République Algérienne Démocratique et Populaire

Université Abou Bakr Belkaid– Tlemcen

Faculté des Sciences

Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en informatique

Option : Réseaux et Système Distribue

Thème :

Mise en place d'une application de gestion des tests en ligne

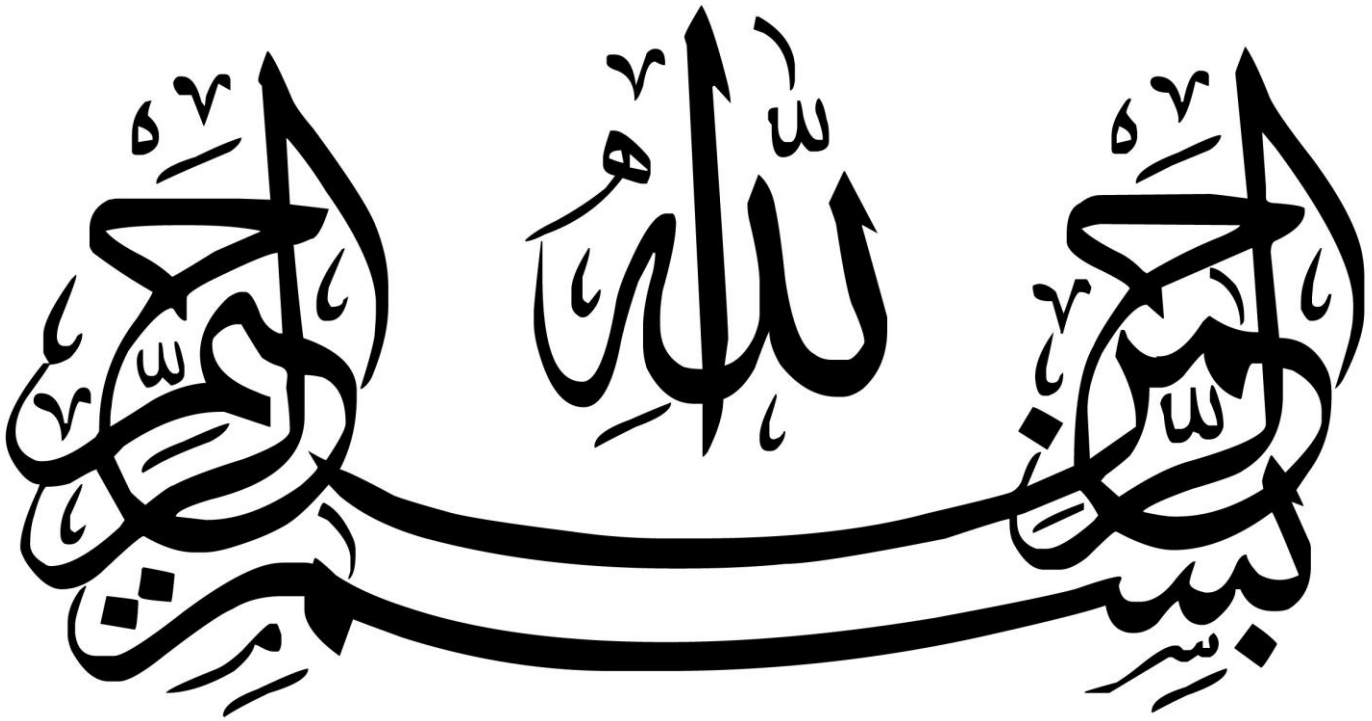
Réalisé par :

- **MERAHI Aissa Nadir**

Présenté le 19-12-2017 devant la commission du jury :

❖ **Mr BEKKARA Chakib**	**Président**
❖ **Mme AMRAOUI Asma**	**Examinateur**
❖ **Mr BELHOCINE Amine**	**Encadreur**

Année universitaire 2016-2017



Dédicace

On dédie ce modeste travail avec respect

A mes chers parents qui ont sacrifié tout ce qu'ils ont de plus cher pour moi, et mes études.

A mes sœurs et frères pour leurs encouragements.

A tous mes amis pour leurs soutiens.

A tous mes collègues de promo informatique pour leurs collaborations.

A mes professeurs durant mes enseignement et formation.

A tous les membres du jury présent à mon soutenance.

²A tous ceux qui ont participé de près ou de loin à la réalisation de mon travail.

Remerciements

Avant d'entamer mon présent rapport, je remercie Dieu le Tout Puissant de nous avoir donné le courage et l'opportunité de réaliser ce travail.

Nous tenons à adresser nos sincères remerciements à l'ensemble du corps professoral), plus particulièrement ceux qui nous ont assistés pour la réalisation de ce projet de fin de deuxième année.

Ensuite, toutes nos pensées de gratitude se dirigent vers pour avoir bien voulu encadrer mon projet, pour l'aide et les renseignements qu'il m'a fourni.

Que toute personne ayant contribué de près ou de loin à la réalisation de mon projet, trouve ici l'expression de nos sincères sentiments.

Enfin, veuillez accepter Madame, Monsieur les membres du jury, toute notre reconnaissance.

Résumé

Le présent document constitue une synthèse de mon projet de fin d'étude.

Ce projet a pour objectif la création d'une application qui a comme but la gestion des tests en ligne. En outre, et vu son côté dynamique elle doit être performante dans la gestion des mises à jour sur les données de la plateforme. De là, on souhaite que ce projet atteigne ces buts.

Dans le présent rapport, je décrire en détails les différentes étapes de développement du projet. Dans le premier chapitre, j'aborderai le contexte et motivation du projet, les besoins auxquels la solution doit répondre. Dans le chapitre suivant, on analysera les entités du projet, et les différents cas d'utilisation. Enfin dans le dernier chapitre, je présente et justifier mon choix technologique, puis je termine par quelques captures d'écran présentant les principales interfaces et scénario d'utilisation.

Abstract

This document is a summary of our project at the end of study. This project aims to create a web application whose goal is the management of on line tests. Furthermore, and given its dynamic side it must be effective in the management of updates on data from the platform. And we hope this project will achieve these goals.

In this report, i describe in details the various steps of our project's development. In the first chapter, we'll describe the context and the motivation of the project, and the needs that the solution is about. In the next chapter, I' analyze the entities of the project, and the different use cases. Finally in the last chapter, i present and justify our technology's choices, then i finish with some screenshots with the main interface and usage scenarios.

Table des matières

Table des matières

Liste des abréviations

Abréviation	Désignation
DAO	Data Access Object
EJB	Entreprise Java Bean
J2EE	Java 2 Enterprise Edition
JPA	Java Persistence API
JSP	JavaServer Pages
JSF	Java ServerFaces
MVC	Model View Controller
UML	Unified Modeling Language

Table des figures

Liste des tableaux

Introduction

Le monde informatique est très vaste, son apparition correspond aux changements de culture. Jour après jour l'informatique occupe une place intéressante dans la société et à tous les niveaux : personnels, académiques, administratifs, industriels, …etc., en effet l'informatique permet de traiter, mémoriser, et de diffuser l'information dans des délais plus courts et avec des méthodes plus efficaces

la tendance actuelle des enterprises et des établissements est l'informatisation de leurs systèmes informatiques ou une partie de ces systems et mettre en oeuvre des applications pour une meilleure gestion de ces différents services

L'évaluation en ligne d'un candidat ou collaborateur est une solution idéale en rapport qualité/prix. C'est aussi un formidable outil pour valoriser les potentiels.

- Moins cher que le centre d'évaluation (assesment center) :

Comparez cette solution à une journée de centre d'évaluation. Elle est réellement plus économique. Elle ne nécessite pas de frais de transports. Vous aurez une réponse écrite sous quelques heures. De plus, pour des évaluations en volume, vous pouvez bénéficier de tarifs réduits.

- Plus facile pour un poste isolé :

Qu'il s'agisse d'une création de poste, d'un remplacement après départ, ou de la préparation d'un remplacement, vous ne disposez souvent pas en interne de l'expert disponible pour évaluer les compétences techniques. C'est justement le candidat qui est sensé posséder l'expertise nécessaire...

- Plus facile pour un remplacement :

Départ en retraite, départ forcé... Le prédécesseur ne possède pas toujours les qualités requises. Ses connaissances ne sont pas obligatoirement à la page. Le moins que l'on puisse demander au remplaçant, c'est justement d'être à la page. Faites appel à nous. Nous mettons régulièrement nos questionnaires à jour en fonction de l'évolution des lois, de l'économie, des pratiques...

- Plus précis qu'un entretien oral :

Menez un entretien oral n'est pas toujours facile. Même en ayant préparé sa liste de questions, on peut dériver, se laisser entraîner par un candidat bavard ou manipulateur. Le temps presse, et toutes les questions ne sont pas toujours posées. Avec nos tests en ligne, et l'option "temps illimité", le candidat doit répondre à toutes les questions. Les réponses sont claires et enregistrées.

- Rapide, Flexible :

Le test typique dure 10 minutes. Il vous suffit d'avoir le candidat dans vos locaux ou chez un tiers de confiance. Plusieurs candidats peuvent passer le test en parallèle. Vous pouvez edemander des modalités ajustées : questionnaire avec limite dans le temps pour répondre ou non, liste de questions identiques pour tous les candidats, liste plus ou moins longue, plusieurs tests enchainés (compétences génériques et compétences spécifiques...), possibilité de revue par un expert donnant un avis circonstancié...

- Idéal pour des formations

Vous voulez que vos formations soient efficaces. Vous avez deux exigences

Le candidat doit avoir le niveau nécessaire et suffisant pour aborder la formation avec motivation. A ce titre, le test étant régulièrement mis à jour, permettent de détecter les collaborateurs qui n'ont pas suivi l'évolution récente des pratiques, des contraintes juridiques, et bénéficieraient d'une remise à niveau.

Par ailleurs, il est important de savoir, sans ambigüité, si le candidat a réellement bénéficié de cette formation. Quoi de mieux que tester ses nouvelles compétences ?

Le test 'avant-après' vous permet ainsi de vérifier la qualité de l'organisme de formation, et le travail réel fourni par votre collaborateur.

Transition entre la formation scolaire et la vie active, le projet place les étudiants dans une situation se rapprochant le plus possible de cette dernière, notamment :

Sur le plan technique :

Mise en œuvre et intégration des connaissances techniques acquises avant et au cours du Projet, aboutissement à un résultat concret

Sur le plan de l'organisation :

Gestion de projet, organisation personnelle, résolution d'un problème avec prise en compte Des contraintes de délais, qualité, …, utilisation de sources d'information diverses.

Sur le plan humain

Communication (écrite et orale), travail en équipe, innovation, créativité, responsabilité, éthique, autonomie, ouverture.

En plus de ces points cités, le projet nous permettra, étudiants en informatique la maîtrise de nouveaux langages de programmation, d'appliquer des méthodes théoriques sur des problèmes réels ainsi que de se familiariser avec le monde des applications.

La structure est comme suit :

➔ Le premier chapitre s'agira d'une prise de connaissance de l'existant pour savoir ce que doit être capable de faire et de quoi va servir notre future application.

➔Le second chapitre sera consacré à la conception de l'application, il s'agit d'une phase de modélisation théorique de l'application. Orthographe

➔Dans le troisième chapitre on va faire le point sur nos choix des méthodes et outils à utiliser pour la réalisation de notre application.

➔Avant de clore on va essayer de présenter les résultats obtenus.

Chapitre 1

Contexte et motivation du projet

Introduction

Afin de définir le cadre de mon projet je commence par définir le contexte du sujet. Ensuite je présente le projet ainsi que ses principaux objectifs.

1 . Le Système d'Information (SI) :

[1] Né dans les domaines de l'informatique et des télécommunications, **le concept de système d'information** s'applique maintenant à l'ensemble des organisations, privées ou publiques. Le système d'information coordonne grâce à l'information les activités de l'organisation et lui permet ainsi d'atteindre ses objectifs. Il est le véhicule de la communication dans l'organisation.

