(15)

UNIVERSITÉ DE HAUTE ALSACE - MULHOUSE

Faculté de : FST

Année d'études (cocher la case correspondant à votre situation)

L1 ☐ précisez .....

L2 ☐ précisez .....

L3 ☐ précisez .....

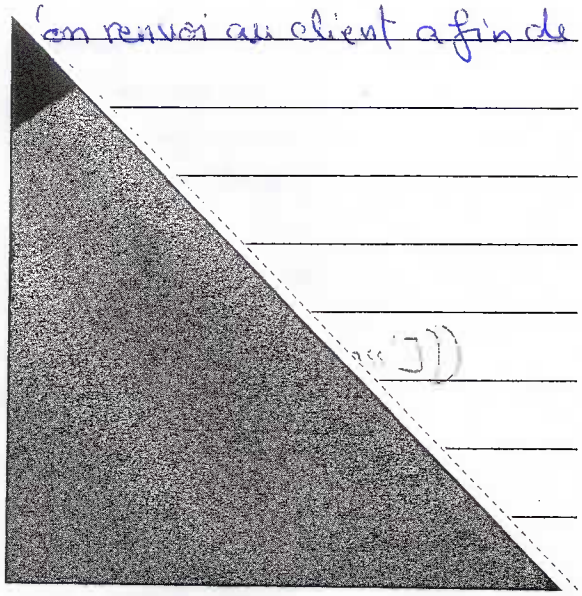
M1 ☒ précisez MIAGE

M2 ☐ précisez .....

Composition de : Architecture N-tiers

Sujet proposé par : .....

Date de l'Examen : ~~Architecture N-tiers~~ 16/02/19

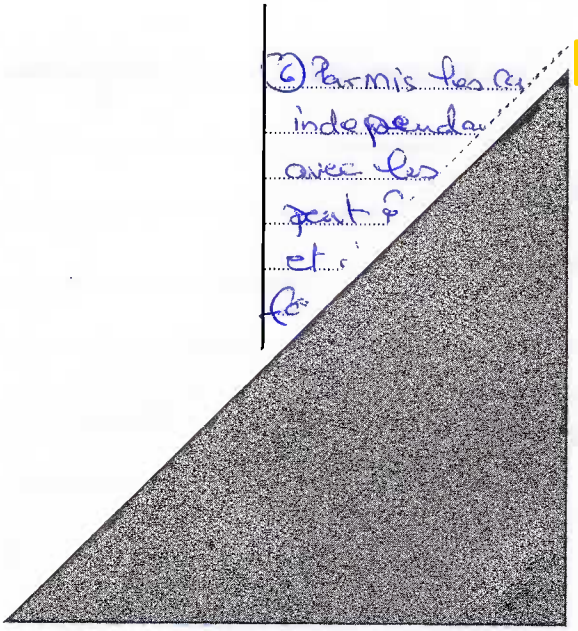



| Note de la composition | Appréciation et signature du correcteur : | N° ./. |
|---|---|---|
| | | |

① un système d'information est un ensemble de ressource et processus qui permettent de collecter, stock, analyser, et diffuser les données (& l'information)

② CI : Continious Integration consiste à faire des mise à jour dans l'application et succedes les Teste (TU, TI, TF)
CD : Continous Deployement : consiste à faire du déploiement dans immediat après les mise à jour apporté dans la phase CI

③ lorsqu'on parle de SI urbanisé on le compare à une ville du qu'à l'origine est venu avec la notion d'urbanisme qui consister sur l'evolution de la ville en fonction des nouveaux besoin
et l'analogie de la ville n'est pas seulement dans l'evolution mais aussi dans la structuration (secteurs) qui permet de bien localiser où on a le besoin



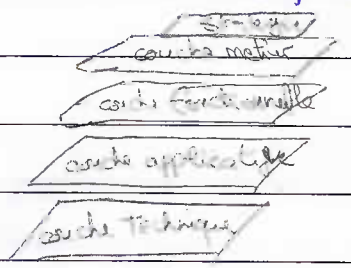
(6) Parmis les av... independa... avec les ... peut ê... et ... fo...

(4) SOA : Service Oriented Architecture

Consiste à concevoir son système d'information sur une architecture qui délivre des application fournissant des services

(8) Le middleware permet d'inter-connecter entre les applications, l'intérêt c'est de resoudre le problème lier au système hétérogènes et propriétaire et aussi resoud le problème de distances qui peut avoir entres les applications.

(7) Un SI urbanisé peut être représenté comme suit :




Cette représentation est une cartographie, elle permet de représenter le SI sous forme de block afin de cibler les intervenants et les fonctionnalités liés à chaque niveau dans le but de l'optimisation des performances et d'assurer l'évolution du SI dans chaque niveau

(5) L'utilisateur envoi une requet http au serveur web

- la requet etant composé du protocol // DNS : port / c - de parlement - liste
- le serveur web interprete grace a PhpEngine la 1er page Index d'il va rencontré. dans cette purge qui sera une sorte de routeur ou dispatcher va recuperer les parametre de la requette pour determiné le controleur à utilisé et quelle methode (controller/controller_departement.php)
- une fois qu'on accede au controleur et on execute la methode list() on va être redirigé vers la vue qui comporte la liste des departement (../view/departement/v_listall.php)

• et c'est cette vue (HTML) qu'on renvoi au client afin de la visualise sur son navigateur •

page_accueil.php

```
<?php
    if(isset($_POST['Identifiant'])&& (isset($_POST['Pass']))
    { $Id = $_POST['Identifiant'];
      if($Id == 'LOGIN')
        { $Pass = isset($_POST['Pass'];
          if($Pass == 'Password')
          { redirecte('page_connecté.php?L='.$Id) ; }
        else{echo "Mot de passe incorrecte" ;}
      }else
      { echo "identifiant incorrect"
?>

<form method='POST' action="" >
    <input name='Id' type='text'>
    <input name='Pass' type='text'>
</form>
```

Page_connecte.php

```
<?php.
if(isset($_POST[L]))
{echo 'Bonjour'.$_POST[L];}
?>
```

⑥ Parmis les caractéristiques d'un microservice c'est qu'il est independant dans la mesure ou il a un faible couplage avec les autre composants (autre microservices) comme ça il peut être evolutif independamment du reste de toute l'application et il est orienté metier, c'est qu'il regroupe toute les ~~fonctio~~ fonctionnalité relié à un metier donnée ; ex un micro service dedié pour la gestion du panier (dans un site e-commerce) on peut egalement les ~~dupliqué~~ dupliquer en fonction de la charge vu qui peut être representé ~~sous~~ dans des containers (type Dockers) qu'on peut ajouter ou supprimé, et la gestion de la charge peut etre assuré par ESB (entreprise Service Bus) qui permet egalement l'interconnection entre eux
L'hétérogeinité des composants permet de beneficier de la force lié au faible de tirer profit de l'ensemble de technologie utilisé dans chaque MS distinctement, contrairement au application monolithe qui consiste à avoir un seul block developpé dans un seul langage. Parti au niveau maintenance c'est ~~fa~~ facile d'operer dans un seule langage mais le probleme qui va se poser c'est de gerer les fonctionnalité distinctement d'ou la force des MS

⑨ Le principe de la modularité : consiste à distinguer les fonctionnalité sous forme de bloc (ex : MS) afin d'essayer d'être d'avoir des elements les plus independant possible tout en gardant un inter connexion transparente

La resistience : consiste à resistence au changement qui peuvent avoir lieu en cas par exemple de cession de l'entreprise a une autre qui a plus de charge en utilisant les service : (n'est pas impacté par le volume)