**Алгоритм решения**

Создание дипломного проекта производилось по методу CI/CD для компании, в которой я работаю. В этой главе будет подробно описан данный принцип.

Непрерывная интеграция (Continuous Integration, CI) и непрерывная поставка (Continuous Delivery, CD) представляют собой культуру, набор принципов и практик, которые позволяют разработчикам чаще и надежнее развертывать изменения программного обеспечения.

Основные принципы CI/CD

* Сегрегация ответственности заинтересованных сторон;
* Снижение риска;
* Короткий цикл обратной связи;
* Реализация среды.

Этапы CI/CD:

**Написание кода.** Каждый из разработчиков пишет код своего модуля, проводит ручное тестирование, а затем соединяет результат работы с текущей версией проекта в основной ветке. Для контроля версий используется система Git, либо аналогичные решения. Когда участники команды опубликуют код своих модулей в основной ветке, начнется следующий этап.

Сборка. Система контроля версий запускает автоматическую сборку и тестирование проекта. Триггеры для начала сборки настраиваются командой индивидуально — фиксация изменений в основной ветке проекта, сборка по расписанию, по запросу и т. д.

**Ручное тестирование.** Когда CI система успешно проверила работоспособность тестовой версии, то код отправляется тестировщикам для ручного обследования. При этом тестовая сборка получает номер кандидата для дальнейшего релиза продукта (например, v.1.0.0-1).

**Релиз.** По итогам ручного тестирования сборка получает исправления, а итоговый номер версии кандидата повышается (например, после первого исправления версия v.1.0.0-1 становится v.1.0.0-2). После этого выпускается версия кода для клиента (например, v.1.0.0) и начинается следующий этап цикла.

**Развертывание.** На этом этапе рабочая версия продукта для клиентов автоматически публикуется на production серверах разработчика. После этого клиент может взаимодействовать с программой и ознакомиться с ее функционалом как непосредственно через готовый интерфейс, так и через облачные сервисы.

**Поддержка и мониторинг.** Конечные пользователи начинают работать с продуктом. Команда разработки поддерживает его и анализирует пользовательский опыт.

**Планирование.** На основе пользовательского опыта формируются запросы на новый функционал для продукта, готовится план доработок. После этого цикл замыкается и переходит в начальную стадию — написание кода. Далее начинается новая итерация CI/CD разработки.

На Рисунке 1 представлена схема этапов CI/CD для более подробного понимания.

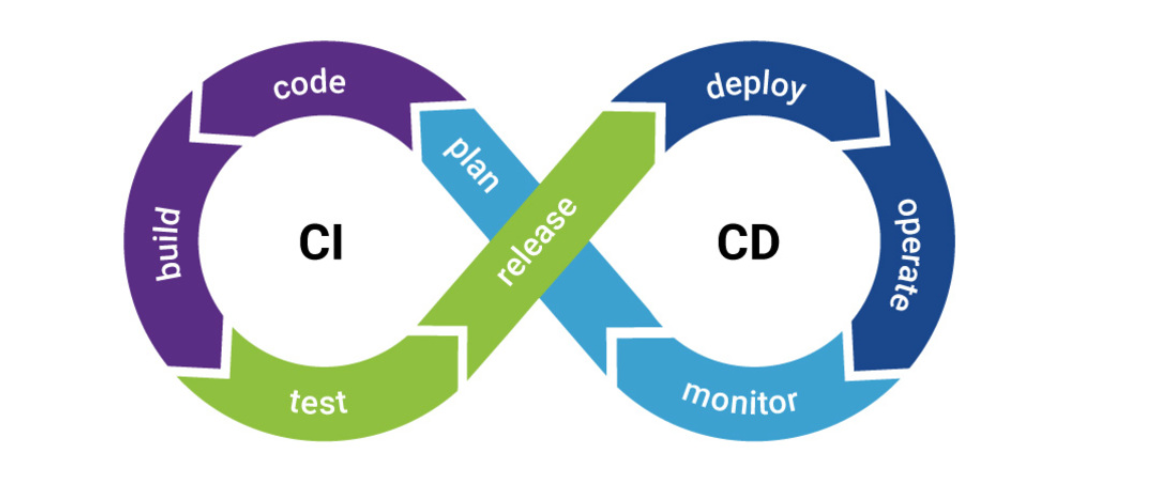


Рисунок 1. Схема этапов CI/CD

Главные цели CI/CD — свести к минимуму ошибки, ускорить сборку и повысить качество конечного продукта. В CI/CD тестирование проводится всякий раз, когда в код вносятся изменения. Обнаружение ошибок на ранней стадии разработки существенно экономит время и ресурсы команды, ведь чем позднее выявляется баг, тем труднее и дороже его исправлять. Что касается ускорения, то автоматизация CI/CD помогает значительно оптимизировать все рутинные процессы сборки.

На Рисунке 2 представлен базовый рабочий процесс CI/CD для наглядного рассмотрения.

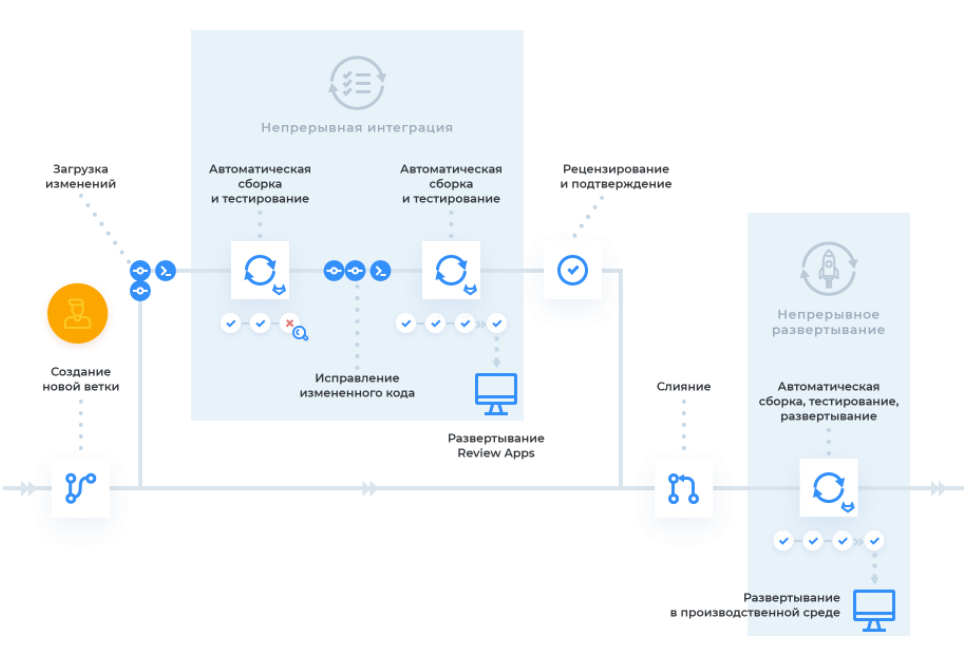


Рисунок 2. Базовый рабочий процесс CI/CD