GitHub as a Platform Engineering Platform

Enhance your experience, deploy in minutes.

bitovi

Where does this come from?

- No DevOps
- DevOps
- Platform Engineering

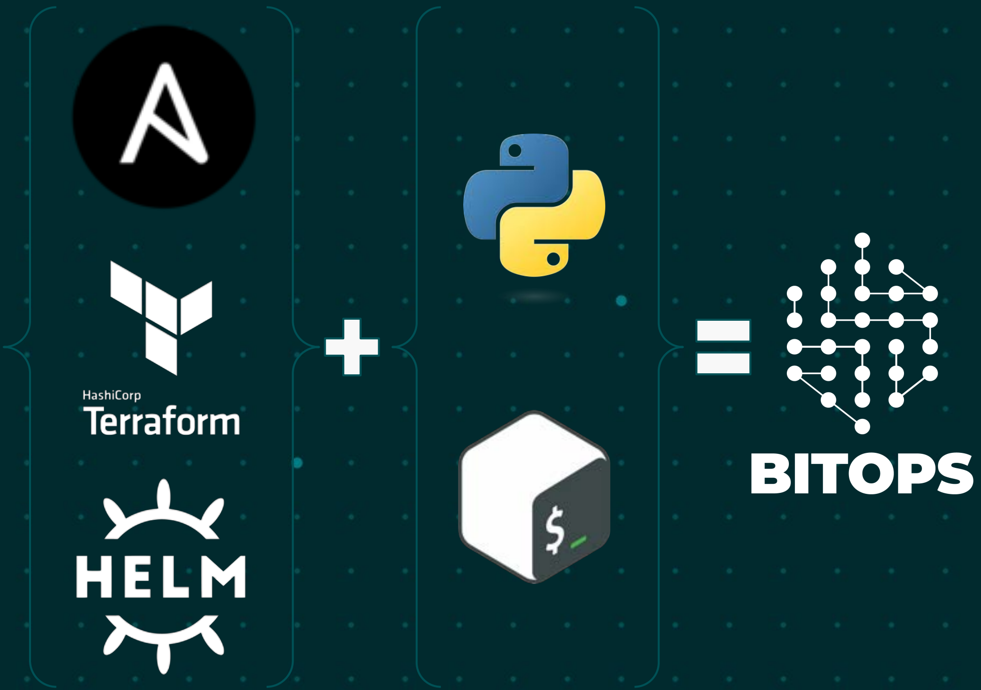
The (techno)logical evolution of - No DevOps

- Some scripts (Bash or Python)
- Manual resource creation
- Your code in some repo
- Lots of manual intervention
- Prone to fail
- Slow to deploy and update

