Documentation Technique WEB

Objectif du document

Ce document est un dossier d’architecture technique détaillé pour les développements d’applications Web. Il s’adresse :

* Aux chargé(s) de projet : Etudiants de 2TSSIOD
* Aux architectes du projet : Etudiants de 2TSSIOD sur la FERME (aidés de leurs collègues de 2TSSIOR)
* À l’équipe de mise en production : Etudiants de 2TSSIOD

Historique du document

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Rédacteur** | **Description** |
| V 1.0 | 01/09/2016 | F. de Robien |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Diffusion

|  |  |  |  |
| --- | --- | --- | --- |
| **Destinataires** | | **Pour**  **validation** | **Pour**  **information** |
| Etudiants de 2 TSSIOD |  |  | X |
|  |  |  |  |
|  |  |  |  |

Validation

|  |  |  |
| --- | --- | --- |
| **Représentants** | **Valideur** | **Date** |
|  |  |  |
|  |  |  |

Sommaire

[1. Préambule 3](#_Toc454207331)

[1.1 Lexique 3](#_Toc454207332)

[1.2 Signalétique 3](#_Toc454207333)

[2. Architecture Technique Générale 3](#_Toc454207334)

[2.1 Schéma global d’architecture 4](#_Toc454207335)

[2.2 Plateforme technique 4](#_Toc454207336)

[3. Description des Données 5](#_Toc454207337)

[3.1 Modèle conceptuel (Préciser la Version) 5](#_Toc454207338)

[3.2 Modèle logique (Préciser la Version) 5](#_Toc454207339)

[3.3 Modèle physique (Préciser la Version) 5](#_Toc454207340)

[4. Description du Code 5](#_Toc454207341)

[4.1 Historique de la solution 5](#_Toc454207342)

[4.2 Architecture du code (Préciser la Version) 5](#_Toc454207343)

[4.3 Analyse par des diagrammes UML (Préciser la Version) 6](#_Toc454207344)

[4.4 Mécanismes d’identification et d’authentification 6](#_Toc454207345)

[4.5 Description du processus de gestion D’ERREUR 7](#_Toc454207346)

[4.6 Gestion des accès concurrents 8](#_Toc454207347)

[4.7 Sécurité 8](#_Toc454207348)

[4.8 Autres éléments techniques 8](#_Toc454207349)

[5. Plateformes Matérielles 8](#_Toc454207350)

[5.1 Environnement preconisé 8](#_Toc454207351)

[5.2 Spécificités relatives aux performances 8](#_Toc454207352)

[6. Annexe : Normes et Standards de Réalisation 8](#_Toc454207353)

# Préambule

## Lexique

Dans ce document, les abréviations suivantes seront utilisées:

[Lister les abréviations utilisées et indiquer leur signification]

## Signalétique

Tout au long de ce document, les pictogrammes ci-dessous sont utilisés afin de souligner des points ou des notions importantes.

|  |  |
| --- | --- |
| icono2 | Information importante |
| icono3 | **Risque** face à un paramétrage ou à une action spécifique |
| icono1 | Action à éviter |
| icono5 | Action obligatoire |
| icono4 | Procédure sensible ou difficile. A prendre en compte impérativement |
| policeman_usa_on | Actions réservées aux administrateurs |
| businessman2_on | Actions réservées aux utilisateurs |

# Architecture Technique Générale

## Schéma global d’architecture

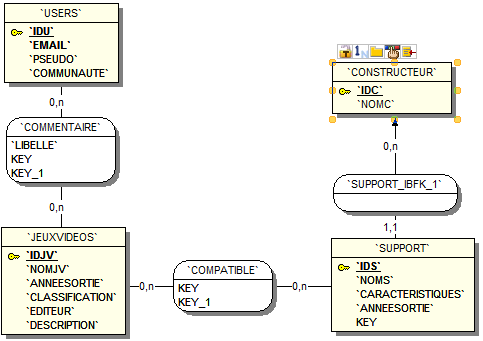
## Plateforme technique icono5

| *Type* | *OS/Plateforme* | *Logiciel* | *Version* |
| --- | --- | --- | --- |
| *Serveur Web* | *Windows* | *IIS* | *8.5* |
| *Serveur de base de données* | *Windows* |  |  |
| *Serveur FTP* | *Windows* | *IIS* | *8.5* |
| *Langage* |  |  |  |
| *Logiciel de gestion de version* |  |  |  |
| *Logiciel de gestion de projet* |  |  |  |

# Description des Données

## Modèle conceptuel (1.0)

## ) icono5



## Modèle logique (V1.0)

table JeuxVideos (**idJV**, nomJV, anneeSortie, classification, editeur, description) : jeux videos

// classification correspond à un âge minimum conseillé

table Support (**idS**, nomS, anneeSortie, caractéristiques, idC#) : console de jeux

table Compatible (**idJV#, idS#**) : permet de vérifier la compatibilité d'un jeu en fonction d'une console

table Constructeur (**idC**,nomC) : liste des constructeurs de supports

table Users (**IdU**, emailU, pseudo, communaute) : liste des utilisateurs qui ont posté un commentaire. La communauté est une information pouvant être (geek, joueur occasionnel, etc…)

table Commentaire (**idJV#, idU#**, libelle) : liste des commentaires mis sur les jeux par les utilisateurs.

