

*Architecture matérielle et applicative*

*// Cobol*

// Gartner Group (5 couches)

|  |  |
| --- | --- |
| Présentation | la **présentation** des données, correspondant à l'affichage, la restitution sur le poste de travail, le dialogue avec l'utilisateur (OUTPUT) |
| Traitement / Métier | le **traitement** métier des données, correspondant à la mise en œuvre de l'ensemble des règles de gestion et de la logique applicative |
| Accès aux données | l'**accès aux données** persistantes : correspondant aux données qui sont destinées à être conservées sur la durée, voire de manière définitive. |

Définition client serveur: Le client fait des demandes et le serveur y répond à l’aide d’un moyen de communication (réseau en informatique).

Il n’y a pas d’architecture classique, on va l’adapter aux besoins.

Interface la plus utilisée = client web (couche présentation)

-



= En architecture informatique, un **middleware**(anglicisme) ou intergiciel est un logiciel tiers qui crée un réseau d'échange d'informations entre différentes applications informatiques.

utilisé aujourd’hui pour se connecter à des bases de données (mysql, merci brubru, postgre, etc. ne gèrent pas de la même façon la manière de se connecter) ex: jdbc, odbc.

<https://fr.wikipedia.org/wiki/Novell_NetWare> -Marie(16/09/2017,

Le dessin en couches = **Urbanisation**

* Métier/Stratégie : décision
* Fonctionnelle
* Applicative
* Infrastructure Technique : matériel qui répond au besoin

Le **système d'information** (**SI**) est un ensemble organisé de ressources qui permet de collecter, stocker, traiter et distribuer de l'information (@Wiki)

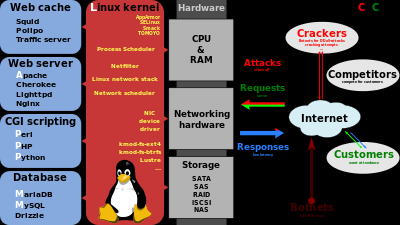
Système d’informations === entreprise / organisation

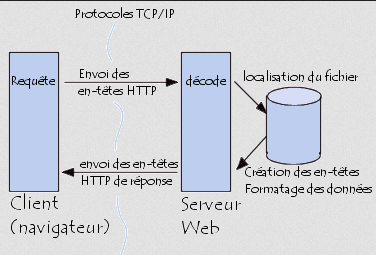
⇒ Acteurs (rôles) + Processus (fonctions) + Données (informations)

ex: DRH, Comptable

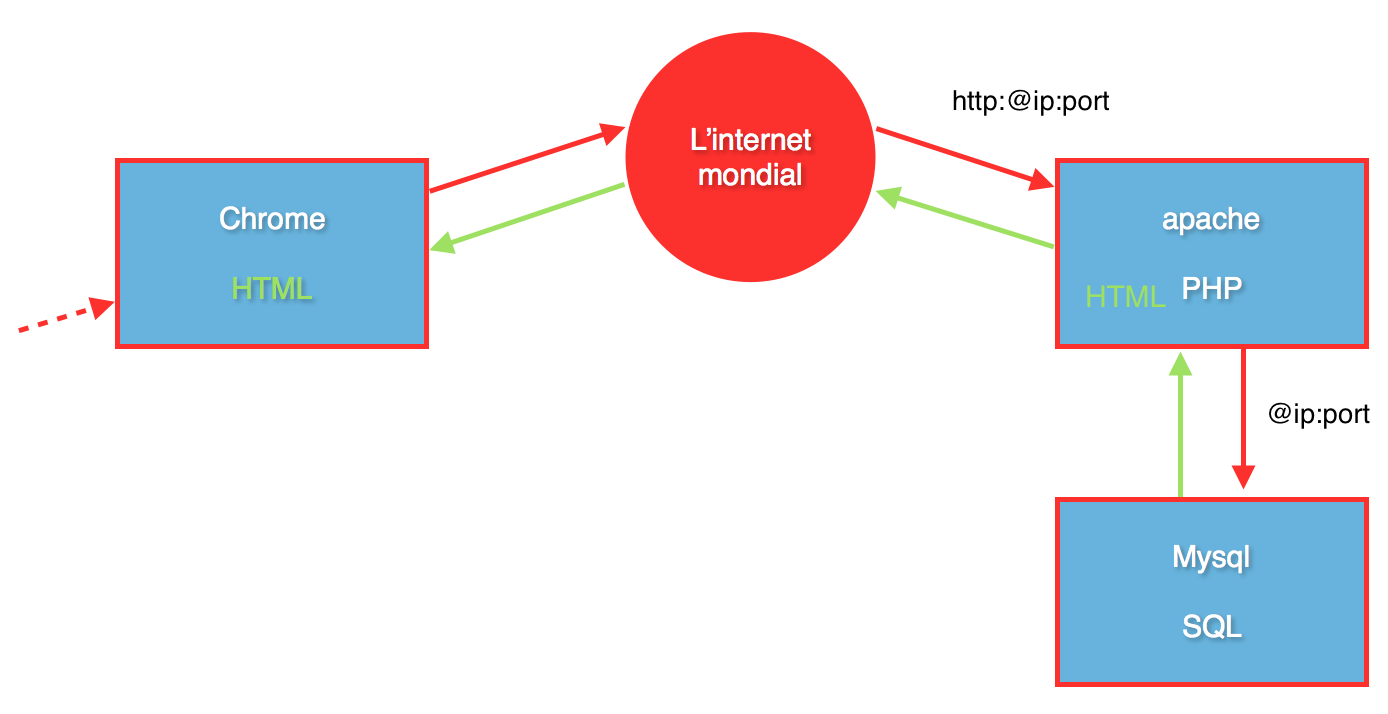
Cours : “allez-y molo avec les supports”

