

E5 : PRODUCTION ET FOURNITURE DE SERVICES

Durée : 4 heures

Coefficient : 5

CAS GEST-PC

*Ce sujet comporte 20 pages dont un dossier documentaire de 10 pages.
Le candidat est invité à vérifier qu'il est en possession d'un sujet complet.*

Aucun matériel ni document autorisé

Ressources documentaires

Description graphique des rôles	11
Extrait du site <i>architectures-web.smile.fr</i> sur les fichiers dans les bases de données	11
Ébauche du schéma relationnel de la base <i>bd_Demandes_PC</i>	12
Script de création de la base <i>bd_Demandes_PC</i>	13
Extrait du tableau récapitulatif des types du secteur protégé	13
Extrait de l'article R. 421-15 du code de l'urbanisme	14
Extrait du formulaire <i>Cerfa n° 13406</i>	14
Interface <i>web</i> avant et après amélioration	16
Code PHP faisant appel à la fonction	16
Extrait du fichier XML existant	17
Code PHP de la fonction <i>liste_déroulante (\$fichier_XML)</i>	17
Extrait du fichier XML modifié	17
Parcours d'une collection en <i>Java</i>	18
Utilisateurs de la base de données	18
Diagramme de classes simplifié du paquetage personnes	18
Extrait des classes du paquetage personnes	19

Barème

Mission 1 : 15 points
Mission 2 : 40 points
Mission 3 : 20 points
Mission 4 : 25 points

CODE EPREUVE : SIE5SL	EXAMEN : BREVET DE TECHNICIEN SUPÉRIEUR	SPECIALITE : SERVICES INFORMATIQUES AUX ORGANISATIONS Parcours SLAM
Session 2013	SUJET	EPREUVE : E5-PRODUCTION ET FOURNITURES DE SERVICES INFORMATIQUES
Durée : 4 h	Coefficient : 5	Code sujet : 13SL03
		Page : 1/20

Présentation du contexte

Hessiauville est une commune de 210 000 habitants faisant partie de la communauté d'agglomération d'Hessiauville (CAH) forte de 235 000 habitants. Pour assurer les services publics et leur gestion, la ville et la CAH comportent 34 directions réparties sur plus de 70 sites.

La direction des systèmes d'information et des télécommunications de la communauté d'agglomération d'Hessiauville est chargée d'assurer toutes les prestations informatiques de la ville et de la C.A.H, à savoir des développements internes spécifiques, la recherche et la mise en œuvre de progiciels et le suivi de leur évolution. Cette structure est constituée d'environ 40 personnes organisées en équipes.

Votre fonction

Vous avez rejoint récemment la *direction des systèmes d'information et des télécommunications* de la communauté d'agglomération d'Hessiauville et intégré l'équipe "Projets" qui a pour rôle de faire évoluer les développements internes.

Parmi les projets qu'elle a en charge, votre équipe travaille sur le projet PC (permis de construire) qui répond à une volonté stratégique de la communauté d'améliorer les relations entre le service urbanisme et les habitants. Ce projet doit donner lieu à la réalisation de deux applications :

- *Gest_PC* est réalisée en *Java* : elle permet aux personnes du service administratif d'assurer la gestion des permis de construire ;
- *Web_Permis* est réalisée en *HTML/PHP* : elle propose aux déposants (personnes qui ont demandé un permis de construire) de consulter et de télécharger les pièces relatives à leur demande.

Actuellement une première version de l'application *Gest_PC* est en production et s'appuie sur une base de données qui lui est propre nommée *bd_Gest_PC*. L'application *Web_Permis* est en phase de test et utilise une base de données nommée *bd_Demandes_PC*.

Votre environnement technologique

Une réplique de l'application *Gest_PC* et de son environnement d'exécution sont mis à votre disposition. Les services nécessaires ont été installés sur une machine virtuelle située dans une ferme de serveurs et accessible en interne depuis certains postes de travail. Cette machine virtuelle abrite un serveur *Windows 2008 R2* et les services suivants :

- ✓ un serveur *web Apache* avec module *PHP*,
- ✓ une ébauche de *bd_Demandes_PC* sous *PostgreSQL Server*,
- ✓ une copie de *bd_Gest_PC* sous *PostgreSQL Server*,
- ✓ l'application *Web_Permis*,
- ✓ une copie de l'application *Gest_PC* utilisée en interne.

Vous accédez à la machine virtuelle depuis un poste de travail sous *Windows Seven* avec comme outil l'environnement de développement intégré *Eclipse*.

Votre feuille de route

Lors d'une réunion avec l'équipe informatique, un dossier complet vous a été remis. Il permet de situer le contexte et les objectifs du projet PC. En tant que nouveau collaborateur de la *direction des systèmes d'information et des télécommunications* de la communauté d'agglomération d'Hessiauville, vous allez participer à différentes missions liées au projet :

Mission 1 : étude de la mise en production

Mission 2 : validation de la base de données

Mission 3 : amélioration de l'interface web

Mission 4 : évolution de l'application interne *Gest_PC*

Mission 1 : étude de la mise en production

En amont du processus de développement et de test de l'application *Web_Permis*, l'équipe s'intéresse à sa future mise en production.

Cahier des charges de la mission

La communauté d'agglomération d'Hessiauville dispose d'un site *web* divisé en une partie publique accessible à tous les internautes et une partie privée accessible uniquement après authentification. L'application *Web_Permis* qui permet de consulter et de télécharger les pièces d'une demande de permis de construire sera intégrée dans la partie privée du site actuel.

Le processus de traitement d'une demande de permis de construire implique l'intervention de plusieurs acteurs :

- ✓ le déposant qui remplit la demande de permis et fournit les pièces obligatoires ;
- ✓ le service urbanisme de la mairie qui réceptionne la demande de permis et remet au déposant un récépissé de dépôt ;
- ✓ l'architecte conseil de la mairie qui émet un avis sur la demande de permis ;
- ✓ la direction départementale de l'équipement qui a le pouvoir d'annuler un permis de construire accordé.

Pour accéder à la partie privée, l'utilisateur devra s'identifier au moyen d'un *login* et d'un mot de passe. L'accès aux pièces des demandes de permis sera fonction de l'utilisateur authentifié selon les règles de gestion suivantes :

- ✓ un déposant pourra consulter les pièces de ses demandes de permis de construire ;
- ✓ les architectes des déposants pourront consulter les pièces des demandes de permis qu'ils ont en charge ;
- ✓ le service urbanisme de la mairie, y compris l'architecte conseil, doit pouvoir consulter toutes les pièces ;
- ✓ l'architecte conseil de la mairie pourra saisir un avis sur toutes les demandes de permis non encore visées ;
- ✓ la direction départementale de l'équipement pourra consulter les pièces des demandes de permis pour lesquels un avis a été émis.

