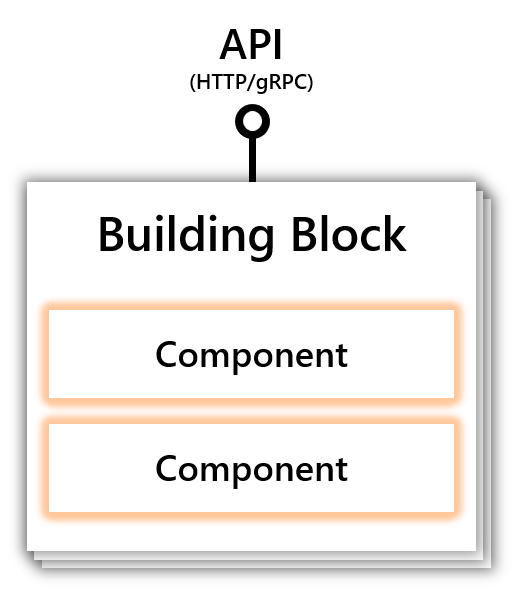
# Giới thiệu

Dapr (Distributed Application Runtime) là một framework mã nguồn mở giúp việc xây dựng các ứng dụng phân tán dễ dàng hơn. Dapr cung cấp một cách tiếp cận độc lập cơ sở hạ tầng để xử lý các vấn đề phức tạp như truy cập cơ sở dữ liệu, nhắn tin, theo dõi, quản lý trạng thái ứng dụng, và nhiều hơn nữa.

Với Dapr, các nhà phát triển có thể tập trung vào việc triển khai tính năng và logic của ứng dụng, thay vì phải quan tâm đến cách triển khai và quản lý hạ tầng phân tán.

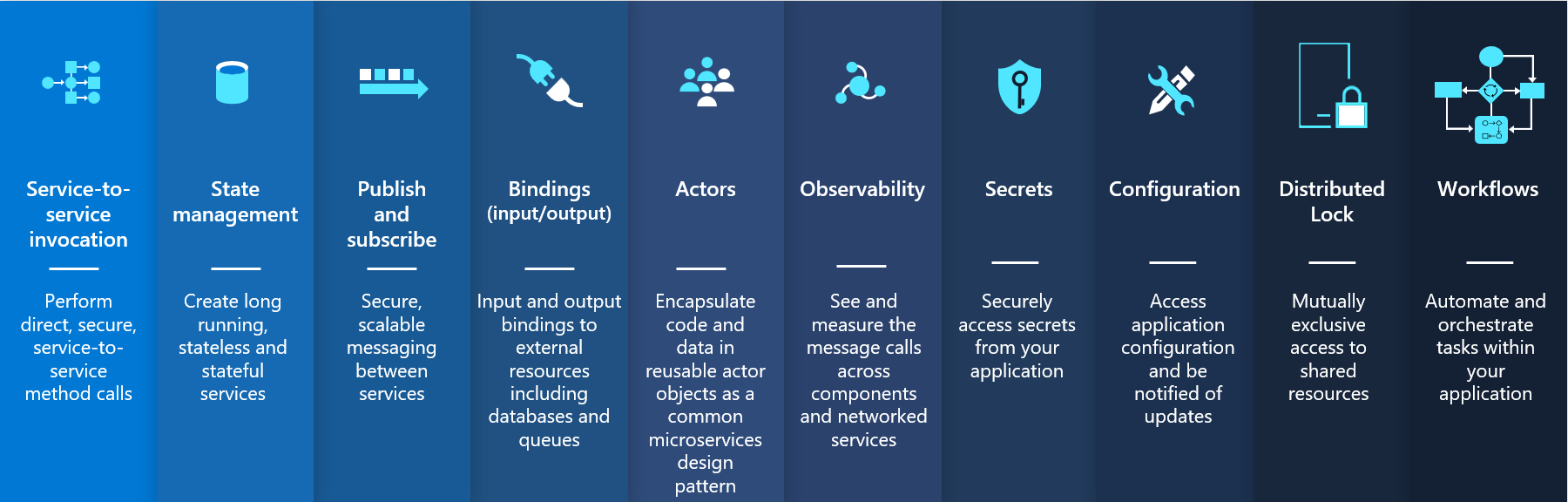
# Building block

Nó cung cấp một loạt các building block để giúp phát triển các ứng dụng phân tán dễ dàng hơn.



