**Rapport Projet : Architecture des composants d’entreprise**

**Moubarik Abdelmoughit, Jarmouni Basma, Abenay Khalid**

**Introduction :**

**1-Aperçu du projet**

L'application Suivi de Budget réinvente la gestion des finances personnelles en offrant une solution complète, combinant simplicité et puissance. Notre interface conviviale permet aux utilisateurs de suivre leurs dépenses en temps réel, de définir des budgets personnalisés et d'obtenir une analyse approfondie de leur situation financière.

En adoptant une approche moderne de développement basée sur des microservices, nous utilisons le microservice `mbk-ba-auth` pour l'authentification, assurant un accès sécurisé à l'application. Le microservice `mbk-ba` prend en charge la gestion des budgets et des utilisateurs, offrant des fonctionnalités avancées. De plus, il permet aux utilisateurs d'accéder à leur budget personnel de manière transparente.

Nous avons également mis en place le microservice `mbk-ba-bu` pour la gestion des catégories de budgets et pour assurer un suivi détaillé de l'historique des dépenses. Cette fonctionnalité permet aux utilisateurs de mieux comprendre leurs habitudes de dépenses et d'optimiser leur planification financière.

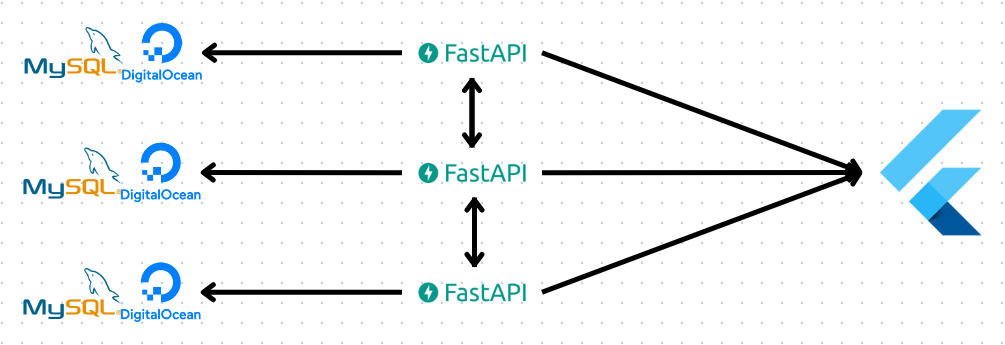
Grâce à la containerisation avec Docker, une intégration continue avec Jenkins, l'analyse de la qualité du code via SonarQube, et un hébergement robuste sur DigitalOcean, nous garantissons une expérience utilisateur fluide, sécurisée et performante.

**2-Importance de l'architecture microservices**

L'architecture microservices est devenue de plus en plus importante dans le domaine du développement logiciel en raison de plusieurs avantages qu'elle offre par rapport aux architectures monolithiques traditionnelles.

**Architecture Microservices :**

**1-Architecture**

****

**2-Description des services**

Microservice mbk-ba-auth (Authentification) :

Le microservice mbk-ba-auth est spécifiquement dédié à garantir une authentification sécurisée au sein de l'application Suivi de Budget, en utilisant des JSON Web Tokens (JWT) pour valider l'identité des utilisateurs. Il assure que seuls les utilisateurs autorisés ont accès à leurs données financières personnelles. La gestion fine des identités renforce la confidentialité des informations tout en offrant une expérience utilisateur fluide et sécurisée.

Microservice mbk-ba (Gestion des Budgets et des Utilisateurs) :

Ce microservice constitue le cœur de l'application Suivi de Budget, prenant en charge la gestion complète des budgets et des utilisateurs.il permet un accès transparent aux budgets personnels des utilisateurs.

Microservice mbk-ba-bu (Gestion des Catégories de Budgets et Historique des Dépenses) :

Spécialisé dans la gestion des catégories de budgets, le microservice mbk-ba-bu offre une granularité exceptionnelle dans la classification des dépenses. Il assure également un suivi détaillé de l'historique des dépenses, permettant aux utilisateurs de comprendre en profondeur leurs habitudes financières. Cette fonctionnalité avancée facilite l'optimisation des plans budgétaires en identifiant les tendances de dépenses.

