UNIVERSITE D'ANTANANARIVO

ECOLE SUPERIEURE POLYTECHNIQUE

****

MENTION ELECTRONIQUE

****

MEMOIRE EN VUE DE L'OBTENTION DU DIPLOME DE MASTER

TITRE : INGENIEUR

Spécialité : Informatique

Parcours : INFORMATIQUE APPLIQUEE

Intitulé :

Création automatique d'un site web à partir d'une base de données relationnelle

Présenté par :

ANDRIANJARASOA Mandrindra Nantenaina

Soutenu le 24 Septembre 2019

N° d'ordre :

105/EN/M2/IA/2019

Année universitaire

2017-2018

UNIVERSITE D'ANTANANARIVO

ECOLE SUPERIEURE POLYTECHNIQUE

****

MENTION ELECTRONIQUE

****

MEMOIRE EN VUE DE L'OBTENTION DU DIPLOME DE MASTER

TITRE : INGENIEUR

Spécialité : Informatique

Parcours : INFORMATIQUE APPLIQUEE

Intitulé :

Création automatique d'un site web à partir d'une base de données relationnelle

Présenté par :

ANDRIANJARASOA Mandrindra Nantenaina

Soutenu le 24 Septembre 2019

Devant les membres de jury composés de

Monsieur RASTEFANO Elisée, Président

Madame RABEHERIMANANA Lyliane Irène, Examinateur

Monsieur HERINANTENAINA Edmond Fils, Examinateur

Monsieur RAKOTOARISOA Hasina Patrick, Examinateur

Monsieur RAMASOMBOHITRA Nivonjy Nomen'Ahy, Rapporteur

N° d'ordre :

105/EN/M2/IA/2019

Année universitaire

2017-2018

TENY FISAORANA

« Misaora an' Jehovah, ry fanahiko; ary izay rehatra ato anatiko, misaora ny anarany masina». Isaorana Andriamanitra noho ny fitantanany ka nahafahana nanatontosa izao fikarohana izao. Na toy inona na toy inona finiavana sy fahazotoana tamin'ny zavatra natao, raha tsy teo Andriamanitra nanolotsaina ary koa nanome fahasalamana dia zava-poana avokoa izany. Ho Azy irery anie ny dera sy ny laza ary ny voninahitra! Amena.

Manaraka izany, isaorana eram-po eran-tsaina koa ianareo mpampianatra rehetra ato amin'ny lalam-piofanana « Electronique », noho ny fanabeazana nataonareo nandritra izay dimy taona izay. Isaorana manokana amin' izany ny mpitarika izao fikarohana izao noho ireo toro-hevitra maro be natolony, misaotra amin' ny famakafakana sy ny fandinihina lalina nataonao ka nahafahana nanatsara izao asa izao. Izao fanehoana voka-pikarohana izao no tontosa, dia teo ianareo rehetra mpitsara, koa isaorana ianareo nahafoy fotoana sy hery amin' izao asa izao.

Tolorana fisaorana feno ihany koa ianareo fianakaviana indrindra fa ianareo Ray aman-dReny nitaiza sy nanolokolo ary koa nanohana ka nahatonga ahy amin' izao tanjona izao. Tsy hadino koa ny misaotra anareo tapaka sy namana nanampy sy nanoro hevitra.

Misaotra indrindra tompoko! Andriamanintra manankarem-pahasoavana anie hamaly avo zato heny ny soa nomenareo.

Mandrindra

REMERCIEMENTS

A Dieu tout-puissant nous rendons grâce et témoignage pour sa gratitude, sa bénédiction de nous avoir donné la force, la santé. Sans Lui, nous n'aurions pas pu bien faire la réalisation de ce travail.

Nos sincères reconnaissances et gratitudes vont également aux personnes suivantes :

A Monsieur ANDRIANAHARISON Yvon, ancien directeur de l'Ecole Supérieure Polytechnique d'Antananarivo.

A Monsieur RAKOTOSAONA Rijalalaina, directeur de l'Ecole Supérieure Polytechnique d'Antananarivo.

A Monsieur ANDRIAMANANTSOA Guy Danielson, ancien chef de la mention Electronique.

A Madame RAMANANTSIHOARANA Harisoa Nathalie, chef de la mention Electronique.

A Monsieur RASTEFANO Elisée qui nous a fait le grand honneur de présider le jury de cette soutenance de mémoire.

A Madame RABEHERIMANANA Lyliane Irène,

A Monsieur HERINANTENAINA Edmond Fils,

A Monsieur RAKOTOARISOA Hasina Patrick qui ont accepté de juger cette soutenance de mémoire.

A Monsieur RAMASOMBOHITRA Nivonjy Nomen'Ahy qui nous a dirigé pendant toutes les réalisations de ce travail.

A tous les Enseignants de l'Ecole Supérieure Polytechnique d'Antananarivo au sein de la Mention Electronique pour leurs enseignements et leurs partages durant ces cinq années.

A tous les membres de nos familles pour leurs soutiens moraux et financiers. Et à nos amis et à tous ceux qui ont contribué de loin ou de près à l'élaboration de ce manuscrit.

Mandrindra

RESUME

Une base de données permet de stocker et de retrouver l'intégralité de données brutes ou d'informations en rapport avec un thème ou une activité ; celles-ci peuvent être de natures différentes et plus ou moins reliées entre elles. Ce travail consiste à créer un site web qui permet de manipuler une base de données relationnelle. Ce site est créé automatiquement à l'aide d'un outil qui collecte les informations sur la configuration de la base de données utilisée. Le site web obtenu possède plusieurs interfaces pour consulter et modifier les informations stockées dans la base de données. Il est protégé et nécessite une authentification pour y accéder. Deux types d'utilisateur existent, un administrateur et un simple utilisateur. L'administrateur a le droit de manipuler toutes les données venant de la base et de faire les paramétrages nécessaires. Un simple utilisateur peut consulter les données qu'on lui a octroyées. Le langage PHP avec l'architecture MVC et le serveur de base de données MySQL ont été utilisés pour réaliser le projet.

SOMMAIRE

LISTE DES ABREVIATIONS

ANSI : American National Standard Institute

API : Application Programming Interface

BD : Base de Données

CSS : Cascading Style Sheet

DBA : Data Base Access

E/A : Entité/Association

HTML : Hypertexte Markup Langage

IBM : International Business Machines Corporation

IDS : Integrated Data Storage

IHM : Interface Homme Machine

IMS : Information Management System

LCD : Langage de Contrôle de Données

LCT : Langage de Contrôle de Transaction

LDD : Langage de Définition de Données

LMD : Langage de Manipulation de Données

MVC : Modèle Vue Contrôleur

PHP : Personal Home Page Hypertext Preprocessor

PDO : PHP Data Object

POO : Programmation Orienté Objet

QBE : Query By Example

QUEL : Query English Language

SEQUEL : Structured English Query Language

SGBD : Système de Gestion de Base de Données

SGBDR : Système de Gestion de Base de Données Relationnelle

SGBDO : Système de Gestion de Base de Données orientée Objet

SQL : Structured Query Language

WAMP : Windows Apache MySQL PHP

LISTE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION

La gestion des informations est primordiale dans plusieurs domaines. Cette gestion s'est évoluée de jours en jours et qui passe d'une gestion manuelle à la gestion informatisée. Actuellement, les informations sont parfois stockées dans une base de données. Cette dernière est au centre des dispositifs informatiques de collecte, mise en forme, stockage et utilisation d'informations. La création d'une interface est utile pour partager ces informations aux personnes qui veulent les utiliser. Le dispositif comporte un système de gestion de base de données un logiciel moteur qui manipule la base de données et dirige l'accès à son contenu. De tels dispositifs comportent également des logiciels applicatifs, et un ensemble de règles relatives à l'accès et l'utilisation des informations.

Ce travail consiste à créer automatiquement le code source d'un site web qui permet de manipuler les informations stockées dans une base de données. Un site web, ou simplement site, est un ensemble de pages web et de ressources reliées par des hyperliens, défini et accessible par une adresse web. Un site est développé à l'aide de langages de programmation web, puis hébergé sur un serveur web accessible via le réseau mondial internet, un intranet local, ou n'importe quel autre réseau. Cet ouvrage s'intitule *« Création automatique d'un site web à partir d'une base de données relationnelle »*.

Le manuscrit est organisé de façon qui suit. Le premier chapitre décrive la généralité sur la base de données relationnelle. Le second chapitre est consacré sur le langage de programmation orientée objet, en particulier le langage PHP. Le dernier chapitre est focalisé sur la conception et la réalisation du projet.

CHAPITRE 1 : GENERALITE SUR LA BASE DE DONNEES RELATIONNELLE

1.1. Introduction

Avant l'invention de la base de données, toutes les informations sont stockées dans des fichiers manipulés par les logiciels applicatifs. En raison du programme Apollo du 1960, un dispositif informatique était créé pour contenir les enregistrements des nombreuses informations spatiaux. Dès cette année IBM (International Business Machines Corporation) a lancé un logiciel informatique « Information Management System » (IMS) sur le marché, et les différentes grandes entreprises qui se travaillent dans le secteur informatique se concurrencent à cette invention. En 1970, les théories de la base de données relationnelle sont apparues après l'œuvre d'Edgar Frank Codd qui est un employé de l'IBM. Ce système de gestion de base de données relationnelle est équipé par un langage spécifique qui est le « Structured Query Langage » (SQL). A nos jours ce langage est l'un de langage le plus célèbre à la manipulation des données relationnelles.

1.2. Définitions

1.2.1. Qu'est-ce qu'une donnée ?

En informatique, une donnée est la représentation d'une information dans un programme : soit dans le texte du programme (code source), soit en mémoire durant l'exécution. Les données, souvent codées, décrivent les éléments du logiciel tels qu'une entité (chose), une interaction, un évènement, un sous-système. Les données peuvent être conservées et classées sous différentes formes : textuelles, numériques, images, sons, etc.

1.2.2. Définition d'une base de données

Une base de données (notée BD) est une collection de données reliées et stockées ensemble avec aussi peu de redondance que possible pour servir une ou plusieurs applications de façon optimale. Elle est accessible à la demande pour plusieurs utilisateurs et des besoins divers.

L'utilisateur dispose de moyens très élaborés pour effectuer tant d'opérations : créer de nouveaux fichiers, consulter, ajouter, modifier et même supprimer de données, calculer et éditer des résultats.

Le schéma de principe d'une BD est représenté par la fig.1.1 qui suit.

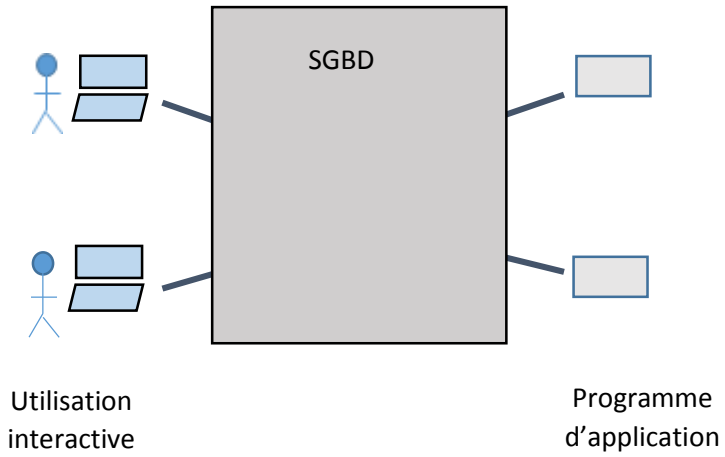

Figure 1.1 : Schéma de principe d'une BD

1.2.3. Système de gestion de base de données

Un Système de Gestion de Base de Données (noté par SGBD) est un logiciel responsable pour la gestion des données. C'est un ensemble de programmes assurant la structuration, le stockage, la mise à jour et la recherche des données. Un SGBD comprend les interfaces nécessaires aux différentes formes d'utilisation des données. Il permet de gérer toutes les informations stockées (description, consultation, adjonction, modification, suppression, autorisation) en toute sécurité dans un contexte multiutilisateurs. Il se présente sous la forme d'un ensemble de modules complémentaires installés sur le système d'exploitation.

La première génération de SGBD est apparue à la fin des années 60. Elle est marquée par la séparation de la description des données et de la manipulation par les programmes d'application. Ce premier SGBD est basé sur les modèles réseau ou hiérarchique, c'est-à-dire des modèles de données organisés autour de types d'articles constituant les nœuds d'un graphe, reliés par des types de pointeurs composant les arcs du graphe. Cette génération a été dominée par l'IDS (Integrated Data Storage) et IMS.

La deuxième génération a grandi dans les laboratoires depuis 1970, à partir du modèle relationnel. En effet, les données sont présentées aux utilisateurs sous forme de relation entre domaine de valeur, simplement représentées par des tables.

La troisième génération a été développée depuis le début des années 80 en supportant les modèles de données extensibles intégrant le relationnel et l'objet**.** Elle conserve les acquis du relationnel en permettant une vision tabulaire des objets et une interrogation vient du langage SQL étendu aux objets.

L'utilisation d'un SGBD est inséparable au grand projet informatique et voici quelques SGBD les plus utilisés [01] :

ACCESS : plate-forme Windows, monoposte, licence commerciale.

SQL SERVER : plateforme Windows, mode client/serveur, licence commerciale.

ORACLE : plateforme Windows et Linux, mode client/serveur, licence commerciale.

SYBASE : plateforme Windows et Linux, mode client/serveur, licence commerciale.

POSTGRESQL : plateforme Windows et Linux, mode client/serveur, licence libre.

MYSQL : plateforme Windows et Linux, mode client/serveur, licence libre.

1.3. Rôles d'un système de gestion de base de données

1.3.1. Décrire les données

Tout SGBD propose un langage de description des données (LDD). Ce langage permet de décrire : les données (type, longueur, nature), les relations entre les données, les règles de gestion, les domaines de valeur. Ces actions sont indépendamment des applications de manière intrinsèque.

1.3.2. Manipuler les données

Le langage de manipulation des données (LMD) dans un SGBD sert à exécuter les opérations d'ajout, de suppression, de modification des données. Il permet également l'interrogation des données. Ce langage peut être interactif en quelque SGBD.

1.3.3. Contrôler les données

Les SGBD assurent la confidentialité et l'intégrité des données. C'est-à-dire qu'ils permettent de définir les droits précis des utilisateurs pour les accès aux données. La plupart des SGBD offrent aussi la possibilité de décrire les règles de gestion du système d'information de l'entreprise. Cette action est effectuée par le langage de contrôle de données (LCD). Cela indique que le moteur vérifie qu'aucun utilisateur n'accède à des informations non autorisées, et qu'aucun utilisateur n'effectue des modifications qui seraient contraire aux règles de cohérence. Cette capacité diminue le risque de piratage des données.

1.3.4. Contrôler les transactions

Chaque SGBD possède son propre moyen d'y entrée pour faire un changement au niveau de données et de sortir l'action qui se déroule au moment où il répond aux requêtes de l'utilisateur. Le langage de contrôle de transaction (LCT) assure la gestion des transactions qui est la fonction très utile pendant l'insertion et la mise à jour des données. Il est la porte de validation ou d'annulation de modification de données dans la BD.

1.3.5. Gestion du stockage

La façon dont le SGBD organise les données en fichier est transparente pour l'utilisateur. Il offre une vue logique des données. Dans un SGBD les données sont indépendantes de programme qui l'utilise.

1.3.6. Sécuriser les données

Le SGBD assure la sécurité des données, à savoir qu'il doit veiller à ce que les données restent cohérentes. Par exemple, il doit faire en sorte qu'une requête ne puisse pas être créée tant que toutes les lignes de requête ne sont pas valides, ou qu'un utilisateur ne soit pas supprimé sans avoir supprimé tous les droits attachés à cet utilisateur. Dans les deux cas (création d'une requête ou suppression d'un utilisateur), il y a plusieurs actions consécutives à exécuter dans la base de données. Pour que la base de données reste cohérente, il faut que ces actions soient toutes exécutées ou non. L'ensemble de ces actions est appelé transaction**.** Le SGBD doit rendre les transactions in-interruptible. Si une interruption intervient au cours d'une transaction, le système remet les données dans l'état où elles étaient avant le début de la transaction.

1.3.7. Indépendance physique et logique

Pour l'indépendance physique, il permette à la modification des structures de stockage sans que cela ait de répercussion au niveau des applications : les disques, les méthodes d'accès, les modes de placement. Alors les codages des données ne sont pas apparentés.
Pour l'indépendance logique, il permette aux différentes applications d'avoir des vues différentes des mêmes données, permette au DBA (Data Base Access) de modifier le schéma logique sans que cela ait de répercussion au niveau des applications.

1.4. Niveau et modèle de description d'un SGBD

1.4.1. Niveau de description d'un SGBD

La plupart des SGBD suivent l'architecture standard ANSI/SPARC qui permet d'isoler trois niveaux de description (ou d'abstraction) nécessaire pour un SGBD (Fig.1.2). Chaque niveau correspond un schéma de représentation :

- Niveau interne (ou physique) : concerne le stockage des données au niveau des unités de stockage des fichiers et les fonctions d'accès. On l'appelle le schéma interne.
- Niveau conceptuel (ou logique) : décrit la structure des données dans la base, leur propriété et leur relation, indépendamment de toute préoccupation technologique d'implémentation ou d'accès par les utilisateurs ; c'est ce que l'on appelle le schéma conceptuel. Ce schéma décrit la structure de la base indépendamment de son implantation.
- Niveau externe : décrit comment chaque utilisateur perçoit les données ; c'est ce que l'on appelle le schéma externe ou vue. Une vue est une représentation abstraite d'une partie de la base de données conceptuelle (ou un sous schéma du schéma conceptuel).

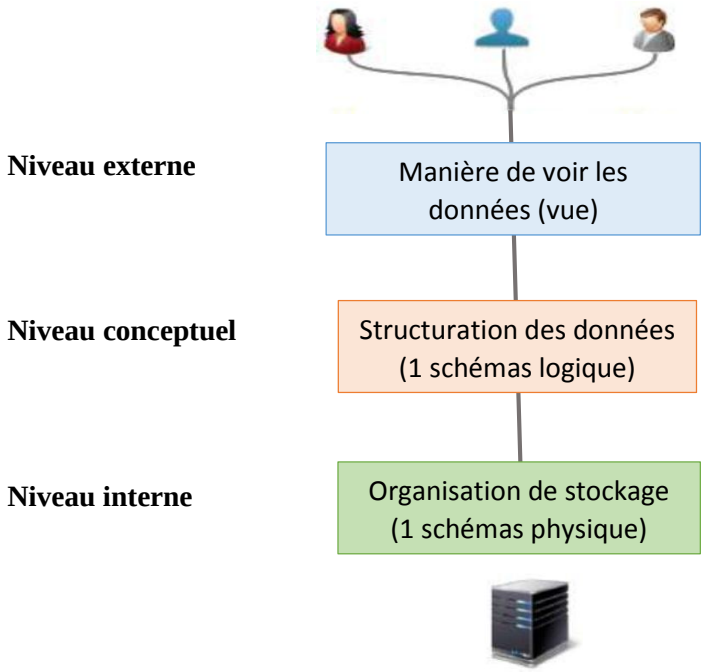

Figure 1.2 : Niveau de représentation de données [02]

Remarque **:**

Le SGBD doit être capable de faire des transformations entre chaque niveau, de manière à transformer une requête exprimée en termes de niveau externe en requête du niveau conceptuel puis du niveau physique.

1.4.2. Modèle de description d'un SGBD

Le résultat de la conception d'une base de données est une description des données (appelée schéma) en termes de propriété d'ensemble d'objets et d'organisations logiques des données. Pour obtenir une telle représentation à partir d'un problème réel, on utilise un outil appelé modèle de description basé sur un ensemble de concepts et de règles formant le langage de description. Un SGBD peut être caractérisé par le modèle de description qu'il supporte. Une fois la base de données ainsi spécifiée, il est possible de manipuler les données en réalisant des opérations de sélection, d'insertion, de modification et de suppression à partir d'un langage spécifique de manipulation de données ou par un langage de programmation classique. Plusieurs types de modèles sont utilisés : le modèle hiérarchique, le modèle réseau, le modèle relationnel, le modèle objet.

a. Modèle hiérarchique

Le modèle hiérarchique présente les données et leur liaison de type 1 à n. Une liaison de type n à n devra être décomposée en liaison 1 à n. L'utilisation de ce type de base de données requiert une parfaite connaissance des hiérarchies et des pointeurs. Les utilisateurs ne peuvent accéder aux données que par l'utilisation de programme écrit spécifiquement.
Son schéma logique est représenté par un arbre. La figure 1.3 représente un exemple de schéma hiérarchique.

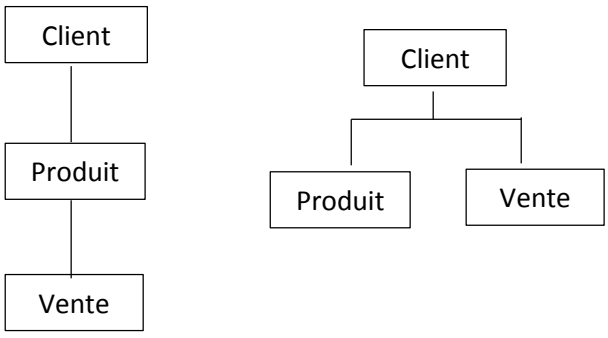

Figure 1.3 : Exemple de schéma hiérarchique

L'accès aux données commence par la racine et descende l'arborescence jusqu'au détail recherché.

b. Modèle réseau

Le modèle réseau est l'un du modèle le plus ancien. Il fonctionne sur le même mode navigationnel que le hiérarchique. C'est-à-dire par pointeur. Le modèle réseau permet de représenter les liaisons n à n. Des nombreux liens existent entre les différents éléments de données. On se promène à l'intérieur du réseau en partant d'un point de départ et en suivant les liens. Et les liens sont possibles dans tous les sens.

Un même type d'enregistrement peut être lié à deux pères. Il n'y a pas de contrainte pour accéder aux fichiers, c'est-à-dire l'accès aux données est réalisé par divers cheminements. Son schéma logique est représenté par un graphe (Fig.1.4).

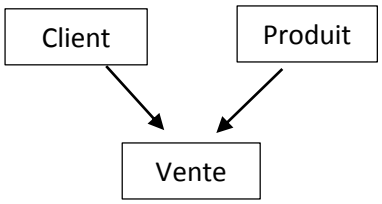

Figure 1.4 : Exemple de schéma réseau

c. Modèle relationnel

C'est le modèle le plus répandu aujourd'hui. Il lève toutes les contraintes du hiérarchique et du réseau. Il permet de ranger les données en vrac. Il utilise un système de clé pour faire une recherche et de voir les relations existantes. Son schéma logique est donc représenté par des relations.

Il existe aussi des correspondances de vocabulaire utilisées dans ce modèle. Les tables se disent par des entités, les colonnes sont des attributs et les lignes sont des n-uplets (ou tuples). Pour illustrer l'utilisation de ces termes, la Fig.1.5 montre un exemple de relation.

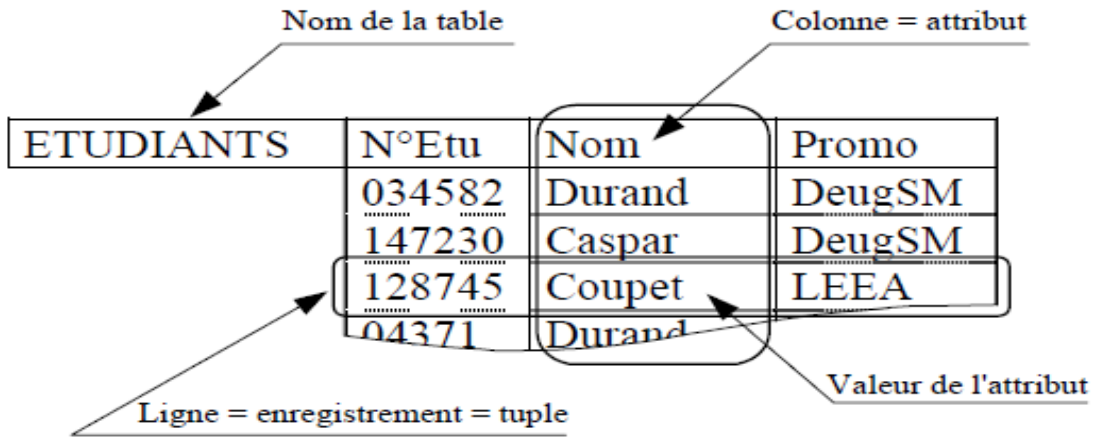

Figure 1.5 : Exemple d'une relation

L'entité étudiante comporte trois attributs qui sont le numéro d'étudiant (N° Etu), le Nom et le Prénom. Chaque ligne des données dans cette entité représente des enregistrements.
Une base de données relationnelle doit pour être performante respecter toutes les règles de la troisième forme normale. Ces règles sont les suivantes : la valeur de la clé identifie un tuple et tous ses attributs. L'existence de la relation implique une dépendance fonctionnelle entre chaque clé et ses attributs. La relation est représentée comme un tableau, où chaque tuple est porté par une ligne, et chaque type d'attribut occupe une colonne. Il n'existe qu'un seul élément par boite (ligne, colonne). Tous les tuples ont le même contenu c'est-à-dire mêmes attributs dans le même ordre. Une entité peut supporter plusieurs types de clés. La clé primaire pour l'identification de chaque tuple et les autres clés pour les relations avec les autres entités.

