

TASKSTEP : AUDIT D'OPTIMISATION DE QUALITE LOGICIELLE

1PEUDINSPI :

- Tristan DAL MOLIN
- Moulay-Wassim ALAOUI
- Jules DUTRION
- Matteo DE MARCO
- Wassim DIOURI

BUT2 Informatique



I. Ancienne conception du projet

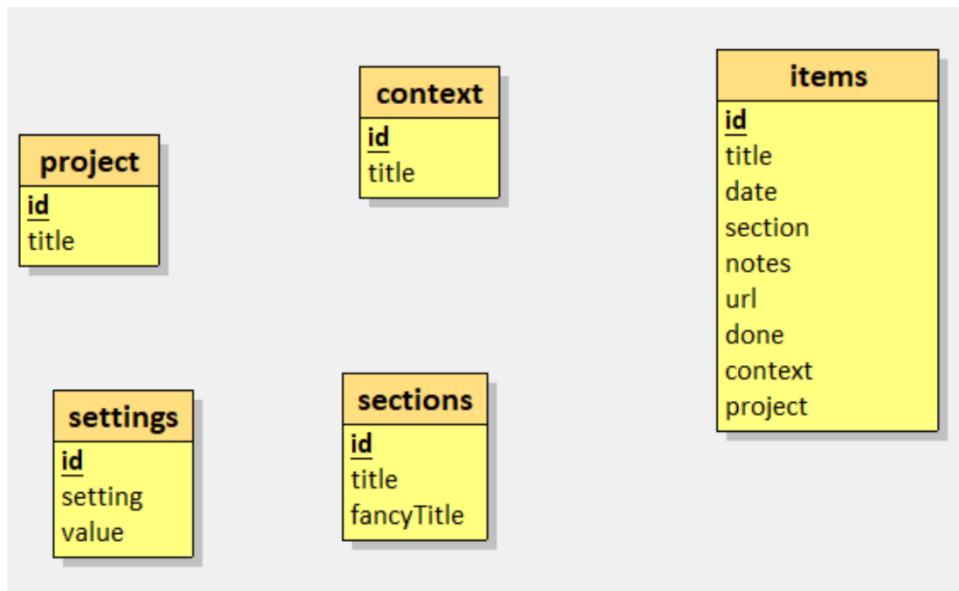


Figure 1 : Modèle de Conception de Données du projet

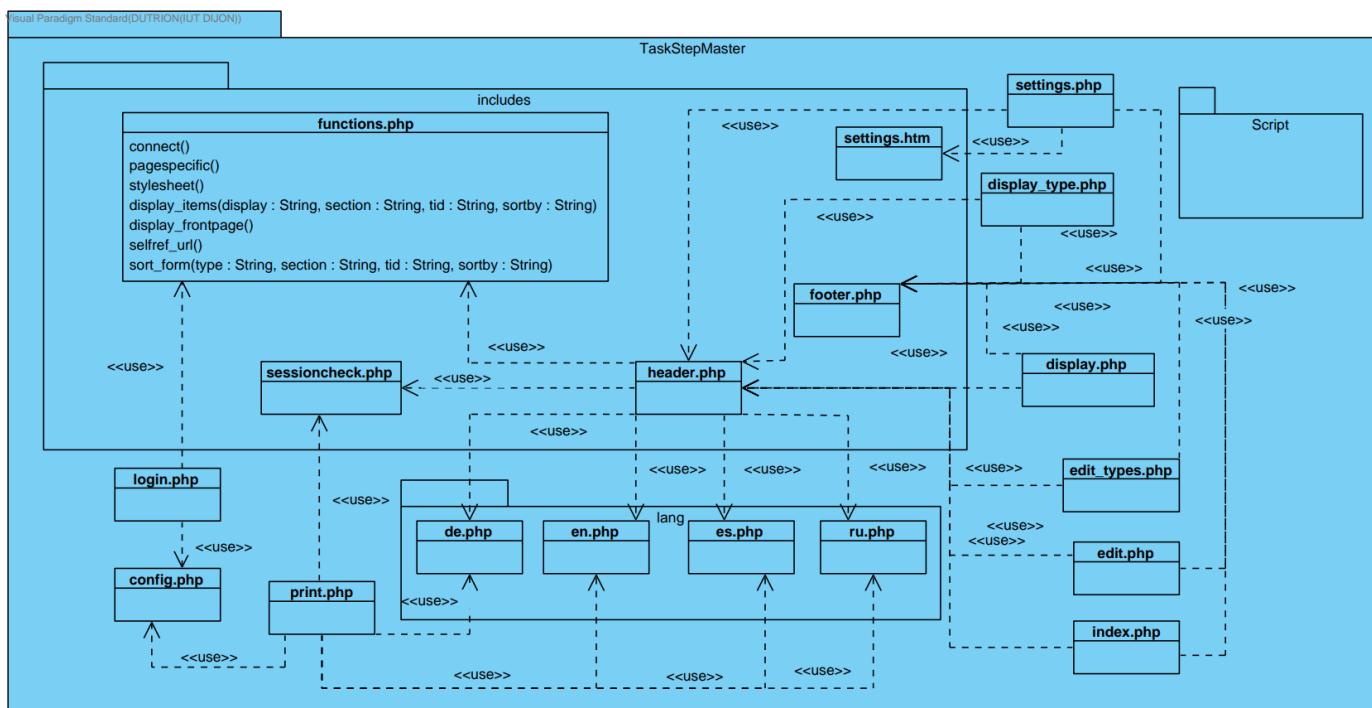


Figure 2 : Conception de l'application (UML)

II. Nouvelle conception du projet

Afin de remanier l'application de façon SOLID et optimisée, nous avons refait la structure du projet de 0 par Programmation Orientée Objets en PHP, utilisant notamment une structure MVC :

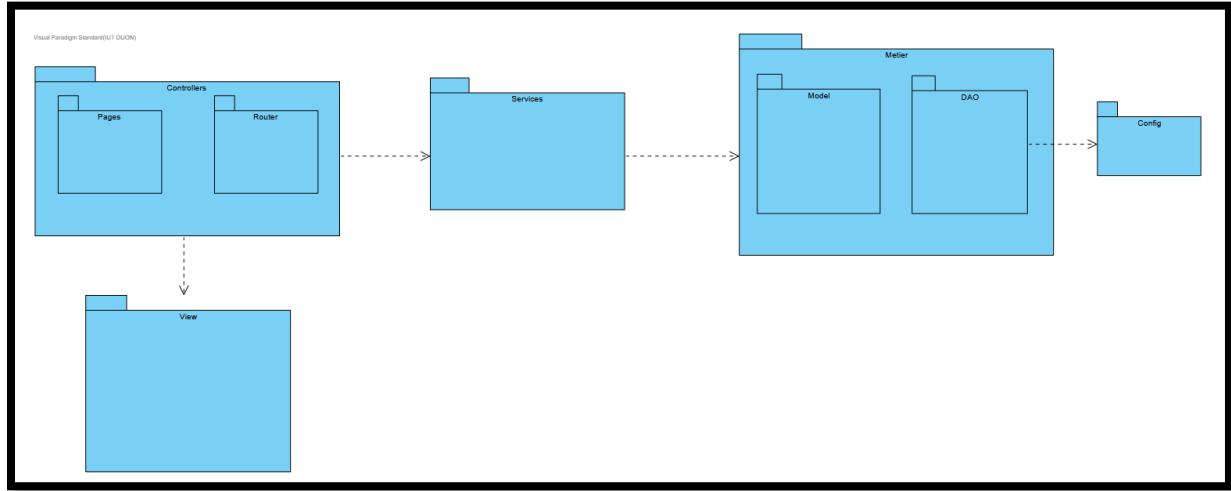


Figure 3 : Diagramme des packages

III. Audit d'optimisation de qualité logicielle

L'audit qui suit contient les résultats de l'audit principal ainsi que ses corrections et optimisations en dernière colonne.

La notation des résultats est la suivante :

- Le point est une bonne pratique identifiée (point très bien respecté) : BP
- Le point est conforme (c'est respecté, éventuellement avec quelques limites) : CF
- Le point est non conforme (c'est insuffisamment respecté) : NC
- Le point est critique (ce n'est pas du tout respecté, c'est totalement absent...) : CR

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
A – Documentation technique				
A.1 – Documentation externe				
A.1.1 Document indiquant la plateforme d'exécution (Qu'installer sur une VM pour exécuter l'appli)	Visual Studio Code	Lecture du fichier README.htm pour analyser les étapes d'installation locale et vérifier les composants nécessaires (AMP).	BP – L'application doit être exécutée sur un serveur local type XAMPP contenant Apache, MySQL et PHP et tout est indiqué en README.	Rien à modifier
A.1.2 Document indiquant la plateforme de développement	Visual Studio Code	Lecture du README.md pour vérifier la plateforme de développement	BP – Le projet est développé en PHP. Aucun environnement spécifique n'est imposé, tout est indiqué en README	Rien à modifier
A.1.3 Document décrivant l'architecture globale de l'application	Visual Studio Code	Vérification que l'architecture du projet est présente dans sa documentation (UML, conception, ...)	N/A – Aucune architecture, non applicable	Un fichier .vpp contient la conception complète du projet (cf la nouvelle conception du projet)
A.1.4 Document décrivant les dépendances aux outils tiers de l'application	Visual Studio Code	Lecture du README.md pour vérifier les dépendances aux outils tiers de l'application (API extérieures, logiciels à installer en plus de l'application de base)	BP – Présence des dépendances du projet en README. La plateforme d'exécution nécessite PHP ≥ 7.0, MariaDB et un serveur local pour exécuter le projet	Aucune dépendance supplémentaire

A.1.5 Cahier des charges ou document décrivant les fonctionnalités de l'application	Visual Studio Code	Lecture du README.md pour vérifier si les fonctionnalités y sont décrites. (Diagrammes de séquences, descriptions des fonctionnalités)	CF – Les fonctionnalités sont listées dans le README : organisation des tâches, filtres contextuels/projets, affichage auto des tâches du jour, impression, surlignage des tâches urgentes ou en retard, multilingue (Anglais, Russe, Allemand, Espagnol).	Un UML du projet entier a pu être réalisé, comprenant notamment un diagramme de cas d'usage de l'application
---	--------------------	---	--	--

