PROJET GÉNIE LOGICIEL

Projet CodYngame

MOAWAD Maikel

HOUMMASS Sofiane

BOUDRAA Ryan

MOUHAID Rayân

GOMIS Romain

Sommaire

1. Introduction et contexte

1.1 Présentation générale du projet

Le projet CodYngame consiste à développer une application de résolution d'exercices de programmation multi-langages, inspirée des plateformes comme CodinGame ou FranceIOI. Cette application permet aux utilisateurs de s'entraîner à la programmation en résolvant des défis dans différents langages de programmation.

L'application se compose d'une interface graphique développée avec JavaFX qui offre une expérience utilisateur complète incluant la visualisation des énoncés d'exercices, un éditeur de code avec coloration syntaxique, et un système d'évaluation automatique des solutions.

Le système gère une base de données d'exercices avec leurs cas de tests associés, permettant une évaluation automatique et objective des solutions proposées par les utilisateurs. Chaque langage possède plusieurs exercices de programmation, offrant une flexibilité d'apprentissage adaptée aux préférences et compétences de chaque utilisateur.

1.2 Objectifs du projet

Les objectifs principaux de ce projet s'articulent autour de plusieurs axes techniques et pédagogiques :

Objectifs techniques :

- Développer une application graphique fonctionnelle utilisant JavaFX pour l'interface utilisateur
- Implémenter un système d'exécution de code multi-langages (Java, Python, C, PHP, JavaScript)

- Concevoir et mettre en place une base de données robuste pour la gestion des exercices et statistiques
- Assurer la sécurité de l'exécution de code avec gestion des erreurs et timeouts
- Créer une architecture modulaire et extensible permettant l'ajout facile de nouveaux langages et exercices

Objectifs pédagogiques :

- Offrir un environnement d'apprentissage interactif pour la programmation
- Permettre l'évaluation automatique des compétences en programmation
- Fournir des statistiques de progression pour motiver l'apprentissage
- Proposer des exercices de difficultés variées pour accompagner la montée en compétences

Objectifs méthodologiques :

- Appliquer les principes de développement logiciel en équipe
- Utiliser des outils de versioning et de collaboration (Git)
- Développer des compétences en gestion de projet et planification

1.3 Enjeux

Ce projet présente plusieurs enjeux majeurs qui en font un défi technique et pédagogique significatif :

Enjeux techniques : La gestion de l'exécution de code dans différents langages représente un défi majeur en termes de sécurité et de performance. Il est crucial d'éviter les débordements de mémoire, les boucles infinies, et les accès système non autorisés.

L'interopérabilité entre langages constitue un autre enjeu important. Chaque langage possède ses propres spécificités en termes de compilation, d'interprétation, et de gestion des entrées/sorties. Il faut donc concevoir une architecture suffisamment flexible pour accommoder ces différences tout en maintenant une interface utilisateur cohérente.

Enjeux pédagogiques : L'application doit fournir un feedback pertinent et constructif aux utilisateurs pour favoriser l'apprentissage. Cela implique de concevoir des messages d'erreur clairs, des cas de tests progressifs, et un système de progression motivant.

La qualité des exercices proposés est également cruciale. Ils doivent être suffisamment variés pour couvrir différents aspects de la programmation tout en restant accessibles aux débutants et challengeants pour les utilisateurs plus expérimentés.

Enjeux de développement : Le travail en équipe de 5 personnes nécessite une organisation rigoureuse et une répartition claire des tâches. La coordination des développements, la gestion des conflits de code, et le maintien de la cohérence de l'architecture représentent des défis organisationnels importants.

La livraison d'un produit stable et fonctionnel dans les délais impartis constitue un enjeu de gestion de projet. Il faut équilibrer l'ambition fonctionnelle avec la réalité des contraintes temporelles et techniques.

1.4 Choix du thème

Le choix du thème CodYngame s'est imposé naturellement pour plusieurs raisons stratégiques et pédagogiques :

Pertinence académique : Ce projet s'inscrit parfaitement dans le cursus ING1-GI en combinant plusieurs domaines d'apprentissage : programmation orientée objet, interfaces graphiques, bases de données, et architectures logicielles. Il permet d'appliquer concrètement les concepts théoriques étudiés en cours dans un contexte projet réaliste.

