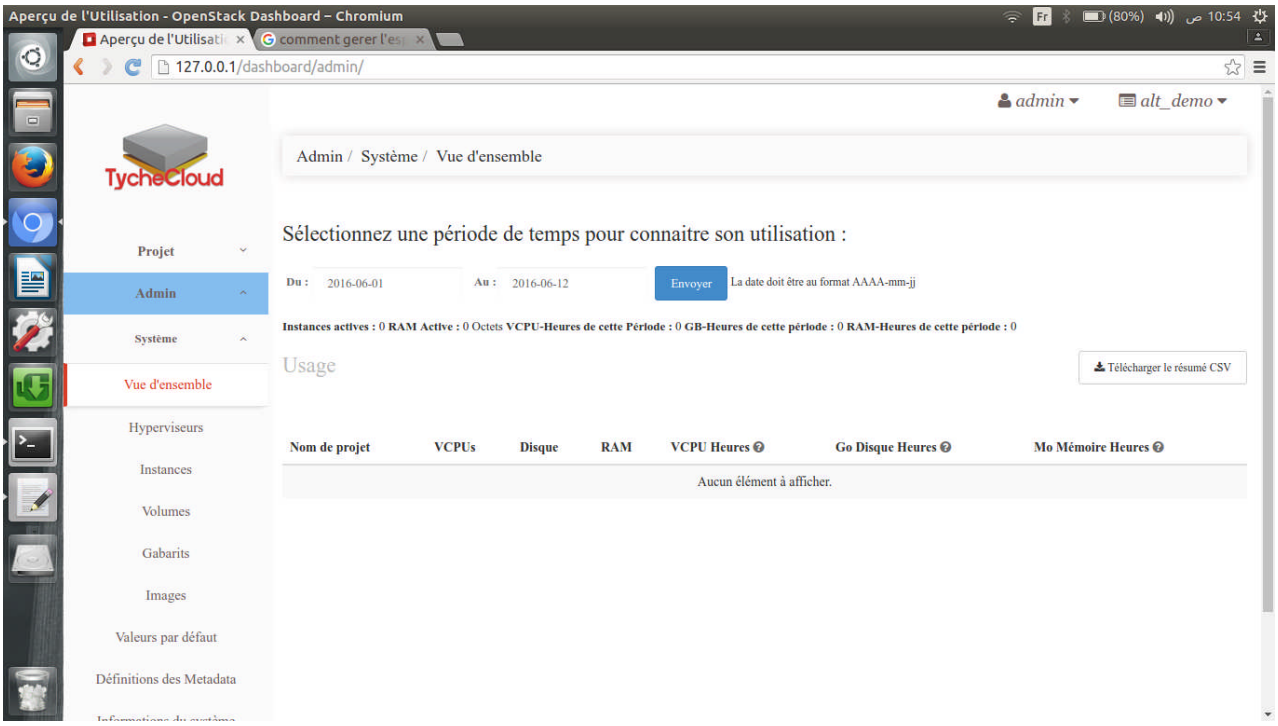


Figure 36.Page du Dashboard "Admin/Système/Vue d'ensemble" après.

IV. Développement du projet de création des utilisateurs :

Après avoir finie les modifications sur le tableau de bord de Openstack on a commencer a travailler sur le projet Django qui permet l'ajout de nouveaux utilisateurs a Openstack. Dans cette partie on vas expliquer comment on a fait la relation entre notre projet Django et l'interface Openstack qu'on a installer et présenter les outils qu'on a utiliser et quelques captures d'écran du site web qu'on a développer avec ce projet Django.

IV.1. Les outils utilisés:

Nous allons parler un peut de django-registration-redux et de maildev.

1. Django-registration-redux :

➢ Definition :

Le package django -registration- redux est une application add-on qui a été développé pour fournir des mécanismes de connexion , d'enregistrement et d'authentification .

Une fois installé, nous pouvons ajouter django-inscription-redux à tout projet basé sur Django que nous développons.

➢ Configuration :

La configuration par défaut permettra l'enregistrement de l'utilisateur avec le flux de travail suivant:

- Un utilisateur signe pour un compte en fournissant un nom d'utilisateur, adresse e-mail et mot de passe.
- A partir de cette information, un nouvel objet utilisateur est créé, avec son champ **is_active** défini sur **False**. En outre, une clé d'activation est générée et stockée, et un courriel est envoyé à l'utilisateur contenant un lien à cliquer pour activer le compte.
- En cliquant sur le lien d'activation, le nouveau compte est rendu actif (le champ **is_active** est réglé sur **True**); après cela, l'utilisateur peut se connecter.

