



**ECOLE MAROCAINE DES
SCIENCES DE L'INGENIEUR**

Membre de **HONORIS UNITED UNIVERSITIES**

Application de Matching Compétences- Opportunités avec IA

Java Avancé / Programmation Orientée Objet

Ktami Lamiae
2025-2026

1.	Introduction Générale	2
1.1	Contexte	2
1.2	Problématique	2
1.3	Objectifs	2
2.	Analyse et Conception	2
2.1	Spécification des besoins	2
2.2	Conception UML (Diagramme de Classes)	2
2.3	Conception de la BD	3
3.	Environnement Technique	3
4.	Architecture et Implémentation	4
4.1	Architecture logicielle (Packages)	4
4.2	Design Patterns	4
4.3	Algorithme de Matching (Cœur du projet)	4
5.	Interface Utilisateur	4
5.1	Captures d'écran	4
6.	Conclusion	10
7.	Webographie	11

1. Introduction Générale

1.1 Contexte

Le recrutement moderne exige une rapidité et une précision accrues. Le tri manuel des CV est devenu inefficace face au volume de candidatures.

1.2 Problématique

Comment coupler efficacement les profils des candidats avec les exigences techniques des offres d'emploi pour réduire le temps de sélection ?

1.3 Objectifs

- Développer un moteur de matching basé sur les compétences.
 - Proposer une interface intuitive pour les recruteurs et les candidats.
 - Centraliser les données sur une base de données robuste.
-

2. Analyse et Conception

2.1 Spécification des besoins

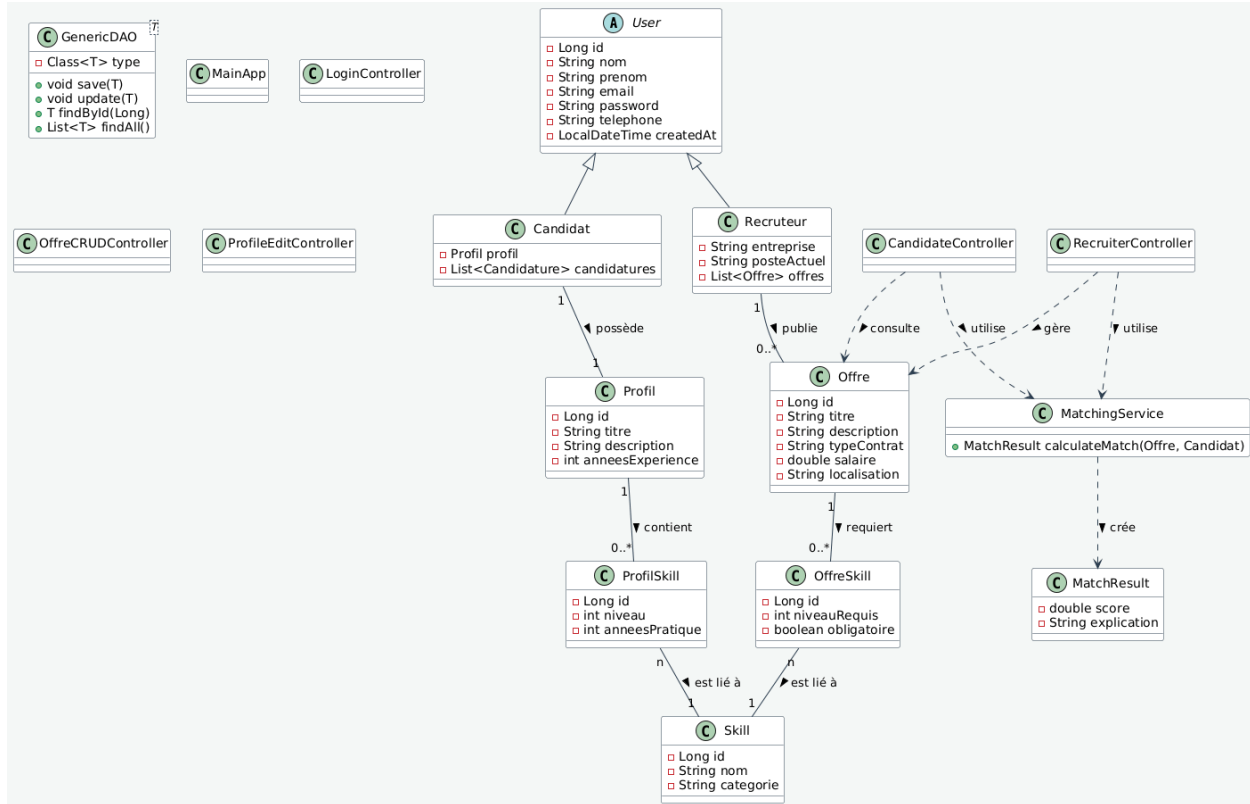
- **Besoin Fonctionnel** : Authentification, Gestion des offres (CRUD), Édition de profil, Calcul automatique du score de compatibilité.
- **Besoin Non-Fonctionnel** : Persistance des données (Hibernate), Interface premium (CSS custom), Architecture modulaire.

