

Rapport de stage – 2^{ème} année de BTS SIO

Option SISR (Solution d'infrastructure
Systèmes et Réseaux)

PANEI TEO

Du 6 janvier au 14 février 2025

Tuteur de stage : Monsieur SKIMANI

Enseignant référent : Monsieur REZZAG

Etablissement de formation : Business School du beaujolais, 96 rue Henri
Depagneux, 69400 Limas

Etablissement de stage : RATPDEV Lyon, 14 rue Olympe de Gouges, 69100
Villeurbanne

Remerciements

Il m'apparaît opportun de débiter ce rapport de stage par des remerciements.

Tout d'abord, je tiens à exprimer ma profonde gratitude envers M. Karim Skimani, responsable du pôle production de RATP DEV. En tant que tuteur, il m'a donné l'opportunité d'intégrer l'entreprise, m'a formé et m'a accompagné tout au long de cette expérience professionnelle. Sa bienveillance, sa patience et sa pédagogie, m'ont permis de progresser, tout en me laissant une grande autonomie.

Je tiens également à remercier les équipes de techniciens, d'administrateurs, d'intégrateurs et de chefs de projet, qui ont su me faire confiance lors de ce stage dans le monde professionnel et ont partagé leurs connaissances de manière très pédagogique. Leur disponibilité et la qualité de leur encadrement ont été des éléments fondamentaux à mon épanouissement au sein de l'entreprise.

Je remercie l'ensemble des employés de RATP DEV et plus particulièrement de la Direction du Système d'Information (DSI) pour les conseils qu'ils ont pu me prodiguer et l'accueil qu'ils m'ont réservé, ce qui a contribué à faire de cette expérience un moment très profitable

Je souhaite exprimer ma gratitude envers l'équipe pédagogique et administrative de la Business School, tout particulièrement à Pierre LAFORET.

Grâce à ce stage, j'ai beaucoup appris, en termes de compétences professionnels et même en termes de compétences humaines.

Sommaire

I.	Introduction	4
A.	Présentation de l'établissement	4
B.	Domaine de l'activité	5
C.	Structure du site	5
II.	Présentation des missions	6
A.	Les logiciels employés au sein de RATP Dev Lyon	6
B.	Missions effectuées	10
III.	Conclusions	16
A.	Apport de stage	16
IV.	Annexes	17

I. Introduction

A. Présentation de l'établissement

La naissance de RATP Dev, en 2002, marque un tournant. L'objectif : développer, exploiter et maintenir de nouveaux réseaux en France, en Europe et dans le monde, en s'appuyant et en déployant l'expertise du groupe RATP. Depuis, nous allons de l'avant. 1,5 milliard de voyageurs empruntent nos réseaux chaque année. Nous avons grandi, notre expertise aussi. Mais notre passion demeure : construire avec vous, pour apporter aux villes du monde l'excellence d'un service innovant et durable.

Notre maison-mère, la RATP, opérateur de transport de Paris et sa région, assure 12 millions de voyages par jour, sur l'un des réseaux les plus denses et multimodaux au monde.

En tant que 3e opérateur mondial de transports publics, le groupe RATP exploite et maintient, dans le monde entier, des dizaines de milliers de kilomètres de lignes de métro, de bus, de tramway et de systèmes ferroviaires urbains et interurbains.

Tous les jours, nous innovons pour une ville plus durable et intelligente, en concevant, exploitant et maintenant des réseaux au service du bien-être de nos voyageurs. Avec passion et détermination.



Figure 1 – RATP Dev au niveau mondial

B. Domaine de l'activité

Exploitation, service et maintenance... RATP Dev conçoit, déploie et maîtrise toute la chaîne de la mobilité. Et vous offre, partout dans le monde, des solutions sur mesure qui redessinent les villes et embellissent les déplacements.

Exploitation :

Le cœur de notre métier. Nous transportons 1,5 milliard de voyageurs par an, partout dans le monde. Le fruit de notre excellence opérationnelle et de nos solutions techniques variées et innovantes. Toutes les ressources d'un leader international avec un accompagnement sur-mesure, de proximité.

Maintenance :

Améliorer et maintenir tous les réseaux, tous les modes, du métro le plus emprunté du monde au train le plus rapide d'Afrique. Des expertises uniques et plus de 100 opérations, dans le monde entier. Pour être transporté, en toute sérénité.