The (techno)logical evolution of - DevOps

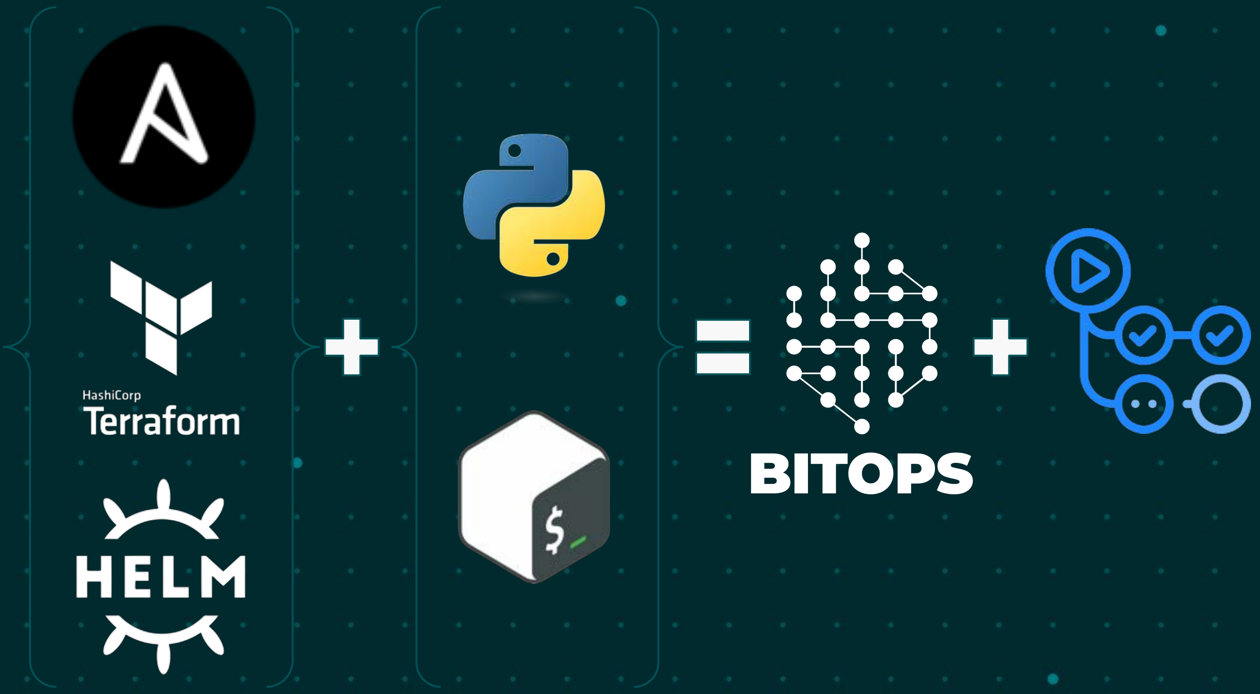
- Some IaC code to spin up your infrastructure
- Automated (up to a point)
- Still needs DevOps manual intervention
- Your code in some repo
- Slow to deploy, faster to update

The (techno)logical evolution of - DevOps



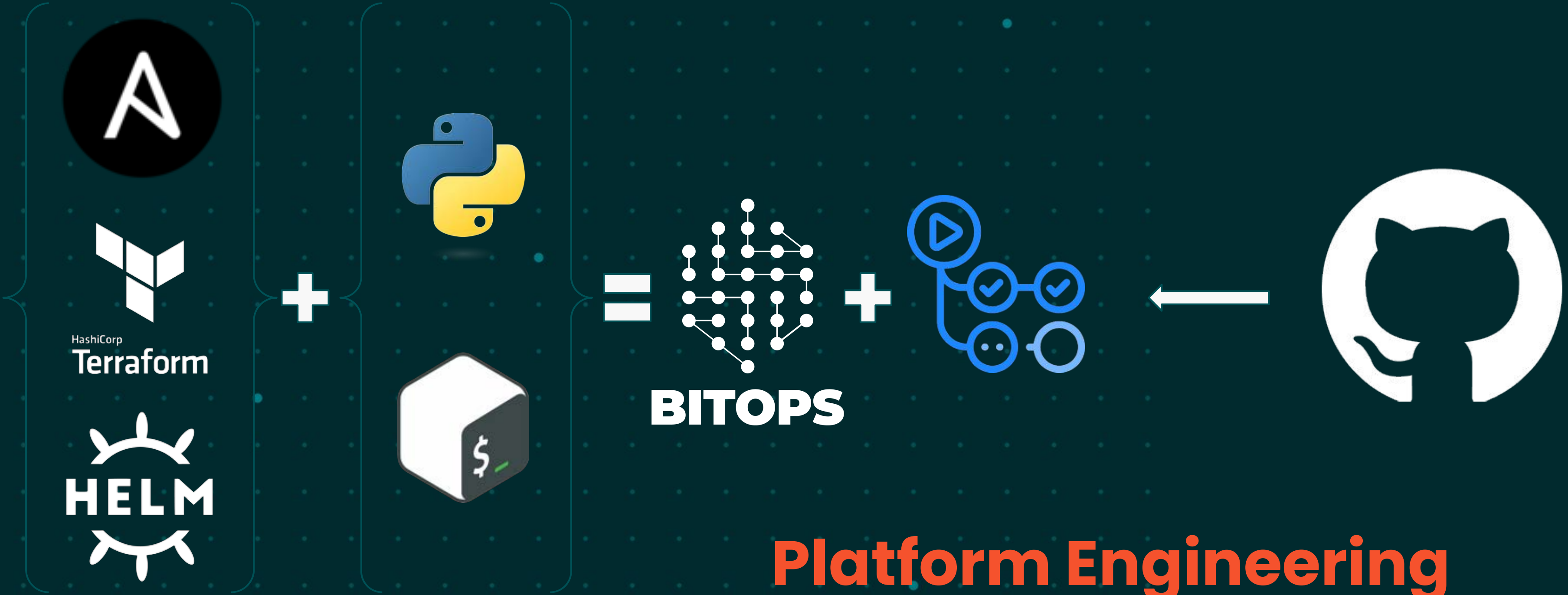
- Baked in IaC tools in a container. No installation needed.
- Yaml config files to tweak the behavior of the tools
- Reduced manual intervention
- Everything could live in the same repo
- Faster to deploy, faster to update

