ЗАДАНИЕ №5

1) Как DevOps повышает качество продукта? за счет чего?
2) Составьте пример инструментов и практик для первого базового уровня зрелости DevOps.
3) Зачем приложения упаковываются в контейнеры и как Kubernetes помогает в их управлении.

РЕШЕНИЕ:

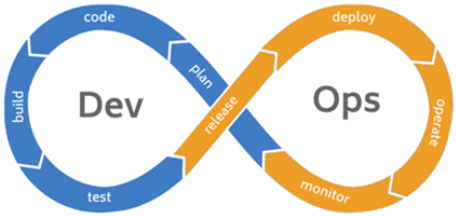
1) Следуя пути DevOps, код требуется переводить из стадии разработки в производство непрерывно в автоматическом режиме, поэтому автоматизацию можно считать синонимом DevOps. Автоматизация упрощает рабочие процессы, сокращает количество сбоев и откатов, уменьшает количество ошибок, которые возникают при ручной настройке. Повышение эффективности, улучшение производительности и польза для конечного потребителя – главные преимущества автоматизации. В идеале автоматизировать нужно почти все:
- инфраструктуру
- выпуски программного обеспечения (software releases)
- тестирование
- развертывание
- основные задачи по безопасности
- политику соглашений
- задачи управления конфигурацией

Быстрое обновление
Увеличьте частоту и скорость релизов, чтобы быстрее обновлять и улучшать продукт. С одной стороны, это позволяет оперативно реагировать на потребности клиентов, с другой — создает дополнительные конкурентные преимущества. Практики непрерывной интеграции и непрерывной доставки помогают автоматизировать процесс выпуска релизов, от сборки до развертывания.

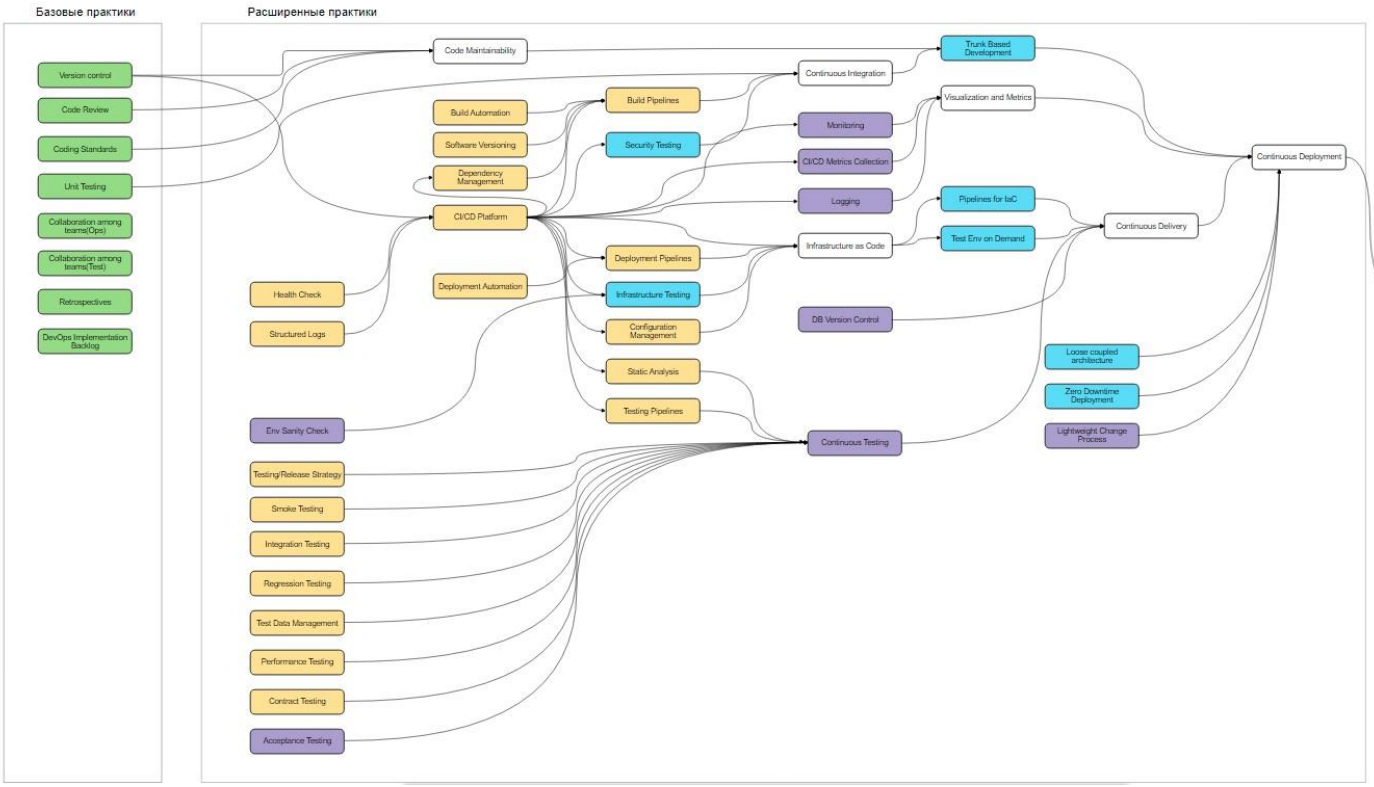
Надежность
Контролируйте качество обновлений и изменений инфраструктуры — такой подход сделает продукт надежнее и поможет сохранить лояльность пользователей. Практики непрерывной интеграции и непрерывной доставки позволяют проверить каждое обновление на работоспособность и безопасность. А мониторинг и ведение журналов — следить за производительностью в режиме реального времени.