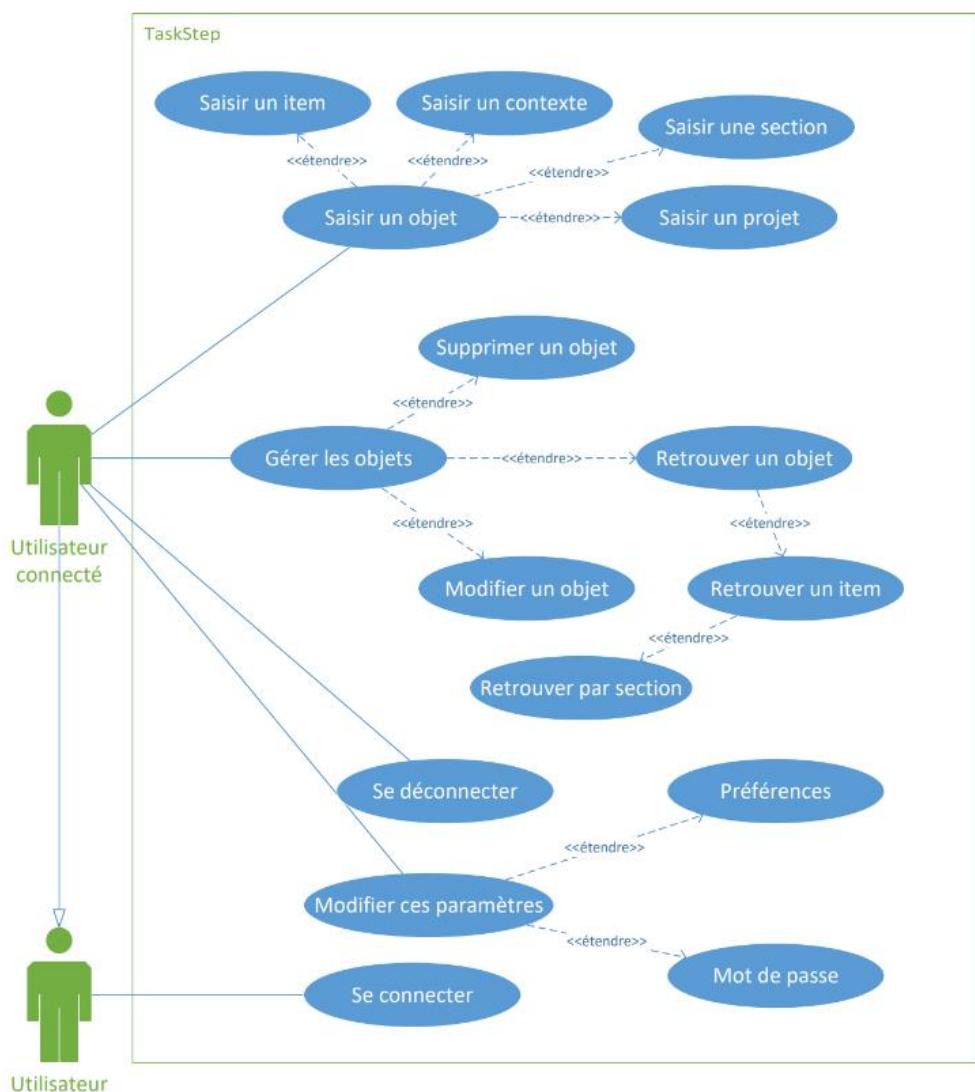


Figure 4 : Diagramme de cas d'usage

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
A – Documentation technique				
A.2 – Documentation interne				
A.2.1 Documentation in-situ du code présente (Code documenté)	Visual Studio Code	Analyse du code existants des différents fichiers présents dans le projet.	CR – Il semble qu'il n'y ait vraiment aucune documentation dans tout le projet. Seulement des commentaires rarement.	
A.2.2 Documentation du code cohérente avec le code et à jour	N/A	Vérification que la documentation a du sens et a une version, que les paramètres y soient ajoutés si ajoutés à une version du site	N/A (Pas de documentation)	Documentation, commentaire, mis en place sur tout le projet (en anglais)
A.2.3 Documentation du code cohérente avec les spécifications fonctionnelles (cf A.1.5., si ça respecte les fonctionnalités décrites)	N/A	Présence de documentation cohérente avec les spécifications	N/A (Pas de documentation)	
A.2.4 Documentation du code claire, lisible, dans la langue souhaitée	N/A	Documentation claire, lisible et dans une même langue	N/A (Pas de documentation)	

Controllers \ Pages \ Home

ControllerHome

in package Application

[ControllerHome.php](#) : 10

The controller for the home page

Table of Contents

Properties

P  \$page : [HomePage](#)

Methods

M  [__construct\(\)](#) : mixed

The class constructor

M  [index\(\)](#) : void

Method showing the home page

Namespaces
Config
Controllers
Pages
Router
Logic
DAO
Model
PSR4
Service
Interface
TestUnitaire
TestBDD
Utils
Composer
Autoload
Laravel
SerializableClosure
Invoker
Exception
ParameterResolver
DI
Attribute
Compiler
Definition
Factory
Invoker

Logic \ DAO \ Interface

I_TaskDAO

in Application

[I_TaskDAO.php](#) : 14

DAO interface for the task table

Table of Contents

Methods

- [AddTask\(\)](#) : void
Methode to add a task to the database
- [DeleteTask\(\)](#) : void
Method to delete a task in the database
- [GetTask\(\)](#) : [Task](#)
Method to get a task from the database
- [ListTask\(\)](#) : array<string|int, [Task](#)>
Method to list the tasks
- [ListTaskByContext\(\)](#) : array<string|int, [Task](#)>
Method to list the tasks by section
- [ListTaskByProject\(\)](#) : array<string|int, [Task](#)>
Method to list the tasks by project
- [ListTaskBySection\(\)](#) : array<string|int, [Task](#)>
Method to list the tasks by section
- [TaskDone\(\)](#) : void
Method to set a task as done
- [UpdateTask\(\)](#) : void

On this page

Table Of Contents

[Constants](#)

[Methods](#)

Methods

- [AddTask\(\)](#)
- [DeleteTask\(\)](#)
- [GetTask\(\)](#)
- [ListTask\(\)](#)
- [ListTaskByContext\(\)](#)
- [ListTaskByProject\(\)](#)
- [ListTaskBySection\(\)](#)
- [TaskDone\(\)](#)
- [UpdateTask\(\)](#)

Routes

[Routes.php](#) : 57

extends [RoutesAbstract](#)
in package Application

Class listing all the roads of the project

Table of Contents

Properties

- [\\$container](#) : [ContainerInterface](#)
- [\\$ctrlList](#) : array<string|int, mixed>
- [\\$routeList](#) : array<string|int, mixed>

Methods

- [__construct\(\)](#) : mixed
The class constructor
- [createControllerList\(\)](#) : void
Method to initialize the list of controllers
- [createRouteList\(\)](#) : void
Method to initialize the list of routes
- [getConnexionRoute\(\)](#) : string
Method to obtain the name of the connexion page
- [getDefaultRoute\(\)](#) : string
The default route getter for the app
- [getRoute\(\)](#) : [Route](#)
Method returning the asked route or the default one

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
B – Tests de l’application				
B.1 – Tests fonctionnels				
B.1.1 Les cas d’usage de l’application sont tous implémentés (Respect des diagrammes d’usage en conception)	N/A	Vérification que les cas d’usage sont tous respectés	N/A -- Pas de diagrammes de conceptions	/
B.1.2 Les spécifications fonctionnelles sont respectées	Firefox/ Edge	Vérifications que les fonctionnalités de la documentation sont bien respectées	CF – La plupart sont conformes, excepté le changement de langue du navigateur (changement de langue possible via le fichier config)	Changement de langue implémenté selon la langue du navigateur directement

Exemple de changement de langue :

The screenshot shows two views of the TaskStep application. The top view is the 'Home Page' in English, and the bottom view is the same page in French. Both pages include a sidebar for adding tasks and a main content area with a task form and welcome message.

Top View (English):

- Header: Page d'accueil
- Top right: Paramètres, Aide, Déconnexion
- Top left: Date: 15/05/2025
- Right sidebar: Accueil, Tous les éléments, Par Contexte, Par Projet
- Main content: Formulaire de tâche, Bienvenue sur TaskStep, Task form, Welcome to TaskStep.

Bottom View (French):

- Header: Page d'accueil
- Top right: Settings, Help, Logout
- Top left: Date: 15/05/2025
- Right sidebar: Home, All items, By Context, By Project
- Main content: Formulaire de tâche, Bienvenue sur TaskStep, Task form, Welcome to TaskStep.

Left Sidebar (Visible in both views):

- Ajouter une tâche
- Formulaire de tâche
- Task form
- Welcome to TaskStep
- Task Step
- Paramètres
- Aide
- Déconnexion

Language Selection Dialog (Visible in the bottom view):

Titre: Paramètres de langue des pages web

Text: Certaines pages web sont proposées dans plusieurs langues. Choisissez les langues d'affichage de ces pages, par ordre de préférence.

Anglais (États-Unis) [en-us]	Monter
Français (France) [fr-fr]	Descendre
Français [fr]	Supprimer
Anglais [en]	

Choisir une langue à ajouter...

OK Annuler

B.1.3 L'application fonctionne sur les plateformes d'exécution spécifiées	Oracle vm virtualbox	Vérifier que l'application fonctionne sous machine virtuelle en suivant le guide d'installation et d'utilisation.	CF – Tutoriel de mise en place du serveur fonctionnel, à mettre à jour avec la mise à jour du projet	Fichier README mis à jour avec l'optimisation du logiciel
--	----------------------	---	--	---

Installation

- Create database, user and rights. Use your own secure credentials!

```
CREATE DATABASE taskstep;
CREATE USER 'taskstep'@'localhost' IDENTIFIED BY 'taskstep';
GRANT ALL PRIVILEGES ON taskstep.* TO 'taskstep'@'localhost';
```



- Go to installation URL: <https://www.example.com/taskstep/install/install.php>
- Go to the main URL to connect to the app : <https://www.example.com/taskstep/index.php?action=ConnexionPage>
- If you want, and for security reasons, remove `install.php` from the project once your configuration is done
- Create an account and connect to use the application

B.1.4 L'application est robuste aux cas particuliers	Firefox	Vérification qu'aucune mauvaise saisie n'est possible (lettres à la place d'un nombre par exemple)	NC – Insertion possibles avec des champs nécessaires vides, même nom de contexte et de projet possibles, une date peut être entrée en tant que texte et non que date	Formulaire mis à jour, empêchant d'entrer autre chose qu'un lien dans le champs URL ou autre chose qu'une date en date par exemple
---	---------	--	--	--

URL(*)

Pas un URL

Veuillez saisir une URL.