Service :

Une approche globale et des services adaptés, pour vous accompagner avant, pendant et après le voyage. Avec un écosystème digital complet, et l'objectif de 100% des démarches accessibles en ligne, 24/7. Nos solutions transforment le transport en temps utile et agréable et facilitent la mobilité.

C. Structure du site

RATP Dev Lyon, c'est 80 métiers répartis sur l'ensemble de l'agglomération lyonnaise. Les effectifs se répartissent entre les différentes unités de transport (UT), les agences commerciales, les services techniques et le siège social. Il est à noter que l'ensemble des métiers peuvent être occupés sans distinction de genre par des femmes ou des hommes. Qui plus est, RATP Dev Lyon mène des actions pour faciliter le recrutement et l'intégration des femmes.

Les métiers proposés par l'entreprise sont liés à la conduite, la maintenance, l'exploitation du réseau, le contrôle et les fonctions supports qui sont réparties entre les différentes directions.

RATP Dev Lyon, c'est 4600 salariés dont 2700 conducteurs répartis sur les différents modes de transport.

Les 1900 salariés hors conducteurs œuvrent au siège social ou sur le terrain pour :




II. Présentation des missions

Dans cette partie, je vais vous présenter mes différentes missions que j'ai faite au cours de mon stage.

Mais tout d'abord je vais vous faire une rapide présentation des outils qui sont utilisées au sein de RATP Dev Lyon

A. Les logiciels employés au sein de RATP Dev Lyon

RATP Dev Lyon dispose de plus de 200 applications ([Annexe 4](#) - Cartographie applicative) afin de faciliter la gestion de l'activité de l'entreprise. L'entreprise dispose également de logiciels spécifiques permettant de superviser, configurer et administrer les équipements réseau, les serveurs et autres machines physiques ou virtuelles.

-  VMware vSphere vSphere est une plateforme de virtualisation développée par VMware. Elle offre un ensemble complet de fonctionnalités pour la création, la gestion et la virtualisation des infrastructures informatiques. vSphere utilise l'hyperviseur VMware

ESXi pour créer et gérer les VMs et permet de partager les ressources matérielles entre les VMs. vSphere fournit des outils avancés pour la gestion des ressources, tels que la

répartition et l'équilibrage de charge ou la gestion de la mémoire et des performances. vSphere offre des fonctionnalités de haute disponibilité pour assurer la continuité des opérations. En cas de défaillance matérielle, les VMs sont automatiquement redémarrées sur d'autres serveurs disponibles. vSphere prend également en charge la migration en temps réel des VMs, permettant de les déplacer d'un serveur physique à un autre sans interruption de service. Enfin, vSphere propose un centre de contrôle centralisé appelé vCenter Server.

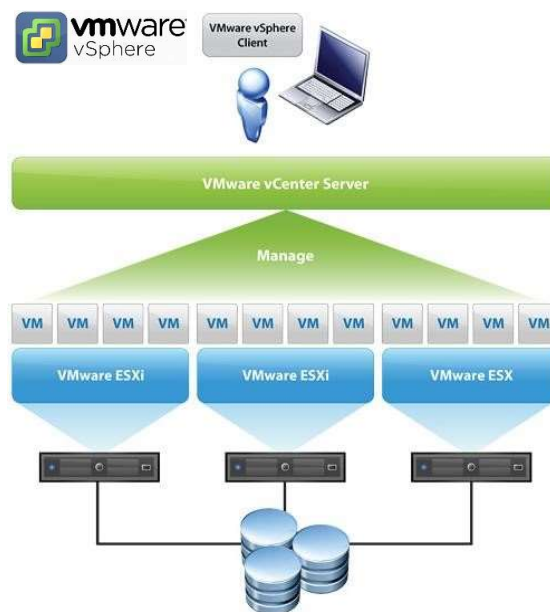


Figure 2 - VMware vSphere

Data Protector

Il s'agit d'un logiciel de sauvegarde et de récupération de données développé par Hewlett Packard Enterprise (HPE). Il prend en charge, notamment, les serveurs physiques, les VMs, les bases de données (BDD), les applications et les environnements cloud.

Veeam Backup & Replication

Veeam est un logiciel de sauvegarde et de restauration développé par Veeam Software. Il est conçu pour protéger les données et les VMs dans les environnements virtuels, physiques et cloud. Veeam prend en charge la sauvegarde incrémentielle, ce qui réduit l'espace de stockage requis et le temps nécessaire pour effectuer les sauvegardes. Il permet de restaurer les données à différents niveaux de granularité. Par exemple, vous pouvez restaurer une VM complète, des fichiers individuels, des applications spécifiques ou même des objets au sein des applications (courriel, compte AD...). Veeam prend également en charge la réplication des données selon les stratégies de stockage adoptées.



