DSL+PVL+DCP

Réalisé par : *Julan HOUEHOU et Wenfeng GAO*

Table des matières

[I- Dossier de Spécifications Logiciel 2](#_Toc414023149)

[Introduction : 2](#_Toc414023150)

[A- Objectifs du projet : 2](#_Toc414023151)

[B- Fonctionnalités principales 2](#_Toc414023152)

[C- Futurs utilisateurs 3](#_Toc414023153)

[D- Définition des IHM 4](#_Toc414023154)

[E- Contraintes mémoire 4](#_Toc414023155)

[F- Fonctionnalités détaillées 4](#_Toc414023156)

[G- Données à stocker 5](#_Toc414023157)

[H- Temps de réponse 5](#_Toc414023158)

[I- Délais 6](#_Toc414023159)

[J- Coûts 6](#_Toc414023160)

[II- Plan de Validation Logiciel 7](#_Toc414023161)

[III- Dossier de Conception Préliminaire 10](#_Toc414023162)

## Dossier de Spécifications Logiciel

# Introduction :

Le dossier de spécification renferme la description complète du logiciel et servira de base pour une validation détaillée. L’accent sera mis sur le processus de développement, ce qui permettra une vérification des prototypes développés en regard des spécifications proposées et acceptées.

# Objectifs du projet :

L’objectif de ce projet est de créer une plateforme de certification pour JAVA.

# Fonctionnalités principales

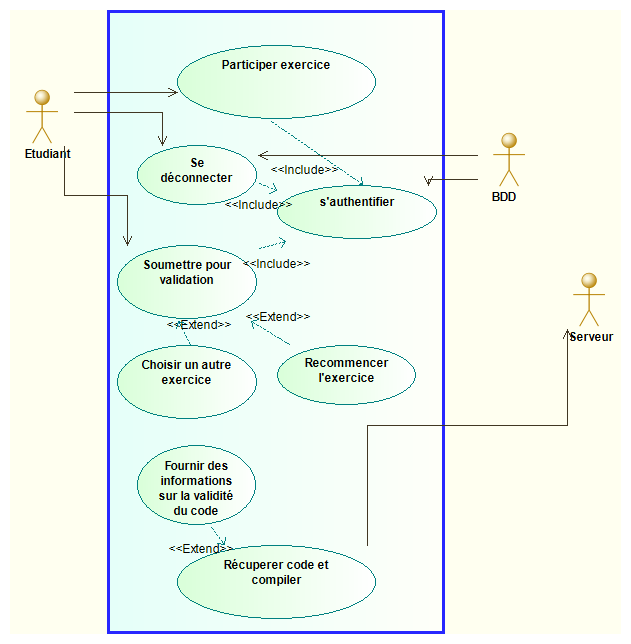
L’outil permettra de diffuser des exercices à des étudiants. L’étudiant complètera la partie du programme manquante puis soumettra son programme à une correction.

Un test unitaire sera alors exécuté sur le serveur afin de valider ou non le programme.

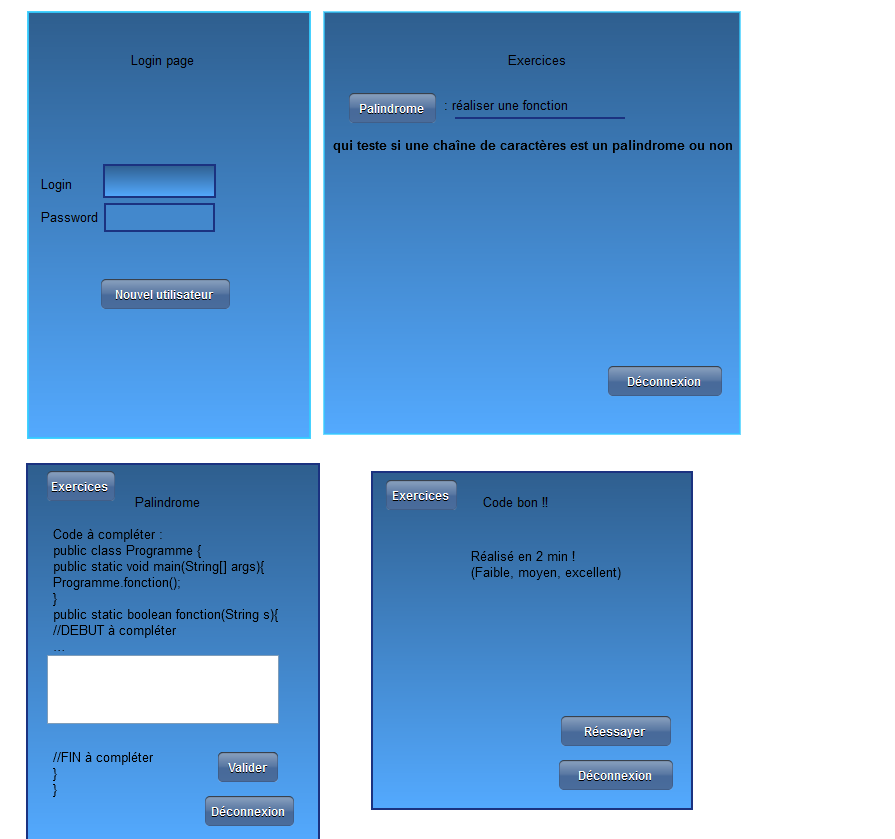
Un retour sera ensuite fait à l’étudiant : soit OK, soit une liste d’erreurs.