The (techno)logical evolution of - DevOps



- Config into action inputs
- Definitions visible in one file
- Everything in your repo, you just call the action
- Easy to automate deployments

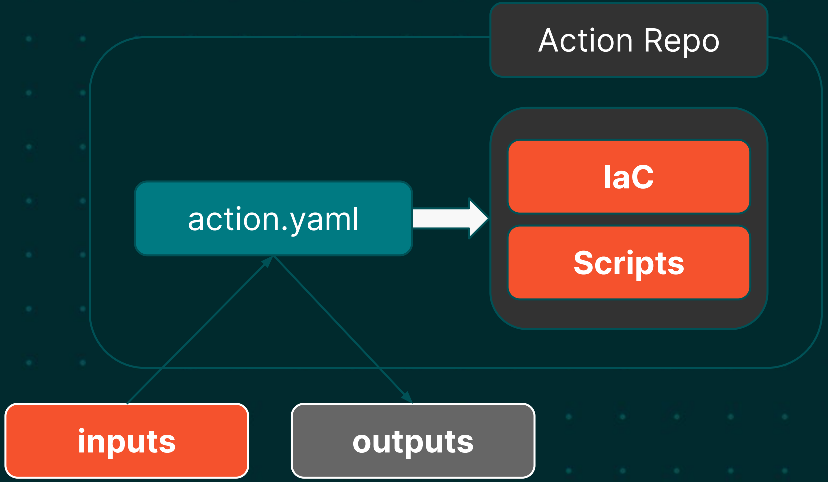
The (techno)logical evolution of - DevOps