Figure 3 - Fonctionnement du stockage Exagrid avec Veeam

Intelligent Management Center

IMC est un logiciel de gestion centralisée de réseau développée par HPE qui permet de superviser et de contrôler efficacement leurs infrastructures réseau.

Check Point Firewall et Smart Console

Il s'agit d'un pare-feu développé par Check Point Software Technologies. Il propose le filtrage des paquets, l'inspection approfondie des paquets (Deep Packet Inspection, DPI) et le contrôle d'accès. Il prend en charge les connexions VPN sécurisées, la prévention des intrusions (Intrusion Prevention System, IPS), et offre une console de gestion centralisée nommée Smart Console.

ZABBIX

ZABBIX est un logiciel libre permettant de surveiller l'état des services réseau, serveurs et autres matériels réseau. Il produit notamment des graphiques dynamiques de consommation des ressources et permet de configurer des alertes permettant de repérer rapidement les incidents.

Rundeck

Rundeck est un logiciel libre permettant l'automatisation d'administration de serveurs via la création de jobs ou tâches.

Ansible

Ansible est une plateforme logicielle libre pour la configuration et la gestion des ordinateurs. Elle combine le déploiement de logiciels multinœuds, l'exécution des tâches ad-hoc et la gestion de configuration.

PuTTY

Les équipements

Au niveau réseau

Des commutateurs et des routeurs qui s'administrent différemment en fonction des marques, la logique restant la même, seuls les commandes varient.



Figure 4 - Exemple de switch CISCO

Au niveau stockage

Sur bande avec des lecteurs dédiés ou sur disque avec les baies de stockage Exagrid.

Au niveau virtualisation

Des serveurs de virtualisation hébergeant les VMs.

B. Missions effectuées

Dans cette partie je vais vous parler des missions effectuées. Mes missions ont été axé sur PowerShell notamment faire des scripts pour automatiser.

PowerShell est une solution multiplateforme d'automatisation des tâches, composée d'un interpréteur de commandes (Shell), d'un langage de script et d'un Framework de gestion de la configuration. PowerShell s'exécute sur Windows, Linux et macOS.

PowerShell est un interpréteur de commandes moderne qui comprend les meilleures fonctionnalités d'autres interpréteurs de commandes populaires. Contrairement à la plupart d'entre eux, qui acceptent et retournent uniquement du texte, PowerShell accepte et retourne des objets .NET.

1^{er} script :

```
1 $names = Get-content "names.txt"
2
3 ForEach ($name in $names) {
4     If (Test-Connection -ComputerName $name -Count 1 -ErrorAction SilentlyContinue) {
5         Write-Host "$name is up" -ForegroundColor Green
6     } Else {
7         Write-Host "$name is down" -ForegroundColor Red
8     }
9 }
```

Ce premier script avait pour but de faire un test ping sur une liste d'imprimante.

Pour cela il faut d'abord créer un fichier txt et mettre tous les IP à tester. Puis faire une boucle afin de tous les tester et d'afficher le résultat en code couleur pour le vert veut dire que l'imprimante ping et le rouge veut dire que l'imprimante ne ping pas.

2^{ème} script :

```
Import-Module ActiveDirectory

$Users = $(Import-Csv -Path 'C:\Users\hmadmin-ext\Desktop\TCL_SYTRAL_users_2025-01-08_12-06.csv' -Encoding UTF8 -Delimiter ';').loginEmail

ForEach ($User In $Users) {
    $mail = $(Get-ADUser -Filter "UserPrincipalName -like '$User'" -Properties mail).mail
    $extensionAttribute14 = $(Get-ADUser -Filter "UserPrincipalName -like '$User'" -Properties extensionAttribute14).extensionAttribute14
    If ($extensionAttribute14 -Eq "RATPDEV") {
        Add-Content -Value "$User;$extensionAttribute14;$mail" -Path 'C:\Users\hmadmin-ext\Desktop\exportRATPDEV.txt'
    }
}
```

Le second script a pour but de vérifier parmi une liste d'utilisateur de savoir si leur attribut 14 était égal à « RATPDEV » et si oui de créer un nouveau fichier avec tous les noms utilisateur, leur mail et leurs attribut 14.