# Futurs utilisateurs

Les futurs utilisateurs de l’application sont des étudiants dans l’optique d’avoir une idée de leur niveau en Java.



# Définition des IHM



# Contraintes mémoire

Le programme doit fonctionner sur les serveurs de l’école.

# Fonctionnalités détaillées

* Réaliser un formulaire d’inscription (login, mot de passe en s’assurant qu’ils ne soient pas déjà utilisés).
* Réaliser un formulaire de connexion (un utilisateur et mot de passe correspondant déjà dans la base de données).
* Possibilité pour l’utilisateur de rejoindre un exercice (deux ou trois maximum).
* L’application récupère le code saisi par l’utilisateur (après appui sur le bouton valider par l’utilisateur) sur un serveur et le compile.
* Si le code saisi comporte des erreurs :

Il faut lui afficher les différentes erreurs.

* Si le code ne comporte pas d’erreurs :

Pour le cas du palindrome par exemple, tester le code avec deux ou trois mots( un palindrome et deux qui ne le sont pas ) et lui afficher un message : « OK, code valide » ou « NOK, le code ne marche pas à tous les coups .

Si « OK, code valide », lui afficher également le temps mis pour réaliser l’exercice en lui indiquant (« Temps faible », « temps moyen », temps excellent »). La comparaison sera effectuée par rapport au temps des deux utilisateurs présents dans la base de données**. (A noter que cette fonctionnalité n’est pas indispensable. Elle ne représente qu’un plus si les précédentes sont effectuées correctement).**

* L’utilisateur peut éventuellement recommencer l’exercice via un bouton recommencer, il peut en choisir un autre via un bouton Retour (au choix des exercices) ou il peut se déconnecter.

# Données à stocker

Les différentes données à stocker sont : le login, le mot de passe, deux(ou trois maximum) exercices, et éventuellement le temps mis pour réaliser un exercice.

# Temps de réponse

Pour se connecter : il faut moins d’une seconde pour accéder à la page des exercices.

Pour traiter le code : il faut moins de deux secondes.

# Délais

Confère EDT (La dernière séance du module Java/ JEE)

# Coûts

Le projet devant être réalisé en une durée de 30 h, le coût total est de 6429€ TTC.

## Plan de Validation Logiciel

Avant-propos

Ce document a été conçu dans le cadre du projet JavaEE par un binôme d’étudiants en deuxième année. Il doit servir de support pour la validation du logiciel lors de la recette auprès du client.

|  |  |
| --- | --- |
| **Numéro de test** : | 1 |
| **Titre :** Procédure d’inscription |  |
| **Objectif :** Vérifier que la procédure d’inscription se fait suivant le cahier des charges | | |

|  |
| --- |
| **Procédure de test** |
| Initialisation : Lancement du logiciel |

