Informaticien

Concours externe 2017

MEILLEURES COPIES

Spécialité "Informaticien d'applications"

ASSEMBLÉE NATIONALE
Service des Ressources humaines

ASSEMBLÉE NATIONALE

SOMMAIRE

ASSEMBLÉE NATIONALE

000009

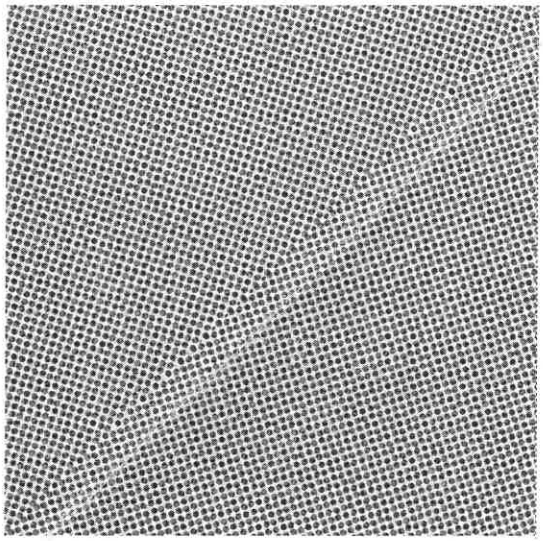
NOTE : 18 /20

Concours : Concours d'Informaticien

Épreuve : Questionnaire commun

Copie n° : 1 / 3

I. On attend d'un gestionnaire de versions les services suivants : la gestion de l'historique des modifications apportées à un fichier, et la possibilité de résoudre d'éventuels conflits qui peuvent survenir lorsque plusieurs personnes modifient le même fichier simultanément.
Un autre service fourni par un gestionnaire de ~~services~~ versions est la possibilité de gérer plusieurs branches d'un même projet, indépendantes les unes des autres et qu'il est possible de fusionner à loisirs, sans réserve de résolution des potentiels conflits (par exemple une branche développement et une branche production)
Un exemple de gestionnaire de versions est git, qui possède la particularité d'être décentralisé. tout le répertoire versionné peut-être présent sur différents postes de travail.



Pour travailler à plusieurs sur un ~~fichier~~ projet comportant plusieurs fichiers, une option consiste à suivre le flot git standard : une branche maîtresse stable en permanence et une multitude de branches pour les évolutions et les correctifs. Ces branches sont ensuite fusionnées dans une branche de développement, qui après des tests et une recette sera fusionnée à son tour dans la branche maîtresse, éventuellement assortie d'une étiquette marquant une nouvelle version.

II Les cinq opérations de base pour manipuler un modèle relationnel sont : SELECT, INSERT, UPDATE, DELETE et JOIN.
L'opération SELECT permet de sélectionner l'ensemble des lignes d'une base qui répond aux arguments de la requête, par exemple un filtre sur une valeur.

L'opération INSERT permet d'ajouter un

enregistrement dans une table de la base.

L'opération UPDATE permet de mettre à jour les enregistrements d'une table qui correspondent à la requête aux paramètres de la requête.

L'opération DELETE permet de supprimer les enregistrements d'une table qui correspondent aux paramètres de la requête.

Enfin, l'opération JOIN permet d'effectuer une jointure entre deux tables, c'est à dire de comparer des données qui n'appartiennent pas à la même relation. Cette opération est utilisée en complément des opérations ci-dessus.

Le language principal reposant sur le modèle relationnel est SQL: Structured Query Language. Il est Turing-complet.

III. Réaliser une analyse de risques d'un système avant sa conception présente plusieurs avantages :

Cela peut permettre de détecter des failles de sécurité qui devront être corrigées dès la conception du système, plutôt que durant le développement ou même plus tard.

Cela permet également d'anticiper les futures difficultés auxquelles seront confrontés les projets (tels que le temps de réponse lors d'une montée en charge, ou la défaillance d'une base de données). En conséquence, le système pourra être modifié lors de sa conception pour tenir compte des ces risques (équilibrage de charge et redondance, par exemple).

Enfin, cela permet de sensibiliser les équipes techniques ou de pilotage sur certains risques, permettant d'allouer plus de temps ou de ressources à des éléments dont on avait minimisé l'importance, tels que la confidentialité ou le chiffrement des données utilisateurs ou la durée de rétention des journaux d'enregistrement de l'application.

ASSEMBLÉE NATIONALE

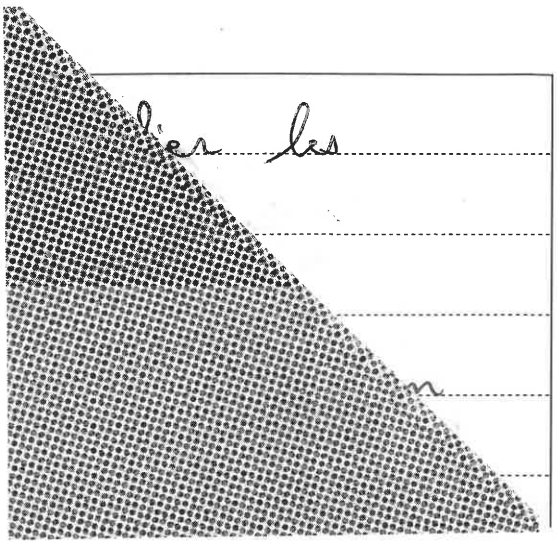
000009

NOTE : ..... /20

Concours : Concours d'Informaticien

Épreuve : Questionnaire commun

Copie n° : 2 / 3



IV. Le service DNS est utilisé pour convertir un nom de domaine en adresse IP.
La base du système est composée de 13 serveurs racines, répartis partout dans le monde et nommés selon des lettres de l'alphabet.
Ces serveurs racines sont consultés pour connaître le gestionnaire d'une zone de noms de domaines, en partant du top level domaine du domaine (.org ou .fr).
La demande de résolution est ensuite redirigée vers les serveurs de l'organisme qui gère la zone DNS concernée (les registrars). Ces organismes peuvent à leur tour rediriger les requêtes vers les serveurs de leur client qui gère le nom de domaine concerné, qui enverra la réponse finale

entière, tan
les spéci
du
l'
T

au demandeur à partir d'un table de correspondance entre noms de domaines et adresses IP (il s'agit de la zone DNS sur laquel ce serveur a autorité').

Les enregistrements les plus demandés sont bien sûrs les adresses, ~~m~~ (champ A) mais également les serveurs de courriel (MX) et des champs testes variés (TXT) qui peuvent contenir des informations diverses (SPF par exemple).

Comme on a pu le voir plus haut, une requête DNS nécessite un grand nombre d'allers et retours entre différentes autorités (serveurs racines, registrars, clients). Pour améliorer les performances un système de cache est donc fortement utilisé, mais il induit une faille.
En effet, les enregistrements DNS n'étant pas authentifiés, il est possible de corrompre un cache en le faisant correspondre à une mauvaise adresse IP. C'est le principe de l'empoisonnement du cache, auquel DNSSEC est censé répondre.

en ajoutant un moyen d'authentifier les requêtes DNS.