2.2 Conception UML (Diagramme de Classes)

Le système repose sur les entités suivantes :

- User (Classe mère) : Candidat et Recruteur s'en inspirent.
- Skill : Référentiel des compétences.
- Profil : Contient les ProfilSkill (compétence + niveau) du candidat.
- Offre : Contient les OffreSkill (compétence + niveau requis + obligation).

- **MatchResult** : Stocke le résultat du calcul (score, explication).



2.3 Conception de la BD

La base est structurée en 8 tables principales (users, candidats, recruteurs, profils, skills, offres, match_results, andidatures).

3. Environnement Technique

- **Langage** : Java 17 / 21
- **Architecture** : Maven (Gestion des dépendances)
- **SGBD** : PostgreSQL 15 via Docker
- **ORM** : Hibernate 6.4 (Mapping XML)
- **Interface** : JavaFX 17 + CSS Modern Theme
- **Conteneurisation** : Docker & Docker Compose

4. Architecture et Implémentation

4.1 Architecture logicielle (Packages)

- `com.iamco.model` : POJOs et fichiers de mapping Hibernate `.hbm.xml`.
- `com.iamco.dao` : Patron DAO générique pour l'accès aux données.
- `com.iamco.service` : `MatchingService` pour l'intelligence de calcul.
- `com.iamco.gui` : Contrôleurs FXML et gestion de la vue.

4.2 Design Patterns

- **Singleton** : Utilisé dans `HibernateUtil` pour garantir une seule `SessionFactory`.
- **Generic DAO** : Pour factoriser les méthodes de CRUD (`save`, `update`, `delete`, `findById`) pour toutes les entités.

