



Projet Génie Logiciel

# DOSSIER D'IMPLÉMENTATION Projet CARI

Juin 2014

Professeur : Adel MEZINE  
Référent : Franck POMMEREAU

Étudiants : Antoine MAZELIN, Mathieu PERNES, Laïny VINCENT

I.	Contexte du projet .....	4
1.	CARI .....	4
2.	Acteurs.....	5
3.	Processus de demandes de projet .....	6
a.	Achats courants .....	7
b.	Achats non courants.....	8
II.	Besoins.....	9
1.	Une plateforme commune .....	10
a.	Centralisation des demandes .....	10
b.	Optimisation des processus .....	10
2.	Unités fonctionnelles.....	11
a.	Un système de gestion des demandes.....	11
b.	Suivi des budgets.....	12
c.	Liste des demandes .....	12
d.	Système de gestion des utilisateurs .....	12
III.	Technologies.....	13
IV.	Diagrammes UML.....	15
1.	Diagrammes de Cas d'Utilisation.....	15
a.	Gestion des demandes CARI.....	15
b.	Gestion des utilisateurs .....	19
c.	Gestion des Budgets.....	21
2.	Diagramme de Classe .....	23
a.	Héritage sur l'Utilisateur .....	23
b.	Le Dossier et les produits qu'il contient.....	25
c.	Entité et Budget.....	26
3.	Diagrammes de Séquences .....	28
a.	Création d'un nouvel utilisateur.....	28
b.	Création d'une demande.....	30
c.	Soumission d'une demande .....	31
d.	Afficher le budget consommé .....	32
e.	Accepter un dossier soumis.....	32
4.	Diagramme d'Etat.....	34
5.	Diagramme de Package .....	35

V.	Maquettes .....	38
VI.	Implémentation de la solution .....	41
1.	Fonctionnalités développées : .....	41
2.	Outils utilisés .....	41
3.	Diagramme de classe modifié .....	42
4.	Base de données.....	44
VII.	Notice d'utilisation .....	46
1.	Scénarios fonctionnels .....	46
2.	Administration de compte et de l'application.....	51
VIII.	Bilan du projet .....	52
1.	Fonctionnalités non développées : .....	52
2.	Difficultés rencontrées .....	52

## I. Contexte du projet

### 1. CARI

CARI signifie Conseil Administratif des ressources informatiques.

C'est une instance qui permet de piloter les investissements informatiques de l'université. Elle valide et initie des projets d'investissements de matériels, de logiciels ou de services informatiques.

Elle est composée des représentants des différentes composantes de l'université ainsi que des commissions et conseils.

Ce conseil évalue les demandes d'achats informatiques et les hiérarchise selon :

- les ressources financières
- les orientations stratégiques de la politique d'enseignement et de recherche
- l'organisation des fonctions support.

Il définit la stratégie des achats et investissements informatiques et il accompagne les usagers de l'université dans l'expression de leurs besoins et le choix de solutions techniques.

Le CARI se prononce sur le financement des projets d'investissements en matériels, logiciels ou prestations informatiques pouvant être effectués en une fois ou selon un plan de financement.

Les demandes dans le cadre d'un appel à projet peuvent être envoyées à tout moment de l'année et seront examinées lors de la réunion du CARI suivant le dépôt de la demande.

Les demandeurs peuvent soumettre leurs projets en effectuant une demande, qui n'est pas complète. Dans ce cas, une commission est désignée afin de compléter les demandes et accompagner les demandeurs dans le montage de leur projet.

Un rapport sur la mise en œuvre des achats effectués devra être remis au CARI en fin d'action. Il conditionnera la soumission de projets par la suite.

Les réponses à cet appel à projet devront être portées par des personnels titulaires de l'université et doivent être transmises aux représentants de la composante dont est issue la demande.

Les demandes correspondant à des investissements courants ne sont pas examinées en séance. Par exemple, l'achat ou le remplacement d'un poste de travail relève de la dépense courante. Ces demandes peuvent être adressées au CARI selon une procédure accélérée.

Les demandes urgentes sont traitées de la même façon, mais peuvent être adressées aux représentants de composante ou directement au président du CARI.

Ces deux types de demandes sont envoyés par courriel sous la forme de la fiche résumée et de l'annexe budgétaire du dossier standard.

Un compte rendu de ces demandes acceptées par cette procédure simplifiée est fait lors de la séance suivante du CARI de manière à contrôler que des dépenses exceptionnelles n'ont pas été traitées en dépenses courantes ou urgentes.

Les investissements informatiques faits sur le budget propre d'une composante ou d'une unité (laboratoire, département, budget de projet, etc.) n'ont pas à faire l'objet d'une demande au CARI, sauf si un complément de financement est demandé. Cependant, tout investissement informatique important devrait être signalé dès que possible au CARI pour lui permettre de :

- garder une visibilité sur l'état des lieux
- proposer des ajustements
- participer financièrement
- proposer une analyse technique

Le CARI peut donc être amené à proposer sa participation à tout projet de cette nature qui se révèle d'intérêt général. Cette possibilité de participation est conçue comme incitative pour informer le CARI des investissements sur budget propre. Dans tous les cas, le CARI n'a pas vocation à s'opposer à de tels investissements.

## 2. Acteurs

Différents acteurs vont intervenir dans le processus de demandes d'investissements informatiques. On va retrouver :

- **Le porteur du projet** : c'est la personne qui fait la demande d'achat. Tous les membres d'une composante peuvent être porteurs de projet. L'université est décomposée en plusieurs composantes, les UFR. Une Unité de Formation et de Recherche (UFR) est une composante de l'université.

Elle est ensuite découpée en plusieurs départements et laboratoires. Chaque UFR définit un programme de recherche qui est mis en place par des professeurs qui sont également des chercheurs. On a par exemple : l'UFR DSP (Droit et Science Politique), l'UFR LAM (Langue, Art et Musique), l'UFR SFA (Sciences fondamentales appliquées)... De plus, chaque UFR est découpée en plusieurs unités : les

## MAZELIN-PERNES-VINCENT

laboratoires, départements ou services. Par exemple, dans l'UFR SFA, on va retrouver le département informatique. On va aussi retrouver une composante particulière : les services communs. (Service commun de documentation, service culturel...)

- **Le bénéficiaire du projet** : c'est le (les) bénéficiaire(s) du projet, c'est-à-dire ceux qui vont obtenir le matériel informatique nécessaire à l'issue du projet. On va donc dissocier la personne qui a besoin du matériel, et la personne qui va effectuer la demande de matériel. Dans chaque unité de composante, chaque personne devra se référer au porteur du projet afin de concrétiser son projet et obtenir le matériel demandé.
- **Le représentant des composantes** (des UFR ou des services) du côté CARI. C'est-à-dire que le porteur va soumettre la demande, puis un représentant va prendre une décision pour la demande. Il y a deux représentants par composante, qui réceptionnent toutes les demandes (courante ou non). Si la demande est courante, elle ne nécessite pas d'être examinée en séance de CARI. Cela représente du matériel de gestion courante comme des fournitures informatiques. Ils réceptionnent donc les demandes de la part des porteurs de projet. De plus, il peut visualiser toutes les demandes soumises au sein de sa composante.
- **Le président** :
  - Il a une visibilité sur toutes les demandes de projet et peut à tout moment modifier, refuser ou valider les demandes de projet, après discussion avec les sujets concernés.
  - Il préside les séances de CARI pour traiter les demandes non courantes qui nécessitent d'être examinées avec différents acteurs (représentants, responsable du budget...)
  - Il peut visualiser l'état des budgets de chaque composante

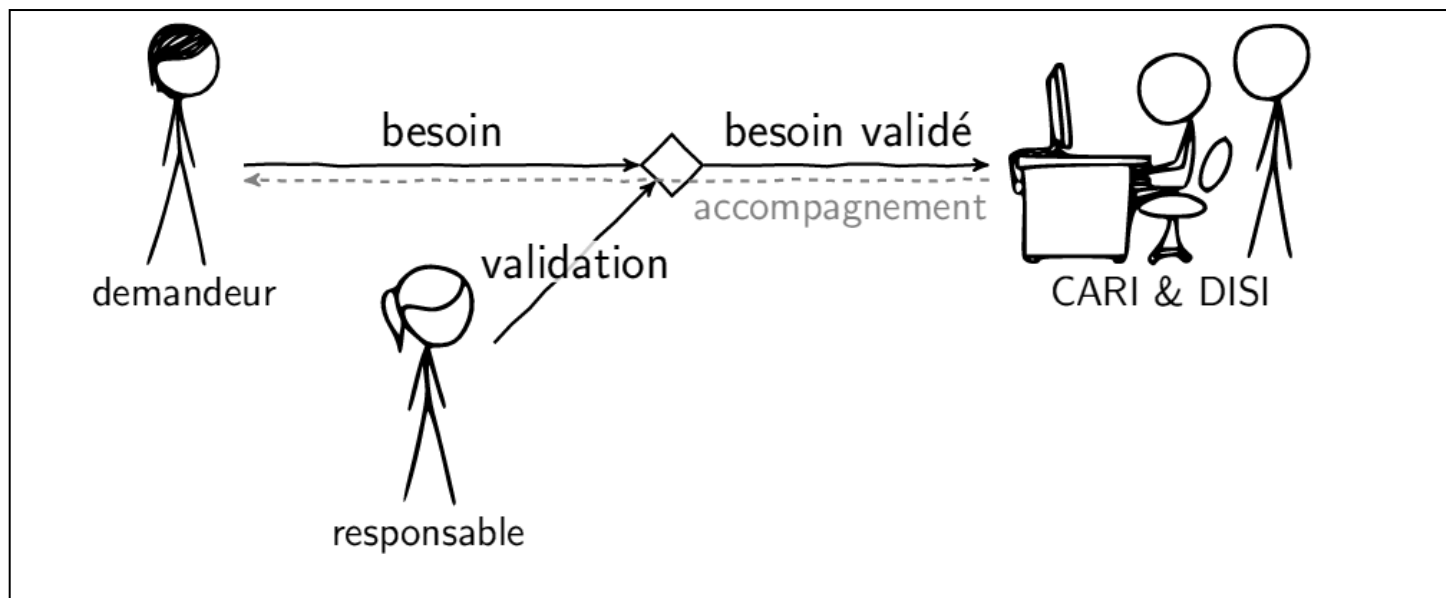
De plus, il y a un vice président en cas d'absence du président. Il a les mêmes droits que le président.

- **Les gestionnaires de budget** : ils n'interviennent pas dans le workflow, ils reçoivent juste des notifications en cas de validation d'une demande qui a été examinée en séance de CARI. Il y a le RAF (Responsable Administratif et Financier) du côté des UFR, les gestionnaires de budget pour les laboratoires, et le DAF (Directeur Administratif et Financier) pour les services communs.

### 3. Processus de demandes de projet

Pour effectuer une demande de financement, l'utilisateur doit renseigner un dossier qui contient :

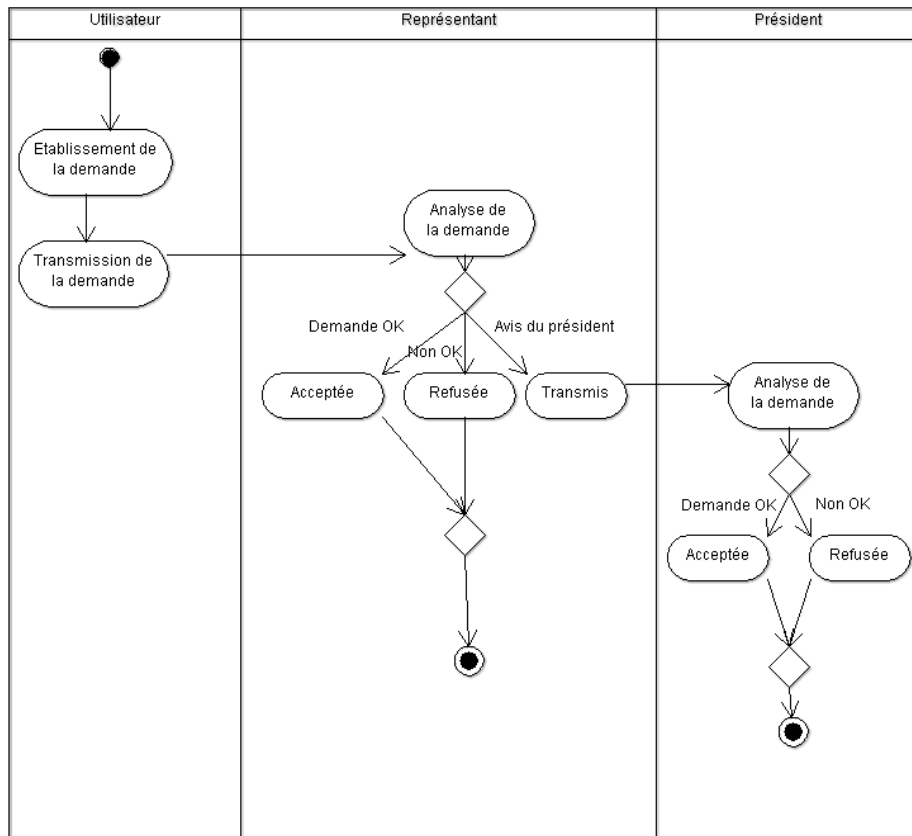
- Une fiche d'identification où on doit préciser les informations sur le projet à financer (Intitulé, montant total, description du projet et ses enjeux...).
- Un fichier Excel où l'utilisateur va apporter les informations sur le financement du projet (partie financée par CARI, fonds propres). Ce fichier permet de savoir précisément le financement demandé.



## Schéma simplifié du processus

### a. Achats courants

Cela concerne les demandes quotidiennes, comme le renouvellement de postes de travail, etc. Le représentant CARI peut les valider, il n'est pas nécessaire de passer par le président pour qu'elles soient acceptées. Ce processus peut être modélisé de la façon suivante :



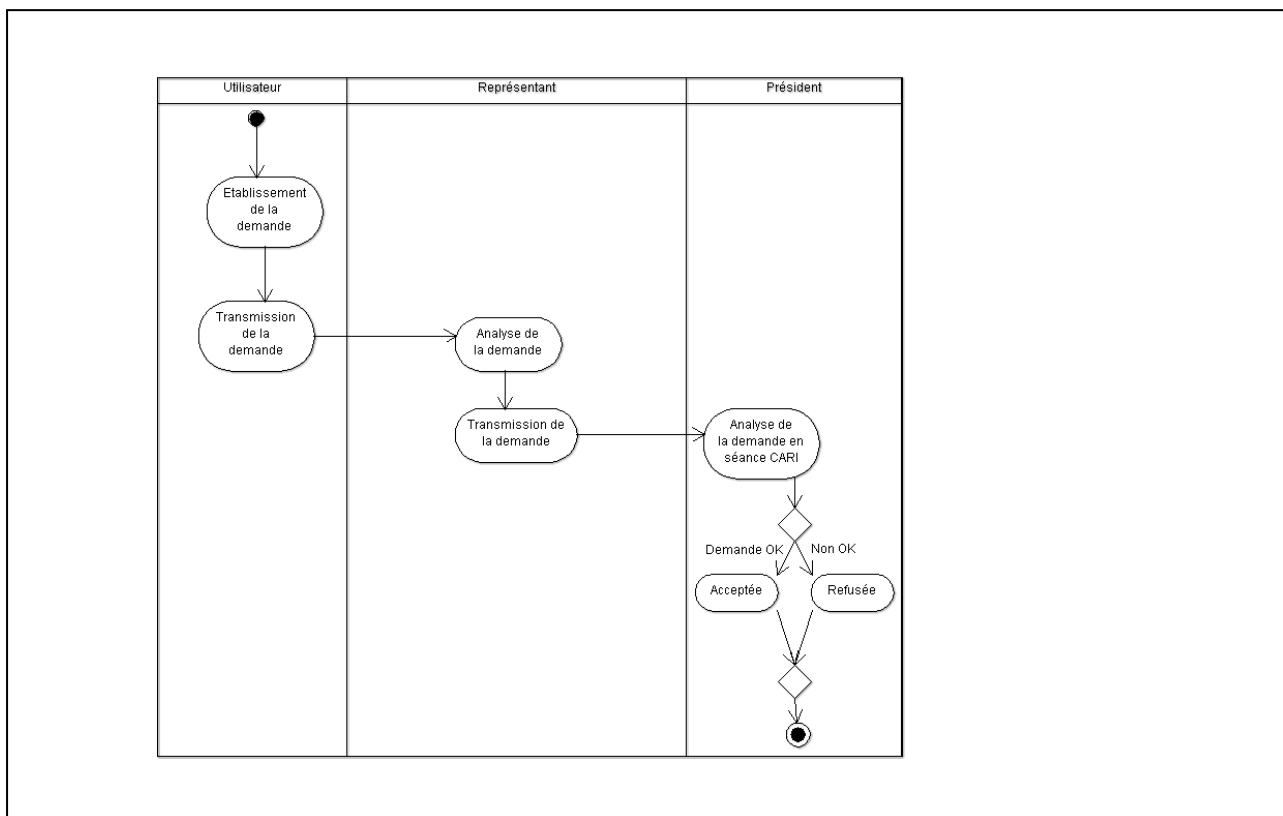
Ici, nous voyons que :

- L'utilisateur, va établir la demande (après une analyse préalable avec le bénéficiaire). Puis il va la transmettre à son représentant CARI.
- Dans le cadre d'un achat courant, le représentant CARI a le pouvoir de valider ou de refuser la demande de financement. Il peut aussi transmettre la demande au président pour validation.