V. Le cycle en V est une méthode de gestion de projet dans laquelle les étapes sont définies à l'avance et n'évoluent que très peu : analyse du besoin, spécifications, développement, recette et livraison.
La méthode Scrum, elle, est basée sur le concept de sprint, à durée fixée (de l'ordre de 15 jours) pour chaque projet. Au cours de ce sprint on effectue les ~~mêmes~~ opérations de spécifications et de développement, ainsi que la recette. Idéalement, chaque sprint s'achève sur une livraison, de taille plus ou moins modeste.

En matière de cycle de vie, il est défini avant le commencement du projet dans le cas d'un cycle en V, alors qu'il est beaucoup plus flou, voir infini dans le cadre de Scrum.

Les spécifications d'un projet pour un cycle en V font l'objet d'une étape à part

entière, tandis que pour un projet SCRUM, les spécifications peuvent évoluer au cours du projet. C'est d'ailleurs de là que l'appelation Agile prend tout son sens. Par conséquent, les changements de spécifications sont plus simples à réaliser dans le cadre d'un projet SCRUM.

Pour ce qui est du contrôle qualité, il évolue en permanence dans le cadre d'un projet SCRUM pour correspondre aux developpements réalisés au cours du sprint. En revanche il est moins flexible dans le cadre d'un cycle en V, puisque le cahier de recette est souvent réalisé en parallèle des spécifications.

Enfin pour ce qui est de la planification, elle est également plus flexible avec un projet SCRUM, puisqu'elle dépend des autres paramètres du projet, et notamment les spécifications. Ce n'est pas le cas dans un cycle en V puisque le cycle de vie est défini à l'avance.

ASSEMBLÉE NATIONALE

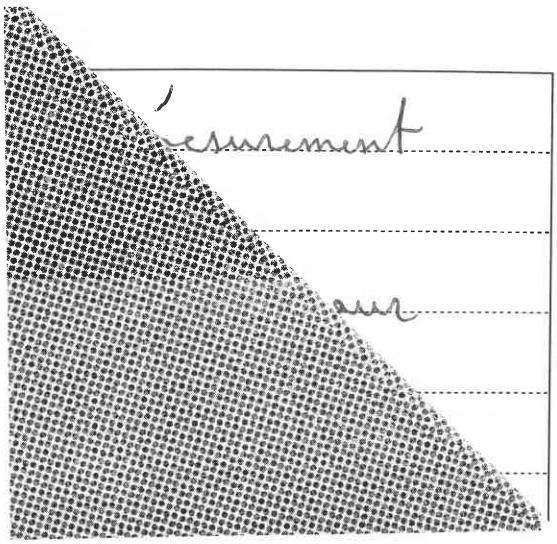
000009

NOTE : ..... /20

Concours : Concours d'Informaticien

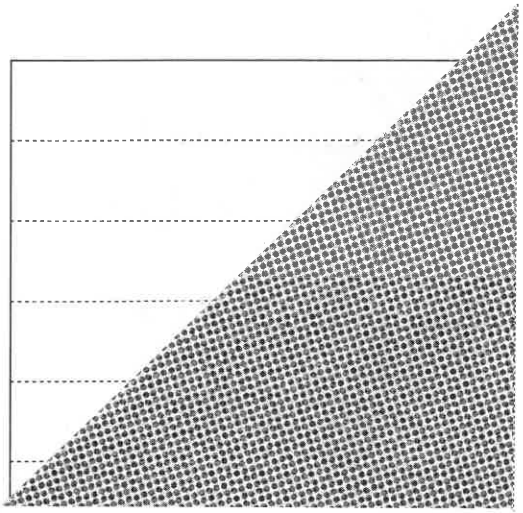
Épreuve : Questionnaire commun

Copie n° : 3 / 3



En conclusion, c'est la nature d'un projet qui fixe le choix de sa méthode de gestion : s'il s'agit d'un projet avec des besoins bien exprimés et peu susceptibles d'évoluer, le cycle en V est parfaitement adapté. En revanche, si le projet nécessite une adaptation constante à un besoin qui évolue rapidement, la méthode Scrum convient très bien.

VI En peut déduire de cette capture que la machine a été démarrée très récemment (uptime de 12 minutes). Apache et MySQL occupent la majeure partie des ressources systèmes, il est donc peu probable que d'autres applications tournent également sur cette machine.



Cette machine est probablement administrée par SSH puisqu'un daemon sshd tourne. Un seul utilisateur est connecté, celui qui effectue la capture.

La charge a fortement augmenté au cours des 15 dernières minutes, passant de 0,95 à 1,92. Puisqu'il y a deux coeurs, la machine est en train d'arriver aux limites des ses capacités CPU (2 normalement). Cette charge est vraisemblablement dûe à MySQL qui utilise 3 minutes 54 de temps CPU, ainsi qu'à Apache, dont le total de temps CPU excède 5 minutes. On peut déduire du nombre de processus Apache que ce dernier est probablement configuré en mode mpm-prefork.

Cette machine est administrée par systemd et non pas par sysinit.

La mémoire libre est correcte, bien moins utilisée que le CPU et ~~principalement~~ à 25% par du cache.

La mémoire SWAP paraît démesurément élevée.

La majorité du temps CPU est utilisé pour des opérations de type USER, il faudra donc songer si les performances sont faibles à modifier

ASSEMBLÉE NATIONALE

000138

NOTE : 17 /20

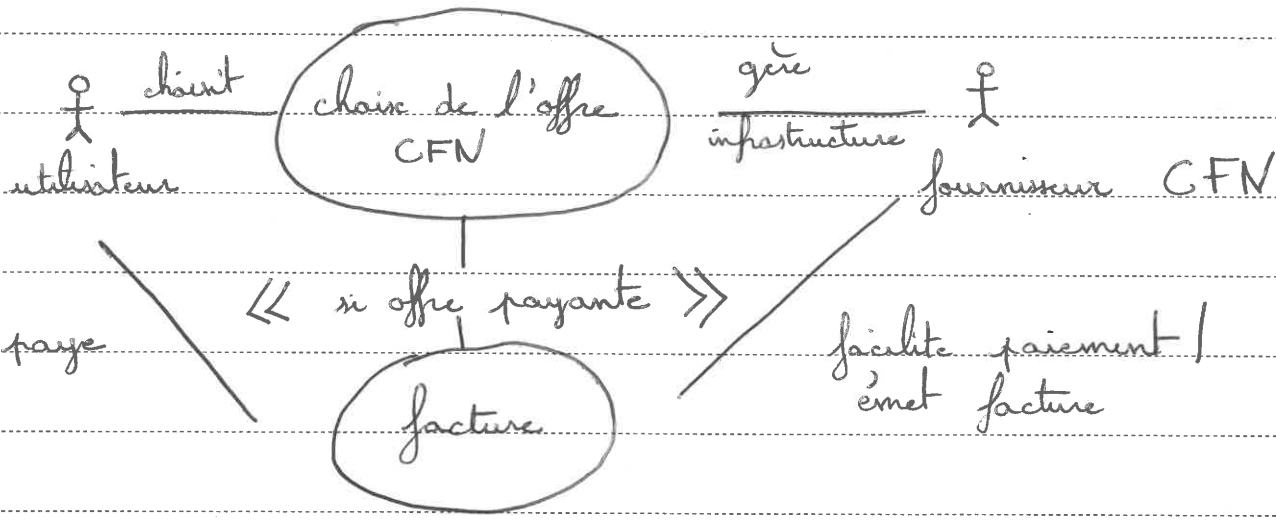
Concours : Concours d'Informaticien

Épreuve : Etude de cas (application)

Copie n° : 1 / 5

I.1 Les principales fonctionnalités du système de CFN sont :

- créer un CFN auprès d'un fournisseur



Cas nominal : l'utilisateur choisit une offre payante, crée un compte et est facturé du montant dû

Cas alternatif : l'utilisateur choisit une offre gratuite, il n'est pas facturé mais il peut créer son compte

Cas erroné : aucune offre ne convient à l'utilisateur, il ne crée pas de compte.