4.3 Algorithme de Matching (Cœur du projet)

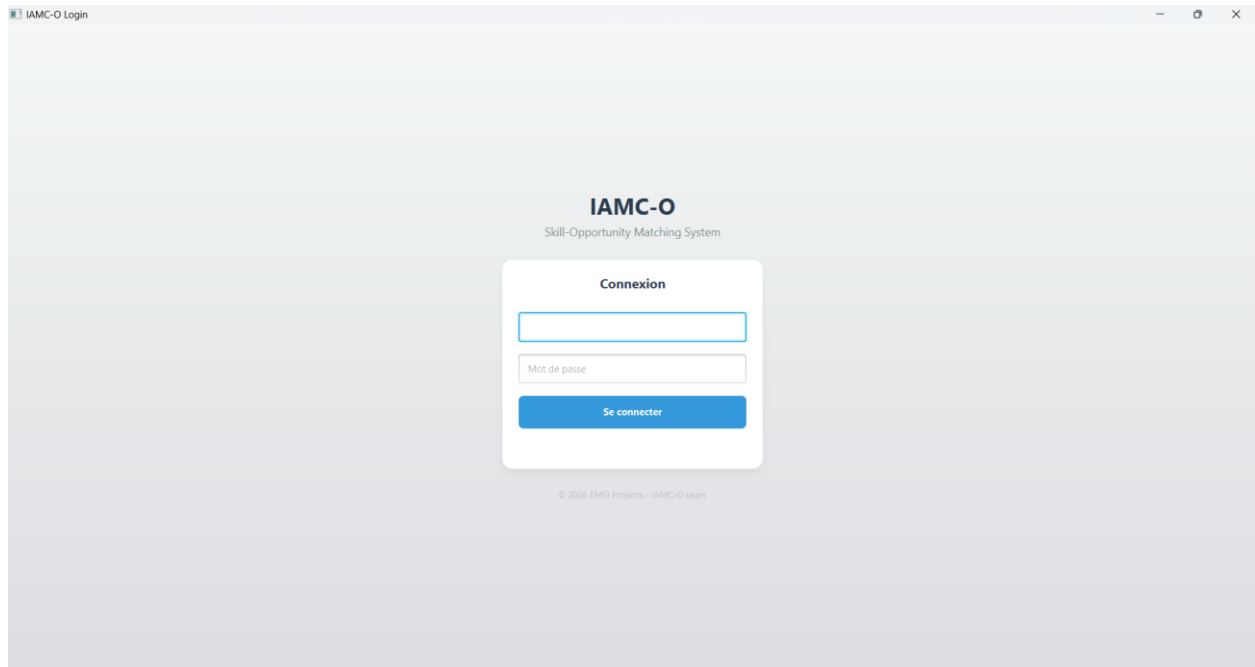
L'algorithme parcourt les compétences requises par une offre et les compare au profil du candidat.

- **Points** : 0.5 pt si la compétence existe, +0.5 pt si le niveau est suffisant.
 - **Pénalité** : Si une compétence est marquée comme **Obligatoire** et qu'elle manque, le score est fortement impacté.
-