Soumettre

Date limite(*)

22/07/2010



Soumettre

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
B – Tests de l’application				
B.2 – Tests unitaires				
B.2.1 Les tests unitaires sont présents et cohérents avec les spécifications	Visual Studio Code	Vérification que l’application a des tests unitaires cohérents pour les cas généraux et particuliers	NC – Pas de tests unitaires	
B.2.2 Les tests unitaires passent tous	Visual Studio Code	Vérification que tous les tests unitaires passent	N/A -- Pas de tests unitaires	
B.2.3 Les tests unitaires sont indépendants les uns des autres	Visual Studio Code	Tests unitaires indépendants entre eux, simulation de couches inférieures, fakes DAO par exemple	N/A -- Pas de tests unitaires	Tests unitaires mis en place séparément sur le site web, vérifiant les interactions avec la base de données.
B.2.4 les cas particuliers (entrées invalides, effets de bord) sont testés	Visual Studio Code	Vérification que les cas particuliers sont testés	N/A -- Pas de tests unitaires	Seuls les tests en couche DAO ont pu être réalisés, ils sont tous fonctionnels
B.2.5 la couverture des tests est correcte	Visual Studio Code	Couverture des tests correcte (pourcentage éventuel)	N/A -- Pas de tests unitaires	

Tests Unitaires

Unit test for Context

Insert Test : Success

ListContext Test : Success

GetByld Test : Success

Update Test : Success

Delete Test : Success

Unit test for Project

Insert Test : Success

GetByld Test : Success

Update Test : Success

list all Test : Success

Delete Test : Success

Unit test for the Task Table

Insert Test : Success

GetByld Test : Success

Update Test : Success

Delete Test : Success

List all Test : Success

List by Context Test : Success

List by Project Test : Success

List by Section Test : Success

Unit test for Section

TestAddSection : Success

TestGetSection : Success

TestUpdateSection : Success

TestDeleteSection : Success

TestListSection : Success

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
C – Format du code source				
C.1 – Construction du code				
C.1.1 Lors de la construction du code (compilation) il n'y a pas d'avertissement (À modifier si c'est dangereux, peuvent être laissés de côté sinon)	« phpstan » sous Composer et Linux, Visual Studio Code	Commande « phpstan analyse » effectuée sur chaque classe PHP du projet	NC – Plus de 700 erreurs sont trouvées au total, dont beaucoup de méthodes appelées à des éléments	
C.1.2 Lors de la construction du code il n'y a pas d'erreurs	« phpstan » sous Composer et Linux, Visual Studio Code	Commande « phpstan analyse » effectuée sur chaque classe PHP du projet, sous une configuration nous permettant d'obtenir tous les résultats possibles	NC – Plus de 700 erreurs sont trouvées au total, beaucoup de non-typage, de méthodes appelées sur des éléments « mixed »	Après refonte du projet dans une version plus récente de PHP, le test sous phpstan ne renvoie désormais plus aucune erreur

```
root@PC-RCC:/mnt/c/xampp/htdocs/S4_C1_1PeuDinspi# vendor/bin/phpstan analyse Config Logic Controllers Service Views Utils install.php Defines.php index.php Psr4AutoloaderClass.php
113/113 [██████████] 100% [OK] No errors

💡 Tip of the Day:
PHPStan is performing only the most basic checks.
You can pass a higher rule level through the --level option
(the default and current level is 0) to analyse code more thoroughly.
```

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
C – Format du code source				
C.2 – Aspect du code source				
C.2.1 L'indentation du code est correcte	Visual Studio Code, Webimpress Coding Standard	On effectue une analyse du code sous Webimpress Coding Standard afin de vérifier le nombre d'erreurs d'indentation	NC- On remarque donc la présence de presque 2000 erreurs d'indentation dans ce projet (voir figure ci-dessous)	Après analyse avec Webimpress Coding Standards et différents nettoyages du projet, plus aucune erreur d'indentation n'est finalement détectée.

```

analyse.txt
1464
1465
1466 esc[1mFILE: /mnt/c/xampp/htdocs/S4_C1_1PeuDinspi/Views/User/RegisterPage.phpesc[0m
1467 -----
1468 esc[1mFOUND 1 ERROR AND 4 WARNINGS AFFECTING 5 LINESesc[0m
1469
1470 | 22 | esc[31mERROResc[0m | Protected method name "RegisterPage::GenerateContent" is not in
1471 | | camel caps format
1472 | 33 | esc[33mWARNINGesc[0m | Line exceeds 120 characters; contains 155 characters
1473 | 36 | esc[33mWARNINGesc[0m | Line exceeds 120 characters; contains 167 characters
1474 | 39 | esc[33mWARNINGesc[0m | Line exceeds 120 characters; contains 174 characters
1475 | 44 | esc[33mWARNINGesc[0m | Line exceeds 120 characters; contains 186 characters
1476 -----
1477
1478
1479 esc[1mFILE: /mnt/c/xampp/htdocs/S4_C1_1PeuDinspi/index.phpesc[0m
1480 -----
1481 esc[1mFOUND 1 ERROR AFFECTING 1 LINEesc[0m
1482 -----
1483 | 47 | esc[31mERROResc[0m | Default case in switch should be as last; another case found here

