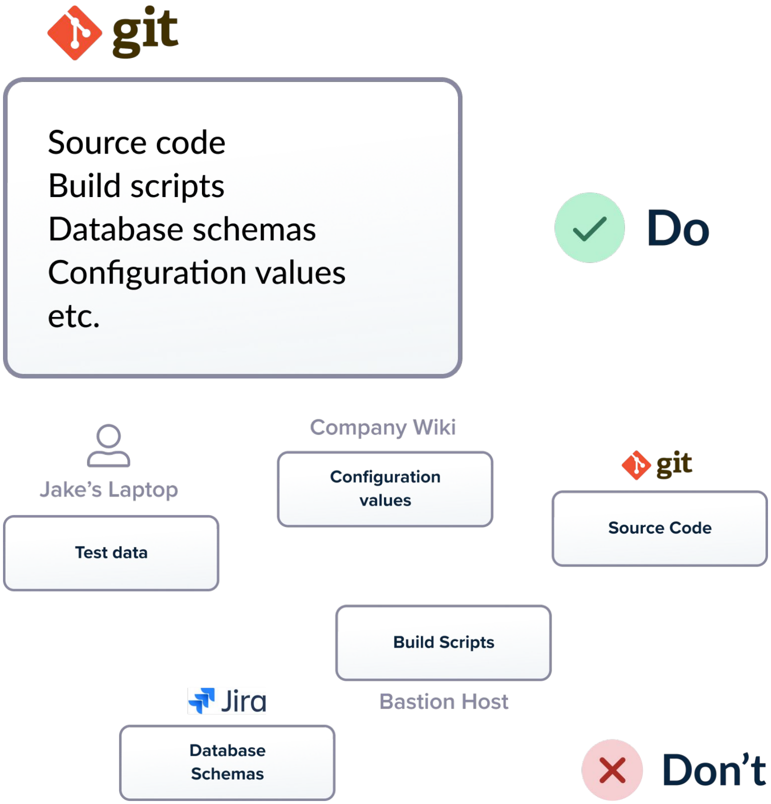
Enterprise CI/CD Best Practices



git

Source code
Build scripts
Database schemas
Configuration values
etc.