1.2 Définition des SI :

Un Système d'Information peut être défini comme un ensemble de ressources (personnel, logiciels, processus, données, matériels, équipements informatique et de télécommunication…) permettant la collecte, le stockage, la structuration, la modélisation, la gestion, la manipulation, l'analyse, le transport, l'échange et la diffusion des informations (textes, images, sons, vidéo…) au sein d'une organisation. Exemples de ressources informatiques : fichiers de données, bases de données et SGBD (Système de Gestion de Bases de Données), progiciels intégrés (ERP, …), outils de gestion : gestion clients (CRM : Customer Relationship Management), gestion de la chaîne logistique (SCM : Supply Chain Management), gestion des employés (ERM : Employee Relationship Management), outils de travail collaboratif (GroupWare), applications métier, serveurs d'application, serveur de présentation (Web,…), système de Workflow, architecture d'intégration (EAI : Enterprise Architecture Integration, SOA : architectures orientées services), infrastructure réseau, … La définition donnée précédemment laisse entrevoir la complexité du SI dont les déclinaisons vont s'exprimer à l'aide de différentes architectures. Il est alors primordial aujourd'hui de différencier système d'information (SI) et système informatique. Un SI peut être considéré comme une vue « automatisable » des métiers d'une organisation et une vue fonctionnelle de l'informatique, donc indépendante de l'implémentation technique (figure 1). Le SI est plus pérenne que l'architecture informatique. Les évolutions applicatives et techniques peuvent être indépendantes du SI en raison de l'évolution des technologies, des configurations ou des besoins utilisateurs.

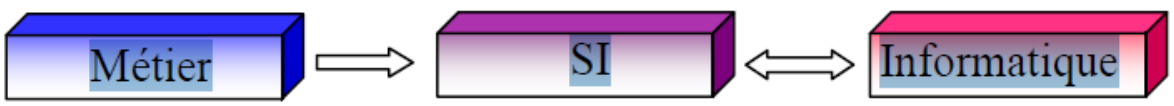


Figure 1. Découplage : Processus métier / Système d'information / Informatique (applications + architecture technique)

La conception de SI d'une entreprise requière des méthodes d'analyse de l'entreprise afin de modéliser les informations et les données, les flux d'information échangés ainsi que les traitements à appliquer sur ces données. Ces traitements sont identifiés grâce à l'analyse des processus métier

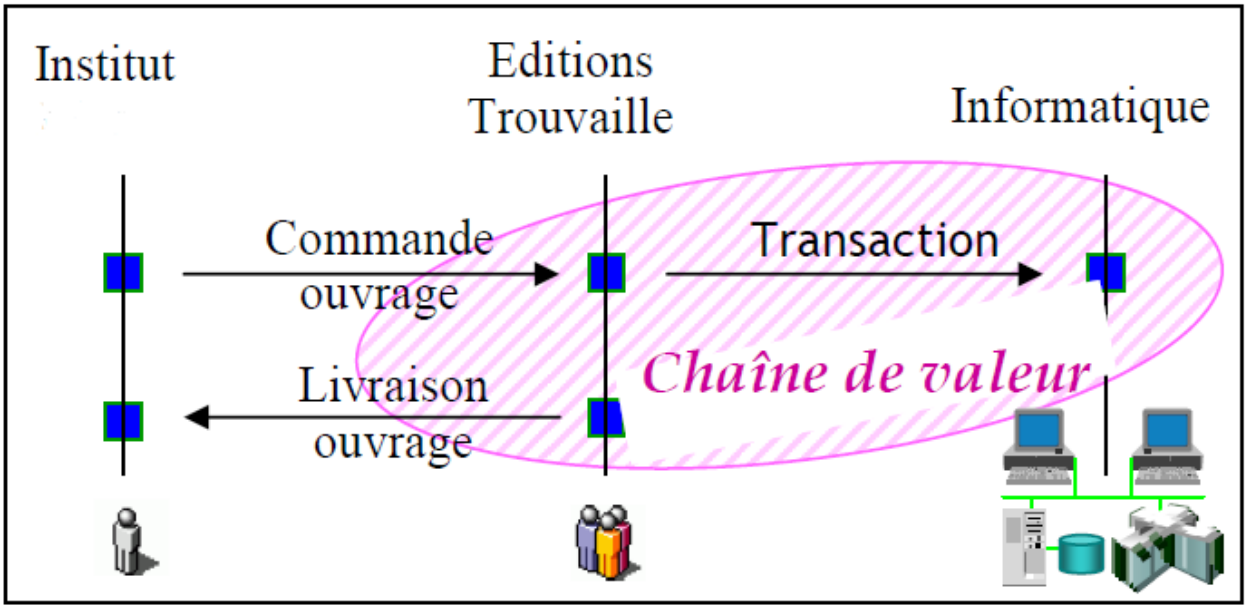


Figure 2. Exemple de diagramme de workflow d'un processus métier général

Des modèles ou langages de modélisation sont donc nécessaires. La méthode Merise et plus récemment UML sont les plus utilisés en monde dans la conception de SI. Métier SI Informatique Institut Martin Editions Trouvaille Informatique Commande ouvrage Livraison ouvrage Transaction Chaîne de valeur Dans beaucoup d'organisations, il ne s'agit plus aujourd'hui de concevoir un système d'information mais de le faire évoluer au rythme des besoins tout en exploitant les avancées technologiques. Force est de constater que la complexité des SI est proportionnelle à la complexité croissante des technologies et des organisations elles-mêmes. Le SI doit répondre aux enjeux stratégiques de l'entreprise, au développement du marché, supporter l'évolution des métiers et des fonctions au sein de l'organisation, et également supporter l'évolution du périmètre de l'organisation (fusion, intégration de différentes entreprises). Le SI doit donc être évolutif, réactif, flexible, ouvert et également sécurisé. C'est l'objectif de la démarche d'urbanisation des systèmes d'information. 2. Conception Plusieurs méthodes de conception de SI co-existent et sont exploitées différemment selon les pays. Parmi celles-ci, on peut citer Merise, **UML**, **AXIAL**, **IDEF**…

2.2 historique du système d'information

Il serait bien sur délicat de réalise une véritable synthèse de 18 études portant sur des périodes particulières des questions spécifique, des échantillons différents …c'est pourquoi il est préférable de présenter une analyse différenciée, en proposant alors de repère cinq grandes périodes dans l'histoire de la recherche en système d'information :

-la période du développement des SI avant 1980 : trois études (Ives Hamilton, Davis 1980 ,1981,1982) concluent que la grande majorité des recherches publiées sont alors de nature non empirique (soit conceptuelle, soit technique) sans formulation d'hypothèse claires.

-la période de la théorisation des SI 1980-1985 : trois études (clan et Swanson 1986, clanan 1987, cheon et al 1991) montrent des progrès importants vers une tradition de recherche cumulative et un abandon des recherches techniques : développement et accumulation de connaissances, travail en équipe, construction de théories, production de références propres à la discipline …

-la période positivisme 1985-1990 : deux études (Alavi et Carlson 1992, Grover et al 1993) montrent que l'orientation positivisme de la discipline de confirme alors, avec une rigueur méthodologique associée. Les études empiriques sur le terrain prennent une place de plus en plus importante

- la période de la diversification 1990-2000 : un certain monolithisme thématique (autour du développement de Si) de termine et sept études démontrent une diversification des projets de recherche (vers la gestion des organisations, le travail collaboratif …) et un renouvellement plus rapide des thèses d'application (nouvelles technologies, internet, ERP)

-la période du contexte social depuis 2000 une étude (Sidorova et al 2008) montre l'importance maintenant attachée au contexte social et la manière dont les individus, les groupes et les organisations interagissent avec les SI. Ceci traduit d'une certaine manière un alignement de la recherche sur les questions l'ordre du jour dans les organisations

2.Base de données

2.1Définition :

[2] Est un ensemble structuré et organisé de données qui représente un système d'informations sélectionnées de telle sorte qu'elles puissent être consultées par des utilisateurs ou par des programmes. Ainsi, dans une grande institution comme la Bibliothèque nationale, il s'agit de l'ensemble formé par les références des ouvrages, des auteurs, des éditeurs, etc. Dans une entreprise, la base de données contient l'ensemble des données concernant les clients, les fournisseurs, les employés, les références des produits fabriqués… et permet d'établir des relations entre ces différentes entités.

Dans les années 1960, les logiciels d'entreprise portaient spécifiquement sur une application. Pour chaque langage, les fichiers étaient définis dans les programmes qui les utilisaient, et données et programmes étaient liés. Pour chaque application, les données étaient stockées dans des fichiers

différents et la même donnée, par exemple l'adresse d'un fournisseur, pouvait figurer dans plusieurs fichiers de l'entreprise. Si cette adresse changeait, il fallait la modifier dans tous les fichiers existant dans l'entreprise. L'existence de fichiers créés au fur et à mesure des besoins nuisait à la cohérence et à la fiabilité des données de l'entreprise. La nécessité d'utiliser un seul fichier unifié pour toute l'entreprise est alors apparue clairement.

Afin d'avoir des données fiables et structurées, cohérentes même si elles sont partagées c'est-à-dire utilisées par plusieurs programmes, on a remplacé les différents fichiers par une seule « base de données », utilisable par tous les programmes. Lorsqu'une donnée est modifiée par un programme, elle est modifiée pour tous les autres. La base de données unifie la structuration des données par un « modèle » unique et cohérent, suffisamment général pour pouvoir s'adapter à toutes les situations particulières. On distingue le modèle logique et le modèle physique de la base de données.

Le modèle logique

Le modèle logique décrit l'organisation des données au niveau conceptuel indépendamment de leur implantation physique. On distingue trois types principaux de modèles logiques de bases de données : les modèles hiérarchiques, les modèles en réseaux et les modèles relationnels.

Les modèles hiérarchiques

Les modèles hiérarchiques sont les plus anciens (1965). Les informations y sont organisées sous forme d'arborescence et ne sont accessibles que par un point d'entrée unique. Le modèle hiérarchique oblige à la redondance de certaines informations (une même information peut figurer plusieurs fois), le maintien de sa cohérence est donc très difficile et son utilisation très rigide.

Les modèles en réseaux

Dans les modèles en réseaux, il n'y a plus d'information privilégiée, les données sont organisées sous forme d'un graphe, et chaque information peut être associée à plusieurs autres. Plus riches que les modèles hiérarchiques, les modèles en réseau sont très difficiles à gérer.

Les modèles relationnels

Le modèle relationnel a été défini de manière théorique par E. Codd en 1970. Dans ce modèle, fondé sur la théorie des ensembles, les données sont représentées sous forme de valeurs dans des tables et l'accent est mis sur les relations entre les données. Totalement indépendants de l'implantation physique des données, ils ont pour avantage leur très grande simplicité de conception. Ils assurent la cohérence, la non-redondance et la sécurité des données.

Le modèle physique

Le modèle physique décrit l'organisation physique des données (implantation dans une mémoire auxiliaire, différents types de stockage, implantation partielle en mémoire…). Il peut donc exister de nombreux modèles physiques associés à un modèle logique donné.

Un système de gestion de bases de données est un ensemble de logiciels qui permettent de gérer une base de données. On distingue trois générations de SGBD.

Les trois générations de SGBD

- La première génération, liée aux modèles hiérarchiques et en réseaux, est pratiquement abandonnée actuellement.
- La deuxième génération, liée aux modèles relationnels, s'est développée à partir des années 1980. Un SGBD relationnel permet de définir le modèle logique de la base de données, et se charge automatiquement de la définition du modèle physique. Un « langage de requêtes » comme SQL (Standard Query Language) permet d'interroger et modifier la base de données. Le SGBD maintient la cohérence (ou encore l'intégrité) des données par la définition de « contraintes d'intégrité » qui évitent par exemple que deux articles différents aient les mêmes références. Le SGBD permet de définir les droits d'accès des utilisateurs aux données, différentes procédures assurant la sécurité des données et évitant que deux utilisateurs modifient en même temps les mêmes données. Les SGBD relationnels sont utilisés principalement pour les applications de gestion.

- La troisième génération est en cours d'évolution. Les recherches se poursuivent pour représenter des structures très complexes, comme les bases de données géographiques, les données qui intègrent du texte, des images et des sons (par exemple les applications multimédias).