```

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
C.2.2 Il n'y a pas d'espace inutiles (blancs) dans le code	Visual Studio Code	Vérification dans chaque fichier qu'aucun grand espace inutile n'est laissé	BP – Aucun réel espace inutile ne se trouve dans le code	/
C.2.3 Le code est facile à lire	Visual Studio Code	On vérifie la présence de commentaires si une fonction n'est pas assez claire sur son propos, la bonne nomenclature de variables et de fonctions, etc.	NC – Les variables sont bien nommées, certains scripts PHP sont correctement commentés (bien que manquant de documentation), mais beaucoup manquent de clarifications et il est difficile de comprendre quelle classe fait quoi (voir figures ci-dessous)	Durant la refonte architecturale du projet, nous avons bien pris le soin, non seulement de rajouter la documentation nécessaire, mais aussi de nommer correctement les différentes variables. La séparation du projet en plusieurs couches rend également plus compréhensible la responsabilité de chacun

```

Views > Task > ListTaskPage.php > PHP > ListTaskPage

11 class ListTaskPage extends AbstractPage {
12
13     /**
14      * @reference | 0 overrides | prototype
15      */
16     protected function GenerateContent(): string
17     {
18         $returnValue = "<main class='main-content'>";
19
20         $returnValue .= $this->GenerateForm();
21
22         if (empty($this->tasks)) {
23             $returnValue .= "<p class='empty-message'>" . $this->languageManager->translate(key: "NoTasksFound") . "</p>";
24         } else {
25             $returnValue .= "<div class='item-list'>";
26             foreach ($this->tasks as $task) {
27                 $title = htmlspecialchars(string: $task->getTitle());
28                 $date = $task->getDate() ? $task->getDate()->format(format: "Y-m-d") : "N/A";
29                 $done = $task->getDone();
30                 $colorClass = $done ? "presentation-card" : "delete-card";
31                 $contextTitle = htmlspecialchars(string: $task->getContext()->getTitle() ?? "No context");
32                 $projectTitle = htmlspecialchars(string: $task->getProject()->getTitle() ?? "No project");
33
34                 $returnValue .= "
35                     <div class='card $colorClass'>
36                         <h2 class='card-title'$title</h2>
37                         <p><strong>Context:</strong> $contextTitle</p>
38                         <p><strong>Project:</strong> $projectTitle</p>
39                         <p><strong>Date:</strong> $date</p>
40                         <p><strong>Status:</strong> " . ($done ? "☒ " . $this->languageManager->translate(key: "isDone") : "☒ " . $this->languageManager->translate(key: "notDone")) . "</p>
41                         <div class='item-actions'>
42                             <a href='index.php?action=ViewTask&id=' . $task->getId() . '" class='btn btn-view'>" . $this->languageManager->translate(key: "View") . "</a>
43                             <a href='index.php?action=TaskDone&id=' . $task->getId() . '" class='btn btn-edit'>" . $this->languageManager->translate(key: "GetDone") . "</a>
44                             <a href='index.php?action=UpdateTask&id=' . $task->getId() . '" class='btn btn-edit'>" . $this->languageManager->translate(key: "Edit") . "</a>
45                             <a href='index.php?action=DeleteTask&id=' . $task->getId() . '" class='btn btn-delete'>" . $this->languageManager->translate(key: "Delete") . "</a>
46                         </div>
47                     </div>
48                 ";
49             }
50             $returnValue .= "</div>";
51         }
52
53         $returnValue .= "</main>";
54         return $returnValue;
55     }
56
57     /**
58      * @reference | 0 overrides | prototype
59      */
60     protected function GenerateForm(): string
61     {
62         $form = "<form method='POST'>
63             <div class='form-group'>
64                 <label>Search:</label>
65                 <input type='text' name='search' value='Search' placeholder='Search' />
66             </div>
67             <div class='form-group'>
68                 <label>Filter:</label>
69                 <select name='filter'>
70                     <option value='All'>All</option>
71                     <option value='Completed'>Completed</option>
72                     <option value='Incomplete'>Incomplete</option>
73                 </select>
74             </div>
75             <div class='form-group'>
76                 <label>Sort By:</label>
77                 <select name='sort_by'>
78                     <option value='Title'>Title</option>
79                     <option value='Date'>Date</option>
80                 </select>
81             </div>
82         </form>
83     }
84
85     /**
86      * @reference | 0 overrides | prototype
87      */
88     protected function GenerateFooter(): string
89     {
90         $footer = "<div class='footer'>
91             <p>Copyright © 2024 Task Management System. All rights reserved.</p>
92         </div>";
93
94         return $footer;
95     }
96
97     /**
98      * @reference | 0 overrides | prototype
99      */
100    protected function GenerateHeader(): string
101    {
102        $header = "<header>
103            <nav>
104                <ul>
105                    <li>Home</li>
106                    <li>Tasks</li>
107                    <li>About</li>
108                </ul>
109            </nav>
110        </header>";
111
112        return $header;
113    }
114
115    /**
116     * @reference | 0 overrides | prototype
117     */
118    protected function GenerateSidebar(): string
119    {
120        $sidebar = "<aside>
121            <ul>
122                <li>Recent Tasks</li>
123                <li>Completed Tasks</li>
124                <li>Incomplete Tasks</li>
125            </ul>
126        </aside>";
127
128        return $sidebar;
129    }
130
131    /**
132     * @reference | 0 overrides | prototype
133     */
134    protected function GenerateFooterLinks(): string
135    {
136        $footerLinks = "<div class='footer-links'>
137            <ul>
138                <li>Home</li>
139                <li>Tasks</li>
140                <li>About</li>
141            </ul>
142        </div>";
143
144        return $footerLinks;
145    }
146
147    /**
148     * @reference | 0 overrides | prototype
149     */
150    protected function GeneratePageHeader(): string
151    {
152        $pageHeader = "<div class='page-header'>
153            <h1>Task Management System</h1>
154        </div>";
155
156        return $pageHeader;
157    }
158
159    /**
160     * @reference | 0 overrides | prototype
161     */
162    protected function GeneratePageFooter(): string
163    {
164        $pageFooter = "<div class='page-footer'>
165            <p>Task Management System</p>
166        </div>";
167
168        return $pageFooter;
169    }
170
171    /**
172     * @reference | 0 overrides | prototype
173     */
174    protected function GeneratePageContent(): string
175    {
176        $pageContent = "<div class='page-content'>
177            <h2>List of Tasks</h2>
178            <table>
179                <thead>
180                    <tr>
181                        <th>Title</th>
182                        <th>Date</th>
183                        <th>Status</th>
184                    </tr>
185                </thead>
186                <tbody>
187                    <tr>
188                        <td>Task 1</td>
189                        <td>2024-01-01</td>
190                        <td>Incomplete</td>
191                    </tr>
192                    <tr>
193                        <td>Task 2</td>
194                        <td>2024-01-02</td>
195                        <td>Incomplete</td>
196                    </tr>
197                    <tr>
198                        <td>Task 3</td>
199                        <td>2024-01-03</td>
200                        <td>Incomplete</td>
201                    </tr>
202                </tbody>
203            </table>
204        </div>";
205
206        return $pageContent;
207    }
208
209    /**
210     * @reference | 0 overrides | prototype
211     */
212    protected function GeneratePageFooterLinks(): string
213    {
214        $pageFooterLinks = "<div class='page-footer-links'>
215            <ul>
216                <li>Home</li>
217                <li>Tasks</li>
218                <li>About</li>
219            </ul>
220        </div>";
221
222        return $pageFooterLinks;
223    }
224
225    /**
226     * @reference | 0 overrides | prototype
227     */
228    protected function GeneratePageHeaderLinks(): string
229    {
230        $pageHeaderLinks = "<div class='page-header-links'>
231            <ul>
232                <li>Home</li>
233                <li>Tasks</li>
234                <li>About</li>
235            </ul>
236        </div>";
237
238        return $pageHeaderLinks;
239    }
240
241    /**
242     * @reference | 0 overrides | prototype
243     */
244    protected function GeneratePageContentLinks(): string
245    {
246        $pageContentLinks = "<div class='page-content-links'>
247            <ul>
248                <li>Home</li>
249                <li>Tasks</li>
250                <li>About</li>
251            </ul>
252        </div>";
253
254        return $pageContentLinks;
255    }
256
257    /**
258     * @reference | 0 overrides | prototype
259     */
260    protected function GeneratePageFooterContent(): string
261    {
262        $pageFooterContent = "<div class='page-footer-content'>
263            <p>Task Management System</p>
264        </div>";
265
266        return $pageFooterContent;
267    }
268
269    /**
270     * @reference | 0 overrides | prototype
271     */
272    protected function GeneratePageHeaderContent(): string
273    {
274        $pageHeaderContent = "<div class='page-header-content'>
275            <h1>Task Management System</h1>
276        </div>";
277
278        return $pageHeaderContent;
279    }
280
281    /**
282     * @reference | 0 overrides | prototype
283     */
284    protected function GeneratePageContentHeader(): string
285    {
286        $pageContentHeader = "<div class='page-content-header'>
287            <h2>List of Tasks</h2>
288        </div>";
289
290        return $pageContentHeader;
291    }
292
293    /**
294     * @reference | 0 overrides | prototype
295     */
296    protected function GeneratePageContentFooter(): string
297    {
298        $pageContentFooter = "<div class='page-content-footer'>
299            <ul>
300                <li>Home</li>
301                <li>Tasks</li>
302                <li>About</li>
303            </ul>
304        </div>";
305
306        return $pageContentFooter;
307    }
308
309    /**
310     * @reference | 0 overrides | prototype
311     */
312    protected function GeneratePageFooterContentHeader(): string
313    {
314        $pageFooterContentHeader = "<div class='page-footer-content-header'>
315            <ul>
316                <li>Home</li>
317                <li>Tasks</li>
318                <li>About</li>
319            </ul>
320        </div>";
321
322        return $pageFooterContentHeader;
323    }
324
325    /**
326     * @reference | 0 overrides | prototype
327     */
328    protected function GeneratePageFooterContentFooter(): string
329    {
330        $pageFooterContentFooter = "<div class='page-footer-content-footer'>
331            <ul>
332                <li>Home</li>
333                <li>Tasks</li>
334                <li>About</li>
335            </ul>
336        </div>";
337
338        return $pageFooterContentFooter;
339    }
340
341    /**
342     * @reference | 0 overrides | prototype
343     */
344    protected function GeneratePageHeaderContentHeader(): string
345    {
346        $pageHeaderContentHeader = "<div class='page-header-content-header'>
347            <ul>
348                <li>Home</li>
349                <li>Tasks</li>
350                <li>About</li>
351            </ul>
352        </div>";
353
354        return $pageHeaderContentHeader;
355    }
356
357    /**
358     * @reference | 0 overrides | prototype
359     */
360    protected function GeneratePageHeaderContentFooter(): string
361    {
362        $pageHeaderContentFooter = "<div class='page-header-content-footer'>
363            <ul>
364                <li>Home</li>
365                <li>Tasks</li>
366                <li>About</li>
367            </ul>
368        </div>";
369
370        return $pageHeaderContentFooter;
371    }
372
373    /**
374     * @reference | 0 overrides | prototype
375     */
376    protected function GeneratePageContentHeaderHeader(): string
377    {
378        $pageContentHeaderHeader = "<div class='page-content-header-header'>
379            <ul>
380                <li>Home</li>
381                <li>Tasks</li>
382                <li>About</li>
383            </ul>
384        </div>";
385
386        return $pageContentHeaderHeader;
387    }
388
389    /**
390     * @reference | 0 overrides | prototype
391     */
392    protected function GeneratePageContentHeaderFooter(): string
393    {
394        $pageContentHeaderFooter = "<div class='page-content-header-footer'>
395            <ul>
396                <li>Home</li>
397                <li>Tasks</li>
398                <li>About</li>
399            </ul>
400        </div>";
401
402        return $pageContentHeaderFooter;
403    }
404
405    /**
406     * @reference | 0 overrides | prototype
407     */
408    protected function GeneratePageContentFooterHeader(): string
409    {
410        $pageContentFooterHeader = "<div class='page-content-footer-header'>
411            <ul>
412                <li>Home</li>
413                <li>Tasks</li>
414                <li>About</li>
415            </ul>
416        </div>";
417
418        return $pageContentFooterHeader;
419    }
420
421    /**
422     * @reference | 0 overrides | prototype
423     */
424    protected function GeneratePageContentFooterFooter(): string
425    {
426        $pageContentFooterFooter = "<div class='page-content-footer-footer'>
427            <ul>
428                <li>Home</li>
429                <li>Tasks</li>
430                <li>About</li>
431            </ul>
432        </div>";
433
434        return $pageContentFooterFooter;
435    }
436
437    /**
438     * @reference | 0 overrides | prototype
439     */
440    protected function GeneratePageFooterContentHeaderHeader(): string
441    {
442        $pageFooterContentHeaderHeader = "<div class='page-footer-content-header-header'>
443            <ul>
444                <li>Home</li>
445                <li>Tasks</li>
446                <li>About</li>
447            </ul>
448        </div>";
449
450        return $pageFooterContentHeaderHeader;
451    }
452
453    /**
454     * @reference | 0 overrides | prototype
455     */
456    protected function GeneratePageFooterContentHeaderFooter(): string
457    {
458        $pageFooterContentHeaderFooter = "<div class='page-footer-content-header-footer'>
459            <ul>
460                <li>Home</li>
461                <li>Tasks</li>
462                <li>About</li>
463            </ul>
464        </div>";
465
466        return $pageFooterContentHeaderFooter;
467    }
468
469    /**
470     * @reference | 0 overrides | prototype
471     */
472    protected function GeneratePageFooterContentFooterHeader(): string
473    {
474        $pageFooterContentFooterHeader = "<div class='page-footer-content-footer-header'>
475            <ul>
476                <li>Home</li>
477                <li>Tasks</li>
478                <li>About</li>
479            </ul>
480        </div>";
481
482        return $pageFooterContentFooterHeader;
483    }
484
485    /**
486     * @reference | 0 overrides | prototype
487     */
488    protected function GeneratePageFooterContentFooterFooter(): string
489    {
490        $pageFooterContentFooterFooter = "<div class='page-footer-content-footer-footer'>
491            <ul>
492                <li>Home</li>
493                <li>Tasks</li>
494                <li>About</li>
495            </ul>
496        </div>";
497
498        return $pageFooterContentFooterFooter;
499    }
500
501    /**
502     * @reference | 0 overrides | prototype
503     */
504    protected function GeneratePageHeaderContentHeaderHeader(): string
505    {
506        $pageHeaderContentHeaderHeader = "<div class='page-header-content-header-header'>
507            <ul>
508                <li>Home</li>
509                <li>Tasks</li>
510                <li>About</li>
511            </ul>
512        </div>";
513
514        return $pageHeaderContentHeaderHeader;
515    }
516
517    /**
518     * @reference | 0 overrides | prototype
519     */
520    protected function GeneratePageHeaderContentHeaderFooter(): string
521    {
522        $pageHeaderContentHeaderFooter = "<div class='page-header-content-header-footer'>
523            <ul>
524                <li>Home</li>
525                <li>Tasks</li>
526                <li>About</li>
527            </ul>
528        </div>";
529
530        return $pageHeaderContentHeaderFooter;
531    }
532
533    /**
534     * @reference | 0 overrides | prototype
535     */
536    protected function GeneratePageHeaderContentFooterHeader(): string
537    {
538        $pageHeaderContentFooterHeader = "<div class='page-header-content-footer-header'>
539            <ul>
540                <li>Home</li>
541                <li>Tasks</li>
542                <li>About</li>
543            </ul>
544        </div>";
545
546        return $pageHeaderContentFooterHeader;
547    }
548
549    /**
550     * @reference | 0 overrides | prototype
551     */
552    protected function GeneratePageHeaderContentFooterFooter(): string
553    {
554        $pageHeaderContentFooterFooter = "<div class='page-header-content-footer-footer'>
555            <ul>
556                <li>Home</li>
557                <li>Tasks</li>
558                <li>About</li>
559            </ul>
560        </div>";
561
562        return $pageHeaderContentFooterFooter;
563    }
564
565    /**
566     * @reference | 0 overrides | prototype
567     */
568    protected function GeneratePageContentHeaderHeaderHeader(): string
569    {
570        $pageContentHeaderHeaderHeader = "<div class='page-content-header-header-header'>
571            <ul>
572                <li>Home</li>
573                <li>Tasks</li>
574                <li>About</li>
575            </ul>
576        </div>";
577
578        return $pageContentHeaderHeaderHeader;
579    }
580
581    /**
582     * @reference | 0 overrides | prototype
583     */
584    protected function GeneratePageContentHeaderHeaderFooter(): string
585    {
586        $pageContentHeaderHeaderFooter = "<div class='page-content-header-header-footer'>
587            <ul>
588                <li>Home</li>
589                <li>Tasks</li>
590                <li>About</li>
591            </ul>
592        </div>";
593
594        return $pageContentHeaderHeaderFooter;
595    }
596
597    /**
598     * @reference | 0 overrides | prototype
599     */
600    protected function GeneratePageContentHeaderFooterHeader(): string
601    {
602        $pageContentHeaderFooterHeader = "<div class='page-content-header-footer-header'>
603            <ul>
604                <li>Home</li>
605                <li>Tasks</li>
606                <li>About</li>
607            </ul>
608        </div>";
609
610        return $pageContentHeaderFooterHeader;
611    }
612
613    /**
614     * @reference | 0 overrides | prototype
615     */
616    protected function GeneratePageContentHeaderFooterFooter(): string
617    {
618        $pageContentHeaderFooterFooter = "<div class='page-content-header-footer-footer'>
619            <ul>
620                <li>Home</li>
621                <li>Tasks</li>
622                <li>About</li>
623            </ul>
624        </div>";
625
626        return $pageContentHeaderFooterFooter;
627    }
628
629    /**
630     * @reference | 0 overrides | prototype
631     */
632    protected function GeneratePageContentFooterHeaderHeader(): string
633    {
634        $pageContentFooterHeaderHeader = "<div class='page-content-footer-header-header'>
635            <ul>
636                <li>Home</li>
637                <li>Tasks</li>
638                <li>About</li>
639            </ul>
640        </div>";
641
642        return $pageContentFooterHeaderHeader;
643    }
644
645    /**
646     * @reference | 0 overrides | prototype
647     */
648    protected function GeneratePageContentFooterHeaderFooter(): string
649    {
650        $pageContentFooterHeaderFooter = "<div class='page-content-footer-header-footer'>
651            <ul>
652                <li>Home</li>
653                <li>Tasks</li>
654                <li>About</li>
655            </ul>
656        </div>";
657
658        return $pageContentFooterHeaderFooter;
659    }
660
661    /**
662     * @reference | 0 overrides | prototype
663     */
664    protected function GeneratePageContentFooterFooterHeader(): string
665    {
666        $pageContentFooterFooterHeader = "<div class='page-content-footer-footer-header'>
667            <ul>
668                <li>Home</li>
669                <li>Tasks</li>
670                <li>About</li>
671            </ul>
672        </div>";
673
674        return $pageContentFooterFooterHeader;
675    }
676
677    /**
678     * @reference | 0 overrides | prototype
679     */
680    protected function GeneratePageContentFooterFooterFooter(): string
681    {
682        $pageContentFooterFooterFooter = "<div class='page-content-footer-footer-footer'>
683            <ul>
684                <li>Home</li>
685                <li>Tasks</li>
686                <li>About</li>
687            </ul>
688        </div>";
689
690        return $pageContentFooterFooterFooter;
691    }
692
693    /**
694     * @reference | 0 overrides | prototype
695     */
696    protected function GeneratePageFooterHeaderHeaderHeader(): string
697    {
698        $pageFooterHeaderHeaderHeader = "<div class='page-footer-header-header-header'>
699            <ul>
700                <li>Home</li>
701                <li>Tasks</li>
702                <li>About</li>
703            </ul>
704        </div>";
705
706        return $pageFooterHeaderHeaderHeader;
707    }
708
709    /**
710     * @reference | 0 overrides | prototype
711     */
712    protected function GeneratePageFooterHeaderHeaderFooter(): string
713    {
714        $pageFooterHeaderHeaderFooter = "<div class='page-footer-header-header-footer'>
715            <ul>
716                <li>Home</li>
717                <li>Tasks</li>
718                <li>About</li>
719            </ul>
720        </div>";
721
722        return $pageFooterHeaderHeaderFooter;
723    }
724
725    /**
726     * @reference | 0 overrides | prototype
727     */
728    protected function GeneratePageFooterHeaderFooterHeader(): string
729    {
730        $pageFooterHeaderFooterHeader = "<div class='page-footer-header-footer-header'>
731            <ul>
732                <li>Home</li>
733                <li>Tasks</li>
734                <li>About</li>
735            </ul>
736        </div>";
737
738        return $pageFooterHeaderFooterHeader;
739    }
740
741    /**
742     * @reference | 0 overrides | prototype
743     */
744    protected function GeneratePageFooterHeaderFooterFooter(): string
745    {
746        $pageFooterHeaderFooterFooter = "<div class='page-footer-header-footer-footer'>
747            <ul>
748                <li>Home</li>
749                <li>Tasks</li>
750                <li>About</li>
751            </ul>
752        </div>";
753
754        return $pageFooterHeaderFooterFooter;
755    }
756
757    /**
758     * @reference | 0 overrides | prototype
759     */
760    protected function GeneratePageFooterFooterHeaderHeader(): string
761    {
762        $pageFooterFooterHeaderHeader = "<div class='page-footer-footer-header-header'>
763            <ul>
764                <li>Home</li>
765                <li>Tasks</li>
766                <li>About</li>
767            </ul>
768        </div>";
769
770        return $pageFooterFooterHeaderHeader;
771    }
772
773    /**
774     * @reference | 0 overrides | prototype
775     */
776    protected function GeneratePageFooterFooterHeaderFooter(): string
777    {
778        $pageFooterFooterHeaderFooter = "<div class='page-footer-footer-header-footer'>
779            <ul>
780                <li>Home</li>
781                <li>Tasks</li>
782                <li>About</li>
783            </ul>
784        </div>";
785
786        return $pageFooterFooterHeaderFooter;
787    }
788
789    /**
790     * @reference | 0 overrides | prototype
791     */
792    protected function GeneratePageFooterFooterFooterHeader(): string
793    {
794        $pageFooterFooterFooterHeader = "<div class='page-footer-footer-footer-header'>
795            <ul>
796                <li>Home</li>
797                <li>Tasks</li>
798                <li>About</li>
799            </ul>
800        </div>";
801
802        return $pageFooterFooterFooterHeader;
803    }
804
805    /**
806     * @reference | 0 overrides | prototype
807     */
808    protected function GeneratePageFooterFooterFooterFooter(): string
809    {
810        $pageFooterFooterFooterFooter = "<div class='page-footer-footer-footer-footer'>
811            <ul>
812                <li>Home</li>
813                <li>Tasks</li>
814                <li>About</li>
815            </ul>
816        </div>";
817
818        return $pageFooterFooterFooterFooter;
819    }
820
821    /**
822     * @reference | 0 overrides | prototype
823     */
824    protected function GeneratePageHeaderContentHeaderHeaderHeader(): string
825    {
826        $pageHeaderContentHeaderHeaderHeader = "<div class='page-header-content-header-header-header'>
827            <ul>
828                <li>Home</li>
829                <li>Tasks</li>
830                <li>About</li>
831            </ul>
832        </div>";
833
834        return $pageHeaderContentHeaderHeaderHeader;
835    }
836
837    /**
838     * @reference | 0 overrides | prototype
839     */
840    protected function GeneratePageHeaderContentHeaderHeaderFooter(): string
841    {
842        $pageHeaderContentHeaderHeaderFooter = "<div class='page-header-content-header-header-footer'>
843            <ul>
844                <li>Home</li>
845                <li>Tasks</li>
846                <li>About</li>
847            </ul>
848        </div>";
849
850        return $pageHeaderContentHeaderHeaderFooter;
851    }
852
853    /**
854     * @reference | 0 overrides | prototype
855     */
856    protected function GeneratePageHeaderContentHeaderFooterHeader(): string
857    {
858        $pageHeaderContentHeaderFooterHeader = "<div class='page-header-content-header-footer-header'>
859            <ul>
860                <li>Home</li>
861                <li>Tasks</li>
862                <li>About</li>
863            </ul>
864        </div>";
865
866        return $pageHeaderContentHeaderFooterHeader;
867    }
868
869    /**
870     * @reference | 0 overrides | prototype
871     */
872    protected function GeneratePageHeaderContentHeaderFooterFooter(): string
873    {
874        $pageHeaderContentHeaderFooterFooter = "<div class='page-header-content-header-footer-footer'>
875            <ul>
876                <li>Home</li>
877                <li>Tasks</li>
878                <li>About</li>
879            </ul>
880        </div>";
881
882        return $pageHeaderContentHeaderFooterFooter;
883    }
884
885    /**
886     * @reference | 0 overrides | prototype
887     */
888    protected function GeneratePageHeaderContentFooterHeaderHeader(): string
889    {
890        $pageHeaderContentFooterHeaderHeader = "<div class='page-header-content-footer-header-header'>
891            <ul>
892                <li>Home</li>
893                <li>Tasks</li>
894                <li>About</li>
895            </ul>
896        </div>";
897
898        return $pageHeaderContentFooterHeaderHeader;
899    }
900
901    /**
902     * @reference | 0 overrides | prototype
903     */
904    protected function GeneratePageHeaderContentFooterHeaderFooter(): string
905    {
906        $pageHeaderContentFooterHeaderFooter = "<div class='page-header-content-footer-header-footer'>
907            <ul>
908                <li>Home</li>
909                <li>Tasks</li>
910                <li>About</li>
911            </ul>
912        </div>";
913
914        return $pageHeaderContentFooterHeaderFooter;
915    }
916
917    /**
918     * @reference | 0 overrides | prototype
919     */
920    protected function GeneratePageHeaderContentFooterFooterHeader(): string
921    {
922        $pageHeaderContentFooterFooterHeader = "<div class='page-header-content-footer-footer-header'>
923            <ul>
924                <li>Home</li>
925                <li>Tasks</li>
926                <li>About</li>
927            </ul>
928        </div>";
929
930        return $pageHeaderContentFooterFooterHeader;
931    }
932
933    /**
934     * @reference | 0 overrides | prototype
935     */
936    protected function GeneratePageHeaderContentFooterFooterFooter(): string
937    {
938        $pageHeaderContentFooterFooterFooter = "<div class='page-header-content-footer-footer-footer'>
939            <ul>
940                <li>Home</li>
941                <li>Tasks</li>
942                <li>About</li>
943            </ul>
944        </div>";
945
946        return $pageHeaderContentFooterFooterFooter;
947    }
948
949    /**
950     * @reference | 0 overrides | prototype
951     */
952    protected function GeneratePageFooterContentHeaderHeaderHeader(): string
953    {
954        $pageFooterContentHeaderHeaderHeader = "<div class='page-footer-content-header-header-header'>
955            <ul>
956                <li>Home</li>
957                <li&gt
```

```
Controllers > Pages > Home > ControllerSettings.php > ...
13  class ControllerSettings
14 { 1
15   |   6 references
16   |   private SettingsPage $page;
17   |
18   |   2 references
19   |   private I_UserService $userService;
20   |
21   |   /**
22   |   * The class constructor
23   |   *
24   |   * @param I_UserService $userService The user's service
25   |   */
26   |   0 references|0 overrides
27   |   public function __construct(I_UserService $userService)
28   |   {
29   |   |   $this->page = new SettingsPage();
30   |   |   $this->userService = $userService;
31   |   }
32   |
33   |   /**
34   |   * Method showing the settings page
35   |   */
36   |   3 references|0 overrides
37   |   public function index(): void
38   |   {
39   |   |   echo $this->page->GeneratePage();
40   |
41   |   /**
42   |   * Method to change a password
43   |   *
44   |   * @param string $login The user's login
45   |   * @param string $password The user's password
46   |   * @param string $newPassword The new user's password
47   |   */
48   |   1 reference|0 overrides
49   |   public function ChangePassword(string $password, string $newPassword, string $confirmPassword): void
50   |   {
51   |   |   $this->userService->ChangePassword(login: $_SESSION['username'], password: $password, newPassword: $newPassword, confirmPassword: $confirmPassword);
52   |   |   $this->page->setMsg(msg: new Message(message: 'SuccessPassChange', error: false));
53   |   |   echo $this->page->GeneratePage();
54   |
55   |   /**
56   |   * Method to change the style
57   |   *
58   |   * @param string $styleName The user's style choice
59   |   */
60   |   1 reference|0 overrides
61   |   public function ChangeDisplay(string $styleName): void
```