Notons que le flux de travail par défaut nécessite django.contrib.auth soit installé, et il est recommandé que django.contrib.sites soit installé aussi bien (et ils sont installés par défaut avec le projet django). Nous avons également besoin d'avoir un serveur de messagerie de travail (pour l'envoi d'e-mails d'activation), et de fournir Django avec les paramètres nécessaires pour faire usage de ce serveur de messagerie (on vas utiliser maidev).

➢ Paramètres :

On a Commencé par ajouter register à INSTALLED_APPS de notre projet , et en spécifiant des paramètres supplémentaires :

ACCOUNT_ACTIVATION_DAYS : Ceci est le nombre de jours pendant lesquelles les utilisateurs devront activer leurs comptes après l'enregistrement . Si un utilisateur ne déclenche pas son activation dans ce délai, le compte restera définitivement inactif et peut être supprimer par les scripts de maintenance fournis dans django -registration- redux.

REGISTRATION_AUTO_LOGIN : Optionnel. Si cela est True , nos utilisateurs se connectent automatiquement quand ils cliquent sur le lien d'activation dans leur e-mail . Par défaut False.

```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
     'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'django.contrib.sites',
    'sarahCloud',
    'registration',
    'django.contrib.auth',
]


#django-registration-redux settings
ACCOUNT_ACTIVATION_DAYS = 7 # One-week activation for email validation;
REGISTRATION_AUTO_LOGIN = True # Automatically log the user in.
SITE_ID = 1
```

Figure 37. Image du settings.py.

Une fois que nous avons fait cela , On exécute python manage.py migrate pour installer le modèle utilisé par la configuration par défaut.

- **Paramétres URLs :**

Le backend par défaut inclut un URLconf Django qui met en place des modèles d'URL pour les vues dans django -inscription - redux , ainsi que plusieurs vues utiles dans django.contrib.auth (par exemple , connexion , déconnexion, changement de mot de passe / reset) . Ce URLconf peut être trouvé à registration.backends.default.urls , et ainsi peut simplement être inclus dans la configuration racine d'URL de notre projet. On a placé les URL dans le préfixe / accounts/ , nous avons ajouté ce qui suit à la racine de URLconf de notre projet :

(r'^accounts/', include('registration.backends.default.urls')),

- **Templates :**

Tous d'abord on a créé un répertoire qu'on a appeler registeration ou on a loger toutes les pages (templates) associées à l'application de Redux Django d'enregistrement .

En trouve les templates suivante :

- **registration/registration_form.html :**

Utilisé pour montrer le formulaire que remplira les utilisateurs pour s'inscrire. Par défaut, a le contexte suivant :

Forme

Le formulaire d'inscription . Ce sera une instance d'une sous-classe de django.forms.Form .

- **registration/registration_complete.html :**

Utilisé après avoir réussi le formulaire d'inscription . Ce modèle n'a pas de variables de contexte de sa propre , et doit simplement informer l'utilisateur qu'un courriel contenant des informations de compte activation a été envoyé.

- **registration/activate.html :**

Utilisé si l'activation du compte échoue. Avec la configuration par défaut , a le contexte suivant :

activation_key La clé d'activation utilisé lors de la tentative d'activation.

- **registration/activation_complete.html :**

Utilisé après l'activation du compte réussi. Ce modèle n'a pas de variables de contexte de sa propre , et doit simplement informer l'utilisateur que son compte est maintenant actif.

- **registration/activation_email_subject.txt :**

Utilisé pour générer la ligne d'objet de l'e-mail d'activation . Parce que la ligne d'objet d'un e-mail doit être une seule ligne de texte , toute sortie de ce modèle sera condensé à une seule ligne avant d'être utilisée . Ce modèle a le contexte suivant:

activation_key

La clé d'activation pour le nouveau compte.

expiration_days

Le nombre de jours restants au cours de laquelle le compte peut être activé .

site

Un objet représentant le site sur lequel l'utilisateur est enregistré.