#### b. Achats non courants

Ici il s'agit de traiter les demandes d'investissements « spéciales » :





Par rapport aux demandes d’achat courantes, nous constatons que les actions de l’utilisateur ne changent pas. Par contre :

- Le représentant CARI qui reçoit la demande, doit transmettre au président pour qu’elle passe en séance.
- En séance, l’ensemble des membres du CARI trancheront sur le sort de cette demande d’investissement, soit une acceptation ou un refus.

Actuellement, des mails sont échangés lors de chaque flux entre les différents acteurs pour qu’une demande soit analysée, en attachant le dossier. Le client voudrait donc changer cela en gérant autrement les demandes qui parviennent au CARI.

## II. Besoins

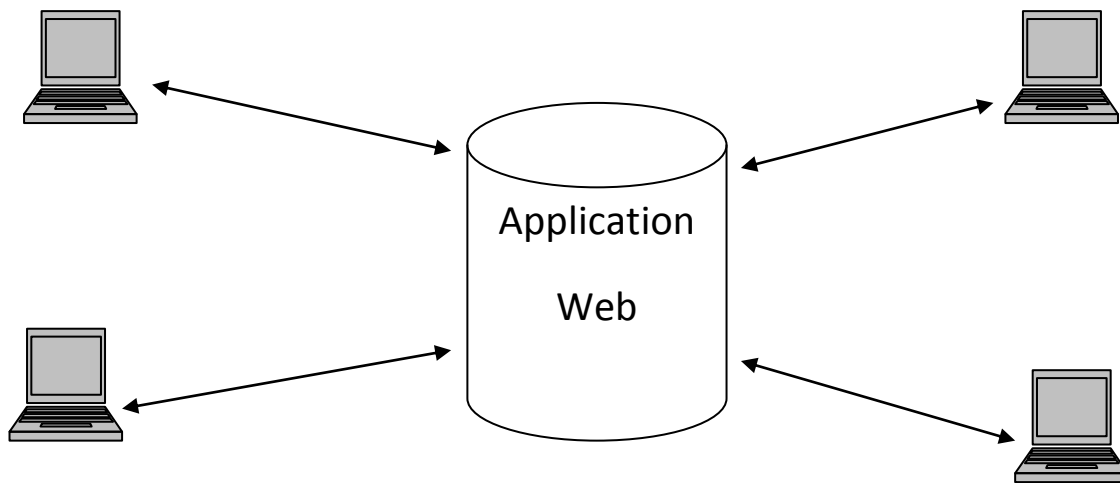
## 1. Une plateforme commune

Le but de la solution proposée est de pouvoir :

- Regrouper les demandes.
- Optimiser certains processus.

### a. Centralisation des demandes

Au sein d'une plateforme commune, l'utilisateur veut retrouver l'intégralité de ces demandes :



## Schématisation de la plateforme

Ainsi, sur ce même serveur, les principaux acteurs pourront :

- Visualiser les demandes en cours.
- Les approuver ou les rejeter.
- Accéder à divers outils d'aide à la décision, tableaux de bord...etc.

De plus, cette solution a un autre intérêt, qui se focalise sur les processus.

### b. Optimisation des processus

Nous avons vu lors de la description du processus, que le dossier de la demande circulait de mails en mails entre les différents acteurs. Donc, avec

## MAZELIN-PERNES-VINCENT

cette centralisation, nous pourrions automatiser certaines manipulations comme :

- Notifications de mail : quand le dossier aura changé d'état et deviendra disponible à la modification pour quelqu'un, cette même personne va être prévenue par un mail automatique. Ce message lui précisera le nouveau statut du dossier et l'invitera à apporter ses modifications.
- Gestion des droits d'accès : En fonction du statut de ce dossier, il sera modifiable que par certaines personnes. Nous allons donc introduire des droits d'accès propre à chaque acteur : le président aura tous les droits. Il pourra lire et modifier toute demande, vérifier et modifier les budgets des composantes, ajouter des utilisateurs sur l'application du CARI... Les représentants pourront donner leurs décisions sur les demandes de leurs UFR (refus, validation ou transmission au président) et ils auront accès à toutes les demandes correspondant à leur composante. Pour finir, le porteur des projets pourra soumettre des projets, les suivre et lire toutes les demandes qu'il a soumises.

## 2. Unités fonctionnelles

### a. Un système de gestion des demandes

Afin de soumettre une demande de projet, le porteur devra remplir un dossier de demande au CARI ainsi que une annexe budgétaire. Le dossier est composé des informations suivantes :

- L'intitulé du projet
- Le nom des bénéficiaires
- Le choix de la priorité de la demande (urgente ou pas). Si elle est urgente, il faut décrire le caractère d'urgence de la demande
- Le matériel demandé peut être un remplacement d'un matériel existant. Si c'est le cas, l'utilisateur devra indiquer la référence du matériel (numéro de série, modèle, ou toute autre information disponible)
- Description résumée qui décrit le projet et l'utilisation des crédits demandés

L'utilisateur aura la possibilité d'ajouter des pièces jointes (pour présenter des devis par exemple)

L'annexe budgétaire sera le deuxième document à remplir. On pourra y ajouter plusieurs produits en y précisant :

#### MAZELIN-PERNES-VINCENT

- la désignation du produit
- la quantité
- le prix unitaire TTC
- le choix du budget : choix entre les différents laboratoires, UFR, services communs ou fonctionnement.
- La date d'achat du matériel
- Le total demandé au CARI

De plus, l'utilisateur aura la possibilité de consulter une notice explicative afin de l'aider à remplir correctement sa demande.

Une fois validée, la demande fera partie du workflow de gestion des demandes. Le porteur, les représentants du CARI et le président pourront suivre ces demandes, voir dans quel état elles sont, et effectuer des actions dessus (en fonction des droits des utilisateurs). Un système de notification de mails permettra de prévenir les différents acteurs concernés par la demande des changements d'états.

#### b. Suivi des budgets

Le président aura la possibilité de consulter tous les budgets de chaque composante. Initialement, chaque composante a un budget, que le président renseigne. Il peut à tout moment modifier ce budget (car chaque année, les budgets sont revus).

Ce budget sera modifié en fonction des dépenses d'investissements réalisées. Il sera possible de visualiser sur une page tous les budgets de chaque composante et de voir la somme des dépenses de chaque budget de l'année en cours. L'utilisateur pourra voir quels sont les budgets qui ont été épuisés.

#### c. Liste des demandes

Les utilisateurs auront accès à la liste des demandes en fonction de leurs droits :

- Le président : toutes les demandes
- Le représentant : les demandes de son UFR
- Le porteur : les demandes qu'il a soumis

Les utilisateurs pourront ainsi voir une liste succincte des demandes. Ils pourront également visualiser le détail de chaque demande.

#### d. Système de gestion des utilisateurs

Le président pourra ajouter des nouveaux utilisateurs en y précisant :

#### MAZELIN-PERNES-VINCENT

- Le nom et prénom
- L'adresse email
- Choisir son rôle : président, représentant ou porteur.
  - Président : il aura tous les droits
  - Représentant : choisir de quel UFR il appartient.
  - Porteur : choisir l'unité de la composante
- Un mot de passe lui sera envoyé par email

### III. Technologies

Afin de réaliser cette plateforme, concernant le langage de programmation, le choix se porte sur l'utilisation d'un Framework réalisé en Python : Web2Py.



- **Python** : c'est un langage de programmation orienté objet. Il est doté d'une gestion de la mémoire (appelé garbage collector) et d'un système d'exception. De plus, il est multi plateforme : Windows, Unix, Mac, Android et IOS. De nombreuses bibliothèques existent sur le web.



- **Web2py** : Framework libre réalisé en Python. Il ne nécessite pas de fichiers de configuration, il est assez simple à prendre en main, et une documentation (en anglais) est disponible sur internet. Il suit le modèle MVC (Modèle, Vue, Contrôleur). Les vues seront des pages au format HTML avec l'utilisation de CSS et de Javascript.

Le serveur web2py offre une interface d'administration s'ouvrant au sein du navigateur. Sur cette interface, il est possible de créer plusieurs applications web. Lors de la création d'application, une hiérarchie du code est déjà proposée : une section modèle, vue,

MAZELIN-PERNES-VINCENT

contrôleur, ressources, etc. L'environnement de développement est alors installé, la création et la modification du code se fait directement via le serveur web2py.



- **MySQL** : système de gestion de base de données sous licence GPL, très utilisé, fiable et gratuit.



- **Git Hub** : application de gestion de versions. Il permet de partager le code et des documents entre les acteurs du projet.

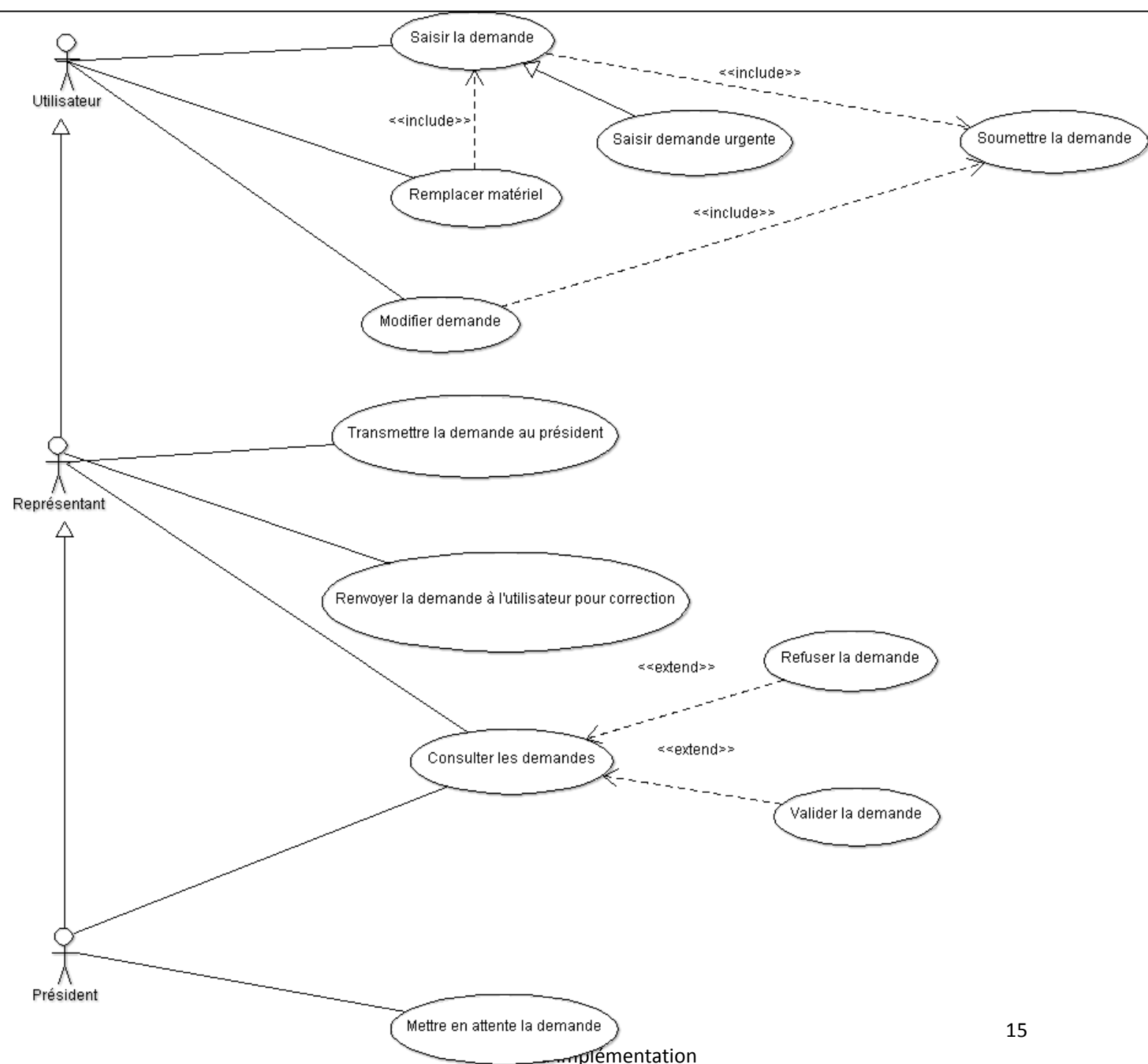
L'utilisation du Framework Web2py va permettre au client d'assurer la maintenabilité de l'application après sa livraison.

## IV. Diagrammes UML

### 1. Diagrammes de Cas d'Utilisation

Pour décrire notre projet, nous avons fait trois diagrammes décrivant les cas d'utilisations qui devraient se produire le plus fréquemment. Nous allons passer en revue chacun d'entre eux.

#### a. Gestion des demandes CARI



Pré condition: l'utilisateur est connecté

## Utilisateur → Saisir la demande

Ce cas concerne la saisit d'une demande par un demandeur

### Cas nominal

1. L'utilisateur saisit les champs suivants : L'intitulé, description etc. Si c'est un remplacement d'un matériel, cas alternatif 1
2. Si c'est une demande urgence, cas alternatif 2
3. Il saisit ensuite l'annexe budgétaire : il saisit le(s) produit(s) désiré(s) avec leurs prix
4. Sauvegarde de la demande
5. Il peut exécuter le cas suivant : **Soumettre la demande**
6. Une notification par mail est envoyée au représentant de l'entité

### Cas alternatif 1

- 1.1 . l'utilisateur devra saisir un la référence du matériel

### Cas alternatif 2

- 2.1. L'utilisateur devra décrire le caractère urgent de la demande

### Cas d'exception

- 3.1. Des erreurs sont survenues lors de la soumission de la demande : champs vides, ou non respect des expressions régulières...