5. Interface Utilisateur

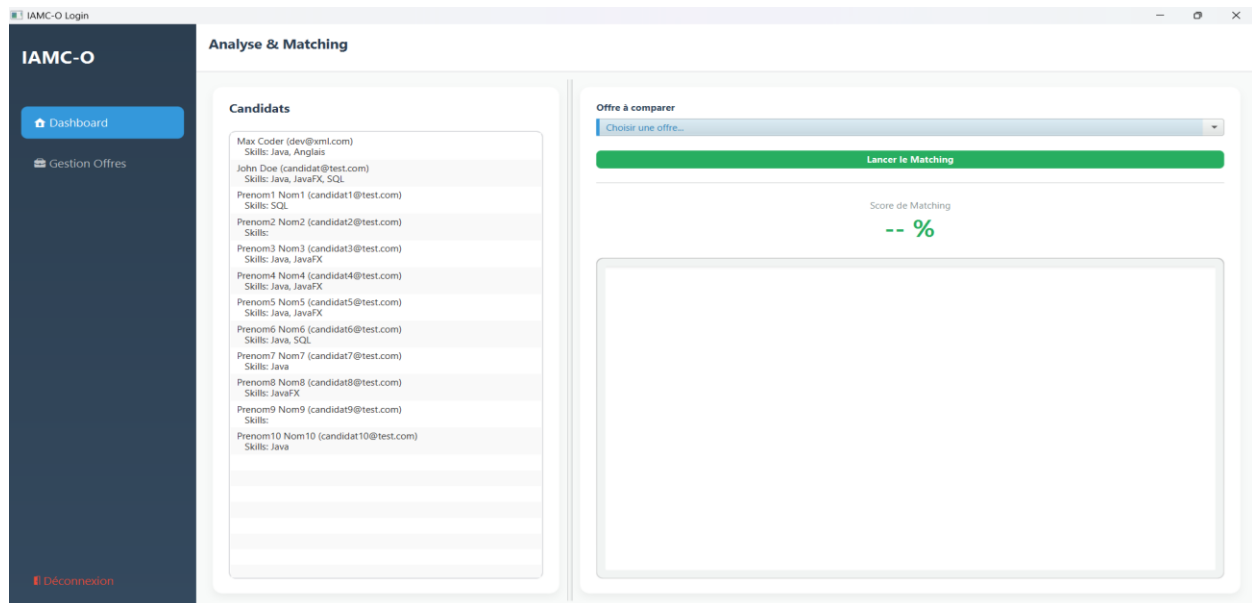
5.1 Captures d'écran

- **Login** : Interface épurée avec gestion des erreurs.



Espace Recruteur :

- **Dashboard** : Liste de matching en temps réel avec barre de score visuelle.

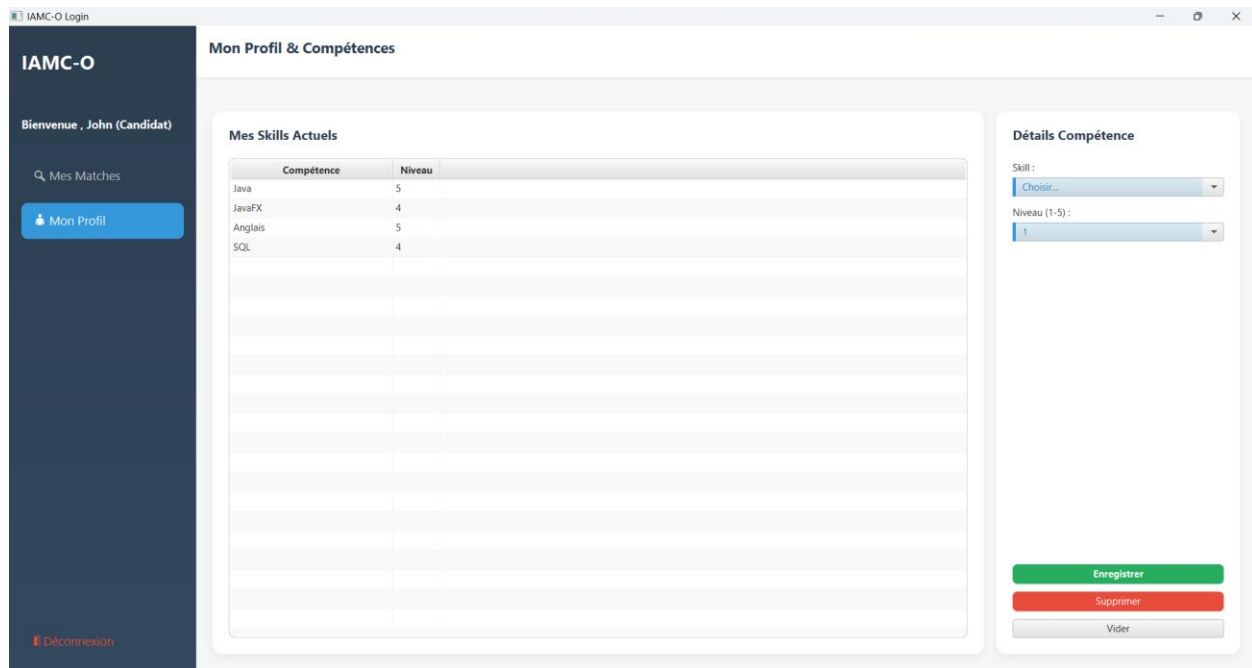


- [illegible]