3^{ème} script :

```
<#
.SYNOPSIS
Script who test connection computers
.DESCRIPTION
this script will do a necessary computer connection test
.INPUTS

.OUTPUTS

.NOTES
Nom du script : Test ping
Auteur : Teo Panei
Usage : Test de connection d'une machine ou d'un ensemble de machine
Version : 0.3
Révisions :
    - 0.3 : ajout autre script
    - 0.2 : Ajout de la fonction Ping
    - 0.1 : Version Initiale
#>

#-----[Initialisations]-----
#-----[Declarations]-----
#-----[Functions]-----

0 references
Function ping {
    $IP = Read-Host " Ip de la machine : "
    if(Test-Connection -ComputerName $IP -count 1 -ErrorAction SilentlyContinue){
        Write-Host "$IP is up " -ForegroundColor Green
    }
    else{
        Write-Host "$IP is down" -ForegroundColor Red
    }
}
```

Pour commencer dans ce script j'ai appris la mise en page pour qu'il soit clair car un script n'est pas uniquement pour soi il est également pour les autres et c'est pour cela qu'il doit être compréhensible pour autrui

Pour cela l'idéal est de faire un en-tête du script avec plusieurs paramètres comme, un synopsis ou l'on résume le but du script, puis la description du script ou bien des commandes qu'on va exploiter dans ce script-là, ensuite les prérequis pour utiliser le script et enfin quelques notes comme la version etc.

Le script va se découper en plusieurs sections : initialisations, déclaration, fonction et exécution,

Pour la section « initialisation », c'est le début du script c'est toutes les commandes nécessaires pour le bon fonctionnement du script comme si on veut importer certains modules.

Pour la section « déclaration », comme son nom l'indique si dans le script on déclare plusieurs variables différentes on peut les regrouper toutes dans cette section.

Pour la section « fonction », c'est l'endroit où toutes les fonctions utilisées dans le script sont regroupées

Pour la section exécution », c'est la partie du script qui va s'exécuter on peut également mettre des commentaires pour que ça soit plus clair.

```
#-----[Executions]-----

#Test-connecting according a list of computer
$IP = Get-Content "names.txt"

foreach($IP in $names){
    if(Test-Connection -ComputerName $IP -count 1 -ErrorAction SilentlyContinue){
        Write-Host "$IP is up " -ForegroundColor Green
    }
    else{
        Write-Host "$IP is down" -ForegroundColor Red
    }
}

#script version with add-content and set-content
$IP = Get-Content "IPPrinter.txt"

Set-Content -Path 'C:\Users\teo.panei\Desktop\ExportIPPrinter.csv' -Value 'Printer Name;Status' -Encoding UTF8 -Force

foreach ($ip in $IP) {
    if (Test-Connection -ComputerName $ip -Count 1 -ErrorAction SilentlyContinue) {
        $status = "UP"
    } else {
        $status = "DOWN"
    }
    Add-Content -Path 'C:\Users\teo.panei\Desktop\ExportIPPrinter.csv' -Value "$ip;$status"
}
```

Au niveau du script c'est le but que le premier mais avec une autre liste d'imprimantes mais également avec d'autre commande comme **set-content** qui permet de fixer un en tête dans le nouveau fichier

Et pour la fonction ping cela permet de vérifier le ping de l'IP qu'on fournit avec la commande **Read-host**

4^{ème} script :

Pour ce script j'ai suivi la même mise en page que pour le script précédent.

```
<#
.SYNOPSIS
Script who create csv with necessary attributes
.DESCRIPTION
this script will search in the Active Directory for the necessary attributes.
.INPUTS

.OUTPUTS
Generate CSV intern at : C:\Users\teo.panei\Desktop\ExportRD LyonvUsersInterne.csv
Generate CSV extern at : C:\Users\teo.panei\Desktop\ExportRD LyonvUserExterne.csv
.NOTES
Name : Research Attributes in Active Directory
Auteur : Teo Panei
Usage : create un csv avec les attributs souhaité
Version : 0.6
Révisions :
- 0.6 : Modification de la fonction UsersAttribut
| | Ajout d'autre version de script dans Executions
- 0.5 : Ajout de la fonction ADAttribut
- 0.4 : Ajout de la fonction UsersAttribut
- 0.3 : Modification de la condition If/Else
| | Ajout du Block try/catch
- 0.2 : Ajout de la condition If
- 0.1 : Version Initiale
#>
#-----[Initialisations]-----

#import necessary modules
Import-Module ActiveDirectory

#-----[Declarations]-----
```