Pour valider une demande de permis de construire, le déposant doit déposer les pièces obligatoires en mairie auprès du service urbanisme (par exemple la pièce PCMI1 doit être fournie en 6 exemplaires). Ces pièces, d'un format imposant pour certaines, sont numérisées (scannées) par une société extérieure. Tous les vendredis, cette société récupère les pièces des demandes déposées dans la semaine et rend celles emmenées la semaine précédente accompagnées de leur version numérique sur un disque compact. Les pièces au format numérique sont alors enregistrées et conservées sous forme de fichiers dans le système d'information.

Les pièces numérisées et enregistrées devront faire l'objet d'un classement hiérarchique organisé en répertoires. Elles feront l'objet d'une sauvegarde de sécurité.

C'est l'équipe "Production" qui sera chargée d'intégrer ce service dans le site *web* et de mettre en place l'accès aux différentes ressources selon l'utilisateur. L'équipe "Projets" fournira à l'équipe "Production" un dossier de spécifications fonctionnelles contenant :

- la liste des utilisateurs et de leurs habilitations selon un formalisme de leur choix ;
- l'arborescence des répertoires de stockage et les règles de nommage des pièces scannées par le service urbanisme ;
- la stratégie de sauvegarde à mettre en place pour conserver les pièces enregistrées dans le système d'information.

Développement

Un membre de l'équipe a décrit sous forme graphique les rôles des utilisateurs "déposant", "service urbanisme" et "DDE". En outre il s'interroge sur le lieu de stockage idéal des fichiers correspondant aux pièces des demandes de permis : est-il préférable d'enregistrer les fichiers directement en base de données dans des champs spécifiques ou dans des fichiers externes à la base de données, ou encore d'adopter une solution mixte ?

Il a été finalement décidé de stocker l'ensemble des documents sous forme de fichiers externes à la base de données.

Vos tâches

Vous êtes chargé-e de compléter le dossier de spécifications fonctionnelles commencé par un membre de l'équipe et vous disposez pour cela de la documentation suivante :

- description graphique des rôles ;
- extrait du site *architectures-web.smile.fr* sur les fichiers dans les bases de données.

Définition des niveaux d'habilitation

1.1 Modéliser les rôles des différents architectes en adoptant le formalisme utilisé par le membre de l'équipe.

Stockage des pièces numérisées

- 1.2 Justifier le choix du lieu de stockage des fichiers correspondant aux pièces des demandes de permis.**
- 1.3 Proposer l'arborescence des répertoires de stockage et les règles de nommage des pièces numérisées.**

Mission 2 : validation de la base de données

Cahier des charges de la mission

La base de données *bd_Demandes_PC* a pour objet de stocker les informations relatives aux demandes de permis de construire (demande, demandeur, architecte, pièce). Construite à partir du formulaire *Cerfa 13406*, elle doit contenir dans une table *Parcelle* les informations concernant les parcelles issues d'une autre base de données nommée "Cadastre".

Il est rappelé qu'une demande :

- concerne un terrain qui peut être constitué d'une ou de plusieurs parcelles ;
- peut nécessiter le recours à un architecte (dans ce cas, la présence de sa signature et son cachet sont obligatoires, sinon c'est la présence de la déclaration sur l'honneur qui l'est).

Structure de la base de données

La base de données liée à la partie *web* du projet PC doit être en parfaite adéquation avec l'imprimé *Cerfa 13406* de demande de permis de construire.

Elle prendra en compte les parcelles situées dans des secteurs protégés. En effet, certaines parcelles appartiennent à un ou plusieurs secteurs protégés et peuvent, à ce titre, faire l'objet d'une réglementation particulière concernant les demandes de permis de construire.

Un secteur protégé se décline en différents types ("secteurs sauvegardés", "monuments historiques", etc.) faisant chacun référence à un article du code de l'urbanisme. Chaque article peut citer un ou plusieurs autres articles du code ou être lui-même cité dans un ou plusieurs autres articles. Les articles peuvent être modifiés par un ou plusieurs décrets.

État récapitulatif

L'application *web* devra fournir un état des pièces jointes obligatoires pour une demande donnée. Nécessaire pour les utilisateurs de la mairie qui ont en charge la saisie et le suivi des demandes, cet état précisera pour chaque pièce jointe le nombre d'exemplaires à fournir et celui effectivement fourni par le déposant. Par

exemple, pour la demande de permis de construire identifié par le numéro 452 dans la table *demande*, la liste serait la suivante :

libelle	nb_exemplaires_a_fournir	nb_exemplaires_fournis
PCMI2	6	4
PCMI3	2	0
PCMI4	1	1

On remarque que, pour cette demande, le déposant n'a pas fourni le nombre d'exemplaires attendu de la pièce PCMI2 contrairement à la pièce PCMI4. Quant à la pièce PCMI3, aucun exemplaire n'a été fourni.

Développement

Une première version de l'application *Web_Permis* et de sa base de données a été conçue et les derniers tests effectués sur l'application ont mis en évidence plusieurs défauts lors de l'insertion de nouvelles demandes :

- A. les demandes dont le terrain possède plusieurs parcelles ne sont pas correctement enregistrées, seule une des parcelles est affectée au terrain (élément 3.1 du formulaire *Cerfa 13406*) ;
- B. les demandes dont une ou plusieurs des situations juridiques (élément 3.2 du formulaire *Cerfa 13406*) sont inconnues, ne sont pas correctement enregistrées ;
- C. la présence de la signature et du cachet d'un éventuel architecte ne sont pas mémorisés (élément 4.1 du formulaire *Cerfa 13406*) ;
- D. les demandes ne déclarant pas de SHOB (surface hors œuvre brute, élément 4.3 du formulaire *Cerfa 13406*) sont refusées.

Enfin, un membre de l'équipe a constaté qu'on pouvait enregistrer des demandes :

- contenant à la fois un numéro d'architecte et une déclaration sur l'honneur ;
- ne contenant ni numéro d'architecte ni déclaration sur l'honneur.

Vos tâches

Vous êtes chargé-e de faire évoluer cette première version afin de répondre aux besoins du cahier des charges et vous disposez pour cela de la documentation suivante :

- ébauche du schéma relationnel de la base *bd_Demandes_PC*,
- script de création de la base *bd_Demandes_PC*,
- extrait du tableau récapitulatif des types du secteur protégé,
- Extrait de l'article R. 421-15 du code de l'urbanisme,
- Extrait du formulaire *Cerfa n° 13406*.

Dans la rédaction de vos solutions, vous pourrez utiliser les formalismes de votre choix : les éléments de code peuvent être présentés à l'aide d'un langage de programmation ou de pseudo-code algorithmique et les schémas de données à l'aide d'un diagramme UML, d'un schéma entité-association ou encore d'un schéma relationnel.