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
C.2.4 Le code est dans la langue spécifiée	Visual Studio Code	On vérifie que le code du projet entier est écrit en une même langue	BP – Le code entier est en anglais, jusque dans sa documentation et ses commentaires	Le code est toujours en anglais, y compris dans sa documentation
C.2.5 La taille du code est correcte	Visual Studio Code	Vérification que chaque fonction fait moins de 30 lignes et qu'une classe/un fichier fasse moins de 300 lignes	CF – Chaque fichier fait effectivement moins de 300 lignes mais beaucoup de fonctions dépassent les 30 lignes (voir figure ci-dessous), refactoring et séparation des responsabilités nécessaires	Aux classes de visuel près, chaque classe fait en effet moins de 300 lignes pour la plupart (d'autres pouvant encore être redéfinies en plusieurs autres, mais n'ayant pas pu l'être par manque de temps, de même pour différentes fonctions)

```

11
12  /** * The controller for the website's settings */
13  14 references | 0 implementations
13  ✓ class ControllerSettings
14  {
15      6 references
15  |     private SettingsPage $page;
16      2 references
16  |     private I_UserService $userService;
17
18  >     /**
18  |     0 references | 0 overrides
19  >     public function __construct(I_UserService $userService) ...
19  |
20
21  >     /**
21  |     3 references | 0 overrides
22  >     public function index(): void...
22  |
23
24  >     /**
24  |     1 reference | 0 overrides
25  >     public function ChangePassword(string $password, string $newPassword, string $confirmPassword): void...
25  |
26
27  >     /**
27  |     1 reference | 0 overrides
28  >     public function ChangeDisplay(string $styleName): void...
28  |
29
30
31  }
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69