## Utilisateur → Soumettre la demande

Ce cas concerne la soumission d'une demande déjà saisie

Pré condition : une demande doit exister

### Cas nominal

1. L'utilisateur clique sur « soumettre la demande »
2. Une notification par mail est envoyée au représentant

## Utilisateur → Modifier la demande

Ce cas concerne la modification d'une demande existante



Pré condition : la demande a été renvoyée par le représentant pour correction

#### Cas nominal

1. L'utilisateur change les valeurs qu'il souhaite modifier
2. Il sauvegarde sa demande
3. Il peut exécuter le cas suivant : **Soumettre la demande**

#### Cas exception

- 2.1. Des erreurs sont survenues lors de la soumission de la demande :  
champs vides, ou non-respect des expressions régulières

### **Représentant → Consulter les demandes**

Ce cas concerne la consultation des demandes de l'entité du représentant

#### Cas nominal

1. Consulter toutes les demandes au sein de son entité
2. Accéder aux détails de chaque demande : informations et annexe budgétaire
3. Il peut exécuter les cas suivants : **valider, refuser, transmettre au président** ou **renvoyer la demande**

### **Représentant CARI → Valider la demande**

Ce cas concerne la validation d'une demande

Pré condition : Il a consulté la demande concernée

#### Cas nominal

1. Il valide la demande.
2. Le porteur du projet est notifié.

### **Représentant CARI → Transmettre la demande au président**

Ce cas concerne la transmission d'une demande au président

Pré condition : La demande a été vérifiée et a été identifiée comme non courante.

Cas nominal

1. Après avoir identifié la demande comme étant non courante, le représentant va la transmettre au président, pour avoir son avis ou pour qu'elle puisse passer en séance CARI.
2. L'utilisateur et le président sont notifiés par mail.

**Représentant CARI → Renvoyer la demande au porteur pour correction**

Ce cas concerne le renvoi d'une demande en cas de problèmes

Pré condition : Le représentant a analysé la demande émanant de l'utilisateur.

Cas nominal

- 1) Il va renvoyer le dossier au porteur concerné pour que celui-ci apporte les corrections nécessaires.
- 2) Le porteur est notifié par mail que le dossier lui est retourné, avec un commentaire pour lui préciser les informations manquantes.

**Représentant → Refuser la demande**

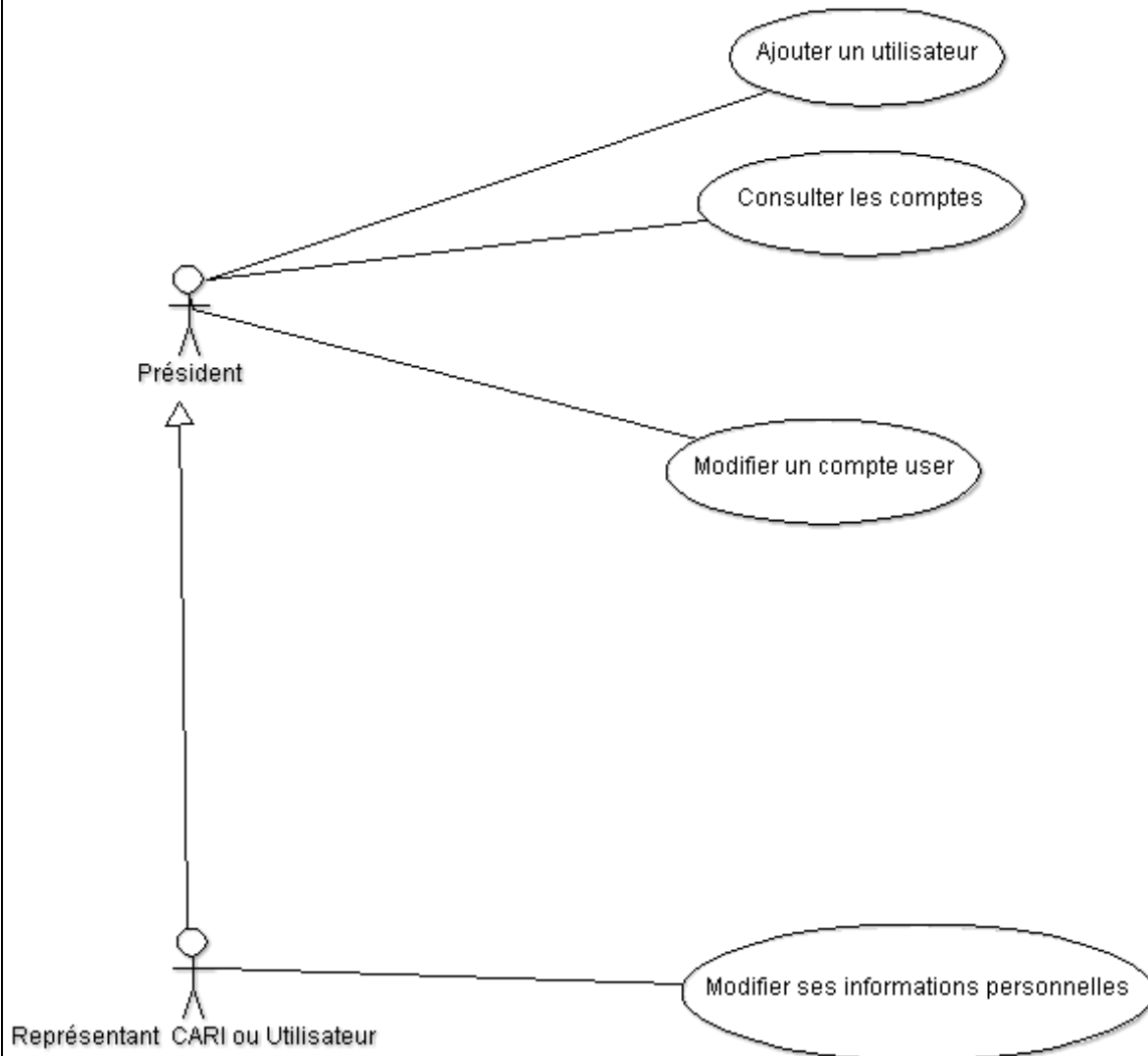
Ce cas concerne le refus d'une demande

Cas nominal

1. Le représentant refuse une demande
2. Une notification de refus de la demande est envoyée au porteur du projet

b. Gestion des utilisateurs

**UC : Gestion des utilisateurs**



Pré condition: l'utilisateur est connecté en tant que président.

**Président → Ajouter un utilisateur**

Ce cas concerne l'ajout d'un utilisateur sur l'application

### Cas nominal

1. Le président choisit le rôle de l'utilisateur : utilisateur, représentant, gestionnaire ou président.
2. Il saisit ensuite les informations relatives au nouvel utilisateur : login, mail, son entité...
3. Il enregistre l'utilisateur
4. Un mail est envoyé à ce nouvel utilisateur avec ces informations et un mot de passe généré automatiquement

### Cas exception

- 2.1. Des erreurs sont survenues lors de l'enregistrement d'un utilisateur comme un champ vide, ou un code postal incorrect...

## **Président → Consulter les utilisateurs**

Ce cas concerne la consultation des utilisateurs sur l'application

### Cas nominal

1. Le président peut consulter tous les utilisateurs de l'application
2. Il peut accéder à chaque fiche utilisateur
3. Il peut exécuter le cas suivant : **Modifier l'utilisateur**

## **Président → Modifier un utilisateur**

Ce cas concerne la modification d'un utilisateur existant

### Cas nominal

1. Le président peut modifier les informations de l'utilisateur
2. Enregistrer les modifications

### Cas exception

- 2.1 . Des erreurs sont survenues lors de l'enregistrement d'un utilisateur comme un champ vide, ou un code postal incorrect...

## **Représentant et utilisateur → Modifier informations personnelles**

Ce cas concerne la modification des informations personnelles des utilisateurs

Pré condition : être connecté en tant qu'utilisateur ou représentant

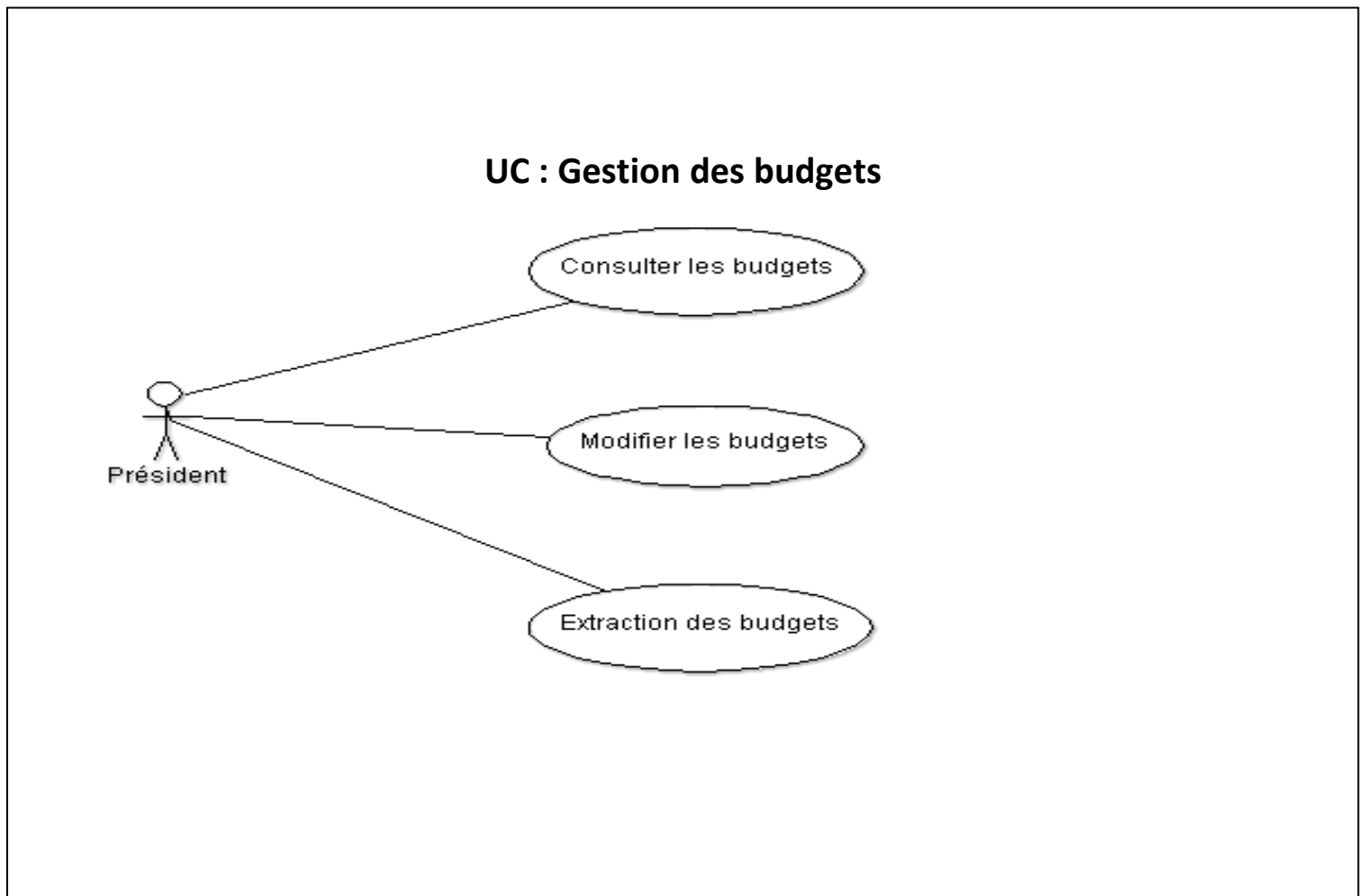
Cas nominal

1. L'utilisateur peut modifier ses informations (adresse, mail...)
2. Enregistrer les modifications

Cas exception

- 1.1 . Des erreurs sont survenues lors de l'enregistrement d'un utilisateur comme un champ vide, ou un code postal incorrect...

c. Gestion des Budgets



Pré condition: l'utilisateur est connecté en tant que président.

## Président → Consulter les budgets

Ce cas concerne la consultation des budgets

### Cas nominal

1. Le président accède à la liste des budgets.
2. Il pourra cliquer sur l'un d'entre eux pour accéder au détail
3. Il accède à l'historique des achats qui ont été imputés sur ce budget
4. Il peut exécuter le cas : **Modifier le budget**

## Président → Modifier le budget

Ce cas concerne la modification d'un budget

Pré condition : Il a consulté la page des budgets

### Cas nominal

1. Il va choisir le budget.
2. Il peut modifier la valeur du budget
3. Enregistre la modification

### Cas exception

- 3.1. La valeur qu'il a renseignée donne une erreur (Lettres au lieu d'un nombre...).
- 3.2. Il la rectifie puis valide de nouveau.

## Président → Extraire les budgets

Ce cas concerne l'extraction des budgets

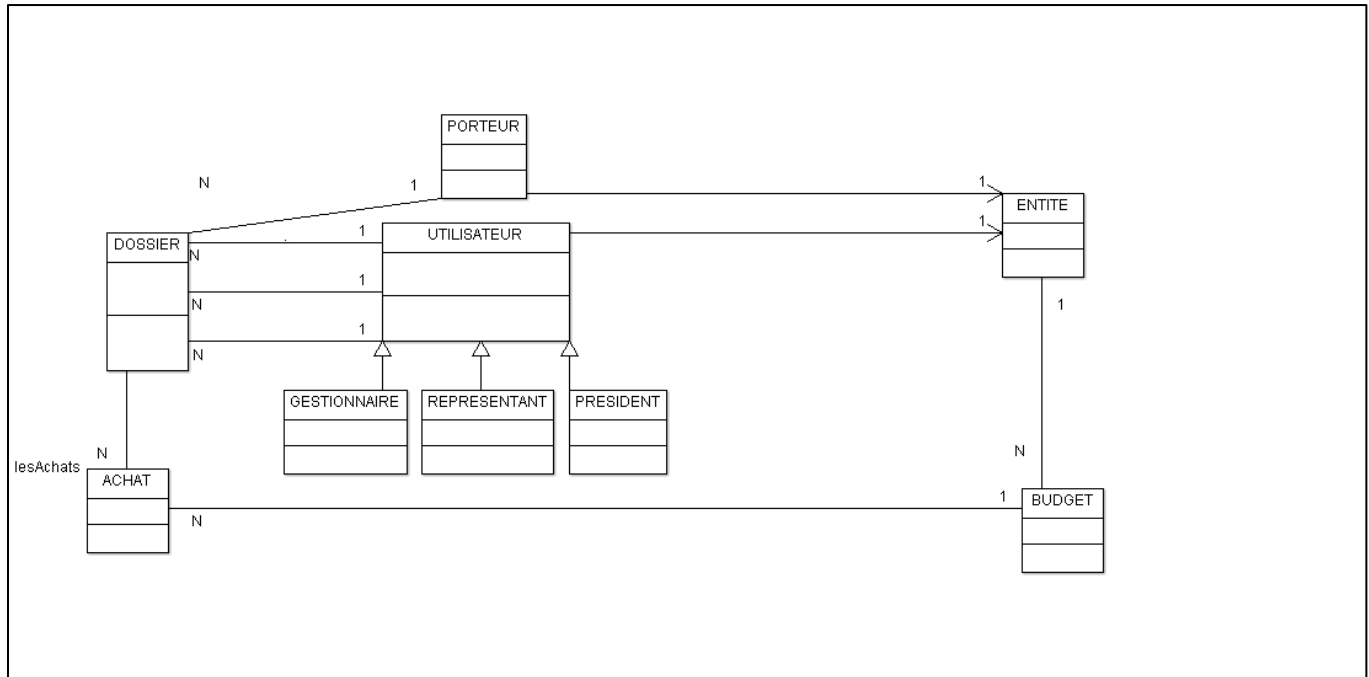
Pré condition : Il a consulté la page des budgets

### Cas nominal

1. En étant sur la page des budgets, il pourra cliquer sur l'icône pour extraire les données relatives aux budgets
2. Choisir l'emplacement du fichier