* <http://igm.univ-mlv.fr/~dr/XPOSE2001/perrot/Intro-Comparatif.htm>
* <https://fr.slideshare.net/HeithemAbbes1/architectures-ntiers>
* <https://www.google.fr/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0ahUKEwiohcTxnanWAhVFK1AKHfFdAxsQFgg3MAI&url=http%3A%2F%2Fdeptinfo.cnam.fr%2Fnew%2Fspip.php%3Fpdoc5259&usg=AFQjCNFYfyc3Y-Lh4FVp7FWQusC7yhJwNw>

+



Le joli dessin de M. le prof :



<http://getbootstrap.com/> (à utiliser pour créer un site)

MCD -> MLD -> MPD -> SQL

annuaire pour stocker les utilisateurs (active directory: annuaire microsoft)

**LDAP** (protocole permettant l'interrogation et la modification des services d'annuaire)

⇒ gestion des authentifications

Garder une table user

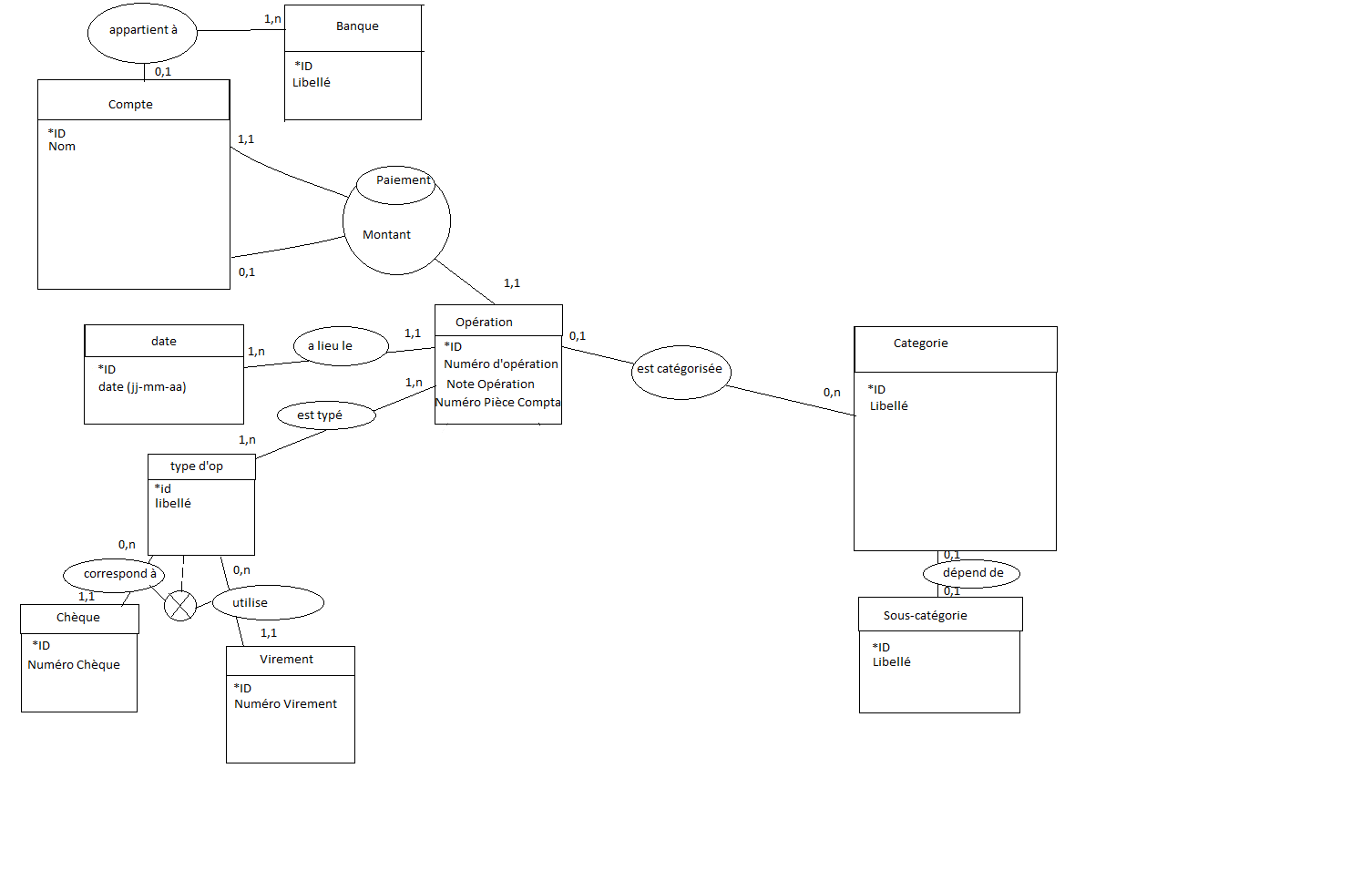
Gestion des authentifications avec un bitmask