Các building block trong Dapr là các module phần mềm được tích hợp sẵn trong framework Dapr. Chúng được sử dụng để xây dựng các tính năng cơ bản của các ứng dụng phân tán, giúp cho việc phát triển và triển khai các ứng dụng phân tán dễ dàng hơn.

Các building block trong Dapr bao gồm:



## Service-to-service invocation:

cung cấp khả năng gọi các dịch vụ khác nhau thông qua các giao thức chuẩn như HTTP hoặc gRPC.

## State management:

cung cấp khả năng quản lý và lưu trữ trạng thái của ứng dụng.

## Publish and subscribe messaging:

cung cấp khả năng gửi và nhận các thông điệp (message) giữa các thành phần của ứng dụng.

## Event-driven compute:

cho phép xử lý các sự kiện (event) và thực thi mã logic tương ứng.

## Resource bindings:

cho phép kết nối và tương tác với các nguồn tài nguyên bên ngoài như CSDL, dịch vụ thư điện tử và các dịch vụ đám mây.

## Observability:

cung cấp khả năng theo dõi và giám sát các thành phần của ứng dụng.

# Component

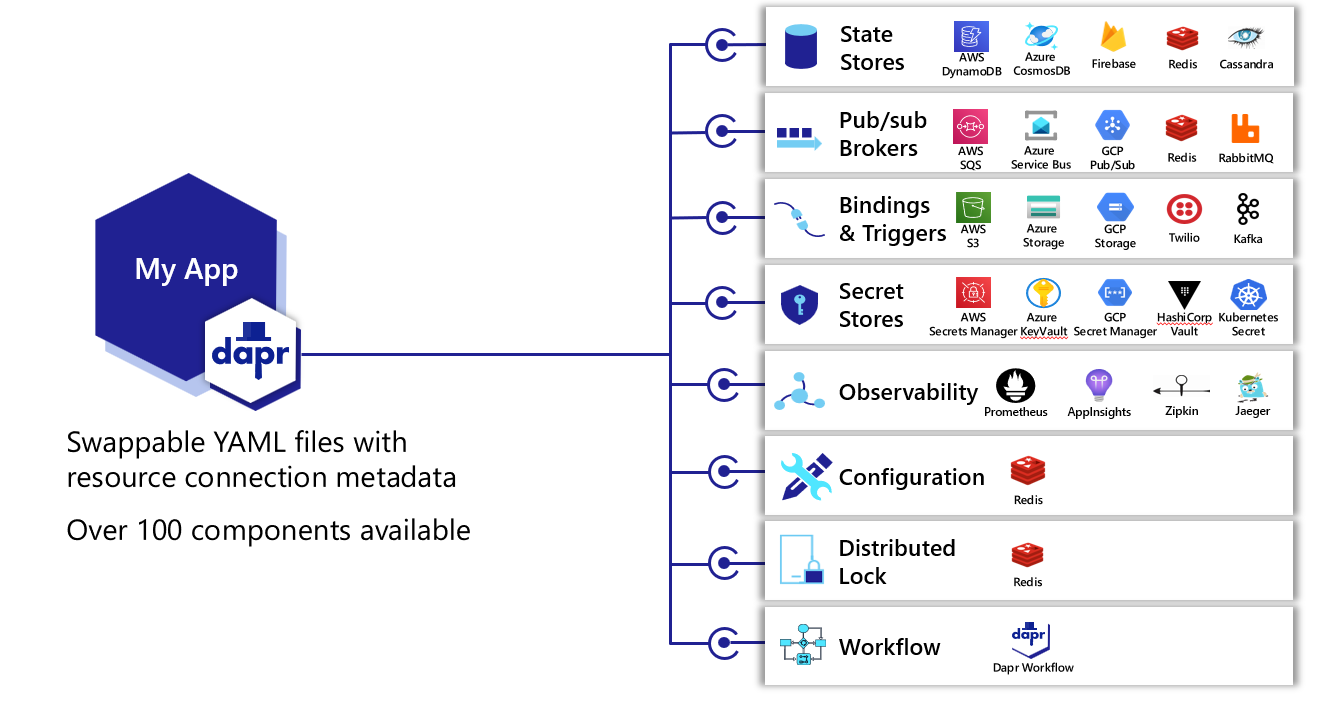
Các component được sử dụng để triển khai các tính năng của building block.