- **registration/activation_email.txt :** Utilisé pour générer le corps du texte de l'email d'activation . Affiche un lien , l'utilisateur peut cliquer pour activer le compte . Ce modèle a le contexte suivant:

activation_key

La clé d'activation pour le nouveau compte.

expiration_days

Le nombre de jours restants au cours de laquelle le compte peut être activé .

site

Un objet représentant le site sur lequel l'utilisateur est enregistré.

user

Le nouveau compte utilisateur.

- **registration/activation_email.html :**

Le modèle du nouvel utilisateur est utilisé pour générer le corps HTML de l'email d'activation . Devrait afficher le même contenu que la version texte de l'email d'activation . Le contexte disponible est la même que la version du texte du template.unt.

2. Maildev :

➢ Definition :

MailDev est un serveur SMTP couplé à une interface Web qui intercepte les emails émanant de votre serveur Web afin de les visualiser et les tester.

MailDev dispose de nombreuses fonctionnalités vraiment très pratique on peut sité :

- Interface web claire et épurée
- Dès qu'un email est intercepté, un indicateur apparaît dans l'onglet.
- Visualisation du header complet des emails en un clic.
- Possibilité de tester la "*responsivité*" des emails.
- Simplicité de mise en place de l'outil.

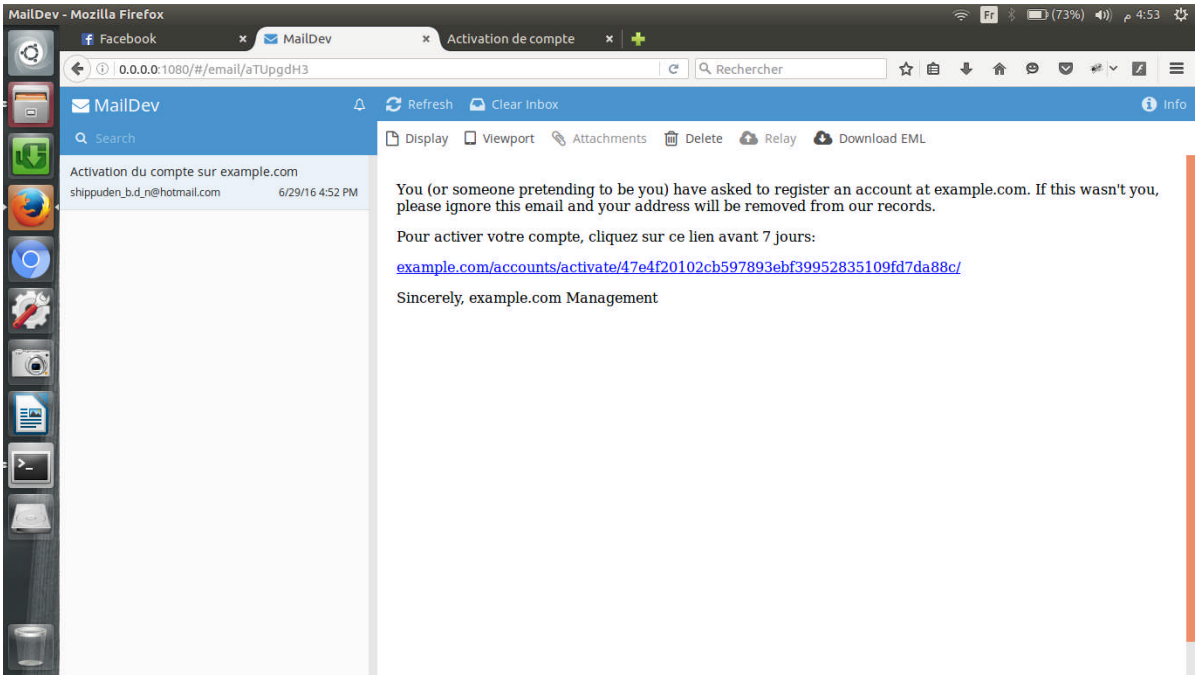


Figure 38. Interface de Maildev.

IV.2. L'interface réalisée :

Comme nous l'avons déjà mentionner, ce projet Django que nous avons développer a pour fonctionnalité de permettre aux clients de TycheCloud de crée un compte utilisateur pour pouvoir accéder à l'interface Openstack. Dans cette partie nous avons montré des captures d'écran, de la page d'acceuil, de la page inscription et de la page authentification. Et nous avons parler aussi de l'interface administrateur propre a Django.