Remarque :

Les systèmes de gestions de bases de données relationnelles (SGBDR) sont les plus courants des SGBD, car la plupart des bases de données étaient relationnelle. Mais à cause des énormes quantités de données utilisées dans le monde du web d'aujourd'hui, les entreprises comme Google, Amazon ou Facebook développent un mouvement important de développement de base de données non relationnelle.

d. Le modèle objet

Les systèmes de gestions de bases de données orientées Objets (SGBDO) étaient à l'origine évoqué comme le successeur des SGBDR. Il repose sur la théorie des objets. Dans cette théorie, le système d'information peut être représenté comme un ensemble d'objets possédant des propriétés et des méthodes et communicant entre eux par échange de message.

1.5. Base de données relationnelle

1.5.1. Modèle entité-association

L'informatisation dénombre d'activités nécessite leur modélisation, c'est-à-dire leur expression sous une forme symbolique (le plus souvent mathématique) susceptible d'être représentée en machine.
Le modèle Entité - Association (EA) ou Entité- (EA) ou Entité - Relation fournit un outil pour analyser les situations du monde réel (entreprises, institutions, etc.).
Les définitions importantes concernant le modèle entité- association (E/A) sont les suivantes.

- Une entité est un objet spécifique, concret ou abstrait, de la réalité perçue. Ça peut être une personne, un véhicule, un concept abstrait, un événement, etc. Les entités de même nature sont regroupées dans un ensemble d'entité. Par exemple toutes les personnes, tous les véhicules, etc. Une classe d'entité représente de manière abstraite un ensemble d'entité.
- Un attribut est une propriété caractéristique des entités d'une même classe. Un attribut associe à chaque entité une valeur appartenant à un domaine.
- Un domaine est un ensemble de valeurs acceptables pour l'attribut considéré.
- Une clé primaire est un attribut choisi dans la description de la table. Sa valeur identifie de manière unique chaque tuple de la relation. Elle est obligatoire pour toutes les relations et elle ne peut pas prendre une valeur indéfinie (NULL).
- Une clé étrangère s'agit d'un attribut (ou groupe d'attribut) dont la valeur est la clé primaire d'une autre relation.
- Une association est une représentation d'un lien entre deux entités ou plus. Elle peut avoir des propriétés particulières.

Le modèle entité-association fournit un outil pour analyser les situations du monde réel (entreprise, institution, etc.) et représenter par un graphe.

Un graphe entité-association (E/A) décrit la structure d'ensemble d'une BD en combinant les objets graphiques qui sont les rectangles pour représenter les entités, les losanges pour les associations et les arêtes qui lient les entités (rectangles) aux associations (losanges).

La figure 1.6 montre une illustration du modèle E/A.

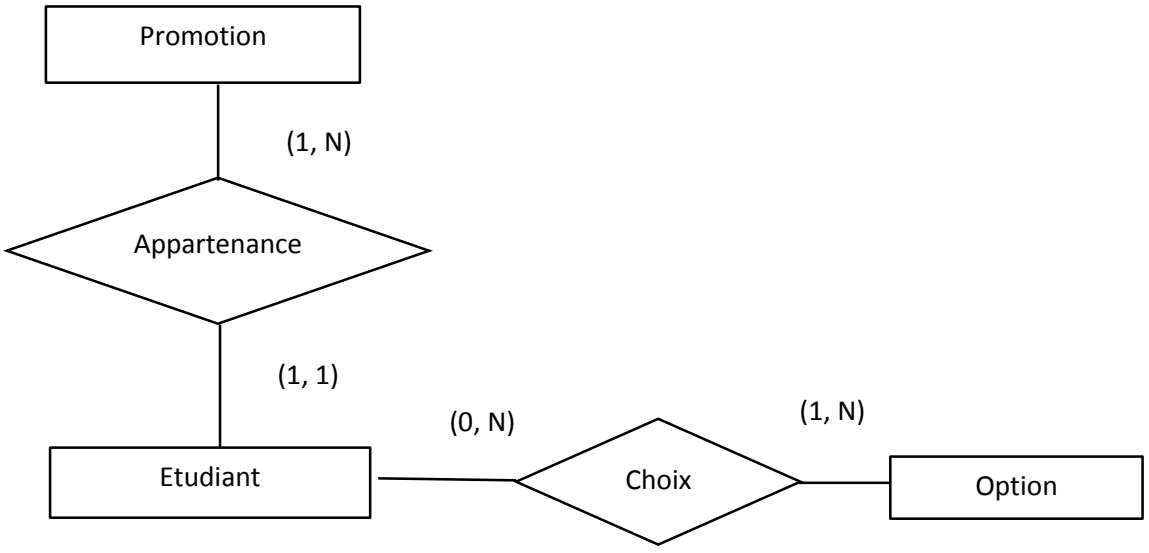

Figure 1.6 : Exemple de modèle entité-association [04]

La promotion représente une entité. Elle a de relation avec l'entité étudiant. Ce dernier se relie avec l'entité option.

1.5.2. Cardinalité

La cardinalité d'une association spécifie le nombre minimum et le nombre maximum de participation de chaque occurrence d'entité à chaque rôle de l'association. Le nombre est dit minimum si le nombre de fois qu'une occurrence de l'entité participe aux occurrences de l'association, généralement 0 ou 1. Et il est maximum si le nombre de fois qu'une occurrence de l'entité participe aux occurrences de l'association, généralement 1 ou n.
Considérons les quatre cas principaux pour la cardinalité suivis des exemples significatifs :
(1,1) (ou de type 1) : Chaque étudiant est inscrit dans une et une seule promotion.
(0,1) (ou conditionnelle) : Chaque étudiant n'est pas forcément représentant de sa promotion.
(1, N) (ou multiple) : Chaque promotion comporte plusieurs étudiants.
(0, N) (ou multiple conditionnelle) : Chaque étudiant choisit un nombre quelconque d'option.
Les deux premiers sont de type simple et les autres sont de type complexe.

1.5.3. Passage du modèle entité-association au modèle relationnel

Toutes constructions de modèle relationnel sont venues de l'existence du modèle entité-association. Alors l'obtention d'une meilleure représentation d'une base de données relationnelle requit une forte connaissance à la modélisation entité-association. L'opération consiste à représenter sous forme de table les entités et les associations. Pour cela, on applique les règles qui suivent : Chaque entité est traduite en une table distincte, dont la clé primaire peut être soit celle de l'entité, soit une autre clé candidate. Les autres attributs de l'entité sont reportés comme attribut de la nouvelle table. La conversion d'une association dépend de sa cardinalité [05].

Règle n°1 :

Une association de dimension 2 de type simple-complexe (par exemple, (1,1) -(1, N)) ne nécessite pas la création d'une nouvelle table. Elle est traduite en définissant une clé étrangère dans la table qui se situe du côté simple de l'association. Cette clé doit faire référence à la clé d'identification de la seconde table, et son nom est judicieusement choisi en conséquence.
La figure 1.7 montre un exemple de passage du modèle E/A en modèle relationnel en utilisant la règle n°1.

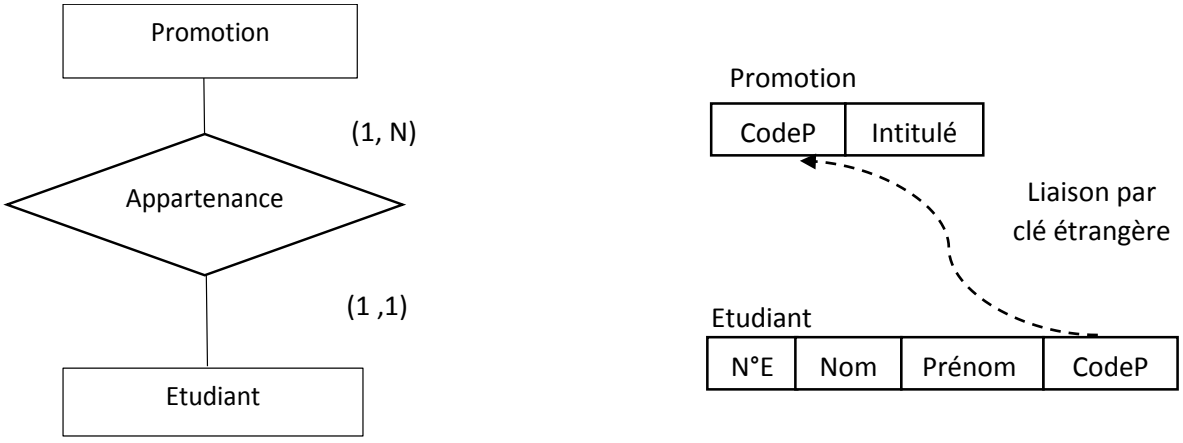

Figure 1.7 : Exemple de passage E/A en modèle relationnel en utilisant la règle n°1

Règle n°2 :

Une association de dimension 2 de type simple-simple (par exemple, (1,1) -(0,1)) se traite de la même façon, en choisissant en principe d'introduire la clé étrangère dans la table située du côté (1,1) de l'association ; voici un exemple (Fig. 1.8).

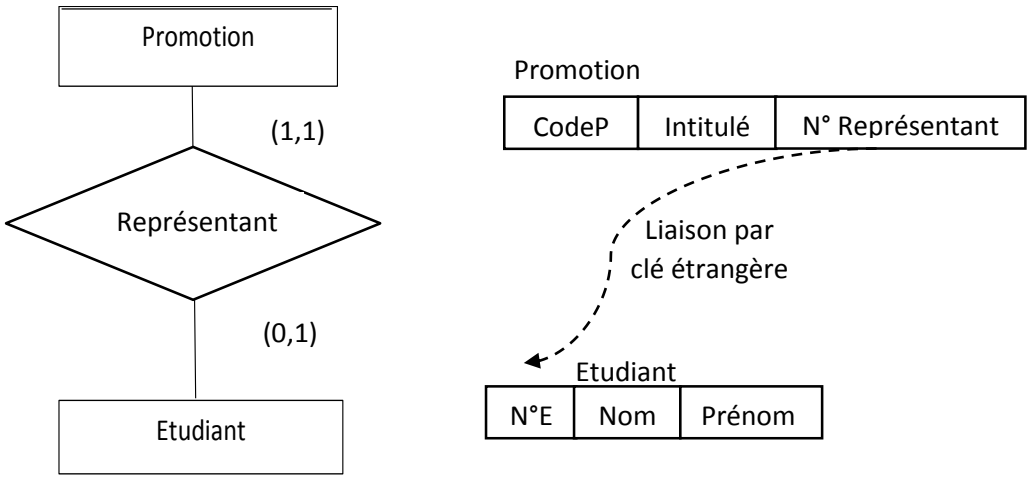

Figure 1.8 : Exemple de passage E/A en modèle relationnel en utilisant la règle n°2

Règle n°3 :

Chaque association de dimension 2 de type complexe-complexe (par exemple, (0, N) -(1, N)) est représentée par une table distincte, contenant les identifiants des deux entités associées comme clés étrangères. Ces attributs constituent souvent, à eux deux, la clé primaire de la nouvelle table. Si l'association comporte d'autres attributs, ceux-ci sont également ajoutés à la table ; prenons un exemple (Fig. 1.9).

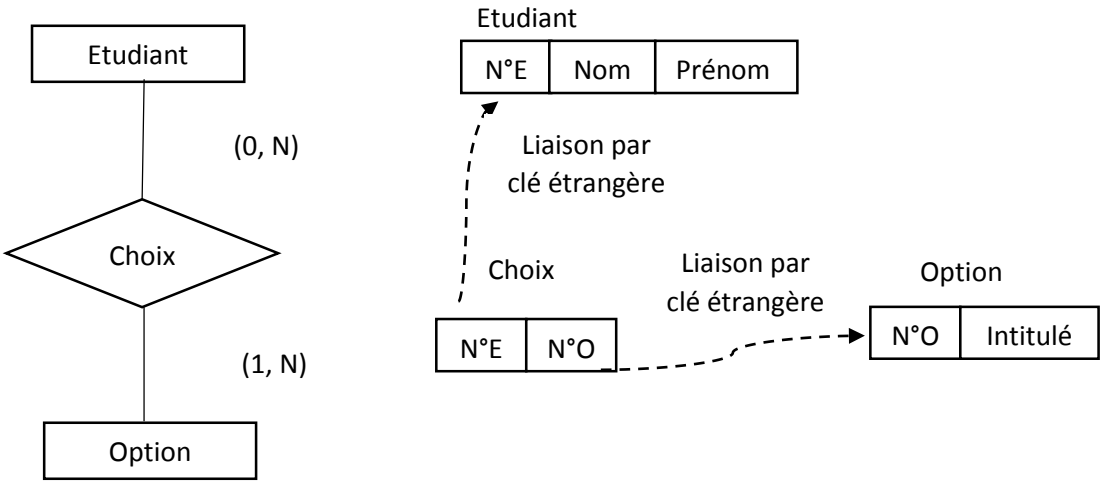

Figure 1.9 : Exemple de passage E/A en modèle relationnel en utilisant la règle n°3

Règle n°4 :

Une association de dimension supérieure à 2 se réécrit selon la règle R3 (Fig. 1.10).

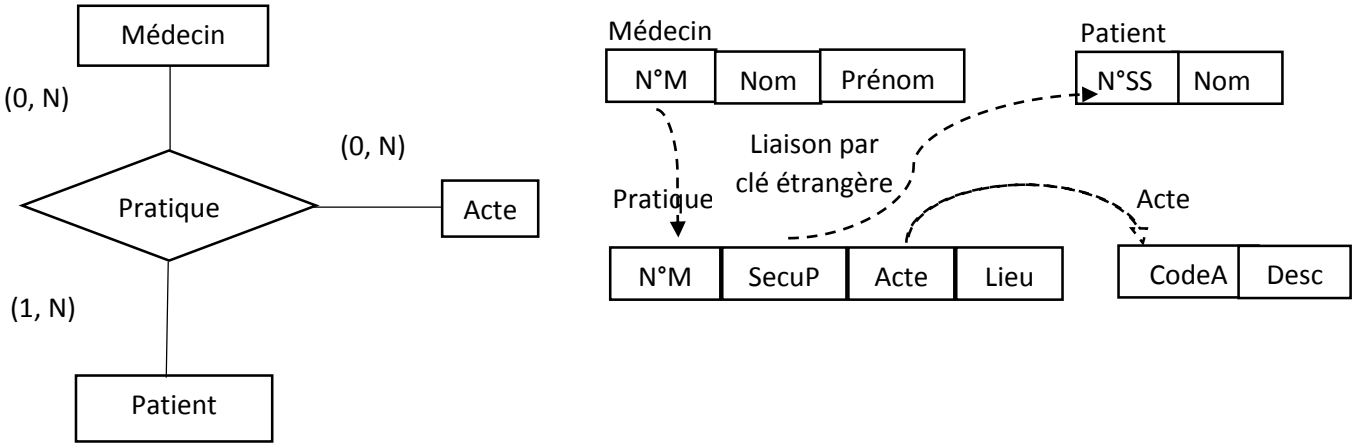

Figure 1.10 : Exemple de passage E/A en modèle relationnel en utilisant la règle n°4

1.6. L'algèbre relationnelle

Les opérateurs de base les plus utilisés dans une base de données sont : union, différence, produit cartésien, jointure.

1.6.1. Union

L'union de deux relations *r* et *s* de même schéma *R* est une relation de même schéma contenant l'ensemble des tuples appartenant à *r* ou à *s*.

Exemple : soient 2 relations P1 et P2 de schéma PROD(No,marque,couleur) ; P3 est l'union de P1 et P2 (Fig.1.11).

P1

No	marque	couleur
100	X	R
110	Y	B

P2

No	marque	couleur
100	X	R
120	Z	G
130	W	V

P3 = P1 U P2

No	marque	couleur
100	X	R
110	Y	B
120	Z	G
130	W	V

Figure 1.11 : Exemple union de deux relations

1.6.2. Différence

La différence de deux relations r et s de même R est une relation de même schéma contenant l'ensemble des tuples appartenant à r et n'appartenant pas à s.

Exemple : soient 2 relations P1 et P2 de schéma PROD(No,marque,couleur) ; P4 est la différence de P1 et P2 (Fig.1.12).

P1

No	marque	couleur
100	X	R
120	Z	G
130	W	V

P2

No	marque	couleur
100	X	R
110	Y	B

P3 = P1 - P2

No	marque	couleur
120	Z	G
130	W	V

Figure 1.12 : Exemple différence de deux relations

1.6.3. Produit cartésien

Le produit cartésien de deux relations r et *s*, de schéma quelconque R et S, est une relation ayant pour attributs la concaténation de ceux de R et de S et dont les tuples sont toutes les concaténations d'un tuple de r à un tuple de *s*.

Exemple : soient P5 de schéma PROD(No,marque,couleur) et P6 de schéma DEPOT(Nd,adresse), P7 est le produit cartésien de P5 et P6 (Fig.1.13).

P5

No	marque	couleur
100	X	R
110	Y	B

P6

Nd	adresse
5	R
7	B

P7= P5 x P6

No	marque	couleur	Nd	adresse
100	X	R	5	R
100	X	R	7	B
110	Y	B	5	R
110	Y	B	7	B

Figure 1.13 : Exemple produit cartésien de deux relations

1.6.4. Jointure

La jointure de deux relations r et s de schéma quelconque R et S, selon une qualification Q est l'ensemble des tuples du produit cartésien $R \times S$ satisfaisant Q. elle est donc programmer comme un produit suivi d'une sélection.

Exemple : soient P8 de schéma PROD(No,marque,adresseF) et P9 de schéma DEPOT(Nd,adresseD) et Q =(adresseF=adresseD) ; P10 est le jointure de P8 et P9 satisfaisant Q (Fig.1.14).

P8

No	marque	adresseF
100	X	A1
120	Y	A2
150	Z	A2

P9

Nd	adresseD
5	A3
7	A1
12	A2

P10= Q(P8,P9)

No	marque	adresseF	Nd	adresseD
100	X	A1	7	A1
100	Y	A2	12	A2
110	Z	A2	12	A2

Figure 1.14 : Exemple jointure de deux relations

1.6.5. Jointure naturelle

La jointure naturelle de deux relations r et s de schémas respectifs schémas respectifs R et S, notée r |×| s, est l'équijointure de r et s sur tous les attributs de même nom dans R et S, en éliminant les tuples en doublons.

Exemple : Soient P13 de schéma PROD(No,marque,FAB) et P14 de schéma PRIX(No,PU,FAB) (Fig.1.15).

P8

No	marque	FAB
100	X	FA1
120	Y	FA2
150	Z	FA2

P9

No	PU	FAB
100	500	FA1
120	650	FA2
150	800	FA2

P8 |×| P9

No	marque	PU	FAB
100	X	500	FA1
120	Y	650	FA2
150	Z	800	FA2

Figure 1.15 : Exemple d'une jointure naturelle de deux relations

1.7. Langage SQL

SQL signifie « Structured Query Language », c'est-à-dire « langage d'interrogation structuré ».

En fait, SQL est un langage déclaratif destiné à la manipulation de base de données au sein des SGBD et plus particulièrement au SGBDR. Il n'est donc pas proprement parlé un langage de programmation mais plutôt une interface standard pour accéder aux bases de données. Il a été conçu par IBM dans les années 70. Il s'agit d'une évolution du langage SEQUEL (Structured English Query Language), commercialisé tout d'abord par ORACLE. Il est reconnu par la grande majorité des systèmes de gestions de bases de données relationnelles. Le SQL est un langage apte à réaliser les opérations existant dans l'algèbre relationnelle comme l'opération ensembliste (union, intersection, différence) et aussi l'opération spécifique (projection, restriction, jointure, division).

Le langage SQL est à la fois :

- un langage d'interrogation de la base (ordre SELECT),
- un langage de manipulation des données
- un langage de définition des données
- un langage de contrôle de l'accès aux données.

Le tableau 1 suivant résume les commandes utilisées dans ce langage.

Tableau 1 : Commandes utilisés dans une requête SQL

Langages	Commandes	Utilisations
Langage de définition de données (LDD)	CREATE DATABASE	Créer une base de données
	USE	Utiliser la base
	CREATE TABLE	Créer une table
	CREATE VIEW	Créer une vue
	ALTER TABLE	Modifier le comportement d'une table
Langage de manipulation de données (LMD)	SELECT	Sélectionner des données
	INSERT INTO	Insérer une ligne dans une table
	DELETE FROM	Supprimer une ligne dans une table
	UPDATE	Mettre à jour les valeurs d'attribut d'une table
Langage de contrôle de transaction (LCT)	COMMIT	Valider tous les transactions
	ROLLBACK	Annuler tous les transactions
Langage de contrôle de données (LCT)	GRANT	Passer des droits d'accès à un utilisateur
	REVOKE	Retirer des droits d'accès à un utilisateur

Le SQL s'utilise de trois manières. Soit par un programme écrit dans un langage de programmation quia de relation avec le SGBD pour lui transmettre des instructions en langage SQL. Soit par la technique dite « embedded » SQL qui est des instructions en langage SQL incorpore dans un code source d'un programme écrit dans un autre langage. Et la dernière c'est la technique de procédure stockée ou de « trigger » qui est une fonction écrite en langage SQL enregistré dans une base de données en vue d'être exécutée par le SGBD.

Cependant, il existe des autres langages tels qu'ALPHA définie par Codd non implanté, QUEL (Query English Language) qui est un système INGRES (Berkeley), DATALOG qui est une base de données déductive, le QBE (Query By Example) souvent offert au-dessus de SQL mais tous ces langages sont moins célèbres que le SQL qui est une version très standard plus utilisée.

1.8. Description du SGBDR MySQL

Presque les applicatifs et le site internet utilise le SGBDR MySQL et c'est pour cette raison le choix d'utiliser le MySQL comme serveur de base de données. MySQL est un SGBDR, qui utilise le langage SQL. C'est un des SGBDR les plus utilisés. Sa popularité est due en grande partie au fait qu'il s'agit d'un logiciel libre, ce qui signifie que son code source est librement disponible. Quiconque ressentit l'envie et/ou le besoin peut modifier MySQL pour l'améliorer ou l'adapter à ses besoins. Une version gratuite de MySQL est par conséquent disponible. À noter qu'une version commerciale payante existe également. Le logo du MySQL est un dauphin (Fig.1.16), nommé Sakila suite au concours « Name the dolphin » (nommez le dauphin).

Figure 1.16 : Logo du MySQL

Le développement de MySQL commence en 1994 par David Axmark et Michael Widenius. En 1995, la société MySQLAB est fondée par ces deux développeurs avec Allan Larsson. C'est dans cette même année sort la première version officielle de MySQL. En 2008, MySQLAB est rachetée par la société Sun Microsystems, qui est elle-même rachetée par Oracle Corporation en 2010.

MySQL est basé sur le modèle client-serveur. C'est-à-dire que la base de données se trouve dans un serveur. Pour interagir avec cette base de données, il faut utiliser un logiciel client qui va interroger le serveur et transmettre la réponse que le serveur lui aura donnée. Le serveur peut être installé sur une machine différente du client, c'est souvent le cas lorsque les données sont importantes.

Le SGBD MySQL peut supporter par tous les différents types de système d'exploitation. Il se peut installer seul ou avec des plateformes de développement web comme le WAMP (Windows Apache MySQL PHP) ou LAMP pour linux, etc.

Remarque :

Pour augmenter une disponibilité élevée des données, il fallait adapter à la méthode de redondance au niveau matériel et logiciel. Et pour lutter contre les accidents vis-à-vis aux catastrophes naturelles ou des incendies, etc. le mieux est de mettre en place une protection dite : « disaster recovery ».

Un serveur doit posséder une capacité de s'ajuster aux besoins de son utilisateur, donc il est essentiel d'utiliser un serveur possédant une propriété de scalabilité. C'est-à-dire il est possible de faire croitre les capacités de serveur au niveau de traitement, de stockage, etc.

1.9. Conclusion

Une base de données est un ensemble des informations quelconques. Grâce au système de gestion de base de données, ces données sont reliées et stockées avec un peu de redondance pour que les utilisateurs puissent les manipuler avec sécurité. Il existe plusieurs types de système de gestion de base de données mais le modèle le plus répandu d'aujourd'hui c'est le modèle relationnel qui est en concurrence avec le non relationnel. C'est pourquoi, on choisit d'utiliser un modèle relationnel. Pour la réalisation du projet, la connaissance à la programmation serait utile. Ainsi le prochain chapitre est consacré pour la programmation orientée objet avec le langage PHP.

CHAPITRE 2. PROGRAMMATION ORIENTEE OBJET AVEC LE LANGAGE PHP

2.1. Introduction

Ce chapitre montre les essentielles à connaitre concernant la programmation orientée objet en PHP ou Personal Home Page Hypertext PreProcessor. Le mode de programmation s'est évolué au cours du temps. Au début, la programmation est faite de manière procédurale. Dans ce cas, les codes sont traités étape par étape sans contrainte des autres lignes. Cette méthode requiert beaucoup d'effort en cas d'erreur, le programmeur doit changer totalement tous les lignes des codes. Après, le mode de programmation orientée objet est apparu. Dans ce mode, les codes sont distribués en plusieurs fichiers et qui sont liés entre elles. Cette méthode facilite la maintenance des codes en cas d'erreur ou d'évolution car juste certain fichier est modifié.