trésorier 32 : 0 0 1 0 0 0 0 0

admin 128 : 1 0 0 0 0 0 0 0

trésorier + admin 160 : 1 0 1 0 0 0 0 0

Si vous voulez voir un MCD de pgm:



Profil / Rôle / Droits

|  |  |  |
| --- | --- | --- |
| **A P A C H E** | **P**  **H**  **P** | privée  page1.php  page2.php **[2]**  page5.php **[4]**  page6.php **[2] [4]**  publique  page3.php  page4.php |

**[2]** Profil 1

**[4]** Profil 2

Grands mécanismes de gestion des droits pour un site web :

phase 1 - authentification : qui suis-je?

phase 2 - autorisation : suis-je autorisé?

* login est unique

Session: login: [joel.heinis@uha.fr](mailto:joel.heinis@uha.fr) / profil: 6

1. Authentification

select login from users as u

where u.login= “[joel.heinis@uha.fr](mailto:joel.heinis@uha.fr)”

and u.mdp=”jojo”;

résultat :

mdp ou login incorrect → 0

mdp et login correct →1

2) Autorisation:

select login, **profil** from users as u

where u.login= “[joel.heinis@uha.fr](mailto:joel.heinis@uha.fr)”

and u.mdp=”jojo”;

1/ Connexion à la bdd :

* authentification
* autorisation

driver: mySQLi / PDO (PHP Data Objects)

**PHP Data Objects** (**PDO**) est une extension définissant l'interface pour accéder à une [base de données](https://fr.wikipedia.org/wiki/Base_de_donn%C3%A9es) avec [PHP](https://fr.wikipedia.org/wiki/PHP:_Hypertext_Preprocessor).

<?php

$host\_name = "localhost";

$user\_name = "user1";

$password = "root";

$database = "bdd1";

$connect = mysqli\_connect($host\_name, $user\_name, $password, $database);

if(mysqli\_connect\_errno())

{

echo '<p>La connexion au serveur MySQL a échoué: '.mysqli\_connect\_error().'</p>';