3.web

3.1 définition web :

[3] Le Web, développé au CERN en Suisse en 1989, est l'abréviation de World Wide Web. Littéralement, l'expression signifie "Toile d'araignée mondiale". C'est en effet un maillage de Serveur d'informations qui permet de Naviguer par simples clics de souris sur

environ 700 millions de pages : actualité, recherche, informatique, loisirs, art, vie pratique, …

C'est un système distribué d'accès à l'information qui s'appuie sur les principes de l'Hypertexte et qui supporte les documents multimédias

4.L'Entreprise

[4] Avant 1970

–L'entreprise était considérée comme une addition de services aux fonctions délimitées

–Les employés percevaient cela comme ayant parfois des visées contradictoires, voire antagonistes

Apparue dans les années 1970

•Entreprise = Système

– « Ensemble d'éléments en interaction dynamique, organisé en fonction d'un but » De

–L'entreprise est alors considérée comme un ensemble d'éléments (des moyens humains, matériels, financiers et techniques) en interrelations

–Toute organisation humaine (l'État, une famille, …) peut être perçue comme un système

Comme tout système, l'entreprise est un système :

–Ouvert sur l'environnement

–Il est finalisé (but = profit…)

–Il est en constante évolution

•Pour parvenir à son but, le système tient compte de son environnement et régule son fonctionnement en s'adaptant aux changements

Les éléments du système sont eux-mêmes des systèmes (ou sous-systèmes)

•L'entreprise peut se décomposer en 3 sous-systèmes :

–Le système de décision

–Le système d'information

–Le système opérant

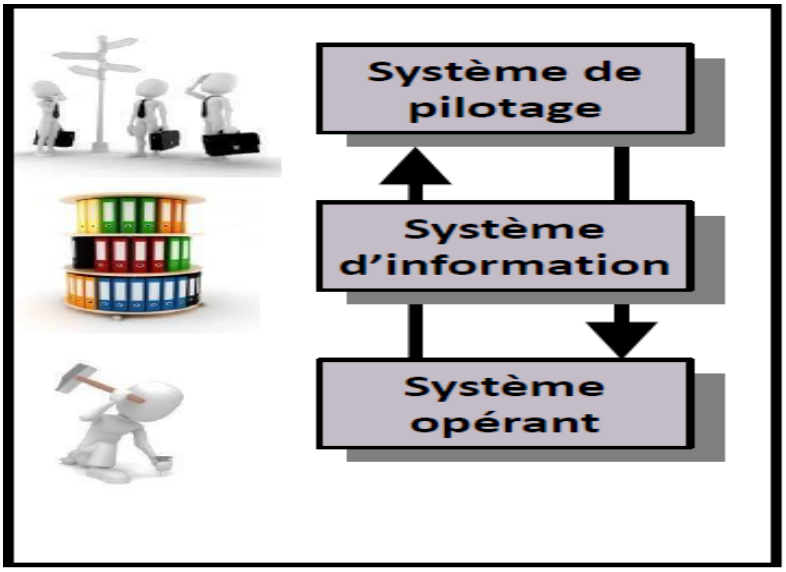


Figure 3. Composition de l'entreprise

5. Contexte et motivation du projet

5.1 Vision du projet

Le but principal de ce projet, est de mettre en place une solution logicielle de gestion des tests en lignes.

Cette solution logicielle permettra aux utilisateurs (candidats) de passer des tests qui contiendront des questions de types différents. Les candidats vont recevoir leurs résultats instantanément lorsqu'ils vont finir le test. Chaque test a une durée fixée, ainsi que chaque question dans le test a un délai. La gestion des tests est faite au préalable par les administrateurs.

L'application va offrir les fonctionnalités suivantes :

- Aux candidats :
  - L'inscription à des tests.
  - Le passage des tests.
  - Obtention des résultats.
- Aux administrateurs :
  - Gestion des candidats.
  - Définitions des tests : Date, durée,
  - Définition des questions, de types de questions, et les réponses.
  - Visualisation des résultats : Recherche par nom, prénom, code, …

5.2 Référentiel des exigences

on peut tirer des exigences primordiales à respecter. Ces exigences concernent d'une part les spécifications générales, et d'autre part les spécifications techniques. Je vais présenter par la suite ces exigences.

5.3 Spécifications générales

En ce qui concerne les spécifications générales, l'application doit répondre autant que possible aux facteurs suivants :

- Sécurité

L'application doit assurer un système de sécurité flexible, permettant d'accéder aux fonctionnalités du système selon les droits de l'utilisateur.

- Fiabilité

Le système doit fonctionner sans défaillance pour une durée donnée (robuste, constant, etc.).

- Efficacité

Le système doit fonctionner avec un optimum de ressources et de temps.

- Réutilisabilité

Il doit être possible de réutiliser certains modules du système.

- Testabilité

Effort requis pour s'assurer le bon fonctionnement du système (jeu d'essais et vérification de résultats).

- Portabilité

L'application ne doit pas dépendre de l'environnement d'exécution (matériel, logiciel).

- Convivialité

Effort requis pour l'apprentissage et le dialogue homme/machine (compréhensible, maniable, documenté, etc.).

Tous ses facteurs vont nous permettre d'assurer un service de qualité pour le client puisque ce dernier est la première priorité de l'entreprise.

5. Conclusion

L'étude préalable aussi appelée contexte et motivation du projet, constitue une phase cruciale, délicate du fait que toute la suite du projet dépend de cette étude, elle doit être faite avec beaucoup de rigueur, plus de soins et plus d'attention pour que le projet réussisse. Dans ce premier chapitre j'ai évoqué les problèmes de l'existant, puis j'ai proposé une approche de solution qui consiste à concevoir et à développer une application qui facilitera les services énumérés précédemment.

Chapitre 2

Conception

Introduction

Dans ce chapitre, je décrirai les étapes de modélisation du projet, cette phase semble la plus importante car elle représente la vue d'ensemble de la solution choisie.

2. Conception

2.1 UML : Unified Modeling Language, norme OMG (Object Management Group)

UML **[5]** est un langage pour spécifier, visualiser et construire des interfaces de modélisation. C'est un système de notation qui définit un ensemble de notations graphiques et leurs sémantiques et qui utilise des concepts orienté objet. Il est important de noter que UML est un langage de modélisation et non pas une méthode. Il définit en fait l'ensemble des notations graphiques nécessaires pour représenter des concepts orienté objet mais ne spécifie pas une démarche pour conduire un projet ou une phase de conception ou d'analyse orienté objet. UML définit neuf diagrammes, on ne va citer que ceux avec lesquelles on a travaillé :

- Diagrammes d'activités : représentent le comportement d'une opération en termes d'actions.
- Diagrammes des cas d'utilisation : représentent les fonctions du système de point de vue de l'utilisateur.
- Diagrammes de classes : représentent la structure statique en termes de classes et de relations d'une opération en termes d'actions.
- Diagrammes de séquences qui sont une représentation temporelle des objets et de leurs interactions.

Pourquoi UML *?*

Les objectifs qui ont été tracés pour UML, par ses créateurs, se résument en ce qui suit :

- Offrir un outil prêt à l'emploi basé sur une modélisation visuelle qui permet d'échanger des modèles compréhensibles.
- Etre indépendant des langages de programmation et des processus de développement
- Intégrer les meilleures pratiques.

En plus de ces raisons, un ensemble de faits peuvent nous pousser à adopter UML dans mon projet dont :

- UML est supporté par beaucoup d'AGLs (Ateliers de Génie Logiciel).
- UML devient un standard.
- UML est un langage formel et normalisé.
- Il facilite la compréhension de représentations abstraites complexes.

- Son caractère polyvalent et sa souplesse en font un langage universel.

Néanmoins, de point de vue pratique, UML présente également certaines faiblesses dont :

- UML est une notation, il faut la combiner avec une méthode
- De part son souci de rester générale et de supporter plusieurs types de modélisation (systèmes d'informations, temps-réel,...), UML laisse le soin aux éditeurs d'AGLs de Choisir les langages qu'ils veulent pour modéliser certains types de sémantiques (pré-conditions des contrats, conditions, expressions,...).

Malheureusement ce n'est pas toujours le cas et un effort de correspondance (mapping) entre la modélisation et le code reste à faire par les développeurs.

2.2 Identification des acteurs

Les acteurs identifiés durant l'analyse sont présentés dans le tableau ci-dessous :

Acteur	Fonction
Candidat	Il a la possibilité de s'inscrire dans le système pour passer des tests, et recevoir des résultats.
Administrateur	Il a une vision globale sur l'application. Il paramètre, contrôle, et supervise tout le domaine et l'environnement du projet.

Tableau 1: Description des acteurs

2.3 Exigences fonctionnelles

- **Module « Administration »**

L'administrateur est responsable du paramétrage de l'application. Le tableau ci - dessous, montre l'ensemble des fonctionnalités que le module « Administration » doit offrir :

Fonctionnalité	Description
Activation des comptes des candidats	Lorsqu'un candidat s'inscrit à un test, l'admin a la possibilité de lui autoriser l'inscription et donc le passage du test. Ou bien il rejette cette inscription.
Définition des questions	L'administrateur a la possibilité de définir des questions pour un test et supprimer d'autres selon les besoins. Et à chaque question il va définir les caractéristiques de celle-ci : type questions, réponses possibles, et la réponse correcte.
Visualisation des résultats	L'administrateur peut faire des recherches selon le code, le nom, le prénom, la filière, ou la date de passage des tests pour visualiser les résultats des candidats.

Définition des tests	Cette fonctionnalité donne la possibilité d'ajouter de nouveaux tests et de définir leurs dates, leurs durées, etc..
Préparer un test	Cette fonctionnalité dépend de la première et a comme but de remplir un test par des questions. Et ceci se fera aléatoirement.

Tableau 2: Exigences fonctionnelles du module « Administration »

- **Module « Gestion des candidats »**

Les fonctionnalités concernant ce module sont présentées dans le tableau ci-dessous :

Fonctionnalité	Description
Inscription des candidats	Cette fonctionnalité offrira les moyens pour qu'un candidat s'inscrive dans un test.
Affichage de récapitulatif d'inscription	Lorsqu'un candidat confirme son inscription, un récapitulatif va être généré pour afficher toutes les informations du candidat.

Tableau 3: Exigences fonctionnelles du module « Gestion candidat »

- **Module « gestion des tests »**

Ce module va offrir les fonctionnalités décrites dans le tableau ci-dessous :

Fonctionnalité	Description
Recherche des résultats	Cette fonctionnalité est celle qui va offrir un moyen pour chercher les résultats selon plusieurs critères.
Calcul des résultats de test instantané	Elle calcule les résultats d'un candidat au Moment de son passage des tests.

Tableau 4 : Exigences fonctionnelles du module « Gestion des résultats »

2.4 Diagrammes des cas d'utilisation du module «Administration»

Le module administration comprend le paramétrage et la gestion totale de l'application.

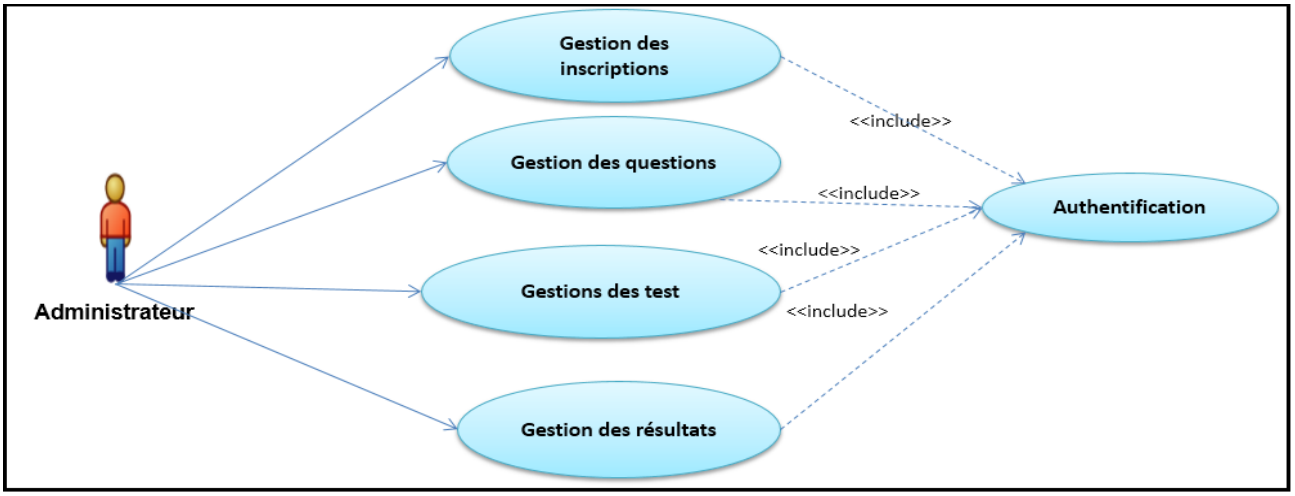


Figure 1 : Diagramme des cas d'utilisation du module « Administrateur »