## 2. Diagramme de Classe

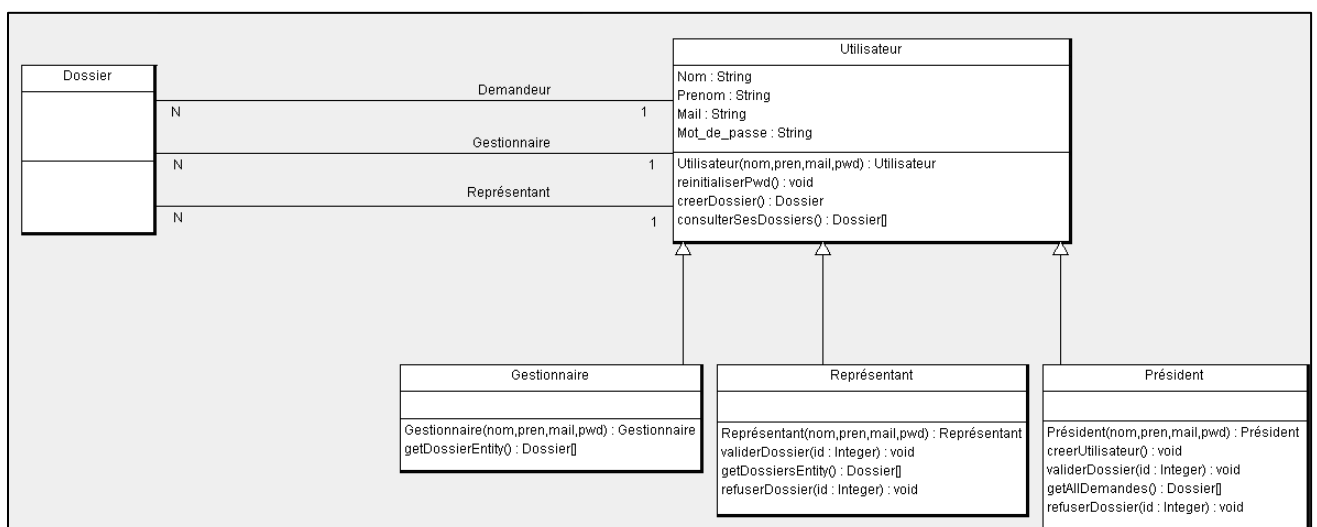
Pour décrire l'architecture de notre projet, nous vous présentons notre diagramme de Classe (sans les attributs dans un premier temps) ci dessous :



Pour voir plus clair dans notre diagramme de classe, nous allons nous intéresser à chacune des parties.

### a. Héritage sur l'Utilisateur

Pour représenter les différents types d'utilisateurs, nous avons utilisé un héritage :



Pour distinguer chaque type d'utilisateur de l'application, nous avons créé trois classes qui héritent d'Utilisateur :

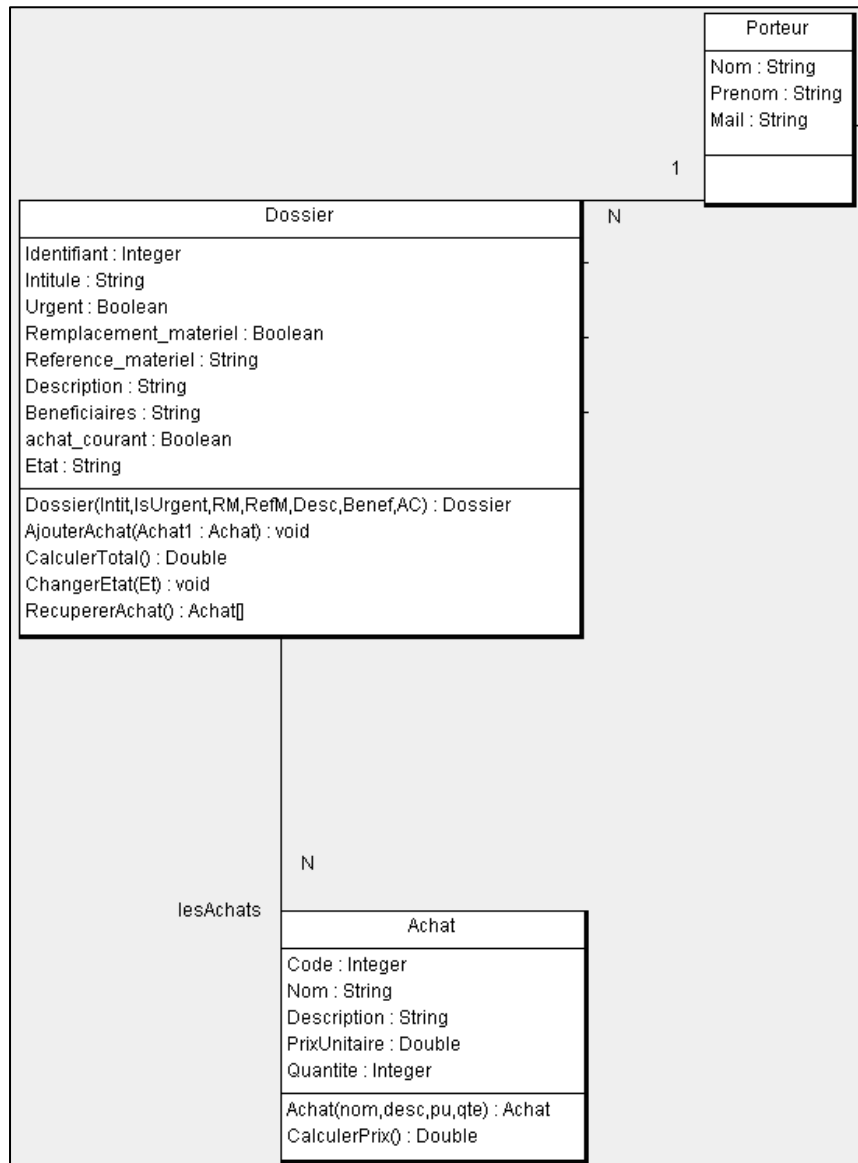
L'utilisateur : il possède les droits de base pour accéder à l'application. Ensuite, trois autres profils disposeront de ces mêmes droits avec des droits supplémentaires :

- Le gestionnaire : En plus des droits de l'utilisateur, il peut accéder à l'ensemble des dossiers de son entité.
- Le représentant : En plus des droits du gestionnaire, il pourra valider ou refuser des dossiers.
- Le président : En plus des droits du représentant il va pouvoir créer de nouveaux utilisateurs et il va pouvoir consulter tous les dossiers présents dans l'application.

Selon les profils des personnes, ils peuvent être rattachés à des dossiers en tant que Demandeur, Gestionnaire ou Représentant.



b. Le Dossier et les produits qu'il contient

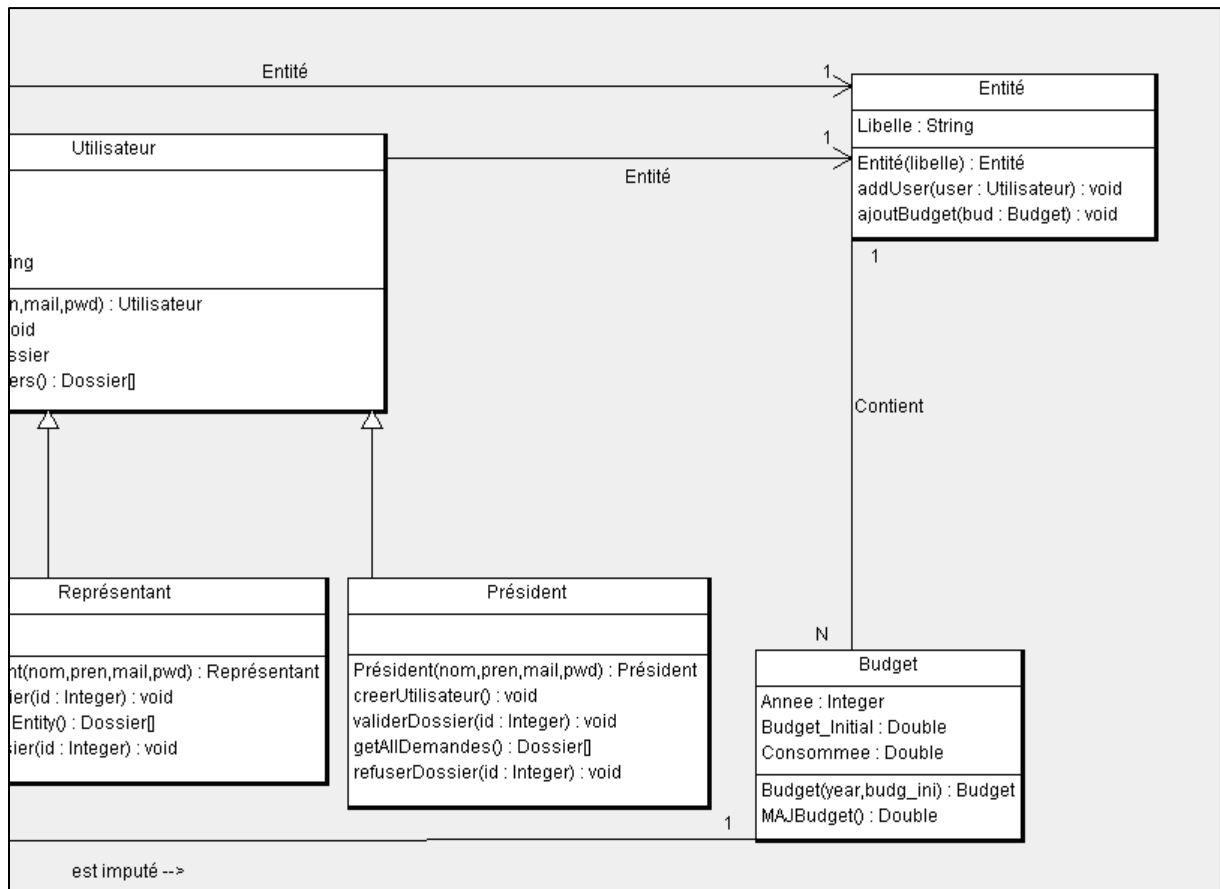


Ici, nous voyons que le Dossier contient les demandes formulées auprès du CARI. Le dossier contient toutes les formations pertinentes pour l'évaluation de ce dernier comme la description de la demande, les bénéficiaires et s'il s'agit d'un remplacement de matériel ou non. De plus, nous avons un attribut Etat qui représente le workflow du dossier. Nous donnerons plus détails sur les valeurs que peut prendre cet attribut dans le Diagramme d'Etat.

Il est relié à la classe Achat qui elle, référence les produits qui constitueront la demande. Ainsi, le dossier contiendra une liste des achats.

Nous avons choisi de distinguer le Porteur du projet par une classe particulière car ce dernier n'a besoin d'accéder à l'application.

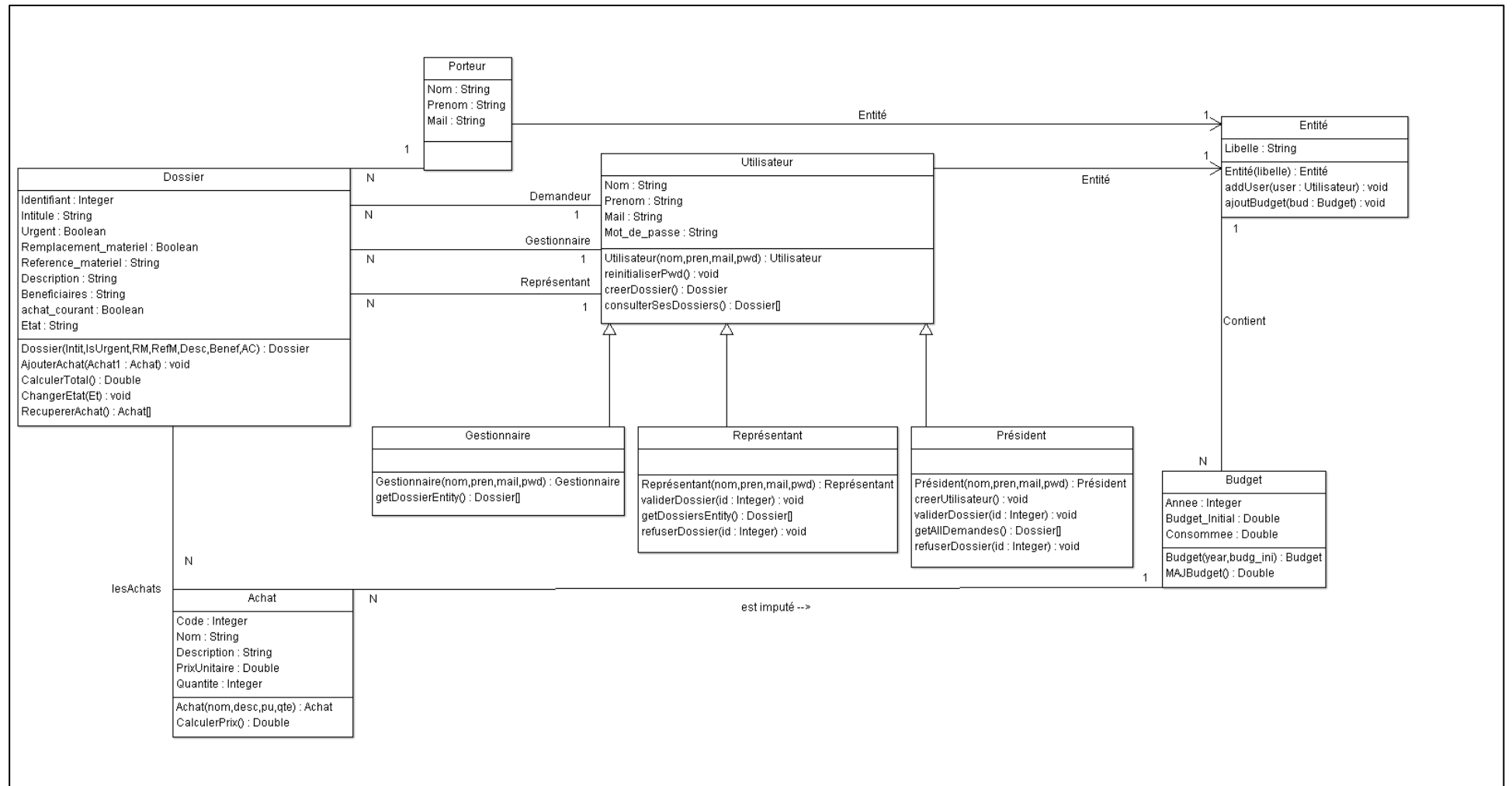
c. Entité et Budget



Ci-dessus, nous voyons que chaque utilisateur (le porteur lui aussi) est rattaché à une entité. Chaque entité possède un ou plusieurs budgets (un par année). Pour montrer que l'achat d'un produit est imputé sur le budget d'une entité, nous avons fait un lien entre budget et la classe Achat.