Écriture de la requête SQL

2.1 Écrire la requête SQL permettant d'alimenter l'état des pièces jointes obligatoires pour une demande de permis.

Résolution des problèmes rencontrés lors de l'insertion de nouvelles demandes

2.2 Expliquer les causes des défauts A, B, C et D et proposer une solution pour chacun d'eux.

2.3 Décrire dans une courte note la solution permettant de bloquer l'insertion d'une demande contenant à la fois un identifiant d'architecte et une déclaration sur l'honneur ou ne contenant ni l'un ni l'autre.

Évolution de la base de données

2.4 Compléter la structure de la base de données afin de prendre en compte les parcelles situées dans des secteurs protégés ainsi que l'ensemble des informations associées à ces secteurs.

Mission 3 : amélioration de l'interface web

Cahier des charges de la mission

La partie *web* de l'application doit permettre de télécharger et de consulter les pièces déposées en mairie suite au dépôt des demandes de permis de construire. Ce service a été mis en œuvre de manière à faciliter :

- ✓ l'étude de certaines pièces difficiles à manipuler sur support papier ;
- ✓ les échanges entre les différents intervenants participant au processus de traitement d'une demande de permis de construire (déposant, architectes des déposants, service urbanisme de la mairie, architecte conseil de la mairie, direction départementale de l'équipement).

Une des pages de l'application présentera une liste déroulante affichant le code et le libellé des pièces (par exemple " PCMI1 – Plan de situation du terrain") dans laquelle les déposants pourront sélectionner la pièce à télécharger en émission (*uploader*).

Cette liste déroulante sera alimentée par la fonction PHP **liste_deroulante(\$fichier_XML)** dont le but est de créer automatiquement des listes déroulantes en *HTML*. Elle reçoit en paramètre le nom d'un fichier *XML* qui contient les éléments permettant de construire la liste. Elle retourne une chaîne de caractères qui correspond au code *HTML* de la liste déroulante (balises *SELECT* et *OPTION*).

Cette fonction est utilisée dans plusieurs pages *web* du site car la direction des systèmes d'information a demandé aux développeurs d'alimenter les éléments graphiques des pages *web* en priorité avec des fichiers XML plutôt qu'avec des données issues d'une base de données. Cette pratique a pour objectif de limiter les accès aux bases de données lors du remplissage de listes déroulantes statiques.

Développement

Un des membres de l'équipe a déjà programmé l'interface *web* mais a omis de placer le libellé des pièces dans la liste déroulante. Après conseil auprès de votre équipe, trois de vos collègues proposent des solutions différentes :

- solution n°1 : créer un nouveau fichier XML contenant un attribut `<detail>` supplémentaire et modifier la fonction **liste_deroulante(\$fichier_XML)** afin qu'elle s'adapte à la structure de ce nouveau fichier ;
- solution n°2 : modifier le fichier XML existant en ajoutant le libellé directement dans la valeur de la balise **<label>** ;
- solution n°3 : modifier la fonction **liste_deroulante(\$fichier_XML)** afin qu'elle récupère le code et le libellé directement dans la table *piece* de la base de données.

Suite à ces propositions, le membre de l'équipe a finalement ajouté un attribut `<detail>` supplémentaire dans le fichier XML existant conformément à la solution n°1.

Vos tâches

Vous êtes chargé-e de faire évoluer cette version afin de répondre aux besoins du cahier des charges et vous disposez pour cela de la documentation suivante :

- interface *web* avant et après amélioration,
- code PHP faisant appel à la fonction,
- extrait du fichier XML existant,
- code PHP de la fonction *liste_deroulante* (*\$fichier_XML*),
- extrait du fichier XML modifié.

Dans la rédaction de vos solutions, vous pourrez utiliser les formalismes de votre choix : les éléments de code peuvent être présentés à l'aide d'un langage de programmation ou de pseudo-code algorithmique et les schémas de données à l'aide d'un diagramme UML, d'un schéma entité-association ou encore d'un schéma relationnel.

Étude des solutions proposées

3.1 Pour chaque solution, indiquer si elle est techniquement viable pour le projet.

Adaptation d'une solution

3.2 Modifier la fonction *liste_deroulante(\$fichier_XML)* afin qu'elle s'adapte au fichier XML modifié conformément à la solution n°1 et qu'elle réponde au besoin exprimé dans le cahier des charges.

Mission 4 : évolution de l'application *Gest_PC*

Cahier des charges de la mission

L'application *Gest_PC*, écrite en *Java* et accédant à sa propre base de données est utilisée quotidiennement par les secrétaires et le directeur du service qui gèrent les permis de construire.

Pour répondre au besoin de la direction informatique l'application *Gest_PC* a évolué afin de permettre d'accéder à la nouvelle base de données *bd_Demandes_PC*. De nouveaux écrans et de nouvelles classes d'objet capables de recevoir les données de la base ont donc été ajoutés à l'application *Gest_PC*. Toutes ces modifications ont été réalisées par vos collègues.

Création d'une classe générateur

Afin de tester l'application de manière approfondie, il a été demandé à l'équipe informatique d'alimenter la base de données *bd_Demandes_PC* avec un jeu d'essai conséquent. Les données seront insérées dans la base à l'aide d'un programme de test nommé *Générateur* et écrit en langage *Java*. Il alimentera massivement et aléatoirement la base de données, dans un contexte proche de la réalité, avec des données générées aléatoirement. À chaque lancement du générateur, l'ensemble des données générées seront affichées au fur et à mesure afin d'avoir une trace de l'exécution.

Le générateur utilisera un paquetage de classes nommé *personnes* contenant les classes *personne*, *déposant*, *architecte* et *adresse*. Les données seront d'abord créées sous forme d'instance de la classe *personne*, puis insérées dans la base.

Bien que l'on utilise une fonction aléatoire pour la création des données, celles-ci devront être suffisamment réalistes. Par exemple, pour générer un architecte, son nom et son prénom seront pris au hasard dans des listes de nom et de prénom, son numéro de téléphone sera généré aléatoirement mais aura le format 0x.xx.xx.xx.xx (où x est un chiffre), il doit y avoir une cohérence entre la ville et le code postal dans les adresses, etc.