Platform Engineering

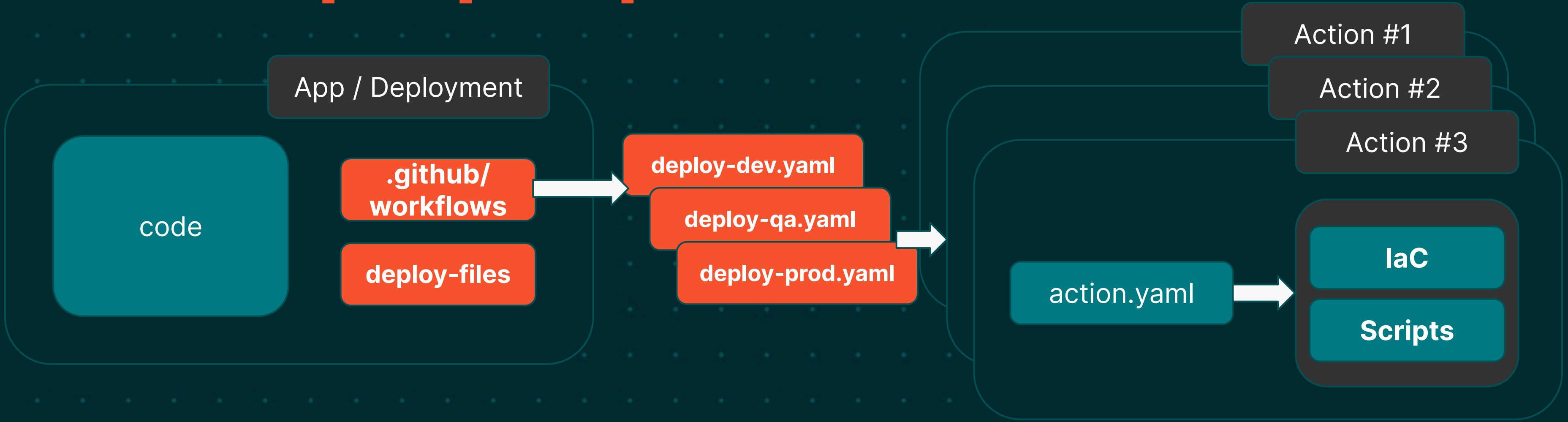
End-user experience

DevOps Engineer perspective



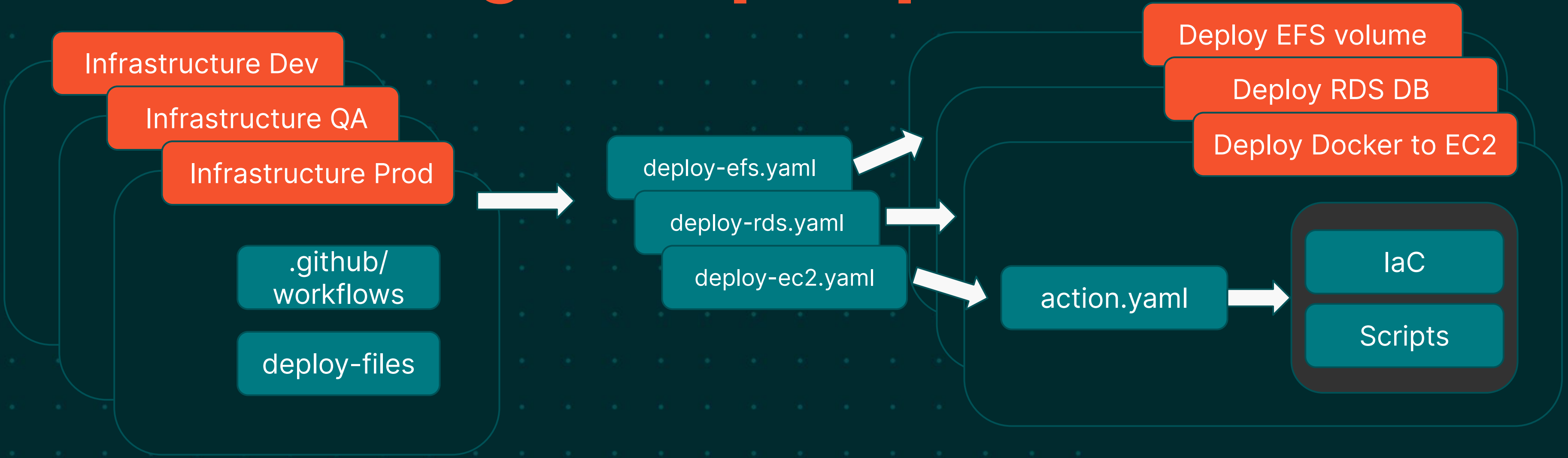
- Expose **inputs** for developers
- Could produce outputs
- Trigger **Scripts**, **IaC** or any command defined in code through steps
- Could call another action

Developer perspective



- Dev as consumer of actions. Just add workflows with steps.
- Could have specific **deploy-files** defined as inputs for the actions.
- Could chain and/or call multiple actions at the same time.
- Could be repo-based deployments, branch based, tag based...