✓ Do

Jake’s Laptop

Test data

Company Wiki

Configuration values

git

Source Code

Build Scripts

Bastion Host

Jira

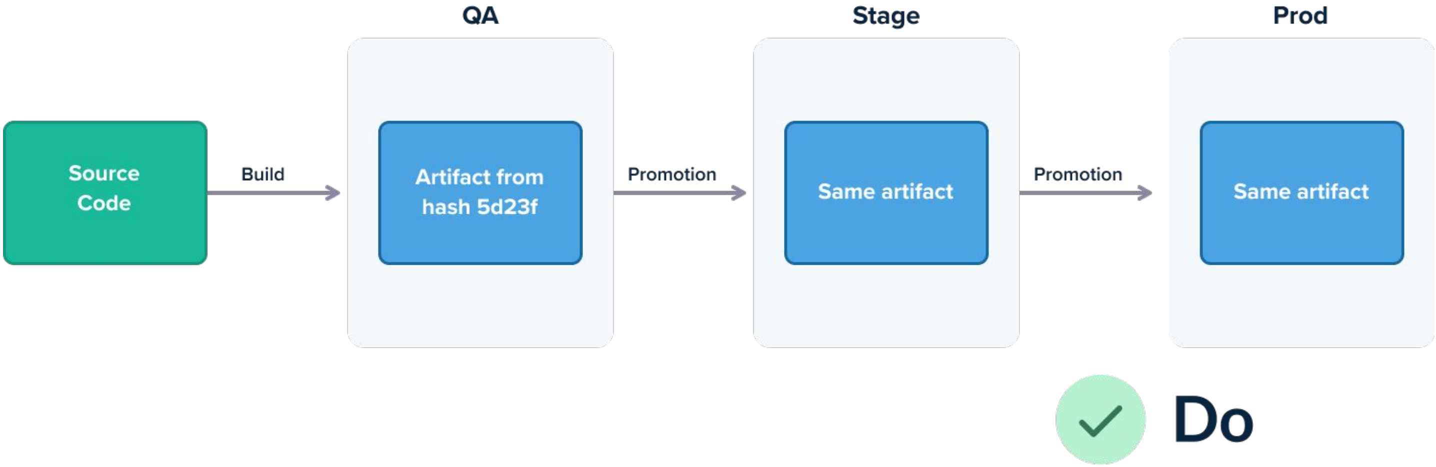
Database Schemas

✗ Don’t

Best Practice 1 – Place Everything Under Source Control

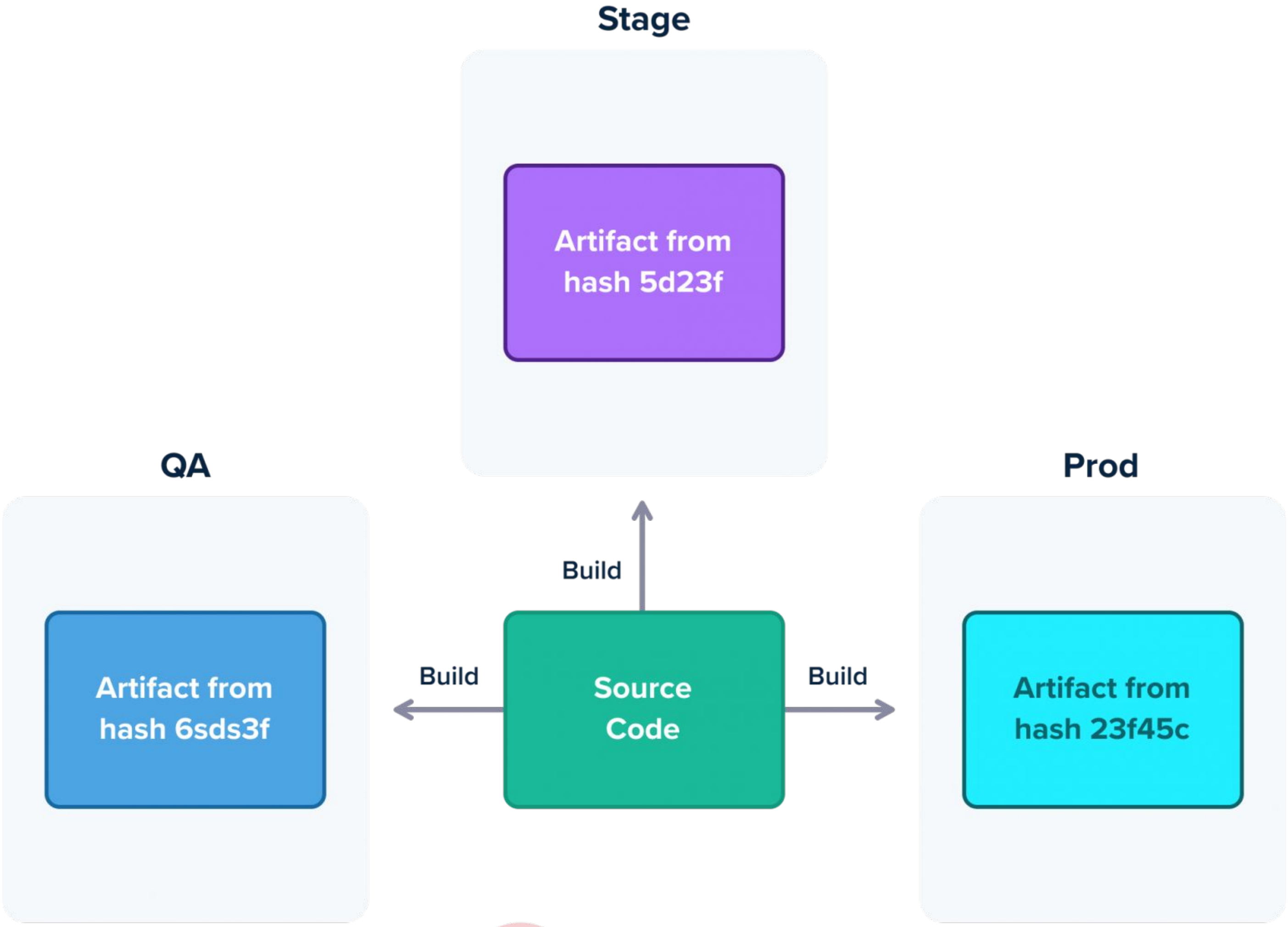
Everything that takes part in the application lifecycle should be checked into source control

Source: Kostis Kapelonis



Best Practice 2 – Create a Single package/binary/container for All Environments

You need to make sure that what is being tested and prodded within a pipeline is also the same thing that gets deployed.