# Description du Code

## Historique de la solution

V1 fournie:

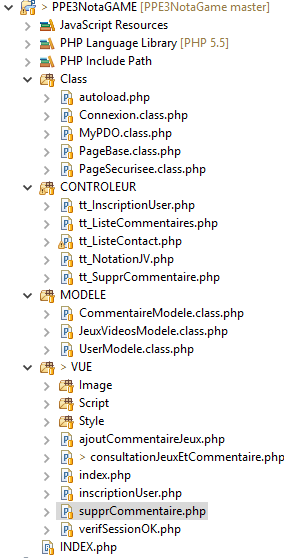
La société 1FO@SIO a une première version V1 (principe de développement par itération : méthode AGILE). Cette version V1 comprend un **site Internet** en M.V.C (Modèle VUE CONTROLEUR)

* + qui permet à un utilisateur (pseudo) de consulter tous les jeux et de mettre un commentaire (max 100 caractères) sur un jeu précis.
  + qui possède un mode administrateur qui permet de visualiser les commentaires postés et de les supprimer s'il estime que ces commentaires ne sont pas *corrects* (charte de déontologie de l'association)

V2 A REALISER icono5

## Architecture du code (Préciser la Version)

***ARCHITECTURE MVC (Modèle Vue Controleur)***

***les différentes fonctionnalités avec les fichiers associés :***

# 1 Consultation des commentaires icono5

# 2 S'inscrire

InscriptionUser.php

tt\_InscriptionUser.php

UserModele.class.php

Connexion.class.php

MyPDO.class.php

# 3 Notation des Jeux vidéos

ajoutCommentaireJeux.php

tt\_NotationJV.php

CommentaireModele.class.php

Connexion.class.php

MyPDO.class.php

UserModele.class.php

Connexion.class.php

MyPDO.class.php

# 4 supprimer les commentaires

supprCommentaire.php

Affichage de le liste des commentaires

ajaxListeCommentaire.js

tt\_listeCommentaires.php

CommentaireModele.class.php

Connexion.class.php

MyPDO.class.php

### 4.1 suppression du commentaire sélectionné

ajaxSupprCommentaire.js

tt\_SupprCommentaire.php?idJV= x &idU= x

CommentaireModele.class.php

Connexion.class.php

MyPDO.class.php

### 4.2 Rafraichir la liste des commentaires

ajaxListeCommentaire.js

tt\_listeCommentaires.php

CommentaireModele.class.php

Connexion.class.php

MyPDO.class.php

***V1.0 : DIAGRAMME de CLASSES TECHNIQUES:***

La classe PageBase permet de construire tous les pages présentes sur le site. Cela procure l’avantage d’une grande efficacité dans les changements visuels ou fonctionnels.

La PageSecurisee profite de tous ce que sait faire la PageBase. Elle permet de compléter le menu si l’on est connecté à l’application (session active)



***V1.0 : DIAGRAMME de CLASSES METIERS:***

Pas de classes métiers ici, l’application est toujours connectée donc nous interagissons directement avec la BDD via le MODELE

## Analyse par des diagrammes UML (Préciser la Version)

***V2.x : DIAGRAMME Etat-Transition (DET sur le commentaire) :*** icono5

## Mécanismes d’identification et d’authentification

***Mécanisme d'authentification :***

Sur le site, une authentification en Session est réalisée afin de donner des droits différents :

businessman2_onaccès aux pages basiques :

V1.0 :

* Cet utilisateur doit se créer un compte (s'inscrire)
* il a le droit de poster des commentaires sur des jeux (notation des jeux vidéo)
* il peut visualiser les commentaires de tout le monde sur les jeux présents

policeman_usa_onAccès aux pages sécurisées :

login : adminNG

mot de passe : *QSDFG*

V1.0 : Cet administrateur a le droit de supprimer des commentaires déjà postés

***V1.0 : DIAGRAMME de CAS D'UTILISATION :*** icono5

***V2.x : DIAGRAMME de CAS D'UTILISATION :*** icono5

## Description du processus de gestion D’ERREUR

**Chaque page** issue de Page base ou de Page Sécurisée possède **une DIV nommée infoERREUR** (qui contient elle-même une autre div dialog1)

Cette DIV est mise en forme par le CSS (gestionErreur.css) dès la création de la page.