**3-Mécanismes de communication**

Chaque microservice exposera une API via FastAPI avec ses propres endpoints.

Les endpoints peuvent être définis à l'aide des décorateurs FastAPI, indiquant les opérations supportées (GET, POST, etc.).

Les microservices peuvent communiquer entre eux via des requêtes HTTP. FastAPI facilite cela en permettant l'utilisation des méthodes HTTP standard (GET, POST, PUT, DELETE, etc.) pour effectuer des opérations.

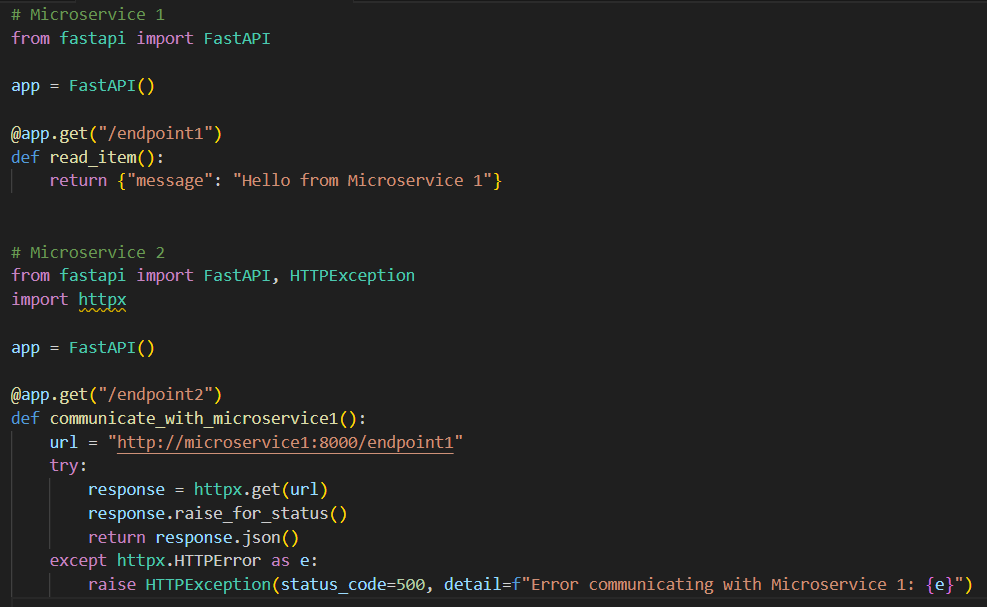
FastAPI génère automatiquement une documentation interactive (via Swagger) qui facilite la compréhension et l'exploration de l'API par les développeurs.

Assurez-vous que la documentation inclut les informations nécessaires sur la manière de consommer chaque microservice.

FastAPI prend en charge les opérations asynchrones à l'aide de async et await. Cela peut être utilisé pour des opérations asynchrones, par exemple, lors de l'appel d'un microservice distant.

Utilisation de bibliothèques telles que httpx pour effectuer des requêtes HTTP de manière asynchrone.

exemple :



**Conception des Microservices :**

La conception des microservices pour l'application Suivi de Budget implique la division des fonctionnalités en trois microservices distincts : `mbk-ba-auth`, `mbk-ba`, et `mbk-ba-bu`. Chaque microservice est conçu pour répondre à des besoins spécifiques, tout en interagissant avec les autres pour former une application cohérente. Voici une conception générale de ces microservices :

Microservice `mbk-ba-auth` (Authentification) :

Responsabilités :

1. Authentification et Autorisation :

- Gère l'authentification des utilisateurs.

- Fournit des tokens JWT pour valider l'identité des utilisateurs.

- Gère les autorisations d'accès aux ressources.

2. Gestion des Utilisateurs :

- Stocke et gère les informations d'identification des utilisateurs.

Architecture et Technologies :

- Framework : Utilise un framework léger adapté à la gestion d'authentification, comme Flask (Python).

- Stockage : Utilise une base de données (par exemple, PostgreSQL) pour stocker les informations des utilisateurs et les tokens JWT.

