BINAR

CI/CD

Gold Chapter 6 - Topic 6

BINAR

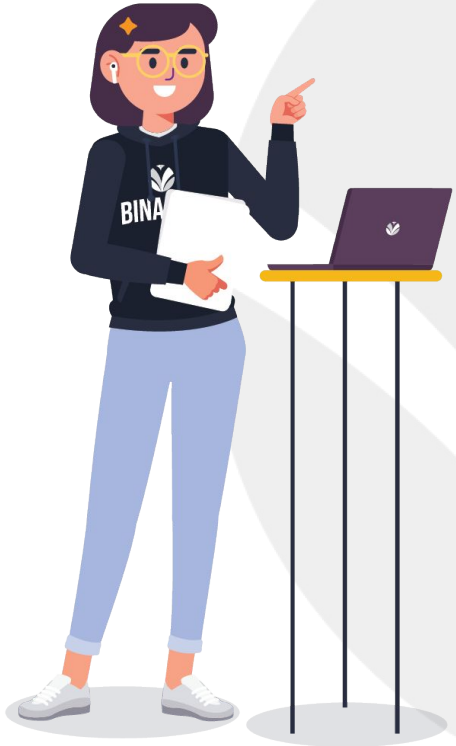
Selamat datang di Chapter 6 Topic 6 online course Back End Java dari Binar Academy!

Ada yang nungguin kamu nih, Binarian

Eng ing eng.. ada topik CI/CD yang perlu kita pelajarin di akhir chapter 6 ini.

Iya dong, setelah pada topik sebelumnya kita belajar tentang Deployment, pada topic keenam kita bakal mengelaborasi tentang **CI/CD,** mulai dari konsep CI sampai konsep CD.

Penasaran? kalau gitu kita langsung geser slide-nya yuuuk~



Dari sesi ini, kita bakal bahas hal-hal berikut:

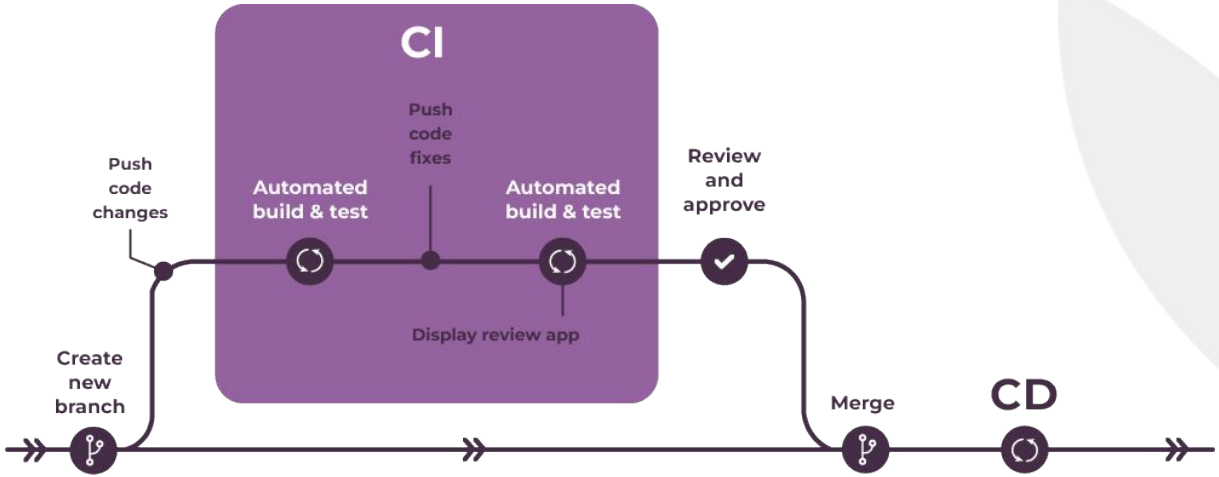
- Konsep CI/CD

Binarian, kita masuk ke materi yang udah dijanjiin di awal tadi, yaitu CI/CD.

CI/CD sendiri merupakan dua bagian keberlanjutan, untuk itu yang pertama mari kita bahas CI dulu~

Kenalan sama yang namanya CI dulu, yuk!

