**Rapport Rendu n°1 :**

**Rapport sur l’utilité et le fonctionnement de notre modèle Entité/Association (E/A) avec Looping et WorkBench**

**Introduction**

Dans le cadre de notre projet, nous avons conçu un modèle Entité/Association (E/A) afin de structurer et d’organiser efficacement les données de notre système d’information. Nous avons utilisé l’outil Looping pour concevoir ce modèle et MySQL WorkBench pour générer et exécuter les scripts SQL correspondants, permettant ainsi d’implémenter notre base de données.

**Utilité du modèle E/A**

Notre modèle E/A joue un rôle fondamental dans la conception de notre base de données relationnelle, car il nous permet de :

* Définir clairement les entités principales et leurs relations.
* Assurer la cohérence et l’intégrité des données.
* Optimiser la structure de la base pour de meilleures performances.
* Faciliter la communication entre les membres de notre équipe.

**Fonctionnement avec Looping**

Nous avons utilisé Looping pour concevoir notre schéma E/A en suivant ces étapes :

1. Définition des entités : Nous avons identifié les objets clés du système et leurs attributs spécifiques.
2. Création des associations : Nous avons défini les relations entre les entités en respectant les cardinalités (0-1, 0-n, 1-1, 1-N, N-N).
3. Attribution des clés primaires et étrangères : Cela garantit l’intégrité référentielle entre les tables de notre base de données.
4. Exportation du modèle : Une fois le modèle finalisé, nous avons exporté le script SQL pour sa mise en œuvre.

**Génération et utilisation du script SQL dans WorkBench**

Après la conception du modèle avec Looping, nous avons intégré le script SQL dans MySQL WorkBench en suivant ces étapes :

1. Importation du script SQL : Nous avons exécuté le script afin de créer les tables et les relations dans notre base de données MySQL.
2. Exécution et tests : Nous avons ensuite effectué des requêtes SQL pour vérifier la cohérence et l’intégrité des données.
3. Optimisation : Nous avons ajouté des index et des contraintes supplémentaires pour améliorer les performances du système.

**Conclusion**

Grâce à l’utilisation conjointe de Looping et MySQL WorkBench, nous avons pu concevoir et implémenter notre base de données de manière structurée et efficace. Le modèle E/A que nous avons réalisé nous a permis d’assurer une meilleure organisation des données et une implémentation optimisée grâce aux scripts SQL générés. Cette approche nous aide à garantir la cohérence du système, la performance globale de notre application et une gestion optimale des données.

**Rapport sur les prompts clés liés à la visualisation du graphe en C#**

**Introduction**

**Ce rapport met en avant les prompts les plus pertinents concernant la visualisation du graphe en C#. Il détaille les problématiques rencontrées et les solutions mises en place pour améliorer l'affichage et l'interaction utilisateur sous Windows Forms.**

**1. Implémentation de la visualisation du graphe**

**Prompt clé :**

*"Passons directement à l'implémentation de la visualisation du graphe. Attention, j'aurai sûrement besoin d'explications très détaillées et d'être guidé."*

**Problématique et solution :**

* Création d'une interface graphique Windows Forms.
* Utilisation de System.Drawing pour représenter les nœuds sous forme de cercles et les liens sous forme de lignes.
* Ajout d'un événement Paint pour rafraîchir l'affichage à chaque mise à jour du graphe.

**2. Amélioration de la disposition des nœuds**

**Prompt clé :**

*"Comment optimiser un maximum la répartition des nœuds et comment avoir un affichage agréable à regarder ? "*

**Problématique et solution :**

* Vérification de la lecture des données et correction des erreurs d'instanciation des liens.
* Mise en place de OrganiserEnCercle() pour répartir les nœuds de manière uniforme autour d'un point central.
* Amélioration de l'affichage des liaisons incorrectes en mettant à jour les références des liens après le repositionnement des nœuds.

**3. Ajout d'interactions avec des boutons**

**Prompt Clé :**

*"Nous avons maintenant un code fonctionnel auquel j'aimerais ajouter des boutons sur la visualisation Windows Forms sur lesquels je pourrai cliquer ou survoler et qui me diraient si le graphe est connexe ou s'il contient des circuits."*

**Problématique et solution :**

* Ajout de boutons interactifs (Button) pour tester la connectivité et la présence de cycles.
* Affichage des résultats dans un Label dynamique pour informer l'utilisateur.
* Liaison des boutons avec EstConnexe() et ContientCycle() via des EventHandler.

**4. Débogage sous Windows Forms**

**Prompt clé :**

*"Je suis sur Windows Forms et je n'ai pas accès à la console, comment voir les logs ?"*

**Problématique et solution :**

* **Proposition de quatre méthodes pour afficher les logs :** 
  1. MessageBox.Show() pour afficher des pop-ups informatives.
  2. Ajout d’un Label sur l’interface pour afficher les dernières informations en temps réel.
  3. Écriture des logs dans un fichier debug\_log.txt pour une analyse après exécution.
  4. Utilisation de Debug.WriteLine() pour afficher les logs dans l’onglet Sortie de Visual Studio.

**Conclusion et améliorations futures**

**La mise en place d'une visualisation efficace du graphe a permis d'améliorer la compréhension et l'interactivité du projet. Les améliorations possibles incluent :**

* L’ajout d’une interface plus dynamique permettant de déplacer les nœuds.
* Une meilleure gestion des superpositions pour éviter le chevauchement des liens.
* L’intégration d’une bibliothèque de visualisation plus avancée comme SkiaSharp ou GraphSharp.

**Rapport sur les Tests Uniatires**

**Introduction**

L'objectif des tests unitaires est de s'assurer du bon fonctionnement des différents méthodes de la classe Graphe. Ces méthodes permettent respectivement d'ajouter des nœuds et des liens entre eux dans un graphe.

Les Test sur les méthodes :

**Difficulté rencontrée**

* Une erreur Une référence d'objet est requise a été détectée, indiquant que graphe n'était pas correctement instancié.
* Correction en s'assurant que l'objet Graphe est bien initialisé avant chaque test.

Une fois que cette difficulté a été surmonté, les tests ont très bien fonctionné et nous ont permis de bel et bien vérifier que l’ajout des nœuds et des liens se faisaient effectivement bien.

**Conclusion**

Les tests réalisés permettent de valider le bon fonctionnement de différentes méthodes de la classe Graphe. Malgré quelques erreurs initiales liées à l’instanciation de graphe, les tests ont été corrigés et permettent d’assurer une meilleure fiabilité du code.