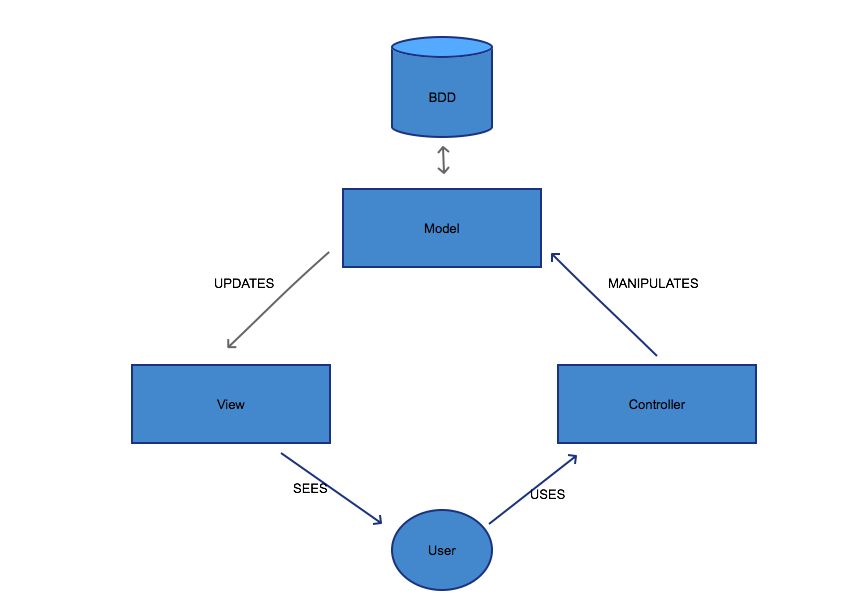
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Données d’entrée(Action)** | **Comportement attendu** | **OK ?** | **N OK ?** |
| **1** | Double cliquer sur l’icône de l’application | L’application se lance |  |  |
| **2** | L’utilisateur clique sur nouvel utilisateur | L’utilisateur peut remplir les champs( login , mdp) Si un login ou un mdp est déjà utilisé, l’utilisateur est convié à le modifier via un message qui apparaît à l’écran.  Si un champ n’est pas rempli, l’utilisateur est convié à le remplir via un message d’erreur. |  |  |
| **3** | L’utilisateur clique sur valider ou clique sur annuler | S’il clique sur valider, son login et mdp apparaissent dans la BDD  S’il clique sur annuler, une page de création d’utilisateur est ré-ouverte |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Numéro de test** : | | | 2 | | |
| **Titre :** Procédure de connexion | | |  | | |
| **Objectif :** Vérifier que la procédure de connexion se fait suivant le cahier des charges | | | | | | |
| **Id** | **Données d’entrée(Action)** | **Comportement attendu** | | **OK ?** | **N OK ?** | | |
| **1** | Double cliquer sur l’icône de l’application | L’application se lance | |  |  | | |
| **2** | L’utilisateur remplit les champs ( login et mdp) et clique valider | Si un champ n’est pas rempli ou que le login ou mdp ne sont pas bons, un message d’erreur le signale.  En cliquant Valider, il arrive sur une fenêtre avec les titres des exercices et une brève description | |  |  | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Numéro de test** : | | | 3 | | |
| **Titre :** Procédure de choix des exercices | | |  | | |
| **Objectif :** Vérifier que la procédure de choix des exercices se fait suivant le cahier des charges | | | | | | |
| **Id** | **Données d’entrée(Action)** | **Comportement attendu** | | **OK ?** | **N OK ?** | | |
| 1 | L’utilisateur clique sur un exercice | L’exercice débute, le code à compléter apparait, la zone de saisie pour l’utilisateur apparaît encadré par **«  //Début à compléter » et « //Fin à compléter »**  **S’il clique valider sans remplir le champ libre, un message lui apparaît  « Vous n’avez pas résolu l’exercice »** | |  |  | | |
| 2 | L’utilisateur rempli le champ vide et clique valider | Un message s’affiche : « êtes-vous sûr ? »  Oui ou non ? | |  |  | | |
| 3 | L’utilisateur clique  non | La fenêtre disparaît et il peut continuer son exercice | |  |  | | |
| 4 | L’utilisateur clique oui | Après quelques instants, ses erreurs lui sont affichées si son code est erroné, ou si le code ne comporte pas d’erreur, le message suivant lui est affiché par une fenêtre « OK, code valide » ou « NOK, le code ne marche pas à tous les coups. » Le programme affiche le temps mis pour réaliser l’exercice(Optionnel) | |  |  | | |
| 5 | L’utilisateur clique exercices | Il est ramené sur la page de choix des exercices | |  |  | | |
| 6 | L’utilisateur clique réessayer | Il est ramené sur la page de l’exercice | |  |  | | |
| 7 | L’utilisateur clique déconnexion | Il est ramené sur la page de connexion | |  |  | | |

|  |  |  |
| --- | --- | --- |
| Rapport de test | Testé par | Commentaire |
| * Excellent * Bon * Moyen * Faible |  |  |

## Dossier de Conception Préliminaire

****

Model : Le modèle représente le cœur (algorithmique) de l'application : traitements des données, interactions avec la base de données, etc. Il décrit les données manipulées par l'application.

View : Ce avec quoi l'utilisateur interagit se nomme précisément la vue(interface). Sa première tâche est de présenter les résultats renvoyés par le modèle. Sa seconde tâche est de recevoir toute action de l'utilisateur. Ces différents événements sont envoyés au contrôleur.

La base de données enregistre les différentes informations.