Stage

Artifact from hash 5d23f

QA

Artifact from hash 6sds3f

Prod

Artifact from hash 23f45c

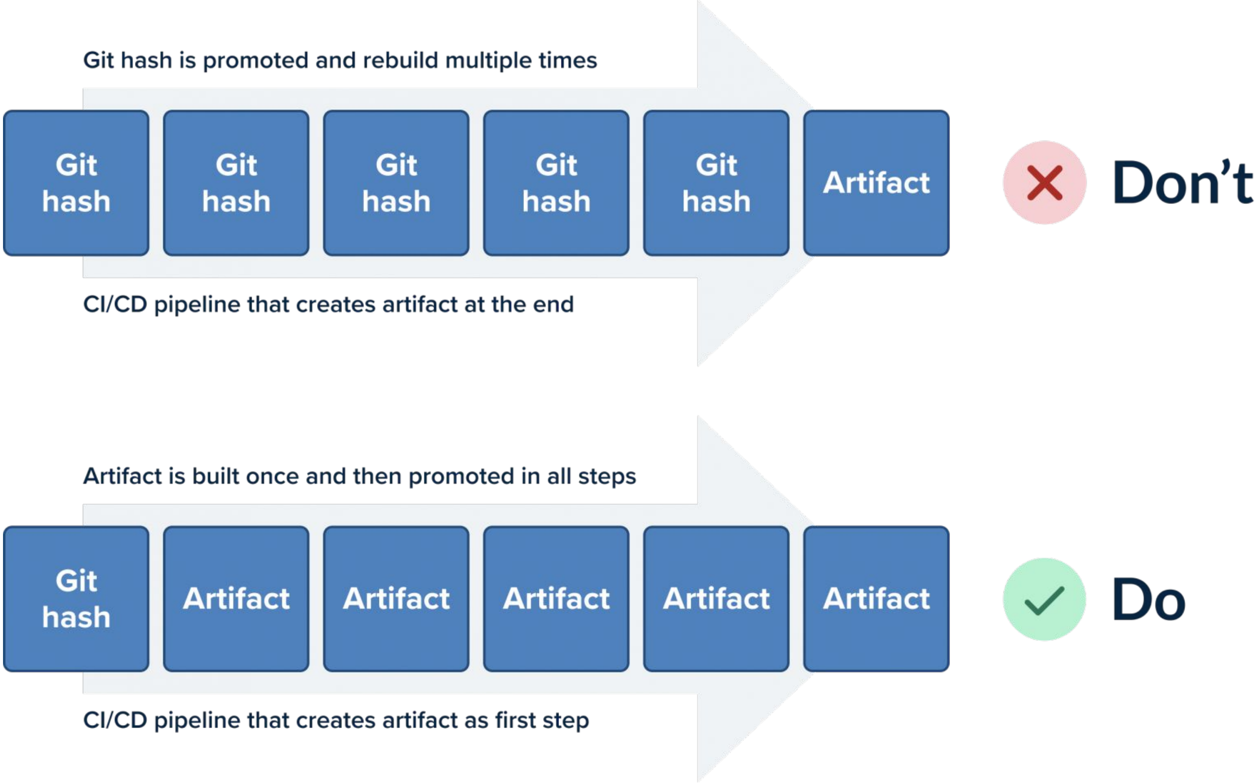
Build

Build

Source Code

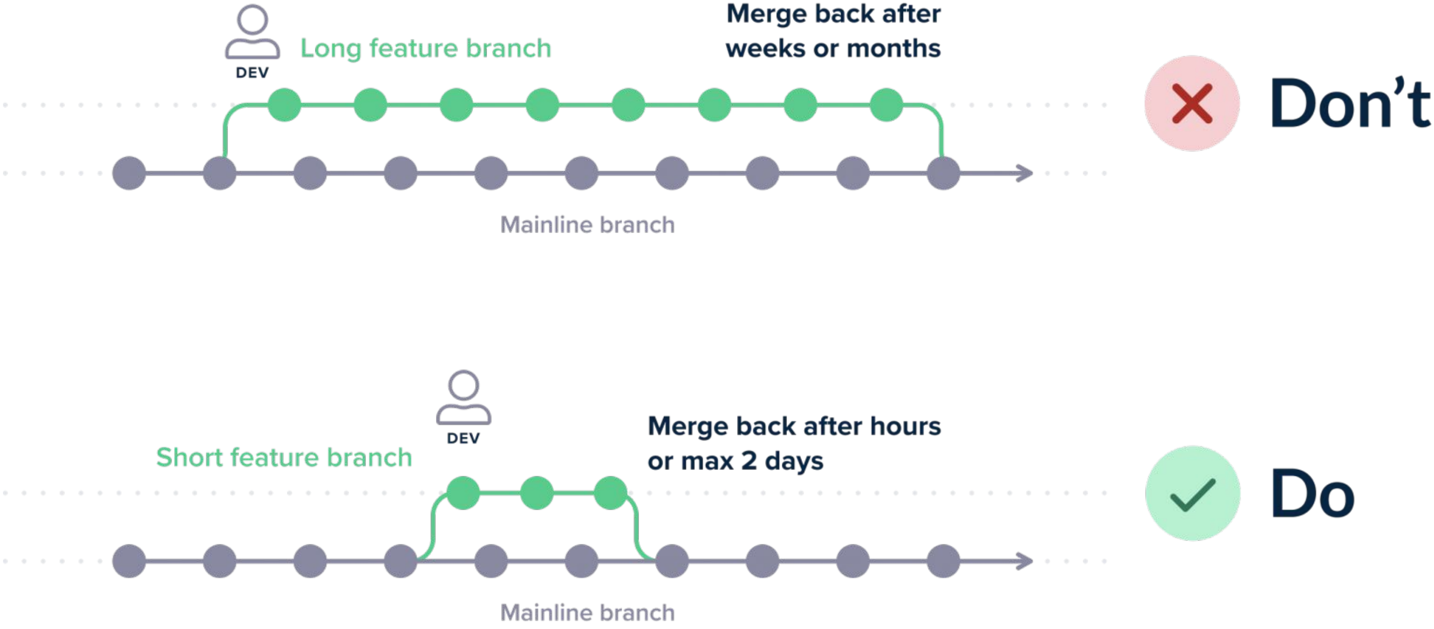
Build

✗ Don’t


Git hash is promoted and rebuild multiple times
Git hash
Git hash
Git hash
Git hash
Git hash
Artifact
✗ Don’t
CI/CD pipeline that creates artifact at the end
Artifact is built once and then promoted in all steps
Git hash
Artifact
Artifact
Artifact
Artifact
Artifact
✓ Do
CI/CD pipeline that creates artifact as first step

Best Practice 3 – Artifacts, not Git Commits, should travel within a Pipeline

A deployment artifact should be built only once. Once that artifact is built, it should move from each pipeline step to the next as an unchanged entity.