2.2. Programmation orientée objet

2.2.1. Définitions

Le mode de programmation orientée objet est une façon de programmer pour rendre le code facile à maintenir. En clarté, on se base sur la manière ou l'objet est manipulable et indépendante des autres éléments.
La connaissance très amplement de la programmation orientée objet dépend de quelques notions qui sont la notion de classe et la notion d'objet [06].

a. Qu'est-ce qu'une classe ?

Une classe est une définition de ce qu'on appelle un objet. Elle contient toutes les identités concernant un objet. En termes de programmation, elle porte les attributs qui sont les variables utilisées et les méthodes qui définissent les logiques d'applications. Donc une classe est un modèle de données définissant la structure commune à tous les objets qui seront créés à partir d'elle. Plus concrètement, une classe est considérée comme un moule grâce auquel on crée autant d'objets de même type et de même structure.
La création d'une classe se base sur la présence des variables qu'on appelle attributs et la présence des fonctions qu'on appelle méthodes.
La figure 2.1 montre la représentation graphique d'une classe.

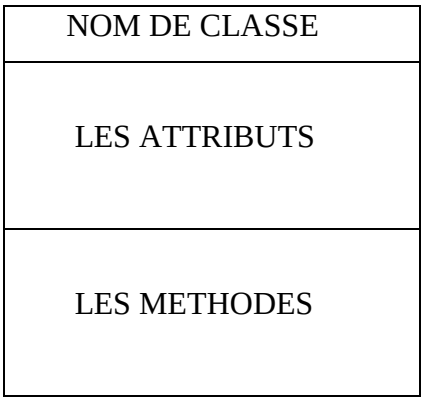

Figure 2.1 : Représentation graphique d'une classe

Le nom de classe est posé selon sa fonction pour faciliter la compréhension de cette classe. Les attributs sont les propriétés de la classe et les méthodes sont les fonctions contenues par cette classe.

b. Qu'est-ce qu'un objet ?

Un objet est une vue extérieure d'une chose sans se précipite de ce qu'il a formé ou de ce qu'il contient. En programmation on a un objet à partir de l'instanciation d'une classe et cet objet assure l'utilisation de l'appel de classe en dehors d'elle-même. C'est à dire on en extrait un nouvel objet qui dispose de ses attributs et de ses méthodes. L'objet ainsi créé aura pour type le nom de la classe.

2.2.2. Encapsulation et les méthodes d'accès

L'encapsulation assure le masquage du code à l'utilisateur (l'utilisateur est ici celui qui se servira de la classe, pas celui qui chargera la page depuis son navigateur). Le concepteur de la classe a englobé dans celle-ci un code qui peut être assez complexe et il est donc inutile voire dangereux de laisser l'utilisateur manipuler ces objets sans aucune restriction. Ainsi, il est important d'interdire à l'utilisateur de modifier directement les attributs d'un objet. Du coté code, l'encapsulation se réside à la notion d'assesseur (getter en anglais) et de mutateur (setter en anglais). Le premier assure la récupération de valeur et la deuxième assure l'affectation de valeur vient de l'extérieur.

Les méthodes d'accès sont la définition du moyen d'entrée à un attribut ou à une méthode. Elles peuvent être publiques, privée, protégé. Si un attribut ou une méthode est public, alors on pourra y avoir accès depuis n'importe où, depuis l'intérieur de l'objet (dans les méthodes qu'ont créé), comme depuis l'extérieur. La méthode d'accès privée impose quelque restriction. Il n'aura accès

aux attributs et méthodes seulement depuis l'intérieur de la classe. C'est-à-dire que seul le code voulant accéder à un attribut privé ou une méthode privée écrit à l'intérieur de la classe fonctionnera. L'accès protégé est de même que l'accès public sauf la classe fille (classe qui hérite de cette classe) qui a l'autorisation d'accès vient de l'extérieur.

2.2.3. Héritage

En programmation, si une classe hérite d'une autre, elle récupère ses attributs et ses méthodes. Elle peut en ajouter d'autres, voir même redéfinir ce qu'elle a récupéré. On dit que la classe fille hérite de la classe mère, cela signifie deux choses : les objets créés avec la classe fille pourront accéder aux attributs et aux méthodes de la classe mère (si l'accès est public évidemment). Les méthodes de la classe fille pourront réutiliser les attributs et les méthodes de la classe mère (si l'accès est public ou protégé) ou bien les redéfinir.

2.2.4. Abstraction

L'abstraction est une contrainte de l'héritage, il peut se présenter avec une classe ou avec une méthode.

a. Classe abstraite

Une classe abstraite est une classe qu'on ne peut pas instancier pourtant pour y accéder, on utilise les classes qui l'héritent car les classes dérivées peuvent accéder aux attributs et aux méthodes de leur classe parent. Alors la classe abstraite est une classe modèle pour les classes filles. Par exemple, si une classe forme est une classe abstraite donc la classe rectangle, la classe cercle, la classe carrée qui l'hérite, prennent comme modèle sa classe mère. Et si A hérite B alors B est un A. Elle crée par le mot clé « abstract » placé avant le mot clé « class ».

b. Méthode abstraite

Une méthode est dite abstraite si avant la déclaration de mode de visibilité de cette méthode, il y a le mot clé « abstract ». Toutes les classes qui héritent la classe que cette méthode placée, doivent redéfinir cette méthode car si les classes filles ne font pas cette action, une erreur se produise. La méthode abstraite est une méthode sans contenue c'est-à-dire sa déclaration se fait par : « abstract » visibilité « function» nomFonction (paramètre(s)).

Remarque :

Une méthode abstraite doit appartenir à une classe abstraite mais une classe abstraite n'est pas obligée d'emporter une méthode abstraite.

2.2.5. Finalisation

Dans le même cas de ce qu'on a vue auparavant c'est-à-dire la finalisation est une contrainte posée à une classe ou à une méthode.

a. Classe finale

Une classe est finale si un mot clé « final **»** se place avant le mot clé « class » de la déclaration d'une classe. Ce qui différencie avec la classe abstraite est qu'on ne peut pas faire d'héritage avec elle. Comme le sens de mot clé l'indique que tout ce qui est finale veut dire il n'y a jamais de suite. Alors il est inutile de mettre la classe en finale sauf si on est obligée pour raison de sécurité. Néanmoins, la classe finale peut instancier.

b. Méthode finale

La déclaration d'une méthode finale est faite en mettant le mot clé« final »à la même place que le mot clé « abstract » de la méthode abstraite. Pourtant, elle a des contenus et se comporte comme les autres méthodes normales. Elle peut exister dans toutes les classes qui existent sans contrainte. Si la classe qu'il l'emporte possède une classe fille, la méthode finale ne peut pas redéfinir dans cette classe fille mais la classe fille peut l'accéder.

Remarque :

Une classe finale n'est pas obligée d'emporter une méthode finale et une méthode finale peut exister dans tous types de classe.

2.2.6. Interface

Les interfaces sont des classes tout à fait abstraites. Les méthodes dans une interface n'ont pas de contenus et tous les classes qui implémentent une interface doivent les redéfinir. L'interface est utilisée pour indiquer des comportements à la classe qui l'implémente.

Pour créer une interface, on remplace le mot clé « class » par un mot clé « interface **»**. Elles sont inséparables au mode de programmation orientée objet pour assurer la lisibilité du code.

Remarque :

On ne peut pas faire d'héritage multiple dans une classe mais on peut implémenter une interface autant de fois que l'on souhaite. Et même une interface peut hériter une autre interface. De plus on ne peut pas écrire une méthode de même nom dans une interface.

2.2.7. Diagramme UML

Avec la programmation orientée objet, la compréhension d'un code est très facile pour tous ceux qui ne savent pas programmer par le biais de l'UML (Unified Modeling Language), ou "langage de modélisation unifié" en français. L'UML est un langage permettant de modéliser les classes et leur interaction. Cela s'effectue par un diagramme. Il n'est pas un langage à proprement parler, plutôt une sorte de méthodologie. Il se reproduit à partir d'une logicielle spécifique pour une création de diagramme.

Pour créer un diagramme UML, des règles de construction et d'écriture doivent suivre, si ce n'est pas le cas, le diagramme est inutilisable. La création de diagramme peut aider aussi les développeurs à comprendre la base de son code car presque le logiciel fait ce travail avec un outil de génération de code dans son interface [07].

2.3. Programmation orientée objet avec le langage PHP

Il existe beaucoup de langages de programmation orienté objet, comme : C++, C#, Python mais c'est le PHP qui est le plus utilisé.

Le langage PHP est un langage de programmation libre, principalement utilisé pour produire des pages web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. Il s'agit d'un langage impératif orienté objet.

Ce langage est créé en 1994 par *Rasmus Lerdorf* pour son site web. *Rasmus* a alors décidé, en 1995 de publier son code, pour que tout le monde puisse l'utiliser et en profiter. PHP s'appelait PHP/FI (Personnal Home Page tools/FormInetrpreter). En 1997, deux étudiants, *Andi Gutmans* et *ZeevSuraki* ont développé le cœur de PHP/FI. Le mode programmation orienté objet est équipé dans ce langage depuis 2003. Ce mode de programmation orientée objet s'est évolué de jours en jours. La dernière version du PHP est le 7.3.7 daté du 04 juillet 2019. Le fichier contenant un script PHP possède une extension (php ou phtml ou php4 ou php3 ou php5 ou phps ou phar, etc.). Le code PHP doit se mettre à l'intérieurde balise< ? php ... ?>.

2.3.1. Mots clés utilisés en PHP orienté objet

L'intérêt de l'existence de la classe en PHP est la possibilité de créer de l'objet facilement en l'instancie par l'utilisation de mot « new » et après la création de l'objet, on peut accéder les attributs et les méthodes de l'objet en utilisant l'opérateur flèche (→). Alors l'inclusion de fichier comme on fait tous les temps dans programmation procédurale se diminue.

Le mot « static » sert à définir les variables statiques et à l'appel statique. Le « self » est pour l'appel à la classe elle-même. Le « parent » est pour l'appel à la classe parent. Le « $this » est pour l'appel à l'objet courant.

L'appel à la classe elle-même et l'appel à la classe parent se font par l'utilisation de l'opérateur de résolution de portée qui est le double deux points (**::**).Et de même aussi pour accéder aux attributs constants, aux attributs statiques et les méthodes statiques de la classe. Ainsi on n'a pas besoin de créer un objet pour faire ses actions car elles sont des attributs et des méthodes de la classe.

Le mot clé « extends » pour marquer la présence d'un héritage entre deux classes différentes.

Pour qu'une classe peut bénéficier les comportements que l'interface contient, il suffit de l'implémenter par le mot clé « implements **».**

2.3.2. Anonyme

La classe anonyme est une classe crée sans nom mais se comporte comme toutes les classes normales. C'est-à-dire elle possède des attributs et des méthodes. Elle est utilisable avec les « design pattern » ou les architectures de conceptions qui sont un des notions développées dans un autre sous-titre.

Et voilà pour la classe, ce principe est applicable aussi avec la méthode. Cette technique s'appelle « clossure » qui est une méthode sans nom. Cette méthode possède la même caractéristique que la méthode normale : peut avoir des arguments, des instructions à l'intérieure d'elle. On peut faire de recourt à cette méthode si la logique d'application est incertaine [08].

2.3.3. Résolution statique à la volée

La résolution statique à la volée se focalise à l'utilisation du mot clé « static » avec un opérateur de résolution de portée. Ce mot ait la même fonction que le mot clé « self » qu'on

aborde auparavant sauf il est utilisé aussi pour appeler la classe afin d'invoquer des méthodes ou accéder à des attributs de la classe à l'extérieure d'elle.

2.3.4. Hydratation

L'hydratation est la façon de récupérer les données viennent de la base de données. Ces données sont partagées aux attributs de la classe de l'application selon leur attribut qui est logiquement de même nom avec les colonnes de la table de référence dans la base de données. Donc hydrater un objet, c'est tout simplement lui apporter ce dont il a besoin pour fonctionner. En d'autres termes plus précis, hydrater un objet revient à lui fournir des données correspondant à ses attributs pour qu'il assigne les valeurs souhaitées à ces derniers.

2.3.5. Auto-chargement de classe

L'auto-chargement de classe est une méthode créée pour assurer l'appel automatique de fichier. Le fichier qui la contient avoir une possibilité d'appeler un autre fichier en cas de besoin. En remarquant que le nom de la classe qu'on veut appeler automatiquement doit porter le même nom que son fichier. Cette action est réalisable en utilisant la fonction interne de PHP qui est le « spl_autoload_register » après la définition de la méthode d'auto-chargement de classe. Cette méthode fait l'appel des fichiers par l'utilisation de « require ».
L'auto-chargement de classe est très indispensable au fonctionnement de tous les « framework » utilisés pour le développement des applications web.

2.3.6. Méthode magique

Une méthode magique est une méthode qui, si elle est présente dans une classe, sera appelée lors de tel ou tel évènement. Si la méthode n'existe pas et que l'évènement est exécuté, aucun effet spécial ne sera ajouté. Le but des méthodes magiques est d'intercepter un évènement, dire de faire ceci ou cela et retourner une valeur utile pour l'évènement. Voilà des exemples qui peuvent aider à se familiariser sur cette notion : __construct () est une méthode appelée à chaque construction d'objet. Le contraire à elle c'est la __destruct (). Le __set () et le __get () comme son nom indique, ils servent à faire d'assesseur et de mutateur sans préoccupation de créer de méthode pour affecter et récupérer de valeur. Le __isset () pour vérifier l'existence de variable. Le __unset () pour enlever une variable ; etc.

2.3.7. Interfaces prédéfinis en PHP

Les interfaces prédéfinies sont les interfaces près à employer dans le langage PHP. Grâce à certaines, la modification du comportement de l'objet est devenue plus facile. Il y a beaucoup d'interfaces prédéfinies mais les essentiels sont les suivants [09].

a. Itérateur

Un itérateur est un objet capable de parcourir un autre objet. Pour réaliser cette action, il faut que l'objet même soit itératif en implémentant l'interface « iterator **»** dans une classe. En implémenter cette interface on peut modifier le comportement de l'objet lorsqu'il est parcouru. L'interface « iterator » possède 5 méthodes qui sont : « current **»** qui renvoie l'élément courant, « key **»** retourne la clé de l'élément courant, « next **»** déplace le pointeur sur l'élément suivant, « rewind **»** qui remet le pointeur sur le premier élément. Et « valid **»** a pour rôle de vérifier si la position courante est valide.

b. Interface « seekableIterator »

L'interface « seekableIterator » est une interface qui hérite l'interface « iterator ». De plus il a sa propre méthode qui est le « seek ». Son but est de mettre le curseur interne à une position précise. C'est-à-dire cette méthode comporte une variable qui est la position voulue.

c. Interface « ArrayAccess »

C'est une interface qui transforme un objet comme un tableau. Cette interface possède 4 méthodes qui sont les suivants : la méthode « offsetExists **»** est une méthode sert à vérifier si une clé dans un objet tableau est existé et n'est pas vide. La méthode «offsetGet **»** est utilisée pour récupérer la valeur correspondant à la clé passée en paramètre de cette méthode. La méthode « offsetSet **»** a pour but d'assigner une valeur à une entrée et elle a deux paramètres qui sont la valeur de la clé et la valeur qu'on veut assigner. Et en dernier la méthode « offsetUnset **»** est appelée lorsqu'on veut enlever une valeur dans l'objet tableau, elle prend un paramètre qui est la clé de la valeur à supprimer.

d. Interface « countable »

Cette interface sert pour compter les nombres d'éléments existant dans un objet tableau. Pour utiliser cette interface, il faut qu'on implémente en même temps avec l'interface

« ArrayAccess ». Et pour pouvoir parcourir un objet, il vaut faire une implémentation de l'interface « seekableIterator ». L'interface « countable » possède une méthode qui est la méthode « count ».

Remarque :

PHP est un langage qui facilite la vie car on est déjà à penne vue les 4 interfaces pour créer un objet tableau. Or tout ça existe dans une seule interface qui est l'interface « ArrayIterator ». Cette interface contient la même fonctionnalité à l'assemblage de ces quatre derniers. Ainsi, il suffit d'utiliser « ArrayIterator » pour implémenter ces quatre interfaces dans une classe. Pourtant la connaissance de leurs fonctions est importante.

2.3.8. Générateur

Un générateur n'est pas du tout une interface mais c'est une classe appelée « Generator » qui implémente une interface « iterator ». C'est-à-dire un générateur est un itérateur et dans le générateur on se focalise sur la méthode « next » de l'interface « iterator ». Il a pour but de résoudre les problèmes de performance ou de code rallongé et plus précisément pour assurer l'optimisation de mémoire. Pour créer un générateur, il suffit d'écrire une seule fonction. Dans cette fonction, on va parcourir les lignes du fichier et pour chaque ligne on va indiquer à PHP qu'il s'agit de la valeur de la prochaine entrée du tableau grâce au mot-clé « yield ».

2.3.9. Trait

Les traits sont un moyen d'externaliser le code. Ils définissent des méthodes que les classes peuvent utiliser pour résoudre au problème de duplication de code.
Un trait n'est autre qu'une mini-classe. Dedans, on pourrait déclarer des méthodes et des attributs. Ensuite, pour l'utiliser déclarons une classe et dans cette classe, le trait l'intègrera grâce au mot-clé « use » suivi du nom de trait. En utilisant ce mot-clé, toutes les méthodes et attributs du trait vont être importés dans la classe.

2.3.10. Annotation

L'annotation sert pour rechercher une chose. En anglais l'annotation pourra facile à comprendre par le mot « mapping » c'est-à-dire l'annotation est utilisée pour lier des choses. C'est un peu difficile à expliquer mais grâce à la bibliothèque « addendum » du langage PHP, on peut s'informer de plus près.Pour utiliser une annotation, en premier temps il faut que le

fichier qui contient la chose à utiliser soit inclure dans le fichier de travail en utilisant le « require » ou « include » ou utilise le « namespace » ce qui veut dire le dossier de travail courant. On peut mettre des annotations autant de fois que l'utilisateur voudrait sans restriction. L'utilisation de l'annotation est marquée par l'existence de l'opérateur « @ ».

2.3.11. Exception

Ce sous-titre est dédié pour s'informer qu'il est intéressant de connaitre tous types des erreurs. Afin que l'utilisateur du produit puisse connaitre la faute qu'il a commise pendant son utilisation. En négligeant l'exception, il est possible que l'utilisateur ait vu des menaces ou restriction qui apparait très laid avec diffèrent forme de chiffre, code, etc. Or si on le prend soin on peut envoyer un beau texte à l'utilisateur quand il fait des mauvaises insertions de valeurs dans le formulaire d'entrée. Les exceptions peuvent apparaitre sous forme des erreurs fatales, des alertes, des erreurs d'analyse ou encore des notices. Pour gérer les erreurs, on peut utiliser les différentes classes d'exceptions déjà existant dans PHP. Pour lancer les classes d'exceptions, on utilise le mot clé **«** throw **»** avant l'instanciation de la classe d'exception. La boucle « try **»** et « catch **»** servent à attraper ces exceptions.

Il y a beaucoup de types de classe d'exception dans PHP, voici des exemples : la classe « Exception » lui-même, la classe « PDOException », la classe « RuntimeException », la classe « InvalidArgumentException », etc. Pourtant, on n'est pas obligé d'utiliser tous ces classes là mais on peut créer une classe d'exception selon l'utilité. Cette classe doit hériter la classe « Exception ».

2.3.12. API de réflexivité

L'API (Application Programming Interface) de réflexivité va se servir pour obtenir les informations concernant une classe. On instancie la classe « ReflectionClass » avec comme paramètre le nom de classe qu'on veut avoir des informations. Ou utiliser la classe « ReflectionObject » qui est une classe héritant de la classe « ReflectionClasse ».
Pour connaitre les informations internes de la classe, on fait appel aux méthodes statiques de la classe « ReflectionClass ». Voici des exemples : la méthode « hasProprety » est utilisé pour questionner à propos des attributs. Il y a la méthode « hasMethod » pour s'informer sur les méthodes. Pour récupérer la classe parente on utilise « getParentClass ». Cette méthode renvoie la classe parente s'il y en a une : la valeur de retour sera une instance de la classe « ReflectionClass » qui représentera la classe parente. Si la classe ne possède pas de parent,

alors la valeur de retour sera « false ». Pour savoir le nom de la classe, on utilise la méthode « getName ». Il a beaucoup de méthode statique dans la classe « ReflectionClass » pour prendre des informations dans une classe et ces exemples cités sont ample pour la compréhension de son fonctionnement.

2.3.13. Motif de conception

C'est un peu difficile de traduire le terme « design pattern » en français. Au sens du terme : il est les moyens réalisés par des développeurs pour lutter contre au problème de conception pendant la phase de développement. Il est à pour but d'aider les autres développeurs à ne pas tarder sur le même cas de difficulté. Il y en a beaucoup avec le langage PHP mais voici quelques indications pour la compréhension de son utilisation.

a. Motif de conception fabrique

Le but de création est d'empêcher l'instanciation de la classe à l'intérieure du code de développement. C'est-à-dire on va créer une méthode statique qui a pour rôle de contenir l'instanciation de classe à utiliser. Une méthode statique est une méthode de la classe mais pas une méthode de l'objet donc on peut le récupérer en utilisant l'opérateur de résolution de portée. Cette architecture était née à cause de l'utilité de changer par exemple une classe. Or si cette classe est utilisée par une autre classe, cette classe doit changer aussi. Pourtant si on instancie dans une méthode, il suffit de changer une seule méthode au lieu de changer une classe tout entière.

La figure 2.2 montre l'utilité de ce modèle de conception. La classe A et B utilisent la classe Fabrique. Donc tous les comportements indiqués dans l'interface Fabrique sont implémentés dans A et B, en redéfinissant toutes les méthodes dans chaque classe. Ces deux classes A et B héritent la classe C.

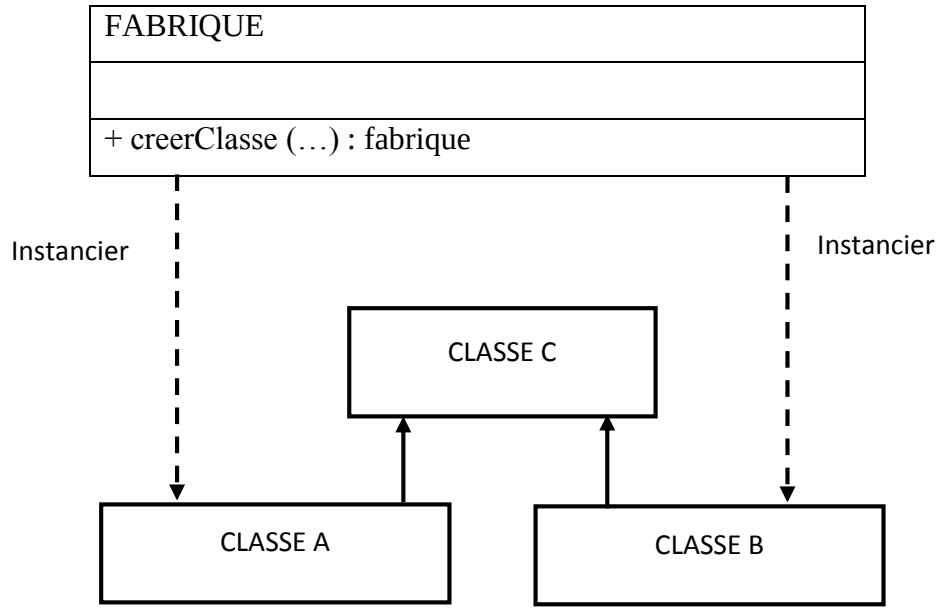

Figure 2.2 : Motif de conception fabrique

b. Motif de conception observateur

Le but est de créer des classes qui s'écoutent entre eux : l'un prend une place de l'observée et l'autre prend une place de l'observatrice (Fig.2.2). Cette fonctionnalité peut se réaliser à l'utilisation de l'interface **«** SplSubject **»** du côté de l'observée et « SplObserver **»** pour le côté d'observatrice. C'est les classes qui implémentent l'interface « SplObserver » ont une fonction de visualiser ce qui se passe dans une seule classe qui implémente l'interface « SplSubject ».