}

else

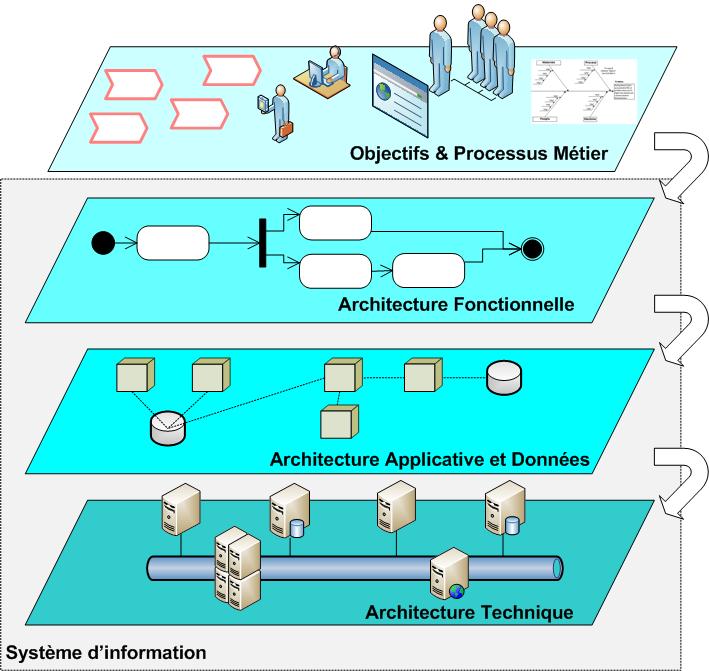
{

echo '<p>Connexion au serveur MySQL établie avec succès.</p>';

}

?>

avec pdo: <https://www.w3schools.com/php/php_mysql_select.asp>



rappels : Le dessin en couches = **Urbanisation**

* Métier/Stratégie : décision
* Fonctionnelle
* Applicative
* Infrastructure Technique : matériel qui répond au besoin

Il faut utiliser un singleton pour être sûr de l’instance de connexion (constructeur privé avec new à l’intérieur de la classe)

* Un attribut statique qui conservera l'instance unique de notre classe
* Une méthode statique qui permet d'instancier l'objet ou de récupérer l'instance unique déjà créée
* Un constructeur privé si on souhaite empêcher l'instanciation en dehors de la classe

*// c.class.php - singleton*  
class C {  
 private function \_\_construct() {

//connexion à la BDD ici

}  
 private static $c;

static function getInstance() {  
 if (isset($this->c)) return $this->c;  
 return $this->c;  
 }  
  
}

*// p1.php*  
<?php

require(c.class.php);

$conn = c::getInstance();

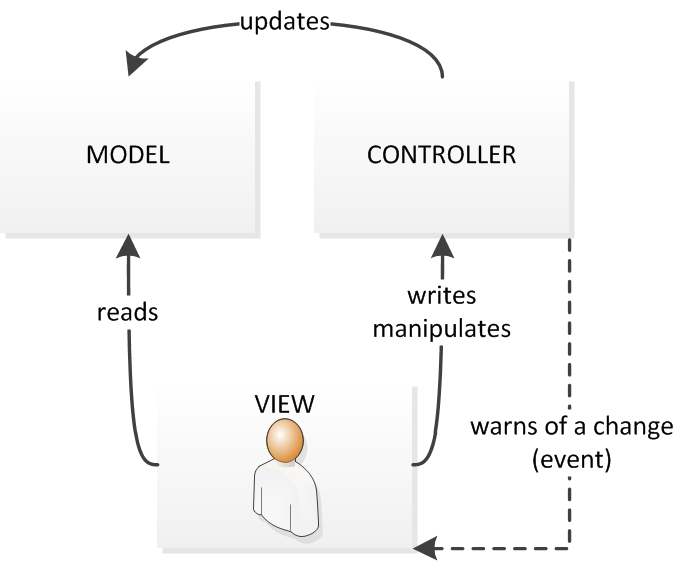
...

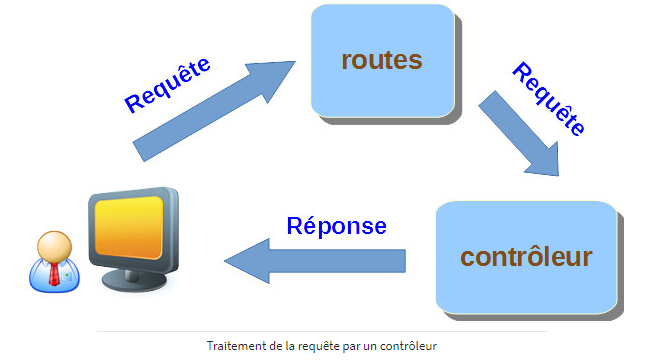
truc.fr -> Router → defaultController → defaultView