- **La page d'acceuil :**

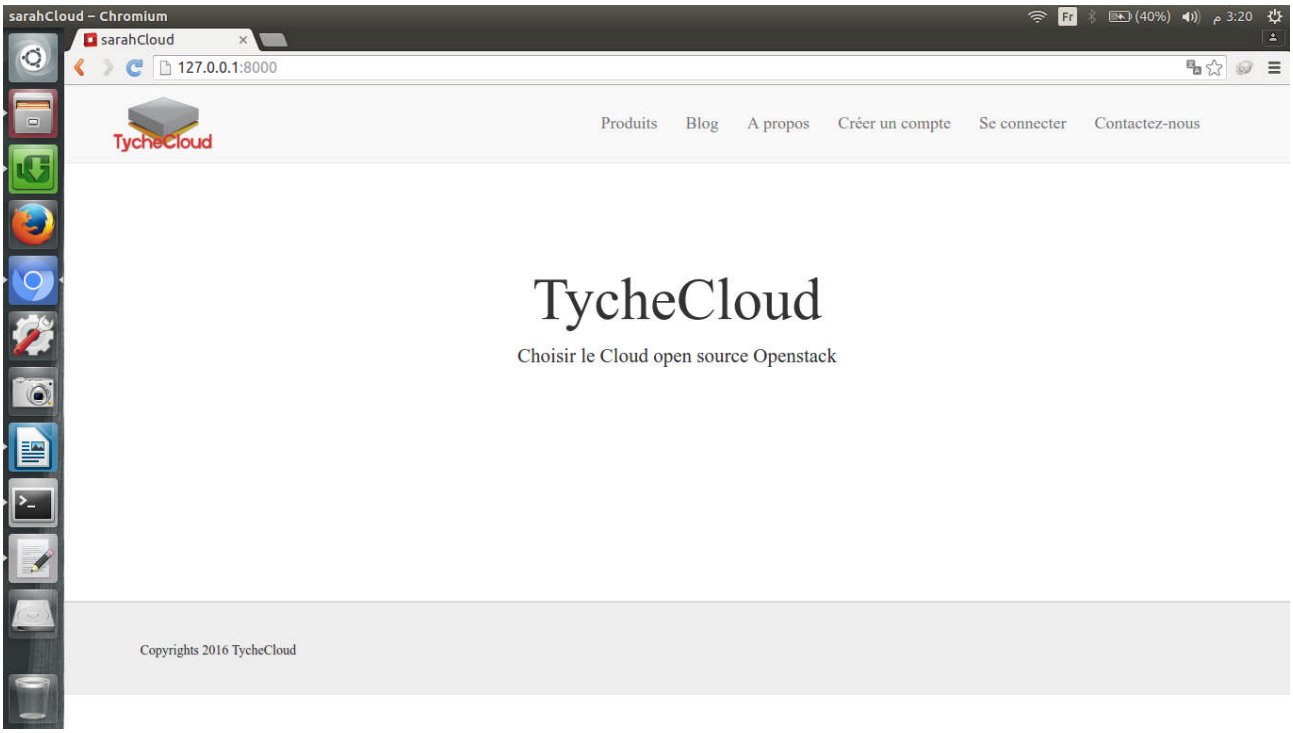


Figure 39. Page d'acceuil.