2.5 Diagrammes des cas d'utilisation du module «Gestion des candidats»

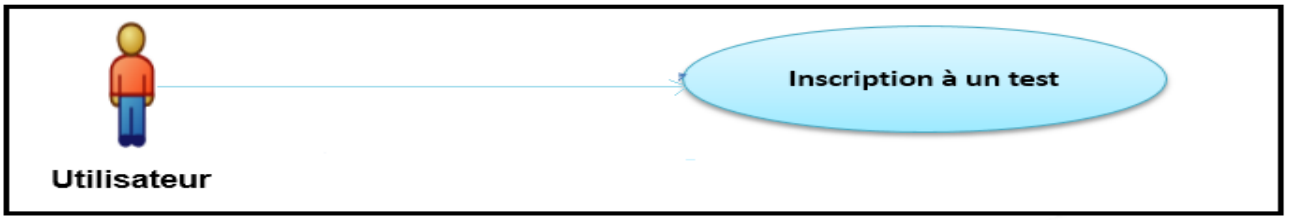


Figure 2 : Diagrammes des cas d'utilisation du module « gestion du candidat »

2.6 Diagramme s des cas d'utilisation du module « Gestion des tests »

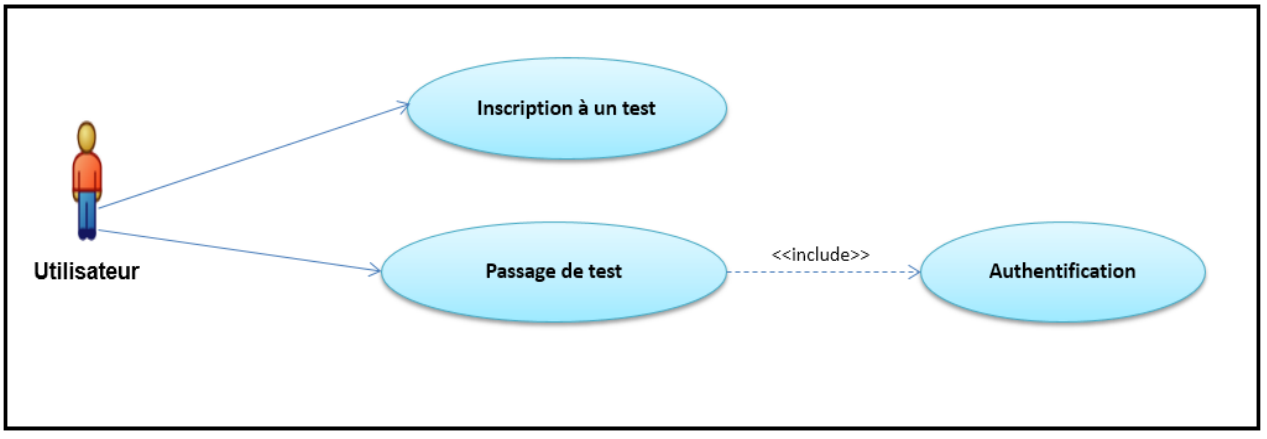


Figure 3 : Diagramme de cas d'utilisation du module « Gestion des tests »

Chaque cas d'utilisation décrit ci-dessus, contient un ou plusieurs scénarios. J'ai décrit dans la suite quelques scénarios considérés comme les plus importants.

- **Cas d'utilisation du module « Administration »**

Cas d'utilisation « Gestion des inscriptions »
Acteur : Administrateur **Pré-condition :** - L'authentification de l'administrateur - Le candidat est déjà inscrit. **Post-condition :** - En cas de confirmation, enregistrement dans la base de données. - En cas de rejet, l'inscription est annulée de la base de données.
Scénario principal
1- Connexion de l'administrateur 2- Consultation de la liste des nouvelles inscriptions 3- Valider/annuler les inscriptions

Tableau 5 : cas d'utilisation « Gestion des inscriptions »

Cas d'utilisation « Gestion des questions »
Acteur : Administrateur **Pré-condition :** - L'authentification de l'administrateur **Post-condition :** - Enregistrement dans la base de données dès confirmation.
Scénario : « créer question »
1- Connexion de l'administrateur 2- Choix du thème de la question 3- Saisie de la question 4- Choix du type de question 5- Saisie des réponses possibles 6- Saisie de la réponse correcte.

Tableau 6 : cas d'utilisation « Gestion des questions »

Cas d'utilisation « Gestion des tests »
Acteur : Administrateur **Pré-condition :** - L'authentification de l'administrateur **Post-condition :** - Enregistrement dans la base de données dès confirmation.
Scénario : « créer test »
1- Connexion de l'administrateur 2- Création d'un nouveau test 3- Précision de la durée par question 4- Précision de la date de passage de test 5- Précision de l'heure de début de fin 6- Précision du nombre de questions

Tableau 7: cas d'utilisation « Gestion des tests »

- **Cas d'utilisation du module « Gestion des candidats »**

Cas d'utilisation « inscription pour un test »
Acteur : Candidat **Post-condition :** - En cas de confirmation, enregistrement dans la base de données.
Scénario : s'inscrire pour un test
1- Candidat accède à l'application 2- Le candidat saisie son nom, prénom, école, filière, email, et GSM. 3- Le candidat envoie sa demande d'inscription.

Tableau 8: cas d'utilisation « inscription pour un test »

- **Cas d'utilisation du module « gestion de test »**

Cas d'utilisation « passage de test »
Acteur : Candidat **Pré-condition :** - L'authentification du candidat - Respect de la date du test. **Post-condition :** - Stockage des résultats dans la base de données puis l'afficher au candidat. - Envoie du résultat par mail.
Scénario : passer test
1- Connexion du Candidat 2- Passage de test

Tableau 9: cas d'utilisation « passage d'un test »

- **Cas d'utilisation du module « gestion des résultats »**

Cas d'utilisation «consultation du résultat»
Acteur : Administrateur **Pré-condition :** - L'authentification de l'administrateur
Scénario : consulter résultat
1- Connexion de l'administrateur 2- Choix du critère de la recherche des résultats. 3- Lancement de la recherche

Tableau 10: cas d'utilisation "gestion des résultats"

2.7 Diagramme de séquence du scénario : « créer test »

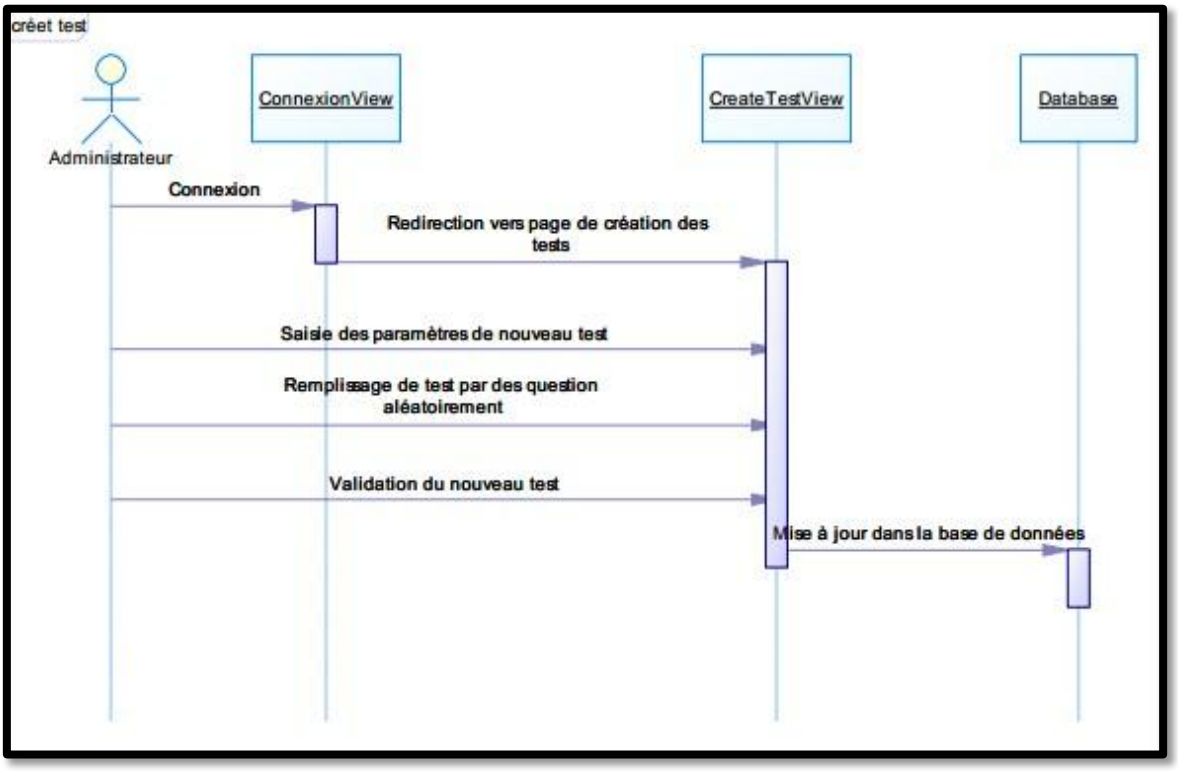


Figure 4 : Diagramme de séquence du scénario : « créer test »