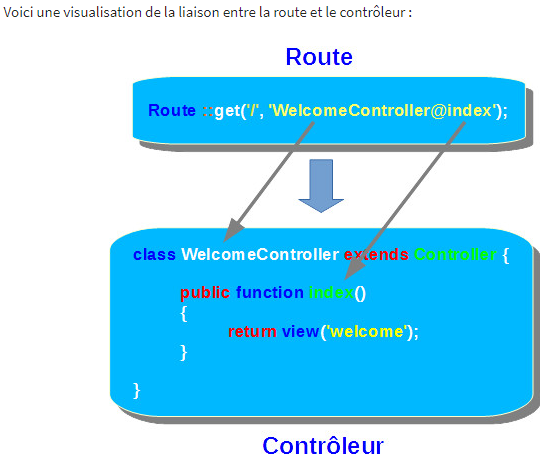
On cherche à s’authentifier :

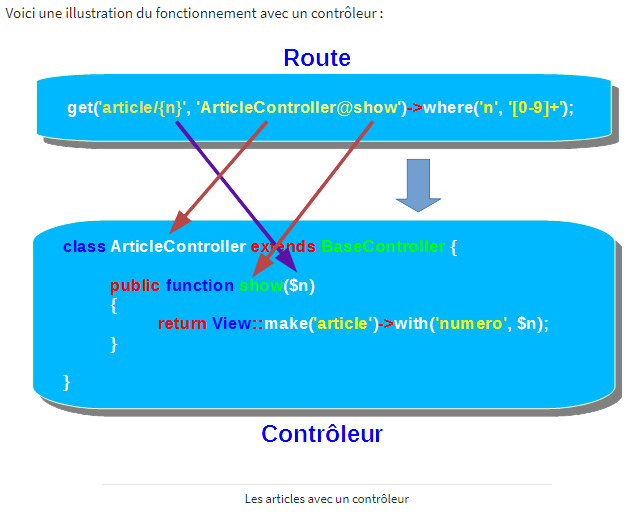
truc.fr/controller/user/login.php → router appelle → userController (avec méthode login, logout) →

Modèle mvc:









Factorisation: apporte de la lisibilité

lib -> classes de base

Technologie d’écriture d’URL pour Apache : htaccess

Autre serveur web : Microsoft IIS (Internet Information Services)

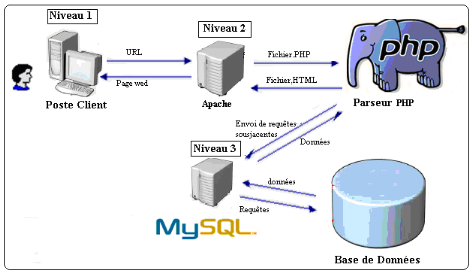
* routage = important
* réécriture d’url (apache)

**gestion des applications web:**

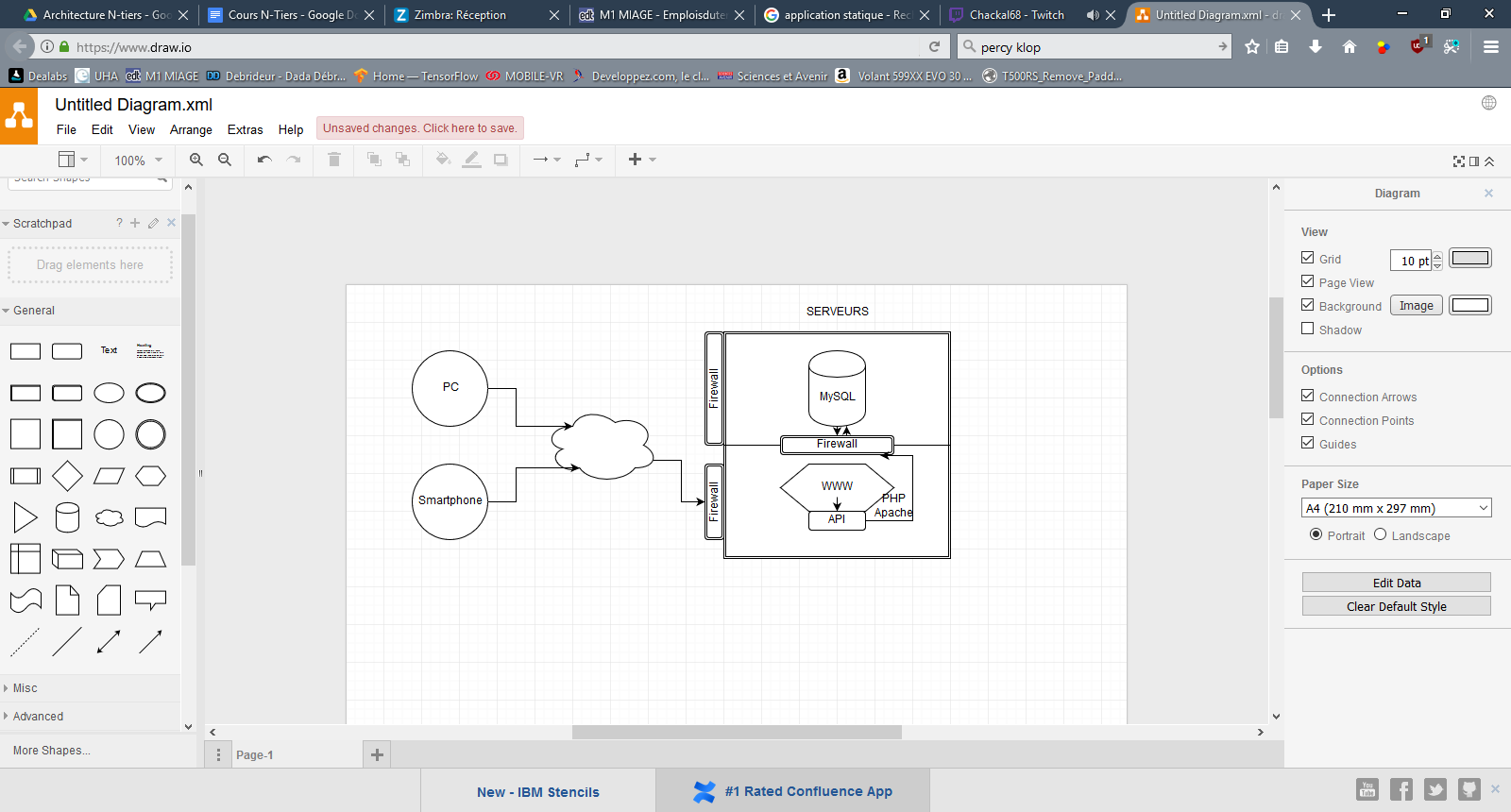
**-** Application mobiles (smartphone)