- **La page création du compte :**

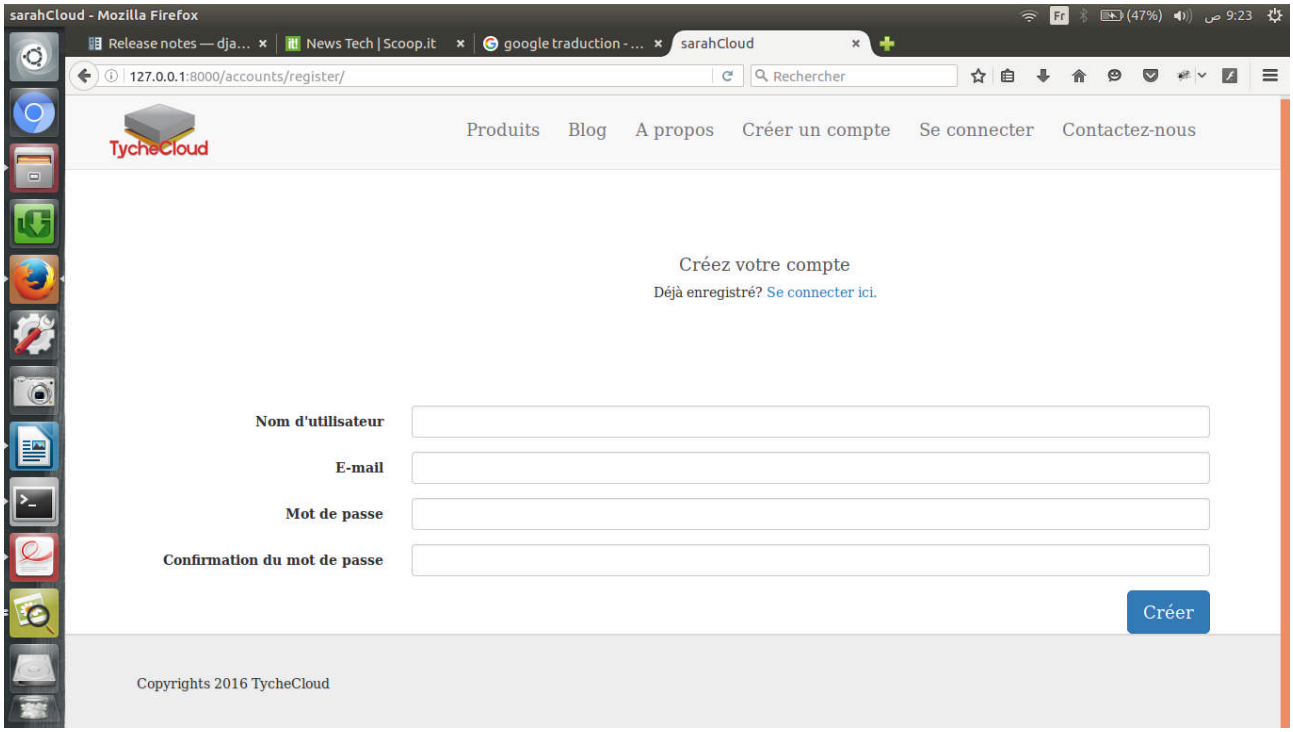


Figure 40. Page inscription.

➢ Lapage d'authentification :