Pour conclure sur le diagramme de classe, nous vous présentons notre diagramme de classes au complet sur la page suivante :

## Diagramme de classe

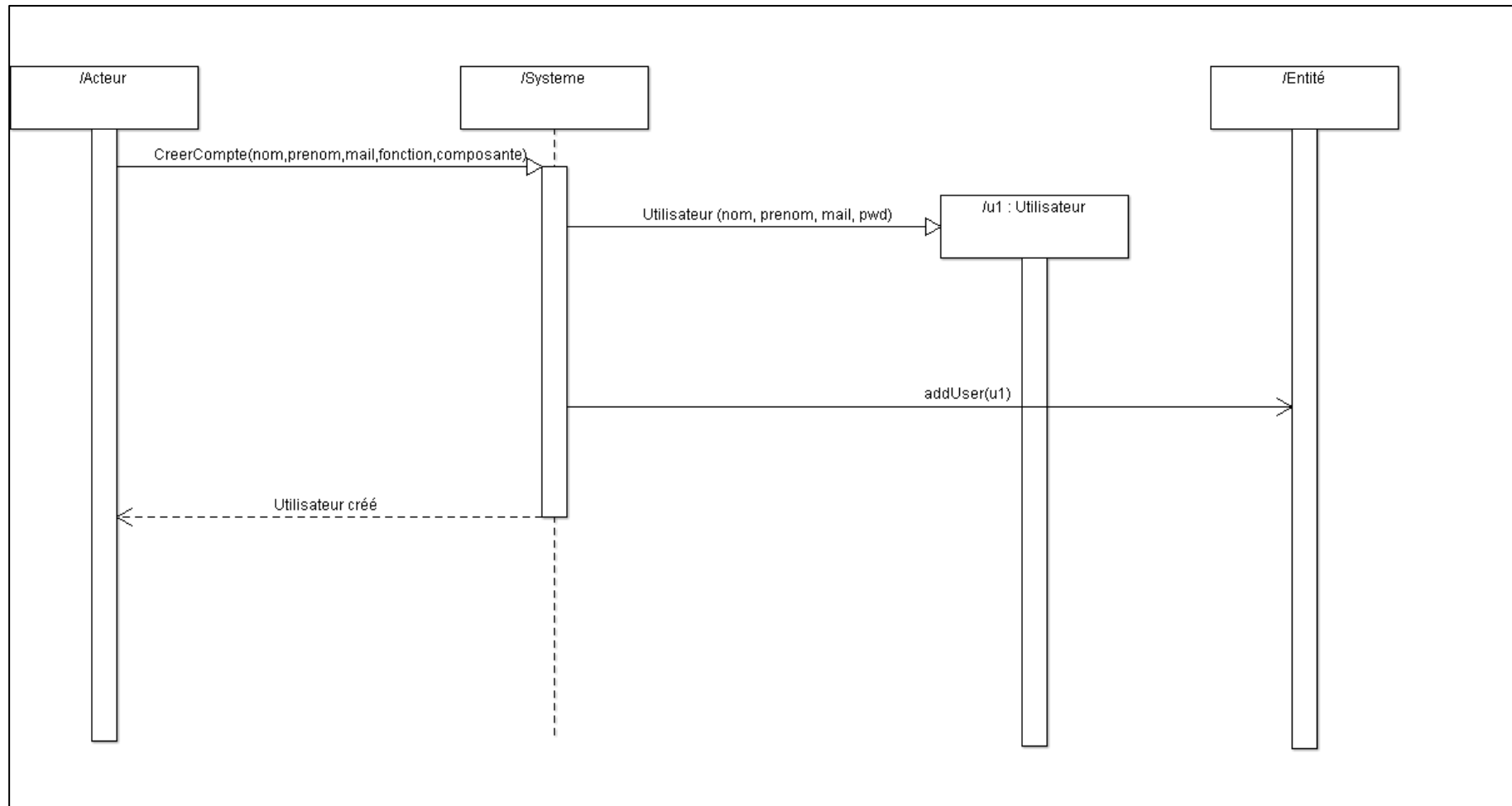


### 3. Diagrammes de Séquences

Pour décrire quelques scénarios d'utilisation de notre projet, nous avons élaboré trois diagrammes de séquences.

#### a. Création d'un nouvel utilisateur

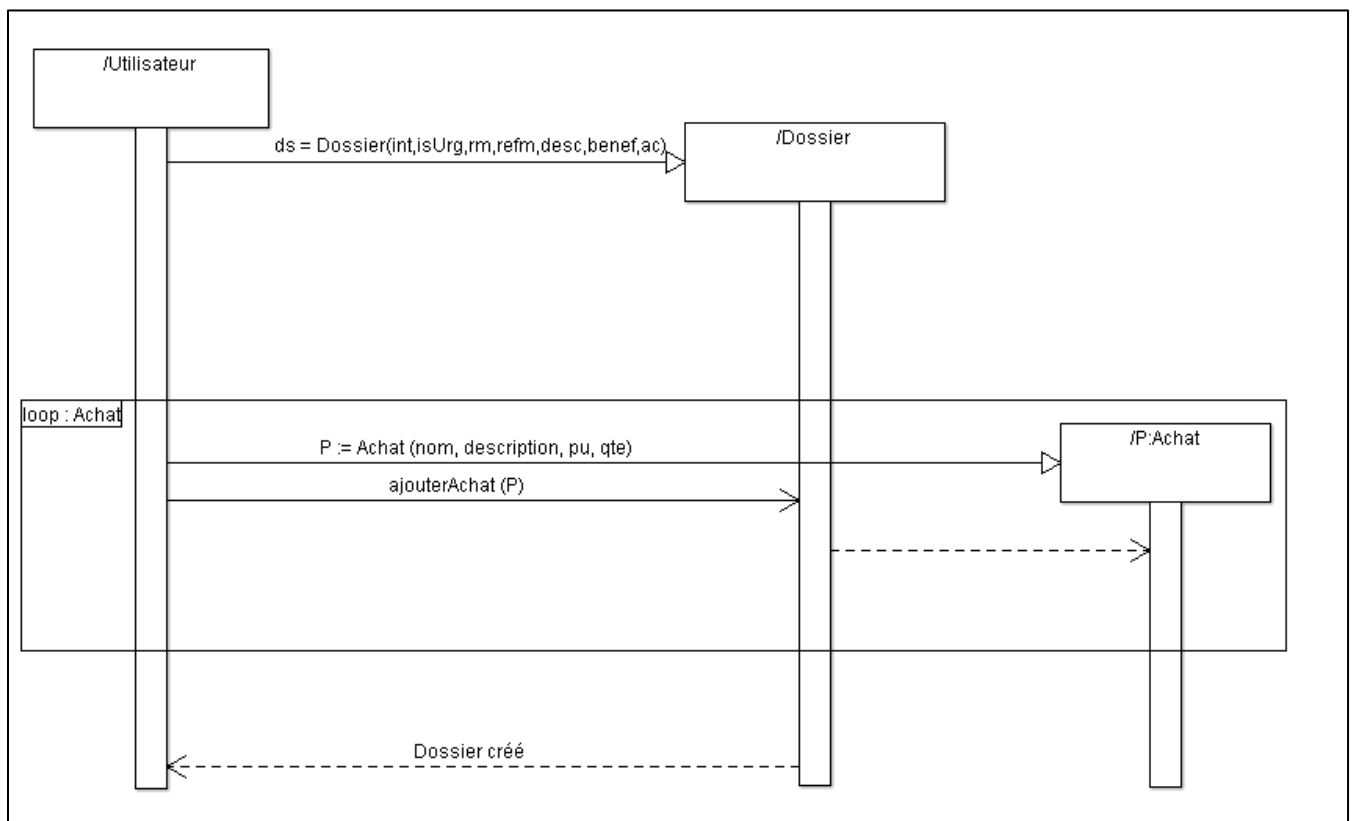
## Ajout d'un utilisateur



Pour créer un nouveau compte, l'administrateur de l'application (Acteur) va donner les informations relatives à ce nouvel utilisateur comme le nom, le prénom, mail, sa fonction et son entité d'appartenance. La fonction de l'utilisateur nous permettra de savoir quel type de profil il faudra lui attribuer : simple utilisateur, gestionnaire, représentant ou président. Ainsi, une nouvelle instance d'Utilisateur sera créée.

Ensuite, nous allons affecter cet utilisateur à l'entité correspondante. Enfin nous allons renvoyer un message à l'acteur pour l'informer que l'utilisateur a bien été créé.

#### b. Création d'une demande

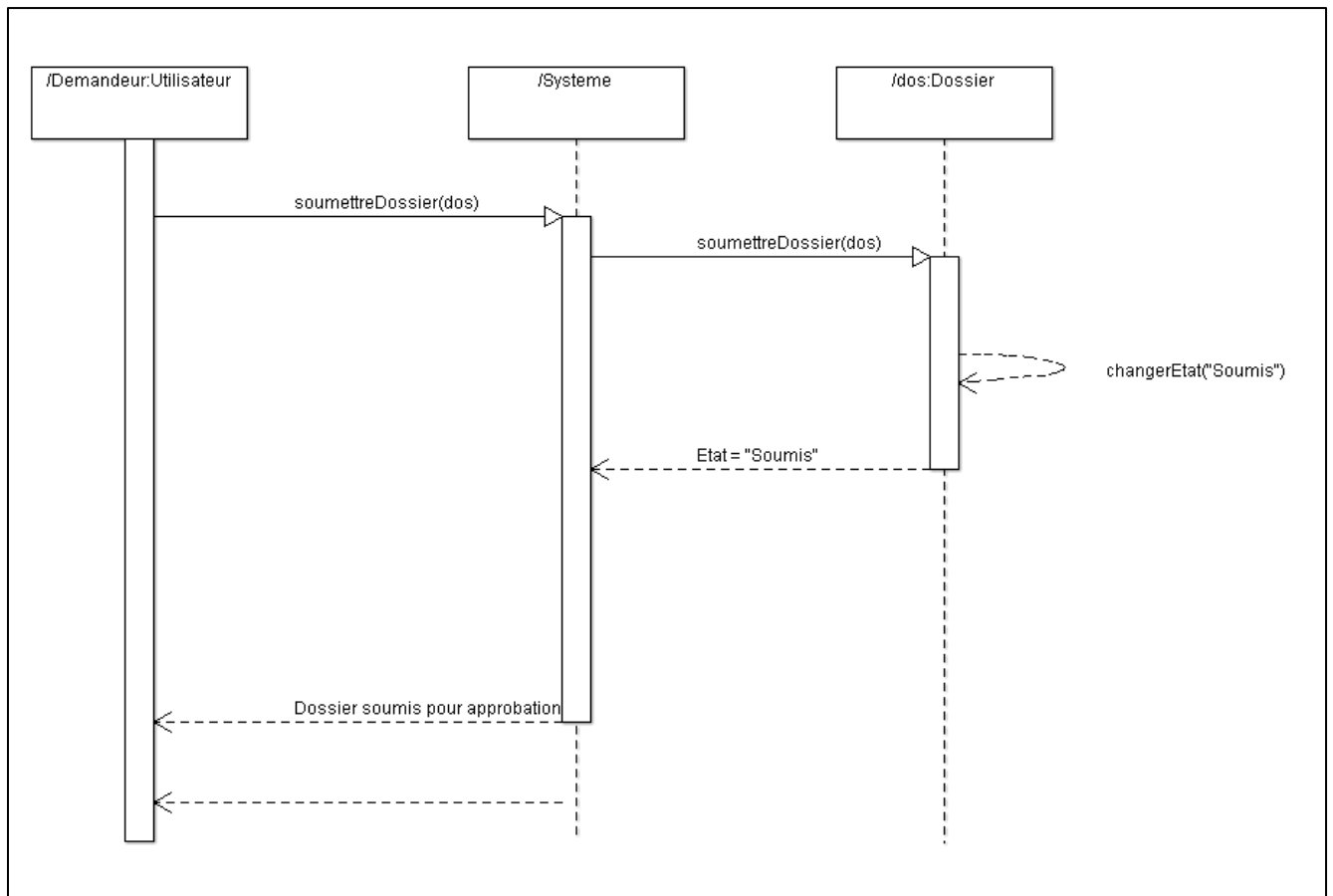


Ici, nous sommes dans le cas où l'utilisateur veut formuler une demande auprès du CARI. Pour cela, il va créer un dossier dans lequel il va renseigner toutes les informations relatives à sa demande :

- l'intitulé de sa demande (paramètre int);
- Si elle est urgente ou pas (paramètre IsUrg);
- Il précise si c'est un matériel à remplacer ou pas (paramètre rm);
- Si c'est le cas, il précise la référence du matériel obsolète (paramètre refm);
- Une description de son projet, son but (paramètre desc);

Une fois que le dossier est créé, nous pourrions ensuite y ajouter des achats. C'est ici que nous avons une boucle sur les différents produits que nous souhaitons ajouter. A la fin nous aurons le montant total du projet.

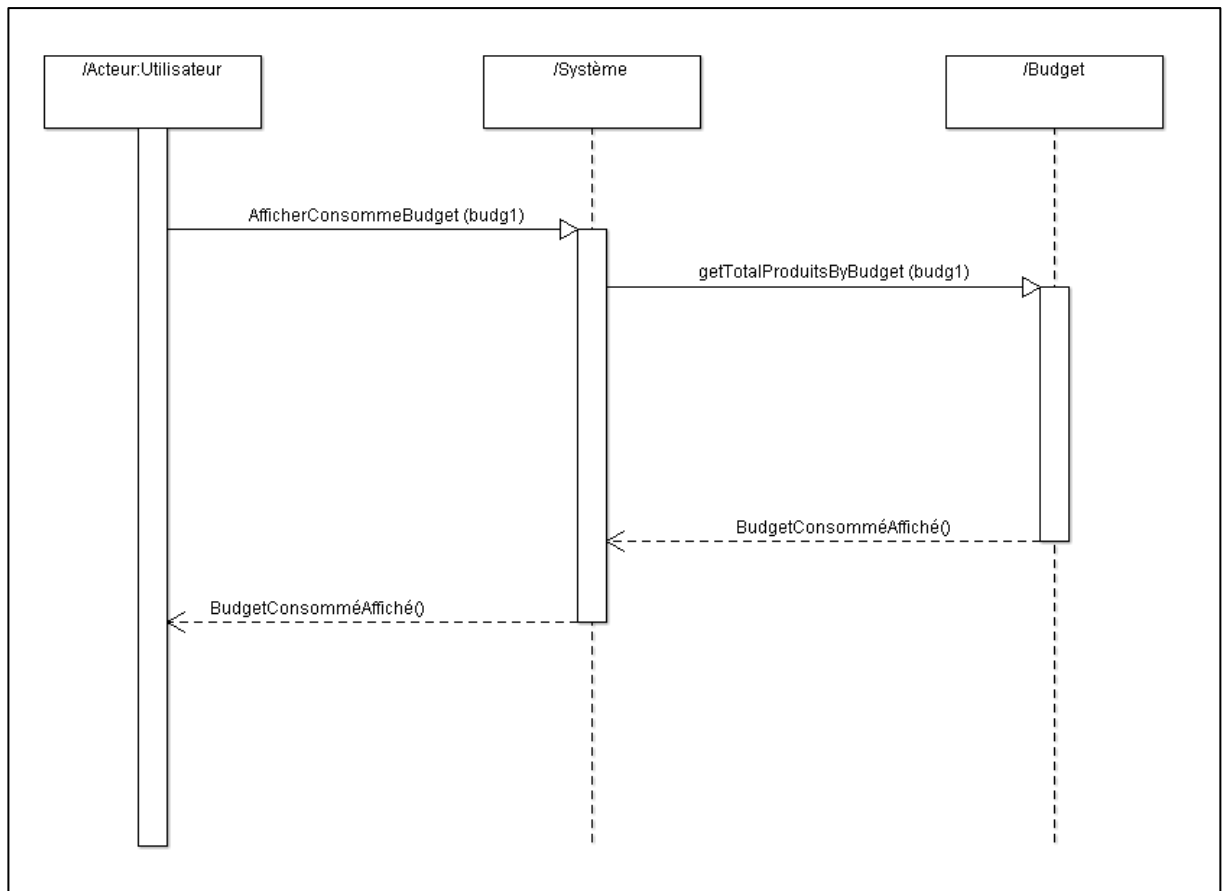
c. Soumission d'une demande



Lorsque l'utilisateur a renseigné toutes les informations dans son dossier, il peut le soumettre pour qu'il puisse être examiné. Lorsqu'il est soumis, l'état du dossier va passer de Brouillon à Soumis. L'utilisateur perd les droits de modifications dessus. Il n'y a que le Représentant ou le Président qui peuvent le modifier.

