PIPELINE IN CLOUD

Camilo Bueno

Juan P Uribe

ESTRATEGIA DE BRANCHING

Desarrollo

Modelo propuesto: Git Flow Simplificado.

main: Contiene el código en producción, estable y desplegado.

develop: Contiene el código que agrupa los cambios en curso; aquí se integran las nuevas funcionalidades antes de pasar a producción.

feature/{nombre}: Ramas creadas a partir de develop para desarrollar nuevas funcionalidades.

ESTRATEGIA DE BRANCHING

Operaciones

Modelo propuesto: Infraestructura como Código + GitOps.

infra/main: Configuración estable de la infraestructura desplegada.

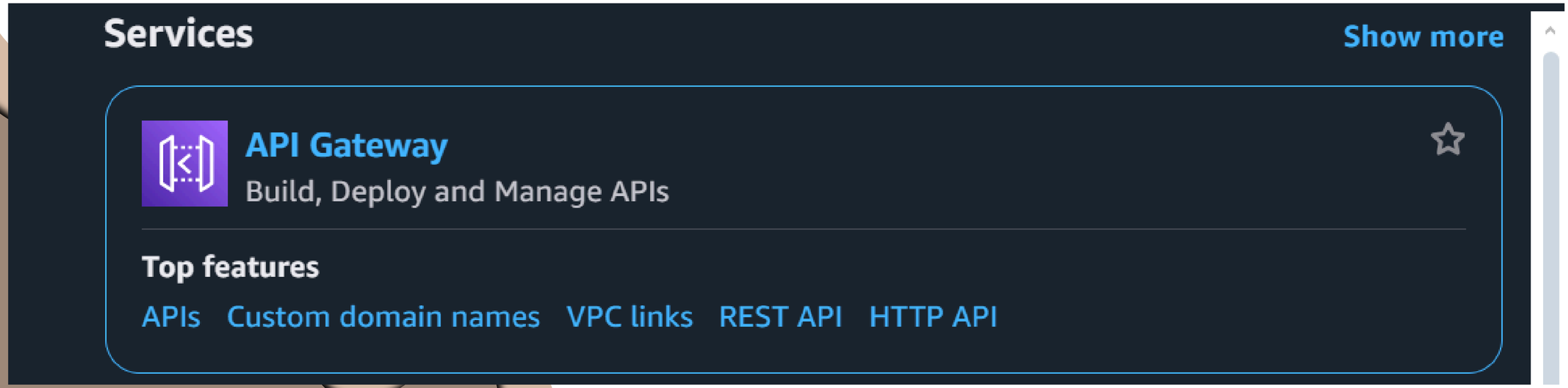
infra/dev: Cambios y pruebas de configuración en ambientes de desarrollo.