Platform engineer perspective

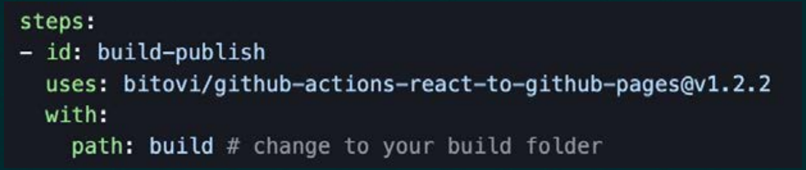


- Resources defined in workflows
- History tracking through repo
- Quick and easy **overview** of resources
- Fast deployments

Some examples of our GitHub Actions steps

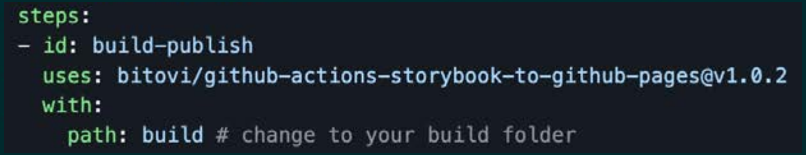
Deploy to GitHub Pages

React



```
steps:
- id: build-publish
  uses: bitovi/github-actions-react-to-github-pages@v1.2.2
  with:
    path: build # change to your build folder
```

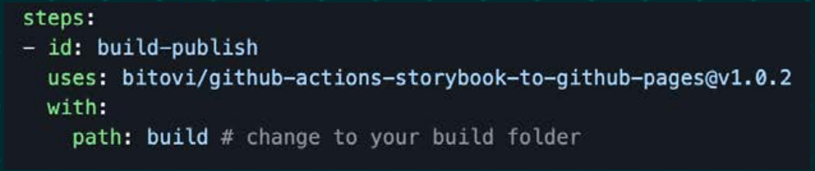
Storybook



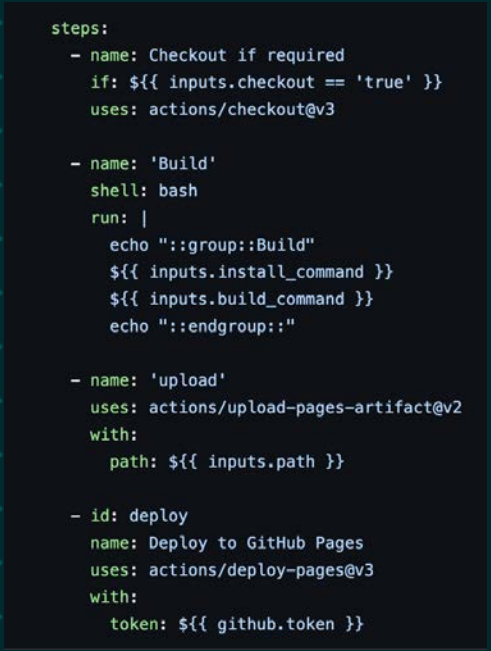
```
steps:
- id: build-publish
  uses: bitovi/github-actions-storybook-to-github-pages@v1.0.2
  with:
    path: build # change to your build folder
```

Storybook to GitHub Pages detail

By just adding this step inside of your deployment yaml file, you can get your Storybook deployment published in a GitHub page



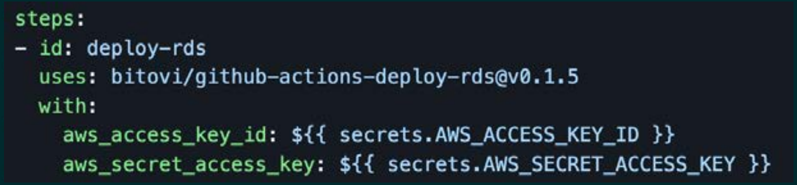
```yaml
steps:
- id: build-publish
  uses: bitovi/github-actions-storybook-to-github-pages@v1.0.2
  with:
    path: build # change to your build folder
```



```yaml
steps:
  - name: Checkout if required
    if: ${{ inputs.checkout == 'true' }}
    uses: actions/checkout@v3

  - name: 'Build'
    shell: bash
    run: |
      echo "::group::Build"
      ${{ inputs.install_command }}
      ${{ inputs.build_command }}
      echo "::endgroup::"

  - name: 'upload'
    uses: actions/upload-pages-artifact@v2
    with:
      path: ${{ inputs.path }}

  - id: deploy
    name: Deploy to GitHub Pages
    uses: actions/deploy-pages@v3
    with:
      token: ${{ github.token }}
```