2.8 Diagramme de séquence du scénario : « créer question »

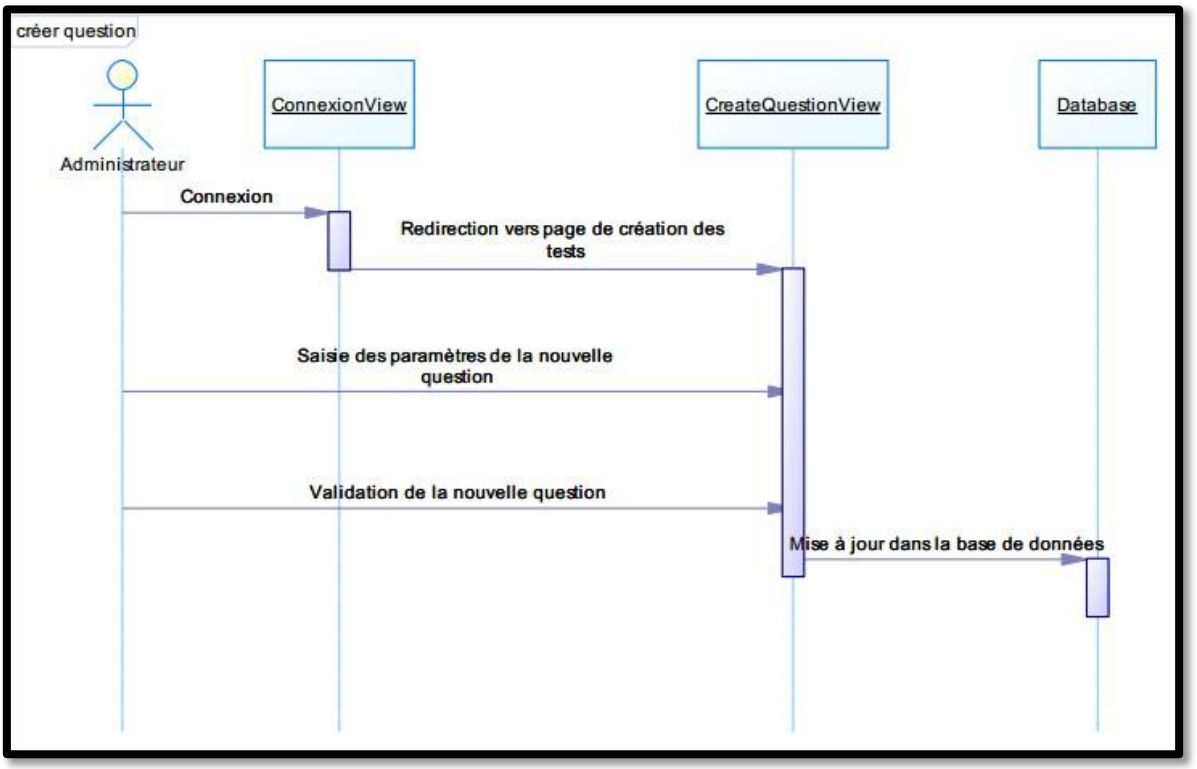


Figure 5: diagramme de séquences du scenario créer question

2.9 Diagramme de séquence du scénario « passer test »

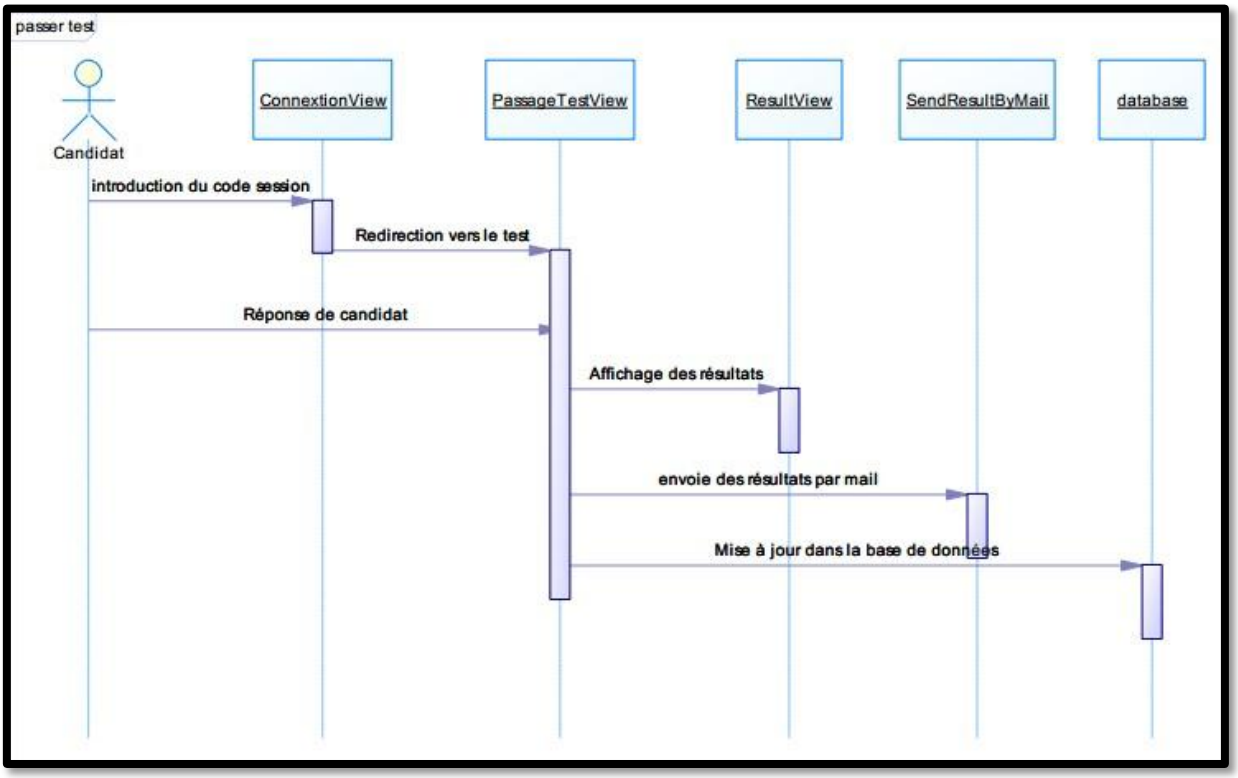


Figure 6: diagramme de séquence du scénario passer test

2.10 Diagramme de séquence du scénario « consulter résultat »

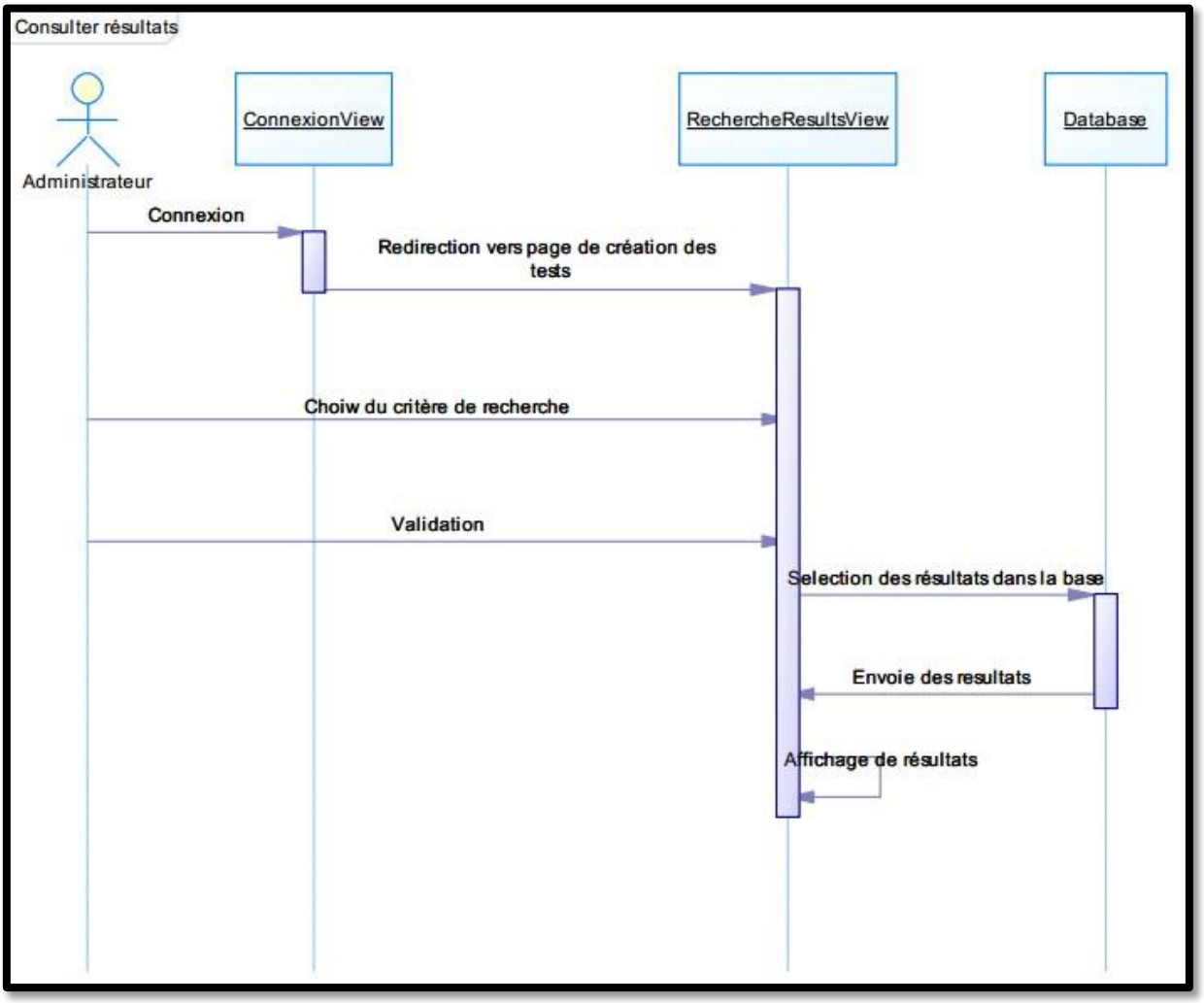


Figure 7: diagramme de séquence du scénario consulter résultat

2.11 Diagramme de classes

Voici le diagramme de classe que j'ai conçu pour notre application

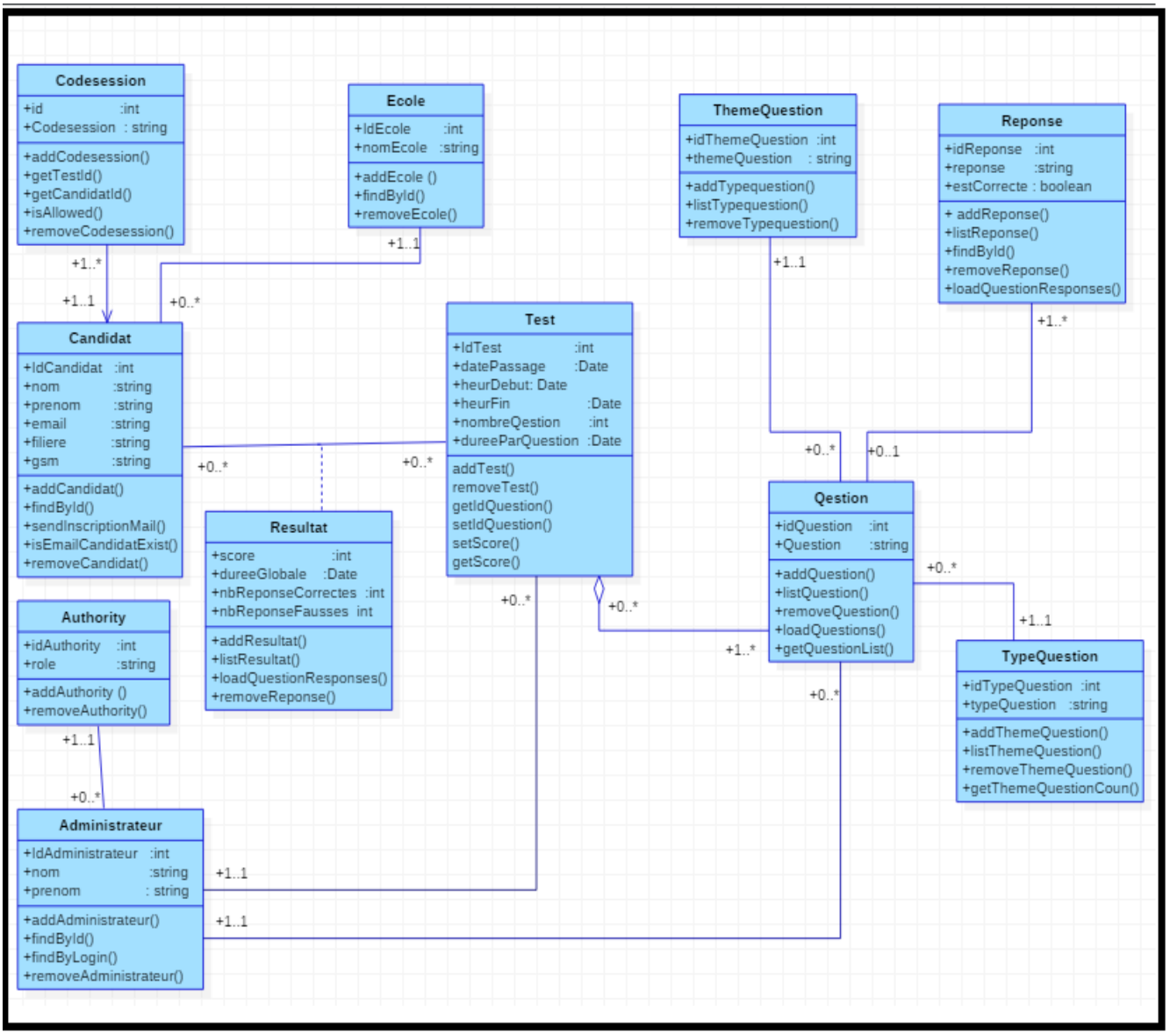


Figure 8: diagramme de classe

Conclusion

Ce chapitre décrit la phase de conception et modélisation du projet. Durant cette phase j'ai identifié les acteurs qui interagissent avec l'application et simplifié la représentation des processus ainsi que les données du projet. Le chapitre suivant présentera les méthodes et outils choisis pour le développement de l'application.

Chapitre 3

Méthode et outils de développement

Introduction

Il est évident que les méthodes et les outils choisis pour concevoir et développer une application doivent être en fonction de l'environnement et du domaine d'application de celle-ci. C'est pourquoi j'ai opté pour les outils ci-dessous :

3. Architecture applicative

L'architecture applicative, constitue une transition réelle du fonctionnel vers le technique, elle a pour but premier de déterminer les éléments structurant l'application et assurant les exigences et les besoins fonctionnels. Dans le cadre de cette architecture je vais traiter le modèle en couches.

3.1 Modèle en couche

Le modèle en couche [6] consistera à diviser une application web en couches pour la séparation de tout ce qui est logique métier avec la présentation et l'accès aux données. Parmi les différentes façons de structurer une architecture, la mieux adaptée et maîtrisée en informatique est l'approche par couches. Une couche (Layer en anglais) est une division logique et horizontale d'un système qui fournit une abstraction particulière du système aux couches supérieures. Chaque couche possède des responsabilités spécifiques.

Une telle architecture a les avantages suivants :

- La gestion des données et la logique métier peuvent être indépendants du type d'interface : La logique de l'application et ses données pourront être utilisées par une page JSP par exemple.
- Pendant la réalisation de la couche métier, on peut se concentrer sur la logique de l'application et l'intégrité des données, sans se soucier de la présentation, et la maintenance en devient plus facile.
- La couche présentation n'accède pas directement aux données, la configuration de l'accès aux données ne se fait alors que sur le serveur où réside la couche donnée.
- Si la couche métier ou celle des données est modifiée, il n'y a pas de réinstallation à faire sur les postes clients.

Le schéma ci-dessous illustre le principe du modèle en couche. Il y'a quatre couches [7] :

- Couche présentation.

- Couche Service.

- Couche DAO (accès aux données).

- Couche persistance.