Mỗi component trong Dapr được định nghĩa bằng một tệp cấu hình YAML. Các tệp cấu hình này mô tả cách mà component được cấu hình để tương tác với các khối xây dựng khác. Ví dụ, một component có thể được cấu hình để truy cập vào một dịch vụ RESTful bằng cách chỉ định URL và các thông tin xác thực cần thiết.

Dapr cung cấp một số component được tích hợp sẵn, chẳng hạn như Redis để quản lý trạng thái và RabbitMQ để xử lý các thông điệp. Bên cạnh đó, người dùng cũng có thể tự định nghĩa và triển khai các component tùy chỉnh của riêng mình để tương tác với các nguồn tài nguyên khác nhau.

Các component trong Dapr giúp giảm độ phức tạp của việc phát triển và triển khai các ứng dụng phân tán bằng cách tách riêng các tính năng và phần mềm của hệ thống thành các thành phần độc lập nhau.

Sơ đồ dưới đây cho thấy một số ví dụ về các thành phần cho từng loại thành phần



# Cài đặt

Chú ý việc cài đặt cần cài đặt cần chú ý đến phiên bản và việc chọn sdk cũng nên phù hợp với phiên bản được cài đặt.

Trong hướng dẫn này sẽ thực hiện cài đặt dapr version mới nhất

## 1 cài đặt dapr cli

Thực hiện cài đặt dapr cli với lệnh sau

curl -s https://raw.githubusercontent.com/dapr/cli/master/install/install.sh | sudo DAPR\_VERSION=1.6 bash

sau khi cài đặt kiểm tra với lệnh

dapr –version

thực hiện cài đặt dapr vào k8s

tạo namescpaces trong k8s với lệnh sau

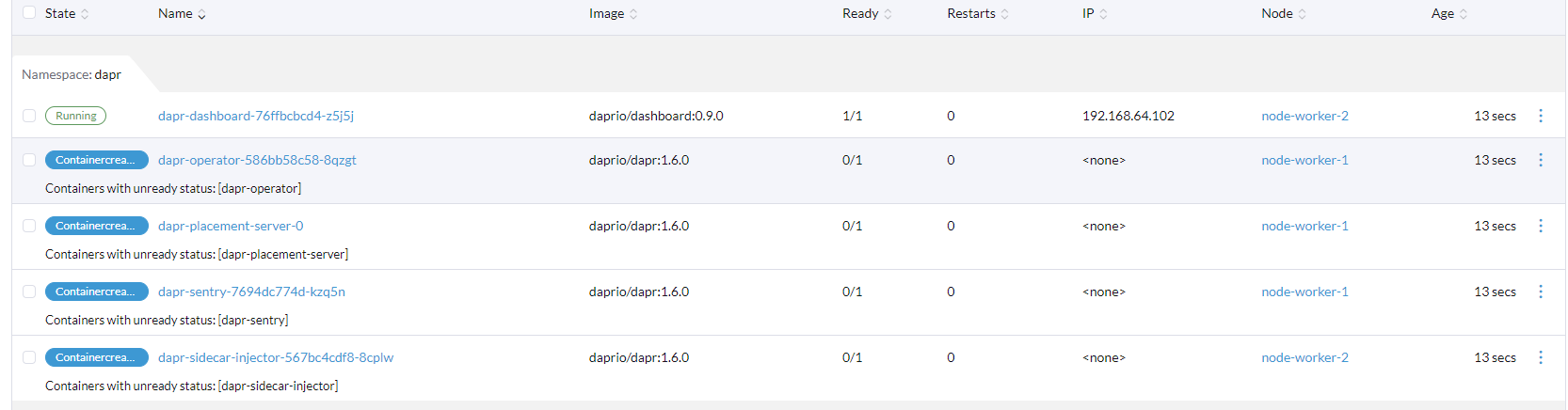
kubectl create namespace dapr

(nếu không tạo namespace thì mặc định khi cài đặt dapr lên k8s sẽ tạo 1 namespace là dapr-system)