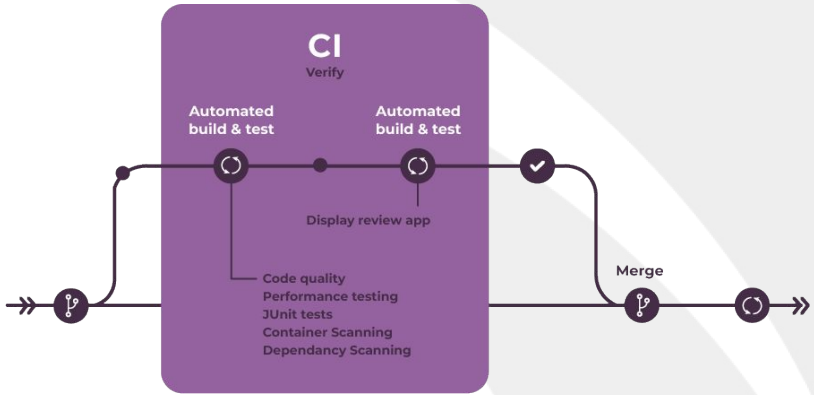
Mari kita kenalan. Continuous Integration atau CI adalah proses pengujian dan pembuatan software secara otomatis. Dilakukan setelah kode aplikasi yang baru, udah terintegrasi ke dalam repository bersama (tempat penyimpanan).



CI ini bisa dibilang sebagai **sebuah proses mengintegrasikan kode ke dalam sebuah repositori yang nantinya bakal menjalani pengujian**.

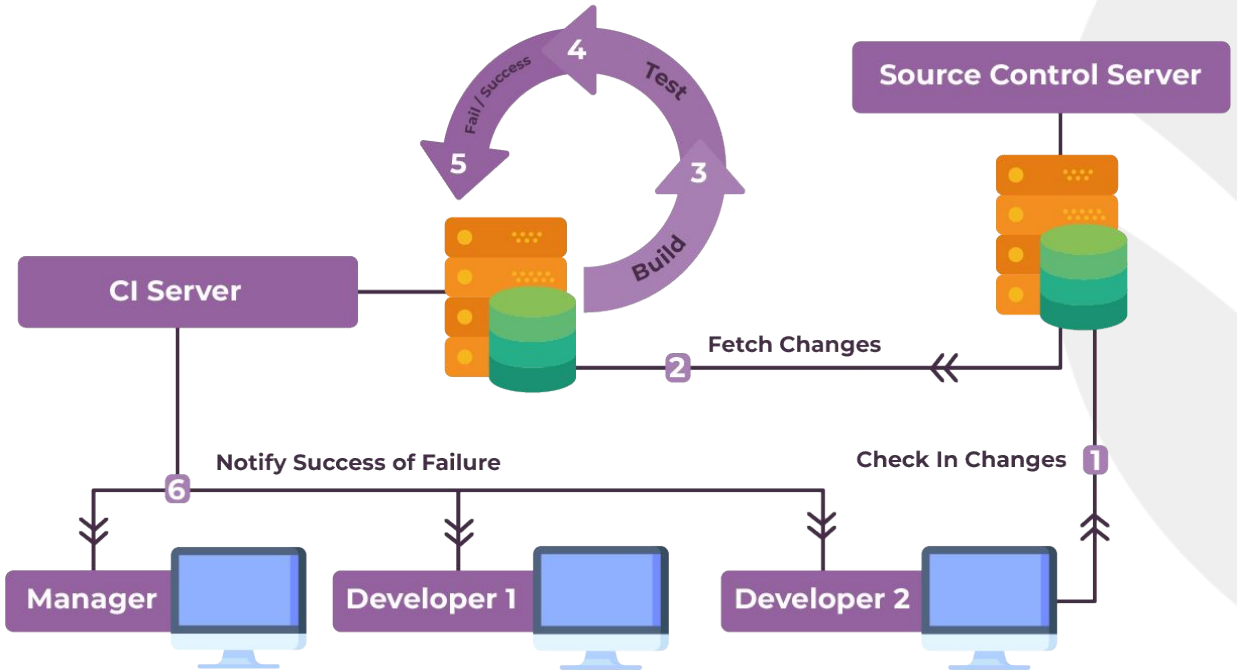
Untuk implementasinya, kamu bisa menerapkan CI pakai perintah commit, lho.