```

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
C.2.6 Le code ne présente pas d'élément redondant	Visual Studio Code	Vérification qu'il n'existe pas 2 classes ou fonctions faisant la même chose (principe DRY), vérifier l'usage d'héritage ou d'interface dans ce cas-là	NC – Les récupérations en BDD depuis du code PHP pourraient être factorisées en un même appel sécurisé	Les différentes récupérations en BDD s'opèrent désormais dans une couche DAO, où des requêtes ont pu être factorisées en une, notamment pour ce qui est des tâches

```

1 reference | 0 overrides
17 public function ListTask(User $user, string $sort, string $sortDirection) : array {
18
19     // Verification of the sort parameters
20     $verifiedSorting = $this->verifySorting(sort: $sort, sortDirection: $sortDirection);
21
22     // Request set-up
23     $request = "SELECT T.id AS id, T.title AS title, date, note, url, done, idProject, idSection, idContext, P.title AS project_title, S.title AS section_title
24         FROM TASK AS T
25         JOIN PROJECT AS P ON P.id = T.idProject
26         JOIN CONTEXT AS C ON C.id = T.idContext
27         JOIN SECTION AS S ON S.id = T.idSection
28         WHERE T.idUser=:idUser
29         ORDER BY $verifiedSorting";
30
31     $parameters = array(
32         "idUser" => $user->getId()
33     );
34
35     // Request execution
36     $response = $this->execRequest(sql: $request, params: $parameters);
37
38     if ($response === false) {
39         throw new Exception(message: "Error while fetching the list of tasks");
40     }
41
42     $tasks = $this->setTaskList(response: $response);
43
44     return $tasks;
45 }

```

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
C.2.7 La norme de casse du langage utilisé est respectée	Visual Studio Code, Webimpress Coding Standard	Le camel case est respecté (attributs en minuscule, propriété en majuscule au début, pas d'underscores (ils peuvent être remplacé par une majuscule dans le mot), constantes en majuscules)	NC – Par analyse Webimpress Coding Standard, on détecte déjà presque 600 erreurs de camel case dans le code	N'ayant pas nécessairement suivi toutes les bonnes pratiques durant ce projet, il restera des erreurs de camel case, bien qu'elles soient moins nombreuses que dans le projet original

```

analyse.txt
108
142 | 23 | [ss]31mERROR[0m | [x] Equals sign not aligned correctly; expected 1 space but found
143 |   |   0 spaces
144 | 23 | [ss]31mERROR[0m | [x] Expected 1 space before "="; 0 found
145 | 23 | [ss]31mERROR[0m | [x] Expected 1 space after "="; 0 found
146 | 25 | [ss]31mERROR[0m | [x] Whitespace found at end of line
147 | 28 | [ss]31mERROR[0m | [x] Missing blank line before comment tags
148 | 28 | [ss]31mERROR[0m | [x] Param tag is redundant
149 | 29 | [ss]31mERROR[0m | [x] Return tag with "void" type is redundant
150 | 31 | [ss]31mERROR[0m | [ ] Public method name
151 |   |   "ControllerAddUpdateContext::UpdateContextPage" is not in
152 |   |   camel caps format
153 | 31 | [ss]31mERROR[0m | [x] There must be exactly 1 space(s) between the closing
154 |   |   parenthesis and the colon when declaring a return type for a
155 |   |   function
156 | 31 | [ss]31mERROR[0m | [x] Opening brace should be on a new line
157 | 32 | [ss]31mERROR[0m | [x] Expected 1 space before "="; 0 found
158 | 32 | [ss]31mERROR[0m | [x] Expected 1 space after "="; 0 found
159 | 32 | [ss]31mERROR[0m | [x] Equals sign not aligned correctly; expected 1 space but found
160 |   |   0 spaces

```

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
D – Principes de qualité logicielle				
D.1 – Principes SOLID				
D.1.1 Le principe de responsabilité unique (S) est respecté	Vscode	On vérifie si chaque fichier fait une seule tache, grâce à ça il serait plus simple de maintenir l'application dans le futur (Une classe, une responsabilité. On cherche différentes classes faisant la même chose ou une faisant plusieurs choses à la fois)	CR - Le fichier display_type.php ne respecte pas le principe S pour la plupart des fichiers les appels à la base de données et la gestion de l'affichage sont mélangés dans le code ce qui est contraire au principe S : exemple ci-dessous	Une architecture MVC est en place, veillant à ce que chaque fichier ait une responsabilité unique. L'ensemble est structuré avec des classes PHP, ce qui permet de respecter le principe S. De plus, une couche service servira de passerelle entre les couches DAO et Controller, permettant de décharger les responsabilités de l'un ou de l'autre avant d'interagir avec le serveur. (Cf architecture du projet pour plus de détails)
D.1.2 Le principe ouvert-fermé (O) est respecté	Vs code	On vérifie s'il y a une isolation des couches et si pour ajouter une fonctionnalité on peut	CR - il n'y a aucune interface ni de séparation des packages. Si on veut ajouter une fonctionnalité, on est obligé de modifier du code de partout pour que ça marche.	La partie métier est isolée à l'aide d'interfaces, ce qui permet d'ajouter de nouvelles fonctionnalités facilement, sans impacter le reste du système. Ceci permet de respecter le principe O. Une bibliothèque d'injection de dépendances (PHP-DI) est également installée afin de pouvoir facilement remplacer un service ou un DAO à l'avenir.

```

24
25     // Configuration of the dependecies injection
26     return [
27
28         // DAO
29         I_TaskDAO::class => autowire(className: TaskDAO::class),
30         I_ProjectDAO::class => autowire(className: ProjectDAO::class),
31         I_SectionDAO::class => autowire(className: SectionDAO::class),
32         I_ContextDAO::class => autowire(className: ContextDAO::class),
33         I_UserDAO::class => autowire(className: UserDAO::class),
34
35         // Services
36         I_TaskService::class => autowire(className: TaskService::class),
37         I_ContextService::class => autowire(className: ContextService::class),
38         I_SectionService::class => autowire(className: SectionService::class),
39         I_ProjectService::class => autowire(className: ProjectService::class),
40         I_UserService::class => autowire(className: UserService::class)
41
42     ];
43
44
45     ?>

```

D.1.3	Le principe de substitution de Liskov (L) est respecté	Visual Studio Code	Vérification qu'en cas d'héritage, le module de bas peut remplacer celui de haut niveau sans en changer le fonctionnement (exemple des exceptions levées par les classes filles)	CR – Pas d'objets donc à mettre en place dans le projet	La transformation du projet pour utiliser des classes PHP aurait pu poser un problème vis-à-vis du principe L. Cependant, ce principe SOLID a bien été pris en compte afin d'éviter toute erreur à ce niveau. On retrouve particulièrement des notions d'héritage du côté des vues (ListTaskBySection héritant de ListTask par exemple) et l'un peut aisément remplacer l'autre
-------	--	--------------------	--	---	---

```

10      6 references | 3 implementations
11  < class ListTaskPage extends AbstractPage {
12
13    > #region Attributes ...
14
15    > /**
16     * ...
17     */
18    > public function setTasks(array $tasks): void ...
19
20    > /**
21     * ...
22     */
23    > public function __construct() ...
24
25
26    > /**
27     * ...
28     */
29    > protected function GenerateContent(): string
30    {
31        $returnValue = "<main class='main-content'>";
32
33        $returnValue .= $this->GenerateForm();
34
35        if (empty($this->tasks)) {
36            $returnValue .= "<p class='empty-message'>" . $this->languageManager->translate(key: "NoTasksFound") . "</p>";
37        } else {
38            $returnValue .= "<div class='item-list'>";
39            foreach ($this->tasks as $task) {
40                $title = htmlspecialchars(string: $task->getTitle());
41                $date = $task->getDate() ? $task->getDate()->format(format: "Y-m-d") : "N/A";
42
43                $returnValue .= "<div class='item'>" . $title . " ({$date})" . "</div>";
44            }
45        }
46    }
47
48    > /**
49     * ...
50     */
51    > protected function GenerateForm(): string { ...
52
53

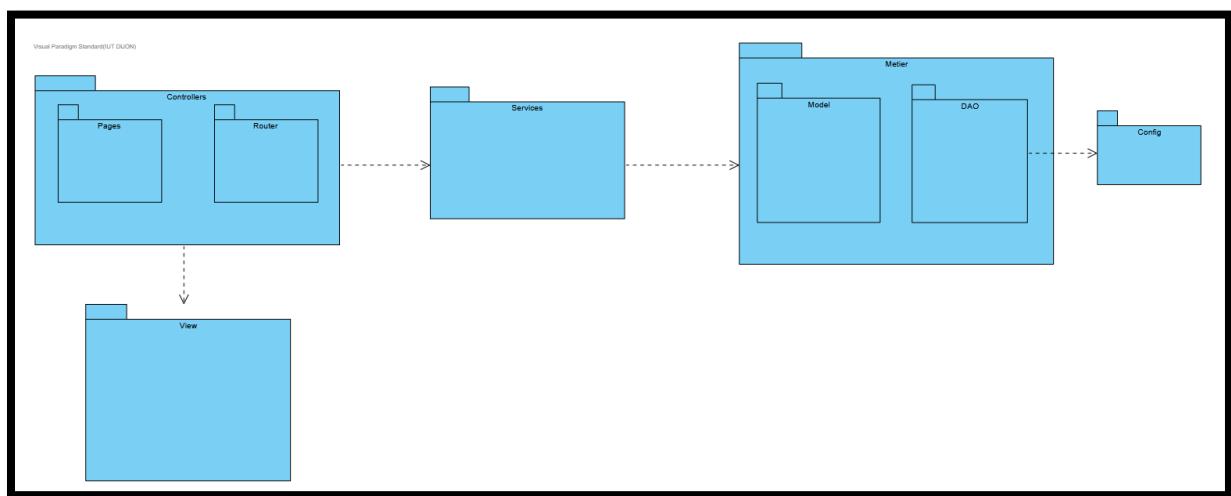
```

```

6
7  > /**
8   * Page to list the tasks by context
9   */
10 > 3 references | 0 implementations
11 < class ListTaskByContextPage extends ListTaskPage {
12
13    > 3 references
14    > private Context $context;
15
16    > /**
17     * Set the value of context
18     */
19    > 1 reference | 0 overrides
20    > public function setContext(Context $context): void
21    {
22        $this->context = $context;
23    }
24
25
26    > 1 reference | 0 overrides | prototype
27    > protected function GenerateForm(): string { ...
28
29
30
31
32
33
34
35
36
37
38
39
39
40
41
42
43
44
45
46
47
48
49
49
50
51
52
53

