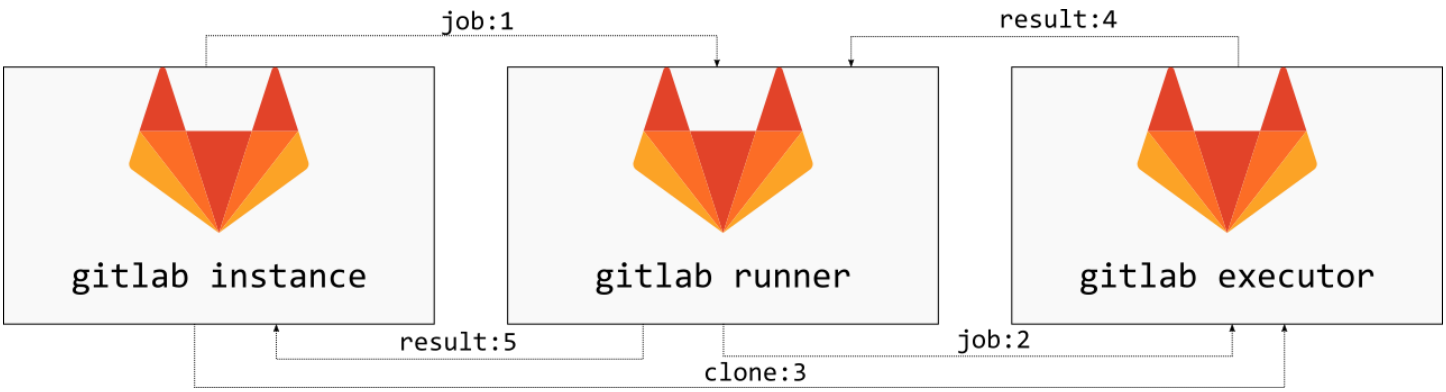
برای پیاده‌سازی خط لوله، از ابزار GitLab Runner می‌توانید استفاده کنید. گیتلب یک پلتفرم کنترل نسخه است که بستری برای فعالیت‌های دوآپس، از جمله یکپارچه‌سازی مستمر/تحویل مستمر، فراهم می‌آورد. سه مؤلفه در یکپارچه‌سازی مستمر/تحویل مستمر گیتلب نقش دارند:

۱. مؤلفه Gitlab Instance مانند gitlab.com

۲. مؤلفه GitLab Runner

۳. مؤلفه GitLab Executor

کدی که در خط لوله به کار می‌رود، به طور پیش‌فرض در فایلی به نام gitlab-ci.yml. قرار می‌گیرد.



شکل ۱. معماری گیتلب برای اجرای خط‌لوله

کوچکترین واحد مستقل قابل اجرا در گیتلب با نام Job شناخته می‌شود. GitLab Instance مسئولیت مدیریت کل خط لوله را بر عهده دارد. فایل gitlab-ci.yml. در مخزن کد GitLab Instance قرار می‌گیرد و در منبع کد آن، تعریف تعدادی Job مشاهده می‌شود. GitLab Instance برای اجرای خط لوله، Job های موجود در خط لوله را به GitLab Runner ارسال می‌کند. اجرای Job ها می‌تواند کاملاً مستقل از هم باشد. یک GitLab Runner مدیریت Job ارسالی را بر عهده دارد و باید در انتها نتیجه را به GitLab Instance بازگرداند. با این حال، GitLab Runner خودش یک Job را اجرا نمی‌کند، بلکه از GitLab Executor استفاده می‌کند. وظیفه GitLab Executor، اجرای Job ارسالی و بازگرداندن نتیجه به GitLab Runner است. GitLab Executor نقش مجری را دارد.

به طور خلاصه، مراحل اجرای خط لوله به شرح زیر است:

۱. GitLab Instance برای اجرای خط لوله، Job های آن را تشخیص می‌دهد و هر Job را برای GitLab Runner ارسال می‌کند.

۲. GitLab Runner یک Job را برای یک GitLab Executor ارسال می‌کند.

۳. GitLab Executor در ابتدا کد را از مخزن GitLab Instance بارگیری می‌کند.

۴. GitLab Executor سپس Job را اجرا می‌کند و نتیجه را به GitLab Runner بر می‌گرداند.

۵. GitLab Runner نتیجه دریافتی را برای GitLab Instance ارسال می‌کند.

۶. GitLab Instance نتیجه خط لوله را به کاربر اعلام می‌کند.

GitLab Runner می‌تواند به صورت مستقیم (باینری) بر روی سیستم عامل نصب شود یا از طریق داکر از خدمات آن استفاده کنید. نگارنده توصیه می‌کند که از نسخه داکر استفاده کنید. برای راه‌اندازی نسخه داکری، ابتدا باید اطمینان حاصل کنید که داکر در سیستم عامل میزبان نصب شده است. در ادامه، نحوه نصب داکر را بر روی سیستم عامل اوبونتو آموزش می‌دهیم.

