Licence Sciences du Numérique

Box Certificative finale

Sujet 2024 – 2025 : Clustering de Camarade

MOUSIN LUCIEN - WEINBERG BENJAMIN – ANTHONY QUERE

28/05/2025

Table des matières

Sujet d'examen : Clustering de Camarades

Ce sujet d'examen certifiant s'adresse aux étudiants de **Licence 3 Sciences du Numérique**. Vous êtes mis en situation professionnelle et devez développer, sur deux jours, une application complète répondant à un cahier des charges précis. Vous serez évalué sur votre capacité à utiliser vos compétences pour choisir les technologies appropriées, à maintenir une application, à proposer des solutions pertinentes et à développer un logiciel robuste.

Bien que le travail soit à réaliser en groupe, vous serez noté de façon individuelle. Il est important que l'ensemble du groupe connaisse la totalité du projet et ait une bonne connaissance du travail réalisé par chaque membre du groupe. Tout membre du groupe doit participer au développement de l'application, vous serez amené au cours de l'épreuve à écrire une partie du code de votre application devant un examinateur.

1. Contexte et Objectifs pédagogiques

1.1. Contexte

Dans le cadre de votre box certificative finale, vous devez réaliser une application dénommée **"Clustering de Camarades"**. Cette application a pour but de générer automatiquement des groupes optimaux de travail à partir des affinités renseignées par les étudiants.

1.2. Définitions

1.2.1.Affinité entre deux camarades

L'affinité entre deux étudiants se réfère au niveau de préférence ou de compatibilité exprimée par chacun lorsqu'ils indiquent leurs souhaits de collaboration. Vous devrez définir une mesure quantitative permettant de refléter le désir de travailler ensemble.

1.2.2.Répartition

La constitution des groupes de camarades doit être réalisée de sorte que chaque étudiant doit appartenir à un groupe, et qu'un étudiant ne peut pas appartenir à deux groupes en même temps.

1.2.3.Score d'une répartition

Il offre une vue d'ensemble de la qualité de la répartition et sert d'indicateur de la satisfaction globale quant à la prise en compte des affinités dans le processus de clustering.

Important : Il est indispensable de réfléchir en profondeur à la manière de calculer ce score pour optimiser la formation des groupes. Vous êtes invités à concevoir et justifier l'approche algorithmique qui vous semble la plus pertinente pour traduire ces notions en une solution concrète.

1.3. Objectifs pédagogiques

Vous serez évalué(e) sur l'ensemble de vos compétences techniques et méthodologiques.

- **Répondre à l'évolution constante de la technologie** en intégrant des outils et des technologies actuels.
- **Assurer la maintenance corrective d'une application informatique** en structurant un code modulable et documenté.
- **Préconiser une solution à un client ou collaborateur** en justifiant vos choix techniques et en proposant des améliorations potentielles.

- **Développer une application informatique** de bout en bout, depuis l'analyse des besoins jusqu'à la livraison finale.

1.4. Remarques

Le sujet présenté est conséquent et ambitieux, avec pour objectif que vous réalisiez le maximum de fonctionnalités pendant les deux jours alloués à l'épreuve. Il est entendu que, compte tenu du temps imparti, il se peut que toutes les fonctionnalités ne soient pas entièrement finalisées. Néanmoins, il est impératif que chaque partie du sujet soit abordée de manière à démontrer une utilisation minimale fonctionnelle de l'application.

Les correcteurs accorderont une attention particulière à la cohérence globale de la solution, à la qualité de l'architecture et au respect des contraintes imposées. Par ailleurs, il est essentiel que le groupe travaille en autonomie : la mise en commun ou le partage de code entre les groupes est formellement proscrit et toute similitude marquée entre les rendus sera sévèrement pénalisée.

En ce qui concerne le critère « Qualité de la documentation pour la maintenance » du BC3, par « Documentation complète », nous entendons une documentation détaillée sur **les parties critiques** de votre application.