Ensuite l'utilisateur reçoit un message lui informant que son dossier à bien été soumis. Le représentant et le gestionnaire rattaché à ce dossier seront également notifiés par mail.

d. Afficher le budget consommé

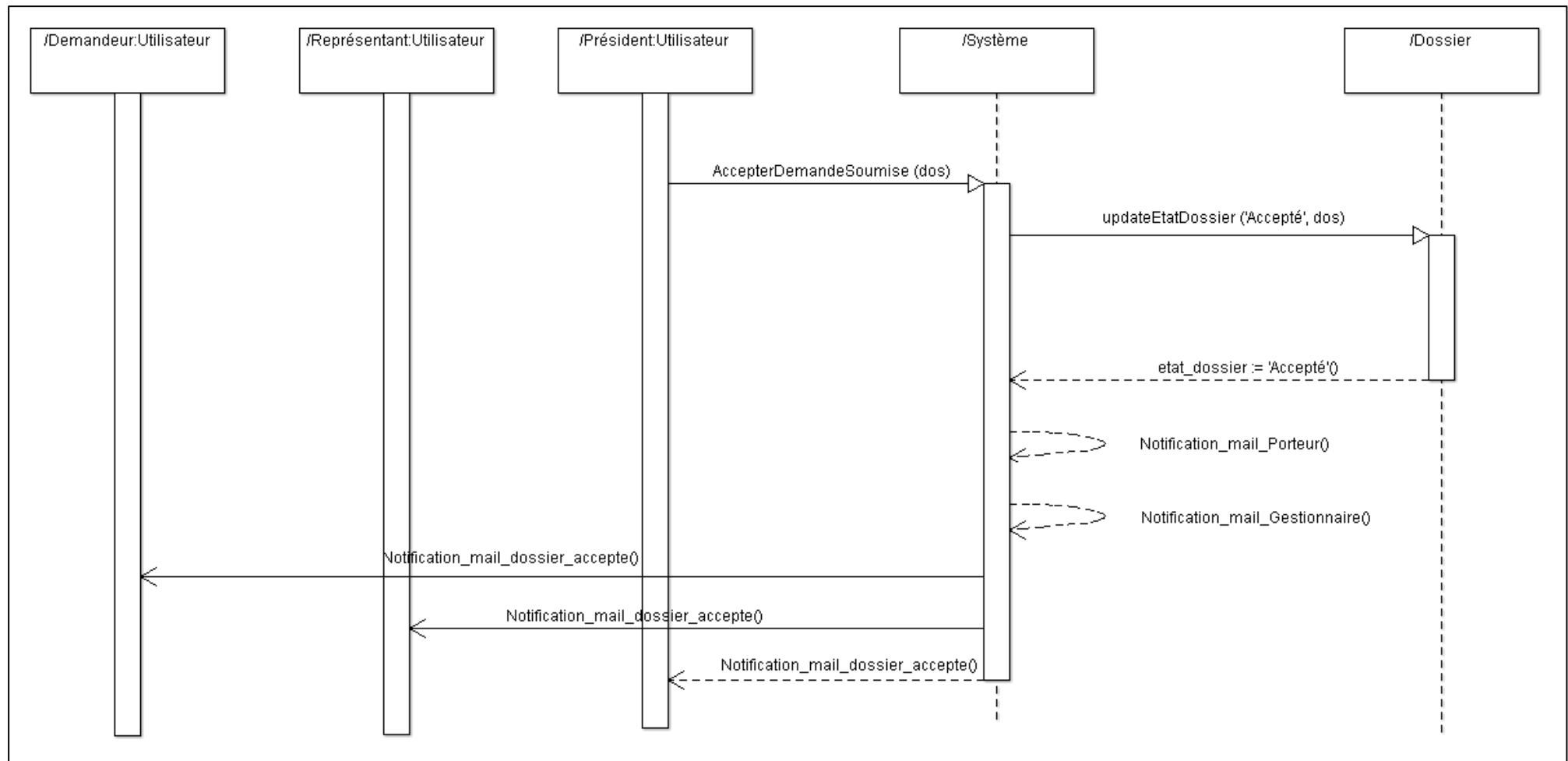


Pour connaître le montant total consommé sur un budget donné, l'utilisateur se rend sur la liste des budgets. Il va cliquer sur un lien qui va l'amener sur une page de détails du budget choisi.

Dans le logiciel, une méthode nommée *getTotalProduitsByBudget(budg1)* va récupérer la liste des produits imputés sur le budget en cours et calculer le montant total. Ainsi, sur le nouvel écran ('Détails Budget'), différentes informations vont s'afficher dont le budget initial, le montant total des produits imputés sur ce budget. Il aura également un pourcentage de consommation du budget qui lui donnera une idée du degré d'utilisation de son portefeuille.

e. Accepter un dossier soumis





A l'issue d'une séance CARI, un dossier a été définitivement accepté, il nous faut donc mettre à jour la demande dans l'application.

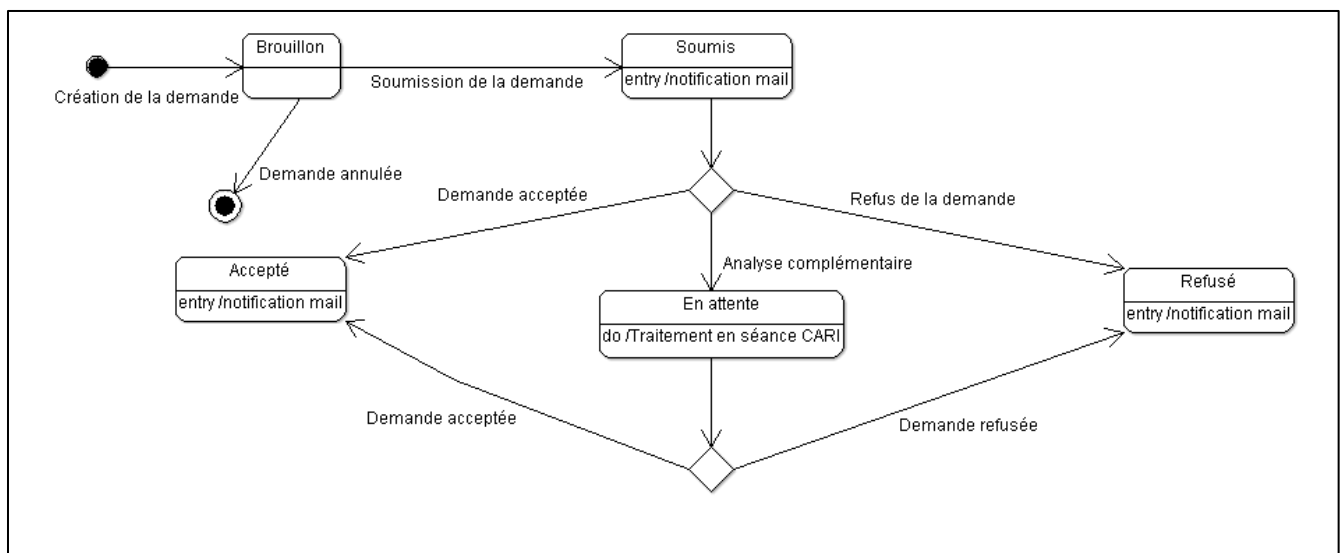
Pour cela, le Président s'identifie et va sur la liste des projets. Il choisi le dossier concerné et change son statut à « Accepté ». Dans le logiciel, la méthode `updateEtat('Accepté',dos)` va mettre à jour l'état du dossier. Après cette mise à jour, un mail de confirmation est envoyé aux principaux acteurs du dossier :

- Le Président
- Le Demandeur initial
- Le Représentant
- Le Porteur du projet
- Le Gestionnaire

Le gestionnaire et le porteur du projet n'ont pas de compte dans l'application. Néanmoins, leurs mails sont renseignés au niveau des propriétés du projet.

Dans cette partie, nous avons décrit les principaux scénarios d'utilisation de l'application. Intéressons-nous maintenant aux différents états du workflow.

#### 4. Diagramme d'Etat



Sur ce diagramme, nous voyons que le dossier peut prendre quatre états différents. Nous allons expliquer chacun d'entre eux :

Lors de la création de la demande, le dossier est à l'état Brouillon. C'est l'utilisateur qui peut le modifier, il va donc rentrer les informations relatives à son projet. S'il décide d'abandonner son projet, il peut effacer le dossier.

Lorsqu'il est soumis, le représentant et le gestionnaire seront notifiés par mail. L'utilisateur n'a plus les droits de modifications dessus. Seul le représentant, le gestionnaire et le président peuvent intervenir dessus. A cet état, il peut y avoir trois cas de figure :

Si le projet est approuvé, le dossier passe à l'état Approuvé et l'utilisateur est alors informé par mail.

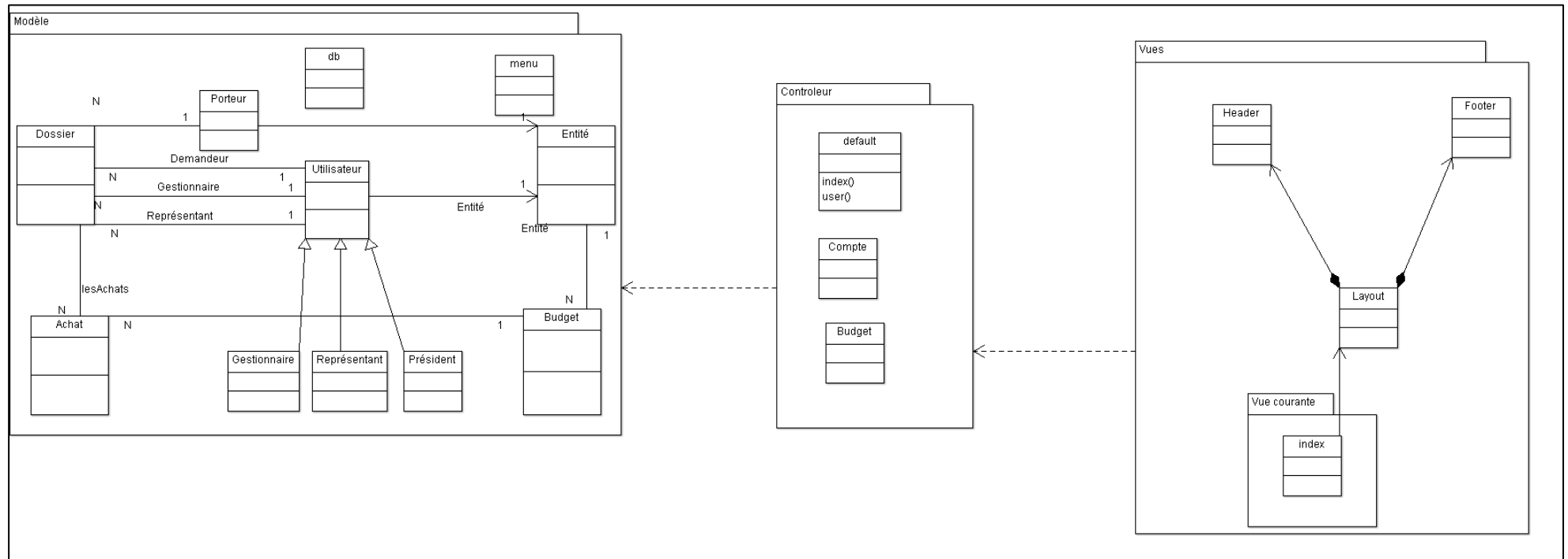
S'il est refusé, le dossier passe à l'état Refusé et l'utilisateur est également notifié.

Si le dossier requiert une analyse approfondie, il passera à l'état En attente, pour qu'il puisse passer en séance CARI. A l'issue de cette séance, le dossier pourra être accepté ou refusé.

Nous pouvons également noter que si le dossier apparaît comme incomplet lors de la soumission, le représentant peut le renvoyer vers l'utilisateur pour correction. Ainsi il reviendra à l'état Brouillon.

## 5. Diagramme de Package

## Diagramme de Package

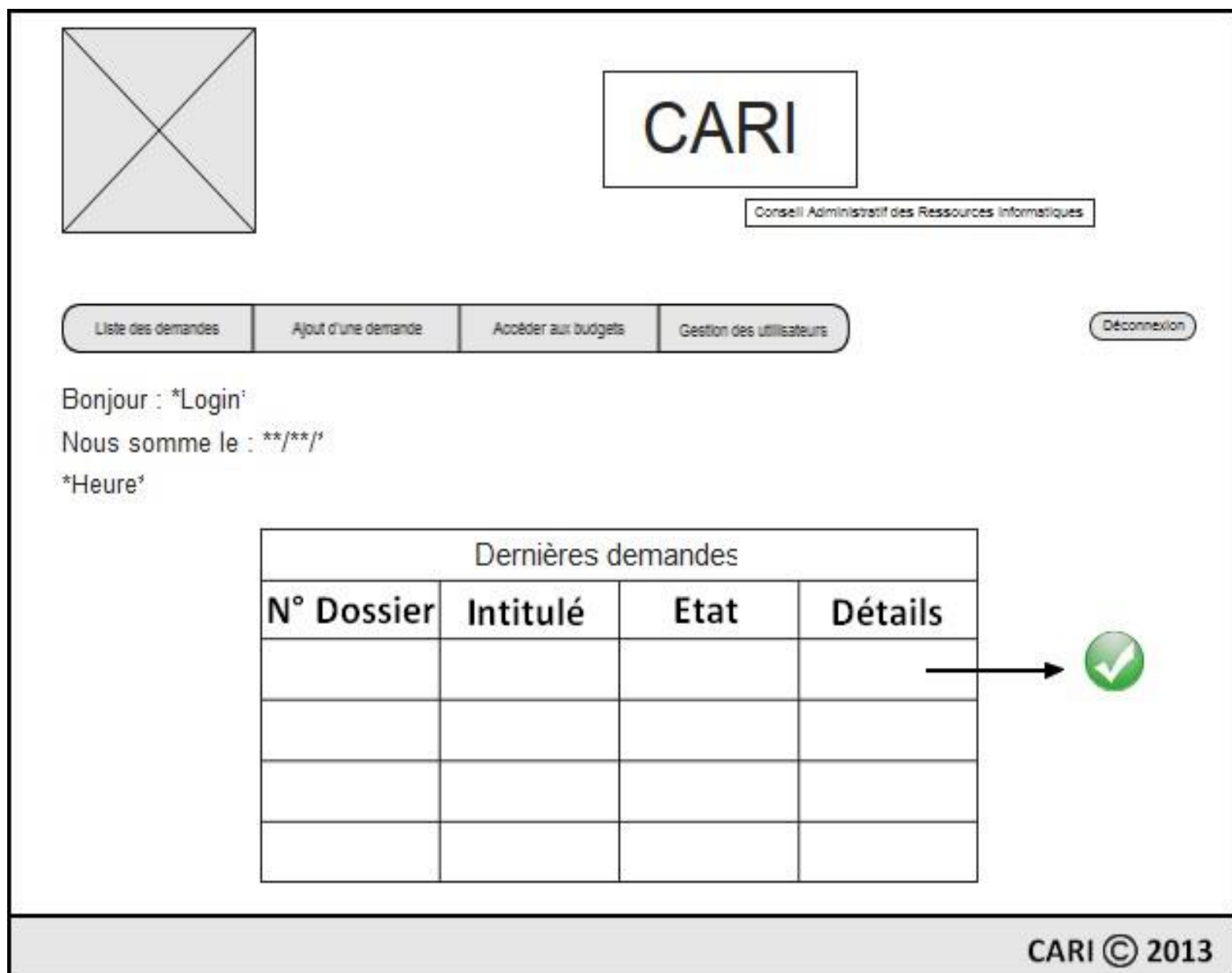


Pour représenter notre diagramme de package, nous nous sommes appuyés sur un patron de conception appelé MVC (Modèle Vue Contrôleur). Ce pattern permet de bien structurer une application en distinguant les données (modèle), la présentation de ces données (vue) ainsi que gestion des événements (contrôleur). L'une des raisons pour lesquelles nous avons structuré notre diagramme de package en suivant ce pattern, est qu'il nous apporte une certaine clarté dans l'architecture de notre application.


