

Développement d'une Application C# pour le Diagnostic Automatique des Maladies du Coeur

Présenté par: **Sidik Klaa**



Plan

- Problématique
- Objectifs
- Méthodologie
- Modélisation UML
- L'Algorithme d'Arbre de Décision
- Gestion des Données et Bases de Données
- Évaluation
- Résultats
- Conclusion

Problématique:

- Les maladies cardiaques représentent la première cause de mortalité mondiale, avec environ 17,9 millions de décès annuels.
- Un diagnostic précoce et précis peut augmenter significativement les chances de survie.
- Les méthodes traditionnelles peuvent manquer les signes subtils et nécessitent souvent des interprétations subjectives.
- Manque d'accès à l'équipement spécialisé et aux experts, surtout dans les zones rurales ou défavorisées.
- Les processus actuels peuvent être lents, retardant le traitement et affectant les résultats de santé.

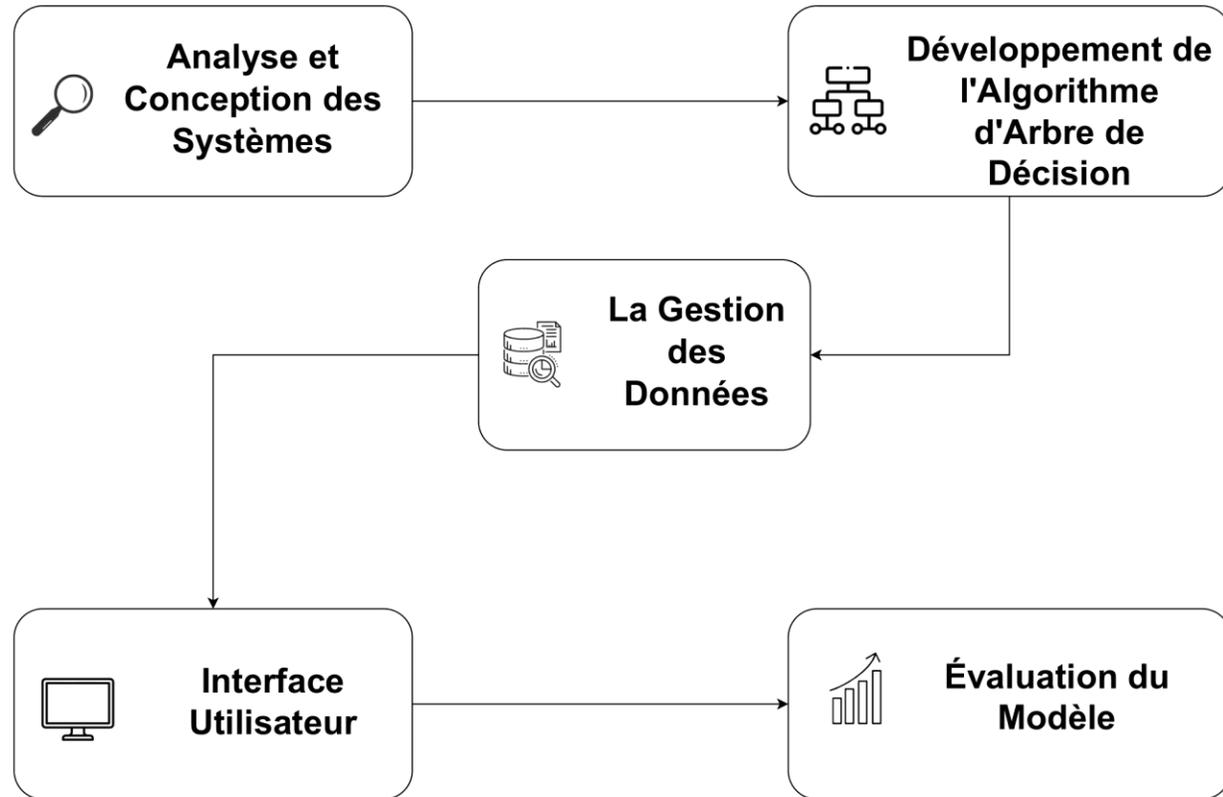




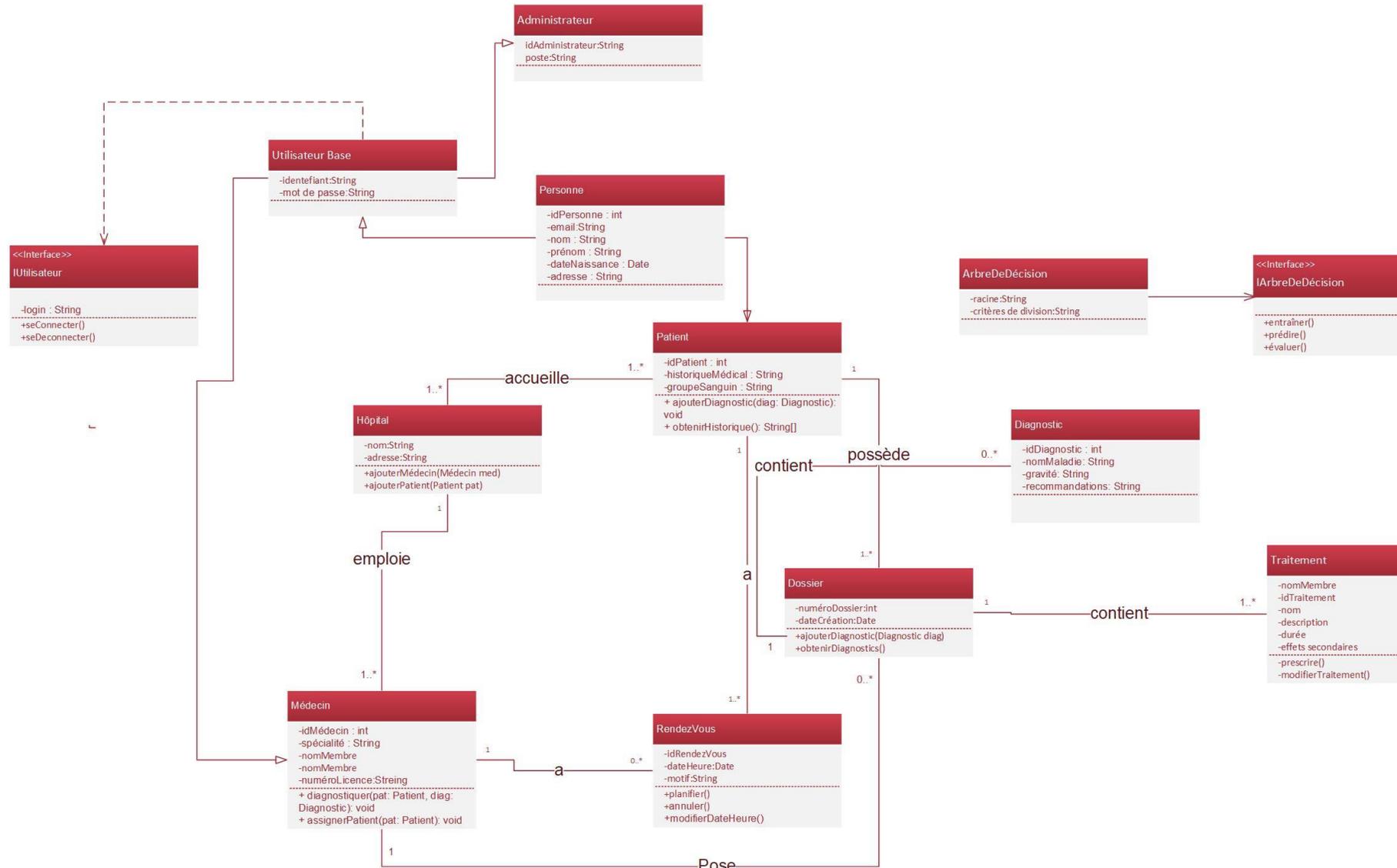
Objectifs:

- Développer une application graphique en C# capable de diagnostiquer de manière automatique les maladies cardiaques, en utilisant des algorithmes d'apprentissage automatique pour offrir des résultats précis et fiables.
- Utiliser des méthodes avancées comme l'arbre de décision pour améliorer la précision des diagnostics, minimisant les erreurs humaines et subjectives.