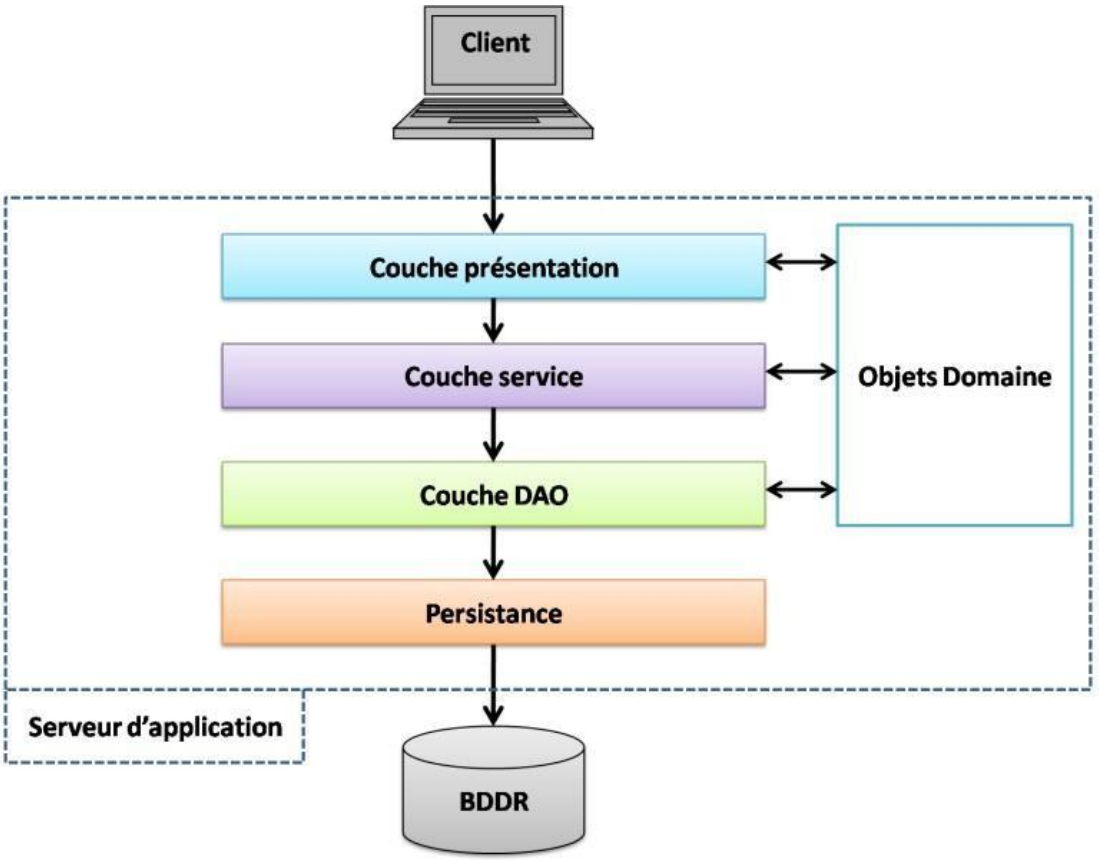


Figure 9 : **Architecture Applicative**

- Couche présentation :

Cette couche concerne l'interaction entre le système et les requêtes de l'utilisateur à travers les interfaces Homme-Machine.

- Couche service :

Cette couche offre des services applicatifs et métier à la couche présentation. Pour fournir ces services, elle s'appuie, le cas échéant, sur les données du système, accessibles au travers des services de la couche inférieure. En retour, elle renvoie à la couche présentation les résultats qu'elle a calculés.

- Couche DAO

Elle consiste en la partie gérant l'accès aux données du système. Ces données peuvent être propres au système, ou gérées par un autre système. La couche métier n'a pas à s'adapter à ces

Deux cas, ils sont transparents pour elle, et elle accède aux données de manière uniforme (couplage faible).

- Couche persistance

Cette couche s'intéresse à la représentation du modèle de données objet en base de Données.

Mise à part ces couches, il y a aussi une couche physique qui est tout simplement le SGBD.

3.2 Recensement des besoins techniques de l'application web

La phase de capture des besoins techniques consiste à extraire et à faire un diagnostic des spécifications physiques et logicielles de l'application en se basant sur l'abstraction élaborée dans la phase de l'étude fonctionnelle. Alors, l'objectif fondamental de cette phase est de dégager les contraintes d'environnement et d'implémentation mises en jeu. La gestion de la performance, la dépendance des plateformes techniques, la capacité de maintenance, l'extensibilité et la fiabilité sont les paramètres à expliciter dans cette phase.

En effet, l'application « Gestion des tests en ligne » doit être conforme, en plus des exigences fonctionnelles extraites, à un nombre de contraintes techniques énumérées dans ce qui suit :

- Un serveur Tomcat 7 [8] est envisagé pour déployer l'application «Gestion des tests en ligne».
- La base de données est de type MySQL.
- Le développement des composants doit être réutilisable permettant l'extensibilité de l'application dans le futur.
- L'architecture applicative doit être organisée en couche distinctes.

3.3 Spécifications logicielles

Dans cette partie, je vais dévoiler la plate-forme applicative. En outre, je vais dresser la suite logicielle l'ensemble des brique choisies pour le déploiement de l'application.

Pour mon application web, j'ai adopté un modèle de conception « Design Pattern » qui décrit une meilleure pratique ou une solution prouvée qui permet de résoudre à toutes contraintes les difficultés qui lui sont attachées, en mettant l'accent sur le contexte et sur les conséquences et les impacts de la solution, et spring framwork Durant le cycle de développement, j'ai choisi d'implémenter le modèle de conception MVC1 (Model View Controller 1), l'objet d'accès aux données DAO, et la persistance des objets avec JPA (qui est managé par Spring Framework).

3.4 Spring Framework

[9] SPRING est effectivement un conteneur dit « léger », c'est-à-dire une infrastructure similaire à un serveur d'application J2EE.Il prend en charge la création d'objets et la mise en relation d'objets par l'intermédiaire d'un fichier de configuration qui décrit les objets à fabriquer et les relations de dépendances entre ces objets. Le gros avantage par rapport aux serveurs d'application est qu'avec SPRING, les classes n'ont pas besoin d'implémenter une quelconque interface pour être prises en charge par le framework (au contraire des serveurs d'applications J2EE et des EJBs). C'est en ce sens que SPRING est qualifié de conteneur « léger ».

Outre cette espèce de super fabrique d'objets, SPRING propose tout un ensemble d'abstractions permettant de gérer entre autres : Le mode transactionnel. L'appel d'EJBs. La création d'EJBs. La persistance d'objets La création d'une interface Web. L'appel et la création de WebServices. Pour réaliser tout ceci, SPRING s'appuie sur les principes du design pattern IoC et sur la programmation par aspects (AOP). Spring est disponible sous licence Apache 2.0.

Les services fournis par Spring

Spring propose les services suivants (liste non-exhaustive) :

Découplage des composants. Moins d'interdépendances entre les différents modules. Rendre plus aisés les tests des applications complexes c'est-à-dire des applications multicouches.

Diminuer la quantité de code par l'intégration de frameworks tiers directement dans Spring.

Permettre de mettre en œuvre facilement la programmation orientée aspect.

Un système de transactions au niveau métier qui permet par exemple de faire du "two-phases-commit".

Un mécanisme de sécurité. Pas de dépendances dans le code à l'api Spring lors l'utilisation de l'injection.

Ce qui permet de remplacer une couche sans impacter les autres.

Une implémentation du design pattern MVC I Un support du protocole RMI. Tant au niveau serveur qu'au niveau du client.

Déployer et consommer des web-services très facilement. I Echanger des objets par le protocole http.

Les modules de Spring

Le framework est organisé en modules, reposant tous sur le module Spring Core :

Spring Core : implémente notamment le concept d'inversion de contrôle (injection de dépendance). Il est également responsable de la gestion et de la configuration du conteneur.
Spring Context : Ce module étend Spring Core. Il fournit une sorte de base de données

d'objets, permet de charger des ressources (telles que des fichiers de configuration) ou encore la propagation d'évènements et la création de contexte comme par exemple le support de Spring dans un conteneur de Servlet.

Spring AOP : Permet d'intégrer de la programmation orientée aspect.

Spring DAO : Ce module permet d'abstraire les accès à la base de données, d'éliminer le code redondant et également d'abstraire les messages d'erreur spécifiques à chaque vendeur. Il fournit en outre une gestion des transactions.

Spring ORM : Cette partie permet d'intégrer des frameworks de mapping Object/Relationnel tel que Hibernate, JDO ou iBatis avec Spring. La quantité de code économisé par ce package peut-être très impressionnante (ouverture, fermeture de session, gestion des erreurs)

Spring Web : Ensemble d'utilitaires pour les applications web. Par exemple une servlet qui démarre le contexte (le conteneur) au démarrage d'une application web. Permet également d'utiliser des requêtes http de type multipart. C'est aussi ici que se fait l'intégration avec le framework Struts. I

Spring Web MVC : Implémentation du modèle MVC. Personnellement j'utilise plutôt Struts mais c'est surtout une question d'habitude, c'est là la grande force de Spring, rien ne vous oblige à tout utiliser et vous pouvez tout mélanger. Ce qui n'est pas forcément une bonne idée mais nous en reparlerons.

Fichier de configuration de Spring : **applicationContext.xml** de l'application gestion teste en ligne

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:aop="http://www.springframework.org/schema/aop"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
            http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
            http://www.springframework.org/schema/context
            http://www.springframework.org/schema/context/spring-context-3.0.xsd
            http://www.springframework.org/schema/mvc
            http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd
            http://www.springframework.org/schema/aop
            http://www.springframework.org/schema/aop/spring-aop-3.1.xsd">

    <!-- Activates various annotations to be detected in bean classes -->
    <context:annotation-config />

    <!-- Scans the classpath for annotated components that will be auto-registered as Spring beans.
     For example @Controller and @Service. Make sure to set the correct base-package-->
    <context:component-scan base-package="com.testonline" />

    <!-- Configures the annotation-driven Spring MVC Controller programming model.
    Note that, with Spring 3.0, this tag works in Servlet MVC only!  -->
    <mvc:annotation-driven />