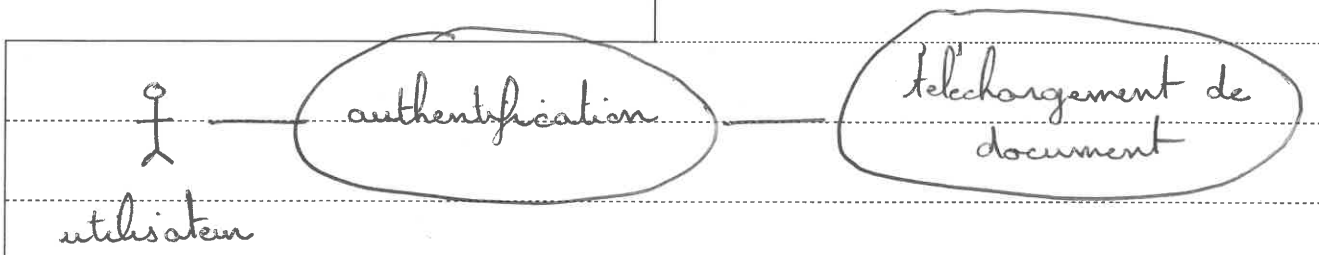
000138

I. 2.

Uti

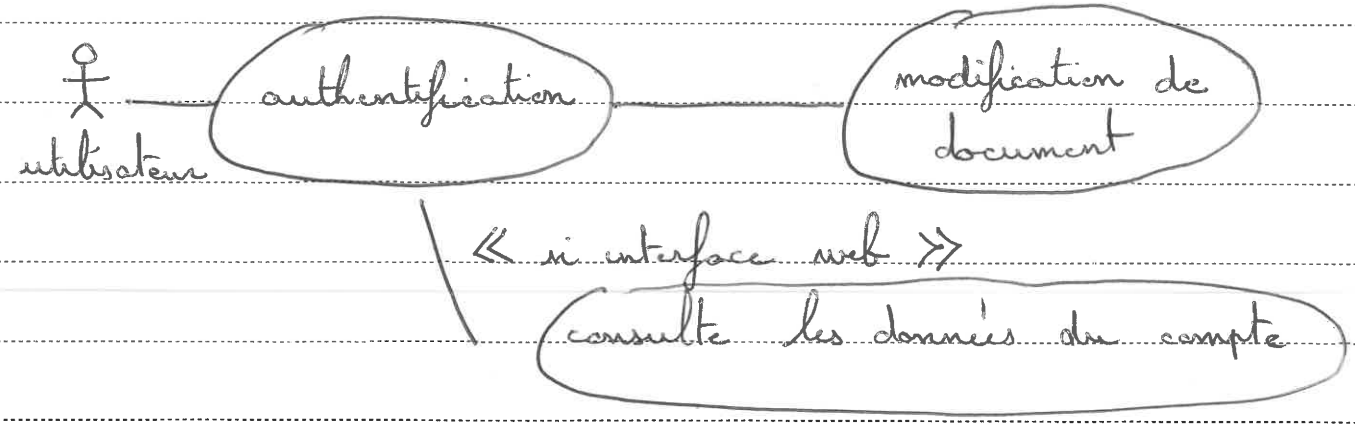
000138

— Stocker des documents en ligne de manière sécurisée dans le CFN.



Cas nominal : l'utilisateur s'identifie via l'interface Web et upload un document
Cas alternatif : l'utilisateur s'authentifie via l'application mobile, puis upload un document
Cas erroné : l'utilisateur échoue à l'identification.

— Accéder aux documents stockés, les modifier ou les supprimer

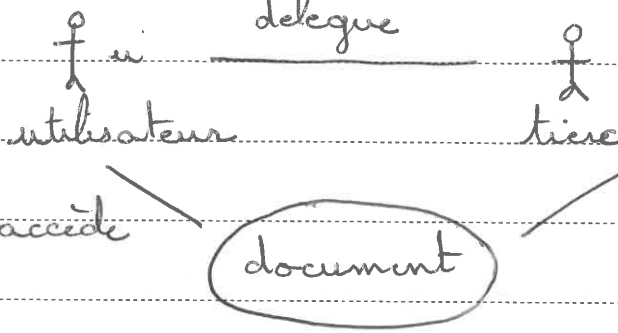


Cas nominal : l'utilisateur s'authentifie via mobile et supprime un document

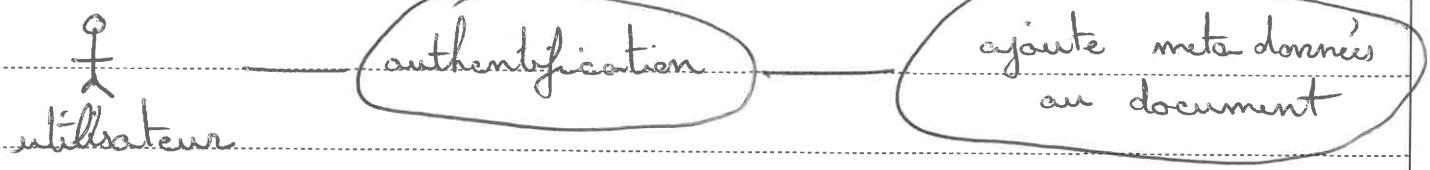
Cas alternatif : l'utilisateur s'identifie via l'interface web, supprime un document et vérifie le gain d'espace disponible.

Cas erroné : l'utilisateur échoue lors de l'authentification.