DEV

Long feature branch

Merge back after
weeks or months

✕ Don’t

Mainline branch

DEV

Short feature branch

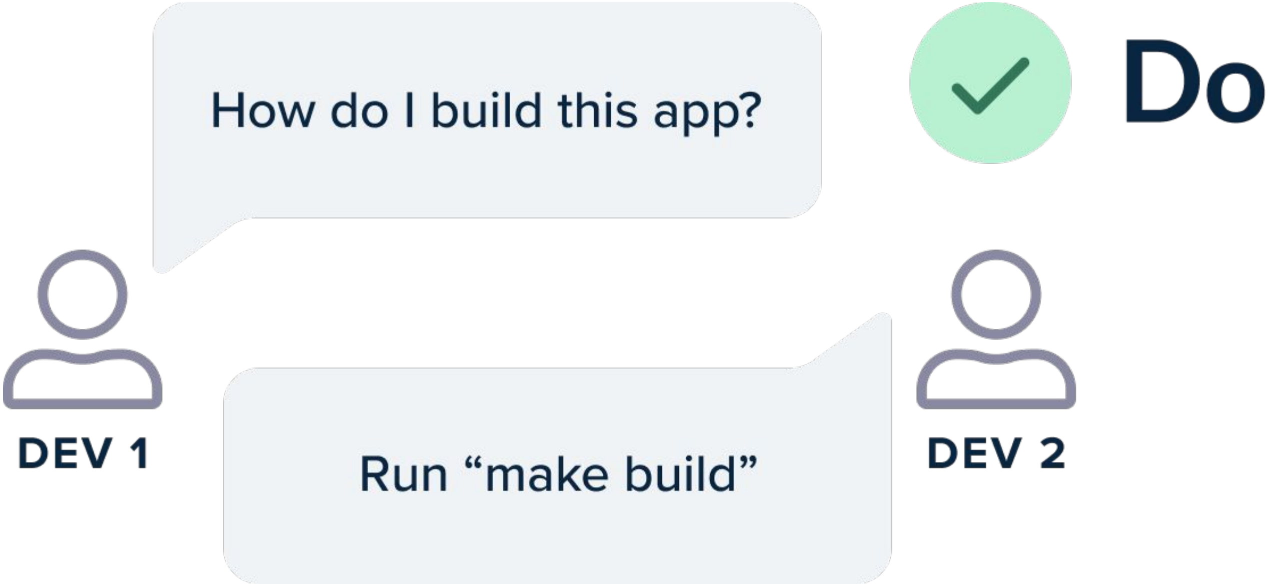
Merge back after hours
or max 2 days

✓ Do

Mainline branch

Best Practice 4 – Use short-lived Branches for each feature

The easiest way to do this is following the feature-per-branch methodology where short-lived features (i.e. that can fit within a single development sprint) correspond to individual source control branches.



How do I build this app?

Run “make build”

Best Practice 5 – A basic build should take a single step

Ideally, a simple build of a project should be a single command. That command usually calls the build system or a script that is responsible for taking the source code, running some basic tests, and packaging the final artifact/container.

✗ Don’t

DEV 1

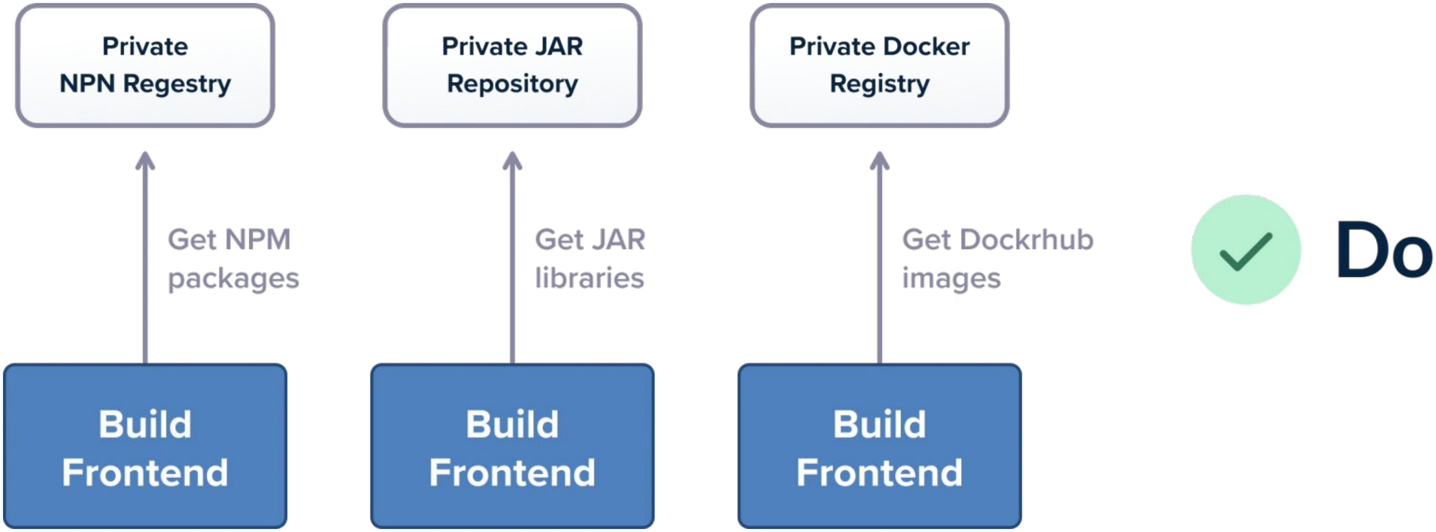
How do I build this app?

DEV 2

1. Copy properties from templates
2. Uncomment lines 34 and 56
3. Run the code generation step
4. Download dependencies
5. Enter backend folder
6. Run compilation
7. Enter frontend folder
8. Create assets
9. Run minification
10. ...

Best Practice 6 – Basic Builds are Fast (5 – 10 minutes)

A basic build should be really fast. Ideally less than five minutes. If it takes more than 10 minutes, your team should investigate the causes and shorten that time. Modern build systems have great caching mechanisms.