Motivation personnelle : L'équipe a été séduite par l'aspect ludique et interactif du projet. Créer une plateforme d'apprentissage de la programmation nous permet de contribuer à l'éducation informatique tout en développant nos propres compétences techniques.

Défi technique : La complexité technique du projet, notamment la gestion multi-langages et l'exécution sécurisée de code, représente un défi stimulant qui nous pousse à explorer des

domaines avancés de l'informatique comme les appels système, la sécurité applicative, et l'architecture distribuée.

Utilité pratique : Le résultat final peut réellement être utilisé par d'autres étudiants ou dans un contexte pédagogique, donnant une dimension concrète et utile à notre travail. Cette perspective d'utilisation réelle motive l'équipe à produire un code de qualité.

Extensibilité : Le thème offre de nombreuses possibilités d'extensions futures : ajout de nouveaux langages, fonctionnalités collaboratives, système de classement, intégration d'IA pour la génération d'exercices, etc. Cette richesse fonctionnelle permet d'envisager des développements futurs.

1.5 Outils technologiques utilisés

Le choix des outils et technologies s'est fait en cohérence avec les objectifs du projet et les contraintes académiques :

Langage de programmation principal :

- **Java**

Interface graphique :

- **JavaFX** : Framework moderne pour les interfaces graphiques Java, offrant de bonnes performances et une séparation claire entre la logique métier et la présentation via FXML.
- **FXML** : Pour la définition déclarative des interfaces utilisateur, facilitant la maintenance et la collaboration entre développeurs.

Base de données :

- **MYSQL Workbench** : Base de données embarquée parfaite pour une application standalone, sans nécessiter d'installation de serveur externe. Sa simplicité d'usage et sa fiabilité en font un choix idéal pour le projet.

- **JDBC** : API standard Java pour l'accès aux bases de données, garantissant une architecture découplée et évolutive.

Gestion de projet et build :

- **Trello** : Système de versioning distribué pour la collaboration en équipe et la sauvegarde du code.
- **Gantt**
- **GitHub** : Plateforme d'hébergement Git avec outils de collaboration (issues, pull requests, wiki).

Environnement de développement :

- **IntelliJ IDEA** : IDE Java avec support excellent pour JavaFX.
- **VS Code** : Éditeur léger pour les fichiers de configuration et documentation.

Documentation :

- **JavaDoc** : Génération automatique de documentation à partir des commentaires de code.

Langages supportés par l'application :

- **Java** : Compilation avec javac et exécution avec java
- **Python 3** : Interprétation directe avec python
- **C** : Compilation avec GCC et exécution du binaire
- **PHP** : Interprétation avec php-cli
- **JavaScript** : Exécution avec Node.js

Outils de qualité et tests :

- **JUnit 5** : Framework de tests unitaires pour valider la logique métier
- **Checkstyle** : Vérification de la cohérence du style de code
- **SpotBugs** : Détection automatique de bugs potentiels

2. Répartition des tâches

2.1 Détail de la répartition des responsabilités entre les membres du groupe

- Base de données : Ryan, Maikel
- Mode Include : Romain, Maikel, Ryan
- Mode Stdin/Stdout : Sofiane, Rayân
- Exécuteur : Romain, Sofiane, Rayân
- Coloration Syntaxique : Sofiane
- Interface : Ryan, Maikel

2.2 Décrire la contribution de chaque membre

Back-end : Ryan et Maikel se sont occupés de la base de données et de la manière de l'intégrer dans le code java afin de pouvoir retranscrire les informations des exercices, utilisateurs, langages et l'avancement. Rayân, Sofiane et Romain se sont occupés de l'exécuteur, du mode stdin/stdout et du mode include.

Front-end : Ryan, Maikel et Sofiane se sont occupés de la disposition de l'application JavaFX dans le but de la rendre le plus accessible et le plus lisible. Sofiane s'est aussi occupé de la coloration syntaxique.