thực hiện cài đặt dapr vào k8s với namescpace vừa tạo

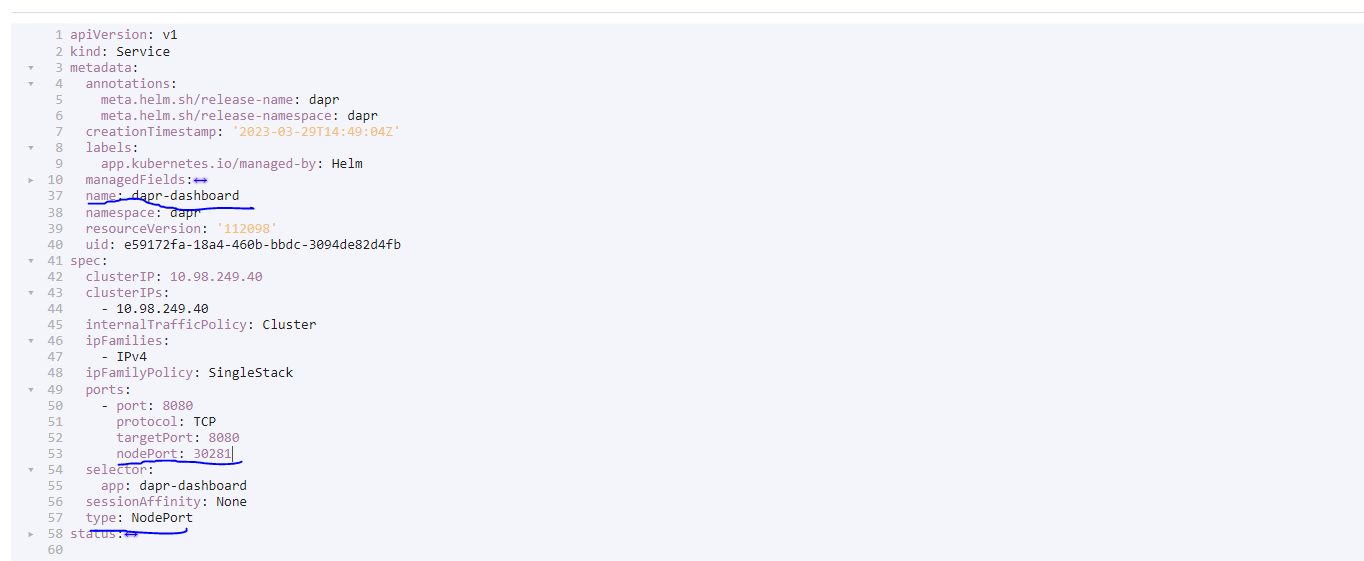
dapr init -k

các pod sẽ được khởi tạo ngay sau khi chạy lệnh

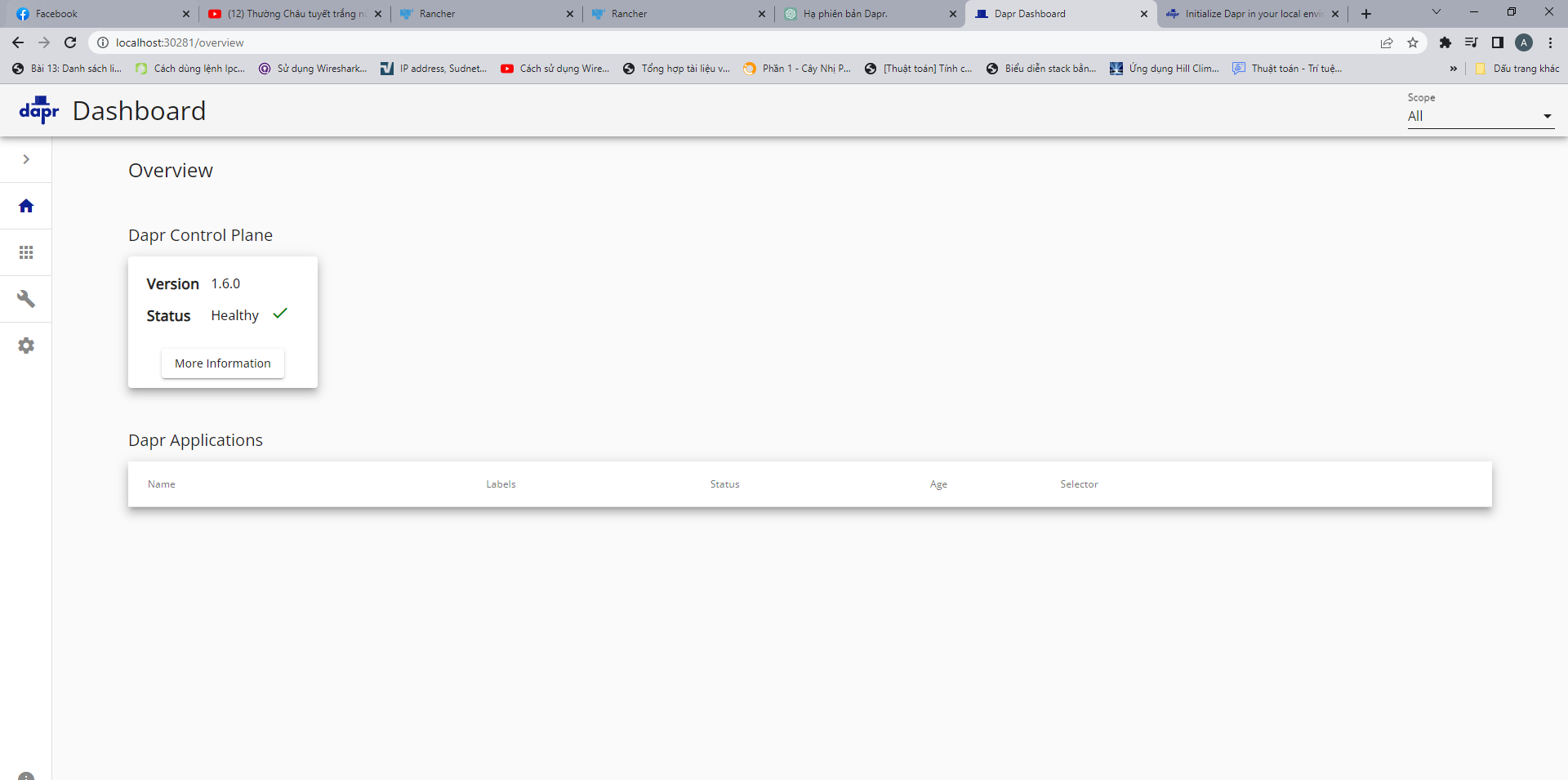


## 2 Chỉnh sửa cấu hình dapr doashboard để có thể truy cập từ bên ngoài

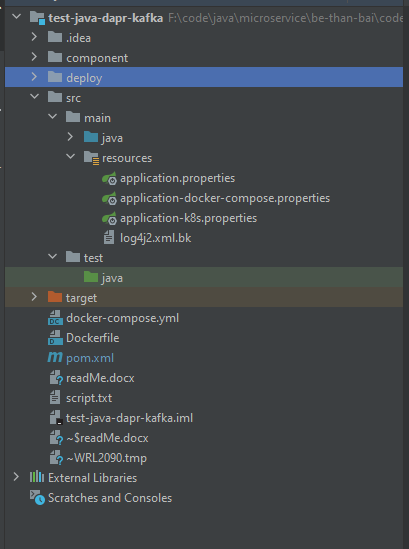
(chuyển service từ clusterIp sang nodeport)



Giờ ta có thể truy cập vào doashboard.