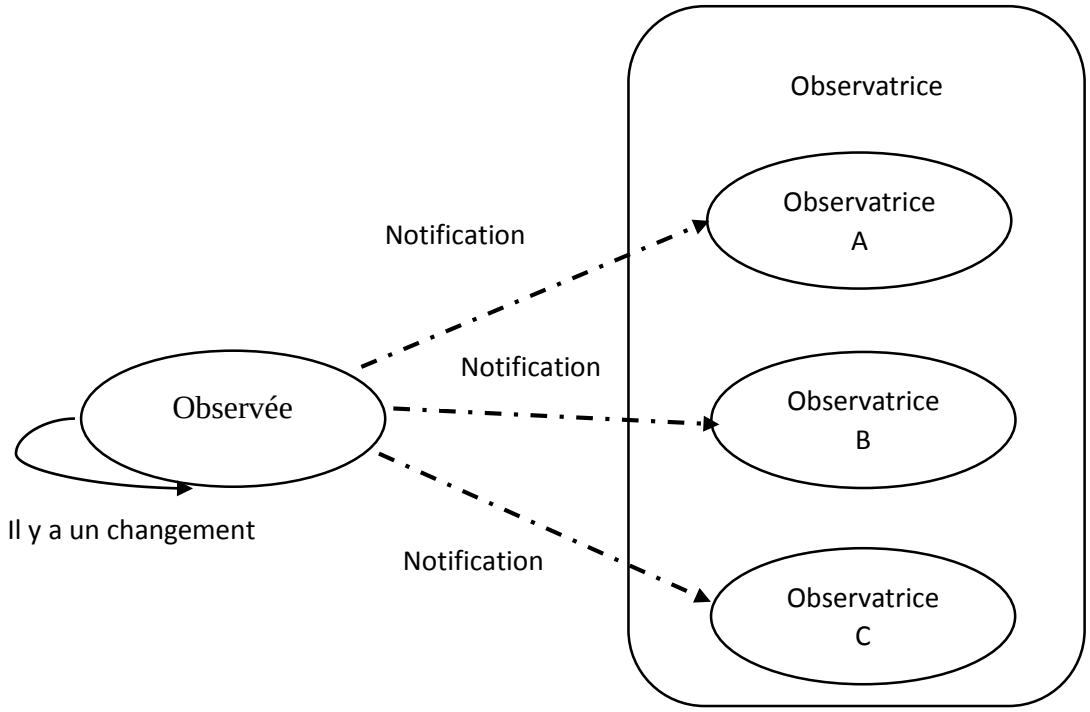

Figure 2.3 : Motif de conception observateur

c. Motif de conception façade

Le but est de distribuer l'algorithme qui réalise des fonctions différentes selon leur spécificité. En créant une interface commune pour ces différentes classes pour lutter aux duplications répétitives des comportements communs pour chaque classe (Fig.2.4).
Prenons un exemple pour l'expliquer : on peut écrire en diffèrent format comme en format html, xml, texte, etc. Or chaque format possède des caractéristiques identiques car si on écrit, on doit penser qu'il peut mettre à jour, le supprimer, le formater, le dupliquer. Tant d'action peut réaliser et alors il est inutile d'écrire toutes ces comportements dans chaque format mais il suffit de mettre dans une interface et tous les formats peuvent l'implémenter.

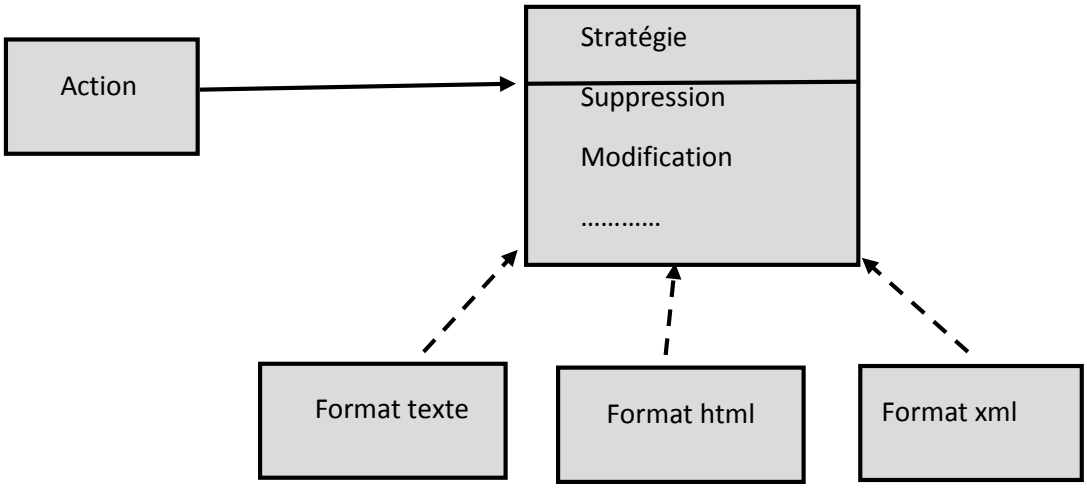

Figure 2.4 : Motif de conception façade

d. Motif de conception singleton

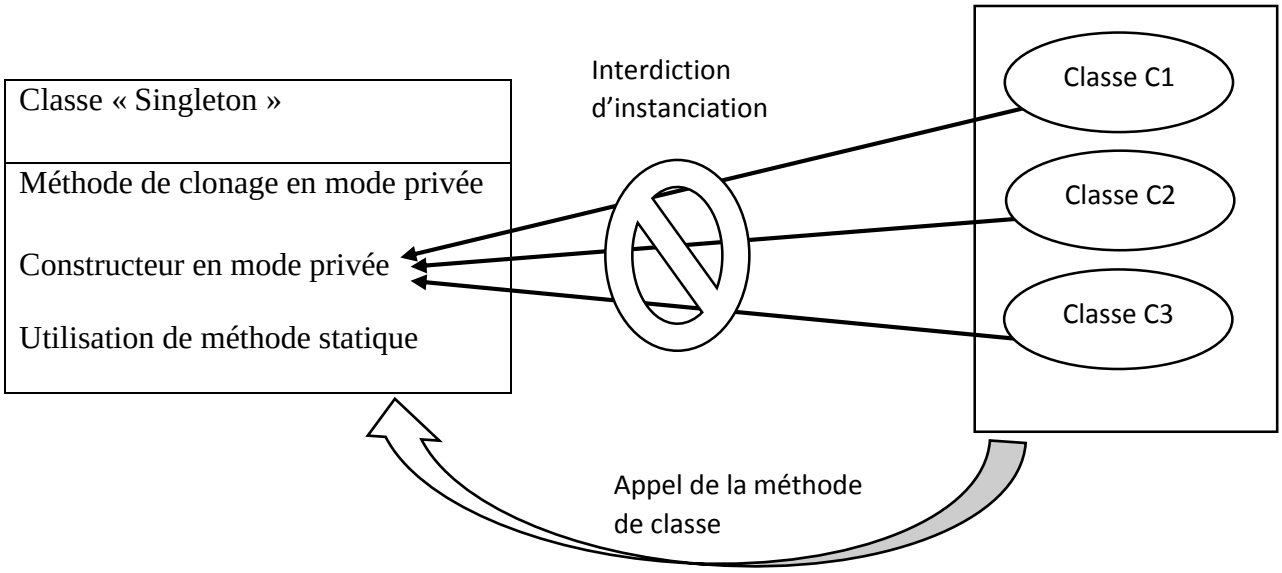

Figure 2.5 : Motif de conception singleton

Le but de ce modèle de conception est d'empêcher la création d'un objet plus d'une seule fois. Et aussi pour empêcher le clonage de l'objet crée. La mauvaise utilisation de cette architecture entraine une création d'anti-conception. Pour le réaliser, il faut que le programmeur le créer à partir d'une méthode statique et fait une instance de classe à l'intérieur d'elle-même. Pour utiliser cette classe, on a recouru à l'appel de méthode de classe.

e. Injection de dépendance

L'injection de dépendance est née à cause de nombreuses classes dépendantes les unes des autres. L'injection de dépendance consiste à découpler les classes. Le but de cette architecture est de mettre les classes dépendantes pour ne pas revenir au problème de réécriture totale de chaque classe en cas d'erreur dans une classe. L'injection de dépendance est inséparable à tous les projets de développement informatique.

f. Motif de conception MVC

Le MVC (Modèle-Vue-Contrôleur) est le modèle de conception la plus utilisée pour se raisonner à la façon de se coder. Le but de MVC est justement de séparer la logique du code en trois parties que l'on retrouve dans des fichiers distincts. La figure 2.6 suivante indique les rôles de chaque partie.

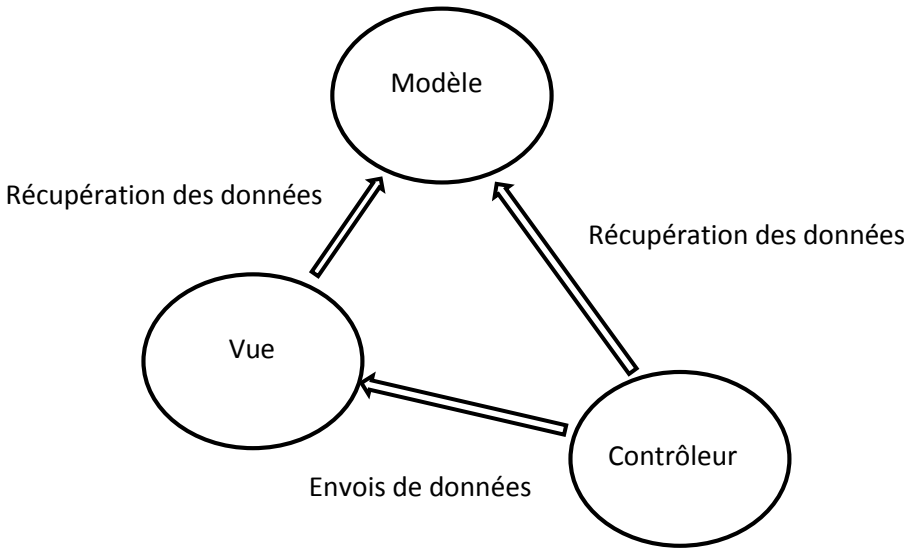

Figure 2.6 : Motif de conception MVC [10]

- **Modèle** : cette partie gère les données du site. Son rôle est de récupérer les informations brutes dans la base de données, de les organiser et de les assembler pour qu'elles puissent ensuite être traitées par le contrôleur. On y trouve donc entre autres les requêtes SQL.
- **Vue :** cette partie se concentre sur l'affichage. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples, ou des éléments de langage spécifiée pour un affichage.
- **Contrôleur :** cette partie gère la logique du code qui prend des décisions. C'est en quelque sorte l'intermédiaire entre le modèle et la vue. Le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. Le contrôleur contient exclusivement du PHP ou utilise des bibliothèques prédéfinies avec PHP.

2.4. Mode d'accès à la base de données MySQL

Pour pouvoir travailler avec une base de données en PHP, il faut d'abord s'y connecter. Il va donc falloir que PHP s'authentifie. Il établit une connexion avec MySQL. Une fois que la connexion sera établie, PHP pourrait effectuer les actions demandées et il se rend compte qu'il a besoin de MySQL.

Dans un réseau informatique, l'ordinateur client est généralement un ordinateur personnel ordinaire, équipé de logiciels. Il envoie des demandes à un serveur. Ce dernier est à la fois un ensemble de logiciels, l'ordinateur les héberge. Son rôle est de répondre de manière automatique, via un réseau, les demandes envoyées par des clients ordinateurs. Apache (le serveur web) cherche dans son arborescence si le fichier existe. Apache transmet ce fichier au parseur PHP. Ensuite, PHP effectue les actions demandées. Il parse le fichier, c'est à dire qu'il va analyser et exécuter le code PHP qui se trouve entre les balises < ? PHP et ?>. Si ce code contient des requêtes vers une base de données MySQL, PHP envoie la requête SQL. La base de données renvoie les informations voulues au script qui peut les exploiter (pour les afficher par exemple). PHP continue de parser la page, puis retourne le fichier dépourvu du code PHP au serveur web.

La figure 2.7 représente le principe d'interaction de PHP avec MySQL.

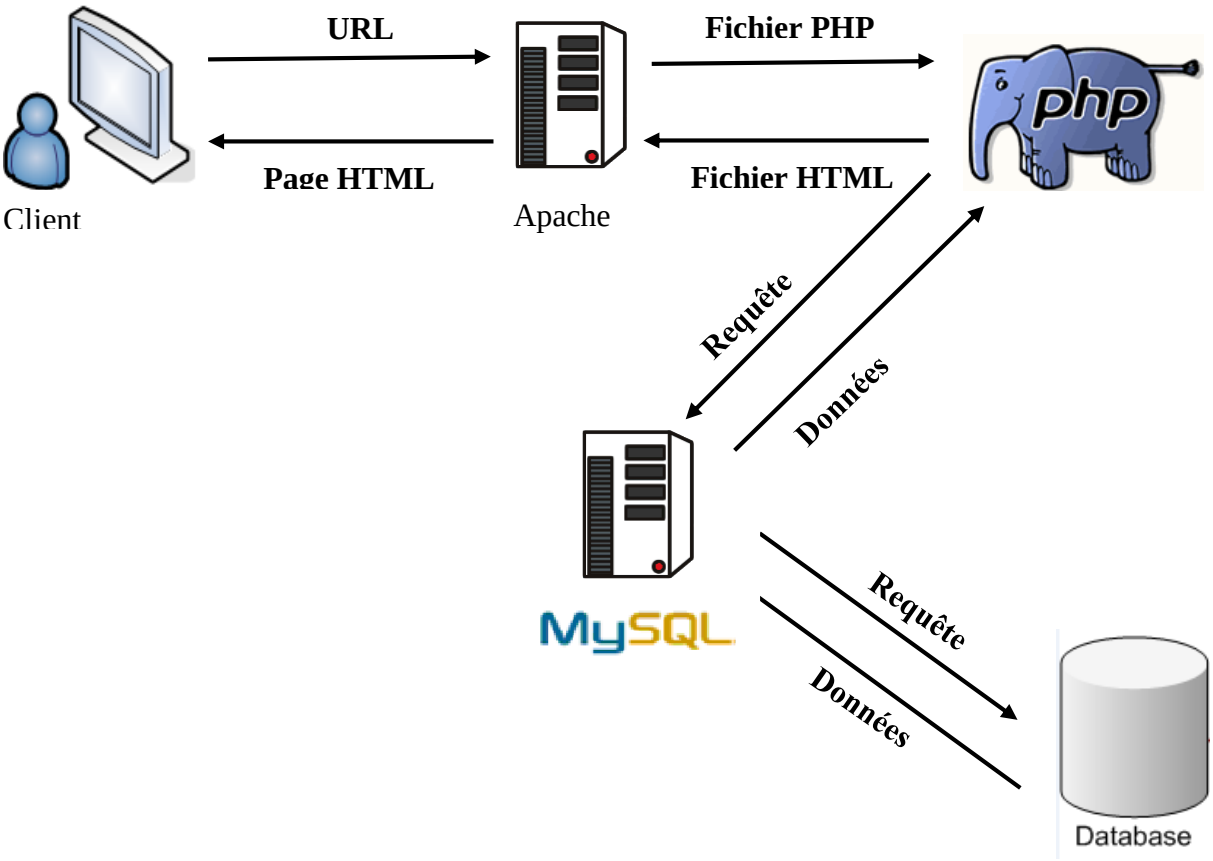

Figure 2.7 : Principe d'interaction du PHP avec MySQL

PHP propose plusieurs moyens de se connecter à une base de données MySQL : l'extension mysql est une fonction qui possède un nom commence par mysql. Toutefois, ces fonctions sont vieilles et on recommande de ne plus les utiliser aujourd'hui. Les extensions mysqli sont des fonctions améliorées pour l'accès à MySQL. Elles proposent plus de fonctionnalités et sont plus à jour. L'extension PDO (PHP Data Object) : c'est un outil complet qui permet d'accéder à n'importe quel type de base de données avec une utilisation de même manière sans changer de fonction. On peut donc l'utiliser pour se connecter aussi bien à MySQL que PostgreSQL ou Oracle, etc. Comme notre façon de développement est orienté objet, il est préférable d'utiliser l'extension PDO qui est spécifié au langage PHP et à la façon de programmer en orientée objet.

2.5. Spécificité de PDO

PDO (PHP Data Objects) est une extension définissant l'interface pour accéder à plusieurs types de base de données.

L'extension PDO comporte trois classes : la classe PDO correspond à une connexion à la base de données. La classe « PDOStatement » représente d'une part une requête préparée et d'autre part le jeu de résultat de la requête une fois qu'elle est exécutée. Cette classe offre des méthodes de parcours, de comptage, d'information. Et la classe « PDOException » représente une erreur émise par PDO.

L'accès à la base de données se fait en instanciant un objet PDO. Les paramètres à indiquer au constructeur sont : la source de la base de données ; optionnellement, le nom d'utilisateur et le mot de passe.

2.6. Conclusion

Dans cette partie, la plupart des termes et les indications avec l'utilité de chaque fonction qui existe dans PHP orienté objet sont tout à fait abordées. L'orientée objet est très avantageux par rapport au procédural en termes de lisibilité de code. La façon de se raisonner en orientée objet est très précise. Le motif de conception est un aspect d'orientation sur l'environnement de travail. Le choix d'accès à la base de données est nécessaire pour faciliter les actions qui peuvent réaliser avec les données. Le chapitre qui va suivre est consacré pour la conception et la réalisation du projet.

CHAPITRE 3 : CONCEPTION ET REALISATION DU PROJET

3.1. Introduction

Le but est de créer un site web pour manipuler une base de données. Sa création se fait automatiquement grâce à l'intermédiaire d'un outil qui capte les configurations de la base à utiliser. La conception de ce projet va suivre la méthodologie de génie logiciel. Des interfaces sont présentées tous au long de ce chapitre pour savoir les divers fonctionnements du projet.

3.2. Analyse du projet

La méthode d'une conception informatique se base toujours sur la méthodologie adoptée dans un génie logiciel. Une connaissance en génie logiciel est alors nécessaire.

3.2.1. Génie logiciel

Le génie logiciel est l'ensemble des activités de conception et de mise en œuvre des produits et des procédures tendant à rationaliser la production du logiciel et son suivi.
L'objectif étant de produire de « bons logiciels ». Le génie logiciel se préoccupe des procédés de fabrication des logiciels de façon à s'assurer que les quatre critères suivants soient satisfaits :
- le système fabriqué répond aux besoins ou exigences des utilisateurs (correction fonctionnelle),
- la qualité correspond au contrat de service initial,
- les coûts et les délais restent dans les limites prévues au départ.

3.2.2. Notion de cycle de vie

Un cycle de vie est un ensemble séquentiel de phases, dont le nom et le nombre sont déterminées en fonction des besoins du projet. Il permet généralement de faire un développement de service ou de produit. Il existe plusieurs modèles de cycle de vie : le modèle en V, le modèle en cascade, le modèle en spiral, etc.
Le mode en V est choisi particulièrement dans le cadre de ce mémoire car il est simple et intuitif à utiliser. Ce modèle est une façon de présenter une démarche qui reste linéaire [11].

Il apparaît mieux les produits intermédiaires à des niveaux d'abstraction et de formalité différents. Les procédures d'acceptation de ces produits intermédiaires sont bien organisées. Le

V est parcouru de gauche à droite en suivant la forme de la lettre. Les activités de construction précèdent les activités de validation et vérification. L'acceptation est préparée dès la construction (flèches de gauche à droite). Cela permet de mieux approfondir la construction et de mieux planifier la remontée. La figure 3.1 englobe les 9 étapes d'un cycle de vie du modèle en V.

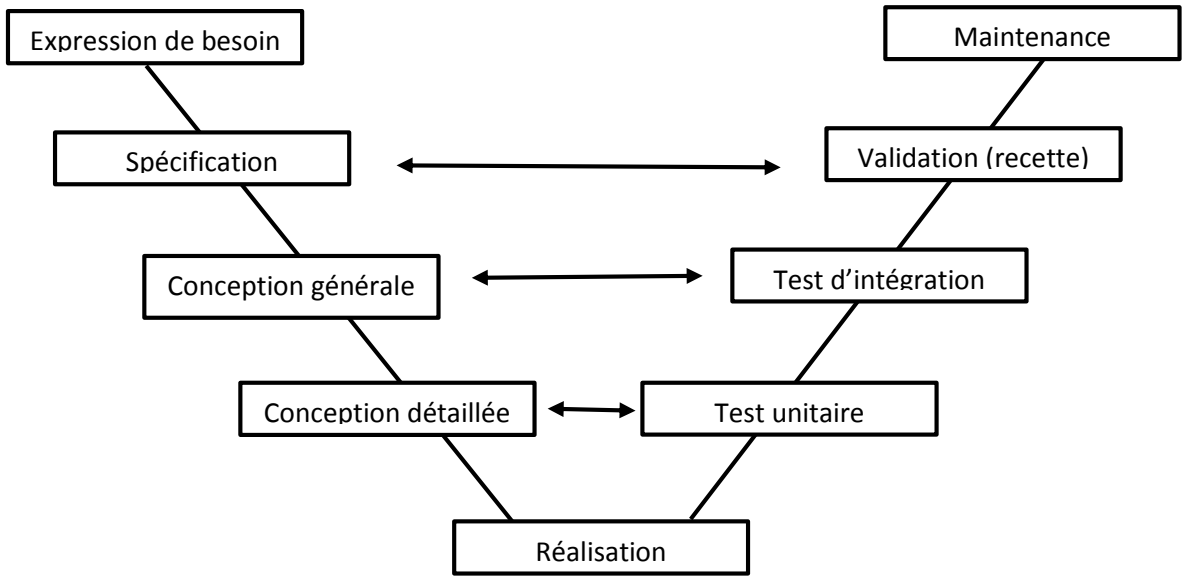

Figure 3.1 : Model en V d'un cycle de vie [11]

L'expression du besoin : le client va exprimer sa demande. Une analyse va être effectuée ensuite par l'équipe pour pouvoir présenter une solution intéressante au client.

Les spécifications : les spécifications fonctionnelles sont un cahier des charges permettant de décrire de manière complète le produit, les différents cas d'utilisation du produit et ses différentes fonctionnalités. Généralement, elles sont faites par le chef de projet tout en restant en contact avec le client pour pouvoir valider les fonctions du produit. Un cahier de recette est rédigé en parallèle avec les spécifications fonctionnelles.

La conception architecturale (générale) : Sous forme des maquettes, ces dernières vont présenter l'interface graphique qui sera développer par un web master et qu'elle sera l'interface client ou utilisateur final.

La conception détaillée : ce sera les spécifications fonctionnelles avec les maquettes pour être plus explicite. Ce document est donné au client pour qu'il valide la version finale avant le commencement des développements.

Le codage (ou développement) : l'équipe commence les développements des fonctionnalités du produit tout en respectant les spécifications détaillées (Spécifications fonctionnelles + Maquettes). Il doit respecter le contrat avec le client et réaliser la demande du client.

Les tests unitaires : après les développements des fonctionnalités du produit, l'équipe réalise les tests unitaires. C'est un test de chaque méthode qui fait un traitement spécifique.

Les tests fonctionnels : cette étape consiste à faire des tests sur chaque fonctionnalité. Les testeurs peuvent passer par plusieurs méthodes pour assurer le bon résultat.

Les tests de validation : permet de vérifier si toutes les exigences client décrites dans le document de spécification d'un logiciel sont respectées. Ce document est écrit à partir de la spécification des besoins. Les tests de validation se décomposent généralement en plusieurs phases :

- *La validation fonctionnelle* : les tests fonctionnels vérifient que les différents modules ou composants implémentent correctement les exigences client.
- *La validation solution* : les tests solutions vérifient les exigences client d'un point de vue d'utilisation. Généralement ces tests sont des tests en volume. Chaque grand cas d'utilisation est validé par lot. Ensuite, tous les cas d'utilisation sont validés ensemble. L'intérêt est de valider la stabilité d'une solution par rapport aux différents modules qui la composent, en soumettant cette solution à un ensemble d'actions représentatif de ce qui sera fait en production.
- *La validation performance et robustesse* : les tests de performance vont vérifier la conformité de la solution par rapport à ses exigences de performance. Les tests de robustesse vont essayer de mettre en évidence des éventuels problèmes de stabilité et de fiabilité dans le temps.

La recette et la maintenance : Le cahier de recette est réalisé en parallèle avec les spécifications fonctionnelles. Il contient les descriptions des fonctionnalités des spécifications fonctionnelles. L'équipe utilise ce cahier pour tester ces fonctionnalités sur le produit pour pouvoir détecter les erreurs, les corriger, afin d'assurer formellement que le produit est conforme aux spécifications.

3.2.3. Étude du besoin

Parmi les 9 étapes du cycle de vie en V, l'expression de besoin doit mettre en place en premier. Il y a deux sortes de besoins : les besoins fonctionnels et les besoins non fonctionnels.

a. Besoins fonctionnels

Ce projet permet de réaliser un site web utilisé pour la manipulation d'une base de données. En générale, la réalisation d'un site se réfère toujours aux éléments qui existaient dans la base de données.

Mais dans le cas présent, il n'a plus besoin de connaitre personnellement les éléments internes de la base pour réaliser un site. L'objectif est de créer un site d'une façon automatique tandis que les autres sites sont codés manuellement par des développeurs. Un outil va récupérer les tables existées dans une base de données et il va créer automatiquement le code source d'un site web qui correspond à cette base de données. Des insertions, suppressions ou modifications des données sont permis grâce au site obtenu.

b. Besoins non fonctionnels

Les besoins non fonctionnels décrivent toutes les contraintes qui ne traitent ni de la description du métier des utilisateurs, ni de la description applicative. C'est les besoins techniques qui vont aborder. Ils sont importants car ils agissent de façon indirecte sur le résultat et sur le rendement de l'utilisateur. Pour cela il faut répondre aux exigences de :

- *La fiabilité* : l'application doit fonctionner de façon cohérente sans erreurs, c'est à dire à chaque fois que l'utilisateur fait un mauvais type de syntaxe, l'outil renvoie de message d'indication.
- *L'ergonomie et bon IHM (Interface Homme Machine)* : l'application doit être adaptée à l'utilisateur sans qu'il fournisse trop d'effort (utilisation claire et facile).
- *L'efficacité* : l'application doit permettre l'accomplissement de la tâche avec le minimum de manipulations.
- *La sécurité* : l'application doit être sécurisée au niveau des données avec un contrôle d'accès pour faire la manipulation de données.