- Sécurité : Implémente des protocoles de sécurité robustes, tels que OAuth 2.0, pour assurer la confidentialité et l'authenticité.

Microservice `mbk-ba` (Gestion des Budgets et des Utilisateurs) :

Responsabilités :

1. Gestion des Utilisateurs :

- Gère les profils des utilisateurs et leurs informations personnelles.

- Permet l'inscription, la modification, et la suppression de comptes.

2. Gestion des Budgets :

- Prend en charge la création, la modification et la suppression de budgets.

- Fournit un accès sécurisé aux données budgétaires des utilisateurs.

Architecture et Technologies :

- Framework : Utilise un framework robuste pour le développement de l'API, tel que FastAPI (Python).

- Stockage : Utilise une base de données adaptée à la gestion des utilisateurs et des budgets.

- Communication : Interagit avec `mbk-ba-auth` pour garantir l'authentification avant l'accès aux ressources sensibles.

Microservice `mbk-ba-bu` (Gestion des Catégories de Budgets et Historique des Dépenses) :

Responsabilités :

1. Gestion des Catégories de Budgets :

- Permet la création, la modification et la suppression de catégories de budgets.

- Classe les dépenses dans des catégories spécifiques.

2. Historique des Dépenses :

- Maintient un historique détaillé des dépenses et des recettes des utilisateurs.

- Fournit des analyses et des tendances basées sur l'historique.

Architecture et Technologies :

- Framework : Utilise un framework adapté à la gestion détaillée des catégories de budgets, tel que Django (Python).

- Stockage : Utilise une base de données pour stocker l'historique des dépenses et les catégories de budgets.

- Communication : Interagit avec `mbk-ba` pour obtenir des informations sur les utilisateurs et leurs budgets.

Cette conception favorise la scalabilité, la flexibilité et la maintenance indépendante des microservices. Chaque microservice est conçu pour être autonome, avec une interface claire pour la communication entre eux. L'utilisation de protocoles de sécurité avancés, comme OAuth 2.0, et la mise en œuvre d'une architecture de microservices bien pensée contribuent à la création d'une application robuste et évolutive.

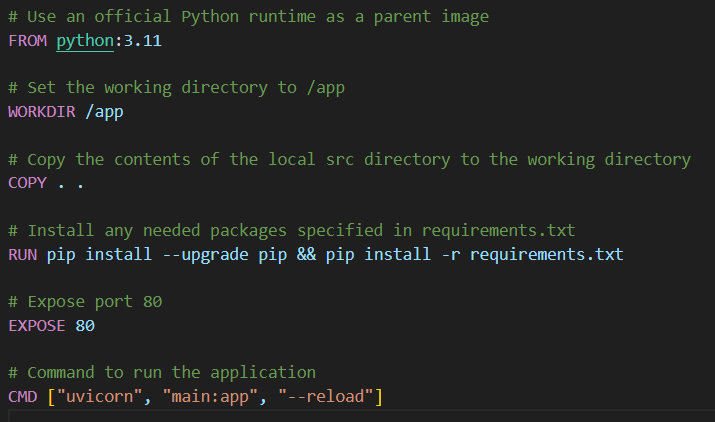
**Conteneurisation avec Docker :**

La conteneurisation avec Docker est une pratique courante dans le développement logiciel moderne. Docker fournit une plateforme ouverte qui permet aux développeurs de créer, déployer et exécuter des applications dans des conteneurs. Voici une brève explication de l'implémentation de la conteneurisation avec Docker et de ses avantages.

**Implémentation de la Conteneurisation avec Docker:**

Un fichier Dockerfile est utilisé pour décrire les étapes nécessaires pour construire une image Docker.

Il spécifie l'environnement d'exécution, les dépendances et le code de l'application.



**Avantages de la Conteneurisation avec Docker :**

**Isolation**:

Les conteneurs fournissent une isolation légère en encapsulant l'application et ses dépendances. Chaque conteneur a son propre système de fichiers, processus, et espace mémoire.

**Portabilité**:

Les conteneurs fonctionnent de manière cohérente sur différents environnements, garantissant que l'application se comporte de la même manière, qu'elle soit exécutée localement, sur un serveur de développement ou dans le cloud.