Ainsi, il nous permet de distinguer clairement les packages suivants :

- **Modèle** : Comme nous pouvons le constater, il constitue notre diagramme de classe expliqué précédemment.
- **Contrôleur** : Ce package contient des classes qui vont faire l'interface entre la vue et le modèle. Nous pouvons le considérer comme le chef d'orchestre.
- **Vue** : Il contient les différentes vues de notre application.

## V. Maquettes



Cette première maquette représente la page d'accueil de l'application. Le tableau contient les dernières demandes créées. Un bouton détail permet d'accéder au dossier et à l'annexe budgétaire de chaque demande. Le menu dépend du rôle de l'utilisateur. Dans la maquette, nous avons la vue d'un utilisateur connecté en tant que « président ».

Budget				
CARI				
Liste des Portefeuilles				
	<u>UFR / Composante</u>	<u>Budget Alloué</u>	<u>Montant consommé</u>	<u>Etat du Budget</u>
<input type="radio"/>	UFR LAM	200 000	170 000	
<input type="radio"/>	Labo IBISC	500 000	450 000	
<input type="radio"/>	UFR SFA	250 000	260 000	
<b>TOTAL</b>		<b>950 000</b>	<b>880 000</b>	
		<input type="button" value="Annuler"/>	<input type="button" value="Consulter"/>	

Cette deuxième maquette représente la vue des budgets. On peut accéder à plus de détails : pour chaque budget, on peut visualiser l'ensemble des dépenses qui ont été imputées sur chaque budget.

Dossier de demande au CARI

Intitulé du projet :

Porteur : Nom :

Prénom :

Email :

Composante :

Unité :

Responsable :

Gestionnaire :

Bénéficiaire :

Urgence : ☒ Check bc

Remplacement : ☒ Check bc

Pièce jointe :

Description résumée

description...

Cette dernière maquette représente le formulaire de demande. Certains champs sont remplis automatiquement (Nom, prénom, email, composante, unité) en fonction de l'utilisateur connecté. Sur cette fiche, le porteur pourra décrire au mieux sa demande qu'il veut financer. Ceci permettra au représentant (voir même au Président), de statuer sur le sort de cette demande.



## VI. Implémentation de la solution

### 1. Fonctionnalités développées :

Pour cette application, nous avons implémenté les fonctionnalités suivantes :

- Page de connexion des utilisateurs
- Page de mot de passe oublié
- Page de modification de ses informations personnelles
- Page de création du dossier :
  - Page permettant de renseigner les informations du dossier
  - Page de gestion des produits demandés dans le dossier (avec création, suppression modification des produits)
  - Page de gestion des pièces jointes (création et suppression)
  - Page de validation du dossier
- Système d'envoi de mail à chaque changement d'état à tous les acteurs : président, responsable, gestionnaire porteur et demandeur du dossier.
- Gestionnaire des utilisateurs : création et modification de comptes
- Gestionnaire des entités : Ajout, modification et suppression d'entités
- Liste des projets :
  - Liste de tous les projets avec gestion des droits en fonction des rôles :
    - Président : voit tous les dossiers
    - Représentant des entités : voit tous les dossiers de son entité
    - Demandeur : voit seulement les dossiers qu'il a soumis
  - Recherche de dossier par son numéro de dossier (Id)
- Liste des budgets des entités :
  - Liste des budgets en cours, c'est-à-dire les budgets de chaque entité, dont la date est la plus récente
  - Liste de tous les budgets : affiche tous les budgets (également les anciens)
  - Export CSV des budgets en cours

### 2. Outils utilisés

Nous avons utilisé le langage Python ainsi que le Framework Web2py pour réaliser ce projet. Avant tout, nous avons appris les bases du langage Python. Ensuite, nous avons parcouru la documentation officielle sur le site de Web2py ainsi qu'un livre conseillé par Franck Pommereau : « web2py Application Development Cookbook ». De plus, nous avons parcouru de nombreux forums tels que « stackoverflow.com » et « google.code ».

Nous avons principalement exploité le framework Web2py pour la réalisation de nos écrans. Web2py prend en charge :

- La création de la base de données
- Les tests sur les champs

- La création de formulaires
- L'envoi de mails...

Dans certains cas, nous avons dû utiliser des méthodes des bibliothèques standards de Python comme par exemple :

- Le formatage des dates
- L'arrondi des pourcentages
- Formatage du texte...

Concernant l'exportation CSV, nous avons utilisé une librairie externe écrite en Python nommé XLWT. Elle permet de générer des tableurs : création de feuilles, styles sur le texte etc. Nous nous sommes servis de cette bibliothèque pour générer la liste des budgets en cours.

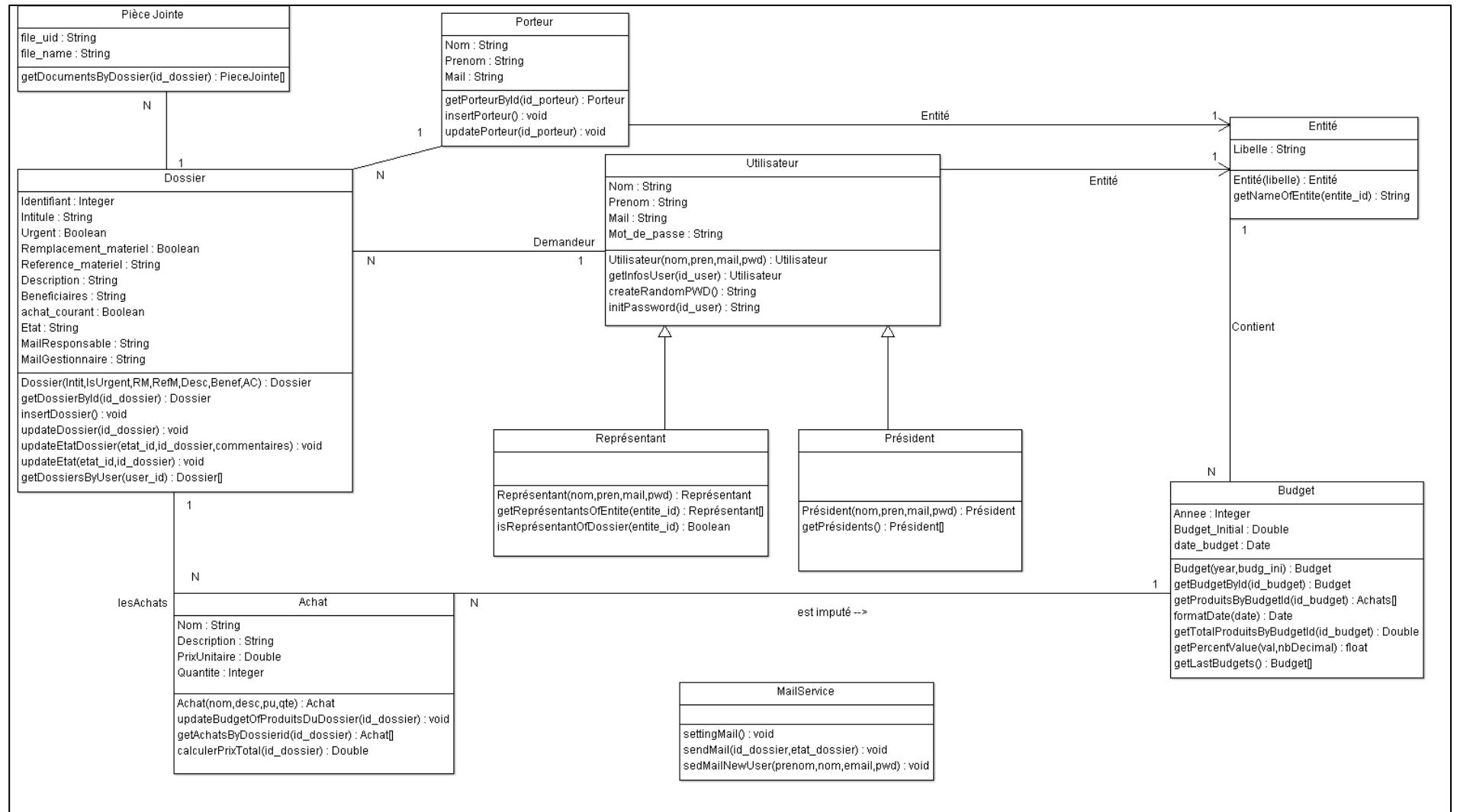
Pour la partie visuelle, nous nous sommes servis d'HTML, CSS ainsi que JQUERY (framework Javascript).

Pour mener à bien notre projet, nous avons utilisé un gestionnaire de version : Github. Cela nous a permis de mutualiser notre code et les documents tels que la documentation, dossier de conception...

### 3. Diagramme de classe modifié

Pour implémenter notre solution, nous avons dû modifier notre diagramme de classe initial. Le nouveau diagramme est présenté sur la page suivante :

Diagramme de classe modifié



Dans ce diagramme de classe, nous avons ajouté les classes suivantes :

- **Pièce jointe** : Car lorsqu'on établit un dossier, il fallait que le demandeur puisse joindre des fichiers. Car plus le dossier est bien renseigné, plus il a de chances d'être bien traité.
- **Mail Service** : Lorsqu'un dossier change d'état (Brouillon → Soumis, Soumis → Accepté...), nous devons notifier les principaux acteurs de la demande. Par exemple, lorsqu'un dossier est accepté, le demandeur, le gestionnaire, le responsable et le président sont notifiés.

Nous avons également ajouté des méthodes à certaines classes. Pour illustrer notre travail, nous allons prendre l'exemple de deux classes modifiées comme :

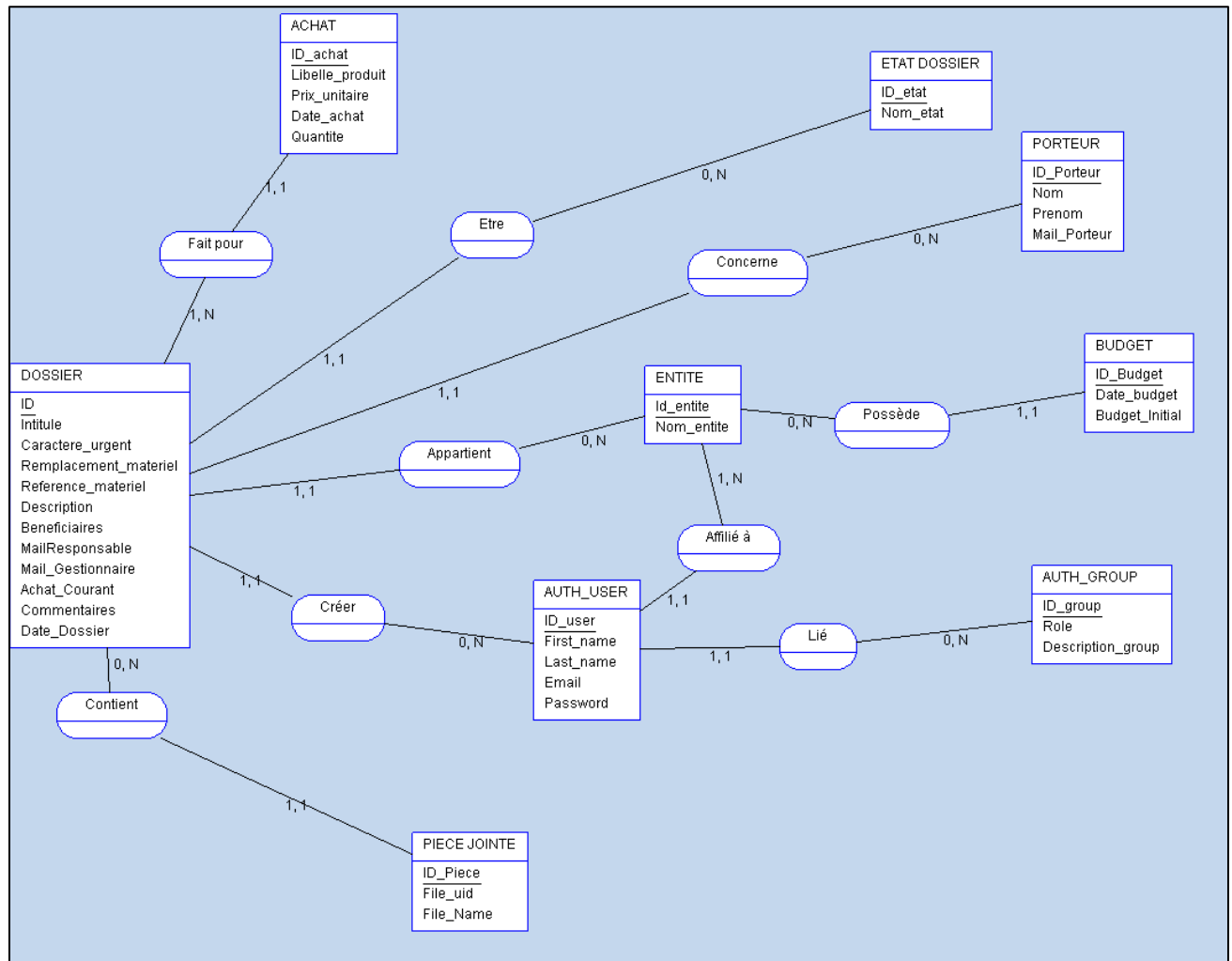
- Classe **Entité** : Nous avons mis une méthode *getNameOfEntite* qui, à partir de l'ID d'une entité, elle va nous renvoyer son libellé.
- Classe **Budget** : Dans cette classe, nous avons mis les méthodes suivantes :
  - *getProduitsByBudgetId(budget\_id)* : Cette fonction nous donne la liste des produits qui ont été imputés sur le budget passé en paramètre.
  - *getLastBudgets()* : Elle nous renvoi la liste des derniers budgets en date pour chacune des entités.
- Classe **Mail Service** :
  - *sendMail(id\_dossier, etat\_dossier)* : A chaque changement d'état de dossier, un mail est envoyé aux principaux acteurs de la demande.
- Classe **Dossier** :
  - *getDossiersByUser(user\_id)* : Obtention des dossiers d'un utilisateur, en se basant sur son ID.

Suite à un point avec Franck POMMEREAU, ce dernier nous a précisé que le gestionnaire n'a pas besoin de compte utilisateur. De ce fait, nous avons supprimé la classe Gestionnaire (ancienne classe fille d'*Utilisateur*). Pour pouvoir notifier cet acteur de tout changement d'état du dossier, nous avons créé un champ *MailGestionnaire* dans la classe Dossier.

#### 4. Base de données

Pour réaliser cette application, nous avons utilisé une base de données.

Pour l'expliquer au mieux, nous allons l'illustrer par un Modèle Conceptuel de Données (MCD) :



Lorsqu'on crée une application, Web2py génère automatiquement des tables standards comme :

- **AUTH\_USER** : Contient les données relatives aux utilisateurs comme nom, prénom, mot de passe...
- **AUTH\_GROUP** : Définit les profils des utilisateurs comme Président, Représentant ou utilisateur.