Un script nommé utile.js est chargé dans la pageBase : il permet d’afficher ou pas la div infoERREUR

/\*ce script sert à montrer ou cacher la zone d'erreur\*/

**function** cacher(){

(document.getElementById('infoERREUR')).style.display = "none";

}

**function** montrer(){

(document.getElementById('infoERREUR')).style.display = "";

}

#### EN PHP classique

Comment utiliser le script utile.js ?

// DIV cachée au chargement de chaque page

**<script type="text/javascript">cacher();</script>**

// TRAITEMENT du RETOUR DE L'ERREUR par le controleur

**if** (*isset*(**$\_GET**['error']) && !*empty*(**$\_GET**['error'])) {

?> <script type=*"text/javascript"*> montrer();</script> <?php

$pageIndex->contenu .= '<div id="infoERREUR"><h1>Informations !</h1><div id="dialog1" >'. **$\_GET**['error'].'</div>';

$verif = *preg\_match*("/ERREUR/",**$\_GET**['error']);

**if** ( $verif == **TRUE** ){ // pour gérer la couleur du bouton selon le type d’erreur

$pageIndex->contenu .= '<a class="no" onclick="cacher();">OK</a></div>';

}**else** {

$pageIndex->contenu .= '<a class="yes" onclick="cacher();">OK</a></div>';

}

}

Lors des traitements en php un message est retourné à la page et est affiché dans cette DIV.

***Exemple en PHP :***// redirection vers la page appelante en lui passant le message

header ('Location: ../VUE/index.php?error=' . $msgERREUR);

#### EN AJAX : c’est le javascript qui va modifier la div d’erreur (contenu de la div et couleur du bouton)

// TRAITEMENT du RETOUR DE L'ERREUR ICI par la requête AJAX

//ici c'est le javascript qui va changer la couleur du boutton OK

$pageSupprCommentaire->contenu .= '<div id="infoERREUR"><h1>Informations !</h1><div id="dialog1" ></div>

<a id="**baliseA**" class="no" onclick="cacher();">OK</a></div>';

*Remarque* : Ajout d’un id baliseA pour changer la couleur du bouton

***en Ajax :***

//TOUJOURS : une fois réceptionné les donnees en JSON, on récupére juste le message d erreur renvoye par l'AJAX

filterDataRequest.always(**function**(data) {

(document.getElementById('infoERREUR')).style.display = ""; //pour la mettre en affichage

$('#dialog1').text(""); //remise à blanc de la div

/\*Pour afficher le message retourné en JSON par la requête AJAX\*/

$.each(data, **function**(index, value) {

**if** (index == "success"){ // attention aux 3 égals : égalité stricte

document.getElementById('baliseA').className= "yes";

}

**else**{

document.getElementById('baliseA').className= "no";

}

$('#dialog1').append(value);

});

});

## Sécurité

Version 1.0 : L'organisation du code en MVC aide à la sécurité des pages

Toutefois pur éviter l'injection SQL dans les zones de texte des formulaires, il faudrait faire des requêtes préparées (PREPARE) puis une exécution ultérieure (EXECUTE).

À prévoir dans une version ultérieure.

# Plateformes Matérielles

## Environnement préconisé icono5

[Décrire l’environnement nécessaire pour installer l’application. Préciser notamment  les caractéristiques Minimales des serveurs (CPU, disque dur, RAM)]

## Spécificités relatives aux performances icono5

[Noter ici quels sont les risques relatifs

à une volumétrie importante(**montée en charge**),

un nombre d’utilisateurs simultanés important,

une exigence en terme de temps de réponse…]

# Annexe : Normes et Standards de Réalisation

***L’application doit répondre aux normes et standards en vigueur chez la SSII 1FO@SIO.***

*Dans la base de données nommée :* ***bd\_PPE3\_notagame***

* Moteur : INNODB pour la gestion des clés étrangères
* tous les champs sont écrits en **MAJUSCULES et sans accents**
* toutes les tables sont écrites en **minuscules et sans accents**

Normes graphiques :

les CSS :

* **base.css et modele.css** : apparence générale du site avec l'entête, le menu, le contenu et le pied de page
* **template.css** regroupe les styles propres aux formulaires et aux tableaux :
* **gestionErreur.css** : gestion de la mise en forme de toutes les erreurs présentes sur le site.

Fin du document