**Facilité de Déploiement:**

La création d'images Docker standardisées facilite le déploiement sur différentes infrastructures, y compris les serveurs locaux, les clusters Kubernetes, et les services cloud.

**Rapidité**:

Les conteneurs sont légers et démarreront rapidement, permettant des déploiements et des mises à l'échelle rapides.

Gestion des Dépendances:

Les dépendances de l'application sont encapsulées dans le conteneur, évitant ainsi les conflits et les problèmes liés aux versions de bibliothèques.

**Évolutivité**:

Les applications conteneurisées peuvent être facilement mises à l'échelle horizontalement en ajoutant ou retirant des instances de conteneurs.

**Gestion des Versions:**

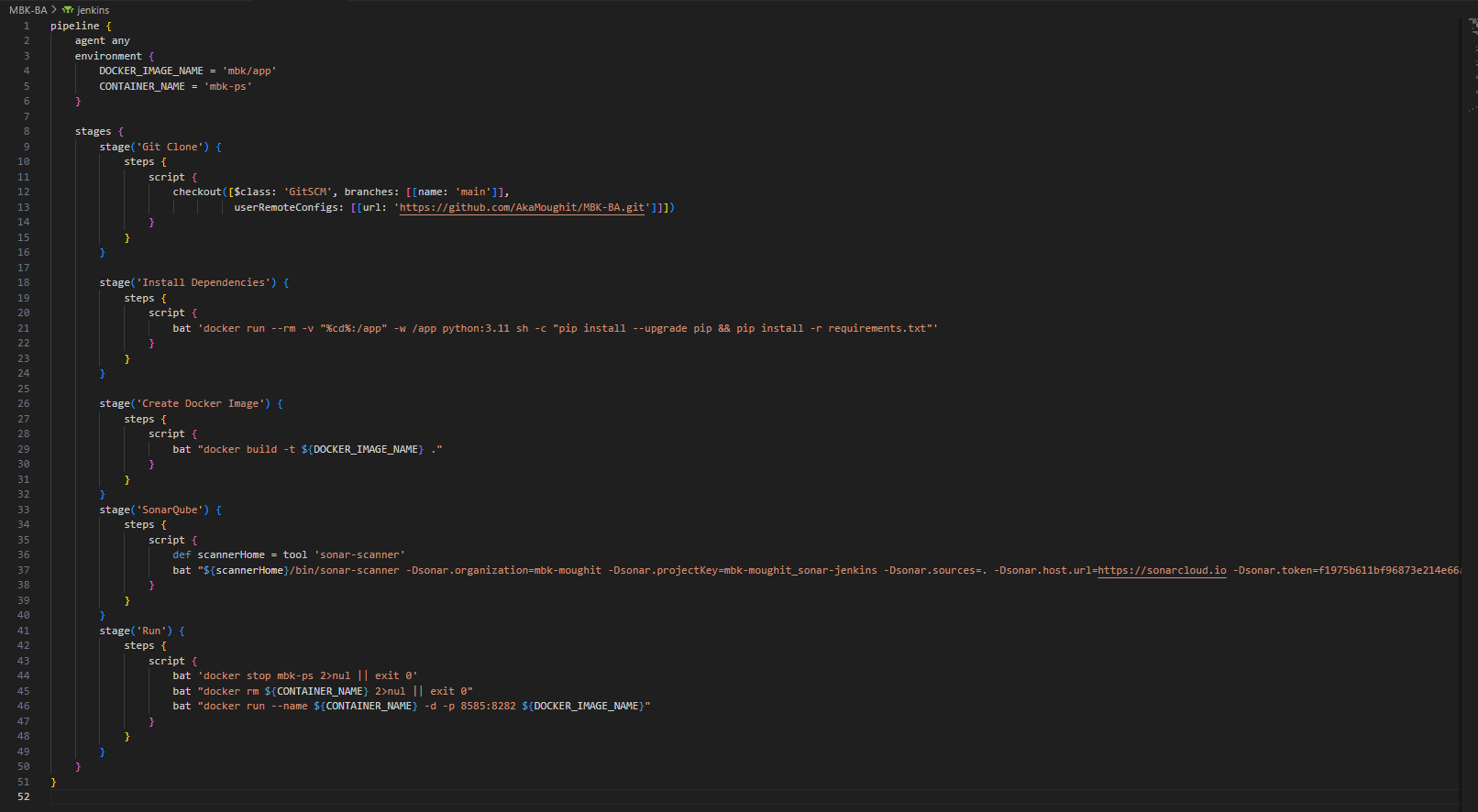
Les images Docker permettent de gérer les versions de l'application de manière plus efficace, facilitant le suivi des changements et les rollbacks en cas de besoin.