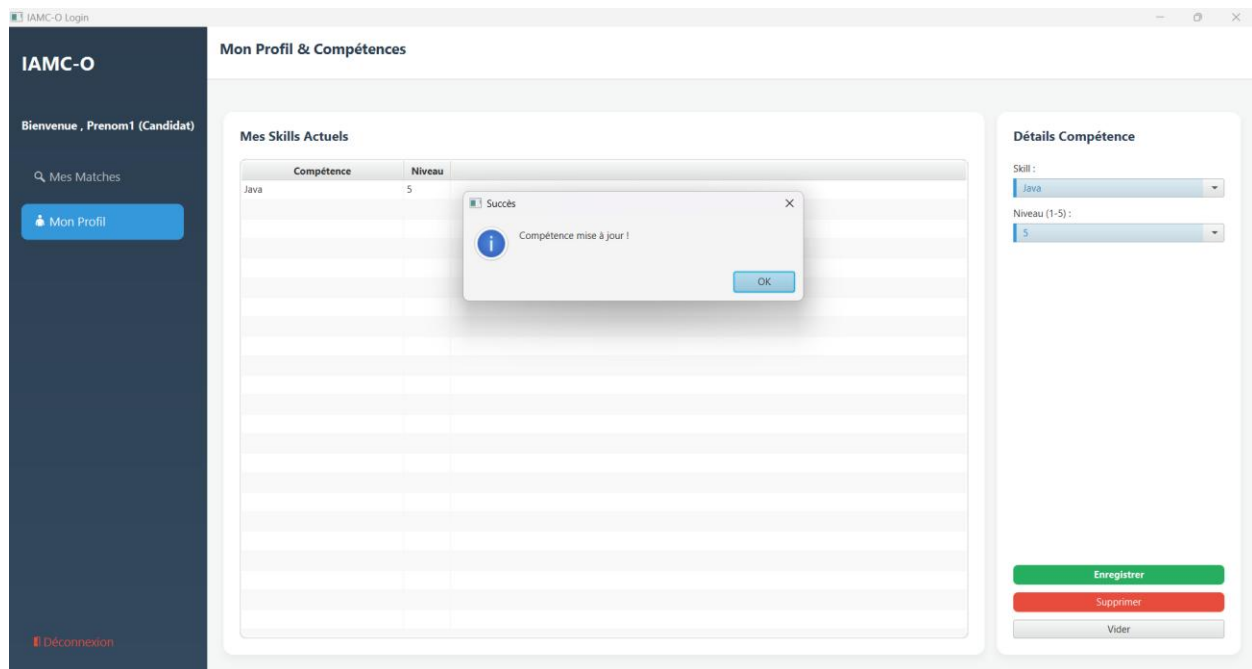
- **Mes Matches :** Affichage des opportunités avec calcul de score personnalisé.

6

- **Mon Profil** : Ajout et mise à jour des compétences avec niveau de maîtrise (1 à 5)



- **Test de Matching** : Vérification que l'ajout d'une compétence "Java" sur un profil augmente instantanément son score pour une offre "Dev Java".