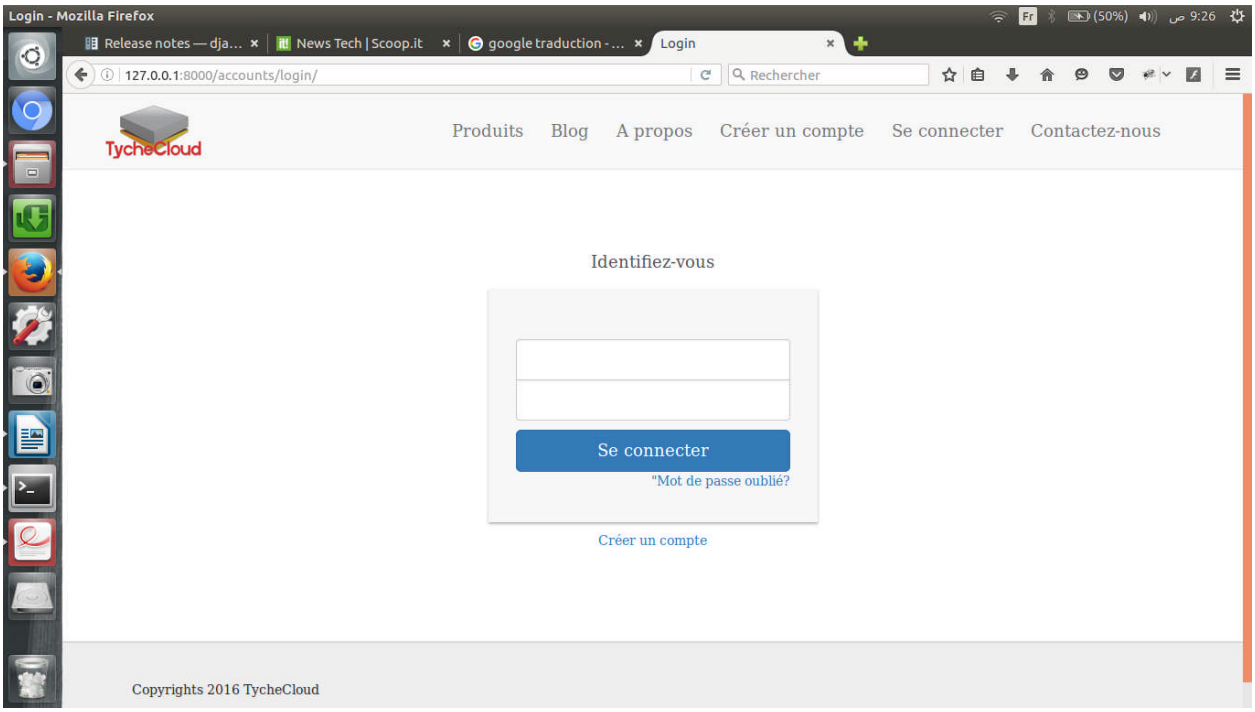


Figure 41. Page Authentification.

➢ L'administrateur de Django :

Une des parties les plus puissantes de Django est l'interface d'administration automatique. Il lit les métadonnées des modèles pour fournir une interface rapide, c'est ou en peut visualiser toutes les données entré par le client durant son passage sur le site .Seul les administrateurs du site qui peuvent accédé a cette interface. C'est dans cette interface qu'on valide l'inscription d'un client et qu'on vas lui crée un compte Openstack.

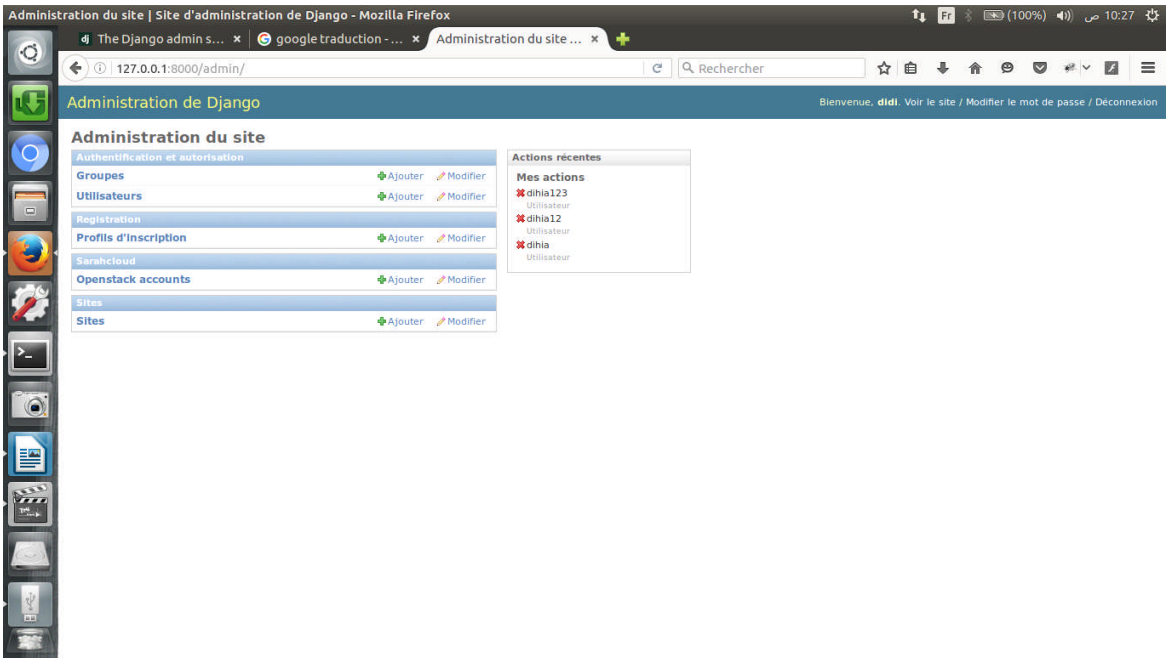


Figure 42. Administration Django.

V. Conclusion

Nous avons vu dans ce chapitre les deux travailles qu'on a accomplis pour l'organisme TycheCloud. Après avoir compris et métriser le fonctionnement du Framework Django, nous avons montré comment on a pu modifier le tableau de bord de Openstack, les templates et les fichiers css. Puis on a exposé le projet qu'on a développé dans le but que les clients de TycheCloud puissent s'inscrire et accéder a l'interface de Openstack en tant que user.