Dans cette base de données, nous avons créé les tables dans lesquels nous aurons des informations à stocker comme :

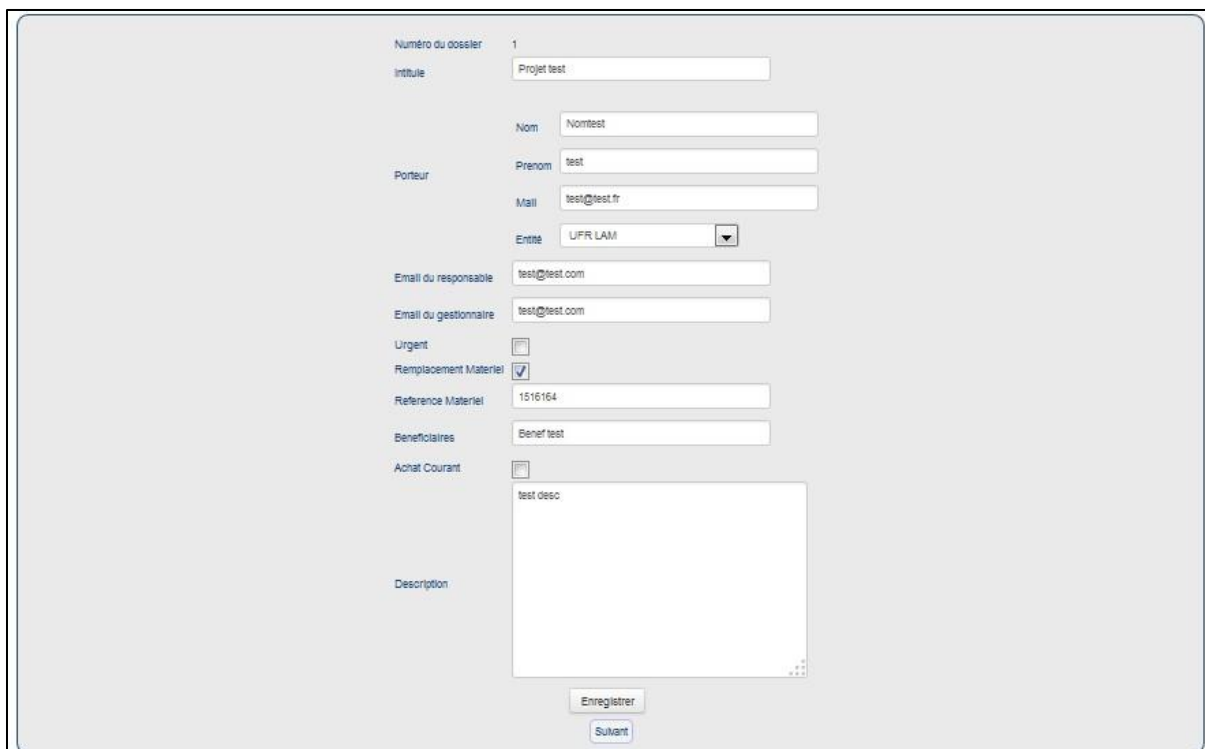
- **Dossier** : Stocke toutes les demandes des utilisateurs. Ces dossiers peuvent contenir des pièces jointes. Chaque dossier possède un état donné. Dans les relations sur le MCD, nous voyons qu'un dossier :
  - Contient un et un seul état (brouillon, soumis, accepté ou refusé).
  - Concerne un et un seul porteur de projet.
  - Appartient à une seule et une seule entité.
  - Est créé par un et un seul demandeur.

- **Achat** : qui va contenir tous les produits de chaque demande. Sur le MCD, nous observons qu'un achat est fait pour un et un seul dossier (cardinalité 1,1), tandis qu'un dossier contient un ou plusieurs achats (produits).
- **Entité** : Il s'agit de référencer toutes les demandes, les utilisateurs et les budgets par entité. Sur le MCD, nous pouvons distinguer qu'une entité:
  - Contient 0 ou plusieurs dossiers.
  - Comprend 0 ou plusieurs budgets.
  - Possède un ou plusieurs utilisateurs.
- **Budget** : Chaque entité possède son propre budget, qu'elle pourra utiliser pour faire des achats.

## VII. Notice d'utilisation

### 1. Scénarios fonctionnels

Sur la capture ci dessous, l'utilisateur débute le processus de demande d'un projet. Le profil utilisé pour ce scénario est le président. Celui-ci saisit les informations concernant le projet et valide afin de passer à l'étape d'ajout de produit au projet.



Numéro du dossier 1

Intitule Projet test

Nom Nomtest

Porteur

Prenom test

Mail test@test.fr

Entité UFR LAM

Email du responsable test@test.com

Email du gestionnaire test@test.com

Urgent ☐

Remplacement Matériel ☒

Reference Matériel 1516164

Bénéficiaires Benef test

Achat Courant ☐

Description test desc

Enregistrer

Suivant

## MAZELIN-PERNES-VINCENT

Un ou plusieurs produits vont pouvoir être ajoutés au projet. Un budget doit être sélectionné.

### Ajout d'un produit

Libelle:

Prix Unitaire:

Date d'achat:

Quantité:

Budget:  ▼

Les produits peuvent ensuite être visualisés au sein d'un tableau. Le total de la demande est affiché à l'utilisateur.

### Etat du dossier

Brouillon
Soumis
En attente
Refusé
Accepté

### Gestion des produits

Saisie de la demande
Ajout de produits
Ajout de documents
Finalisation de la demande

Total de la demande : 15.0 €

Libelle	Prix Unitaire	Date d'achat	Quantité	Budget	Total	
test produit	15.0 €	24/06/2014	1	UFR LAM	15.0 €	Modifier Supprimer

Contact Université d'Evry

L'utilisateur a ensuite la possibilité d'ajouter des pièces jointes.

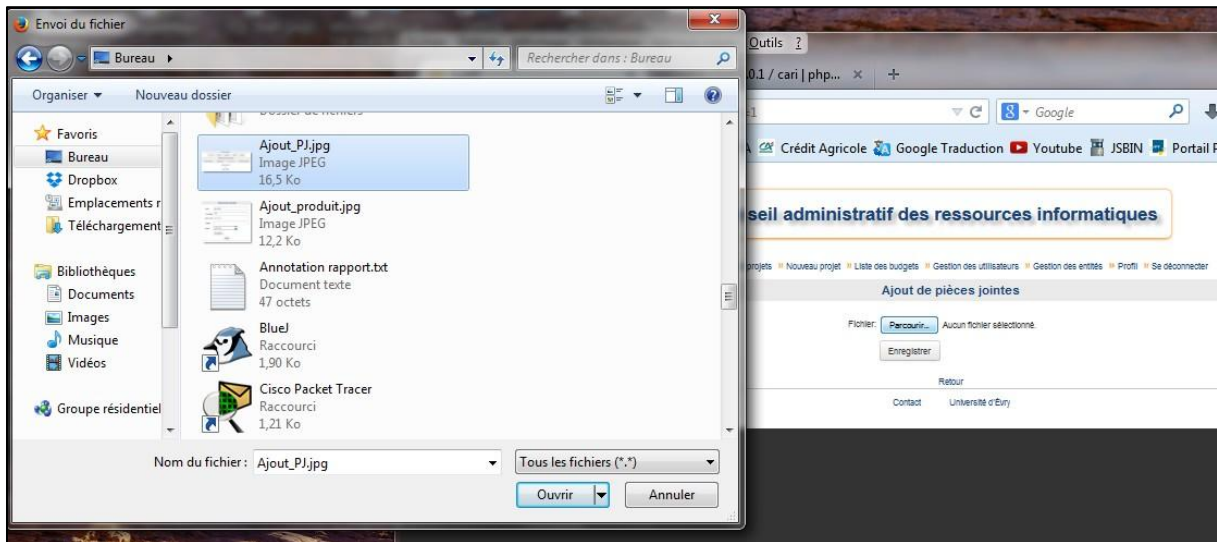
### Etat du dossier

Brouillon
Soumis
En attente
Refusé
Accepté

### Gestion des pièces jointes

Saisie de la demande
Ajout de produits
Ajout de documents
Finalisation de la demande

Vide



La pièce jointe va être stockée sur le serveur. Elle sera visible au sein d'un tableau comme le montre l'impression d'écran ci-dessous.




La demande peut enfin être soumise. A noter que celle-ci est enregistrée en tant que brouillon en attendant sa soumission.





Une fois la demande soumise, un mail sera envoyé au demandeur, responsable, gestionnaire, porteur et aux présidents.

La liste des projets peut ensuite être visible. Notre projet créé précédemment apparaît.

 **Conseil administratif des ressources informatiques**

» [Liste des projets](#) » Nouveau projet » Liste des budgets » Gestion des utilisateurs » Gestion des entités » Profil » Se déconnecter

**Gestion des projets**

Ajouter un projet

Numéro de dossier :

Rechercher

Id	Intitule	Etat Dossier Id	Entité	Date Dossier	
1	Projet test	Soumis	UFR LAM	24/06/2014 à 10:51:35	Détail

Contact Université d'Evry

Le détail de chaque projet est accessible dans le tableau et se présente tel qu'il est montré ci-dessus.

# MAZELIN-PERNES-VINCENT

Brouillon
Soumis
En attente
Refusé
Accepté

Commentaire :

Changer l'état à :

Enregistrer

Détails du dossier

Informations de la demande
Produits
Documents

Informations du demandeur

Nom Administrator  
Prénom System  
Email null@null.com  
Rôle Président  
Entité

Passer en mode édition

Numéro du dossier 1  
Intitulé Projet test

Porteur

Nom Nomtest  
Prénom test  
Mail test@test.fr  
Entité UFR LAM

Email du responsable test@test.com  
Email du gestionnaire test@test.com

Urgent ☐  
Remplacement Matériel ☒  
Référence Matériel 1516164  
Bénéficiaires Benef test  
Achat Courant ☐  
Description test desc

Suivant

La liste des budgets peut être consultée. Le budget en cours est représenté en premier suivi de l'historique de tous les budgets.

université  
evry  
Val d'Essonne  
CARI

Conseil administratif des ressources informatiques

Liste des projets
Nouveau projet
Liste des budgets
Gestion des utilisateurs
Gestion des entités
Profil
Se déconnecter

Gestion des budgets

Ajouter un budget  
Export CSV

Budgets en cours

Entité	Date	Budget Initial	
UFR LAM	25/06/2014	500.0 €	Détail

Historique des budgets

Entité	Date	Budget Initial	
UFR LAM	24/06/2014	1000.0 €	Détail
UFR LAM	25/06/2014	500.0 €	Détail

Contact Université d'Evry

## 2. Administration de compte et de l'application

L'utilisateur ayant le profil président va pouvoir procéder à la gestion des utilisateurs (Modification, Suppression).



**Conseil administratif des ressources informatiques**

» Liste des projets » Nouveau projet » Liste des budgets » Gestion des utilisateurs » Gestion des entités » Profil » Se déconnecter

**Gestion des utilisateurs**

Ajouter un utilisateur

Prénom	Nom	E-mail	Entité	Rôle	
System	Administrator	null@null.com	None	Président	Modifier Supprimer

Contact Université d'Évry

Celui-ci aura également le droit de gérer les entités (Modification, Suppression).



**Conseil administratif des ressources informatiques**

» Liste des projets » Nouveau projet » Liste des budgets » Gestion des utilisateurs » Gestion des entités » Profil » Se déconnecter

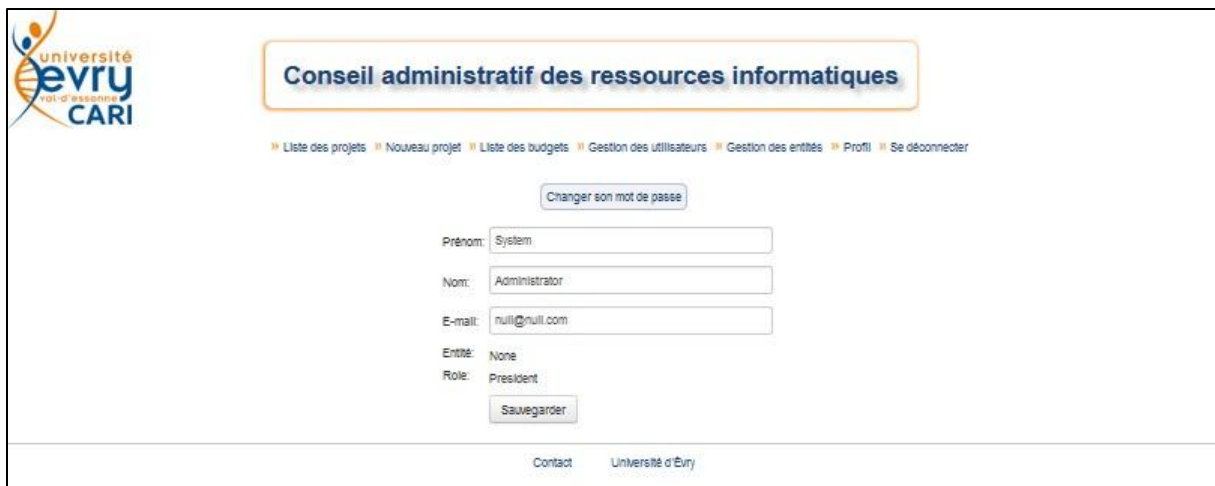
**Gestion des entités**

Ajouter une entité

Name	
UFR LAM	Modifier Supprimer
services communs	Modifier Supprimer

Contact Université d'Évry

Pour finir, un utilisateur, tout profil confondu, aura la possibilité de modifier ses informations. De plus, il pourra modifier son mot de passe.



**Conseil administratif des ressources informatiques**

» Liste des projets » Nouveau projet » Liste des budgets » Gestion des utilisateurs » Gestion des entités » Profil » Se déconnecter

Changer son mot de passe

Prénom: System

Nom: Administrator

E-mail: null@null.com

Entité: None

Rôle: Président

Sauvegarder

Contact Université d'Évry

## VIII. Bilan du projet

### 1. Fonctionnalités non développées :

Par manque de temps, les fonctionnalités suivantes n'ont pas pu être développées :

- Nous avons réalisé l'exportation des données sous format CSV. Le tableau obtenu est la liste de tous les budgets en cours (les derniers en date). Or, ce qui n'a pas été réalisé est l'exportation du détail de chaque budget. Il aurait fallu également exporter tous les produits qui concernent chaque budget.
- La deuxième fonctionnalité non développée est l'auto complétion des adresses mail. En effet, il nous a été demandé de pouvoir choisir l'adresse mail des porteurs, gestionnaires et des responsables à l'aide d'une auto complétion. Avec le Framework *Web2py*, il est possible de spécifier des champs contenus dans des formulaires afin qu'ils soient auto complétés par des valeurs qui sont en base de données.

### 2. Difficultés rencontrées

L'une des principales difficultés rencontrées est l'utilisation du Framework *Web2py* pour la réalisation de certaines fonctionnalités :

- L'exploitation des formulaires avec *Web2py* assez limité en cas de formulaire complexe
- Le requêtage simplifié de *Web2py* ne nous a pas aidés pour la création de certaines requêtes. De ce fait, nous avons dû les réaliser en SQL.
- Des composants pas forcément adaptés : par exemple, la fonctionnalité de recherche de données de *Web2py* est complexe pour un utilisateur, nous avons dû faire autrement.

## Conclusion

Ce projet nous a permis de mettre en pratique nos connaissances théoriques sur la partie modélisation comme les diagrammes UML (Diagramme de classe, diagramme de cas d'utilisation). Nous avons également compris l'importance de la phase de conception dans l'élaboration d'une nouvelle application. Cela nous permet de définir le périmètre du projet, les contraintes à respecter ainsi que les fonctionnalités à développer.

Nous avons découvert un nouveau langage de programmation, le Python. De plus, nous avons appris un Framework écrit en Python permettant de réaliser des applications Web, appelé Web2py. Ce Framework nous a facilité la tâche dans l'implémentation de certaines fonctions comme la gestion des comptes, la liste des projets...

Afin de nous organiser dans le travail, nous nous sommes :

- réparti les modules à développer.
- conçu un espace de mutualisation du projet.
- Entraidé lorsque des difficultés ont été rencontrées.

Ce projet est une source de motivation pour nous car Franck POMMEREAU a l'intention de le mettre en production afin de gérer les demandes de projets.