```

D.1.4 Le principe de ségrégation des interfaces (I) est respecté	Visual Studio Code	Une classe n'implémente pas une interface dont elle n'a pas besoin de toutes les méthodes, éviter les interfaces pour tout, qui ont trop de méthodes (principe S)	CR – Pas d'objets donc à mettre en place dans le projet	L'ajout d'interface aurait pu poser souci quant au respect du principe I mais nous l'avons bien respecté, comme démontré sur le point D.1.2. sur le principe O
D.1.5 Le principe d'inversion des dépendances (D) est respecté		(Code de haut niveau ne dépend pas de bas niveau (métier qui parle à IHM directement))	CR – Pas d'objets donc à mettre en place dans le projet	<p>Le principe D garantit que les dépendances entre classes vont dans le bon sens et qu'il n'existe pas de dépendances circulaires.</p> <p>Ce point a également été pris en compte dans notre conception (cf. diagramme des classes), ayant une claire séparation des couches : Controllers a accès aux Services qui ont accès aux DAO et aux Models, Controllers ont également accès aux Vues qui ont accès aux Models.</p> <p>Cette séparation des couches évite donc des dépendances circulaires</p>



Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
D – Principes de qualité logicielle				
D.2 – Gestion des erreurs				
D.2.1 – Les cas d'erreurs sont traités à l'aide des exceptions	VsCode	Test via l'IHM et analyse du code pour trouver des erreurs gérées	CR – Aucune présence d'exception	
D.2.2 Le point d'entrée du programme capture les exceptions non encore capturées	VsCode	Test via l'IHM et analyse du code pour trouver des erreurs gérées et trouver des blocs de try catch	CR – Vu qu'il n'y a aucune exception les seules erreurs affichées viennent d'if/else	Les exceptions sont directement capturées dans le router de sorte que l'on n'ait aucune exception n'étant pas gérée dans l'application
D.2.3 Les exceptions ne sont pas muselées mais capturées pour être traitées (catchs vides, à documenter si c'est volontaire)	VsCode	Test via l'IHM et analyse du code	CR - Aucune présences de try / catch sauf dans la bibliothèque	

TaskStep

Nom d'utilisateur ou mot de passe incorrect.
Veuillez réessayer

Nom d'utilisateur
Entrez votre identifiant

mot de passe
Entrez votre mot de passe

Connexion

Pas encore de compte ? [Créer un compte](#)

Page d'erreur

16/05/2025

Paramètres Aide Déconnexion

Accueil Tous les éléments Par Contexte Par Projet

Ajouter une tâche

- Immédiat
- Idées
- Achats potentiels
- Cette semaine
- Ce mois-ci
- Cette année

ERREUR :

No returned tasks

Veuillez retourner sur la page précédente

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
D – Principes de qualité logicielle				
D.3 – Règles de codage				
D.3.1 – Il n'y a pas de « valeur magique » dans le code	Visual Studio Code	Il n'y a pas de valeurs arbitraires sorties de nulle part dans le code (limites de boucles par exemple)	CF – Une valeur magique se trouve au milieu des fonctions (display_frontpage()) et devrait donc être échappée en dehors de la fonction (variable globale par exemple) ou en paramètre optionnel avec valeur par défaut (voir figures ci-dessous)	Cette valeur magique n'existe maintenant plus et les variables globales de l'application sont contenues dans un fichier séparé pour l'application

```
Defines.php > ...
1  <?php
2
3  #region Connexion
4
5  // Time to wait before trying another connexion again
6  define(constant_name: "TENTATIVES_TIMEOUT",value: 600);
7
8  // Number of tries before waiting the time above
9  define(constant_name: "TENTATIVES_CONNEXIONS",value: 5);
10
11 // The time you can remain connected
12 define(constant_name: "INACTIVITY_TIMEOUT",value: 1200);
13
14 #endregion
15
16
```

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Optimisation
D.3.2 – chaque fonction ne possède qu'un return, sur la dernière ligne de la fonction	Visual Studio Code	Vérification qu'aucune fonction ne possède plus d'un return, ce dernier devant être placé en dernière ligne de fonction	BP – Aucune fonction ne possède plus d'un return dans le projet	De même, après réarrangement du projet, chaque fonction ne possède qu'un return
D.3.3 – aucune procédure ne contient de return	Visual Studio Code	Vérification qu'il n'y a pas de returns en procédure	BP – Il n'y a pas de returns en procédures du projet	Les fonctions ne renvoyant rien ne possèdent pas de return
D.3.4 – il n'y a aucun « goto » ou autre rupture du flux d'exécution	Visual Studio Code	Vérification qu'aucune rupture du flux n'a lieu dans le projet, qu'il n'y a pas de goto renvoyant à une autre ligne de code en particulier, ni de break en milieu de fonction	BP – Il n'y a pas de rupture du flux	Aucune rupture de flux n'a été implémentée du projet

D.3.5 – Le nom des variables est pertinent	Visual Studio Code	On vérifie que les noms donnés sont pertinents comme les noms de variable ou autre	NC – Certains noms ne sont absolument pas pertinents et sont non professionnels parfois par manque d'inspiration le nom est le même avec un numéro ce qui rend le code encore plus illisible	Chaque variable a été nommée de façon cohérente et logique avec l'application
D.3.6 – le nom des opérations (fonctions, procédures) est pertinent	Visual studio code	Vérification que les noms des fonctions est pertinent et clair	NC – Certains noms de fonction ne sont pas très clair et ne possède pas de nomenclature	De même, chaque fonction et procédure est nommée de façon pertinente, les rendant reconnaissables et utilisables

```

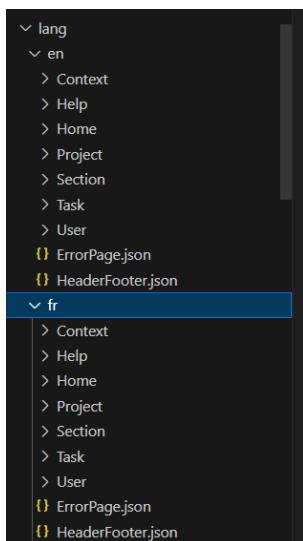
6      18 references | 1 implementation
7  < interface I_SectionService{
8
9 >   /**
10    * ...
11    * 1 reference | 1 override
12    */
13   public function AddSection(string $title);
14
15 >   /**
16    * ...
17    * 1 reference | 1 override
18    */
19   public function UpdateSection(int $id,string $title);
20
21 >   /**
22    * ...
23    * 1 reference | 1 override
24    */
25   public function DeleteSection(int $id);
26
27 >   /**
28    * ...
29    * 3 references | 1 override
30    */
31   public function ListSection():array;
32
33 >   /**
34    * ...
35    */
36   public function GetSection(int $id):Section;
37
38 > }
39 }
```

```

7   2 references | 0 implementations
8 < class ProjectCreateRoute extends Route {
9     3 references
10    /**
11     * constructor
12     * @param \Controllers\Pages\Project\ControllerAddUpdateProject $controllerProject project controller
13     */
14    1 reference | 0 overrides
15    public function __construct(ControllerAddUpdateProject $controllerProject){
16      $this->controllerProject = $controllerProject;
17    }
18    #region methods
19    1 reference | 0 overrides | prototype
20    public function get(array $params=[]): void{
21      $this->controllerProject->AddProjectPage();
22    }
23    1 reference | 0 overrides | prototype
24    public function post(array $params=[]): void{
25      $title = parent::getParam(array: $params,paramName: "title",canBeEmpty: false);
26      $this->controllerProject->AddProject(title: $title);
27    }
28    #endregion
29
30 /**
31  * Abstract class for a route
32 */
33 131 references | 31 implementations
34 abstract class Route {
35
36   /**
37    * Method to act on a road, calling the method GET or POST based on the given parameter
38    * @param array $params Parameters
39    * @param string $method Action method (get by default)
40    * @return void
41    */
42  2 references | 0 overrides
43  public function action(array $params = [], string $method='GET'): void {
44
45    try {
46      switch ($method) {
47        case 'GET':
48          $this->get(params: $params);
49          break;
50
51        case 'POST':
52          $this->post(params: $params);
53          break;
54      }
55    }
56    catch (Exception $e) {
57      $controller = new ControllerError();
58      $controller->index(message: $e->getMessage());
59    }
60  }
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88

```

Critères d'audit	Outils utilisés	Descriptions des vérifications effectuées (tests, conditions de test, etc.)	Résultats de l'audit (constats)	Temps estimé d'optimisation
D.3.7 – il n'y a pas de code « mort » (commenté, non atteignable...) (fonctions jamais appelées, ou alors code qui n'est jamais exécuté du programme)	Visual Studio Code	Vérification qu'il n'y a pas de code commenté ou non atteignable, de fonctions jamais appelées ou de code qui n'est jamais exécuté du programme	CF – Plusieurs zones de code sont commentées	Aucun code mort n'a été laissé dans ce nouveau projet
E – Pile technique				
E.1 – Langage utilisé				
E.1.1 – Le(s) langage(s) utilisés sont pertinents pour le problème donné (Exemple non pertinents : application en Python, IA en C#)	Visual Studio Code, XAMPP	Vérifications que les langages utilisés ne sont pas hors-sujet et sont pertinents à utiliser.	CF – Comme il s'agit d'un site web, le code PHP est adapté dans l'ensemble à de l'interaction côté serveur et à la communication avec une Base de Données Pour la base de données la communication se fait par l'utilisation de MySQL Pour la traduction, quant à elle, elle n'est pas réalisée dans le bon langage, elle est réalisée en PHP alors qu'elle devrait être en .json	Le reste des éléments étant corrects dans l'ensemble, nous avons simplement changé le système de traduction afin qu'il utilise des fichiers .json pour traduire le site en différentes langues



```

10 class LanguageManager {
11     /**
12      * Method to get a traduction from the loaded files
13      * @param string $key The table key
14      * @return string Value associated to the entered key
15     */
16     public function translate(string $key) : string {
17         return htmlspecialchars(string: $this->translations[$key]) ?? htmlspecialchars(string: $key);
18     }
19     /**
20      * Method to get a translation from the loaded files without special chars
21      * @param string $key The table key
22      * @return string Value associated to the entered key
23     */
24     public function translate_without_specialchars(string $key) : string {
25         return $this->translations[$key] ?? $key;
26     }
27 }

```

E.1.2 – La version utilisée du langage est supportée et maintenue (Ex : page en PHP5, failles non corrigées aujourd’hui)	Visual Studio Code, README	Vérification que le langage de programmation est supporté par les navigateurs web et maintenu encore aujourd’hui (lecture du README pour trouver les versions)	CR – La version utilisée est php 7.0 qui n'est plus maintenue, il faut mettre à jour les différentes méthodes n'étant plus d'actualité (PDO pour interactions BDD, POO, ...)	Le code a pu être mis à jour et codé en PHP 8.2, permettant ainsi des interactions plus sécurisées avec la base de données, évitant notamment les injections SQL
E.2 – Outils tiers				
E.2.1 – Les bibliothèques/frameworks utilisés sont pertinents et réellement utiles	Visual Studio Code	Analyse du code, on vérifie que les bibliothèques utilisées servent réellement et n'ont pas un trop grand nombre de fonctionnalité pour un usage trop petit	NC – L'unique bibliothèque utilisée était utile en 2006 mais n'est plus utiles actuellement est pris en charge nativement par HTML depuis 2015 (voir figure ci-dessous)	<p>La bibliothèque des dates en HTML a donc été retirée au profit d'une simple balise <input type="date"></p> <p>Une seule bibliothèque sera désormais utilisée, servant à l'injection de dépendances en PHP (PHP-DI 7.0)</p>
E.2.2 – Les bibliothèques/frameworks utilisés sont dans une version supportée et maintenue	Visual Studio Code	Vérification que la ou les bibliothèques du projet sont encore d'actualité et maintenue(s), si besoin, les mettre à jour	NC – La seule bibliothèque utilisée n'est plus maintenue.	PHP-DI est encore actuellement maintenu et supporté, fonctionnant parfaitement bien avec PHP 8.2