**Écosystème Étendu:**

Docker a un vaste écosystème d'outils qui simplifient le développement, le déploiement et la gestion des applications conteneurisées.

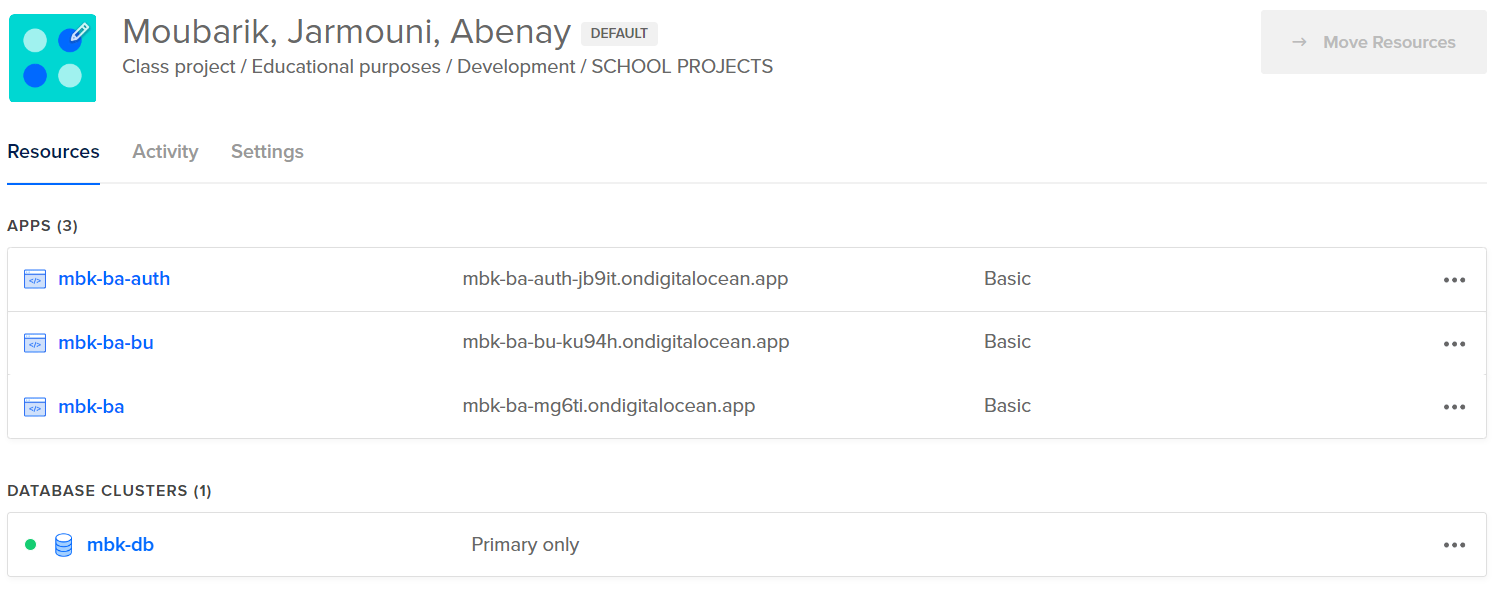
**CI/CD avec Jenkins**

Déclaration de Pipeline (Jenkinsfile)