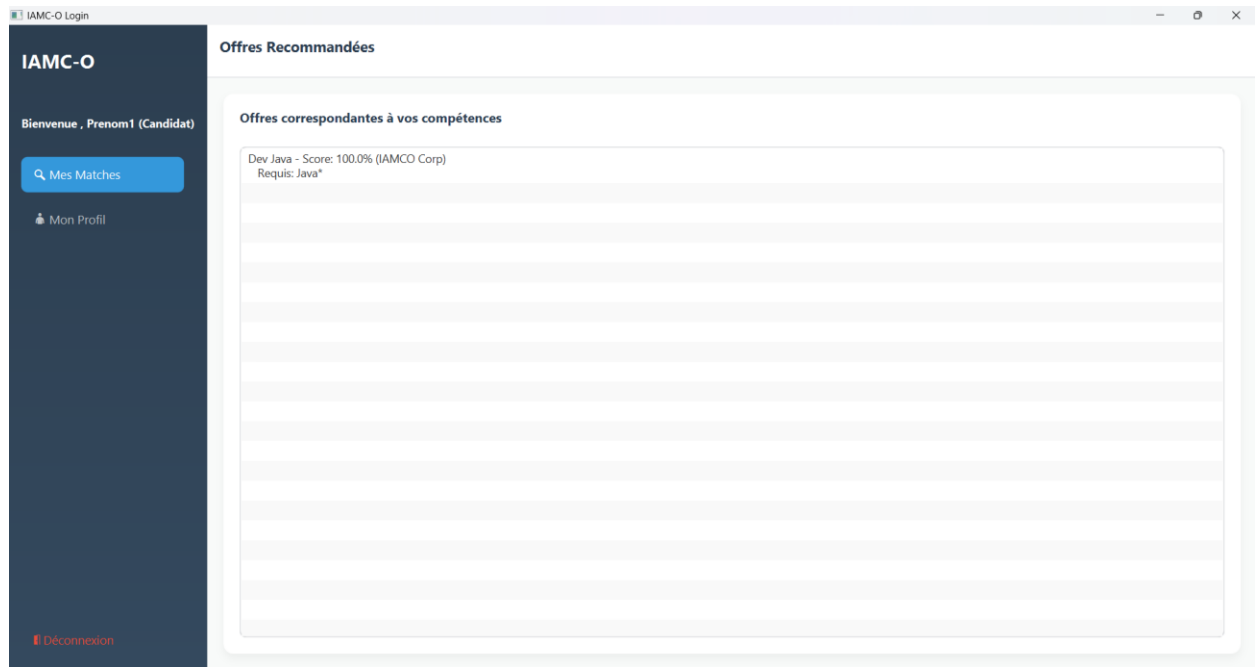
Ajout d'une nouvelle offer

The screenshot shows the 'Gestion des Offres d'Emploi' (Job Offer Management) interface. On the left is a dark sidebar with 'IAMC-O' and navigation links for 'Dashboard' and 'Gestion Offres'. The main area is titled 'Gestion des Offres d'Emploi' and contains a 'Liste des Offres' table with columns 'ID', 'Titre', and 'Salaire'. A modal dialog titled 'Succès' is displayed in the center, stating 'Offre ajoutée avec succès !' (Offer added successfully!) with an 'OK' button. On the right, the 'Détails de l'Offre' (Offer Details) form is visible, with fields for 'Titre' (Title), 'Salaire' (Salary), and 'Compétences (Skills)' (Skills). The 'Compétences' field is a multi-select dropdown showing 'Java', 'SQL', 'Anglais', 'JavaFX', 'Communication', 'Laravel', and '.NET Core'. At the bottom of the details panel are buttons for 'Enregistrer New' (Register New), 'Mettre à jour' (Update), 'Supprimer' (Delete), and 'Vider' (Clear).

Tester le score côté recruteur

The screenshot shows the 'Analyse & Matching' (Analysis & Matching) interface. The left sidebar is the same as the previous screen. The main area is titled 'Analyse & Matching' and is divided into two panels. The left panel, 'Candidats' (Candidates), lists candidates with their email, skills, and a selection status. The right panel, 'Offre à comparer' (Offer to compare), shows a dropdown menu with 'Dev Java' selected and a green 'Lancer le Matching' (Launch Matching) button. Below this, the 'Score de Matching' (Matching Score) is displayed as 'Score : 100.0 %' with a green progress bar. An 'Explication' (Explanation) box below the score states: 'Explication : Score: 100.0%. Analyse de 1 skills (1 obligatoires). Compétences matchées : Java (Niveau OK) Compétences manquantes : Aucun'.

Le score doit également s'afficher côté candidat.



6. Conclusion

Ce projet nous a permis de maîtriser l'écosystème Java avancé : de la gestion des données complexe avec Hibernate à la création d'interfaces réactives avec JavaFX. La principale difficulté résidait dans l'intégration GUI dans un environnement conteneurisé, ce qui a été résolu par une approche hybride (BD sous Docker / App sous Maven).

7. Webographie

- Documentation officielle Hibernate : [Hibernate. Everything data.](#)
- JavaFX OpenJFX Documentation : [JavaFX](#)