Biasanya pengujian dalam proses ini berlangsung cepat dan dilakukan secara otomatis.



BINAR

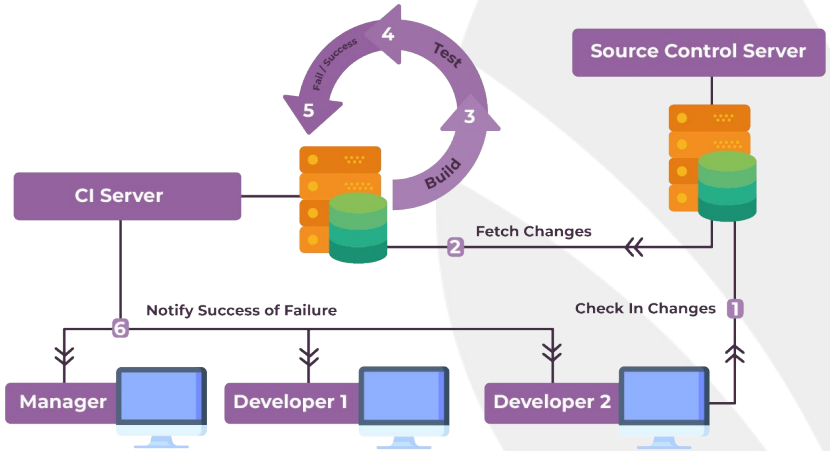
Begini cara kerja dari sebuah CI~



Kita perjelas cara kerjanya, yaaa~

1. Developer melakukan commit ke repository.
2. Setelah itu, remote branch bakal menerima changes tersebut.
3. Terus remote branch menjalankan job build.

   Fungsinya buat memastikan dependency yang ada di code udah tercover atau belum. Kalau gagal, berarti nggak bakal dilanjutkan ke tahap selanjutnya.

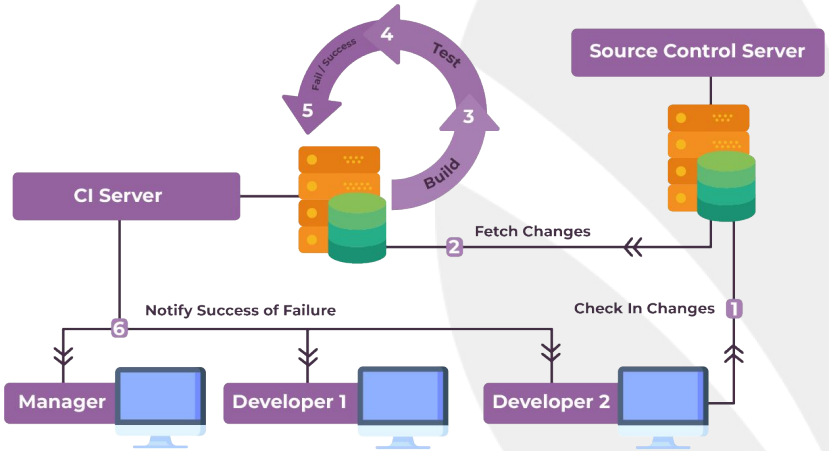


4. Setelah itu, job bakal dijalankan buat mengeksekusi seluruh unit test.

   Kalau masih ada unit test yang failed, maka job bakal gagal dan nggak dilanjutkan ke tahap selanjutnya.