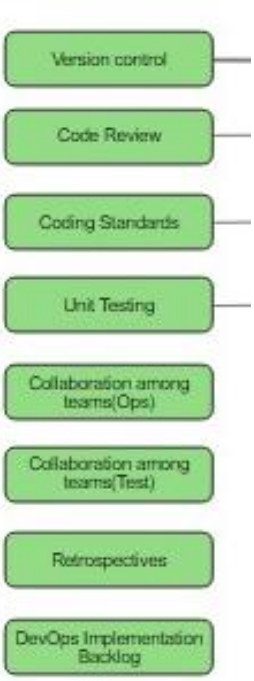
Масштабируемость
DevOps упрощает управление разработкой и поддержку инфраструктуры: это обеспечивает стабильную работу софта при любых масштабах и помогает контролировать сложные системы с минимальными рисками. Например, практика «инфраструктура как код» повышает эффективность управления средами разработки, тестирования и производства, а также обеспечивает их воспроизводимость.

Оптимизация совместной работы
Культурная модель DevOps предполагает сопричастность и ответственность: это означает, что команды разработки и эксплуатации работают в тесном контакте друг с другом, разделяют большинство обязанностей и объединяют свои рабочие процессы. Такой подход повышает эффективность работы и экономит время: разработчики быстрее передают дела инженерам по эксплуатации, а при написании кода не приходится ориентироваться на среду, в которой он будет запущен.