Private NPN Regestry

Private JAR Repository

Private Docker Registry

Get NPM packages

Get JAR libraries

Get Dockrhub images

✓ Do

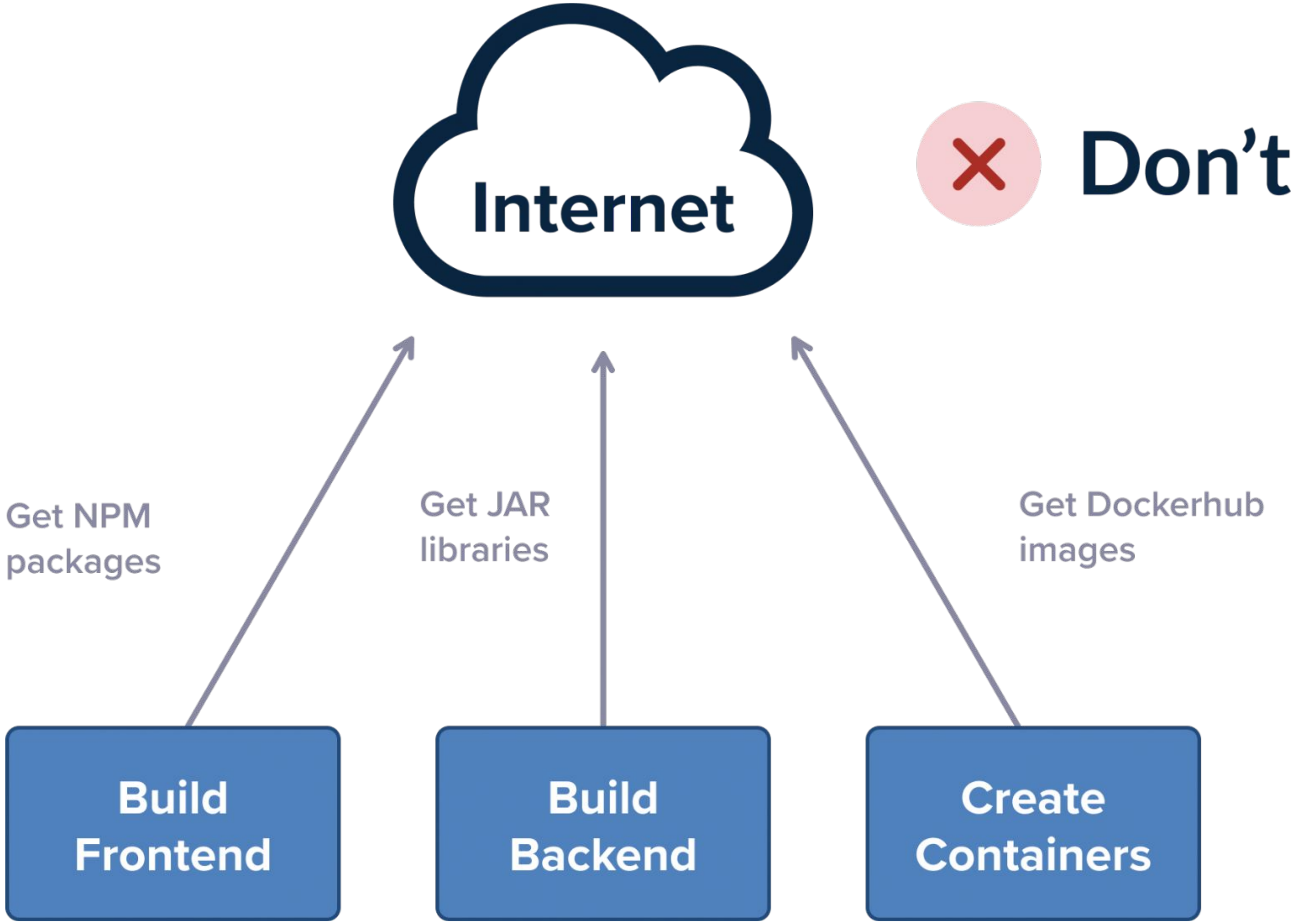
Build Frontend

Build Frontend

Build Frontend

Best Practice 7 – Store/Cache Your Dependencies

Every considerable piece of code uses external dependencies in the form of libraries or associated tools. Your code should of course be always stored in Git. But all external libraries should be also stored by you in some sort of artifact repository.



✕ Don’t

Internet

Get NPM packages

Get JAR libraries

Get Dockerhub images

Build Frontend

Build Backend

Create Containers