2. Description fonctionnelle

L'application **"Clustering de Camarade"** doit répondre aux spécifications suivantes :

- **Authentification & Gestion des rôles :**
  - L'application doit proposer une interface d'authentification sécurisée avec gestion de rôles. Au moins deux types d'utilisateurs seront présents : **Étudiant** et **Enseignant**.
  - Vous êtes libre de choisir la méthode d'inscription qui vous semble la plus pertinente, de même pour l'affectation de rôles.
- **Saisie & Modification des affinités :**
  - Les étudiants peuvent renseigner ou modifier, en continu tant que l'application est ouverte, leurs affinités (c'est-à-dire les camarades avec lesquels ils souhaitent travailler).
- **Fermeture de la période de vote :**
  - La période de vote doit pouvoir être fermée, soit **manuellement** par un enseignant, soit **automatiquement** à l'atteinte d'une date prédéfinie.
- **Génération des Groupes :**
  - Un algorithme de clustering doit traiter les affinités saisies par les étudiants afin de constituer des groupes de travail optimaux.
  - Le critère d'optimisation repose un **score de satisfiabilité** qui reflète le degré de correspondance entre les préférences exprimées.
  - Un enseignant doit pouvoir (re)lancer le processus de clustering en spécifiant des contraintes supplémentaires (exemple : exclusion d'un étudiant dans le calcul en cas d'absence).
- **Publication des Groupes :**
  - La publication des groupes est paramétrée par l'enseignant.
  - Les groupes sont rendus publics et accessibles aux utilisateurs.

3. Contraintes techniques

Votre solution devra impérativement respecter les contraintes suivantes :

- **Langue :** L'ensemble du projet (code, documentation dans le code) doit être rédigé en anglais.
- **Langage/Technologie :** Le choix du langage de programmation et du framework est libre, à condition de justifier vos choix techniques dans la documentation. Aucune technologie ne sera avantagée lors de l'évaluation.
- **Algorithmique :** Un algorithme de clustering, optimisant un score de satisfiabilité, doit être conçu et implémenté.
- **Interface Homme-Machine (IHM) :** Vous devez réaliser une IHM fonctionnelle qui peut être sous forme de console, d'application web, de logiciel (desktop), ou autre. L'ergonomie et la facilité d'utilisation seront évaluées au regard de la technologie choisie.
- **Documentation technique :** Votre code doit être documenté dans un formalisme standard au regard de la technologie choisie.
- **Protocoles de tests :** Les fonctionnalités principales de l'application doivent faire l'objet de tests unitaires ou un protocole de tests détaillé.
- **Utilisation de Git :** La gestion de version est obligatoire. Le dépôt Git doit montrer des commits réguliers et bien documentés.
- **Mode d'Emploi :** Un guide d'utilisation (user guide) expliquant le fonctionnement de l'application doit être fourni. Fournissez également une documentation détaillée couvrant l'architecture de l'application, les choix technologiques et la description de l'algorithme de clustering.

4. Livrables

Vous devrez soumettre sur I-Campus, à la fin de l'épreuve une archive contenant les éléments suivants :

- **Code Source Complet**
- **Le dépôt Git :** Un fichier texte contenant un lien vers votre dépôt git, contenant l'historique complet des commits. Le dépôt doit être rendu public à la fin de l'épreuve.
- **Documentation technique :** Document détaillé incluant l'architecture du système, le choix des technologies, les *interprétations* retenues du sujet, le fonctionnement de l'algorithme, et les instructions d'installation et de mise en route.
- **Tests unitaires / Protocole de Tests :** Ensemble de tests (ou description précise du protocole de test) vérifiant les fonctionnalités principales de l'application.
- **Guide d'Utilisation (User Manual) :** Un document expliquant comment utiliser l'application (installation, connexion, saisie des affinités, lancement du clustering, etc.) selon le rôle (Etudiants, Enseignants, ...).