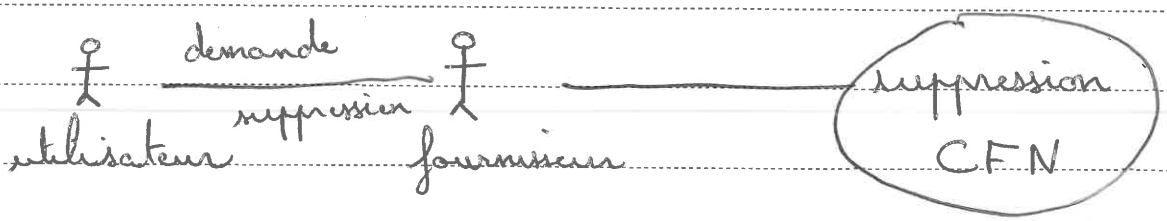
— déléguer des droits



— ajouter des meta données aux documents



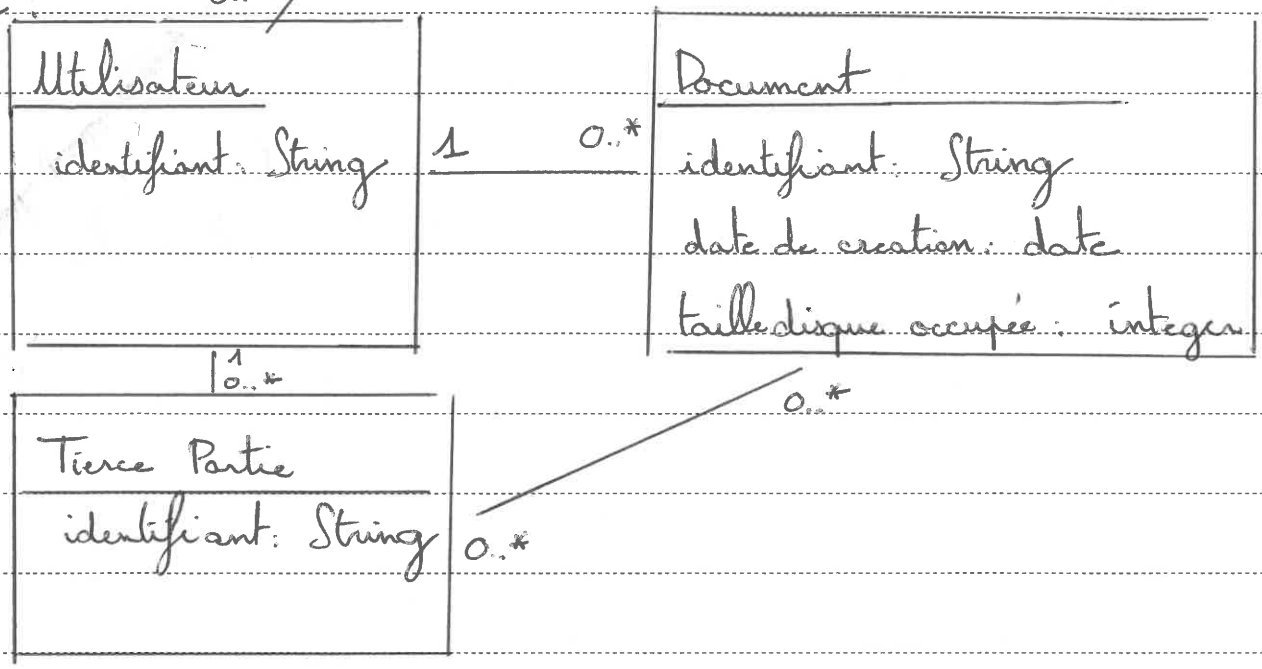
— supprimer le CFN de l'utilisateur



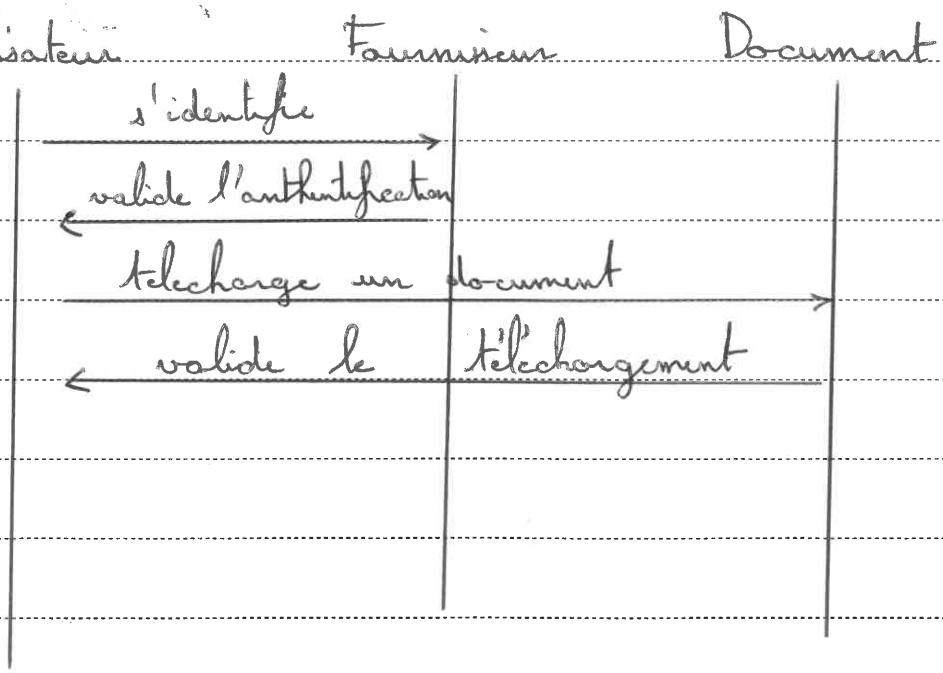
000138

1 | Fournisseur | nom : String | 1..* | 1 | Offres | nom : String

I. 2



I. 3 Scénario nominal d'upload de document



ASSEMBLÉE NATIONALE

000138

NOTE : ..... /20

Concours : Concours d'Informaticien

Épreuve : Etude de cas (application)