Dans la section initialisation, j'ai mis le module que je voulais importer dans ce la cas c'est le module Activedirectory.

```
#-----[Functions]-----

0 references
function UsersAttribut {
    Import-Module ActiveDirectory

    $NameUser = Read-Host "Nom de l'utilisateur : "

    $attribut = [PSCustomObject]@{
        Name= (Get-ADUser -Filter { Name -eq $NameUser } -Properties Name).Name
        mail= (Get-ADUser -Filter { Name -eq $NameUser } -Properties Mail).Mail
        EmployeeID= (Get-ADUser -Filter { Name -eq $NameUser } -Properties EmployeeID).EmployeeID
        EmployeeType = (Get-ADUser -Filter { Name -eq $NameUser } -Properties EmployeeType).EmployeeType
    }

    $attribut | Out-GridView -Title "Informations de l'utilisateur"
}

0 references
function ADAttribut {
    Import-Module ActiveDirectory

    $Users = Get-ADUser -Filter * -Properties Manager, extensionAttribute14, comment

    if ($Users) {
        foreach ($User in $Users) {
            $Manager = $User.Manager
            $extensionAttribute14 = $User.extensionAttribute14
            $Name = $User.Name
            $comment= $User.comment

            if ($extensionAttribute14 -eq "RATPDEV") {
                Add-Content -Value "$Name ; $extensionAttribute14; $Manager; $comment" -Path 'C:\Users\teo.panei\Desktop\exportRATPDEV2.csv'
            }
        }
    }
}
```

J'ai transformé chaque version de mes scripts en fonctions pour pouvoir les réutiliser.

```
#autre version de la fonction ADAttribut
0 references
function ADAttribut {
    try {
        foreach ($User in $Users) {
            $Manager = (Get-ADUser -Identity $User -Properties Manager).Manager
            $extensionAttribute14 = (Get-ADUser -Identity $User -Properties extensionAttribute14).extensionAttribute14
            $Name = (Get-ADUser -Identity $User -Properties Name).Name
            $comment= (Get-ADUser -Identity $User -Properties comment).comment

            if ($extensionAttribute14 -eq "RATPDEV" -and $Manager) {
                $Manager = (Get-ADUser -Identity $Manager -Properties Name).Name
                Add-Content -Value "$Name ; $extensionAttribute14; $Manager; $comment" -Path 'C:\Users\teo.panei\Desktop\exportRATPDEV3.csv'
            }
        }
    } catch {
        Write-Warning "il y a une erreur !!! : $_ "
    }
}
```

```
#autre version avec Out-GridView
0 references
function ADAttribut {
    Import-Module ActiveDirectory

    $attribut = @()

    $Users = Get-ADUser -Filter * -Properties Manager, extensionAttribute14, Name, Comment

    foreach ($User in $Users) {
        $Manager = $User.Manager
        $extensionAttribute14 = $User.extensionAttribute14
        $Name = $User.Name
        $Comment= $User.Comment

        if ($extensionAttribute14 -eq "RATPDEV" -and $Manager) {
            # $Manager = (Get-ADUser -Identity $User.Manager -Properties Name).Name
            $attribut += [PSCustomObject]@{
                Name = $Name
                extensionAttribute14 = $extensionAttribute14
                Manager = $Manager
                Comment = $Comment
            }
        }
    }

    if ($attribut.Count -gt 0) {
        $attribut | Out-GridView -Title "Utilisateurs RATPDEV"
    }
}
```



```
#Research for Attributs Users in AD with Out-GridView
try {
    $attribut = @()

    $Users = Get-ADUser -Filter * -Properties Manager, extensionAttribute14, Name, Comment

    foreach ($User in $Users) {
        $Manager = $User.Manager
        $extensionAttribute14 = $User.extensionAttribute14
        $Name = $User.Name
        $Comment = $User.Comment

        if ($extensionAttribute14 -eq "RATPDEV" -and $Manager) {
            #Manager = (Get-ADUser -Identity $User.Manager -Properties Name).Name
            $attribut += [PSCustomObject]@{
                Name = $Name
                extensionAttribute14 = $extensionAttribute14
                Manager = $Manager
                Comment = $Comment
            }
        }
    }

    if ($attribut.Count -gt 0) {
        $attribut | Out-GridView -Title "Utilisateurs RATPDEV"
    }
}
catch {
    Write-Warning "il y a une erreur !!! : $_ "
}
```

```
#-----[Executions]-----
#search for attributes according to csv
$Users = $(Import-Csv -Path 'C:\Users\teo.panei\Desktop\rdlyon 1 - Copie.csv').Name

Try{
    foreach ($User in $Users){
        $Mail = $(Get-ADUser -Filter "Name -like '$User'" -Properties Mail).Mail
        $EmployeeID = $(Get-ADUser -Filter "Name -like '$User'" -Properties EmployeeID).EmployeeID
        $EmployeeType = $(Get-ADUser -Filter "Name -like '$User'" -Properties EmployeeType).EmployeeType

        if ($EmployeeType -eq "Actif normal"){
            Add-Content -Value "$User;$Mail;$EmployeeID" -Path 'C:\Users\teo.panei\Desktop\ExportRDLYonvUsersInterne.csv'
        }
        Else {
            Add-Content -Value "$User;$Mail;$EmployeeID" -Path 'C:\Users\teo.panei\Desktop\ExportRDLYonvUserExterne.csv'
        }
    }
} catch {
    Write-Warning "il y a une erreur !!! : $_ "
}

#Research for Attributs users in AD
try {
    foreach ($User in $Users) {
        $Manager = (Get-ADUser -Identity $User -Properties Manager).Manager
        $extensionAttribute14 = (Get-ADUser -Identity $User -Properties extensionAttribute14).extensionAttribute14
        $Name = (Get-ADUser -Identity $User -Properties Name).Name
        $comment = (Get-ADUser -Identity $User -Properties comment).comment

        if ($extensionAttribute14 -eq "RATPDEV" -and $Manager) {
            $Manager = (Get-ADUser -Identity $Manager -Properties Name).Name
            Add-Content -Value "$Name ; $extensionAttribute14; $Manager; $comment" -Path 'C:\Users\teo.panei\Desktop\exportRATPDEV3.csv'
        }
    }
}
catch {
    Write-Warning "il y a une erreur !!! : $_ "
}
```