Développement

Vous disposez d'une partie du code du programme *Générateur* :

```
22 public void demarrer(int nbDeposants, int nbArchitectes) {
    // Instruction de connexion à la base de données qui reçoit en
    // paramètre
    // l'adresse du poste, le nom de la base de données,
    // les login et mot de passe de l'utilisateur
23     pg_connect(' host="127.0.0.1" dbname="bd_Demandes_PC"
                  user="permis_ro" password="pass_permis" ');
24     Personne     personne     =     new     Personne(genererNom(),
    genererPrenom());
    ... }
```

Au lancement du programme *Générateur*, deux messages d'erreur sont apparus :

- à la compilation :
`Generateur.java:24: Personne is abstract; cannot be instantiated`
- à l'exécution:
`Generateur.java:23: Data insert not allowed for current user;`

Après exécution du programme *Générateur*, deux défauts ont été remarqués :

- le résultat produit par la méthode *toString()* de la classe *Architecte* n'est pas celui attendu car il ne contient ni le nom ni le prénom de l'architecte ;
- la relation entre un architecte et ses déposants n'est pas prise en compte. Il est indispensable que la méthode *toString()* de la classe *Architecte* retourne également, pour un architecte donné, les nom et prénom des déposants qu'il assiste.

Vos tâches

Vous êtes chargé-e de faire évoluer cette version afin de répondre aux besoins du cahier des charges et vous disposez pour cela de la documentation suivante :

- parcours d'une collection en *Java*,
- utilisateurs de la base de données,
- diagramme de classes simplifié du paquetage personnes,
- extrait des classes du paquetage personnes.

Dans la rédaction de vos solutions, vous pourrez utiliser les formalismes de votre choix : les éléments de code peuvent être présentés à l'aide d'un langage de programmation ou de pseudo-code algorithmique et les schémas de données à l'aide d'un diagramme UML, d'un schéma entité-association ou encore d'un schéma relationnel.

Étude et correction des erreurs du programme Générateur

- 4.1 Expliquer les causes des deux messages d'erreur rencontrés à la compilation et à l'exécution du programme *Générateur* et donner les solutions permettant de supprimer ces causes.**

Étude et correction du défaut d'affichage

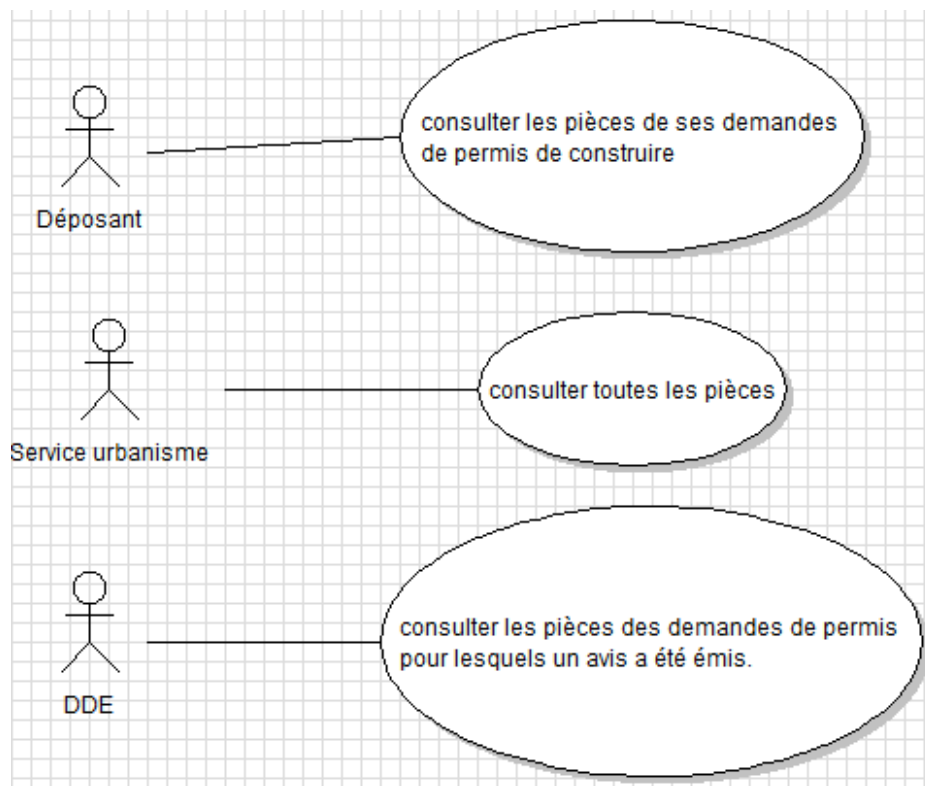
- 4.2 Compléter la méthode *toString()* de la classe *Architecte* afin qu'elle retourne les nom et prénom de l'architecte (*écrire seulement le complément de code*).**

Étude et correction du défaut de prise en compte des déposants

- 4.3 Modifier le diagramme de classe afin de prendre en compte les déposants assistés par un architecte (*reproduire succinctement le diagramme existant afin de faire apparaître la modification*).**
- 4.4 Compléter la méthode *toString()* de la classe *Architecte* de manière à ce qu'elle retourne, pour un architecte donné, les nom et prénom des déposants qu'il assiste. (*écrire seulement le complément de code*).**

Ressources documentaires

Description graphique des rôles



Extrait du site *architectures-web.smile.fr* sur les fichiers dans les bases de données