# Giới thiệu cấu trúc cơ bản của project



## Deploy

Chưa các khai báo về các deployment, service mà ta sử dụng trong project

## Componet

Chứa các componet của dapr

## Hướng dẫn chạy

Di chuyển vào thư muc deploy để thực hiện chạy kafka và redis

Kafka

kubectl apply -f zookeper.yaml -n dapr-system

kubectl apply -f zookeper-backup.yaml -n dapr-system

kubectl apply -f kafka.yaml -n dapr-system

kubectl apply -f kafka-backup.yaml -n dapr-system

chạy thêm kafdrop để xem chi tiết tin nhắn

kubectl apply -f kafdrop.yaml -n dapr-system

kubectl apply -f kafdrop\_backup.yaml -n dapr-system

redis

kubectl apply -f redis.yaml -n dapr-system

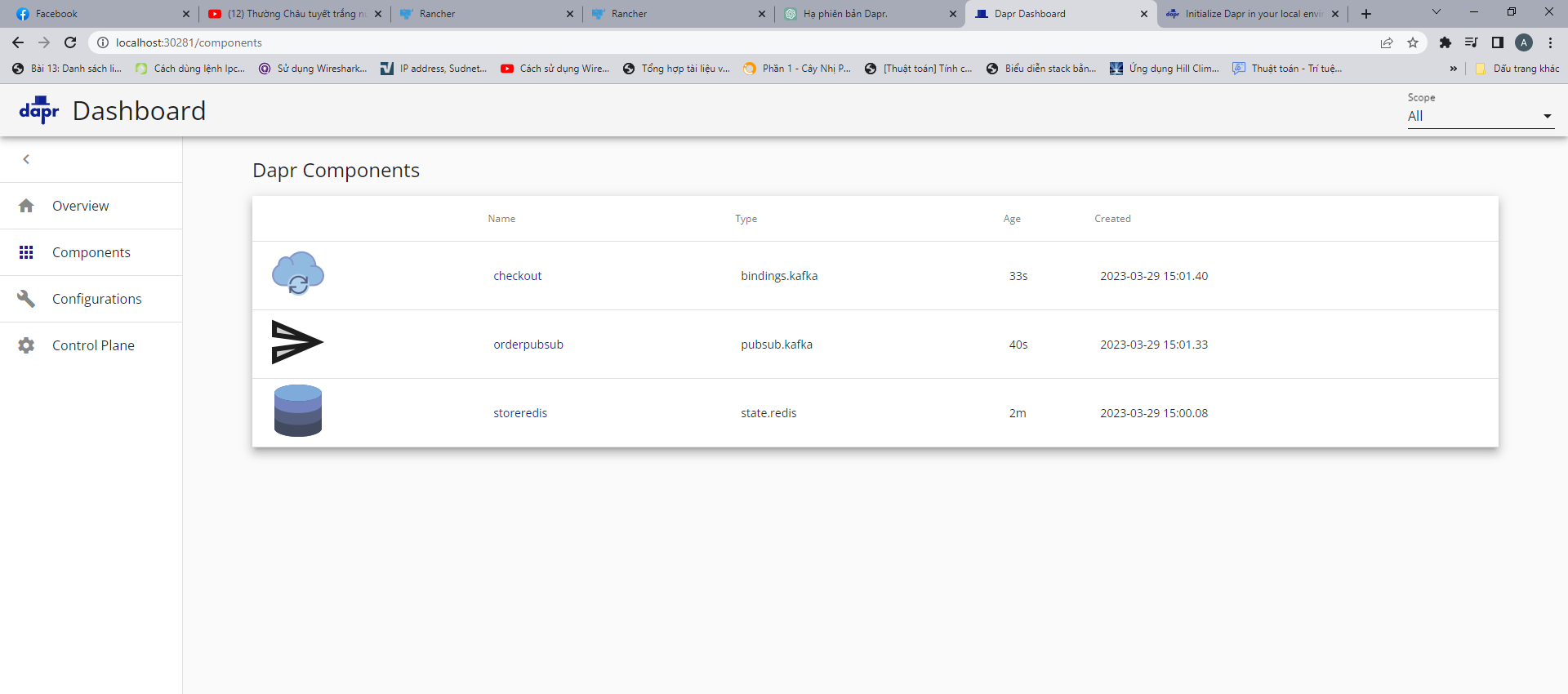
Di chuyển vào thực mục component và thực hiện lần lượt các lệnh sau