3.2.4. Motif de conception utilisé

Pour développer ce projet, le motif de conception MVC (Modèle Vue Contrôleur) qui a pour objectif d'organiser la réalisation de l'application et à séparer complètement la présentation (Vue) des données (Modèle) et des traitements (Contrôleur) est utilisé. Le fonctionnement du modèle MVC a indiqué la Fig.3.2.

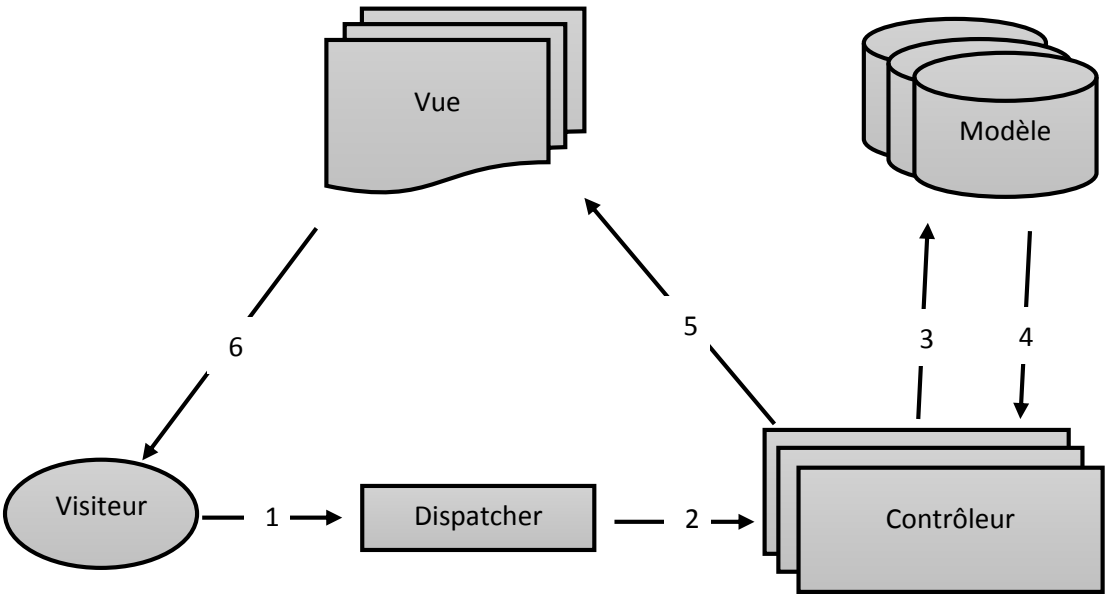

Figure 3.2 : Déroulement d'un modèle MVC [13]

3.3. Conception du projet

En génie logiciel, la réalisation des diagrammes conduit à une bonne compréhension concernant le mode d'utilisation du projet à réaliser. Il existe plusieurs diagrammes mais les quatre diagrammes (le diagramme de cas d'utilisation, le diagramme de séquence, le diagramme d'activité et le diagramme de classe) sont assez pour décrire le fonctionnement de ce projet.

3.3.1. Diagramme de cas d'utilisation

Les diagrammes de cas d'utilisation sont des diagrammes UML permettant d'avoir une vision globale du comportement fonctionnel du projet. Ces diagrammes identifient les fonctionnalités du système, les utilisateurs (appelés acteurs) et leurs interactions. Leurs objectifs sont d'identifier les acteurs du système, de fournir une vue d'ensemble de quoi le système pourra servir. Il permet ainsi de mieux cerner les secteurs nécessitant une interface homme-machine. Un acteur peut se représenter symboliquement par un bon homme qui doive être identifié par son nom au-dessous. En générale, les acteurs ne font pas partie du système (ils interagissent avec le système) [14].

La figure 3.3 représente le diagramme de cas d'utilisation correspond au site web obtenu :

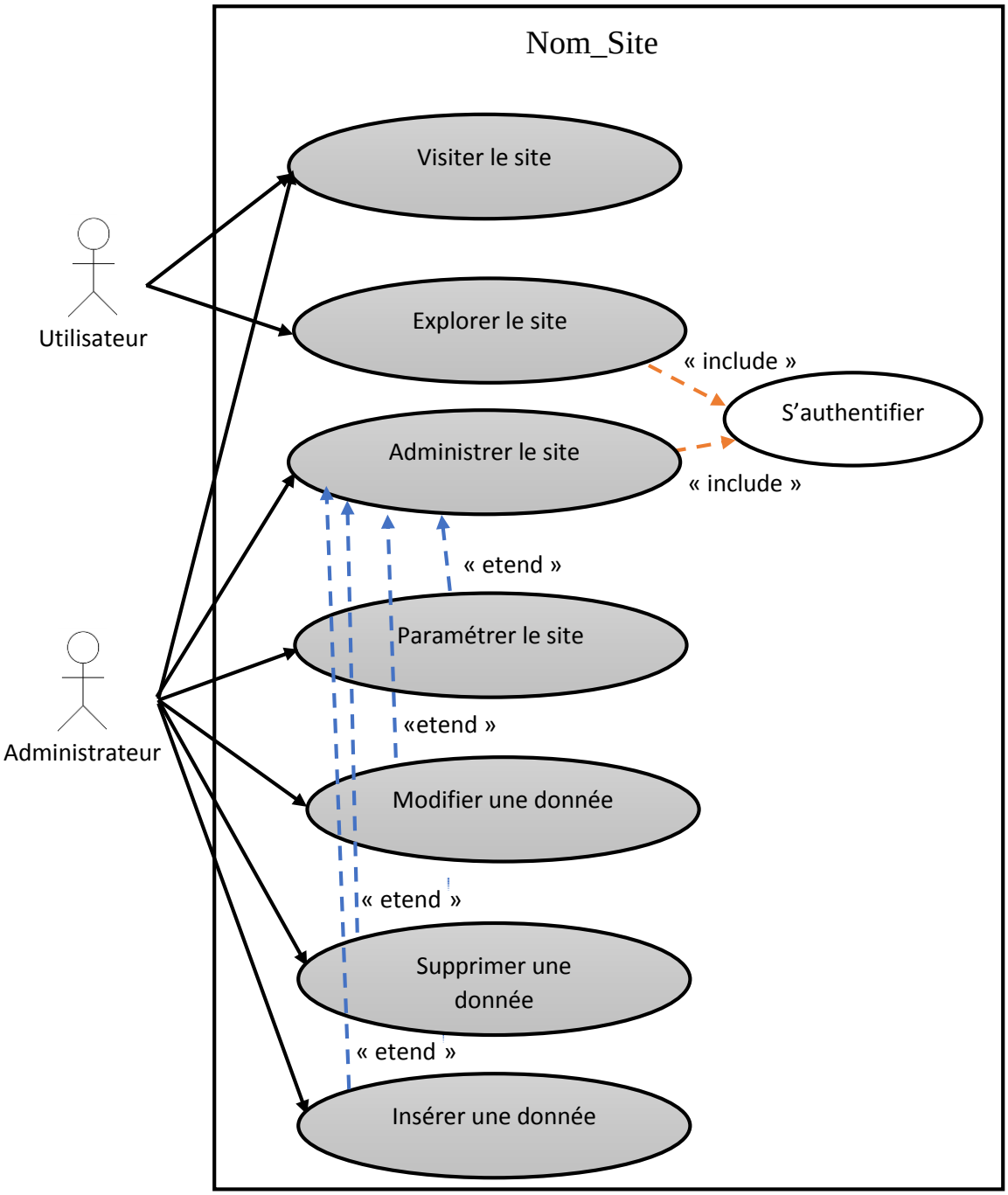

Figure 3.3 : Diagramme de cas d'utilisation du site

3.3.2. Diagramme de séquence

Le diagramme de séquence assure la validation des cas d'utilisation pour comprendre la logique de l'application. Il complète le diagramme de cas d'utilisation en mettant en évidence les objets et leurs interactions d'un point de vue temporel. Un diagramme de séquence se représente globalement dans un grand rectangle avec indication du nom du diagramme précédé d'un symbole sd (sequence diagramm).

La figure.3.4 indique le diagramme de séquence du site obtenu en connectant avec le compte d'un simple utilisateur.

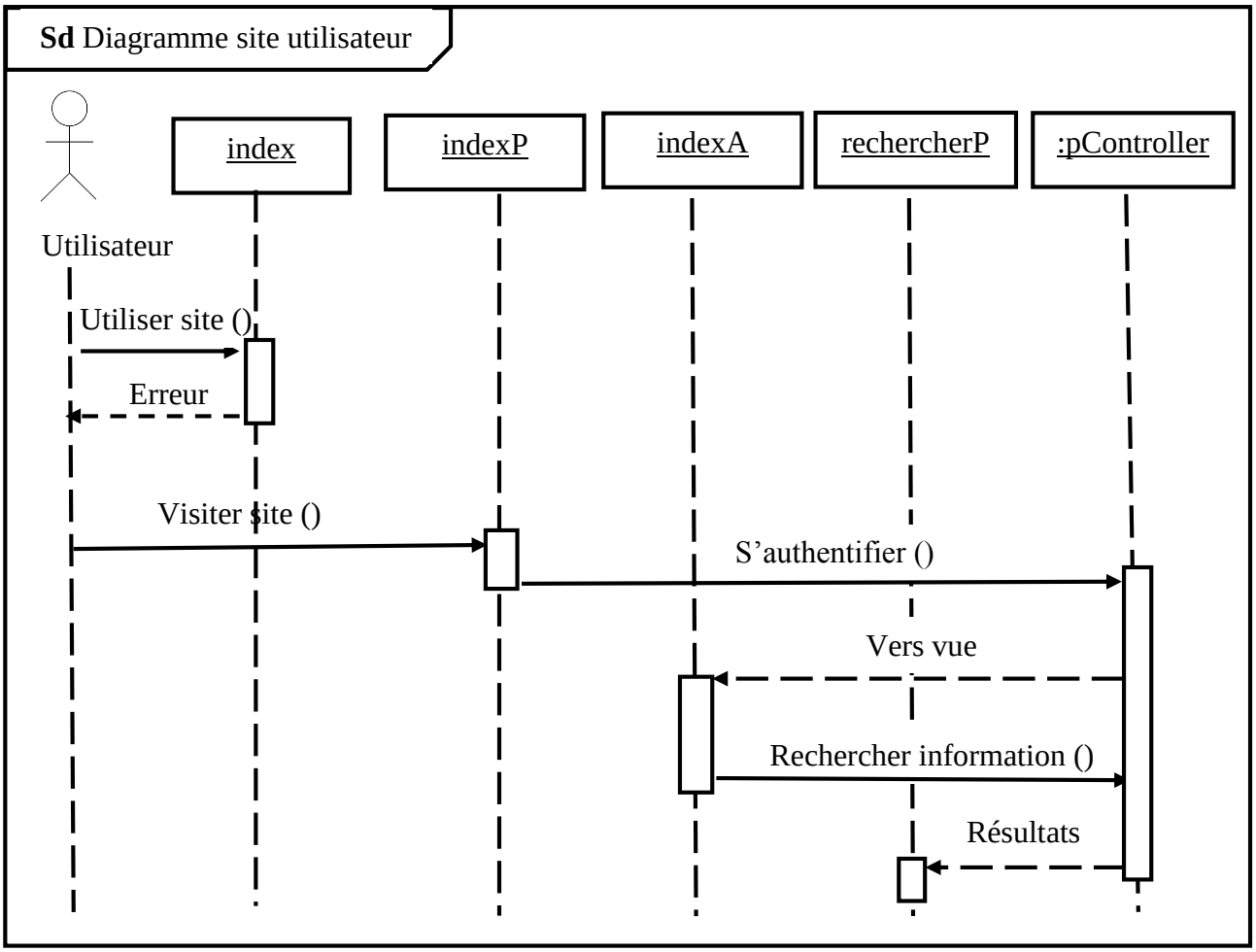

Figure 3.4 : Diagramme de séquence pour un simple utilisateur

Rappelons que deux types d'utilisateur existent : un simple utilisateur et un administrateur. Pour un simple utilisateur, la modification des données n'est pas autorisée, de même pour la suppression et l'insertion. Son intérêt est de s'informer, il peut rechercher les informations que l'administrateur lui a octroyées.

En se connectant avec un compte administrateur, la manipulation des données est autorisée. En plus, l'administrateur peut personnaliser les couleurs de la barre et le logo du site. Son diagramme de séquence est représenté à la Fig.3.5.

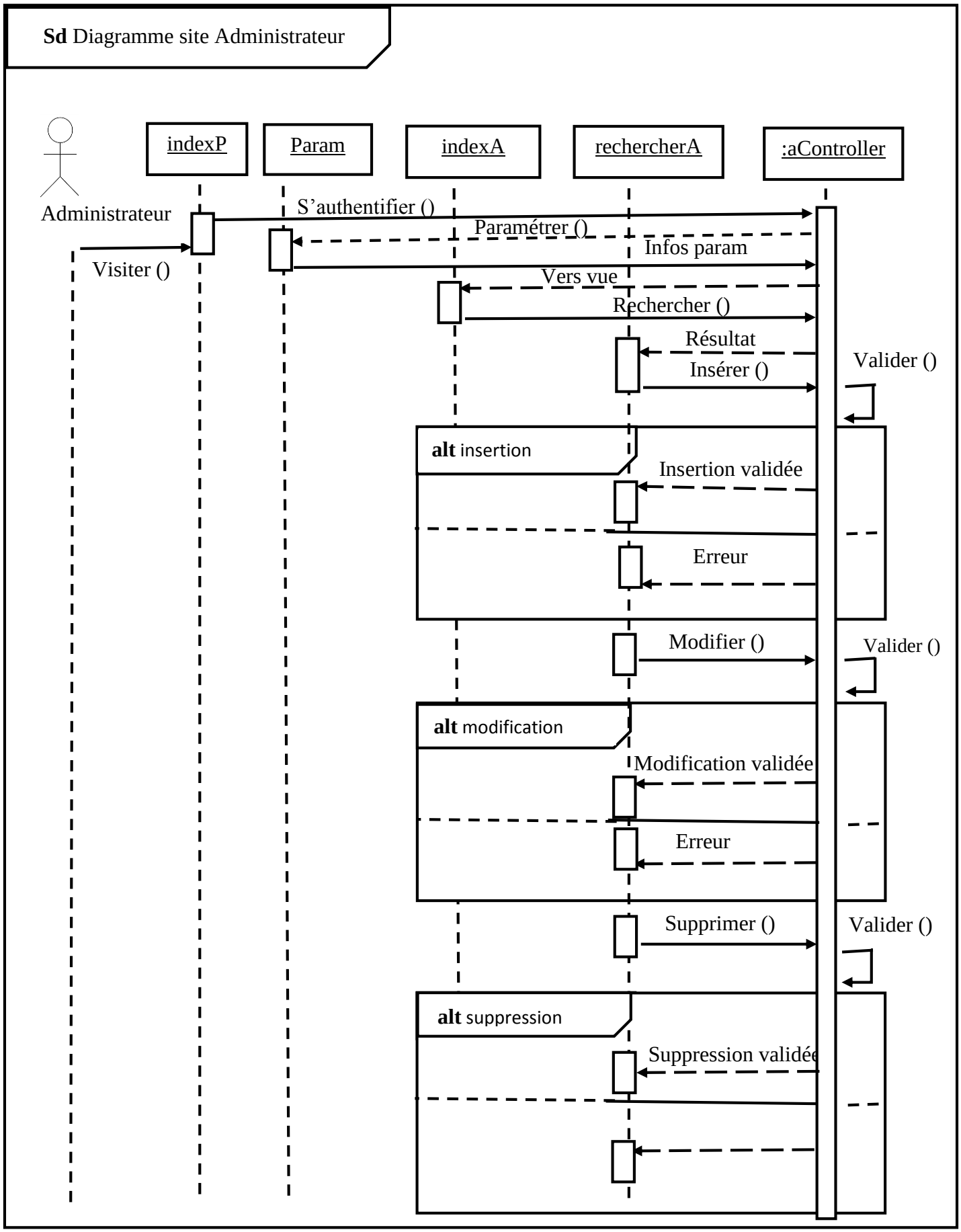

Figure 3.5 : Diagramme de séquence pour l'administrateur

3.3.3. Diagramme d’activité [15]

Le diagramme d'activité est un diagramme comportemental d'UML, permettant de représenter le déclenchement d'événements en fonction des états du système et de la modélisation des comportements en parallèle. Il permet de modéliser un processus interactif, global ou partiel pour un système donné (logiciel, système d'information). Il est recommandable pour exprimer une dimension temporelle sur une partie du modèle.

La figure 3.6 représente le digramme d’activité du site.

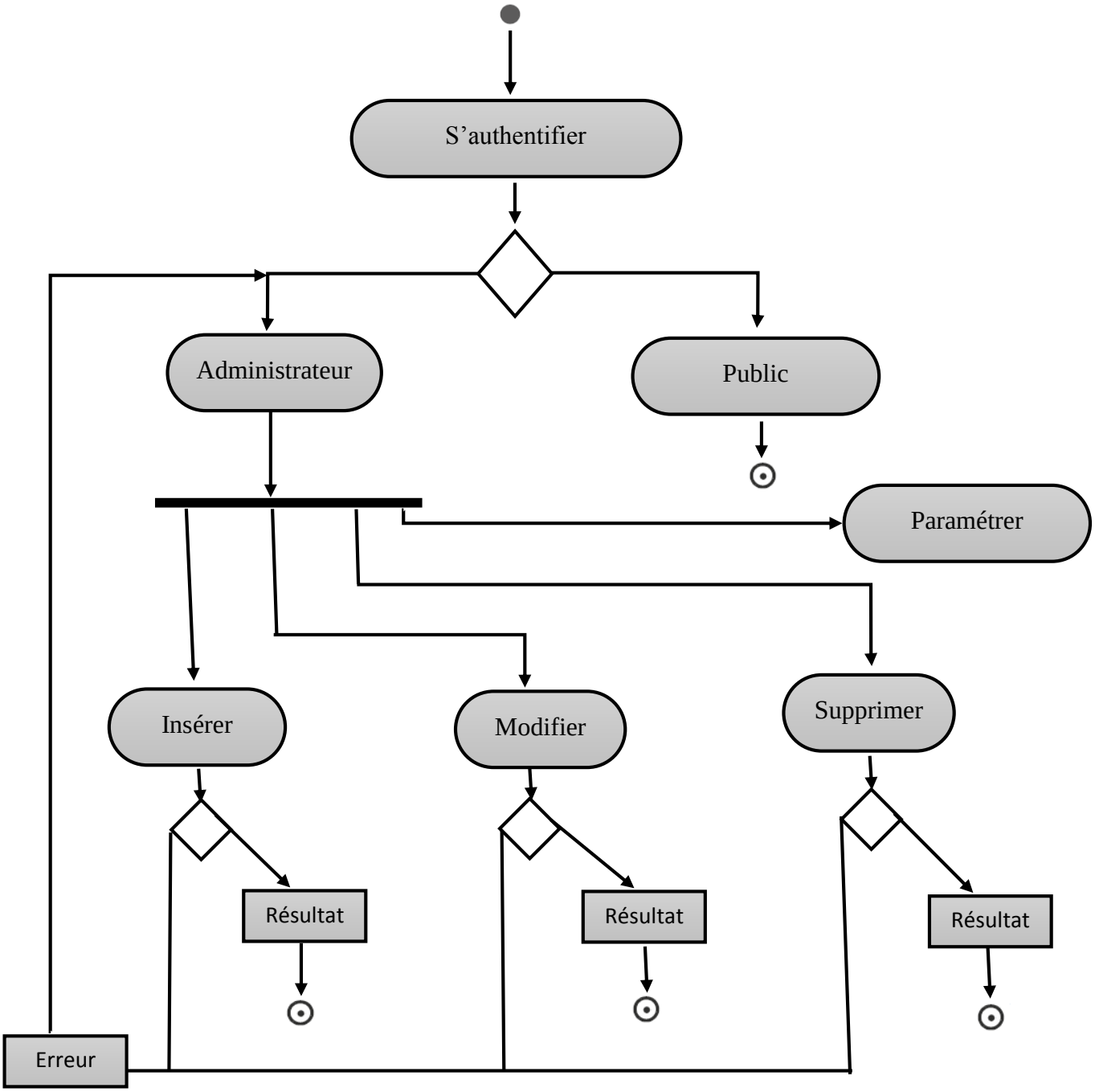

Figure 3.6 : Diagramme d’activité du site

Légende :

 Indique un nœud initial (état initial),

 Indique un nœud final (état final),

 Indique une action (correspond à un traitement qui modifie l'état du système

 Indique un résultat de traitement,

 Indique un nœud de décision (choix).

 Indique un nœud de bifurcation (permet à partir d'un flot unique entrant de créer plusieurs flots concurrents en sortie de la barre de synchronisation).

Ce diagramme d'activité résume le mode de fonctionnement du site obtenu. Ce dernier possède deux fonctionnalités selon le type de visiteur soient utilisateur simple ou administrateur. D'où l'existence d'un nœud de décision après l'authentification présenté dans ce diagramme. La partie utilisateur simple considérée généralement comme public, peut bénéficier les données que l'administrateur lui a octroyées. D'autre part l'administrateur peut faire l'action d'insertion, de modification, de suppression, et le paramétrage du site selon leur gout.

3.3.4. Diagramme de classe

Une classe décrit un ensemble d'objets qui partagent les mêmes attributs, opérations, références et sémantique. Elle est une abstraction car elle met en évidence certaines caractéristiques et supprime d'autres caractéristiques. Le diagramme de classes a toujours été le diagramme le plus important dans toutes les méthodes orientées objet. C'est également celui qui contient la plus grande gamme de notation et de variante. Ce diagramme centralise l'organisation des classes de conception, c'est lui qui se transforme le plus aisément en code. Ce diagramme est le plus simple possible pour que tous les mondes puissent le comprendre sans difficulté. Il indique tous ceux qui existent au niveau de code.
Le diagramme de classe correspondant à note projet est représenté par Fig.3.7.

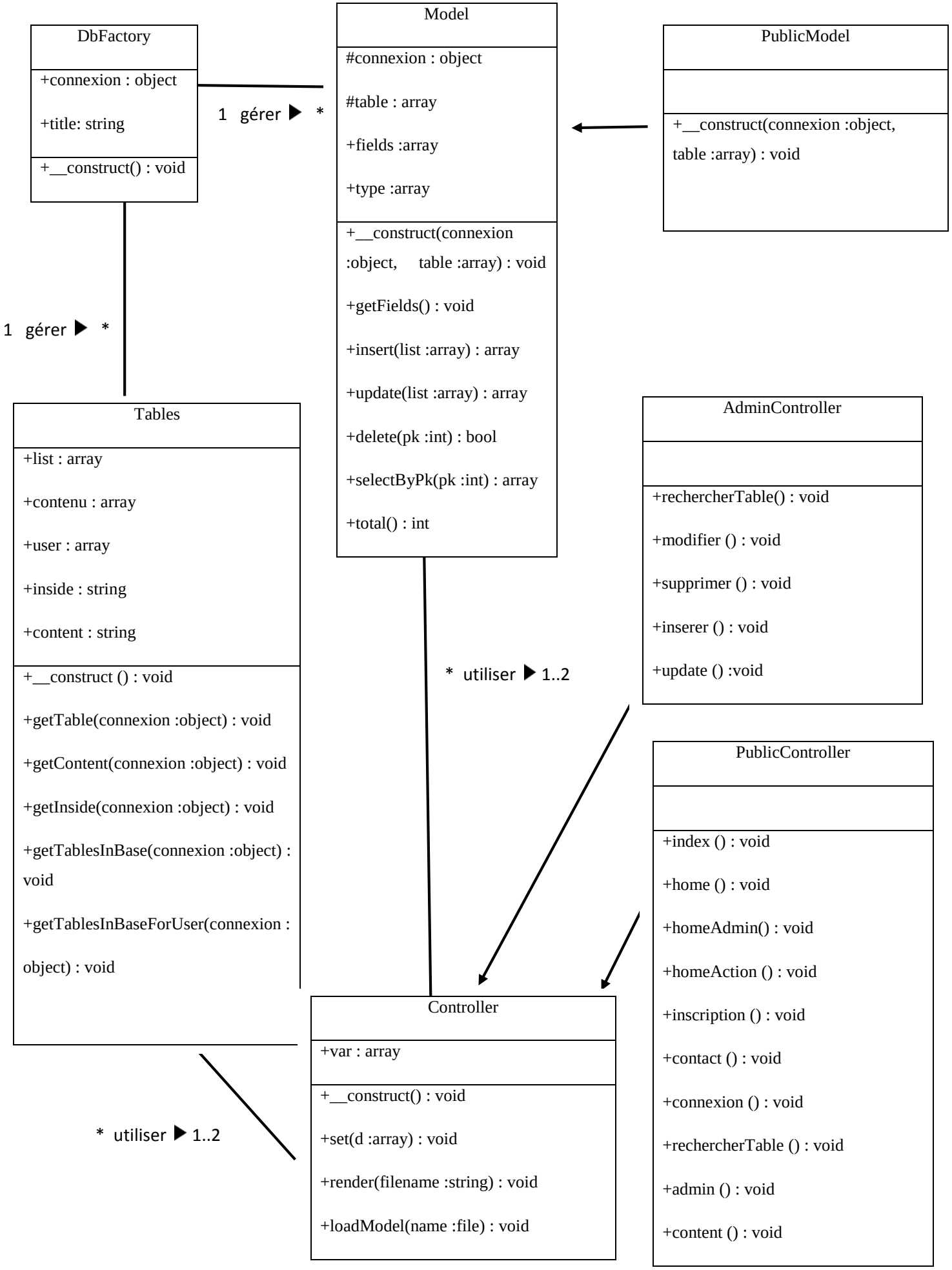

Figure 3.7 : Diagramme de classe du site obtenu

3.4. Réalisation du projet

Le site est apte à toutes les bases qui peuvent exister dans un serveur de base de données MySQL. Le but est de consulter, modifier, supprimer et insérer des données à partir de ce site. L'accès à la base de données est autorisé à tous les moments et n'importe quel lieu si le site est hébergé.