IV. Tableau des tâches d'optimisation réalisées

Le tableau suivant liste les tâches d'optimisation qui étaient définies par l'audit de qualité logicielle, la dernière colonne indiquera en vert si la tâche a pu être réalisée, en orange si elle ne l'est que partiellement, en rouge si elle n'a pas pu l'être

Recommandations	Priorité	Complexité	Effort	gravité	Fait / Non fait
tutoriel hébergement local	1	1	1	1	
changement de langue	2	3	2	2	
tests unitaires	2	3	2	1	
présence de valeur arbitraire	2	2	2	2	
code mort	2	1	1	1	
bibliothèque pour la date n'est plus utile maintenant	2	1	1	1	
bibliothèque non maintenue	2	1	1	1	
mauvaise gestion des cas particulier	3	3	2	2	
erreur et warning dans le code	3	2	3	2	
non typage	3	1	2	2	
indentation	3	1	1	1	
code difficile à lire	3	3	2	2	
non respect du camel case	3	2	2	1	
traduction en PHP au lieu de json	3	3	2	2	
documentation	4	1	2	1	
fonction trop longue	4	2	2	2	
non présence d'exception	4	3	2	3	
non présence de try catch (sauf dans la bibliothèque)	4	2	3	3	
noms impertinents (variables et fonctions)	4	4	3	3	
mettre à jour la version de PHP	4	4	3	3	
diagramme cas d'usage	5	3	2	2	
mélange requête SQL et code PHP	5	3	2	3	
architecture	6	3	3	3	
principe SOLID non respecté	6	6	4	3	

Par manque de temps, nous n'avons pas pu analyser chaque cas particulier du projet et n'avons pas pu résoudre tous les warnings du projet, mais le reste des éléments, notamment tous ceux à priorité très élevée, a pu être corrigé, rendant donc globalement l'application beaucoup plus qualitative en termes de code.

V. Synthèse d'optimisation

L'application a pu être jugée selon les principes SOLID, les bonnes pratiques de programmation et le PSR (PHP Standards Recommandations) durant l'audit de qualité logicielle. En découlait donc les soucis d'architecture, de documentations, de tests ou même de respect des principes évoqués précédemment.

Nous avons donc pris la décision de refaire intégralement l'architecture de l'application étant donné que Taskstep n'avait originellement aucune structure, ni même de programmation orientée objet. Ceci se faisant dans le respect des principes SOLID, en documentant chaque classe, interface et fonction publique, et en respectant chacun des critères décrits durant tout cet audit d'optimisation.

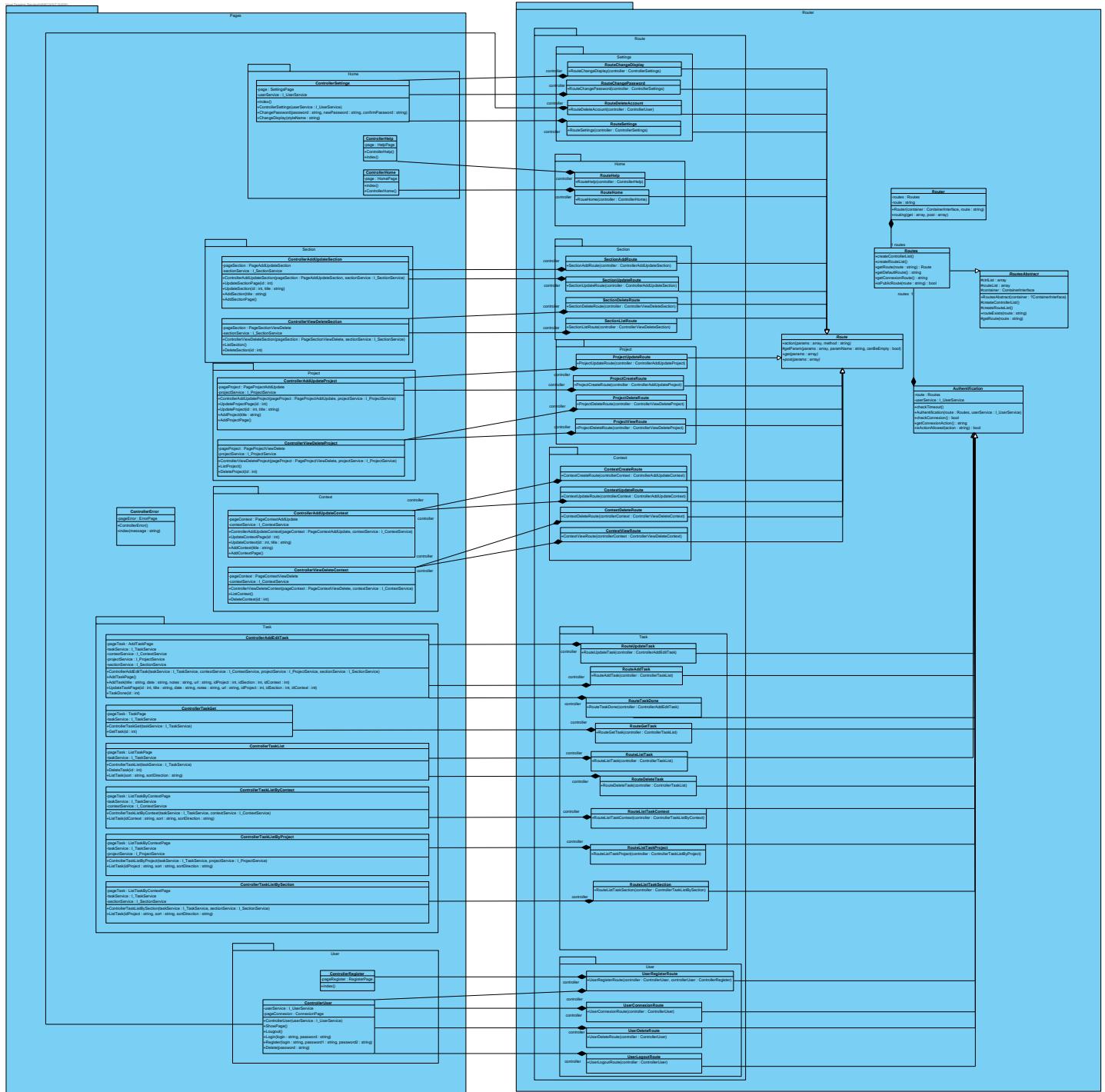
Pour ce qui est de la sécurité et de la gestion des données dans le code, il aura été réalisé de sorte qu'aucune injection SQL ne soit possible. Des tests unitaires de la couche DAO, donc pour les interactions avec la base de données, auront également été réalisés afin de pouvoir tester la validité des tables du projet.

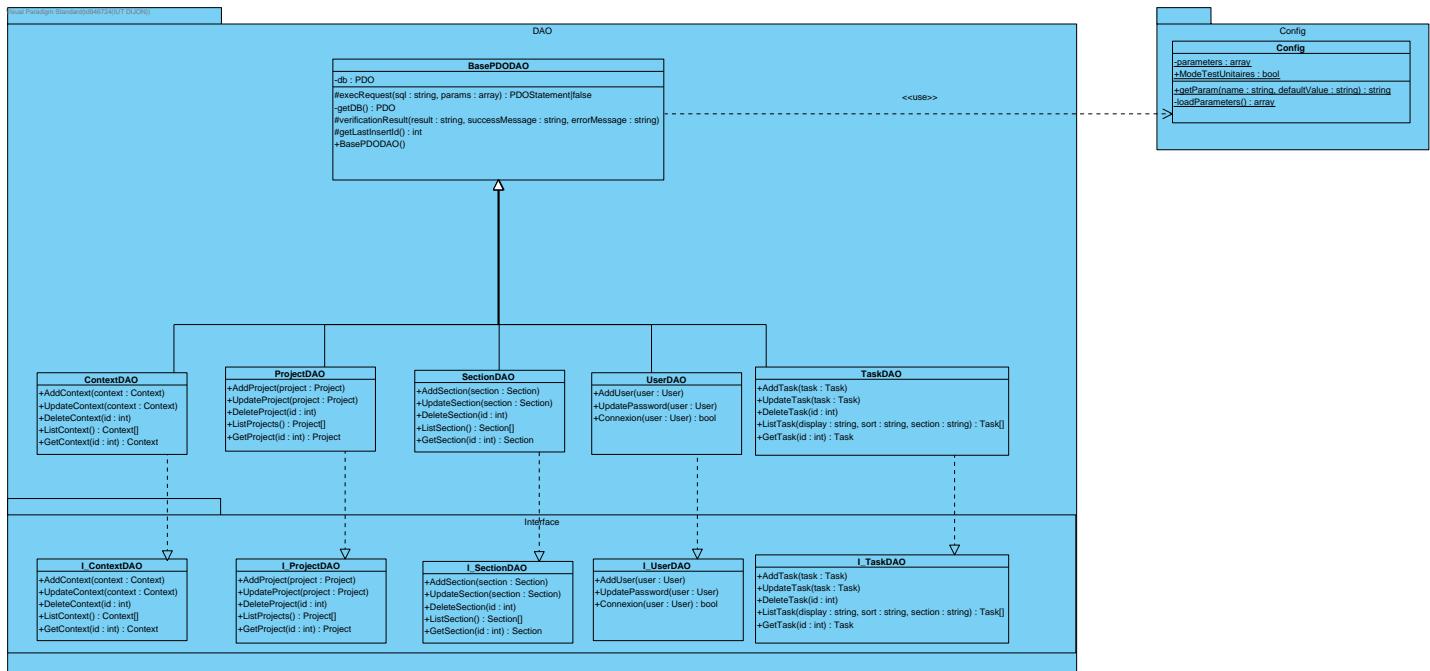
La traduction du site est désormais gérée par différents fichiers JSON, rendant la mise en place d'une nouvelle langue beaucoup plus facile.

Globalement, ces changements permettent de rendre l'application beaucoup plus modulaire et évolutive qu'elle ne l'était auparavant, le respect des bonnes pratiques de programmation en étant en grande partie responsable. D'autres optimisations secondaires n'ont pas pu être finalisées dans les temps (comme la gestion de cas particuliers ou l'implémentation de tests unitaires hors-couche DAO) et pourraient être réalisés lors d'une poursuite du projet.

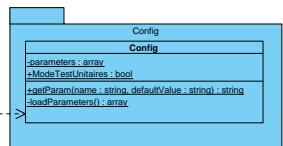
VI. Annexe : UML du projet

Ci-dessous se trouvera le nouvel UML du projet dans son intégralité





<<use>>



DAO

Interface