Pour ces différentes versions de script j'ai utilisé différentes commandes notamment **add-content**, qui permet de créer un nouveau fichier et d'importer les résultats souhaités mais aussi la commande **try** et **catch**, qui permet de vérifier si ton script a une erreur et si le cas se présente stop le script et ça affiche un message d'erreur avec la commande **write-warning**, la commande **out-**

gridview, qui permet de créer une fenêtre d’affichage avec les résultat souhaité au lieu de créer un nouveau fichier.

III. Conclusions

A. Apport de stage

Ce stage m’a beaucoup apporté que ce soit au niveau des compétences professionnelles ou encore des compétences humaines.

Mais plus précisément au niveau des compétence professionnelles, des commandes PowerShell, et plein d’autres choses qui me seront utile pour ma vie professionnelle. De plus cela m’a permis d’avancer dans mon projet PowerShell à présenter a l’oral en fin d’année pour l’obtention de mon BTS SIO

Ce qui est pour les compétence humaines, j’ai déjà appris à m’organiser dans mes idées lors de cette résolution de script mais aussi à avoir une autre manière de réflexion, de recherche également pour penser à toutes les éventualités.

Pour m’avoir appris tous ces compétences et ce savoir, je remercie ce stage.

Annexe 1 – Mon poste de Travail



Annexe 3 : Prise de note sur Ansible

Ansible Comprendre Ansible

- ↳ Permet l'Automatisation des déploiement
 - éviter l'humaine
 - (+) les processus de déploiement cohérent et fiable
- ↳ Gestion des Configurat° → permet l'intégrité et la stabilité
- ↳ Orchestration
- ↳ Gestion des Evénements

Composants clés

- Rôle
- Playbooks → fichier en format YAML
- Templates → Jinja

Playbook

Arborescence de base d'un playbook Ansible

Arborescence

```

graph TD
    Project[Projet - ansible] --> Inventory[inventory]
    Inventory --> Host[host]
    Host --> Group[group - vars]
    Group --> All[all.yml]
    Group --> Web[web server.yml]
    Host --> HostVars[host - vars]
    HostVars --> Server1[server 1.yml]
    HostVars --> Server2[server 2.yml]
    Host --> Files[files]
    Files --> Sample[sample - file.txt]
    Host --> Templates[templates]
    Templates --> Service[service.j2]
  
```

Booles

- ↳ with - items → standard
- ↳ with - nested → complexe
- ↳ with - dict → dictionnaire

ansiblen.bolten.

Modules

- ↳ service
- ↳ package
- ↳ apt
- ↳ raw / command / shell
- ↳ import
 - ↳ host
 - ↳ playbook
 - ↳ role

↳ Template

↳ Jinja2