5. Terus di cek apakah prosesnya udah berhasil atau malah gagal.

6. Terakhir, user bakal menerima notifikasi apakah job dari CI berhasil dijalankan atau nggak.



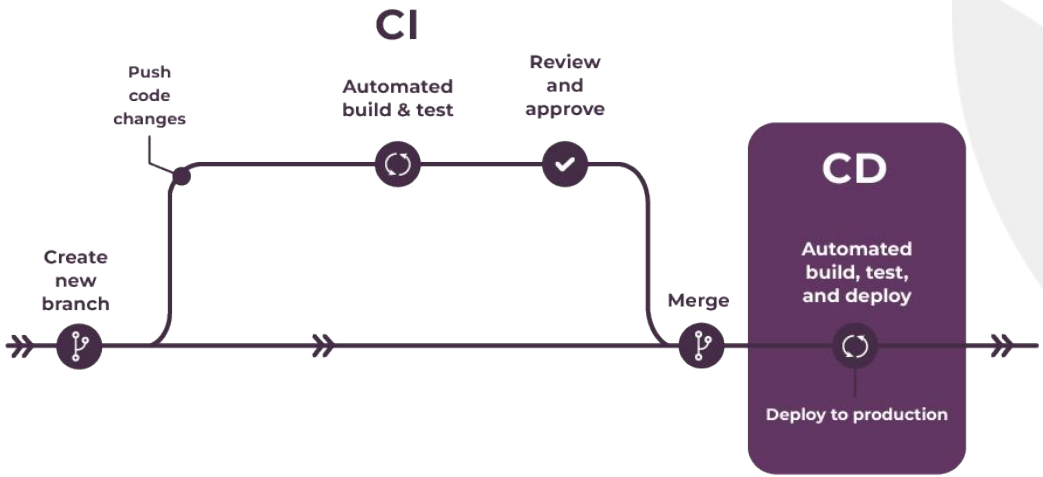
Kalau tadi udah bahas CI, berarti sekarang saatnya kita kenalan sama lanjutannya, yaitu CD.

CD sendiri merupakan kepanjangan dari continuous development.

“Emang CD itu maksudnya gimana, sih?”

CD atau Continuous Deployment adalah **proses delivery aplikasi yang dibuat dalam proses CI ke bagian lingkungan produksi, yang dimasukkan melalui automated test**.

Jadi nih, setelah changes yang di-commit berhasil lolos di semua tahap CI, berarti si aplikasi siap buat di-deploy kapan aja. Inti dari CD adalah **mempermudah dan mempersingkat proses deployment.**

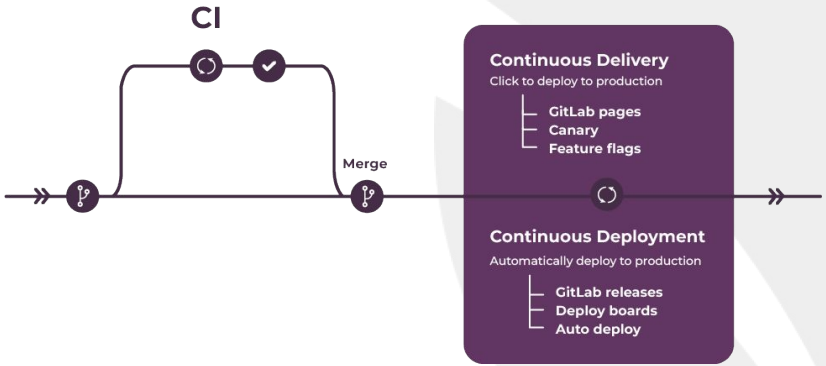


Tanpa CD, proses deployment harus dijalankan secara manual, lho

Iya, misalnya gini di suatu server harus terinstall dulu JDK dan Maven supaya bisa melakukan compile dan menjalankan aplikasi Java.

Terus, dari code yang udah dibikin, kita harus membentuk file JAR atau WAR yang nantinya bakal dipindahkan ke server.

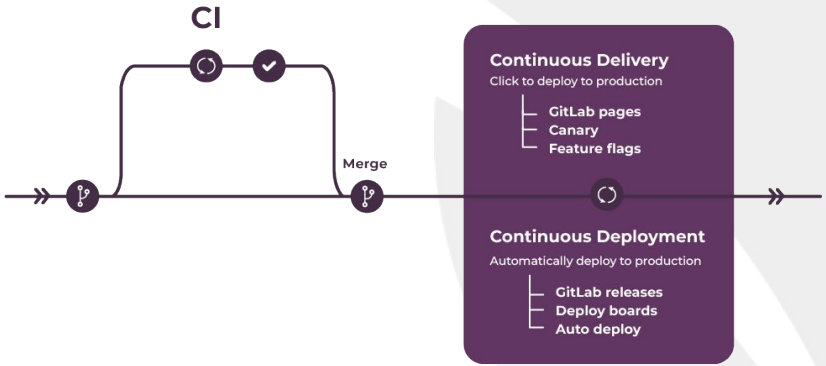
Setelah server menerima file JAR atau WAR tersebut, maka bakal dijalankan pakai Java Argument.