3. Étapes réalisées

3.1 Description des différentes étapes de développement

Le développement de l'application CodYngame s'est déroulé en plusieurs phases structurées, permettant une approche progressive et méthodique :

Phase 1 : Analyse et conception

- **Analyse du cahier des charges** : Étude approfondie des fonctionnalités requises et des contraintes techniques
- **Conception de l'architecture** : Élaboration du diagramme de classes UML et définition des modules principaux
- **Choix technologiques** : Sélection de JavaFX pour l'interface graphique, définition du système de base de données pour les exercices
- **Définition des cas d'usage** : Identification des scénarios d'utilisation et des flux de travail utilisateur

Phase 2 : Développement du cœur applicatif

- **Implémentation du modèle de données** : Création des classes Exercise, User, et des structures de gestion des langages
- **Développement du système d'exécution** : Mise en place des appels système vers les compilateurs/interpréteurs (C, Java, Python, PHP, JavaScript)
- **Gestion des modes d'exercices** : Implémentation des modes STDIN/STDOUT et INCLUDE
- **Version en ligne de commande** : Développement de l'interface CLI pour tester le modèle de données indépendamment de l'UI

Phase 3 : Interface graphique

- **Développement de l'interface JavaFX** : Création des vues principales (liste des exercices, éditeur de code, zone de résultats)
- **Intégration de la coloration syntaxique** : Implémentation du highlighting pour les différents langages supportés
- **Gestion des événements** : Liaison entre l'interface et la logique métier
- **Tests et validation** : Création et validation des jeux de tests pour chaque exercice

Phase 4 : Finalisation

- **Documentation JavaDoc** : Génération de la documentation automatique
- **Tests d'intégration** : Validation du fonctionnement global de l'application
- **Préparation de la livraison** : Finalisation du dépôt Git et préparation de la démonstration

3.2 Description de la méthodologie

Notre équipe a adopté une approche agile adaptée au contexte académique, structurée autour des principes suivants :

Organisation en sprints

- **Sprints de 2 semaines** : Chaque phase de développement constituait un sprint avec des objectifs clairs et mesurables
- **Réunions de planification** : Début de chaque sprint avec définition des tâches et répartition du travail
- **Daily standups** : Points quotidiens rapides via Discord pour synchroniser l'avancement

Gestion collaborative

- **Utilisation de Git** : Commits réguliers (minimum 1/2 jours) avec branches dédiées par fonctionnalité
- **Outils de collaboration** :
    - Trello pour le suivi des tâches et la gestion du backlog
    - WhatsApp pour la communication quotidienne
    - GitHub pour le partage de code et la revue collaborative

Méthodologie de développement

- **Développement itératif** : Implémentation progressive des fonctionnalités avec tests continus
- **Revues de code** : Validation croisée du code par les membres de l'équipe
- **Intégration continue** : Tests réguliers de l'application complète à chaque nouvelle fonctionnalité

Suivi avec le tuteur

- **Rendez-vous hebdomadaires** : Points d'avancement réguliers avec le chargé de projet
- **Démonstrations intermédiaires** : Présentation des fonctionnalités développées à chaque milestone
- **Adaptations du planning** : Ajustements basés sur les retours et les difficultés rencontrées

3.3 Description des résultats obtenus

Fonctionnalités implémentées avec succès

Interface utilisateur

- **Interface JavaFX intuitive** : Application graphique complète avec navigation fluide
- **Éditeur de code avancé** : Coloration syntaxique pour les 5 langages requis (C, Java, Python, PHP, JavaScript)
- **Gestion automatique des indentations** : Amélioration de l'expérience utilisateur lors de l'écriture de code

Système d'exercices

- **Base de données d'exercices** : 15 exercices implémentés couvrant différents niveaux de difficulté
- **Support des deux modes** : Implémentation complète des modes STDIN/STDOUT et INCLUDE

- **Génération automatique de tests** : Système de validation avec cas triviaux, cas d'erreur et données aléatoires