5. Déroulement de l'Épreuve

- **Durée :** L'épreuve est répartie sur deux jours consécutifs. Commence le 27 mai à 8h00 et se termine le 28 mai à 17h00. La période du 27 mai à 17h00 jusque le 28 mai à 8h00 n'est pas comptabilisée comme faisant partie de la durée de l'épreuve. Vous n'avez pas à travailler durant cette période.
- **Début de l'épreuve :** Le sujet et des consignes vous seront remis dès le début de l'épreuve.
- **Constitution des groupes :** Les groupes ne seront communiqués qu'au début de l'épreuve.
- **Accès aux Outils :** Vous aurez accès à l'ensemble des outils de développement habituels ainsi qu'aux ressources documentaires nécessaires, accès à internet.
- **Questions/Réponses :** Les examinateurs réaliseront des entretiens pour valider l'avancement du travail et la participation de chaque membre du groupe.
- **Démo :** Lors du 2e jour à partir de 16h, chaque membre d'un groupe devra réaliser une démonstration de l'application individuellement montrant l'état actuel de l'application.

6. Critères d'Évaluation

Votre projet sera évalué sur les aspects suivants :

- **Adaptabilité & Originalité :** Capacité à intégrer des technologies actuelles et à adapter l'application aux évolutions futures.
- **Qualité du Code et de l'Architecture :** Lisibilité, modularité, et efficacité de l'algorithme de clustering.
- **Gestion des données :** Pertinence de la structuration des données et intégration dans l'application.
- **Interface utilisateur :** Ergonomie et facilité d'utilisation de l'IHM au regard des technologies choisies.
- **Sécurité :** Un aspect sécuritaire de votre application sera observé.
- **Documentation et Tests :** Qualité de la documentation technique et du guide d'utilisation. Robustesse des tests unitaires ou du protocole de test.
- **Utilisation de Git :** Clarté et fréquence des commits, ainsi que l'organisation du dépôt.
- **Respect des Consignes linguistiques :** Rédaction de certaines parties du livrable en anglais.

7. Recommandations et Remarques

- **Planification :** Organisez votre travail dès le début pour respecter les délais. Pensez à définir une architecture modulaire pour faciliter la maintenance et l'évolution de l'application. **Faites des schémas**.
- **Documentation : Garder une trace de vos décisions**, cela vous permettra de **justifier vos choix** à spécifier dans la documentation. En ce qui concerne le critère « Qualité de la documentation pour la maintenance » du BC3, par « Documentation complète », nous entendons une documentation détaillée sur les parties *critiques* de votre application.
- **Tests :** Ne négligez pas les protocoles de tests. Ils seront essentiels pour valider la robustesse et la fiabilité de votre solution.
- **Gestion de Version :** Utilisez Git de manière régulière.

Grilles critériées

BCC1 – Répondre à l'évolution constante de la technologie

Critère	Non-Acquis	En cours d'acquisition	Acquis	Au-delà des attentes
C1.1. Réaliser une veille technologique et rechercher l'information	Aucun effort de veille n'est réalisé, et l'étudiant peine à identifier des sources fiables.	La veille technologique est réalisée de manière sporadique ; l'étudiant utilise quelques sources, mais l'analyse demeure superficielle.	Une veille régulière et méthodique est effectuée ; l'étudiant identifie et exploite des sources fiables pour enrichir le projet.	L'étudiant utilise une variété d'outils de recherche et intègre des informations de sources diversifiées. L'étudiant garde une trace de ses recherches.
C1.2. Choix et intégration des technologies	Les choix technologiques sont inadaptés ou obsolètes et ne répondent pas aux exigences du projet	Les choix technologiques sont pertinents à certains égards, mais leur intégration dans l'application est partielle ou mal optimisée.	Les technologies choisies sont cohérentes avec les besoins du projet et sont intégrées de manière efficace, en assurant une évolution et une adaptabilité.	L'étudiant démontre une compréhension poussée des technologies de pointe, propose des solutions alternatives innovantes et intègre ces technologies de manière optimisée et justifiée.
C1.3. Utilisation du langage technique en anglais	L'étudiant rencontre de grandes difficultés pour comprendre ou utiliser le vocabulaire technique en anglais, compromettant l'analyse de la documentation internationale.	L'exploitation du langage technique est partielle ; des erreurs de traduction ou d'interprétation apparaissent régulièrement.	L'étudiant maîtrise le vocabulaire technique en anglais, ce qui lui permet de comprendre et d'intégrer efficacement des documentations.	L'étudiant excelle en anglais technique, et est capable d'adapter et vulgariser des concepts complexes.
C1.4. Répondre au besoin avec la bonne solution technologique	L'étudiant envisage une unique solution.	L'étudiant constate que plusieurs solutions sont possibles, mais ne sait pas justifier son choix ou se retrouve dans l'incapacité de faire un choix.	L'étudiant propose plusieurs solutions et sait justifier son choix.	L'étudiant propose plusieurs solutions et sait justifier son choix, de plus, l'étudiant mesure les répercussions à long terme des choix.