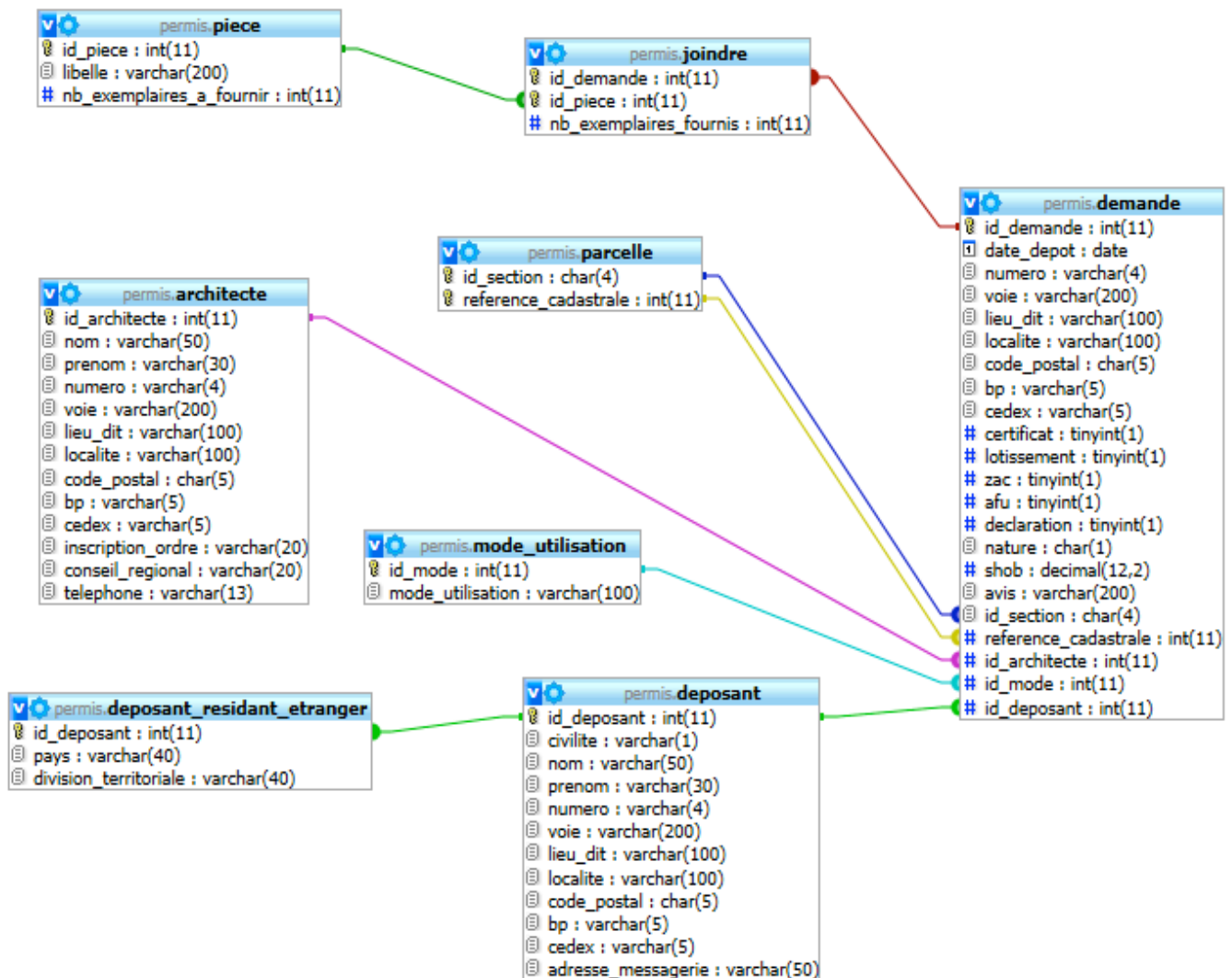
Les bases de données acceptent des objets binaires (BLOB) de grande taille, qui peuvent stocker des fichiers. Gérer des fichiers en base de données peut être une option, lorsque la volumétrie est faible. Il y a quelques bénéfices à en attendre :

- l'homogénéité des accès : données structurées et fichiers sont gérés de la même manière ;
- la cohérence : typiquement, aussitôt qu'une application manipule à la fois une référence à un fichier, dans la base de données, et le fichier proprement dit, il y a des risques d'incohérences entre l'un et l'autre ;
- l'homogénéité et la cohérence des sauvegardes : en une seule opération, on sauvegarde l'état courant de la base et des fichiers associés.

Par ailleurs, dès que la volumétrie grandit, c'est un mauvais choix :

- la base de données devient énorme et les opérations d'exploitation deviennent difficiles ;
- les performances sont inférieures à celles d'un *file system* ;
- il y a une certaine opacité du dispositif pour les exploitants.

Ébauche du schéma relationnel de la base *bd_Demandes_PC*



À l'insertion d'une nouvelle demande, les lignes de la table **joindre** sont automatiquement créées (avec un nombre d'exemplaires à zéro) en fonction des pièces nécessaires.

Script de création de la base *bd_Demandes_PC*

-- La table parcelle est déjà existante.

create table depositant

```
(id_depositant integer primary key,  
civilite varchar(1) null,  
nom varchar(50) not null,  
prenom varchar(30) not null,  
numero varchar(4) null,  
voie varchar(200) not null,  
lieu_dit varchar(100) null,  
localite varchar(100) not null,  
code_postal char(5) not null,  
bp varchar(5) null,  
cedex varchar(5) null,  
adresse_messagerie varchar(50) null);
```

create table piece

```
(id_piece integer primary key,  
libelle varchar(200) not null,  
nb_exemplaires_a_fournir integer not null);
```

create table architecte

```
(id_architecte integer primary key,  
nom varchar(50) not null,  
prenom varchar(30) not null,  
numero varchar(4) null,  
voie varchar(200) not null,  
lieu_dit varchar(100) null,  
localite varchar(100) not null,  
code_postal char(5) not null,  
bp varchar(5) null,  
cedex varchar(5) null,  
inscription_ordre varchar(20) null,  
conseil_regional varchar(20) null,  
telephone varchar(13) null);
```

create table mode_utilisation

```
(id_mode integer primary key,  
mode_utilisation varchar(100) not null);
```

create table joindre

```
(id_demande integer not null references  
demande(id_demande),  
id_piece integer not null references  
piece(id_piece),  
nb_exemplaires_fournis integer not null,  
primary key (id_demande, id_piece));
```

create table depositant_resident_etranger

```
(id_depositant integer primary key,  
pays varchar(40) not null,  
division_territoriale varchar(40) not null,  
foreign key (id_depositant) references  
depositant(id_depositant));
```

create table demande

```
(id_demande integer primary key,  
date_depot date not null,  
numero varchar(4) null,  
voie varchar(200) not null,  
lieu_dit varchar(100) null,  
localite varchar(100) not null,  
code_postal char(5) not null,  
bp varchar(5) null,  
cedex varchar(5) null,  
certificat boolean not null,  
lotissement boolean not null,  
zac boolean not null,  
afu boolean not null,  
declaration boolean not null,  
nature char(1) not null,  
shob numeric(12,2) not null,  
avis varchar(200) null,  
id_section char(4) not null,  
reference_cadastrale integer not null,  
id_architecte integer not null references  
architecte(id_architecte),  
id_mode integer not null references  
mode_utilisation(id_mode),  
id_depositant integer not null references  
depositant(id_depositant),  
foreign key (id_section,  
reference_cadastrale) references  
parcelle(id_section, reference_cadastrale));
```

Extrait du tableau récapitulatif des types du secteur protégé

Secteurs sauvegardés	Les travaux situés à l'intérieur des immeubles ou parties d'immeuble visés par le PSMV, ou sur un élément identifié comme présentant un intérêt patrimonial ou paysager par le PSMV : PERMIS DE CONSTRUIRE	Art. R.421-15 du code de l'urbanisme
Monuments historiques inscrits	Tous les travaux sont soumis à PERMIS DE CONSTRUIRE	Art. R.421-16 du code de l'urbanisme

Extrait de l'article R. 421-15 du code de l'urbanisme

Article R*421-15

Modifié par Décret n°2007-18 du 5 janvier 2007 - art. 8 JORF 6 janvier 2007 en vigueur le 1er octobre 2007

Modifié par Décret n°2007-18 du 5 janvier 2007 - art. 9 JORF 6 janvier 2007 en vigueur le 1er octobre 2007

Dans les secteurs sauvegardés dont le plan de sauvegarde et de mise en valeur est approuvé, sont en outre soumis à permis de construire, à l'exception des travaux d'entretien ou de réparations ordinaires :

- a) Les travaux exécutés à l'intérieur des immeubles ou parties d'immeubles visés au III de l'article [L. 313-1](#), lorsqu'ils ont pour objet ou pour effet de modifier la structure du bâtiment ou la répartition des volumes existants ;
- b) Les travaux qui portent sur un élément que le plan de sauvegarde et de mise en valeur a identifié, en application du 7° de l'article [L. 123-1](#), comme présentant un intérêt patrimonial ou paysager.