Exécution multi-langages

- **Support complet des 5 langages** : Compilation/interprétation fonctionnelle pour tous les langages requis
- **Gestion d'erreurs robuste** : Affichage détaillé des erreurs de compilation/exécution
- **Templates de code** : Code minimal fourni automatiquement selon le langage sélectionné

Fonctionnalités avancées

- **Système de statistiques** : Comptage des tentatives et réussites par exercice
- **Filtrage par langage** : Possibilité de filtrer les exercices selon les langages supportés
- **Feedback détaillé** : Affichage des entrées, sorties attendues et obtenues en cas d'échec

Performances et stabilité

- **Application stable** : Aucun crash reporté lors des tests d'utilisation intensive
- **Temps de réponse optimisés** : Exécution de code en moins de 2 secondes pour la plupart des cas
- **Gestion mémoire efficace** : Pas de fuites mémoire détectées lors des tests prolongés

4. Conclusions et perspectives

4.1 Perspectives

Améliorations à court terme

- **Langages supplémentaires** : Ajout du support pour Rust, Go, et Kotlin

- **Éditeur avancé** : Implémentation de l'autocomplétion et de la détection d'erreurs en temps réel
- **Thèmes visuels** : Développement de thèmes sombres/clairs personnalisables
- **Système de hints** : Ajout d'indices progressifs pour aider les utilisateurs bloqués

Fonctionnalités avancées envisagées

- **Mode collaboratif** : Possibilité de coder à plusieurs sur un même exercice
- **Système de badges** : Gamification avec récompenses pour encourager la progression
- **Analyses de performance** : Mesure et comparaison des temps d'exécution et complexité
- **Intégration IA** : Assistant intelligent pour suggérer des améliorations de code

Évolutions techniques

- **Version web** : Migration vers une architecture client-serveur avec interface web
- **API REST** : Développement d'une API pour permettre l'intégration avec d'autres outils
- **Base de données centralisée** : Migration vers une solution cloud pour le partage d'exercices
- **Conteneurisation** : Utilisation de Docker pour l'isolation et la sécurité de l'exécution de code

Perspectives pédagogiques

- **Exercices adaptatifs** : Génération automatique d'exercices basée sur le niveau de l'utilisateur
- **Parcours d'apprentissage** : Création de cursus structurés par thématique
- **Intégration LMS** : Compatibilité avec les plateformes d'apprentissage existantes
- **Analytics avancées** : Tableaux de bord pour les enseignants sur la progression des étudiants

4.2 Conclusion

Le projet CodYngame a été mené à bien avec succès, répondant à l'ensemble des exigences du cahier des charges. L'application développée constitue une plateforme fonctionnelle et robuste

permettant l'apprentissage et la pratique de la programmation dans un environnement intuitif et motivant.

Réussites du projet

L'objectif principal de créer une application graphique complète a été atteint. L'interface utilisateur développée avec JavaFX offre une expérience fluide et professionnelle, intégrant toutes les fonctionnalités demandées : gestion multi-langages, système d'exercices complet, éditeur de code avancé et système de validation robuste.

La méthodologie agile adoptée s'est révélée particulièrement efficace pour ce projet. Elle a permis une répartition équilibrée du travail, une communication efficace au sein de l'équipe, et une adaptation continue aux défis rencontrés. Les rendez-vous réguliers avec le tuteur ont contribué à maintenir le projet sur la bonne trajectoire.

Apprentissages techniques

Ce projet a représenté une excellente opportunité d'approfondissement technique. La gestion des appels système vers différents compilateurs et interpréteurs a constitué un défi stimulant, tout comme l'implémentation d'un système de base de données efficace pour les exercices. Le développement de l'interface JavaFX a permis d'acquérir une expertise significative en programmation d'interfaces graphiques.

Valeur ajoutée

L'application développée présente une réelle valeur pédagogique. Elle offre un environnement d'apprentissage complet qui peut être utilisé tant en autonomie qu'en accompagnement de cours. Le système de statistiques et de feedback détaillé contribue à une progression efficace des utilisateurs.