**-** Application web statique : c’est une application où l’utilisateur peut juste consulter ces services sont avoir la possibilité d'interagir avec application , exemple site vitrine.

**schéma qui illustre l'envoie d'une requête sql:**

****

**Schéma d’une application mobile/web de type blablacar :**



système d’information (SI): <https://fr.wikipedia.org/wiki/Syst%C3%A8me_d%27information>

Pour l'examen:

\_ revoir définitions

\_ savoir la définition des acronymes (MySQL, AJAX, VPN, proxy, DMZ…)

\_ requêtes sql

\_ structure globale HTML

\_ méthode POST

La sécurité (des données) est primordiale dans le web.



000000000000000000000000000000000000SOAP : standard de communication pour les webservices (<https://fr.wikipedia.org/wiki/SOAP>)

Correction Partiel 2017:

Dossier 1 :

1. Créer la base + importer les données
2. Modifier les identifiants de connexions à la BDD
3. Supprimer le dossier CAS

**Cours du 18/11/2017**

On continue à développer l’application MVC

**cf. W:\HEINIS\_archiweb\archiweb\_td\_mvc\_class**

Détails sur l’utilisation de bootstrap pour les messages (alert, warning,...)

**Travail à faire :** Vérifier user/pwd d’un utilisateur via un WebService



**exemple prof**

controller: public logincheck($u,$p);

model: private logincheckDB($u,$p);

model: private logincheckWS($u,$p);

**Examen final :**

soit sur feuille = support à réviser qu’il va envoyé

soit sur PC = des (5 ou 6) bout de code avec problématiques à résoudre + question de cours

* + j’vous donne un WS, appelé-le
  + faites la page qui envoie le JSON
  + du SQL aussi

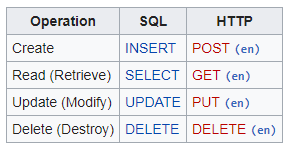
**choix** : à donner le plus rapidement possible

WSDL : <https://www.w3schools.com/xml/xml_wsdl.asp>

**Exemple SOAP** sur le drive : “exemple\_soap”

→ c/c le dossier [testsoap] dans votre [.../wamp/www]

CRUD => Create Read Update Delete



Pour supprimer l’utilisateur dont l’id est 12 :

[Méthode=Delete] <http://localhost/user/12>

|  |  |
| --- | --- |
| Service | |
| SOAP | REST |
| Exemple client  $c = new SOAPClient(‘file.wsdl’);  $c->checkLogin($u,$p); | Exemple client  url(url) |