Liens relatifs à cet article

Cite:

[Code de l'urbanisme - art. L123-1 \(VT\)](#)

[Code de l'urbanisme - art. L313-1 \(VT\)](#)

Cité par:

[Code de l'urbanisme - art. R*421-22 \(M\)](#)

[Code de l'urbanisme - art. R*421-22 \(M\)](#)



[Code de l'urbanisme - art. R*421-22 \(MMN\)](#)

[Code de l'urbanisme - art. R*421-36 \(Ab\)](#)

[Code de l'urbanisme - art. R*421-36 \(M\)](#)

[Code de l'urbanisme - art. R*421-36 \(M\)](#)

Extrait du formulaire Cerfa n° 13406

 <small>Liberté • Égalité • Fraternité REPUBLIQUE FRANÇAISE</small>	<h2 style="margin: 0;">Demande de Permis de construire</h2> <h3 style="margin: 0;">pour une maison individuelle et / ou ses annexes</h3>	 N° 13406*01
* 1 - Identité du ou des demandeurs Le demandeur indiqué dans le cadre ci-dessous sera le titulaire de la future autorisation et le redevable des taxes d'urbanisme <small>Si la demande est présentée par plusieurs personnes, indiquez leurs coordonnées sur la fiche complémentaire. Les décisions prises par l'administration seront notifiées au demandeur indiqué ci-dessous. Une copie sera adressée aux autres demandeurs, qui seront co-titulaires de l'autorisation et solidairement responsables du paiement des taxes.</small>		
<div style="display: flex; justify-content: space-between;">Vous êtes un particulierMadame <input type="checkbox"/> Monsieur <input type="checkbox"/></div> <div style="display: flex; justify-content: space-between;">Nom : _____Prénom : _____</div>		
2 - Coordonnées du demandeur		
<div style="display: flex; justify-content: space-between;">* Adresse : Numéro : _____Voie : _____</div> <div style="display: flex; justify-content: space-between;">Lieu-dit : _____Localité : _____</div> <div style="display: flex; justify-content: space-between;">Code postal : _____BP : _____Cedex : _____</div> <div style="display: flex; justify-content: space-between;">Si le demandeur habite à l'étranger : Pays : _____Division territoriale : _____</div>		

3 - Le terrain

3.1 - localisation du (ou des) terrain(s)

Les informations et plans (voir liste des pièces à joindre) que vous fournissez doivent permettre à l'administration de localiser précisément le (ou les) terrain(s) concerné(s) par votre projet.

Le terrain est constitué de l'ensemble des parcelles cadastrales d'un seul tenant appartenant à un même propriétaire

Adresse du (ou des) terrain(s)

Numéro : _____ Voie : _____

Lieu-dit : _____ Localité : _____

Code postal : _____ BP : _____ Cedex : _____

Références cadastrales : section et numéro : _____

Superficie du (ou des) terrain(s) (en m²) : _____

3.2 - Situation juridique du terrain (ces données, qui sont facultatives, peuvent toutefois vous permettre de faire valoir des droits à construire ou de bénéficier d'impositions plus favorables)

Êtes-vous titulaire d'un certificat d'urbanisme pour ce terrain ? Oui ☐ Non ☐ Je ne sais pas ☐

Le terrain est-il situé dans un lotissement ? Oui ☐ Non ☐ Je ne sais pas ☐

Le terrain est-il situé dans une Zone d'Aménagement Concertée (Z.A.C.) ? Oui ☐ Non ☐ Je ne sais pas ☐

Le terrain fait-il partie d'un remembrement urbain (Association Foncière Urbain) ? Oui ☐ Non ☐ Je ne sais pas ☐

4 - Caractéristiques du projet

4.1 - Architecte

*Vous avez eu recours à un architecte : Oui ☐ Non ☐

Si oui, vous devez lui faire compléter les rubriques ci-dessous et lui faire apposer son cachet

Nom de l'architecte : _____ Prénom : _____

Numéro : _____ Voie : _____

Lieu-dit : _____ Localité : _____

Code postal : _____ BP : _____ Cedex : _____

N° d'inscription sur le tableau de l'ordre : _____

Conseil Régional de : _____

Téléphone : _____

En application de l'article R. 431-2 du code de l'urbanisme, j'ai pris connaissance des règles générales de construction prévues par le chapitre premier du titre premier du livre premier du code de la construction et de l'habitation et notamment, lorsque la construction y est soumise, les règles d'accessibilité fixées en application de l'article L. 111-7 de ce code.

Signature de l'architecte : _____	Cachet de l'architecte : _____
-----------------------------------	--------------------------------

Si vous n'avez pas eu recours à un architecte (ou un agréé en architecture), veuillez cocher la case ci-dessous :

☐ Je déclare sur l'honneur que mon projet entre dans l'une des situations pour lesquelles le recours à l'architecte n'est pas obligatoire.

4.2 - Nature des travaux envisagés

☐ Nouvelle construction

☐ Travaux sur construction existante

4.3 - Surface hors œuvre brute (SHOB)

Si votre projet de construction se situe dans une commune non dotée de plan local d'urbanisme (PLU) ou d'un document en tenant lieu (plan d'occupation des sols, plan de sauvegarde et de mise en valeur, plan d'aménagement de zone), indiquez la surface hors œuvre brute (SHOB) totale du projet

SHOB des travaux de construction (en m²) : _____

4.4 - Informations complémentaires

◆ Mode d'utilisation principale des logements :

Résidence principale ☐

Résidence secondaire ☐

Vente ☐

Location ☐

1) Pièces obligatoires pour tous les dossiers :