3.4.1. Environnement de travail

Quelques outils de développement sont nécessaires pour réaliser le projet. Il y a le serveur PHP, le serveur de base de données MySQL et le serveur Apache. Le WAMP englobe ces trois serveurs. Il est utilisé avec un système d'exploitation Windows. Il existe plusieurs paquetages tous prêts pour Windows et le logiciel WAMP Server est l'un d'entre eux. Il a son propre avantage et peut être mis à jour régulièrement. Le style du maquettage est géré par les « framework » de mise en forme « Bootstrap », « JQuery » et « Popper ». L'éditeur de texte utilisé est le PhpStorm. C'est un éditeur de texte plus avancé et contient des documentations en langage PHP. Ce dernier est disponible pour Windows, MacOs et Linux. Le choix de l'environnement de travail n'a pas de raison précise, c'est juste une préférence.
L'architecture de ce travail s'oriente sur l'architecture MVC. Cette architecture est très facile à manipuler et à réaliser. Pour rappel, MVC est un motif de conception qui propose une solution générale au problème de la structuration d'une application. La structuration des fichiers appartenant à cet outil respecte alors la structure MVC.

3.4.2. Arborescence de fichiers

Etant donné que l'architecture utilisée est le MVC, le dossier de travail doit contenir au minimum trois sous dossiers qui sont le contrôleur (Controllers), le modèle (Models) et la vue (Views). Ces trois dossiers sont placés dans un même dossier nommé « projet ». Le sous dossier corps (Core) est complémentaire pour contenir les configurations. Le fichier « .htaccess » assure la redirection de la requête du client dans le fichier PHP « index.php ». Ce fichier joue le rôle de dispatcheur, c'est-à-dire il assure l'appel de la classe et de la méthode qui existe dans le dossier de travail pour réaliser la demande de l'utilisateur. Pour le sous-dossier « template » de la vue, il contient les fichiers qui assurent le rendu visuel du code pour les utilisateurs.

La figure 3.8 montre l'arborescence du fichier de ce projet.

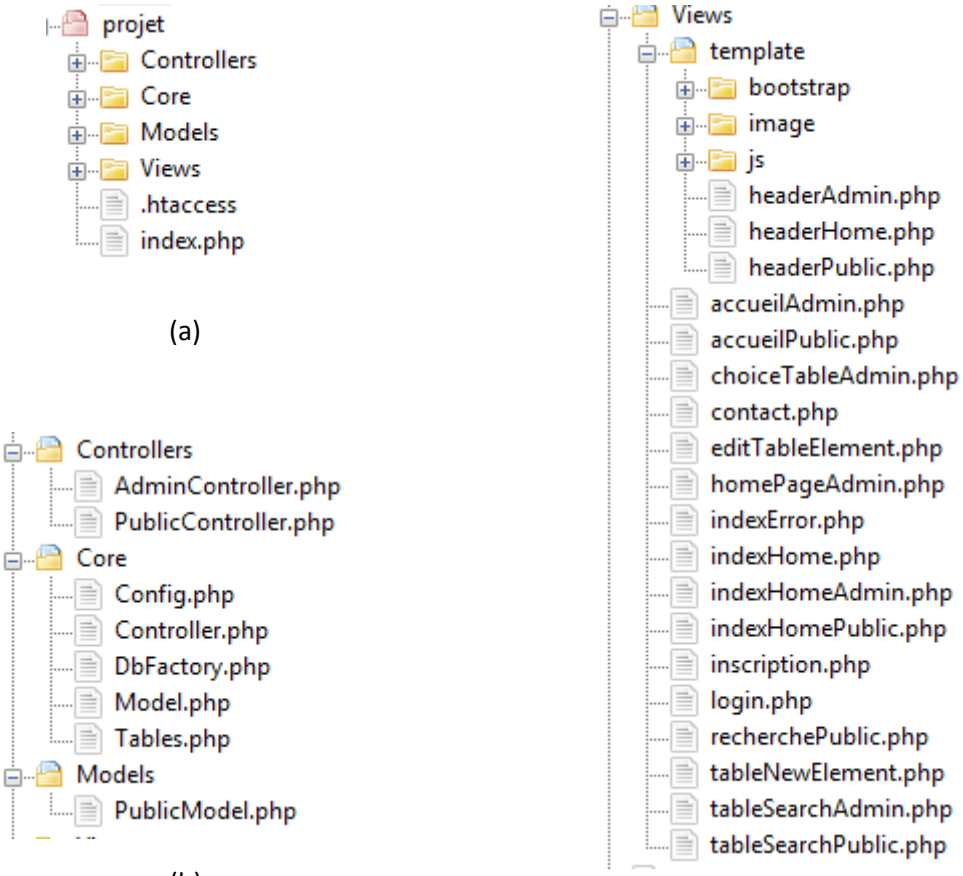

Figure 3.8 : (a)Dossiers du projet, (b) Fichiers des sous dossiers contrôleur, corps et modèle, (c) Fichiers de sous dossier vue

3.4.3. Vues appropriées au site

Pour utiliser le site, il est préférable de connaitre le mode de fonctionnement donné par le fournisseur. Le site peut marcher indépendamment dès qu'il est configuré. C'est-à-dire après que toutes les informations contenant les configurations de la base sont bien enregistrées.

a. Page d'accueil du site

Le site va utiliser la base de données « zone_forestier », son titre est choisi depuis l'insertion de la configuration. Le site obtenu par cette base de données porte le titre « zone ecologique ».

L’interface du site comporte le titre et le logo du site. Ce logo est personnalisé par l’administrateur. Les barres de navigation contiennent la minimum utilité pour un site. Ces menus peuvent ajouter ou enlever par le propriétaire du site sauf le menu de connexion qui assure la sécurité du site. La figure 3.9 montre l’interface du site.

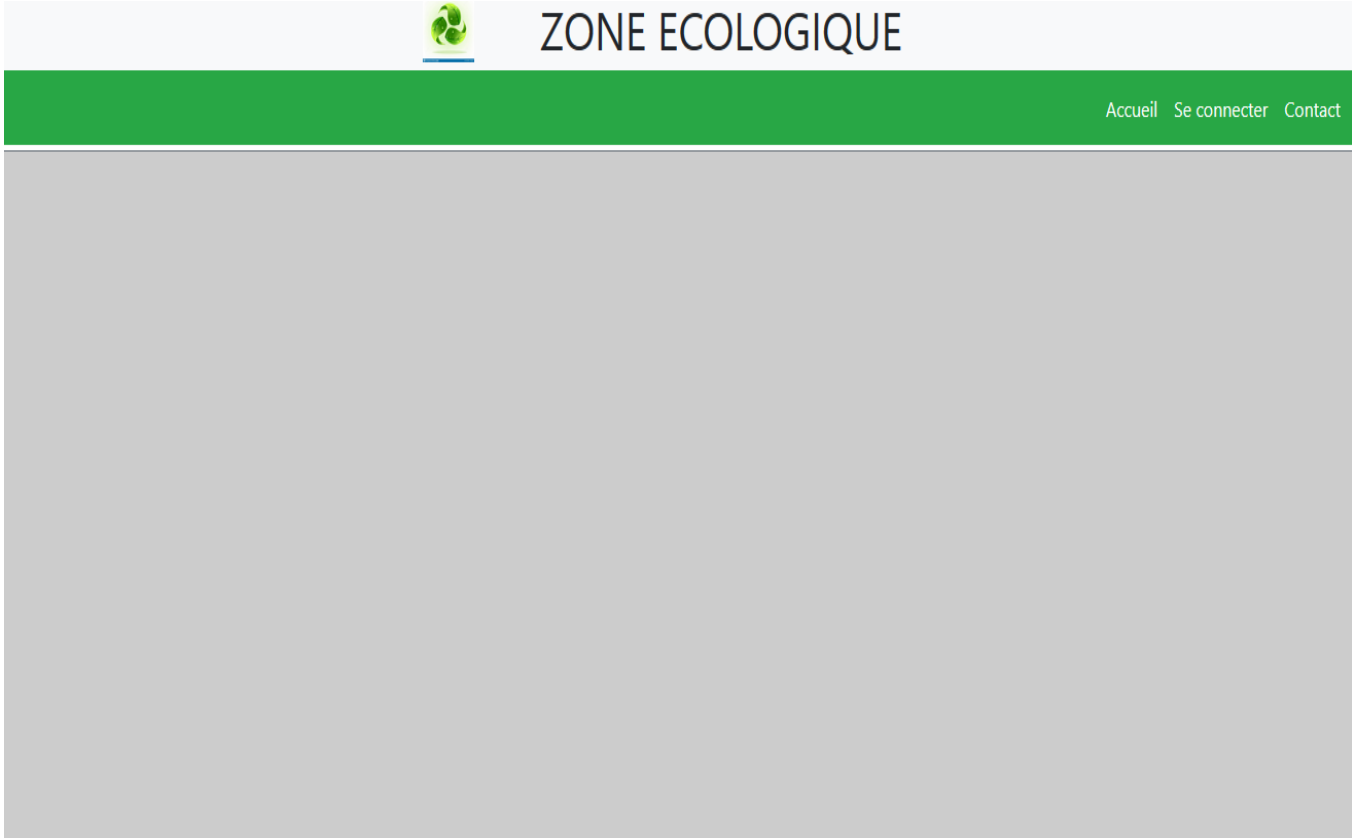

Figure 3.9 : Page d’accueil du site

La page d’accueil reste vide pour que le propriétaire du site puisse choisir son style d’apparence envers les visiteurs.

b. Page de connexion et d’inscription

La page de connexion est faite pour que les utilisateurs puissent s’authentifier. C’est-à-dire tous les utilisateurs doivent avoir un compte pour raison de sécurité. Cette page demande l’adresse e-mail et le mot de passe de l’utilisateur. Dans le cas où l’utilisateur n’a pas encore inscrit, la page de connexion comporte un lien d’inscription. Il y a aussi un lien pour tous ce qui oubli son mot de passe. Elle est illustrée à la Fig.3.10.

Figure 3.10 : Page de connexion

La page d'inscription est pour le nouvel utilisateur. Toutes les informations doivent remplir (Fig.3.11).

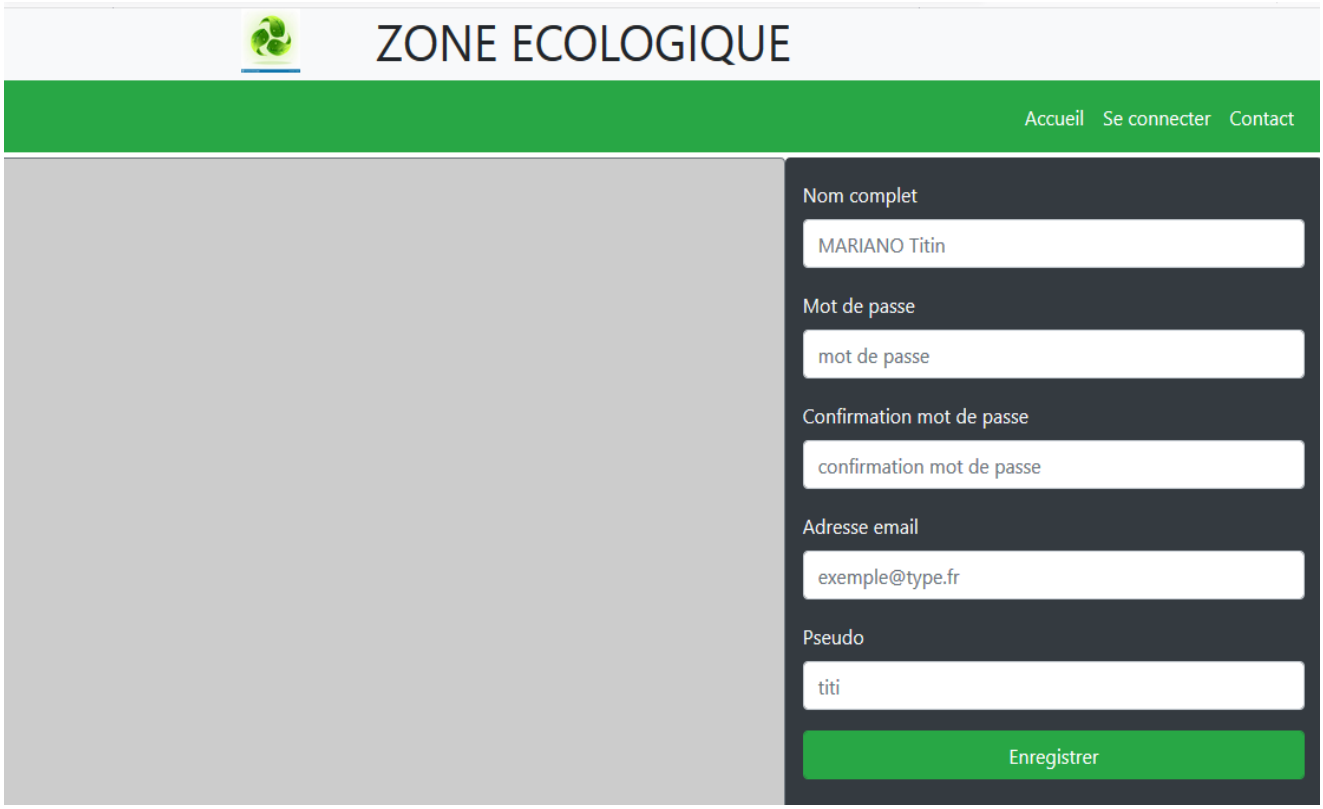

Figure 3.11 : Page d'inscription

Rappelons que deux types d'utilisateurs existent : soit un administrateur, soit un utilisateur simple.

L'administrateur a le droit de manipuler les informations contenues dans la base (insertion, modification, suppression). Il peut choisir les informations autorisées pour les utilisateurs simples. Le paramétrage de l'apparence du site dépend de son gout (couleur de barre de menu, couleur de barre de navigation, logo du site).

Pour un utilisateur simple, il bénéficie les données autorisées par l'administrateur. Les données sont affichées sous forme de table. Pour faciliter l'accès aux données pour l'utilisateur, il y a une barre de recherche et de trier dans son interface.

c. Partie administrateur du site

Après la connexion si l'utilisateur est un administrateur, il est dirigé vers la page à la Fig.3.12 suivante.

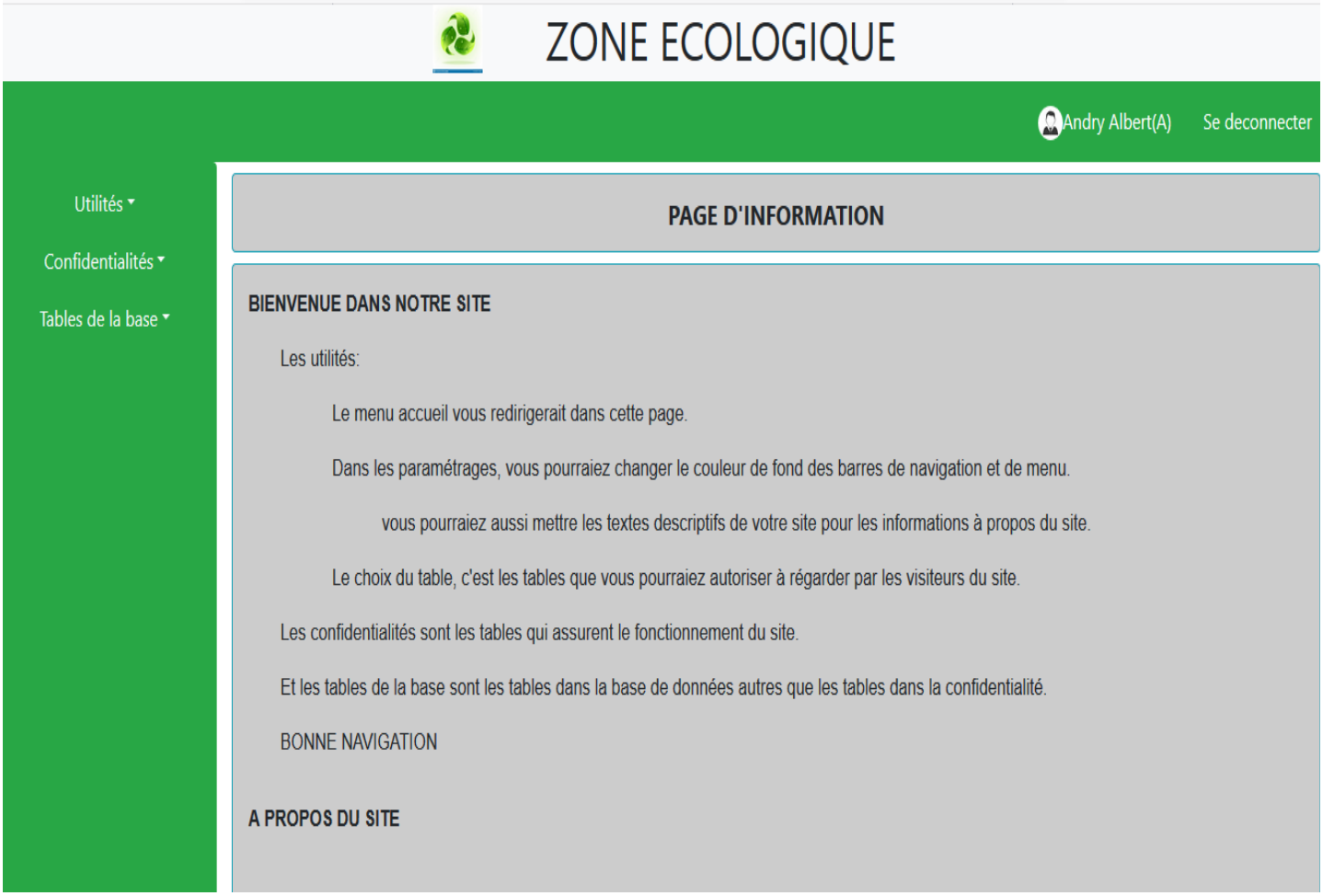

Figure 3.12 : Page d'accueil pour un administrateur

Dans cette page, les menus qui scrollent à gauche contiennent les utilités, les confidentialités, et les tables de la base. « *L'utilité* » contient trois sous menus qui sont l'accueil, le paramétrage et le choix table. L'accueil affiche des instructions à l'utilisation du site comme dans la Fig.3.12. Le paramétrage sert pour paramétrer le couleur de barre de navigation, de barre de menu, le logo du site et pour insérer les informations concernant le site (Fig.3.13). Le choix table est une page qui permet à l'administrateur de définir les tables autorisées aux utilisateurs simples (Fig.3.14).

Le menu « *confidentialités* » porte trois sous menus. Ces sous menus sont les noms de table créés par le projet dans la base de données utilisée. Ces tables contiennent les informations de paramétrage au fonctionnement du site et les identités de chaque utilisateur.

Le menu « *tables de la base* » comporte tous les noms des tables qui existent dans la base.

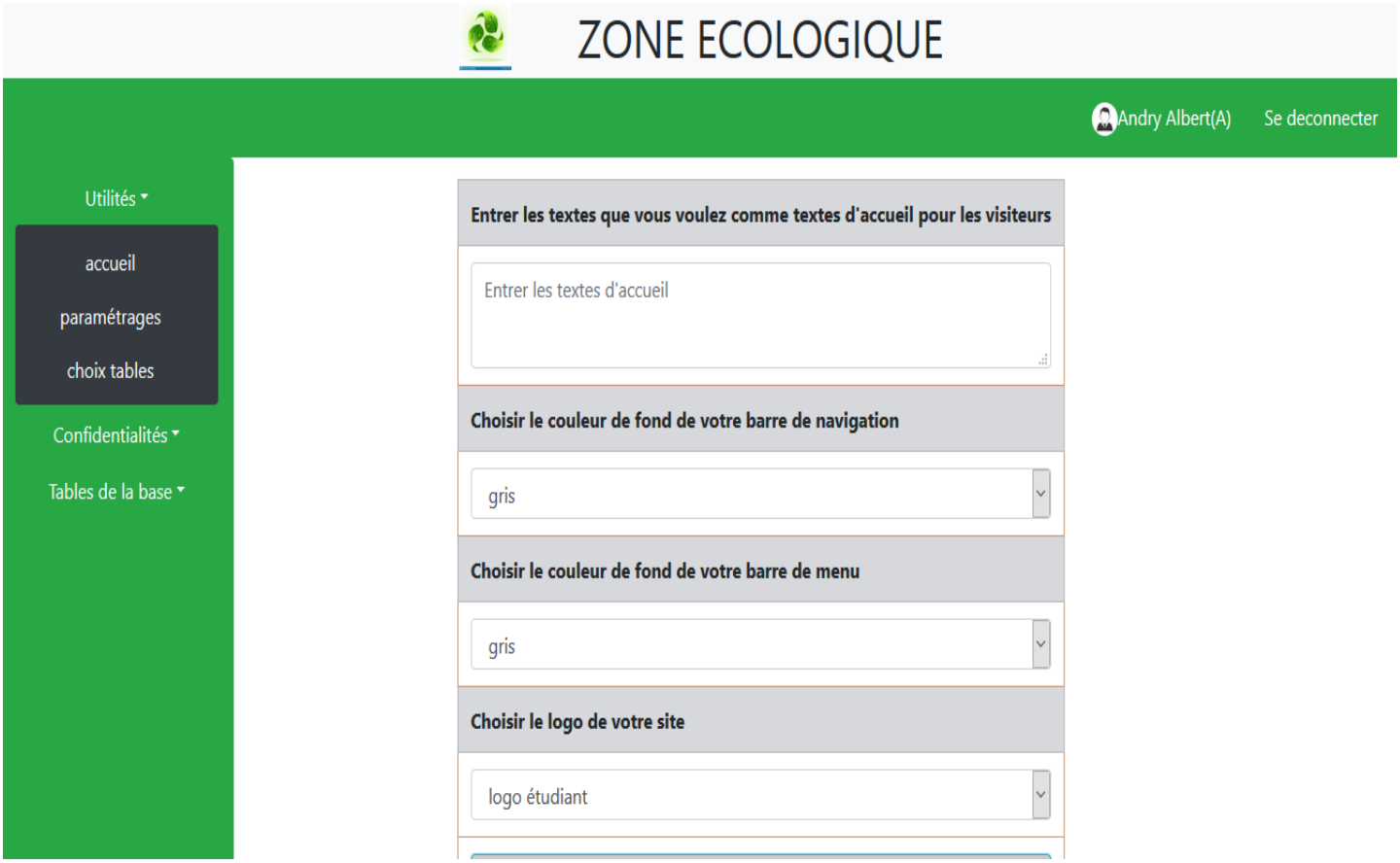

Figure 3.13 : Page de paramétrage

Ici le couleur de menu choisi est gris. Le couleur de barre de navigation est gris. Le logo est le logo étudiant. Après soumission de ce formulaire, l'apparence du site va se changer. Ce changement est constaté dans la fig.3.14 qui suit.