kubectl apply -f redis.yaml -n dapr-system

kubectl apply -f pubsub.yaml -n dapr-system

kubectl apply -f binding.yaml -n dapr-system

trở vê giao diện doashboard ta thấy các component đã được tạo



Di chuyển sang thư mục deploy thực hiện chạy lệnh sau

kubectl apply -f application.yaml -n dapr-system

để thực hiện chạy ứng dụng

các câu lệnh sử dụng trong ví dụ

curl -i localhost:30094/redis-dapr

curl -i localhost:30094/send-message

curl -i localhost:30094/send-pubsub

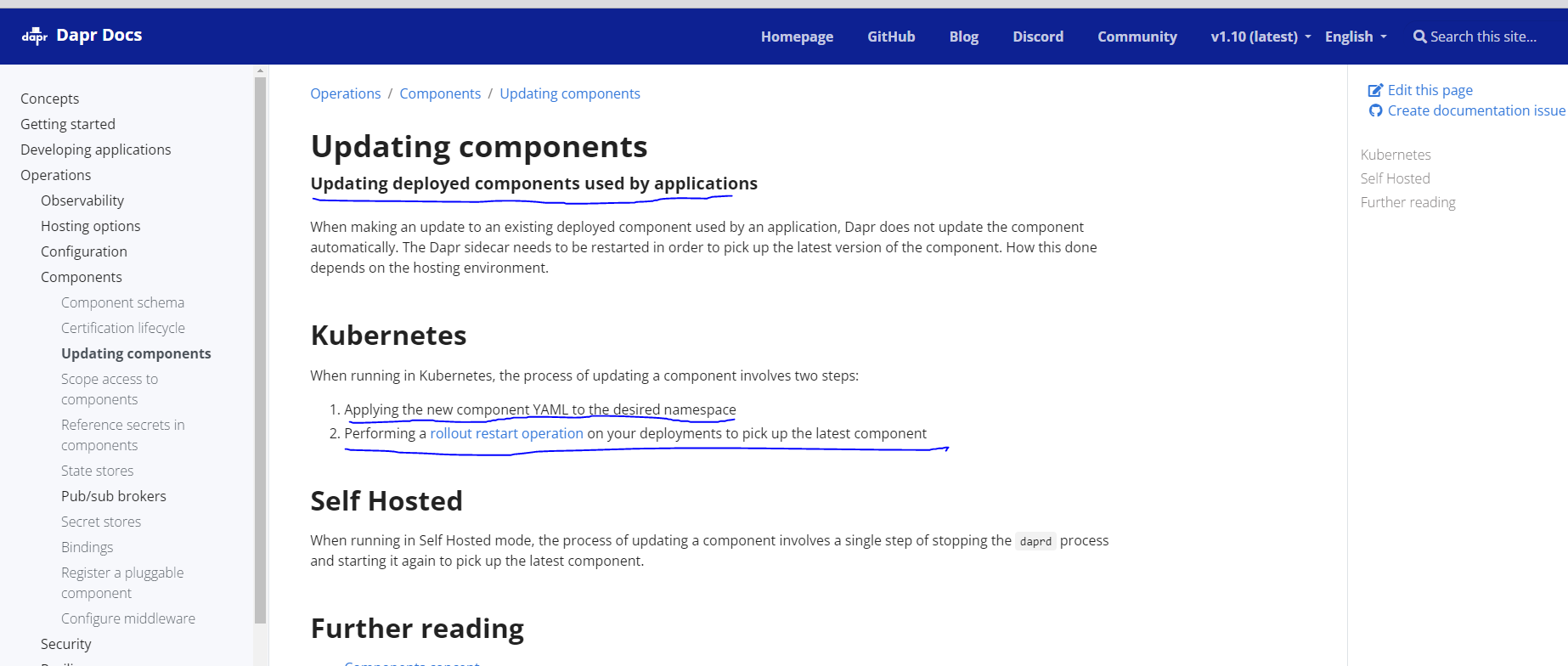
xóa thông tin pubsub cũ

kubectl delete -n dapr-system component orderpubsub

cập nhật thông tin pubsub mới

kubectl apply -f pubsub-backup.yaml -n dapr-system

muốn cập nhật cấu hình mới cần phải thực hiện



export PATH=/opt/apache-maven/bin:$PATH

286.660617484 cho pubsub

Chỉ cần 12.8 s cho kafka thông thường trong spring boot;