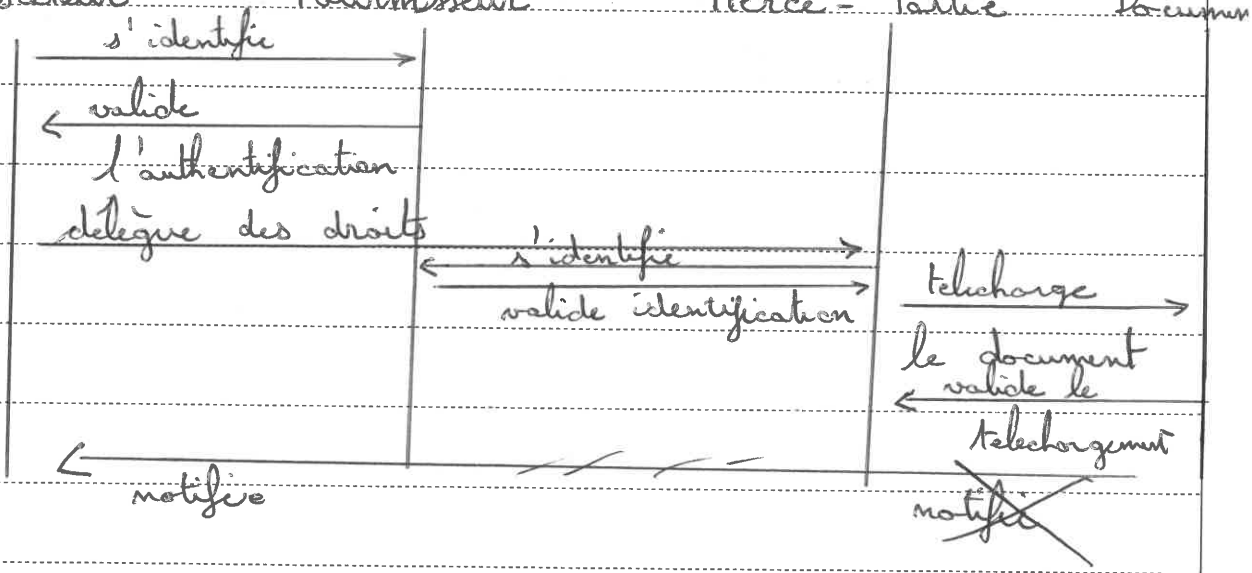
Copie n° : 2 / 5

...s de dépots

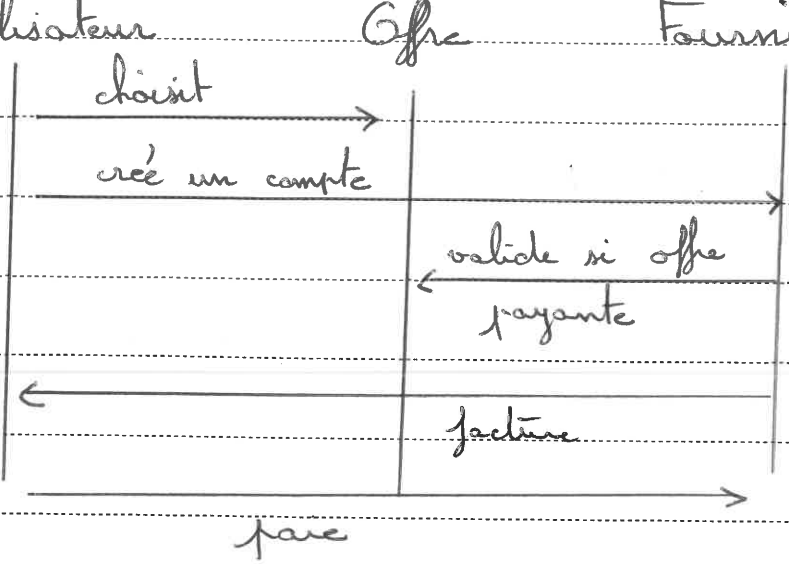
...ystème de

...en

I.3 Scénario nominal de délégation de droits



Scénario nominal de création de compte

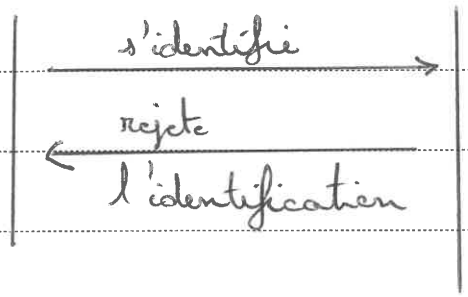


000138

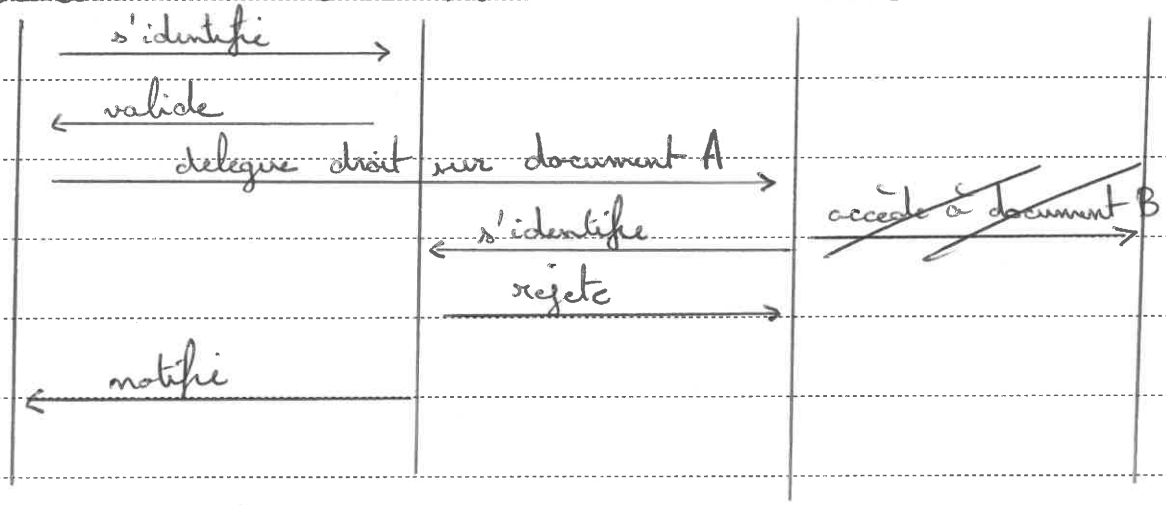
II 1 La disp
mettre en
pour q
pas l
de

000138

Scénario erroné téléchargement de document



Scénario erroné delegation de droits



I 4.

1. Le DP de l'héritage est adapté aux classeurs et sous-classeurs puisque ces éléments auront un fonctionnement similaire qui sera géré dans une classe maitresse tandis que les spécificités le seront dans la classe enfant.

2. La composition est adaptée aux modes de dépots très divers : on peut envisager un système de dépôt global qui va déléguer à une entité spécifique la gestion du document en fonction du type de dépôt choisi

3.

4. Le DP listener est utilisable dans le cadre d'une alarme : il écoute en permanence et peut se déclencher lorsqu'un événement spécifique survient (par exemple la modification d'un document)

000138

II 1 La disponibilité d'un système implique de mettre en place des systèmes de redondance pour que la défaillance d'un composant n'entraîne pas l'arrêt complet du système. Cela implique de dupliquer notamment les serveurs webs qui serviront l'application, idéalement derrière un équilibreur de charge en haute disponibilité. Il en va de même pour les bases de données, qui devront suivre un schéma de réplication maître - esclave. Enfin, les serveurs d'application devront pouvoir se mettre à l'échelle de manière horizontale sans que la perte de l'un d'eux n'induise de pertes de données.
Pour l'intégrité des données, il est possible d'utiliser un système de hashage qui va permettre de valider que les données n'ont pas été altérées lors de leur transfert sur le réseau ou leur dernière utilisation. On pourra notamment employer la fonction de hashage SHA 256.

En ce qui concerne la confidentialité, il est utile de mettre en place un chiffrement de bout en bout à l'aide d'une clef symétrique de type AES qui changera régulièrement et de manière

ASSEMBLÉE NATIONALE

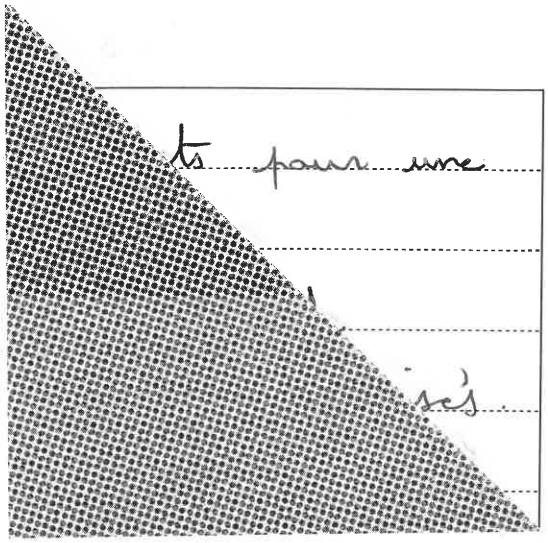
000138

NOTE : ..... /20

Concours : Concours d'Informaticien

Épreuve : Étude de cas (application)

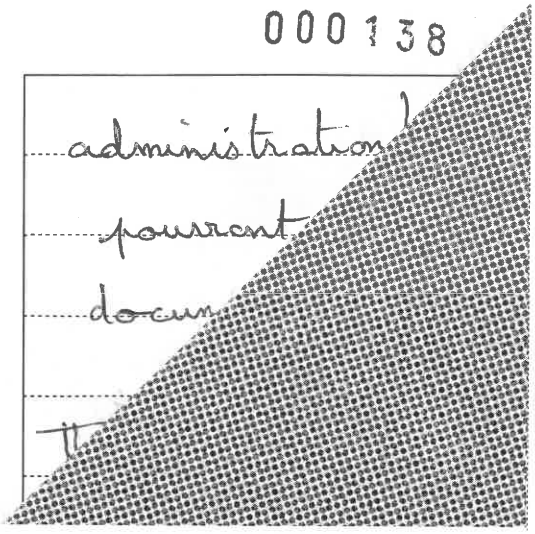
Copie n° : 3 / 5



transparente pour l'utilisateur.
Enfin, pour assurer une traçabilité optimale, il faudra prévoir un système de logs hors de l'applicatif (type rsyslog) de manière à ce qu'une machine compromise ne puisse pas modifier ses propres enregistrements. Cela apporte également le bénéfice de centraliser la gestion et la rotation des journaux.