Pièce	Nombre d'exemplaires à fournir
<input type="checkbox"/> PCMI1. Un plan de situation du terrain [Art. R. 431-7 a) du code de l'urbanisme]	1 exemplaire par dossier + 5 exemplaires supplémentaires
<input type="checkbox"/> PCMI2. Un plan de masse des constructions à édifier ou à modifier [Art. R. 431-9 du code de l'urbanisme]	1 exemplaire par dossier + 5 exemplaires supplémentaires
<input type="checkbox"/> PCMI3. Un plan en coupe du terrain et de la construction [Article R. 431-10 b) du code de l'urbanisme]	1 exemplaire par dossier + 5 exemplaires supplémentaires
<input type="checkbox"/> PCMI4. Une notice décrivant le terrain et présentant le projet [Art. R. 431-8 du code de l'urbanisme]	1 exemplaire par dossier
<input type="checkbox"/> PCMI5. Un plan des façades et des toitures [Art. R. 431-10 a) du code de l'urbanisme]	1 exemplaire par dossier
<input type="checkbox"/> PCMI6. Un document graphique permettant d'apprécier l'insertion du projet de construction dans son environnement [Art. R. 431-10 c) du code de l'urbanisme]*	1 exemplaire par dossier
<input type="checkbox"/> PCMI7. Une photographie permettant de situer le terrain dans l'environnement proche [Art. R. 431-10 d) du code de l'urbanisme]*	1 exemplaire par dossier
<input type="checkbox"/> PCMI8. Une photographie permettant de situer le terrain dans le paysage lointain [Art. R. 431-10 d) du code de l'urbanisme]*	1 exemplaire par dossier

2) Pièces à joindre selon la nature ou la situation du projet :

Pièce	Nombre d'exemplaires à fournir
Si votre projet se situe dans un lotissement :	
<input type="checkbox"/> PCMI9. Le certificat indiquant la surface constructible attribuée à votre lot [Art. R. 431-22 a) du code de l'urbanisme]	1 exemplaire par dossier
<input type="checkbox"/> PCMI10. Le certificat attestant l'achèvement des équipements desservant le lot [Art. R. 431-22 b) du code de l'urbanisme]	1 exemplaire par dossier
Si votre projet se situe dans une zone d'aménagement concertée (ZAC) :	
<input type="checkbox"/> PCMI11. Une copie des dispositions du cahier des charges de cession de terrain qui indiquent le nombre de m² constructibles sur la parcelle et, si elles existent, des dispositions du cahier des charges, qui fixent les prescriptions techniques, urbanistiques et architecturales imposées pour la durée de réalisation de la zone [Art. R. 431-23 du code de l'urbanisme]	1 exemplaire par dossier
<input type="checkbox"/> PCMI12. La convention entre la commune ou l'établissement public et vous qui fixe votre participation au coût des équipements de la zone [Art. R. 431-23 b) du code de l'urbanisme]	1 exemplaire par dossier
Si votre projet est tenu de respecter les règles parasismiques et paracycloniques :	
<input type="checkbox"/> PCMI13. L'attestation d'un contrôleur technique [Art. R. 431-16 b) du code de l'urbanisme]	1 exemplaire par dossier

Interface web avant et après amélioration

Choisir la pièce que vous souhaitez obtenir

Puis cliquez sur [télécharger]

- PCMI1
- PCMI2
- PCMI3
- PCMI4
- PCMI5
- PCMI6
- PCMI7
- PCMI8
- PCMI9
- PCMI10
- PCMI11
- PCMI12

Choisir la pièce que vous souhaitez obtenir

Puis cliquez sur [télécharger]

- PCMI1 - Plan de situation du terrain
- PCMI2 - Plan de masse des constructions à édifier ou à modifier
- PCMI3 - Plan en coupe du terrain et de la construction
- PCMI4 - Notice décrivant le terrain et présentant le projet
- PCMI5 - Plan des façades et des toitures
- PCMI6 - Document graphique permettant d'apprécier l'insertion du projet
- PCMI7 - Photographie permettant de situer le terrain dans l'environnement proche
- PCMI8 - Photographie permettant de situer le terrain dans le paysage lointain
- PCMI9 - Certificat indiquant la surface constructible attribuée à votre lot
- PCMI10 - Certificat attestant l'achèvement des équipements desservant le lot
- PCMI11 - Copie des dispositions du cahier des charges de cession de terrain
- PCMI12 - Convention entre la commune ou l'établissement public et vous

Code PHP faisant appel à la fonction

```
<form action="telecharger.php">
    Choisir la pièce que vous souhaitez obtenir
    <?php      echo liste_deroulante("pieces1.xml");
    ?>
    <br />Puis cliquez sur [télécharger]
    <input type="submit" value="Télécharger"/> <br /><br />
</form>
```


Extrait du fichier XML existant

```
<?xml version="1.0" encoding="utf-8"?>
<liste name="piece">
  <ligne>
    <value>1</value>
    <label>PCMI1</label>
  </ligne>
  ...
  <ligne>
    <value>4</value>
    <label>PCMI4</label>
  </ligne>
  ...
  <ligne>
    <value>12</value>
    <label>PCMI12</label>
  </ligne>
</liste>
```

Code PHP de la fonction liste_déroulante (\$fichier_XML)

```
<?php
//Chargement du fichier xml
01 function liste_deroulante($fichier_XML)
02 { if (file_exists($fichier_XML))
03     { $ma_liste = simplexml_load_file($fichier_XML); }
04 else
05     {$ma_liste = simplexml_load_string('<liste name="erreur">
06     <ligne><value>0</value>
07     <label>Erreur - Le fichier est manquant !</label>
08     </ligne></liste>');
09     }
10 //Création de la liste
11 $name = $ma_liste->attributes()->name;
12 $listed = '<select name="'. $name. '">';
13 foreach($ma_liste->children() as $ligne)
14     {$select = '';
15     if($ligne->attributes()->selected == 'true')
16         { $select = ' selected="selected" '; }
17     $listed .= '<option ' . $select. 'value="'. $ligne->value. '">';
18     $listed .= $ligne->label. '</option>';
19     }
20 $listed .= '</select>';
21 return $listed; } ?>
```

Extrait du fichier XML modifié

```
<?xml version="1.0" encoding="utf-8"?>
<liste name="piece" detail="1" separateur=" - ">

<!-- l'attribut detail à 1 signifie que le detail doit être affiché -->
<!-- l'attribut separateur precise le séparateur à utiliser entre le label et le
détail -->

  <ligne selected="true">
    <value>1</value>
    <label>PCMI1</label>
    <detail>Plan de situation du terrain</detail>
  </ligne>
```

```

...
<ligne>
  <value>4</value>
  <label>PCMI4</label>
  <detail>Notice décrivant le terrain et présentant le projet</detail>
</ligne>
...
<ligne>
  <value>12</value>
  <label>PCMI12</label>
  <detail>Convention entre la commune et vous</detail>
</ligne>
</liste>

```

Parcours d'une collection en Java

```

// Parcours de la collection les_clients
// On récupère dans l'objet un_client de type Client les objets contenus dans la collection les_clients

for( Client un_client : les_clients) {      //Traitement .... }