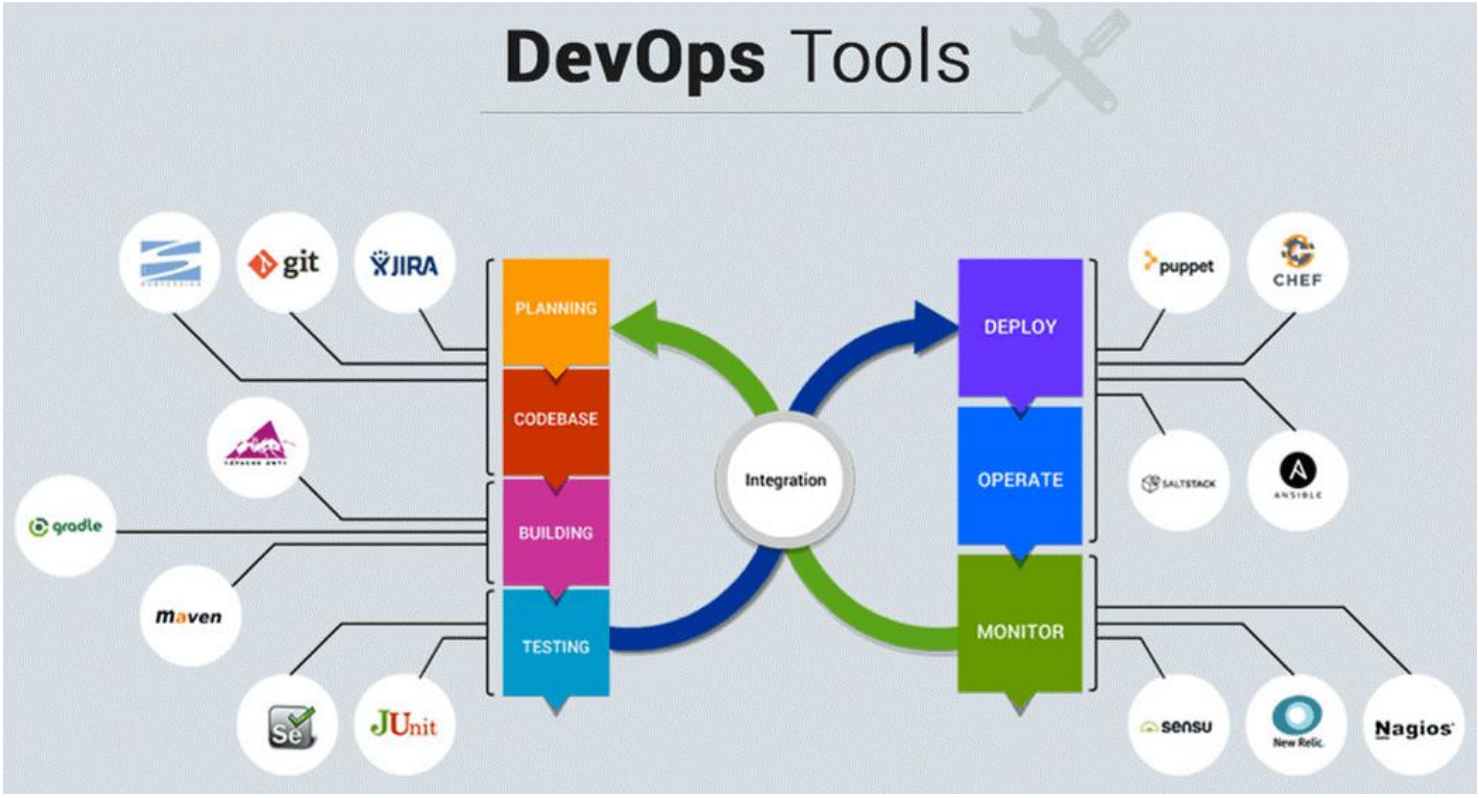


2) Исходя из таблицы уровня зрелости DevOps, для 1 уровня достаточно освоения базовых практик из зеленых и белых прямоугольников:

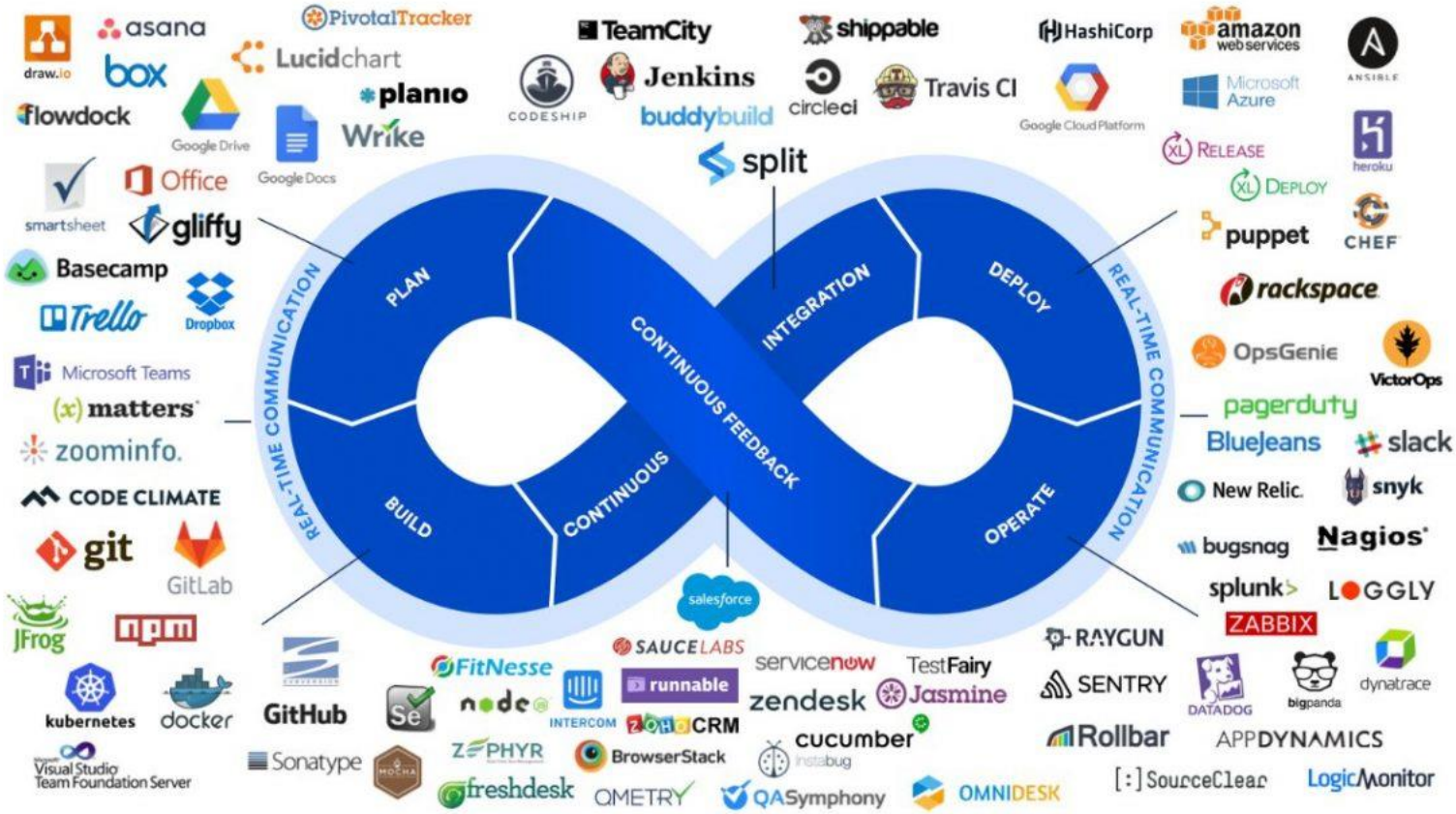




В качестве инструментов может быть пример ниже:



Но в процессе развития, работы над разными проектами и в разных компаниях, список инструментов хорошего инженера может существенно расшириться… Примерно так:



3) Контейнеры — это способ упаковать приложение и все его зависимости в единый образ. Этот образ запускается в изолированной среде, не влияющей на основную операционную систему. Контейнеры позволяют отделить приложение от инфраструктуры: разработчикам не нужно задумываться, в каком окружении будет работать их приложение, будут ли там нужные настройки и зависимости. Они просто создают приложение, упаковывают все зависимости и настройки в единый образ. Затем этот образ можно запускать на других системах, не беспокоясь, что приложение не запустится. Контейнеры в целом упрощают работу как программистам, и тем, кто развертывают эти приложения.

Так как контейнеры надо как-то автоматически взводить на боевых серверах, возникла необходимость в софте, который будет управлять всеми контейнерами через свое API — и так возник Kubernetes. Kubernetes предоставляет:

- **Мониторинг сервисов и распределение нагрузки** Kubernetes может обнаружить контейнер, используя имя DNS или собственный IP-адрес. Если трафик в контейнере высокий, Kubernetes может сбалансировать нагрузку и распределить сетевой трафик, чтобы развертывание было стабильным.

- **Оркестрация хранилища** Kubernetes позволяет вам автоматически смонтировать систему хранения по вашему выбору, такую как локальное хранилище, провайдеры общедоступного облака и многое другое.

- **Автоматическое развертывание и откаты** Используя Kubernetes можно описать желаемое состояние развернутых контейнеров и изменить фактическое состояние на желаемое. Например, вы можете автоматизировать Kubernetes на создание новых контейнеров для развертывания, удаления существующих контейнеров и распределения всех их ресурсов в новый контейнер.

- **Автоматическое распределение нагрузки** Вы предоставляете Kubernetes кластер узлов, который он может использовать для запуска контейнерных задач. Вы указываете Kubernetes, сколько ЦП и памяти (ОЗУ) требуется каждому контейнеру. Kubernetes может разместить контейнеры на ваших узлах так, чтобы наиболее эффективно использовать ресурсы.

- **Самоконтроль** Kubernetes перезапускает отказавшие контейнеры, заменяет и завершает работу контейнеров, которые не проходят определенную пользователем проверку работоспособности, и не показывает их клиентам, пока они не будут готовы к обслуживанию.

- **Управление конфиденциальной информацией и конфигурацией** Kubernetes может хранить и управлять конфиденциальной информацией, такой как пароли, OAuth-токены и ключи SSH. Вы можете развертывать и обновлять конфиденциальную информацию и конфигурацию приложения без изменений образов контейнеров и не раскрывая конфиденциальную информацию в конфигурации стека.