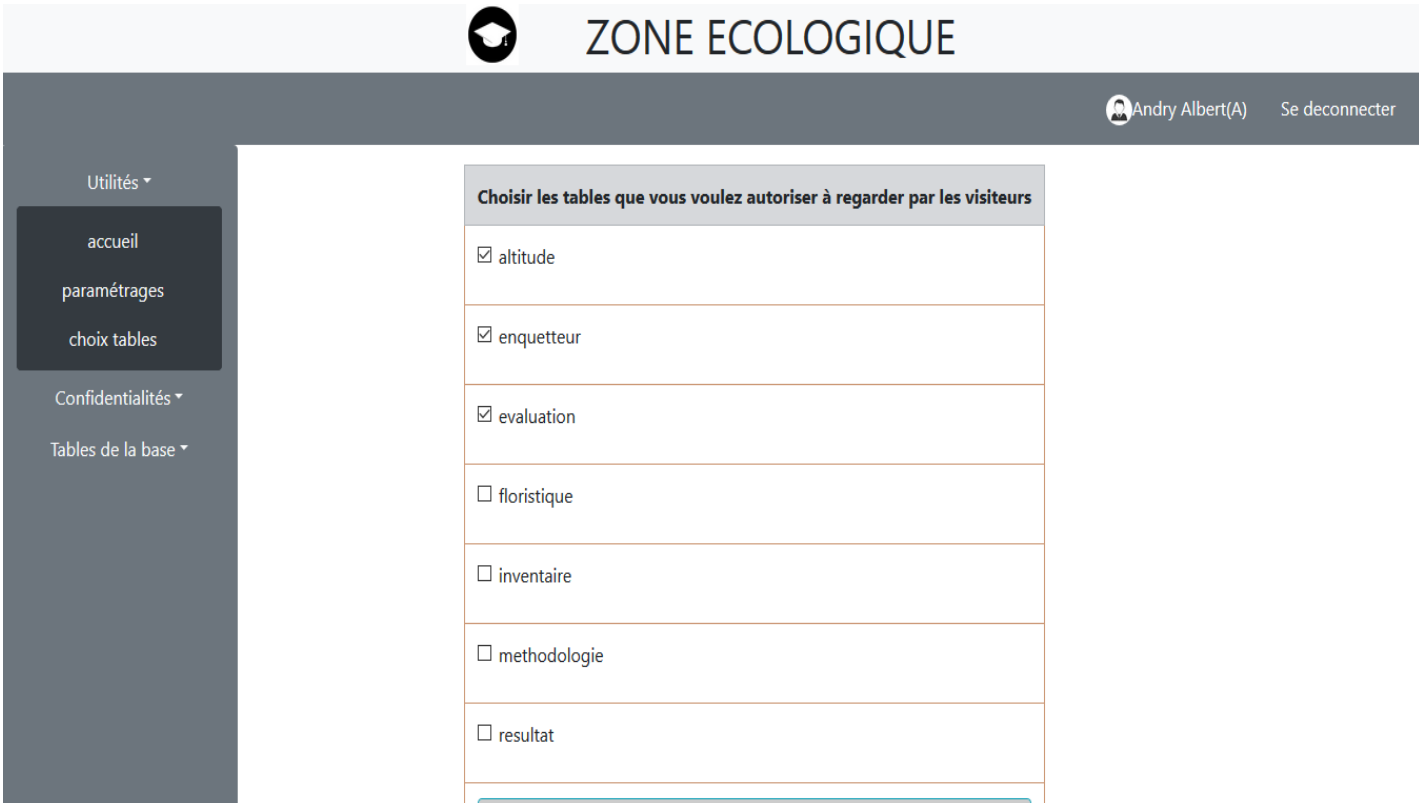

Figure 3.14 : Page de choix de table

Dans cette page de choix de tables, l'administrateur sélectionne trois tables donc ces trois tables vont afficher aux utilisateurs simples. Pour l'affichage des données, la table qui les contient se ressemble comme dans la Fig.3.15.

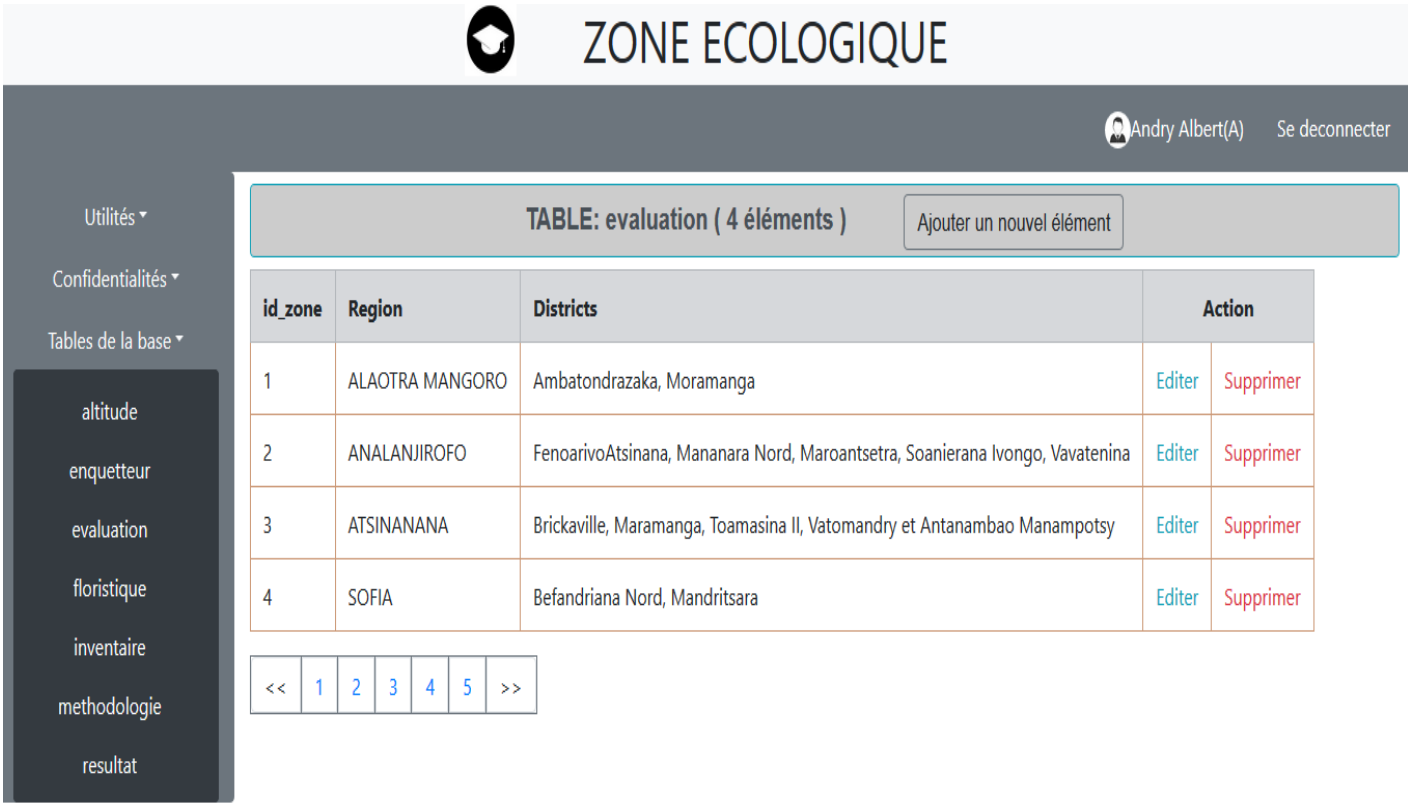

Figure 3.15 : Exemple d'affichage d'une table

La table « évaluation » est choisie pour représenter la contenue de la page qui présente les données de la base. Cette page comporte le nom de table sélectionnée avec les nombres des éléments dont elle contient. Le bouton « ajout un nouvel élément » sert pour insérer un nouvel tuple de données. Sur la colonne action, l'administrateur a le droit de modifier ou supprimer chaque ligne de données. Chaque page peut porter douze lignes de données. Or si les nombres des éléments de table dépassent cette limite, on peut servir du lien de pagination.

e. Partie utilisateur simple du site

Après avoir insérer un compte utilisateur simple, l'utilisateur se redirige vers la page d'accueil d'un utilisateur simple représentée dans la Fig.3.16.

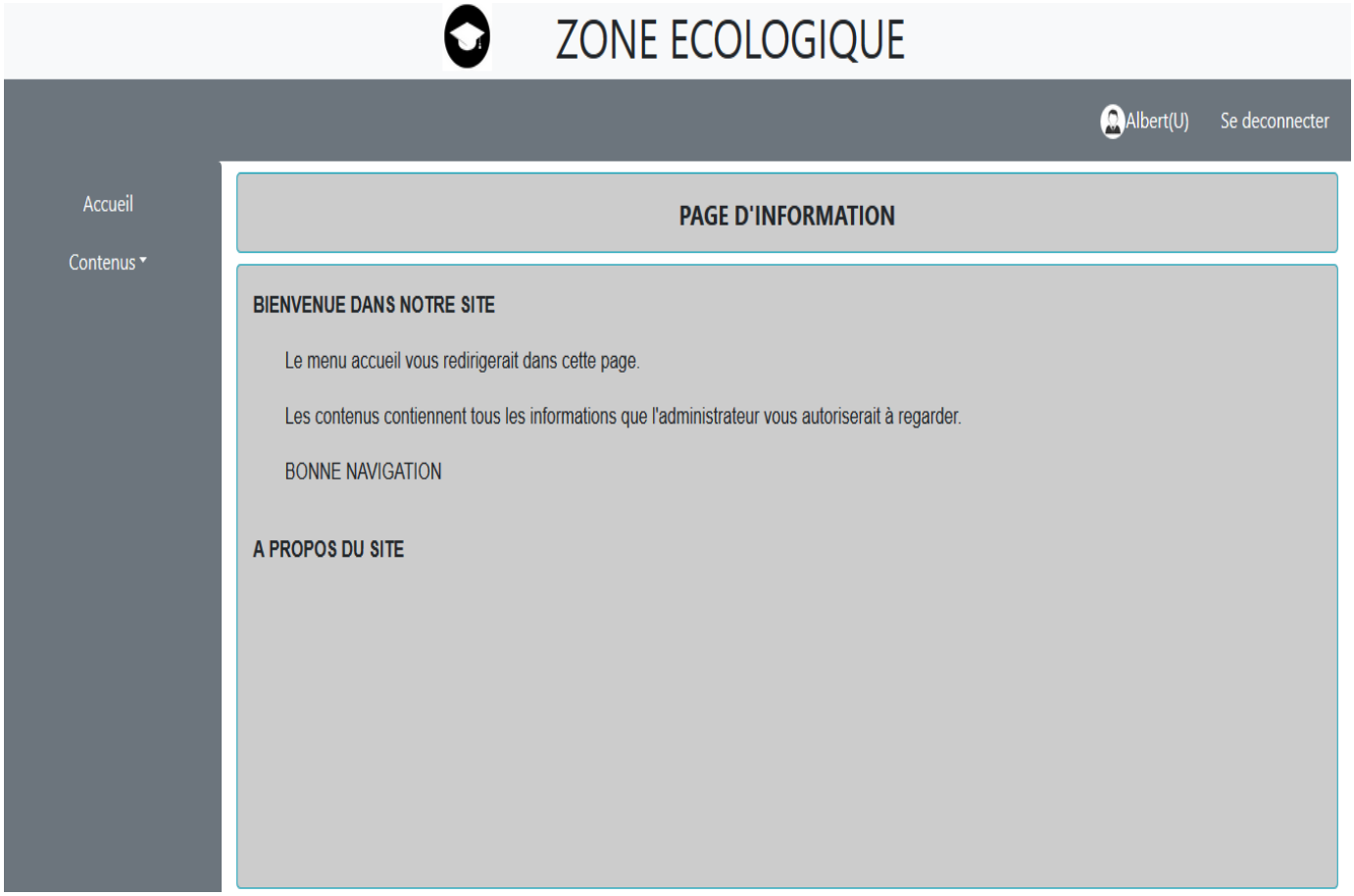

Figure 3.16 : Page d'accueil pour un utilisateur simple

Cette page comporte dans la partie gauche une barre de menu. Le menu « accueil » qui redirige vers la page actuelle. Le menu « contenus » contient comme sous menu les noms de table que l'administrateur autorise à l'utilisateur simple.

Dans le corps du page, la partie à propos du site dépend à l'information insérer par l'administrateur dans la page de paramétrage. Et dans la partie utilisateur simple du site les données se présentent comme dans la Fig.3.17 qui suit.

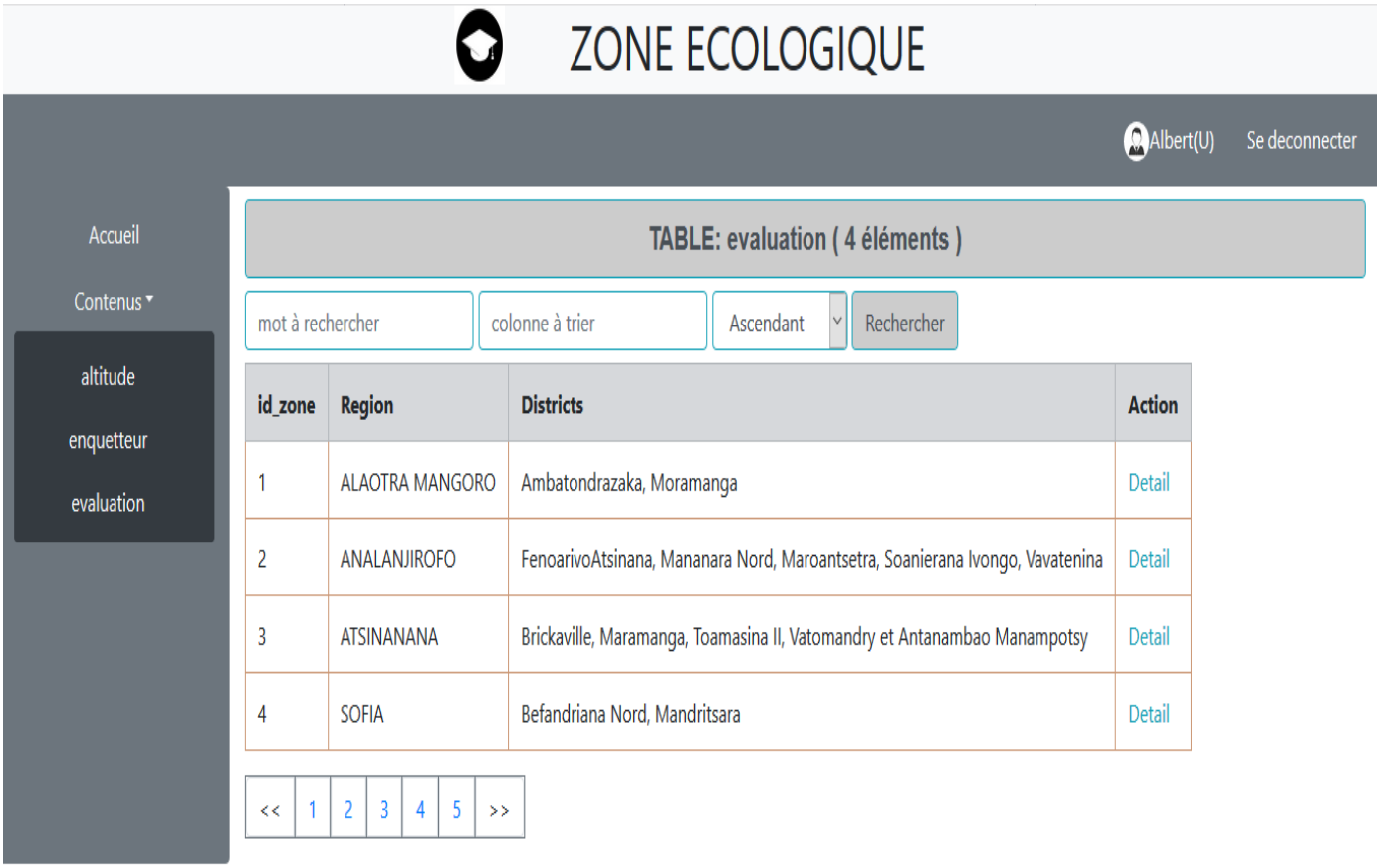

Figure 3.17 : Exemple d'affichage des données dans la partie utilisateur simple

Lors de paramétrage dans la partie administrateur, il sélectionne trois tables donc dans la partie utilisateur simple, ces trois tables sont autorisées. Dans cette page, le nom de la table et les nombres des éléments qu'elle contient, sont affichés. La page contient aussi une barre de recherche et de trier. La colonne « action » dans la table sert pour détailler les informations dans une ligne qui ne sont pas afficher dans la page. Et en bas de page, il y a toujours de lien pagination. Si les éléments dans une page est supérieur à douze alors la page suivante de la pagination sera fonctionnelle. Sinon une page sans informations sera affichée.

La figure 3.18 montre un exemple d'une ligne d'information détaillée. La clé primaire numéro 1 de la table « altitude » est choisie ici pour l'exemple. On trouve huit colonnes dans cette table or cinq colonnes sont le maximum à afficher dans chaque page autre que la colonne « action ». L'utilisateur simple n'a le droit ni de modifier ni de supprimer ces données. Il peut

revenir à la table qui contient ces données en appuyant sur le lien « revenir à la table » suivi du nom de cette table.

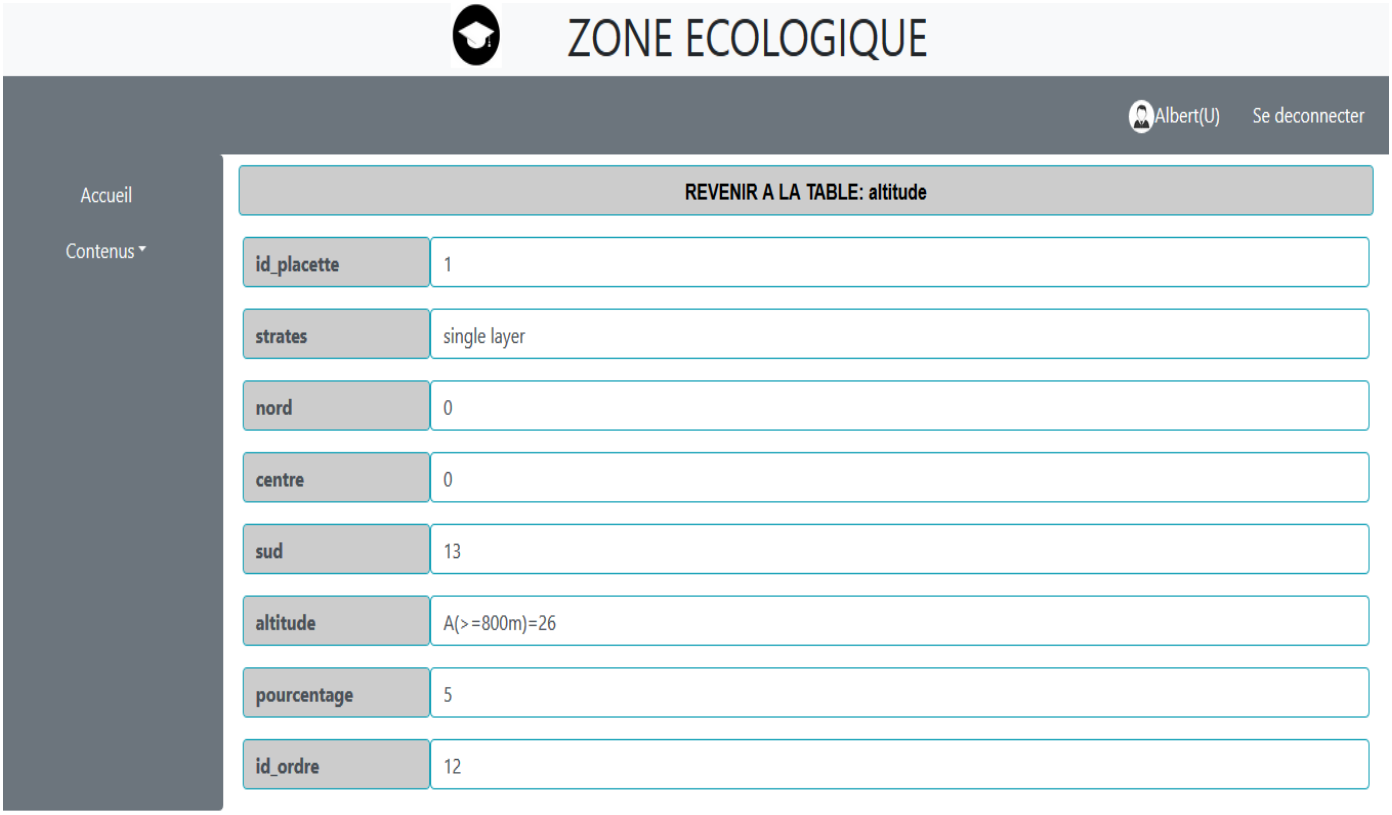

Figure 3.18 : Exemple d'une ligne d'information détaillée

Un simple utilisateur peut détailler chaque ligne de données pour avoir plus d'information. Mais il peut rechercher aussi le mot voulu. La figure 3.19 montre un exemple de résultat de recherche.

Figure 3.19 : Résultat de recherche

Un simple utilisateur fait une recherche, considérons comme exemple qu'il a recherché dans la table « altitude » le mot « single ». Le résultat de recherche affiche toutes les lignes des données qui contiennent le mot à rechercher. Il affirme aussi le nombre des éléments trouvés. En plus, cet utilisateur peut trier chaque colonne de données, c'est-à-dire les informations dans une colonne peuvent ordonnées par ordre croissante ou ordre décroissante. La recherche et le tri des données peuvent fonctionner simultanément. Prenons l'exemple précédant, la recherche du mot single est faite avec le tri de données dans la colonne « sud ». Ce dernier est trié par ordre décroissant. Alors le résultat de recherche et de tri s'affiche en même temps (Fig.3.19). Mais un utilisateur peut faire ce tri indépendamment de la recherche.

3.4. Conclusion

Ce chapitre montre en détail la conception et la réalisation d'un site web utilisé pour manipuler une base de données. La méthodologie de modélisation UML est adoptée alors plusieurs diagrammes est réalisés pour clarifier le mode d'activité de projet. Des différentes interfaces sont aussi exposées pour expliquer le fonctionnement du site. Deux types d'utilisateur existent, un administrateur et un simple utilisateur. L'administrateur a le droit de manipuler toutes les données et de faire les paramétrages nécessaires. Un simple utilisateur peut consulter les données qu'on lui a octroyées.

CONCLUSION

Le travail a permis de créer automatiquement le code source d'un site web pour faciliter la gestion des informations stockées dans une base de données relationnelle. Des étapes ont été suivies pour aboutir à la finalisation de ce projet.

L'étude de la base de données est effectuée pour pouvoir l'utiliser. Le système de gestion de base de données est un logiciel moteur qui manipule la base de données et dirige l'accès à son contenu. Toutes les applicatives informatiques qui utilisent des données sont inséparables à la gestion de ses données. La base de données est donc un conteneur informatique permettant de stocker l'intégralité des informations en rapport avec une activité.

La notion de programmation orientée objet est aussi étudiée. La programmation est un art pour mettre en relation la pensée humaine avec un matériel informatisé. Le langage PHP avec une architecture MVC est choisi car il est apte et fiable pour la réalisation de ce travail. Ce langage est facile à apprendre et à manipuler sans tenir compte le typage d'un objet qui est un moyen utilisé par les programmeurs pour mettre une fonction interne de code semblable à un scénario réel.

Avant la réalisation de ce projet, une conception est faite en adoptant la méthodologie de modélisation UML. Cette méthodologie englobe les études et les analyses du projet pour assurer sa qualité et son bon fonctionnement.

Le site web est fonctionnel et protégé par une authentification. Deux types d'utilisateur existent, l'administrateur qui a le droit de manipuler toutes les données et de faire les paramétrages, un simple utilisateur qui peut consulter les données qu'on lui a octroyées.

Ce projet de fin d'étude m'a permis de promouvoir les connaissances sur le domaine des applications web. La conception et la réalisation m'ont permis aussi de faire un code propre et compréhensible. Ce travail n'est que commencé, plusieurs tâches peuvent être encore y ajouter.

ANNEXE A : INSTALLATION Apache/PHP/MySQL

WAMP est un acronyme désignant un ensemble de logiciel libre permettant de construire un serveur de site web, il signifie : Windows Apache MySQL PHP.Il existe aussi d'autre architecture : LAMP pour un système Linux, MAMP pour Mac et XAMP pour toutes les plates-formes.
Mais les informations qui suivent sont des installations pour les trois serveurs : Apache, MySQL et PHP, dans un système Windows sans utiliser le plateforme WAMP.

A.1. Installation du serveur Apache

L'installation des fichiers se fait à partir d'une distribution binaire (fichier .msi) que vous pouvez trouver dans : http://http.apache.org/dist/httpd/binaries/ win32/. Voici les étapes à suivre : faites un double clic sur le fichier "apache.x.x.x.msi" pour lancer l'installation. Pensez à lire le "read this first" qui contient un tas d'information utile. Renseignez les champs qui sont demandés : le « network domain » qui s'agit du domaine auquel la machine va être rattachée, le « server name » pour le champ du nom complet de la machine ou son adresse IP, « administrator's email address » indiquez votre email (ou celui du responsable du serveur) et enfin choisissez la méthode d'installation en fonction de votre cas. Après un lancement, le répertoire de destination (c:\soft) est créé automatiquement.
Si l'installation est terminée, redémarrez la machine. Après le redémarrage, lancez le serveur à la main dans le menu démarrer. Une console apparait et affiche "Apache Running". Vous devez la laisser ouverte pendant tout le temps où vous utilisez le serveur. Lancez un navigateur et tapez l'adresse http://localhost/.Une page apparait avec quelques informations sur apache : le serveur fonctionne. Pour arrêter le serveur, fermez la console ou cliquez dedans et appuyez sur <ctrl-c>. Le fichier de configuration d'Apache est un simple fichier texte qui peut être édité avec n'importe quel éditeur de texte. Il se nomme "httpd.conf" et se trouve dans le répertoire *"*C:\soft\apache \conf*"*.