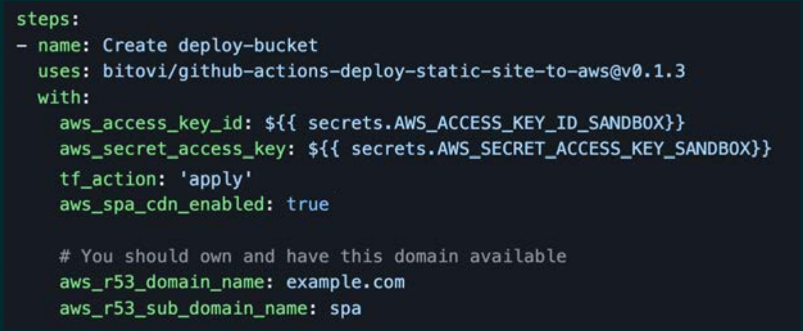
More complex deployments

RDS Database



```
steps:
- id: deploy-rds
  uses: bitovi/github-actions-deploy-rds@v0.1.5
  with:
    aws_access_key_id: ${{ secrets.AWS_ACCESS_KEY_ID }}
    aws_secret_access_key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
```

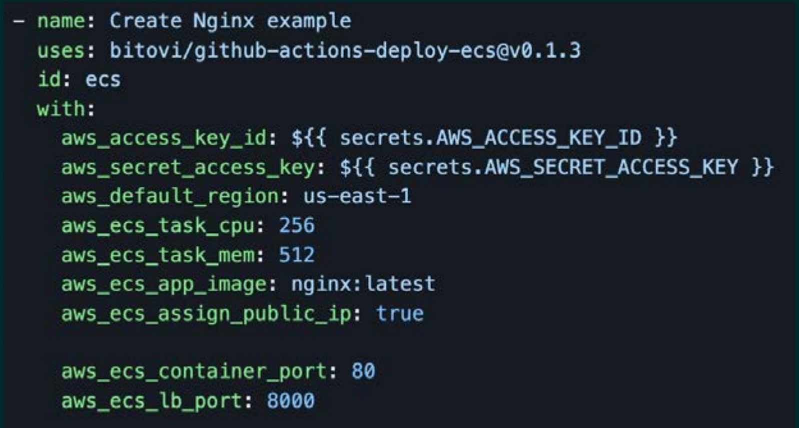
Static site to CDN



```
steps:
- name: Create deploy-bucket
  uses: bitovi/github-actions-deploy-static-site-to-aws@v0.1.3
  with:
    aws_access_key_id: ${{ secrets.AWS_ACCESS_KEY_ID_SANDBOX}}
    aws_secret_access_key: ${{ secrets.AWS_SECRET_ACCESS_KEY_SANDBOX}}
    tf_action: 'apply'
    aws_spa_cdn_enabled: true

    # You should own and have this domain available
    aws_r53_domain_name: example.com
    aws_r53_sub_domain_name: spa
```

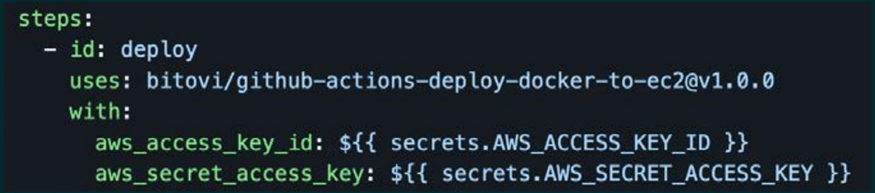
More complex deployments

ECS Cluster



```yaml
- name: Create Nginx example
  uses: bitovi/github-actions-deploy-ecs@v0.1.3
  id: ecs
  with:
    aws_access_key_id: ${{ secrets.AWS_ACCESS_KEY_ID }}
    aws_secret_access_key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
    aws_default_region: us-east-1
    aws_ecs_task_cpu: 256
    aws_ecs_task_mem: 512
    aws_ecs_app_image: nginx:latest
    aws_ecs_assign_public_ip: true

    aws_ecs_container_port: 80
    aws_ecs_lb_port: 8000
```