CONCLUSION GENERALE

CONCLUSION

Notre projet porte sur le développement d'une interface Web autour du module d'openstack (Cloud Computing) pour la société TycheCloud.

Pour mener à terme notre travail, on a donné une présentation du Cloud Computing, , ce qui nous a permis de bien comprendre et concevoir notre application. Ainsi une présentation d'Openstack qui nous a permis d'en tirer les objectifs de notre application.

Ensuite nous avons bien étudier le langage de style Css avec quoi l'interface d'Openstack a été programmée, ce qui nous a permis de le modifier pour répondre aux besoin de notre société .

Enfin on a concrétisé notre travailen réalisant notre projet (site)pour la société TycheCloud qui répond aux objectifs à base du framework Django et le langage de programmation Python sous ,Linux (Ubuntu) .

Le site que nous avons réalisé permet d'assurer :

- la création des utilisateurs du site ,donc des utilisateurs Openstack.
- Aux utilisateurs, la possibilité d'accéder à leurs comptes openstack après les avoir activer par l'administrateur du site (La société).Ainsi avoir des instances ou des serveurs....etc aprsès avoir payer pour et les géreà n'importe quel endroit et à n'importe quel moment.

- A la société, la possibilité de gérer et de communiquer facilement avec ses clients.

Ce travail nous a permis de répondre aux plusieurs besoins de l'entreprise, néanmoins les perspectives de développement restent ouvertes telle que :

- Réalisation d'une plateforme de discussion en ligne entre les membres.
- Réalisation d'une plateforme du Cloud Computing permettant de fournir plusieurs services aux utilisateurs.

ANNEXES

I.Installation d'Openstack :

Nous devons avoir une version de linux pour l'installation de openstack, alors en a installé la version Ubuntu 16.04 LTS Server 64bits. Les étapes de l'installation d'openstack ALL-IN-ONE sur une même machine sont :

- **Ajouter un utilisateur stack :**

```
groupadd stack
useradd -g stack -s /bin/bash -d /opt/stack -m stack
sudo passwd stack
```

- **Ajouter les droits d'accés pour l'utilisateur stack :**

```
Vi /etc/sudoers
Puis en clic sur "i"
Et en colle ça "stack ALL=(ALL:ALL) NOPASSWD: ALL "
```

- **Installer Devstack**

```
sudo apt-get install git –y
git clone git://github.com/openstack-dev/devstack.git
```

- **Les droits d'accés pour les deux repertoires :**

```
sudo chown -R stack:stack /opt/stack
sudo chown -R stack:stack /home/le repertoire de devstack/devstack
```

- **Configuration de Devstack :**

```
Cd /home/le repertoire de devstack/devstack
Vi local.conf
En clic sur “i”
Et en colle ça :
[[local|localrc]]
ADMIN_PASSWORD=secret
DATABASE_PASSWORD=$ADMIN_PASSWORD
RABBIT_PASSWORD=$ADMIN_PASSWORD
SERVICE_PASSWORD=$ADMIN_PASSWORD
```

- **A la fin exécuter la commande ./stack.sh :**

```
Su stack
Cd /home/le repertoire de devstack/devstack
./stack.sh
```

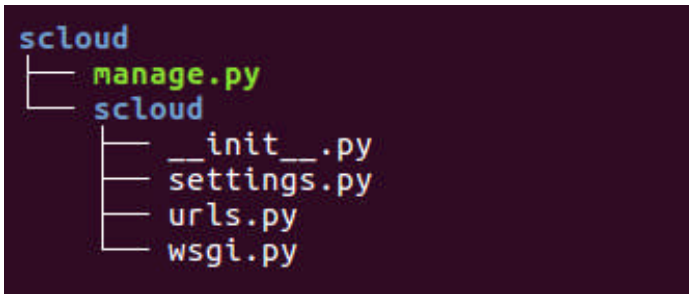
II. Création du projet Django :

On n’a pas besoin de faire l’installation de django et de python par ce que leur installation est faite automatiquement avec l’installation d’Openstack.