- Intégrer des fonctionnalités pour la gestion des dossiers des patients et la coordination avec les médecins, améliorant ainsi le suivi des cas.

Méthodologie



Modélisation UML



L'Algorithme d'Arbre de Décision

```
public interface IDecisionTree
{
    3 références
    bool IsNumericAttribute(List<Diagnostic> data, string attributeName);
    6 références
    double CalculateEntropy(List<Diagnostic> data);
    2 références
    double CalculateInformationGain(List<Diagnostic> data, string attributeIndex);
    2 références
    double CalculateInformationGainNumeric(List<Diagnostic> data, string attributeName, out double? splitValue);
    2 références
    bool GetMostCommonClass(List<Diagnostic> data);
    2 références
    string GetBestAttribute(List<Diagnostic> data, List<string> attributes, out double? splitValue);
    2 références
    List<object> GetAttributeValues(List<Diagnostic> data, string bestAttribute);
    6 références
    Noeud BuildTree(List<Diagnostic> data, List<string> attributes);
    3 références
    void AfficherArbre(Noeud noeud, string indentation = "");
    4 références
    string Classify(Diagnostic instance);
    0 références
    string Classify(Diagnostic instance, Noeud node);
    2 références
    float Evaluate(List<Diagnostic> testData);
    2 références
    void ConfusionMatrix(List<Diagnostic> testData);
}
```