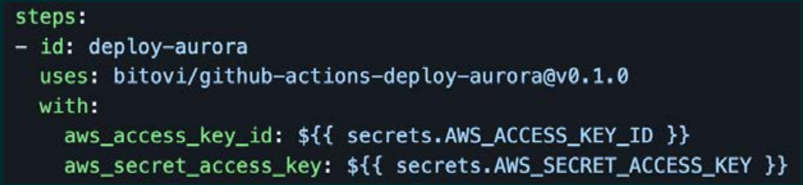
Docker to EC2



```yaml
steps:
  - id: deploy
    uses: bitovi/github-actions-deploy-docker-to-ec2@v1.0.0
    with:
      aws_access_key_id: ${{ secrets.AWS_ACCESS_KEY_ID }}
      aws_secret_access_key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
```

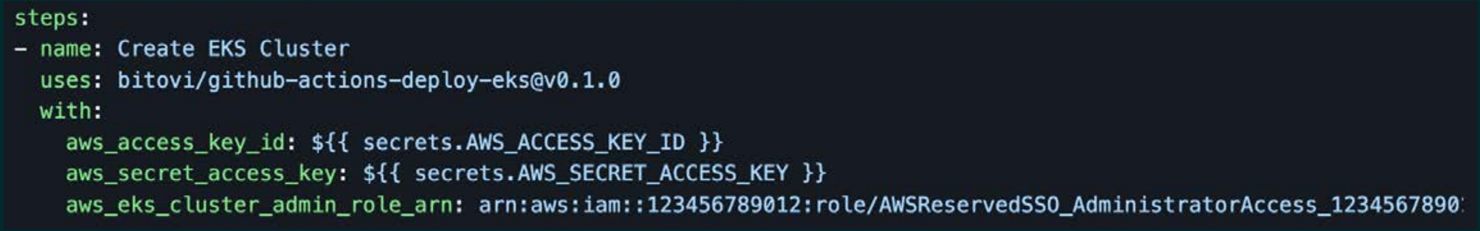
More complex deployments

Aurora DB Cluster



```yaml
steps:
- id: deploy-aurora
  uses: bitovi/github-actions-deploy-aurora@v0.1.0
  with:
    aws_access_key_id: ${{ secrets.AWS_ACCESS_KEY_ID }}
    aws_secret_access_key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
```

EKS Cluster



```yaml
steps:
- name: Create EKS Cluster
  uses: bitovi/github-actions-deploy-eks@v0.1.0
  with:
    aws_access_key_id: ${{ secrets.AWS_ACCESS_KEY_ID }}
    aws_secret_access_key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
    aws_eks_cluster_admin_role_arn: arn:aws:iam::123456789012:role/AWSReservedSSO_AdministratorAccess_1234567890
```

Some of our GitHub Actions

- React to GitHub Pages
- Storybook to GitHub Pages
- Static site to AWS (S3+CDN+R53)
- Docker Build Tag Publish
- Deploy Prometheus and Grafana
- Deploy Stackstorm Single VM
- Deploy Helm to EKS
- Deploy Docker to EC2
- Deploy EKS Cluster
- Deploy ECS Cluster
- Deploy Aurora DB Cluster
- Deploy RDS DB instance
- Deploy Redis DB Cluster (AWS)
- Deploy EFS
- Deploy GitHub Runner

And a lot more!

Search for **Bitovi** in the GitHub Actions Marketplace or reach us through Discord!

We ❤ OpenSource

Thanks!

Keep in touch

leo@bitovi.com

in linkedin.com/in/leonardodiazlonghi

bitovi