A.2. Installation du MySQL

A.2.1. Installation des fichiers

Ouvrez le fichier mysql-x.x.x-win.zip et double-cliquez sur setup.exe (répondez ok quand winzip vous dit qu'il va décompresser les fichiers dans un répertoire temporaire).

Installez avec les options par défaut, choisissez la méthode d'installation "typical". (Changez seulement le répertoire d'installation par défaut en c:\soft\mysql).

A.2.2. Configuration de l'environnement Windows pour MySQL

Il faut éditer les paramètres DOS (Disk Operating System) par défaut de la machine (autoexec.bat) pour une utilisation plus confortable. Avec l'explorateur de fichier, allez sur le disque C, cliquez avec le bouton droit sur le fichier autoexec.bat et choisissez "edit".A la fin du fichier, ajoutez la ligne de code suivant : set path=%path%; C:\soft\mysql\bin. Sauvez le fichier et redémarrez Windows.

Maintenant, les commandes MySQL sont dans le chemin par défaut. On peut les utiliser sous DOS quel que soit le répertoire où on se trouve. Pour lancer, on ouvre une console et tapez mysqld. Le serveur est démarré et s'exécute en tâche de fond. Vous pouvez maintenant utiliser le serveur. Pour l'instant, il n'y qu'un seul utilisateur qui est le racine (root), et n'a pas de mot de passe, mais il a tous les droits existant dans le serveur.

A.3. Installation d'un PHP

PHP est un langage de programmation script côté serveur permettant de produire de page web dynamique. Sous Windows, Apache exécute un script PHP en appelant l'interpréteur php.exe. L'essentiel de l'installation consiste donc à paramétrer le fichier de configuration Apache pour lui indiquer où se trouvent ce programme et quel fichier doit lui être transmis. Dans le répertoire Apache/conf, éditez le fichier httpd.conf et on suit la procédure suivante : cherchez la ligne avec « ServerAdmin » et indiquez votre email. Cherchez la ligne avec « ServerName » et indiquez le nom de l'ordinateur (ou localhost). Cherchez une ligne avec « ScriptAlias » et ajoutez par la ligne de code suivante pour indiquer où se trouve l'interpréteur PHP : ScriptAlias /php7/ "C:/php7/".

Maintenant il faut indiquer à Apache l'extension des fichiers PHP. Cherchez la section avec le terme « AddType » et ajoutez : « AddType application/x-httpd-php.phtml.php puis AddType application/x-httpd-php-source.phps ». Finalement il faut associer les scripts PHP à l'interpréteur, en ajoutant la ligne suivante :« Action application/x-httpd-php /php4/php.exe ».

Nous allons maintenant passer au test pour voir si la configuration du serveur PHP fonctionne correctement. Pour cela, lancer un bloc-note et taper cette ligne de code :< ? phpphpinfo(); ?> puis sauvegarder avec le nomphpinfo.php dans le répertoire de base du serveur Apache. Ce code PHP va nous permettre d'afficher les options de configuration du serveur PHP. Démarrer

votre serveur Web et essayez d'accéder à l'URL localhost/phpinfo.php : la page d'accueil de PHP doit s'afficher. Et c'est terminé : l'API MySQL est incluse d'office dans l'interpréteur PHP, ce qui évite toute manipulation supplémentaire.

ANNEXE B : EDITEURDE TEXTE PHPSTORM

Les éditeurs de texte sont des programmes dédiés à l'écriture de code. On peut en général les utiliser pour de multiples langages : HTML, CSS, PHP, JavaScripts, etc. Ils se révèlent être de puissants alliés pour les créateurs de site web. On peut tout à fait créer un site web uniquement avec Bloc-notes qui s'agit d'un logiciel d'édition de texte intégré par défaut à Windows. Mais il existe énormément d'éditeur de texte aux fonctionnalités plus ou moins équivalentes. Pour notre choix, on va décrire comment installer un éditeur PhpStorm. Le premier pas à faire est de visiter le lien de téléchargement et suivre les instructions des schémas.

On tombe sur le site officiel de jetbrains et il suffit de choisir le lien « download » (Fig.A.1).

Figure A.1:Page d'accueil du site officiel jetbrains

En cliquant sur le bouton de téléchargement, le chargement des fichiers est lancé automatique. La figure A.2, figure A.3, figure A.4 et la figure A.5 les étapes d'installation à suivre.

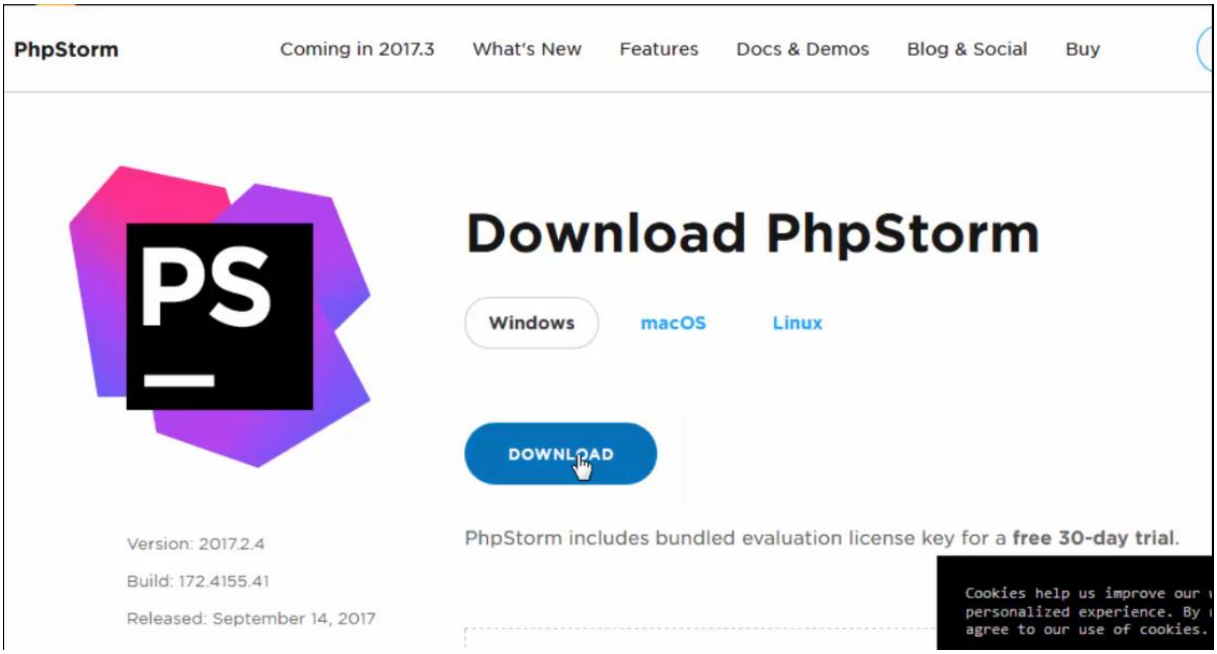

Figure A.2:Téléchargement de PhpStorm

Quand les fichiers sont tous chargés, faites un double clic sur le fichier jetbrains.exe et l'installation se lancée.

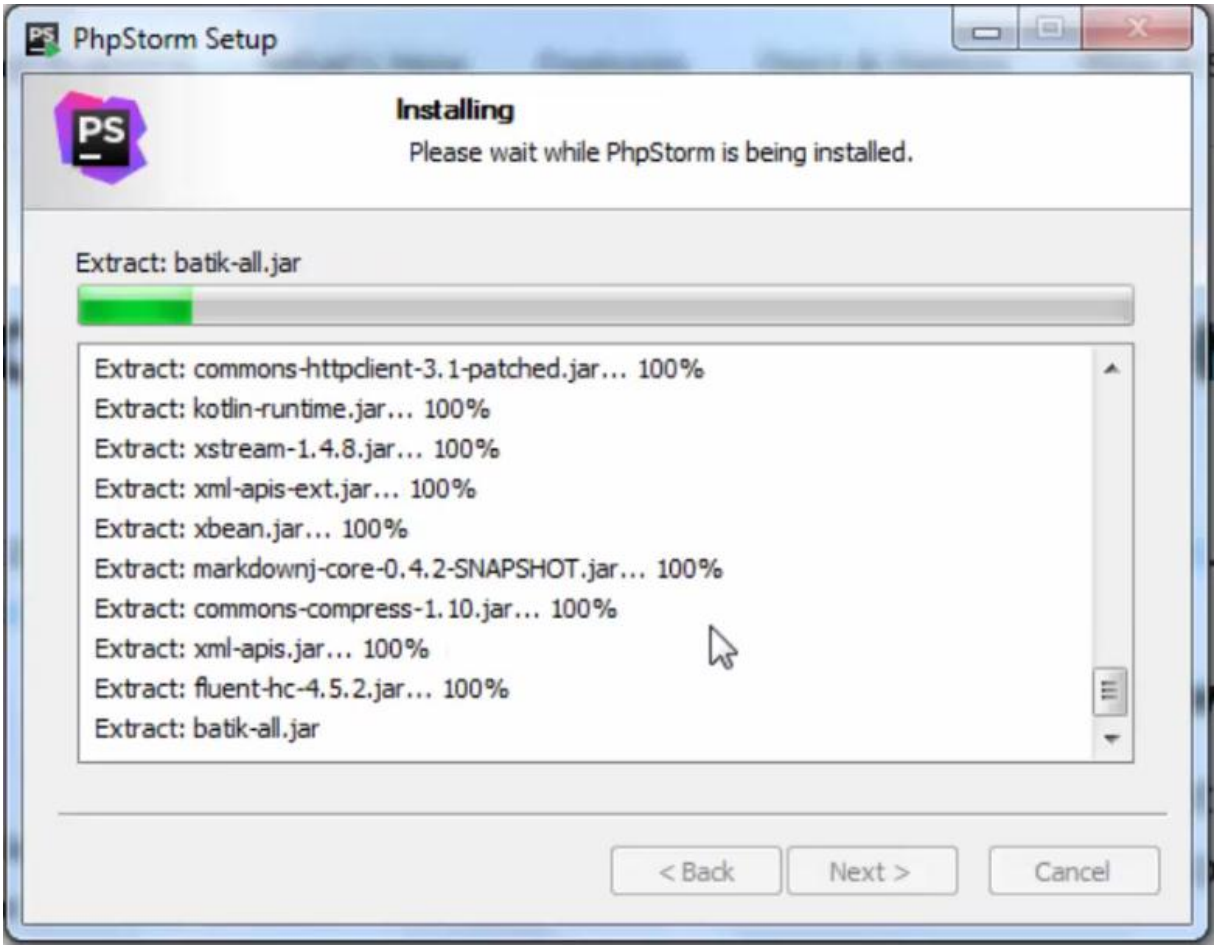

Figure A.3: Installation de PhpStorm

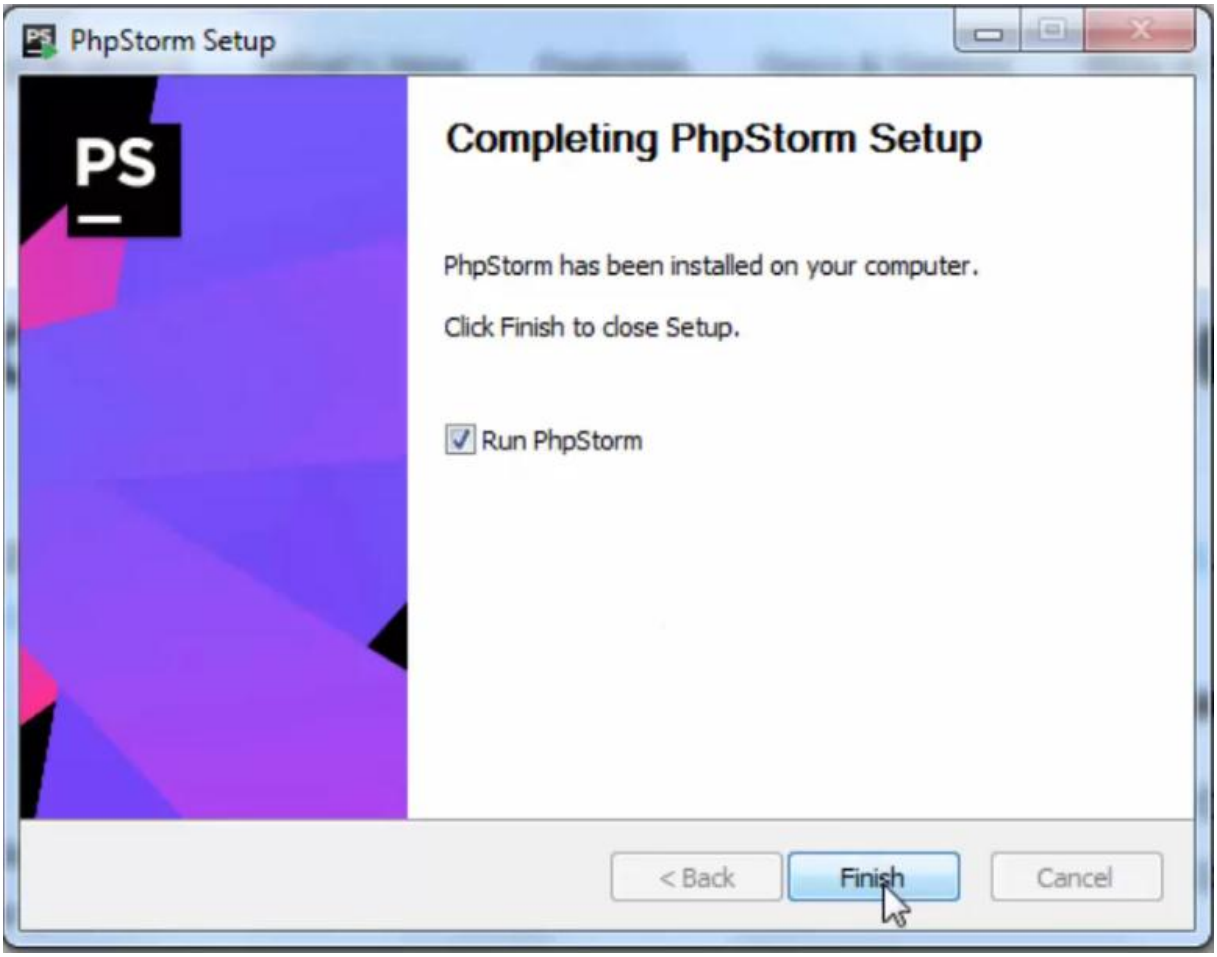

Figure A.4: Marque de fin d'installation

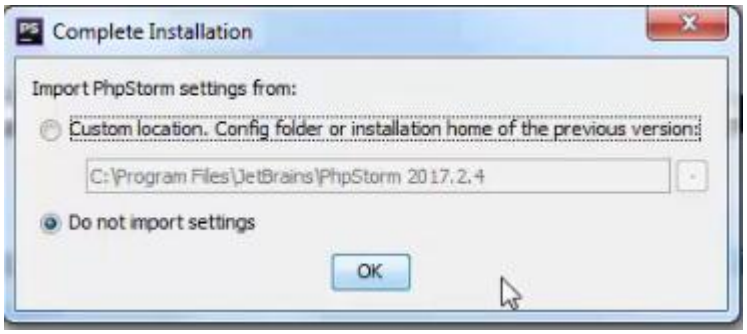

Figure A.5: Chargement de fichier de dépendance

Les fichiers de dépendances sont très utilisables pour que l'éditeur peut faire une autocorrection des codes et aussi pour qu'il puisse faciliter les tâches.

A propos de licence d'utilisation, il faut choisir le mode gratuit si on travaille en local, comme la figure A.6 l'indique.

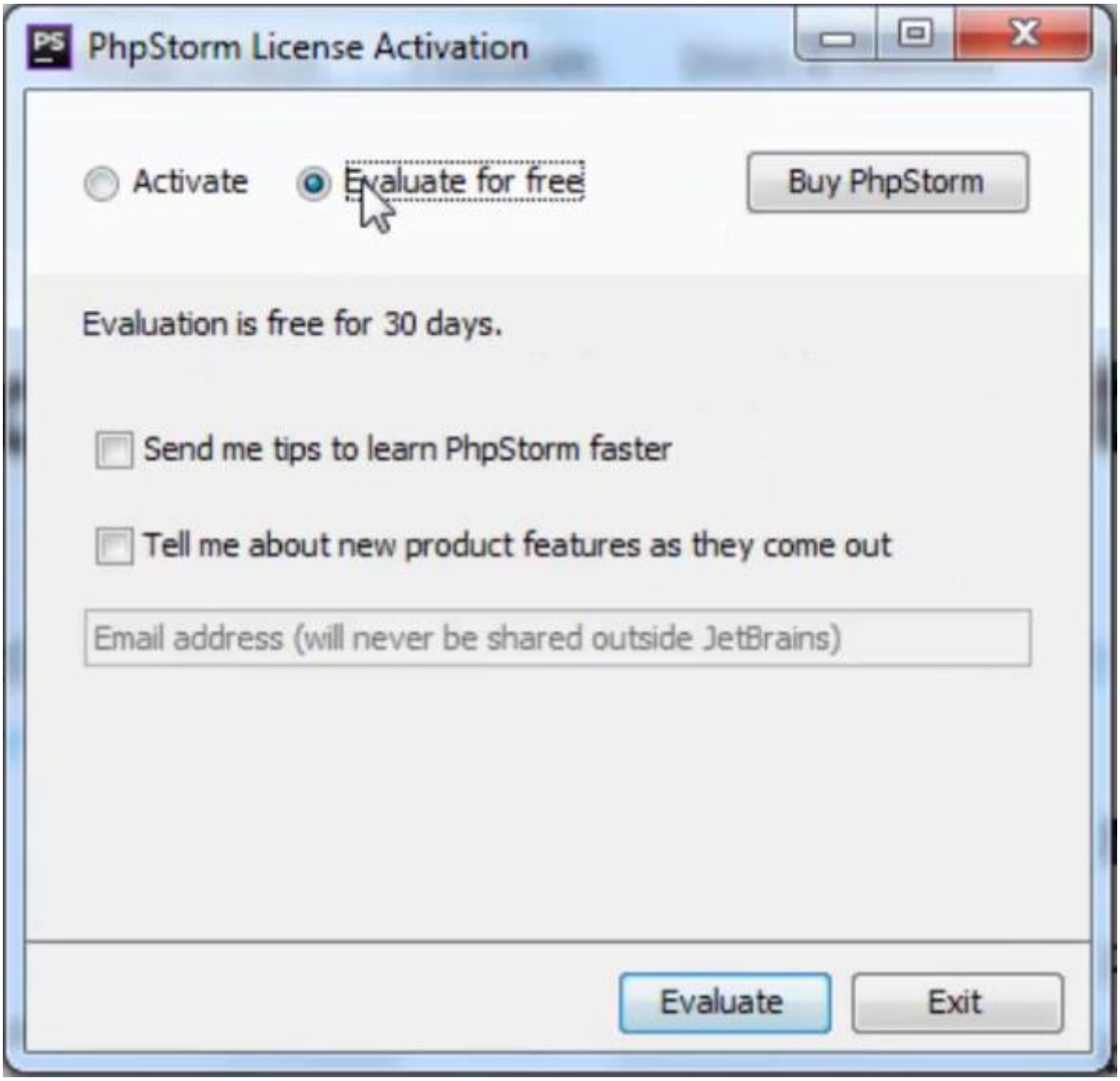

Figure A.6:Licence d'utilisation

Maintenant, notre éditeur de texte est prêt à utiliser.

Remarque :

Il n'est pas obligatoire d'utiliser cet éditeur de texte car il y existe tant d'éditeurs qui peuvent utiliser en développement informatique.

ANNEXE C : NAVIGATEUR

Le navigateur est le programme qui nous permet de voir les sites web. Son travail est de lire le code pour afficher un résultat visuel à l'écran. Le rôle du navigateur est donc essentiel, mais c'est un programme extrêmement complexe. En effet, comprendre le code n'est pas une mince affaire. Le principal problème, c'est que les différents navigateurs n'affichent pas un même site exactement de la même façon. Il faudra donc faire et prendre l'habitude de vérifier régulièrement que le site fonctionne correctement sur la plupart des navigateurs. Il existe de nombreux navigateurs et voici les principaux à connaître (Fig.A.7).

Navigateur	OS	Téléchargement	Commentaires
Google Chrome  Google Chrome	Windows Mac Linux	Téléchargement	Le navigateur de Google, simple d'emploi et très rapide. C'est le navigateur que j'utilise au quotidien.
Mozilla Firefox  Logo Firefox	Windows Mac Linux	Téléchargement	Le navigateur de la fondation Mozilla, célèbre et réputé. Je l'utilise fréquemment pour tester mes sites web.
Internet Explorer  IE9	Windows	Téléchargement (Déjà installé sur Windows)	Le navigateur de Microsoft, qui équipe tous les PC Windows. Je l'utilise fréquemment pour tester mes sites web.
Safari  Logo Safari	Windows Mac	Téléchargement (Déjà installé sur Mac OS X)	Le navigateur d'Apple, qui équipe tous les Mac.
Opera  Logo Opera	Windows Mac Linux	Téléchargement	L'éternel *outsider*. Il est moins utilisé mais propose de nombreuses fonctionnalités.

Figure A.7: Quelques exemples de navigateur

Chaque navigateur possède sa propre spécificité. Si on développe un site internet, il faut que le rendu soit le même dans tous les divers navigateurs pour assurer l'image de marque du site.

REFERENCES

[01] LAURENT Audibert ; Introduction au Système de Gestion de Base de Données et aux Bases de Données ; version 1 ; Département informatique de l'institut universitaire de technologie de Villetaneuse ; 17 juin 2008

[02] MICHEL Cartereau ; Introduction aux bases de données ; AgroParisTech - UFR d'informatique - 16, rue Claude Bernard - F 75231 PARIS CEDEX 05 ; Octobre 2014

[03] OLIVIER Losson ; Introduction aux Systèmes de Gestions des Base de Données Relationnelles ; Université de Lille-Sciences et Technologie, Master ASE et GI ; 27 Juin 2008

[04] PIERRE Edouard ; Concepts et langages des Bases de Données Relationnelles ; Support de cours SGBD1 ;IUT de Nice-Département Informatique ; Juin 2003

[05] E833SGB ; Système de Gestion de Base de Données ; Semestre 8 ; Mention Electronique ; ESPA ; Année Universitaire 2016-2017

[06] E661INFIA ; Programmation en langage orientée objet ; Semestre 6 ; Mention Electronique ; ESPA ; Année Universitaire 2015-2016

[07] MAILLOT Thibault; INF340 Systèmes d'informations-Outils et méthodes informatiques pour le multimédia ; HAL Id : cel-01830944 ; 27 Juillet 2018

[08] ALEXANDRE Bacco (Winzou) ; Développez efficacement votre site web avec le Framework symfony2 ; Le Site du Zéro ; 8 Janvier 2013

[09] http://www.programmez-en orientée objet en PHP- site OpenClassrooms ; Cours ; 11 Mai 2019

[10] http://www.adoptez une architecture MVC en PHP- site OpenClassrooms ; Cours ; 15 Mai 2019

[11] E733GLO ; Génie logiciel ; Semestre7 ; Mention Electronique ; ESPA ; Année Universitaire 2016-2017

[12] ZAIRI Mohamed Ali ; Conception et réalisation d'un intranet ; Mémoire de Projet de Fin d'Études pour l'Obtention du Diplôme de Mastère professionnel en Nouvelles Technologies des Télécommunications et Réseaux ; Année Universitaire 2010-2011

[13] CABIANCA Jean-Claude; Modélisation PHP Orientée Objet pour les Projets Modèle MVCMini Framework ; BTS IRIS, 25 Octobre 2006

[14] E971GLOIA ; Génie logiciel II ; Semestre 9 ; Mention Electronique ; ESPA ; Année Universitaire 2017-2018

[15] GABAY Joseph; Analyse et conception ; UML2 Algeria-Educ.com, Décembre 2010

Titre : « Création automatique d'un site web à partir d'une base de données relationnelle»

Auteur : ANDRIANJARASOA Mandrindra Nantenaina

Nombre de pages : 80
Nombre de figures : 41
Nombre de tableaux : 1

RESUME

Ce travail consiste à créer un site web qui permet de manipuler une base de données relationnelle. Ce site est créé automatiquement à l'aide d'un outil qui collecte les informations sur la configuration de la base de données utilisée. Le site web obtenu possède plusieurs interfaces pour consulter et modifier les informations stockées dans la base de données. Il est protégé et nécessite une authentification pour y accéder. Deux types d'utilisateur existent, un administrateur et un simple utilisateur. L'administrateur a le droit de manipuler toutes les données venant de la base et de faire les paramétrages nécessaires. Un simple utilisateur peut consulter les données qu'on lui a octroyées. Le langage PHP avec l'architecture MVC et le serveur de base de données MySQL ont été utilisés pour réaliser le projet.

Mots clés : Création automatique d'un site web; Base de données relationnelle; PHP ; MySQL

DIRECTEUR DE MEMOIRE : Monsieur RAMASOMBOHITRA Nivonjy Nomen'Ahy

Contact :
mandrindr@gmail.com (+26134 33 215 84)