УДК 004
Фінько В.С.[1], Скрупський С.Ю.[2]
[1] студ. гр. КНТ-527 НУ «Запорізька політехніка»
[2] доц. НУ «Запорізька політехніка»

## АЛГОРИТМ БЕЗПЕРЕРВНОЇ ІНТЕГРАЦІЇ ТА БЕЗПЕРЕРВНОЇ ДОСТАВКИ

Традиційно випуск нових версій ПЗ був пов'язаний з чималими труднощами. Проблеми починалися на ранніх етапах розробки. Окремі групи працювали над своїми модулями незалежно один від одного і перед випуском нових версій повинні були спочатку інтегрувати їх разом. У процесі інтеграції, коли модулі вперше об'єднувалися один з одним, система часто навіть не компілювалася. Витрачалося багато часу для того, щоб інтегрувати всі зміни. Також процес перенесення був дуже трудомістким. Вигоди від впровадження якої-небудь однієї функції не виправдовували зусиль, що витрачалися на інтегрування і розгортання.

За допомогою технологій безперервної інтеграції і безперервного розгортання можна запобігти виникненню подібних проблем. Знижується трудомісткість інтеграції, система стає більш передбачуваною, несправності виявляються і виправляються швидко, розгортання здійснюється частіше. Це

скорочує час введення в дію нових можливостей, що дає суттєву перевагу для бізнесу: він отримує можливість швидше реагувати на зміни ринку.

CI / CD (Continuous Integration, Continuous Delivery - безперервна інтеграція і доставка) - це технологія автоматизації тестування та доставки нових модулів, розроблюваного проекту зацікавленим сторонам (розробники, аналітики, інженери якості, кінцеві користувачі і ін.) [1].

Процеси розгортання мають високий рівень автоматизації, тому результати легко відтворити. Тобто після розгортання і перевірки системи в тестовому оточенні, той же результат буде отримано в робочому оточенні, тому що самі оточення повністю ідентичні. Це дозволяє практично повністю усувати будь-які помилки і, відповідно, зменшувати ризики.

Крім того, сам процес тестування програмного забезпечення стає простішим завдяки автоматизації його виконання. Це ще більше підвищує якість, тому що тестування може виконуватися набагато частіше.

Чим частіше проводиться розгортання нових версій, тим нижче ризики, тому що з кожним разом розгортається менша кількість змін і нововведень. Чим менше змін потрапляє в робоче оточення, тим нижче ризик появи помилок [2].

Для організації процесу CI / CD на виділеному сервері запускається служба, в завдання якої входять: отримання вихідного коду зі сховища, визначення гілки проекту, збірка проекту, виконання тестів, відправка образів у віддалений репозиторій, видалення використаних образів, розгортання готового проекту.

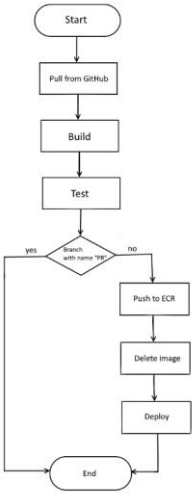


Рисунок 1 – Блок схема запропонованого алгоритму

В роботі пропонується наступний алгоритм. На першому етапі витягується Dockerfile з репозиторію на GiHub. Команда docker build будує образ, звертаючись до файлу Dockerfile, який знаходиться в поточній директорії, на що вказує точка в кінці.

На наступному етапі тестуємо образ, використовуючи команду docker run, яка запускає контейнер на основі побудованого образу.

Надалі визначаємо належність до певної гілки. Після цього відправляємо образ до віддаленого репозиторію.

Наступна дія – це видалення образу. Останній етап – це розгортання образу у сервісі ECS. Створюємо набір змін для оновлення стеку CloudFormation за допомогою заданого шаблону.

Якщо дотримуватися такого алгоритму, можна забезпечити побудову стійкої та налагодженої роботи будь-якої системи, що збільшує прибуток компанії і зменшує час простою обладнання та робочої сили.

## СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Поняття CI/CD [Електроний ресурс]. Режим доступу: https://selectel.ru/blog/what-is-ci-cd

2. Еберхард В. Continuous delivery. Практика безперервних апдейтів. – СПб.:Пітер, 2018. – 320с.