BCC3 – Assurer la maintenance corrective d'une application informatique

Critère	Non-Acquis	En cours d'acquisition	Acquis	Au-delà des attentes
C3.1. Identification et résolution des bugs	L'étudiant ne parvient pas à détecter ni à corriger les erreurs, entraînant une application instable ou non fonctionnelle.	Certains bugs sont identifiés et résolus, mais des problèmes critiques persistent ou des solutions sont temporaires.	L'étudiant identifie efficacement les dysfonctionnements et met en place des corrections pertinentes, assurant une stabilité correcte de l'application.	L'étudiant anticipe les problèmes potentiels, propose des solutions robustes et documente minutieusement le processus de résolution, garantissant une maintenance préventive et corrective exemplaire.
C3.2. Qualité de la documentation pour la maintenance	Aucune documentation n'est fournie.	La documentation couvre partiellement les aspects essentiels, mais manque de clarté ou de détails sur certains points critiques.	La documentation est complète, structurée et accompagnée de schémas permettant une compréhension et une prise en main efficace pour la maintenance corrective.	La documentation est très détaillée, incluant des diagrammes et des explications approfondies qui facilitent non seulement la maintenance, mais aussi la montée en compétences des équipes futures.
C3.3. Réactivité et proactivité dans la maintenance	L'étudiant ne montre aucune réactivité face aux erreurs ou aux besoins de maintenance.	L'étudiant réagit aux problèmes une fois signalés, mais ne met pas en place de stratégie pour anticiper les dysfonctionnements futurs.	L'étudiant corrige les erreurs de manière efficace et montre une capacité d'anticipation raisonnable des problèmes récurrents.	L'étudiant met en place des mécanismes de monitoring et de maintenance préventive, assurant ainsi une gestion proactive de la stabilité de l'application.
C3.4. Qualité des protocoles de tests	Aucune démarche de tests n'est mise en place.	Quelques tests sont réalisés, mais ils couvrent peu de cas et manquent de rigueur.	Un ensemble pertinent de tests est mis en œuvre, garantissant la fiabilité des fonctionnalités principales de l'application.	L'étudiant met en place une stratégie de tests complète qui permet d'assurer une robustesse maximale et facilite la maintenance future.
C3.5. Gestion des erreurs côté utilisateur (IHM)	Aucune gestion des messages d'erreurs pour l'utilisateur de l'application	Des messages d'erreurs sont présents, mais peu clairs.	Des messages d'erreurs clairs et précis sont envoyés aux utilisateurs de l'application.	Les messages d'erreurs clairs et précis sont envoyés aux utilisateurs de l'application et un logeur est mis en place pour les administrateurs de l'application

