Прихідько В’ячеслав

Добре, розпочнемо з написання функції для обчислення факторіалу в мові програмування Python:

# factorial.py

def factorial(n):

if n < 0:

return "Факторіал не визначений для від'ємних чисел"

elif n == 0:

return 1

else:

result = 1

for i in range(1, n + 1):

result \*= i

return result

Тепер напишемо юніт-тести для цієї функції за допомогою бібліотеки `unittest`:

# test\_factorial.py

import unittest

from factorial import factorial

class TestFactorial(unittest.TestCase):

def test\_positive\_integer(self):

self.assertEqual(factorial(5), 120)

def test\_zero(self):

self.assertEqual(factorial(0), 1)

def test\_negative\_integer(self):

self.assertEqual(factorial(-5), "Факторіал не визначений для від'ємних чисел")

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

Тепер запустимо тести та переконаймося, чи працює наша функція правильно.

$ python test\_factorial.py

...

----------------------------------------------------------------------

Ran 3 tests in 0.000s

OK

Docker і розгортання програмного забезпечення:

Легкість розгортання: Docker дозволяє упаковувати програмне забезпечення та всі його залежності в контейнер, що робить процес розгортання простішим та консистентним.

Портабельність: Контейнери Docker можуть запускатися на будь-якій системі, що підтримує Docker, незалежно від конфігурації цієї системи.

Ізоляція: Контейнери забезпечують ізоляцію різних компонентів програмного забезпечення, що допомагає уникнути конфліктів між залежностями та середовищами.

Управління командою в контексті розробки програмного забезпечення:

Комунікація: Забезпечення ефективної комунікації в команді.

Лідерство та мотивація: Спрямування команди на досягнення спільних цілей.

Розподіл завдань: Ефективне розподілення робочих завдань у команді.

Управління конфліктами: Розв'язання конфліктів та забезпечення спільної роботи.

Git та управління змінами:

Версійний контроль: Збереження та відстеження змін у програмному коді.

Колаборація: Можливість багатокористувацької роботи над проектом.

Відгалуження та злиття (Branching and Merging): Дозвіл експериментів та безпечне об'єднання змін.

Стратегії деплойменту програмного забезпечення:

Поступовий деплоймент (Rolling Deployment): Поступове оновлення частини системи, зберігаючи доступність.

Сині-зелені деплойменти (Blue-Green Deployment): Робота двох середовищ (одне продакшн, інше нове), з перемиканням між ними.

Канарські релізи (Canary Releases): Послідовне впровадження нововведень у невеликих частинах користувачів для тестування.

Перевірка та валідація в розробці ПЗ:

Юніт-тести та інтеграційні тести: Впевненість у правильності окремих компонентів та їх взаємодії.

Автоматизоване тестування: Зменшення ручного тестування та підвищення ефективності.

Code reviews (Огляд коду): Перегляд коду під час розробки для виявлення помилок та покращення якості коду.

Стрес-тести та навантажувальне тестування: Оцінка стабільності та продуктивності системи під великим навантаженням.