Dengan cara manual kayak gitu, kalau terjadi perubahan code yang cukup sering, bakal menyulitkan developer karena prosesnya yang lambat banget.

Tapi, kalau pakai CD, **prosesnya bakal jadi lebih singkat karena segalanya udah dilakukan secara otomatis**.

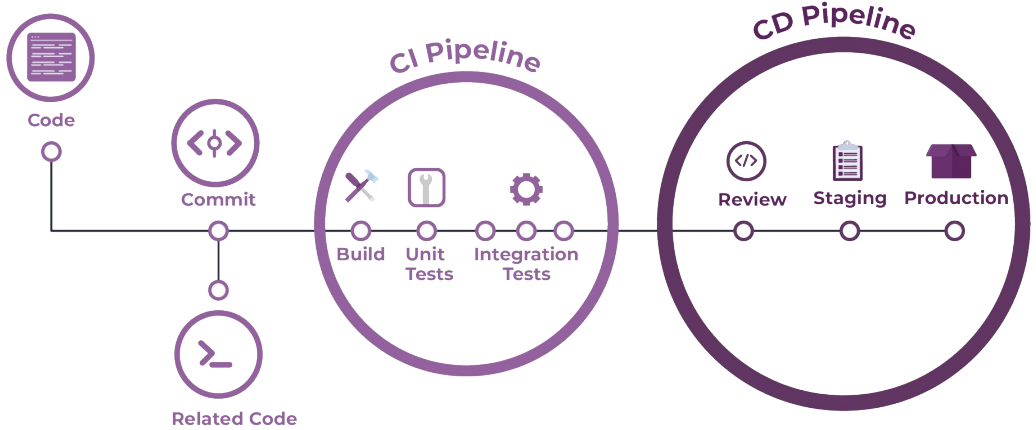
Kita nggak perlu memindahkan file aplikasi dan nggak perlu juga buat menjalankan aplikasinya secara manual~



“Kenapa perlu sebuah CI/CD?”

Karena CI/CD memungkinkan distribusi kode yang lebih sering. CD secara otomatis mendistribusikan aplikasi dan menjalankan unit test yang udah ditambahkan.

Pipeline CI/CD dirancang untuk tim yang perlu perubahan cukup sering di aplikasi, dengan proses delivery yang handal.



Selain itu, CI/CD juga punya sebuah klaim yang diberikan, yaitu: **sekali setup, deploy lancar seterusnya~**

Setelah setup CI/CD selesai, berkurang lah urusan kita dengan deployment yang meresahkan itu dan kita bisa santai sebentar buat fokus ke task yang lain!

Bukan sembarang proses, tapi CI/CD punya beberapa keuntungan, gengs!

- **Mendapatkan Feedback Lebih Cepat**

  Dalam CI/CD Pipeline ini, setiap kode bakal selalu di-test secara bersamaan supaya proses software development bisa dilakukan dengan seimbang.

  Dengan CI Tools, feedback atau tanggapan atas masalah atau error yang terjadi juga bisa diterima lebih cepat, lho.