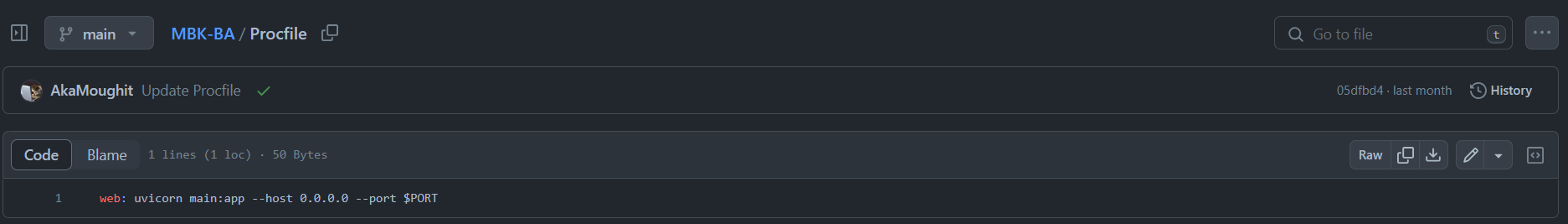
****

Ce Jenkinsfile définit un pipeline simple avec des étapes de build, de test et de déploiement.

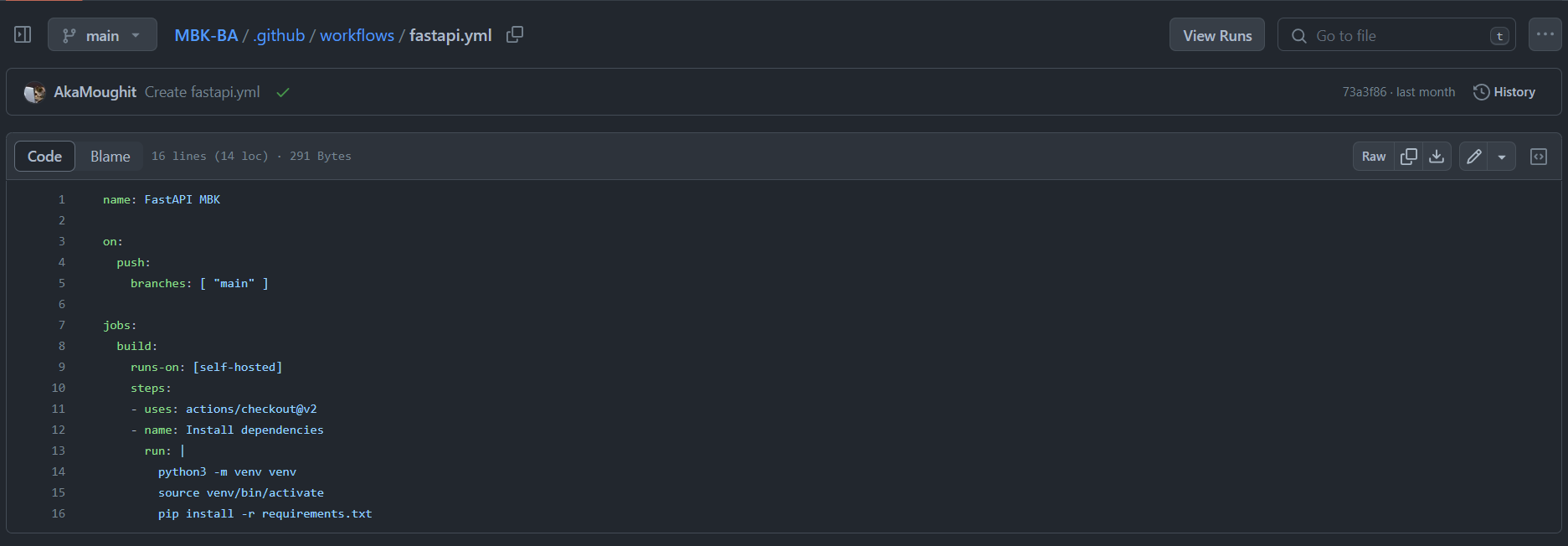
**Déploiement Automatique ( DigitalOcean )**

