2025-02-19 CloudCode

Назва проекту: “Створення клієнт-серверного додатку хмарних обчислень коду”

CloudCode - назва програмного продукту.

Github: GitHub - DarkCard1nal/CloudCode: Client-server application of cloud computing Python code.

1. Призначення

Створення клієнт-серверного додатку хмарних обчислень має на меті надання користувачам можливості обчислень коду за запитом користувача. Алгоритм дій виглядає наступним чином: сервер, який приймає POST-запит, робить обчислення та повертає відповідь (результати обчислення). Клієнт відправляє POST-запит на сервер через API за своїм API ключем, приймає відповідь та виводить її. Такий підхід до обчислень коду надає можливість користувачам виконувати складні обчислення навіть без наявних для цього фізичних ресурсів (мається на увазі потужний користувацький ПК).

2. Глосарій

Термін	Значення

3. Бізнес-цілі

3.1.1. Довідкова інформація проекту

CloudCode - веб-застосунок, який дозволяє користувачам виконувати складні обчислення без наявного пристрою із прийнятними для цього характеристиками.
Модель розробки ПЗ - інкрементна модель.

3.1.2. Стейкхолдери

Учасник	Роль	Опис	Тип
Винокуров Артем			Внутрішній
Шкільний Владислав			Внутрішній
Гулько Анастасія			Внутрішній
Користувачі (зацікавлена сторона)	Кінцеві користувачі системи	Взаємодіють із сервісом через API або веб-інтерфейс. Вони можуть реєструватися, отримувати API-ключі та надсилати код для виконання.	Зовнішній

		Очікують надійність, безпеку та швидкість роботи сервісу.	
Сервери (надають послуги)	Інфраструктура для виконання запитів	Сервери обробляють запити, виконують код у безпечному середовищі (Docker), логують запити, взаємодіють із базою даних і забезпечують шифрування передачі даних. Від них залежить стабільність і продуктивність системи.	Внутрішній
Інвестори (зацікавлена сторона)	Фінансова підтримка проєкту	Надають фінансування на розробку, інфраструктуру та подальший розвиток сервісу. Очікують прибутку або зростання цінності проєкту, тому їх цікавлять метрики використання, бізнес-модель та масштабованість.	Зовнішній

Хто такі стейкхолдери і як з ними дружити - стаття 2025 - *інформація про типи стейкхолдерів*

3.2. Цілі

Можна виділити такі цілі проекту:

1. Надання користувачам можливості важких обчислень без наявності потужного комп’ютера – створення сервісу, що дозволяє виконувати складні обчислення на сервері, використовуючи віддалений доступ;
2. Впровадження елементів безпеки: шифрування передачі даних між сервером та клієнтом;
3. Покращення проекту та додавання нових можливостей для користувачів;
4. Створення зручного та легкого в користуванні інтерфейсу для користувачів;
5. Обмеження небезпечного виконання коду – застосування Docker-контейнерів, обмеження доступу до системних ресурсів, контроль часу виконання та фільтрація шкідливого коду;
6. Можливість паралельного виконання запитів – підтримка багатопотокової обробки запитів для швидкої роботи сервісу навіть при високому навантаженні;
7. Захист від несанкціонованого доступу – автентифікація через API-ключі, реєстрація користувачів;
8. Логування всіх запитів та результатів – збереження історії виконання коду для зручного аналізу та усунення можливих проблем;
9. Забезпечення доступу через REST API – можливість легкої інтеграції сервісу з іншими додатками та платформами;
10. Можливість комерціалізації проєкту – створення тарифних планів, обмежень за ресурсами та платного доступу до розширених можливостей сервісу;

3.3. Функціональні вимоги

Опис вимоги	Основні функції
Додаток повинен виконувати код користувача	Сервер виконує код, що був надісланий користувачем, у розробленому безпечному середовищі (Docker).
Додаток повинен мати можливість входу за логіном та паролем	Користувач може зареєструватися та увійти в систему, отримавши персональний API-ключ.

Додаток має характеризуватися наявністю веб-інтерфейсу	Надання веб-інтерфейсу для зручного користування (відправлення коду, перегляд результатів).
Додаток повинен приймати запити від користувача	Приймання POST-запитів через API, обробка коду та повернення результату.
Додаток повинен мати логування подій	Збереження історії запитів, виконаного коду та помилок.
Додаток повинен перевіряти API-ключі користувачів	Використання аутентифікації для доступу до сервісу.

3.4. Нефункціональні вимоги

Опис вимоги	Основні функції
Безпека	Код користувача виконується у ізольованому середовищі (Docker), передача даних здійснюється через HTTPS, автентифікація через API-ключі.
Обмеження доступу	Виконуваний код має працювати в обмеженому середовищі без можливості модифікації файлової системи, мережі та використання шкідливих команд.
Інтуїтивний UI/UX	Веб-інтерфейс повинен бути зрозумілим, адаптивним та зручним для користувачів різного рівня.
Захист від DoS-атак	Реалізація rate limiting, обмеження кількості запитів на користувача, автоматичне блокування підозрілих дій.
Захист API	Використання OAuth2 для авторизації запитів.
Кросплатформеність	Клієнтський інтерфейс та API мають підтримувати роботу на всіх сучасних браузерах та операційних системах.
Підтримка паралельного виконання	Система повинна дозволяти одночасне виконання кількох запитів без затримок.
Конфігурованість	Адміністратори повинні мати можливість налаштовувати ліміти ресурсів, обмеження доступу та параметри безпеки без перезапуску сервера.