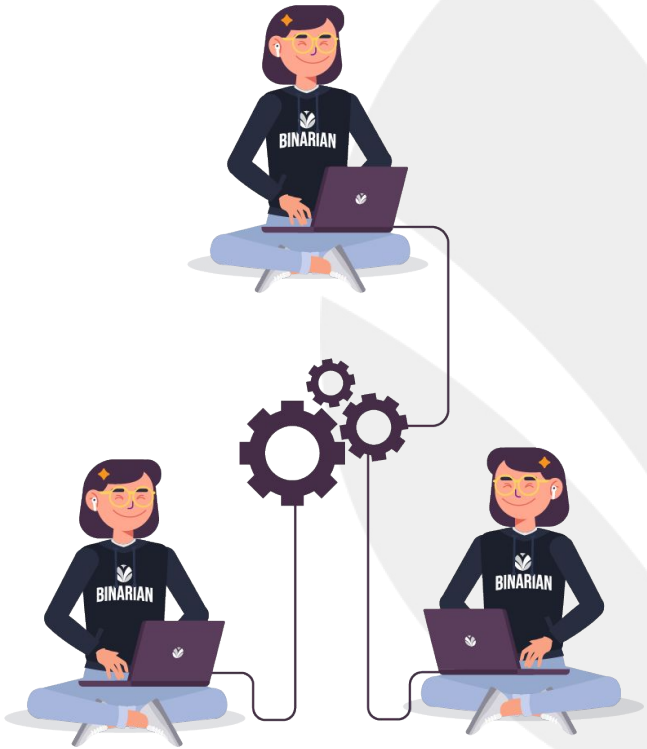
  Akhirnya pihak terkait pun bisa langsung menindaklanjuti feedback atau tanggapan tersebut, apapun bentuknya.

- **Visibilitas Lebih Baik**

  CI/CD Pipeline punya sifat yang transparan sob, jadi memudahkan developer buat mengontrol perubahan sekaligus menghindari kerusakan pada software atau aplikasi.

  Selain itu, developer juga dimudahkan dalam menganalisa pengembangkan aplikasi dari awal sampai akhir, jadi semua masalah bisa diatasi dengan segera.

  Mantap!



- **Deteksi Bug Lebih Awal**

  Kayak yang udah dibahas tadi, CI/CD adalah proses yang otomatis, jadi kalau ada bug bakal langsung terdeteksi.

  Developer nggak bakal kesulitan dalam mengembangkan aplikasi karena semua bug muncul bisa langsung diketahui mana yang perlu diperbaiki.

  Kuerenn!

Set Up GitHub Action

FYI nih, dalam CI/CD, ada satu script format **.yml** yang dijalankan otomatis.

File ini sebenarnya cuma mengatur proses CI aja soalnya proses CD udah dibantu sama Railway.app dimana kita nggak perlu melakukan deployment layaknya melakukan deploy aplikasi Java secara manual.

Kalau menggunakan github action, file yml ini perlu kamu simpan di folder

.github/workflow/file_name.yml

Contoh seperti gambar di samping.

Untuk detail penggunaan github action, kamu bisa mengikuti panduan **ini** atau dari **tutorial ini.**

```yaml
name: learn-github-actions
run-name: ${{ github.actor }} is learning GitHub Actions
on: [push]
jobs:
  check-bats-version:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - uses: actions/setup-node@v3
        with:
          node-version: '14'
      - run: npm install -g bats
      - run: bats -v
```

Lanjut nih, kali ini kita bakal bikin 3 stage buat CI-nya, yaitu build, unit-test, dan deploy!

Pada implementasi sebenarnya, kita bisa menambahkan berbagai macam stage, gengs.

Misalnya melakukan back up terhadap hasil deploy sebelumnya sehingga nanti juga bisa menambahkan job buat rollback.

Kita juga bisa menambahkan banyak environment di masing-masing stage.

Misalnya, pas udah lolos di environment dev, maka bisa diteruskan ke SIT dan diteruskan lagi ke stage terakhir, yaitu deploy ke production. Tempat dimana aplikasi bisa dipakai secara umum.

Kita udah berpetualang dari pengenalan Spring Security, Java Logging, Docker, Deployment sampai dengan CI/CD. Selamat! Kita udah tuntas belajar Chapter 6!□□

Dari semua topik yang udah dipelajari, ada topik yang kamu sulit pahami?

Kalau ada kira-kira bagian yang mana, sob? Coba kamu catat dan silakan pelajari kembali materinya ya!

Nah, selesai sudah pembahasan kita di Chapter 6 ini. Luar biasa!

Selanjutnya, kita bakal siap-siap untuk belajar pada Chapter 7. Kira-kira ada apa ya di chapter 7?

Sampai jumpa pada chapter selanjutnya~