BCC4 – Préconiser une solution à un client ou un collaborateur

Critère	Non-Acquis	En cours d'acquisition	Acquis	Au-delà des attentes
C4.1. Analyse des besoins et adéquation de la solution proposée	L'étudiant ne parvient pas à identifier les besoins essentiels du client ou propose une solution qui ne correspond pas aux attentes.	Une analyse des besoins est réalisée de manière partielle, avec une solution proposée qui répond en partie aux exigences.	L'étudiant réalise une analyse complète des besoins et propose une solution adéquate, répondant de manière cohérente aux attentes du client ou collaborateur.	L'analyse est approfondie et l'étudiant anticipe des besoins non exprimés, proposant une solution innovante et sur-mesure, accompagné d'une argumentation solide et structurée.
C4.2. Clarté et efficacité de la communication à un client	La communication est confuse, incomplète ou ne parvient pas à convaincre le client.	La communication est claire sur certains points, mais manque de précision ou de détails essentiels pour justifier les choix.	L'étudiant communique de manière claire et structurée, expliquant les choix techniques et fonctionnels de manière convaincante.	La communication est non seulement claire, mais également persuasive et adaptée aux différents interlocuteurs (technique et non-technique), renforçant la crédibilité de la solution proposée.
C4.3. Clarté et efficacité de la communication à un collaborateur	Absence de communication avec ses collaborateurs.	L'étudiant à une communication restreinte lors des prises de décisions. Ne participe pas activement à la vie globale du projet.	L'étudiant participe activement aux prises de décisions.	L'étudiant est force de proposition lors des prises de décisions et s'assure de la bonne compréhension des objectifs intermédiaires.
C4.4. Capacité à proposer des alternatives et à justifier les choix	Absence de proposition, se contente d'accomplir les solutions qu'on lui soumet sans chercher à comprendre les tenants et aboutissants.	L'étudiant propose une solution unique sans envisager d'alternatives, ou ses justifications sont insuffisantes.	Quelques alternatives sont évoquées, mais les justifications restent partielles.	L'étudiant présente plusieurs alternatives viables, justifie clairement ses choix et explique pourquoi la solution retenue est la plus appropriée.
C4.5. Organisation de l'équipe et gestion de la collaboration	L'étudiant ne sait pas expliquer son rôle ou les décisions prises collectivement	L'étudiant connaît son rôle dans l'équipe, mais ne sait pas ce que les autres font exactement	L'étudiant justifie sa contribution et connaît celle de ses camarades.	L'étudiant connaît son rôle et celui de ses camarades, et encourage l'utilisation des outils collaboratifs de façon stratégique.

BCC5 – Développer une solution informatique

Critère	Non-Acquis	En cours d'acquisition	Acquis	Au-delà des attentes
C5.1. Qualité du code (lisibilité, modularité, robustesse)	Le code est difficile à lire, non structuré et présente de nombreuses erreurs affectant la robustesse de l'application.	Le code est fonctionnel, mais manque de modularité ou de clarté, rendant la maintenance et l'évolution plus complexes.	Le code est bien structuré, lisible et modulaire, permettant une maintenance aisée et assurant la robustesse de l'application.	De plus, le code utilise des design patterns pertinents, démontrant une grande maîtrise des bonnes pratiques de développement.
C5.2. Respect des contraintes fonctionnelles et techniques	L'application ne répond pas aux contraintes de base définies dans le cahier des charges.	Certaines contraintes sont respectées, mais l'ensemble du projet présente des incohérences ou des lacunes importantes.	L'application répond de manière satisfaisante à toutes les contraintes fonctionnelles et techniques, avec une implémentation cohérente et complète.	En plus de respecter les contraintes, l'étudiant propose des améliorations ou des fonctionnalités additionnelles pertinentes, renforçant la valeur ajoutée du projet.
C5.3. Gestion de projet et utilisation d'outils de versioning (Git)	L'utilisation de Git est absente ou très insuffisante, sans historique de commits structuré.	Git est utilisé, mais de manière sporadique ou avec un historique de commits peu détaillé et non structuré.	L'étudiant utilise Git de manière régulière avec des commits clairs et bien documentés, témoignant d'une bonne gestion de projet.	La gestion de version est exemplaire, avec un historique riche, des branches clairement définies et une utilisation avancée des outils collaboratifs, reflétant une organisation de projet optimale.
C5.4. Pertinence et efficacité algorithmique	L'étudiant ne sait pas développer une solution algorithmique à un problème donné.	L'étudiant propose une solution globalement fonctionnelle, mais qui n'est pas optimisée ou pertinente. Il peine à analyser ou justifier clairement ses choix algorithmiques.	L'étudiant développe une solution algorithmique adaptée au problème posé, en justifiant clairement ses choix. La solution proposée est efficace et correctement optimisée.	L'étudiant propose une solution algorithmique efficace et il sait justifier son efficacité en donnant sa complexité algorithmique.