- **Création du projet scloud :**

```
Cd /home/repertoire
django-admin.py startproject scloud
```

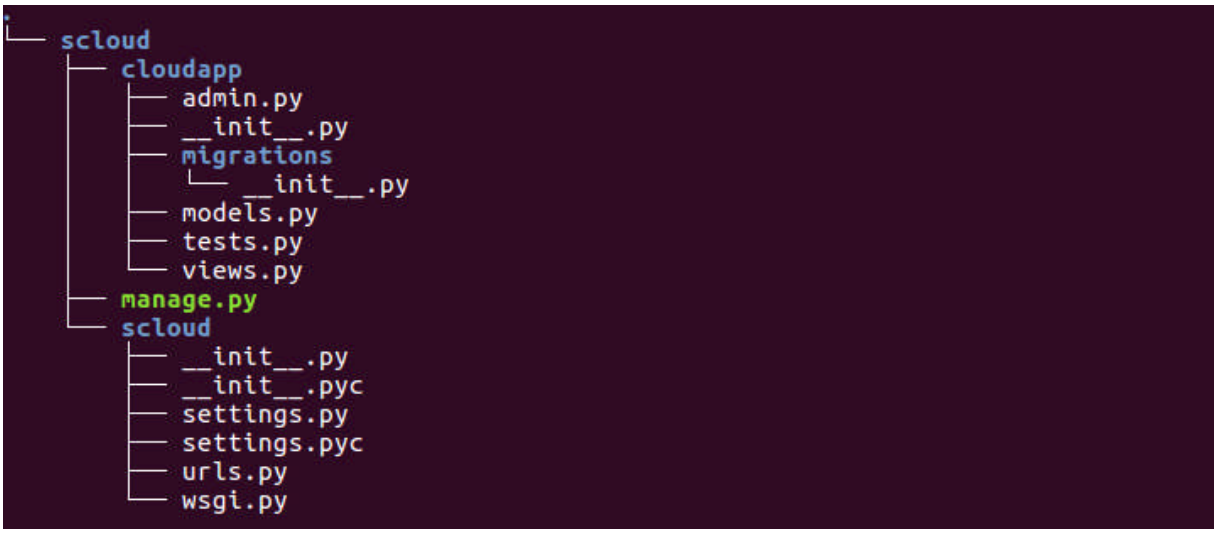
Un nouveau dossier nommé scloud est apparu et possède la structure suivante :



- **Creation de l'application :**

```
Cd scloud
python manage.py startapp blog
```

Comme avec startproject , startapp crée un dossier avec plusieurs fichiers à l'intérieur. La structure de notre projet ressemble à ceci :



- **Installation de Django-registration-redux :**

Installation automatique via un gestionnaire de paquets Pip :

```
Sudo pip install django-registration-redux
```

- **Mise en place des templates :**

```
git clone https://github.com/macdhuibh/django-registration-
templates.git
```

- **Installation de Maildev :**

```
Cd /home
apt-get install nodejs-legacy
apt-get install -g npm
apt-get update
npm install maildev
```

BIBLIOGRAPHIE /WEBLIOGRAPHIE

Bibliographie

[1] : La virtualisation Projet de Recherche et Communication Scientifique

Année académique 2009_2010
Université Libre de Bruxelles, Université d'Europe.

[2]: Cloud Application Architectures

Building Applications and Infrastructure in the Cloud

de Georges Reese

[3]:ObjectifCloud Une démarche pratique orientée services

[4]: Frediric lau 2013

Livre blanc sur les fondamentaux du cloud computing

[5]: Le guide complet Sylvain Caicoya

Jean george sory

Webiographie

[S1] https://fr.wikipedia.org/wiki/Cloud_computing

[S2]http://www.histoire-cigref.org/blog/cloud-computing-histoire-de-linformatique-en-nuages/

[S3] http://www.figer.com/Publications/nuage.htm#.V3SyfvmLTIU

[S4] http://www.vinci-energies.com/cest-deja-demain/pour-un-monde-connecte/data-center-bienvenue-dans-les-usines-a-donnees/

[S5] http://www.channelnews.fr/qui-sont-les-principaux-vendeurs-de-services-de-cloud-l-computing-r-19989

[S6] http://www.journaldunet.com/solutions/expert/50770/migration-vers-le-cloud---avantages-et-inconvenients-pour-les-pme.shtml

[S7] http://www.lesechos.fr/idees-debats/cercle/cercle-118688-cloud-computing-la-securite-en-question-1065146.php

[S8] http://linuxfr.org/news/presentation-d-openstack

[S9] http://docs.openstack.org/liberty/fr/install-guide-ubuntu/

[S10] https://openclassrooms.com/courses/developpez-votre-site-web-avec-le-framework-django