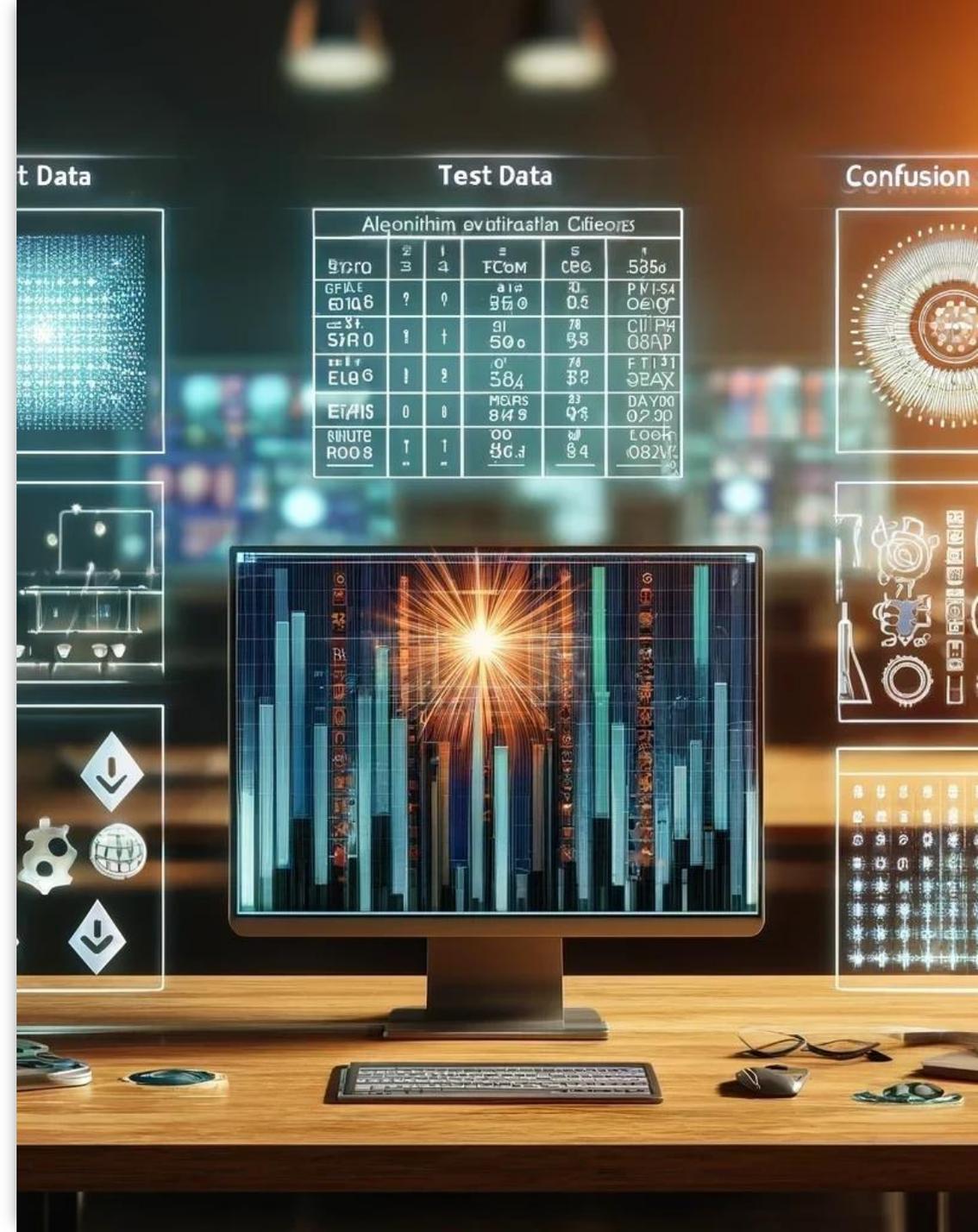
Gestion des Données et Bases de Données

- **Structure des Données:**
 - Données d'Apprentissage : Pour entraîner l'algorithme.
 - Données de Test : Pour évaluer la précision de l'algorithme.
- **Technologie de Base de Données:**
 - SQL Server : Choix pour sa robustesse et sa compatibilité avec .NET.
 - Entity Framework Core : Utilisé pour simplifier les opérations CRUD et gérer les relations entre tables.



Évaluation

- **Données de Test** : Utilisation des données de test pour évaluer la précision de l'algorithme.
- **Métriques d'Évaluation** : Précision, sensibilité, spécificité.
- **Matrice de Confusion** : Pour visualiser la performance de l'algorithme sur différentes classes de diagnostic.



Résultats

Fichier Client

Accueil Patient Apprentissage et Évaluation Prédiction

CP 2.0

THAL 1.0

CA 0.0

OLDPEAK 160

THALACH 1

Prédire

La prédiction n'indique pas la présence d'une maladie cardiaque.

OK

Taux de Reconnaissance:

0.9230769

Matrice de Confusion:

	Prédit Non	Prédit Oui
Vrai Non (Actual)	22	3
Vrai Oui (Actual)	1	26

Conclusion



Développement réussi d'une application C# pour le diagnostic automatique des maladies cardiaques, utilisant des arbres de décision.



Amélioration notable de la précision et de la rapidité des diagnostics médicaux.



Merci pour votre attention