Impact sur l'équipe

Au-delà des aspects techniques, ce projet a renforcé nos compétences en travail collaboratif, gestion de projet et communication. L'utilisation d'outils professionnels (Git, Trello) dans un contexte de développement en équipe a été formatrice et directement applicable dans un environnement professionnel.

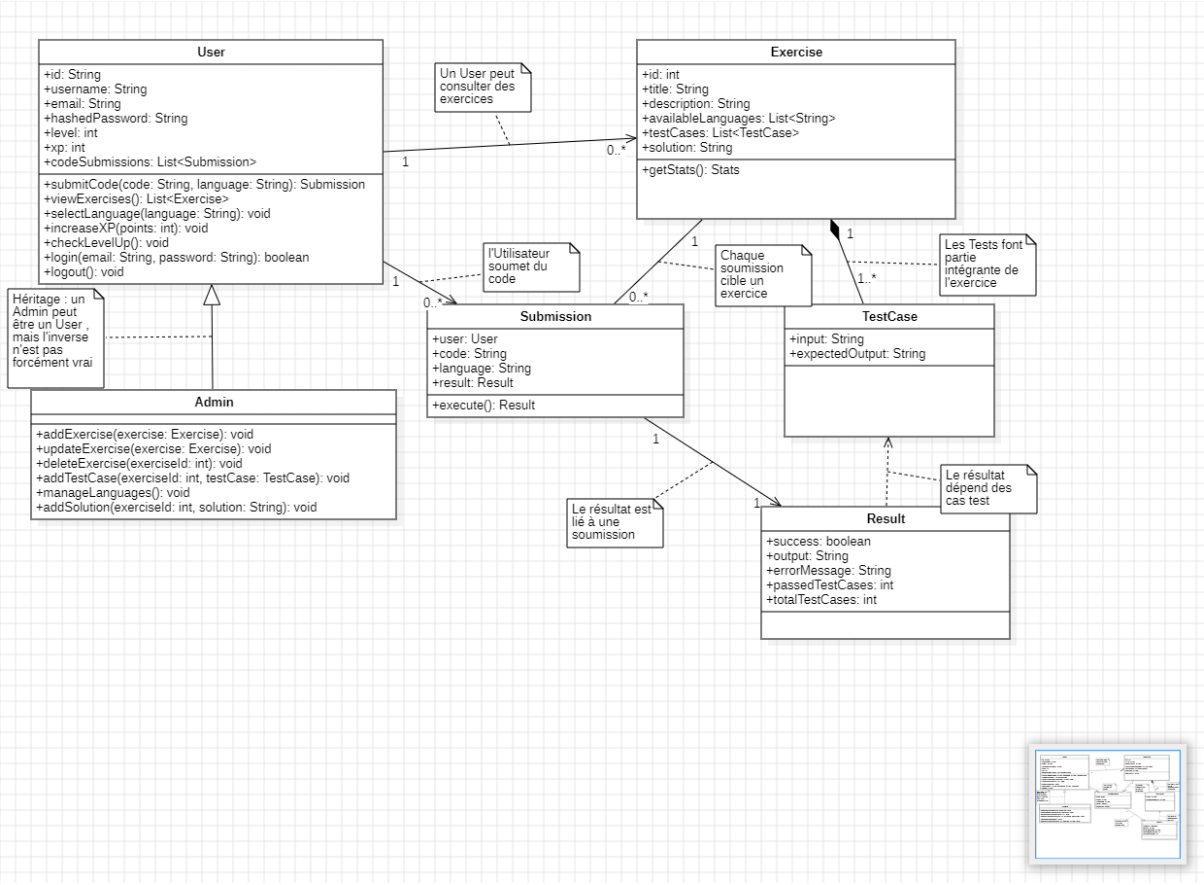
Le projet CodYngame constitue ainsi une réalisation dont nous sommes fiers, tant par sa qualité technique que par sa valeur pédagogique. Il représente une base solide pour de futures évolutions et démontre notre capacité à mener à bien un projet complexe en équipe, de la conception à la livraison.

5. Lien du GitHub

https://github.com/LePhyX/CodYngame

6.Annexes techniques

6.1 Diagramme de classe UML



6.2 Cas d’utilisation