II.2. La connection au site doit se faire à l'aide du protocole HTTPS, idéalement ~~à l'a~~ par l'intermédiaire d'un certificat ECDSA plutôt que RSA. La mise en place d'une redirection HSTS paraît indispensable pour rediriger toute connexion HTTP vers du HTTPS. On peut également prévoir une authentification par certificat client.

000138



000138

administration
pourront
docu
Il

II 3 Idéalement les utilisateurs doivent être gérés en propre par le système. Bien que l'option Single Sign On puisse être intéressante pour les utilisateurs, elle représente un risque majeur dans le cas où le compte utilisé pour l'authentification est compromis. Les utilisateurs doivent donc être stockés en base avec les précautions d'usage : mots de passe salés et hashés, utilisation de la fonction bcrypt pour vérifier les identifiants (minimise les risques d'attaque par force brute). Il faut également mettre en place une politique de mots de passe forts (supérieurs à 15 caractères) et une rotation obligatoire fréquente. On peut mettre en place une authentification par certificat client HTTPS ou avec une authentification multi-facteurs avec un jeton matériel (exemple Yubikey). Enfin, il faut prévoir un mécanisme de récupération de mots de passe basés sur des codes à usage unique, ou éventuellement sur téléphone.

II.4 L'autorisation d'accès aux documents pour une tierce partie se fait à l'initiative du propriétaire du document. Cela implique de pouvoir créer des groupes d'utilisateurs autorisés. On peut estimer que chaque détenteur d'un compte du CFN est considéré comme un administrateur de son "royaume" (realm) et qu'il peut inviter ou créer d'autres utilisateurs avec des droits restreints. Pour gérer ces droits, on peut s'inspirer des systèmes UNIX, à savoir qu'un fichier est accessible non seulement par son propriétaire mais également par un groupe d'utilisateurs. Une complexité supplémentaire découle du fait que potentiellement plusieurs groupes distincts peuvent avoir besoin d'accéder au fichier.

On peut donc concevoir une solution basée sur des méta-groupes, ou groupes de groupes : chaque fichier possède un propriétaire et 3 méta-groupes : lecture, modification, suppression. Chacun de ces méta-groupes est constitué d'une liste de groupes que l'on peut modifier (par exemple un groupe comptabilité et un groupe

administration). Ainsi différents groupes d'utilisateurs pourront bénéficier de différents droits sur les documents.

II.5 L'architecture optimale est probablement celle de l'architecture 3 tiers, avec une particularité pour le stockage des fichiers sur disque, plutôt qu'en base.
D'un point de vue matériel, une architecture cloud basée sur des ~~VM de~~ machines virtuelles de grande taille apparaît comme une bonne solution : il est possible de bénéficier de l'élasticité du cloud pour optimiser les coûts, et de nombreux fournisseurs proposent des services spécifiquement dédiés aux fichiers (par exemple S3 sur AWS). S3 intègre nativement les notions de classeur / sous-classeurs et métadonnées, ce qui permettrait de l'intégrer facilement dans l'applicatif.

Pour ce qui est de la pile logicielle, privilégier les logiciels libres permettra également de réduire les coûts, sans sacrifier la sécurité. Un parc homogène de machines Linux CENTOS 7 pourra héberger tant la

ASSEMBLÉE NATIONALE

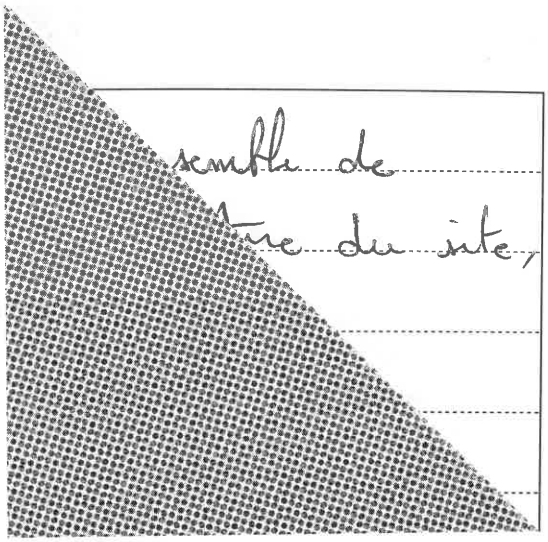
000138

NOTE : ..... /20

Concours : Concours d'Informaticien

Épreuve : Étude de cas (application)

Copie n° : 4 / 5



pile front-end (serveurs web Nginx avec équilibreurs de charge Haproxy et éventuellement un cache Varnish), que les serveurs d'applications en eux-mêmes, probablement programmés en Python pour une plus grande facilité de maintenance. Les bases de données, de type PostgreSQL pourront également tourner sous CentOS 7. Chacun de ces éléments communiquera avec les autres par le biais de canaux sécurisés en TLS, et sera placé dans un VLAN approprié à son usage, avec un pare-feu adapté.
Enfin, les journaux de logs seront centralisés sur une machine isolée du reste du réseau, et redondée également, faisant tourner un deamon rsyslog. L'ensemble des machines sera administré par clef SSH de type ECDSA.

des dossiers s
- Si notre
dispose
do

III. 1 Pour respecter les préconisations de la CNIL il faut ajouter les spécifications suivantes au cahier des charges

- Préciser à l'utilisateur que certains types de données sont à déconseiller (exemple les données de type santé en dehors d'un agrément ministériel spécifique)
- Prévoir le fait que la suppression à la demande d'un utilisateur soit immédiate et que les sauvegardes ne persistent pas plus d'un mois.
- Faire en sorte que le chiffrement soit maitrisé uniquement par l'utilisateur et qu'il respecte les recommandations de l'ANSSI.
- Tracer l'utilisation éventuelle d'une clef de déchiffrement de sauvegarde.
- Prévoir l'augmentation de la taille des clefs pour faire évoluer les protocoles de chiffrement.
- Prévoir de conserver les données sur le territoire de l'UE ou pays aux lois correspondantes
- Prévoir l'authentification des "facteurs ~~numériques~~ électroniques"

— Informer les utilisateurs sur un ensemble de contraintes éventuelles, liés à la fermeture du site, ou au danger d'utilisation des "facteurs électroniques".

III. 2. Un service de CFN doit faire l'objet d'une déclaration à la CNIL, qui précise les données à caractère personnel traitées par le fournisseur du CFN. La CNIL recommande notamment de cibler la finalité des informations (identification et journaux de connexion) ~~not~~, de vérifier leur pertinence, de minimiser leur conservation, de les sécuriser, et de s'assurer qu'elles sont collectées de manière "loyale". Il est également nécessaire de préciser à la CNIL le nom du service qui traitera les données collectées. Ces données sont celles qui permettent aux fournisseurs de faire fonctionner son service.