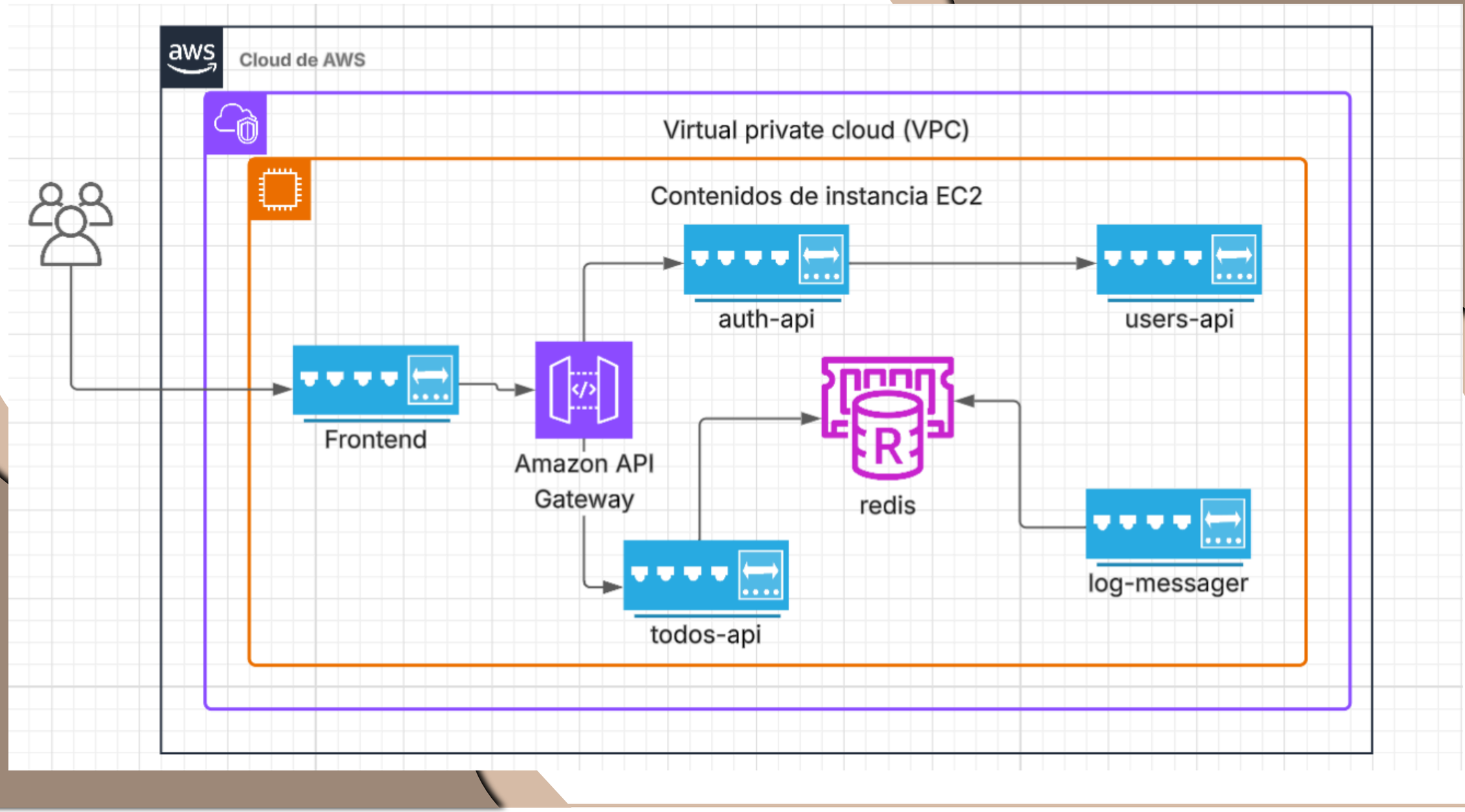
infra/feature/{nombre}: Para nuevas pruebas o configuración de recursos como pipelines, clusters, etc.


aws

PATRONES DE DISEÑO DE NUBE

API Gateway




aws
Cloud de AWS
Virtual private cloud (VPC)
Contenidos de instancia EC2
auth-api
users-api
Frontend
Amazon API Gateway
redis
log-messager
todos-api

PUBLISH.YML

```yaml
name: Publish to Docker

on:
  pull_request:
    branches:
      - main

permissions:
  packages: write
  contents: read

jobs:
  build-and-push:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout code
        uses: actions/checkout@v4

      - name: Login to GHCR
        uses: docker/login-action@v3
        with:
          registry: ghcr.io
          username: ${{ github.repository_owner }}
          password: ${{ secrets.GITHUB_TOKEN }}

      # auth-api
      - name: Docker meta auth-api
        id: meta-auth-api
        uses: docker/metadata-action@v5
        with:
          images: ghcr.io/solosoyjuan/auth-api
          tags: |
            type=raw,value=latest
            type=sha

      - name: Build and push auth-api
        uses: docker/build-push-action@v5
        with:
          context: ./auth-api
          push: true
          tags: ${{ steps.meta-auth-api.outputs.tags }}

      # frontend
      - name: Docker meta frontend
        id: meta-frontend
        uses: docker/metadata-action@v5
        with:
          images: ghcr.io/solosoyjuan/frontend
          tags: |
            type=raw,value=latest
            type=sha
```

DEPLOY.YML

```yaml
name: Deploy to EC2

on:
  workflow_run:
    workflows: ["Publish to Docker"]
    types:
      - completed

jobs:
  deploy:
    if: ${{ github.event.workflow_run.cor
    runs-on: ubuntu-latest

    steps:
      - name: Hola mundo
        run: echo "Este es un workflow de
```

```yaml
- name: Checkout code
  uses: actions/checkout@v4

- name: Copy docker-compose.yml to EC2
  uses: appleboy/scp-action@v0.1.3
  with:
    host: ${{ secrets.EC2_HOST }}
    username: ${{ secrets.EC2_USER }}
    key: ${{ secrets.EC2_SSH_KEY }}
    source: "docker-compose.yml"
    target: "/home/${{ secrets.EC2_USER }}/deploy"

- name: SSH into EC2 and deploy
  uses: appleboy/ssh-action@v1.0.0
  with:
    host: ${{ secrets.EC2_HOST }}
    username: ${{ secrets.EC2_USER }}
    key: ${{ secrets.EC2_SSH_KEY }}
    script: |
      cd ~/deploy
      echo ${{ secrets.GITHUB_TOKEN }} | docker login ghcr.io -u solosoyjuan
      docker-compose pull
      docker-compose up -d --force-recreate
```

G R A C I A S