    <!-- Load Hibernate related configuration -->
    <import resource="hibernate-context.xml" />
    <import resource="security.xml" />
    <import resource="spring-datasource.xml" />
```

3.4 Design pattern : modèle vue contrôleur (mvc)

Pour séparer, les données des traitements et de la présentation, j'utiliserai le design pattern MVC.

Le MVC [10] est un modèle de conception qui repose sur la volonté de séparer les données, les traitements et la présentation. Ainsi l'application,

sujet de mon projet, se retrouve segmentée en trois composants essentiels.

Chacun de ces trois composants joue un rôle bien défini :

- **Le modèle :** représente les données et les règles métiers. C'est dans ce composant qu'on effectue les traitements liés au cœur du métier. Les données peuvent être liées à une base de données, des EJBs ou des services Web. Il est important de noter que les données sont indépendantes de la présentation. En d'autres termes, le modèle ne réalise aucune mise en forme, par ailleurs, ces données peuvent être affichées par plusieurs vues.
- **La vue :** correspond à l'IHM, elle présente les données et interagit avec l'utilisateur. Dans le cadre de l'application la vue correspond aux différentes pages JSP.

- **Le contrôleur**, quant à lui, se charge d'intercepter les requêtes de l'utilisateur, d'appeler le modèle puis de rediriger vers la vue adéquate. Il ne doit faire aucun traitement. Il ne fait que de l'interception et de la redirection.

Il existe deux versions de MVC : il y a MVC1 et MVC2. J'ai choisi d'utiliser MVC1 qui consiste à associer à chaque vue un contrôleur.

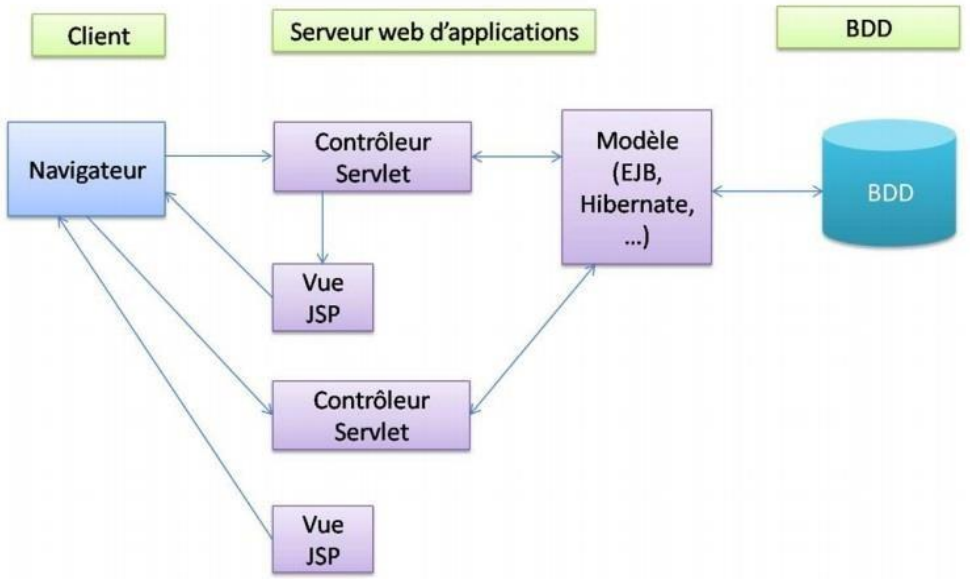


Figure 10 : Principe de MVC1

- Dans ce modèle chaque requête est traitée par un contrôleur sous forme d'une servlet.

- Celle-ci traite la requête, fait appel aux éléments du modèle si nécessaire et redirige la requête vers une JSP qui se charge de créer la réponse à l'utilisateur.

3.5 L' objet d'accès aux donnée s DAO

Ce pattern permet de faire le lien entre la couche d'accès aux données et la couche métier d'une application. Il permet de mieux maîtriser les changements susceptibles d'être opérés sur le système de stockage des données, donc, par extension, de préparer une migration d'un système à un autre (BDD vers fichiers XML par exemple...).

3.6 Persistance des objets

La Java Persistance API [11] repose essentiellement sur l'utilisation des annotations, introduites dans Java 5. Elles permettent de définir très facilement, et précisément des objets métier, qui pourront servir d'interface entre la base de données et l'application.

3.7 Environnement de développement de la version web

Choisir un environnement de développement est une tache trop complexe qui nécessite beaucoup d'expertise technique pour répondre aux exigences fonctionnelles d'une application. Connaissant un petit peu, j'ai proposé de développer mon application web avec une architecture JEE. Dans cette partie, je présentera cet environnement avec les différents Frameworks et outils qui vont entrer en jeu dans la réalisation de ce projet.

- **L'IDE Eclipse :**

Eclipse [12] est un projet de la Fondation Eclipse visant à développer tout un environnement de développement libre, extensible, universel et polyvalent.

Son objectif est de produire et fournir divers outils gravitant autour de la réalisation de logiciel, englobant les activités de codage logiciel proprement dites (avec notamment un environnement de développement intégré) mais aussi de modélisation, de conception, de test, de reporting, etc. Son environnement de développement notamment vise à la généricité pour lui permettre de supporter n'importe quel langage de programmation.

Le projet Eclipse est pour cela organisé en un ensemble cohérent de projets logiciels distincts, sa spécificité tenant à son architecture totalement développée autour de la notion de plugin : toutes les fonctionnalités de l'atelier logiciel doivent être développées en tant que plug-in bâti autour de l'IDE Eclipse Platform.

Eclipse recouvre donc notamment également à cet effet tout un Framework de développement logiciel fournissant des briques logicielles à partir desquelles développer tous ces outils. C'est la raison pour laquelle Eclipse est présenté dans la littérature tout autant comme un EDI ou comme un Framework.

- Mise en œuvre des couches de l'application web

- **Couche présentation**

Pour la mise en œuvre de cette couche je vais choisi d'utiliser les Frameworks Spring MVC pour implémenter le modèle, vue, contrôleur, ainsi que JSF, et les librairies RichFaces pour enrichir les composants jsp.

- **Couche de service**

Cette couche fournie des services pour la couche présentation. Tout ce qui est logique métier de l'application doit apparaitre à ce niveau. J'utilise le Framework Spring pour satisfaire cette tache. Le Framework Spring va gérer aussi le JPA que je explique son principe dans la suite.

- **Couche d'accès aux données**

  Pour la mise en œuvre de cette couche j'ai choisi d'utiliser JPA (managé par Spring).

- **Couche de persistance**

Pour cette couche, j' utilise Hibernate et JPA (managé par Spring).

- Présentation des principes des Frameworks de développement :

- **Spring MVC**

Spring MVC [13] aide à développer des applications web flexible et faiblement couplés. Le modèle de conception Modèle-Vue-Contrôleur permet de séparer la logique métier, la logique de présentation et la logique de navigation.

Les modèles sont responsables pour encapsuler les données d'application.
Les Vues ont comme rôle de rendre réponse à l'utilisateur à l'aide de l'objet modèle.
Les contrôleurs sont chargés de recevoir la demande de l'utilisateur et d'appeler des services.

La figure ci-dessous montre le flux de la demande dans le Framework MVC Spring :

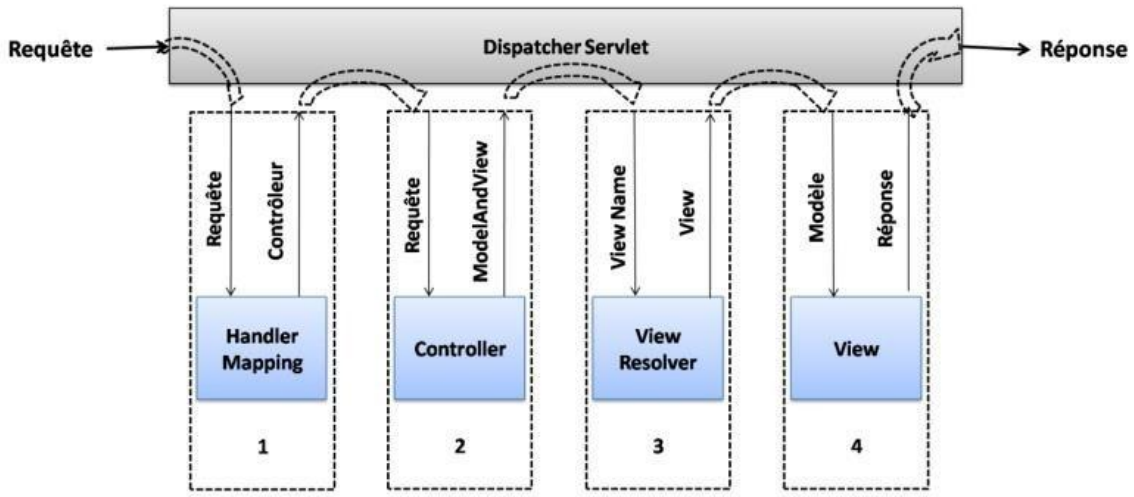


Figure 11 : Principe Spring MVC

Quand une requête est envoyée, la séquence suivante d'événements se produit.

- Le « DispatcherServlet » reçoit d'abord la demande.

- Le « DispatcherServlet » consulte le « HandlerMapping » et invoque le contrôleur associé à la demande.

- Le contrôleur fait appel aux méthodes appropriés de la couche service et retourne un objet «ModeAndVie » au « DispatcherServlet ». L'objet « ModeAndView » contient les données du modèle et le nom de la vue.

- Le « DispatcherServlet » envoie le nom de la vue au « ViewResolver » pour trouver la vue actuelle à invoquer.

- Le « DispatcherServlet » passe l'objet modèle à la vue.

- Enfin, la vue, à l'aide du modèle de données, rend le résultat à l'utilisateur.

- **Spring IOC**

L'injection des dépendance, ou l'inversion de contrôle est un concept qui intervient généralement au début de l'exécution de l'application, Spring IOC commence par lire un fichier XML qui déclare quelles sont différentes classes à instancier et d'assurer les dépendances entre les différentes instances.

Quand on a besoin d'intégrer une nouvelle implémentation à une application, il suffirait de la déclarer dans le fichier xml de beans spring.

- **Java ServerFaces (JSF)**

Java Server Faces (JSF) est une technologie dont le but est de proposer un Framework qui facilite et standardise le développement d'applications web avec Java. Son développement a tenu compte des différentes expériences acquises lors de l'utilisation des technologies standards pour le développement d'applications web (servlet, JSP, JSTL) et de différents Frameworks (Struts, ...).

- **Java Persistense API (JPA) managé par Spring**

La Java Persistence API (abrégée en JPA), est une interface de programmation Java permettant aux développeurs d'organiser des données relationnelles dans des applications utilisant la plateforme Java.

L'utilisation de JPA dans un environnement non managé peut se révéler délicate et problématique. Ceci est dû essentiellement à la gestion de la session de persistance, qui dans le mode non-managé doit être gérée à la main par le développeur, or la méthode la plus simple qui consiste à ouvrir une session de persistance chaque fois qu'on en a besoin est inefficace.

C'est pour cela qu'il vaut mieux (il le faut parfois) utiliser un conteneur pour la gestion de la session de persistance (JPA, Hibernate, etc.) comme par exemple le conteneur Spring.

Fichier de configuration de Spring : **Hibernate-Context.xml** de l'application gestion teste en ligne

```
    <!-- Declare a datasource that has pooling capabilities-->
    <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"
                destroy-method="close"
                p:driverClass="${database.driver}"
                p:jdbcUrl="${database.url}"
                p:user="${database.user}"
                p:password="${database.password}"
                p:acquireIncrement="5"
                p:idleConnectionTestPeriod="60"
                p:maxPoolSize="100"
                p:maxStatements="50"
                p:minPoolSize="10" />

    <!-- Declare a JPA entityManagerFactory-->
    <bean id="entityManagerFactory" class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean" >
        <property name="persistenceXmlLocation" value="classpath*:META-INF/persistence.xml"></property>
        <property name="persistenceUnitName" value="hibernatePersistenceUnit" />
        <property name="dataSource" ref="dataSource"/>
        <property name="jpaVendorAdapter">
            <bean class="org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter" >
                <property name="showSql" value="true"/>
            </bean>
        </property>
    </bean>