III. 3 Note aux utilisateurs du CFN.

Conformément aux recommandations de la CNIL, veuillez prendre connaissance des éléments suivants :

— n'ayant pas reçu l'agrément ministériel, nous vous déconseillons de stocker des informations relatives à

des dossiers santé sur ce site.

— Si notre service devait être amené à fermer, vous disposeriez d'un délai de 3 mois pour migrer vos données.

— Nous collectons certaines données personnelles afin d'assurer le bon fonctionnement du site, notamment vos identifiants et horaires de connexion. Ces données sont stockées en France, et vous disposez d'un droit d'accès et de modification de ces données (le service concerné est le service informatique, sous la direction de M. Brassonet)

— Lorsque vous utilisez la fonctionnalité des "facteurs électroniques", vérifiez auprès des tiers concernés que cela n'enfreint pas leurs conditions d'utilisation.

— Si quelqu'un utilise une clef de sauvegarde pour accéder à vos dossiers, vous en serez immédiatement notifié. Cette clef de sauvegarde est confiée à un tiers de confiance.

— Nos fichiers sont chiffrés à l'aide du chiffrement AES 256; tous les accès sont protégés par du TLS 1.2 avec clef ECDSA.

— Vous pouvez résilier votre abonnement à tout moment, et vous disposerez alors de 3 mois pour exporter vos documents.

— Toutes vos données sont stockées en France Métropolitaine

Bonne utilisation !

ASSEMBLÉE NATIONALE

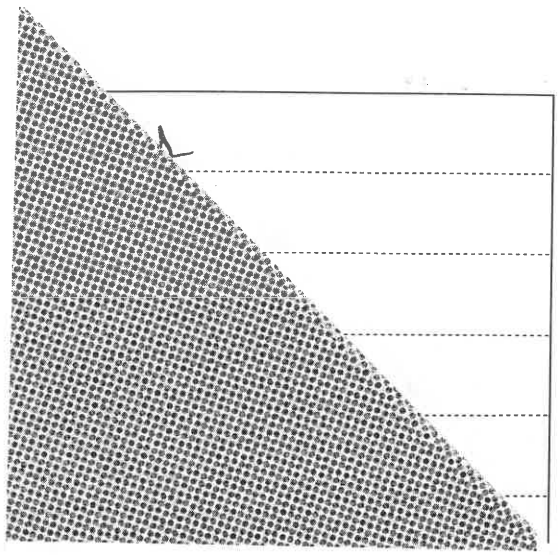
000138

NOTE : ..... /20

Concours : Concours d'Informaticien

Épreuve : Etude de cas (application)

Copie n° : 5 / 5



IV. Note de demande d'approbation du projet CFN :

La demande pour les coffres fort numériques ne cesse de grimper chez les particuliers. La CNIL elle-même le dit dans son rapport du 9 octobre 2013. Nous disposons d'une opportunité unique pour percer sur ce marché. En effet les principaux acteurs sur ce secteur sont basés aux Etats-Unis, et la loi Safe Harbour incite de plus en plus les particuliers à choisir des solutions locales à leur pays, pour mieux protéger leurs données personnelles. Au cours des dernières semaines nous avons conçu et modélisé une architecture pour un projet correspondant à ces critères, basé à 100 % sur des machines de type cloud. Ces machines permettent d'être utilisées de manière élastique, à la demande, ce qui maximise la rentabilité du projet. D'autre part, il est possible de n'utiliser comme

V5.2 : def t...
dossi...

briques d'architecture du projet que des composants issus des logiciels libres (Linux) dont les coûts de licence sont faibles sinon inexistants. Le détail de la pile logicielle est disponible sur demande. Une estimation du budget est la suivante (projet de 6 mois)

* Gestion de projet : 36 K euros (1 CdP)
* Développement : 110K euros (3 développeurs)
* Test et recette : 110K euros (3 testeurs)
* Exploitation/Opérations : 20K euros (cloud AWS)
* Marketing : 10K euros
* Divers : 5K euros

Total : 290 K euros

On peut donc constater que pour un budget relativement restreint (à l'échelle d'une société comme la nôtre), il est possible de viabiliser ce projet en moins d'un an, si l'on parvient à fidéliser 20 000 clients sur notre offre la plus basse (estimée à 5 €) par mois. Ces objectifs sont parfaitement atteignables, à la condition toutefois de ne pas trop tarder à lancer le projet : une telle opportunité ne se présentera

pas 2 fois, et nos compétiteurs en sont pleinement conscients.

V. 1. Soit la structure fichier :

un hashmap ayant pour clef / valeurs :

nom : String

taille : Integer

metadata : MetaData

emplacement : Emplacement

structure dossier :

hashmap ayant pour clef / valeurs :

nom : String

fils : Liste (tableau) de fichiers ou dossiers.

alors arbre est une hashmap avec pour clef / valeurs :

nom : String

files : Dossier

V.2 ~~def taille (Arbre)~~
~~dossiers fichiers = [ ] (liste vide)~~
~~taille = 0~~
~~for each elem in Arbre.fils~~
~~if~~

```
def taille (Arbre)
    dossiers = [ ] (liste vide)
    taille_ = 0
    ~~for each~~
    dossiers . append ( Arbre . fils)
    for each elem in dossiers :
        if elem isinstanceof (Fichier)
            taille_ += elem taille
        else if elem instance of (Dossier)
            dossiers append ( elem . fils)
    return taille
```

V.7 def obtenir (arbre, fichier, utilisateur)
returns Emplacement, boolean

Les solutions proposées au cas où le fichier n'existerait pas consisterait à lever une exception, ou un objet null pour Emplacement. Il en va de même dans le cas où l'utilisateur n'a pas le droit de consulter le fichier.

300006

ASSEMBLÉE NATIONALE

NOTE : 15,5 /20

Concours : Concours externe d'informaticien

Épreuve : Technique - Infor. d'application

Copie n° : 1 / 3

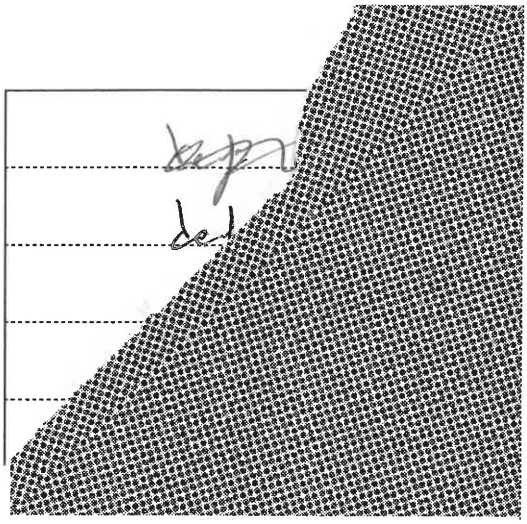
Le langage choisi est Python v3.x

1 - On a une classe "Bibliothèque" qui contiendra les descriptions de documents. Cette liste peut contenir des objets de différentes classes dont chacune correspond à un type de publication (article, book...). Ces classes héritent d'une classe abstraite "Description" qui contiendra les champs éventuellement communs, les méthodes communes et les méthodes obligatoires.

```
import abc
class Bibliotheque():
    def __init__(self):
        self.__listeDesDescriptions = []
        self.__listeDesDescriptionsInverse = []

class Description(metaclass=abc.ABCMeta):
    def __init__(self):
        self.__champs = dict()
```

300006



Pour chaque classe fille de Description, on a un modèle similaire à celui-ci :

```
class Article(Description):
    def __init__(self, author, title,
                 journal, year, eprint,
                 volume=None, number=None,
                 pages=None, month=None):
        super().__init__(self)
        self._champs['author'] = author
        " " ['title'] = title
        " " ['journal'] = journal
        " " ['year'] = year
        " " ['eprint'] = eprint
        " " ['volume'] = volume
        " " ['number'] = number
        " " ['pages'] = pages
        " " ['month'] = month
```

Commentaires :

Dans le constructeur, les champs obligatoires sont des arguments non optionnels.