****

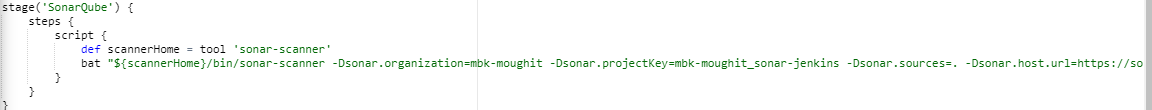
**Fichier Procfile :**

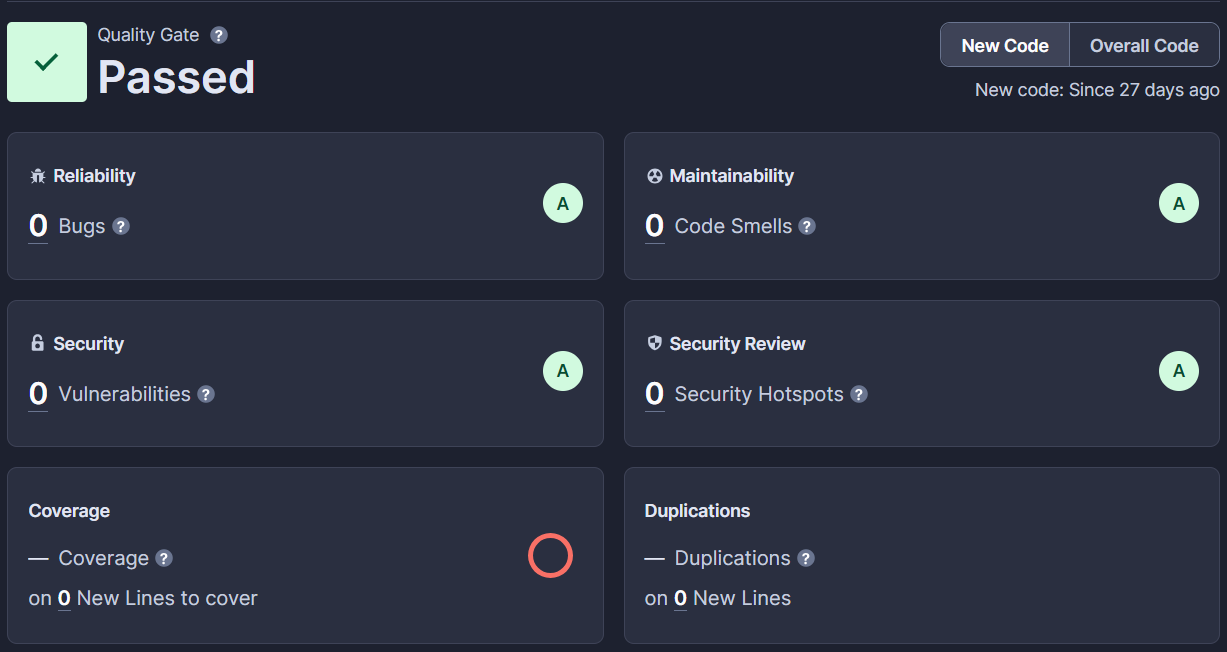
****

**Github Action :**

****

**Intégration de SonarQube :**

****

****

**Conclusion :**

Le projet de conception et d'implémentation de l'architecture des composants d'entreprise pour l'application Suivi de Budget a été mené avec succès. On a adopté une architecture microservices, mettant en œuvre les microservices `mbk-ba-auth`, `mbk-ba`, et `mbk-ba-bu` pour une gestion complète des finances personnelles.

La communication entre les microservices a été assurée par FastAPI, garantissant une expérience utilisateur transparente et sécurisée. La conteneurisation avec Docker a offert des avantages tels que l'isolation, la portabilité, et la facilité de déploiement.

L'intégration continue avec Jenkins a automatisé le processus de développement, et le déploiement automatique sur DigitalOcean a renforcé l'efficacité. L'intégration de SonarQube a contribué à maintenir une haute qualité de code.

En résumé des accomplissements,On a réussi à créer une architecture robuste et évolutive, offrant une expérience utilisateur optimale. Pour l'avenir, des perspectives excitantes incluent l'exploration de nouvelles technologies, l'ajout de fonctionnalités avancées, et la vigilance envers les évolutions de l'industrie.

Le projet a jeté des bases solides pour une application Suivi de Budget moderne, et l'avenir promet des opportunités passionnantes pour continuer à innover dans le domaine de la gestion des finances personnelles.