اگر از قبل نسخه‌های قدیمی داکر وجود دارد، با دستور زیر می‌توانید داکر را حذف کنید.

```
for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done
```

آدرس مخزن داکر را به لیست مخازن اوبونتو اضافه می‌کنیم.

```
# Add Docker's official GPG key:

sudo apt-get update

sudo apt-get install ca-certificates curl gnupg

sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

sudo chmod a+r /etc/apt/keyrings/docker.gpg



# Add the repository to Apt sources:

echo \

  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \

  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \

  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update
```

حالا می‌توانیم داکر را نصب کنیم.

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

مطمئن شوید که سرویس داکر بعد از راه‌اندازی سیستم عامل شروع به کار می‌کند.

```
sudo systemctl enable docker

sudo systemctl start docker
```

کاربر فعلی را به گروه docker اضافه کنید تا نیازی نباشد هر بار دستور sudo را قبل از دستورات داکر بنویسید.

```
sudo groupadd docker

sudo usermod -aG docker $USER

newgrp docker
```

حالا می‌توانیم از GitLab Runner به صورت داکری استفاده کنیم. کافی است دستور زیر را در ترمینال وارد کنید تا هر زمان که سیستم عامل شروع به کار کرد، سرویس GitLab Runner نیز اجرا شود.

```
docker run -d --name gitlab-runner --restart always \

  -v /srv/gitlab-runner/config:/etc/gitlab-runner \

  -v /var/run/docker.sock:/var/run/docker.sock \

  gitlab/gitlab-runner:latest
```

هر GitLab Runner می‌تواند تعدادی GitLab Executor داشته باشد. GitLab Executor می‌تواند از نوع داکری، شِل یا ماشین مجازی باشد. نویسنده توصیه می‌کند که از نوع داکری آن استفاده کنید. مزیت آن این است که برای اجرای هر Job، یک کانتینر کاملاً جدید و ایزوله ایجاد می‌شود و در انتها کانتینر حذف می‌شود. در نتیجه، برای اجرا، محیطی دست نخورده خواهید داشت.

برای راه‌اندازی GitLab Executor، ابتدا وارد ترمینال GitLab Runner شوید.

```
docker exec -it gitlab-runner /bin/bash
```

فایل‌های پیکربندی GitLab Runner در آدرس /etc/gitlab-runner/config.toml درون کانتینر قرار می‌گیرند.

بعد از ورود به محیط ترمینال کانتینر GitLab Runner، تنظیماتی مشابه زیر را انجام دهید.

به جای <token-provided-by-gitlab>، توکنی که GitLab Instance در اختیارتان قرار می‌دهد را بنویسید.

به جای <gitlab-instance>، آدرس URL آن را بنویسید (مثلاً https://www.gitlab.com/).

```
concurrent = 1

check_interval = 0

shutdown_timeout = 0

[session_server]

  session_timeout = 1800

[[runners]]
```

```
name = "A gitlab runner (docker) with ubuntu 22.04 LTS"
url = <gitlab-instance>
...
token = <token-provided-by-gitlab>
...
executor = "docker"
[runners.custom_build_dir]
[runners.cache]
  MaxUploadedArchiveSize = 0
  [runners.cache.s3]
  [runners.cache.gcs]
  [runners.cache.azure]
[runners.docker]
  tls_verify = false
  image = "ubuntu:22.04"
  privileged = false
  disable_entrypoint_overwrite = false
  oom_kill_disable = false
  disable_cache = false
  volumes = ["/cache"]
  shm_size = 0
  allowed_pull_policies = ["always", "if-not-present"]
```

برای تنظیمات فوق و راحتی کار، می‌توانید دستور زیر را وارد نمایید و سپس به سوالاتی که می‌پرسد پاسخ دهید.

```
docker run --rm -it -v /srv/gitlab-runner/config:/etc/gitlab-runner gitlab/gitlab-runner register
```

نکته: برای استفاده از رانر خود در خط لوله، باید تمام تگ‌هایی را که برای رانر تعریف کرده‌اید به gitlab-ci.yml. اضافه کنید.