On a une classe différente pour chaque type de docs :

class Book(Description)
class Inbook(Description)
" Inproceedings(")
" PhdThesis(")

L'initialisation de "_champs" est sur le même modèle également pour chaque classe fille.

300006

Dans toutes les classes, on suppose qu'on dispose de getters et de setters sur le modèle :

```
@property
def nomAttribut(self):
    return self._nomAttribut

@nomAttribut.setter
def nomAttribut(self, v):
    self._nomAttribut = v
```

2- `b = Bibliotheque()`
`iterateur = (v for v in b.listeDesDescriptions)`

3- `def imprimer(self):  # dans classe Bibliotheque`
`    for desc in self.listeDesDescripteurs:`
`        print(desc)`

et dans la classe abstraite "Description", on a une méthode `__str__(self)` qui renvoie une chaîne de caractères, la méthode vérifie dans le champ "_champs" si le champ est défini :

300006

~~def changement(self):~~

```
def __str__(self): # dans classe Documents
    s = ""
    d = self.champs
    for k, v in d.items():
        if v:
            s += "%s = %s\n" % (k, v)
    return s
```

4- Dans la classe "Documents", on a une méthode "chercheMot(mot)" qui recherche si un mot existe dans la description:

```
def chercheMot(self, mot): # dans classe Documents
    ~~result = False~~
    d = self.champs
    for v in d.values():
        if v.index(mot):
            return True
    return False
```

Puis dans "Bibliotheque", on a la méthode chercher:

```
def chercher(self, listeMots):
    result = []
```

300006

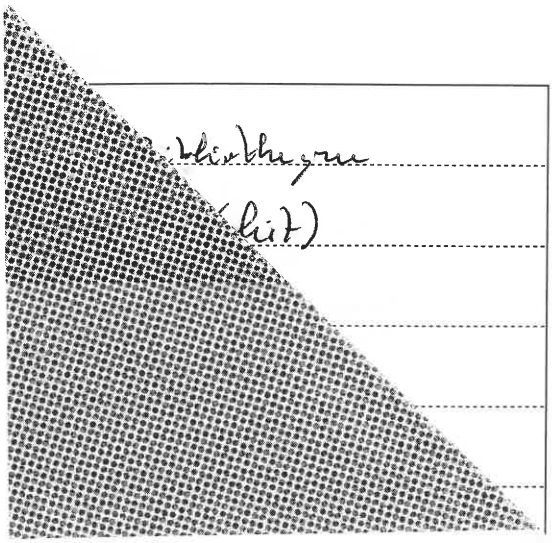
ASSEMBLÉE NATIONALE

NOTE : ..... /20

Concours : Concours externe d'informaticien

Épreuve : Technique - Info. d'applications

Copie n° : 2 / 3



```
for m in liste Mots :
    for desc in self.listeDesDescriptions :
        if desc.cherche(m) :
            result.append(desc)
return result
```

5 - On a deux boucles for, on a donc une complexité égale à longueur(listeMots) x longueur(listeDesDescriptions). Ie on a du O(n x m)

6 - Dans la classe abstraite "Description", on ajoute la méthode "mots" :

300006

9- Sa comp
de hist
nou

```
def mots(self):  # dans classe Description
    lmots = []
    d = self.champs
    for v in d.values():
        l = v.split()
        lmots.extend(l)
    return lmots
```

300006

import collections

7- def listeInverse(self): # dans classe Bibliothèque
    inverse = collections.defaultdict(list)
    for desc in self.listeDescriptions:
        mots = desc.mots()
        for m in mots:
            inverse[m].extend(desc)
    self.listeDesDescriptionsInverse = inverse

8- def chercher(self, listeMots):
    result = []
    for m in listeMots:
        result.extend(self.listeDesDescriptionsInverse[m])
    return result

300006

9- Sa complexité devient $O(n)$ avec n = longueur de liste Mots. Car la recherche dans un dictionnaire (avec table de hash) est de complexité $O(1)$.

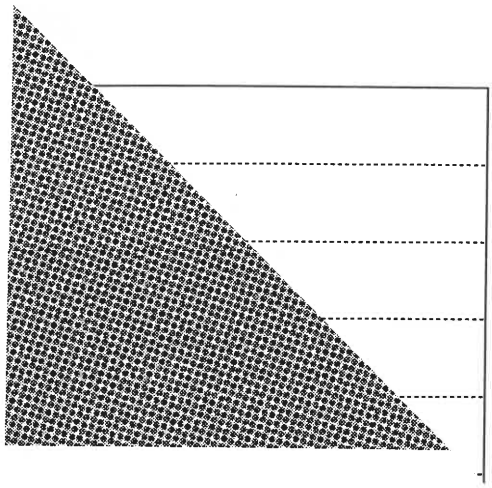
300006

ASSEMBLÉE NATIONALE

NOTE : ..... /20

Concours : Concours externe d'informaticien

Épreuve : Technique - Info. d'application



Copie n° : 3 / 3

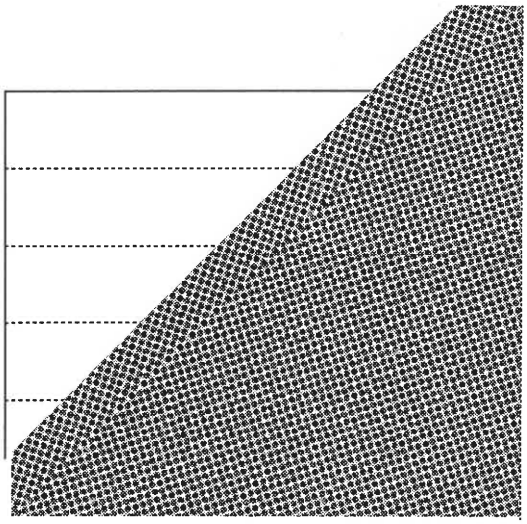
1.0- Les problèmes :

- Obligation de lecture sur disque dont les opérations sont plus lentes.
- Les modifications sont difficiles car pas d'espace prévu. Il faut donc recréer à chaque fois un nouveau fichier texte avec la modification.
- Si un champ contient un saut de ligne, cela pose problème car comment distinguer d'une fin de description

Algo. Dans la ~~précédente~~ fonction chercher() : lire chaque ligne du fichier.

Chercher() est à priori impossible si le fichier n'indique pas le type de publication pour chaque ligne

300006



A noter qu'une solution plus pertinente serait d'utiliser un fichier texte mais avec un format structuré comme du XML ou du JSON.

On pourrait également créer un fichier d'index à part, comme le propose BibTeX je crois.