    <!-- Declare a transaction manager-->
    <bean id="transactionManager" class="org.springframework.orm.jpa.JpaTransactionManager">
        <property name="entityManagerFactory" ref="entityManagerFactory" />
    </bean>
</beans>
```

- **Framework Hibernate**

Hibernate [14] est une solution open source de type ORM (Object Relational Mapping) qui permet de faciliter le développement de la couche persistance d'une application. Hibernate permet donc de représenter une base de données en objets Java et vice versa.

Hibernate facilite la persistance et la recherche de données dans une base de données en

réalisant lui même la création des objets et les traitements de remplissage de ceux-ci en accédant à la base de données. La quantité de code ainsi épargnée est très importante d'autant que ce code est généralement fastidieux et redondant.

Hibernate a besoin de plusieurs éléments pour fonctionner :

- une classe de type javabean qui encapsule les données d'une occurrence d'une table
- un fichier de configuration qui assure la correspondance entre la classe et la table (mapping)
- des propriétés de configuration notamment des informations concernant la connexion à la base de données

Une fois ces éléments correctement définis, il est possible d'utiliser Hibernate dans le code des traitements à réaliser.

L'architecture d'Hibernate est donc la suivante :

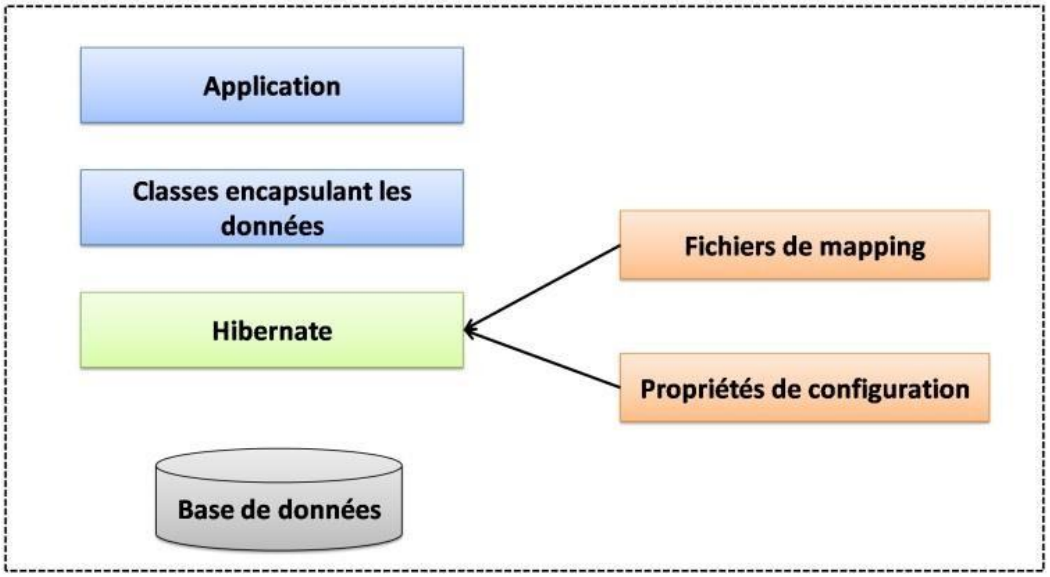


Figure 12 : Architecture d'Hibernate

3.8 Architecture technique globale de l'application web

Voici ci-dessous, un schéma qui illustre l'architecture technique et logique de notre application web :

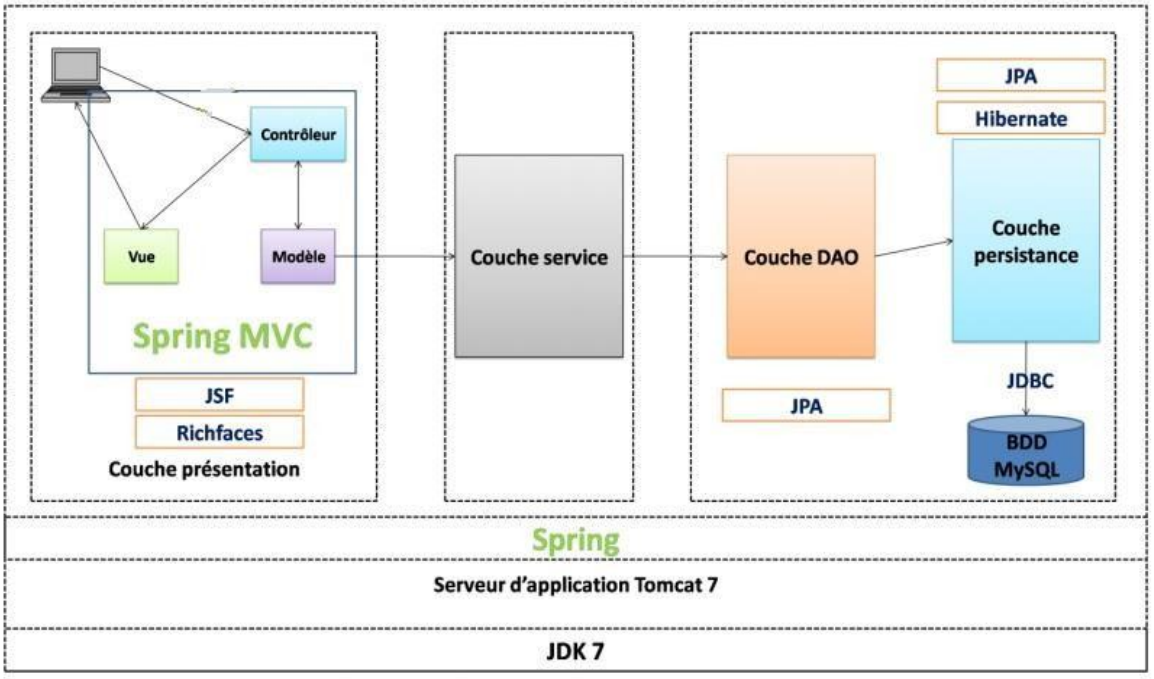


Figure 13 : Architecture General de l'application

Conclusion

Dans ce chapitre, j'ai fait le point sur mon choix des méthodes et outils à utiliser pour la réalisation de cette application. Dans le chapitre suivant, je vais présenter l'application réalisée sous forme de prises d'écran.

Chapitre 4

Réalisation

4. Réalisation :

4.1 Présentation visuelle du travail obtenu :

Notre plateforme est un ensemble des pages dynamiques conçues pour être parfaitement interactive avec et les utilisateurs et l'administrateur avec convivialité et facilité de la consultation.

La figure ci-dessous représente la page d'accueil de l'application. C'est la première page qu'un utilisateur rencontre. L'administrateur peut s'authentifier et accéder à son espace personnel, par contre un candidat désirant passer le test doit d'abord s'inscrire, attendre la validation de son inscription par l'administrateur et enfin accéder à l'espace réservé au passage des tests.

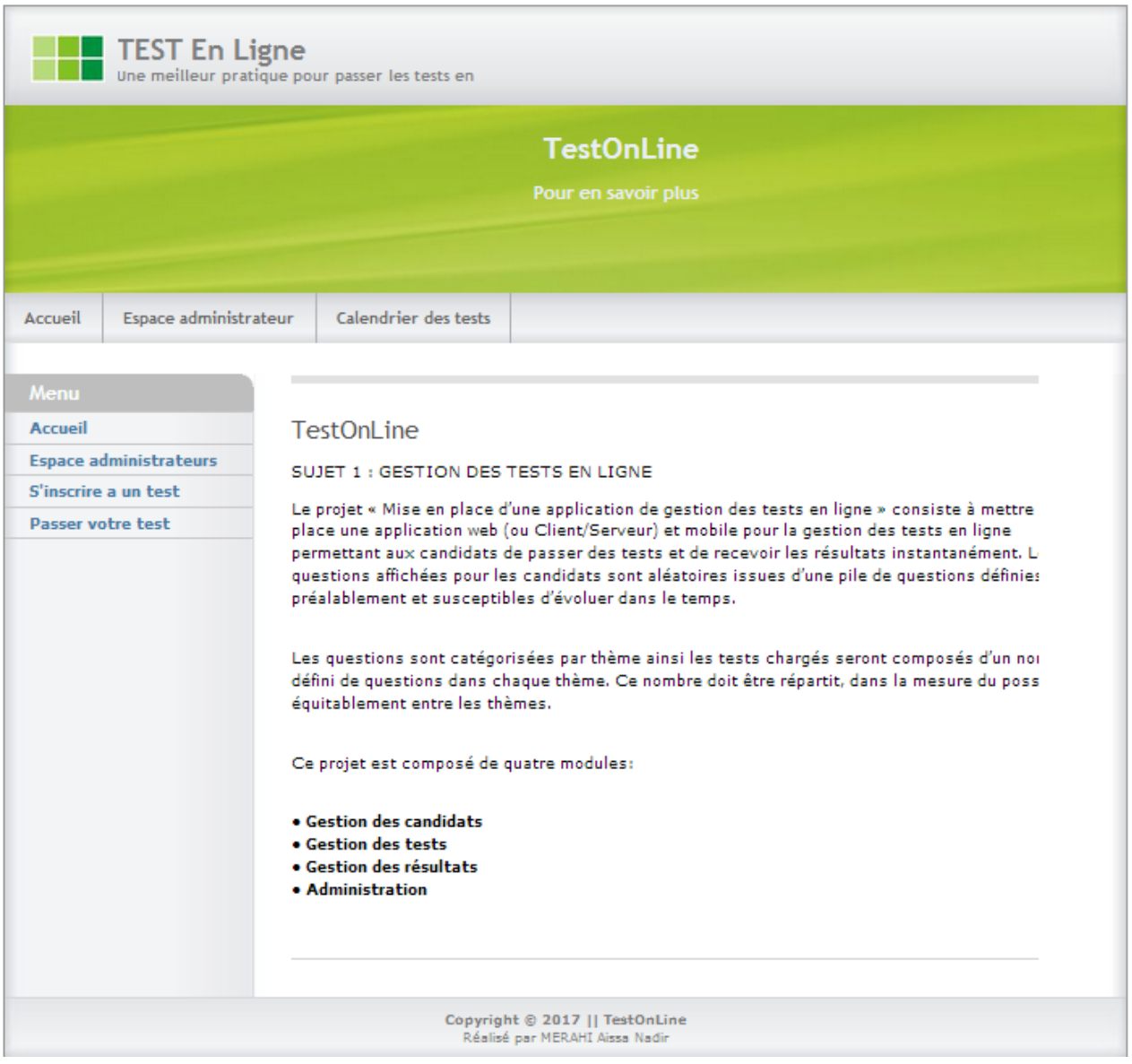


Figure 14: Page d'accueil de l'application

Pour accéder à son espace, l'administrateur doit tout d'abord entrer son login et son mot de passe pour s'authentifier.

Figure 15: authentification de l'administrateur

Il se peut que l'administrateur se trompe et entre un login et/ou un mot de passe incorrect, dans ce cas la boite de dialogue ci-dessous apparait :

Figure 16: erreur lors de l'authentification

Si les données entrées lors de l'authentification sont valides, l'administrateur peut donc accéder à son espace, à partir duquel il a le privilège de gérer les candidats inscrits, gérer les tests ainsi que les résultats.

Figure 17: Espace administrateur

Pour gérer les candidats, une liste contenant les informations de chacun s'affiche et donc l'administrateur peut soit supprimer ou modifier … selon son besoin

Figure 18: Espace gestion des candidats

De même pour les tests, une liste s'affiche contenant la date de passation de l'examen, sa durée, son heure de début, le nombre de questions et la durée estimée pour chacune

Figure 19: Espace gestion des tests

le candidat reçoit un mail de confirmation d'inscription

Figure 20 : mail de confirmation d'inscription

projet.testonline@gmail.com <projet.testonline@gmail.com>
À : m_nadir81@yahoo.com

Bonjour Mr koi
Vous venez de terminer votre test.

Voici vos informations d'inscription:

- Nom: koi
- Prénom: koi
- Email: m_nadir81@yahoo.com
- Filière ghh
- Ecole: INPT

Voici le détail de votre test:

- Score: 2 sur 5 points
- Nombre de réponses correctes: 2
- Nombre de réponses fausses: 3
- Temps consommé: 01:00:41

Vouz etre :admis

Merci et bon courage par la suite !.

Figure 21 : mail de résultat

Conclusion

Ce chapitre a été réservé pour la présentation du projet réalisé. Ainsi, j'ai fait quelques prises d'écran de certaines pages pour montrer et expliquer le fonctionnement de l'application.

Conclusion générale

Ce projet me permis de mettre en œuvre et d'intégrer en situation réelle les capacités en cohérence avec mon projets personnels et professionnels.

Pour récapituler, dans le cadre du projet de fin d'étude, je vais réalisé une application visant la gestion des tests en ligne. Cette solution logicielle permet aux utilisateurs (candidats) de passer des tests qui contiennent des questions de types différents. A la fin de l'examinassions, les candidats vont recevoir leurs résultats instantanément lorsqu'ils vont finir le test. Par ailleurs, chaque test a une durée fixée, ainsi que chaque question dans le test a un délai. La gestion des tests est faite au préalable par les administrateurs.

Finalement, mon projet ne s'arrête pas ici, la prochaine étape sera consacrée au perfectionnement des fonctionnalités qui vont rendre cette application plus fiable et ses services plus pertinents.

Bibliographie

[1] **livre** Conception, architecture des systèmes d'information
Maître de Conférences, LIRIS, INSA-Lyon, F-69621 Villeurbanne Cedex

[2] **Article sur internet** : Les bases de données: définition et concepts
Présentation et illustration des principaux concepts: base de données, SGBD, SQL, modèle relationnel
Agence Wallonne des Télécommunications 18/04/2001

[3] https://www.astuces-aide-informatique.info/70/qu-est-ce-que-le-web(consulté le 02/03/17)

[4] **Cours** : ESTIA 2è année – Guillaume Rivière Dernière révision : Avril 2014

[5] http://uml.free.fr/ (consulté le 02/03/17)

[6] http://www.techno-science.net/?onglet=glossaire&definition=5266 (consulté le 02/03/17)

[7] http://fr.wikipedia.org/wiki/Architecture_trois_tiers (consulté le 12/03/17)

[8] http://tomcat.apache.org/ (consulté le 13/03/17)

[9] livre : Spring par la pratique Mieux développer ses applications Java/J2EE avec Spring, Hibernate, Struts, Ajax... - Spring 1.2 et 2.0 Date de publication originale : 2006
Auteurs : Julien Dubois, Jean-Philippe Retaillé, Thierry Templier

[10] http://www.siteduzero.com/informatique/tutoriels/apprenez-a-programmer-en-java/mieux-structurer-son-code-le-pattern-mvc (consulté le 25/05/17)

[11] http://docs.oracle.com/javaee/6/tutorial/doc/bnbpz.html (consulté le 26/05/17)

[12] http://www.siteduzero.com/informatique/tutoriels/creez-votre-application-web-avec-java-ee/l-ide-eclipse-2 (consulté le 26/05/17)

[13] http://tahe.developpez.com/java/springmvc-part1/ (consulté le 01/06/17)

[14] http://www.mistra.fr/tutoriel-hibernate-introduction.html (consulté le 01/06/16)

[15] http://www.commentcamarche.net/contents/548-j2ee-java-2-enterprise-edition (consulté le 02/09/17)