```

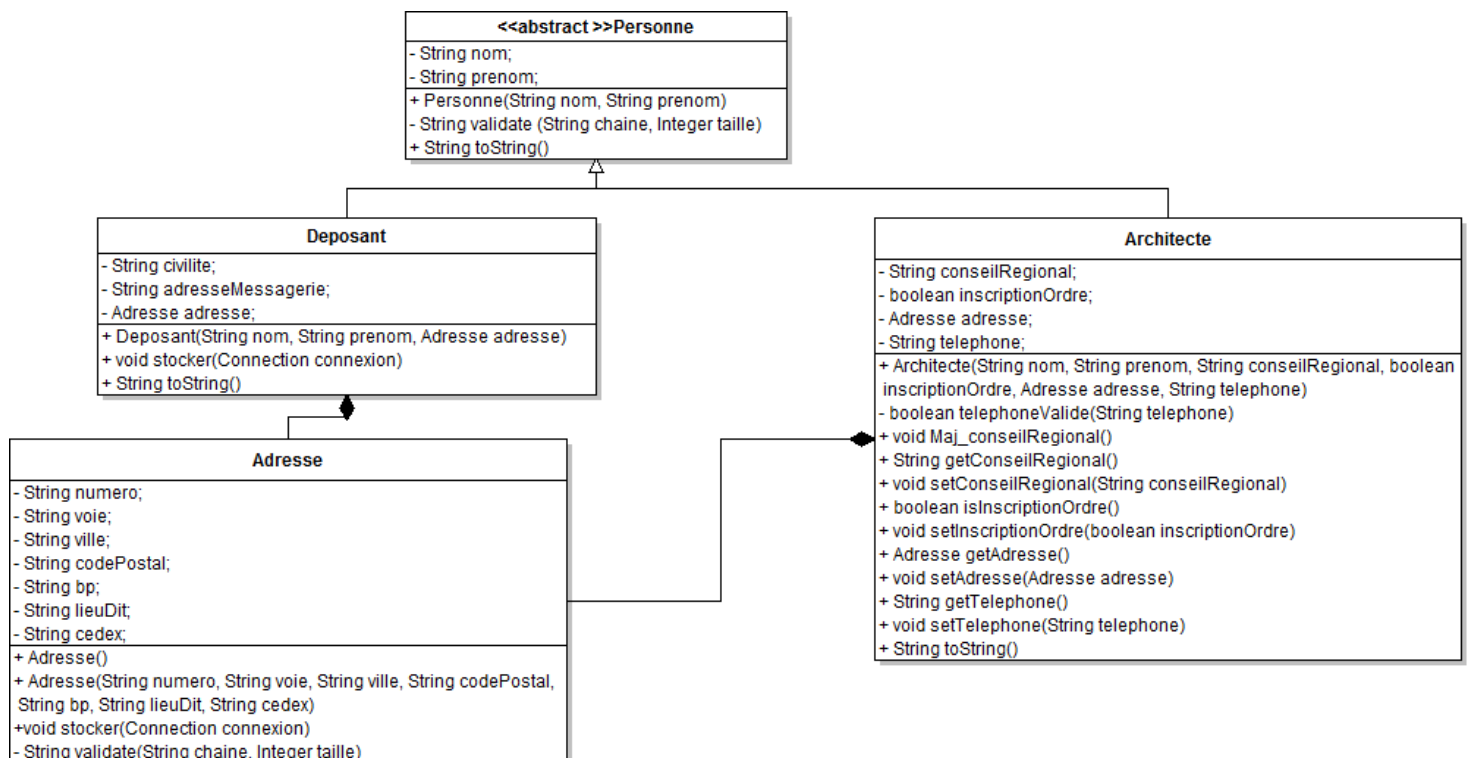
Utilisateurs de la base de données

Trois utilisateurs peuvent accéder à la base de données *bd_Demandes_PC* :

- ✓ **permis_admin** : utilisateur qui aura tous les droits sur la base et qui permettra aux développeurs de travailler sur les données et la structure ;
- ✓ **permis_rw** : utilisateur qui aura les droits en lecture et écriture sur des tables de la base "permis" ;
- ✓ **permis_ro** : qui n'aura que des droits de lecture sur des tables de la base de données.

Pendant la phase de développement, il est attribué à chaque rôle le mot de passe *pass_permis*.

Diagramme de classes simplifié du paquetage personnes



Extrait des classes du paquetage personnes

- **Personne.java**

```
package personnes;
public abstract class Personne
{
    private String nom;
    private String prenom;

    public Personne(String nom, String prenom)
    {
        this.nom = validate(nom, 50);
        this.prenom = validate(prenom, 30);
    }

    private String validate (String chaine, Integer taille)
    { // Vérifie la taille et renvoie une chaîne tronquée
      // si la taille est supérieure à taille
      return chaine;    }

    public String toString() {return this.nom + " " + this.prenom; }
}
```

- **Deposant.java**

```
package personnes;
import java.sql.Connection;

public class Deposant extends Personne
{
    private String civilite;
    private String adresseMessagerie;
    private Adresse adresse;

    public Deposant(String nom, String prenom, Adresse adresse)
    {
        super(nom, prenom);
        this.adresse = adresse;
    }

    public void stocker(Connection connexion) { }
}
```

- **Architecte.java**

```
package personnes;
import java.sql.Connection;

public class Architecte extends Personne {

    private String conseilRegional;
    private boolean inscriptionOrdre;
    private Adresse adresse;
    private String telephone;

    public Architecte(String nom, String prenom, String conseilRegional, boolean
inscriptionOrdre, Adresse adresse, String telephone)
    {
        super(nom, prenom);
        this.conseilRegional = super.validate(conseilRegional, 20);
        this.inscriptionOrdre = inscriptionOrdre;
        this.adresse = adresse;
        if (telephoneValide(telephone))
            { this.telephone = telephone; }
        else
            { this.telephone = ""; }
        Maj_conseilRegional();
    }

    private boolean telephoneValide(String telephone)
    { boolean b = true; return b; }

    public void Maj_conseilRegional()
    {
        if (this.getConseilRegional() == null)
            this.setConseilRegional("Conseil regional non renseigne");
    }

    // renvoie une chaine contenant le nom, le prenom, le conseil regional et
    // le numero de telephone de l'architecte ainsi que la mention "inscrit"
    // s'il est inscrit au conseil de l'ordre ou "non inscrit" dans le cas contraire
    public String toString ()
    {
        String chaine = "";
        chaine = chaine + this.conseilRegional + " ";
        if (this.inscriptionOrdre)
            chaine = chaine + "inscrit";
        else
            chaine = chaine + "pas inscrit";
        chaine = chaine + " " + this.telephone;
        return chaine;
    }
}
```