Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

5 Kubernetes 배포를 위한 GitHub와 ArgoCD 활용

02 Github Repository 생성 및 설정

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

실습 내용

순서

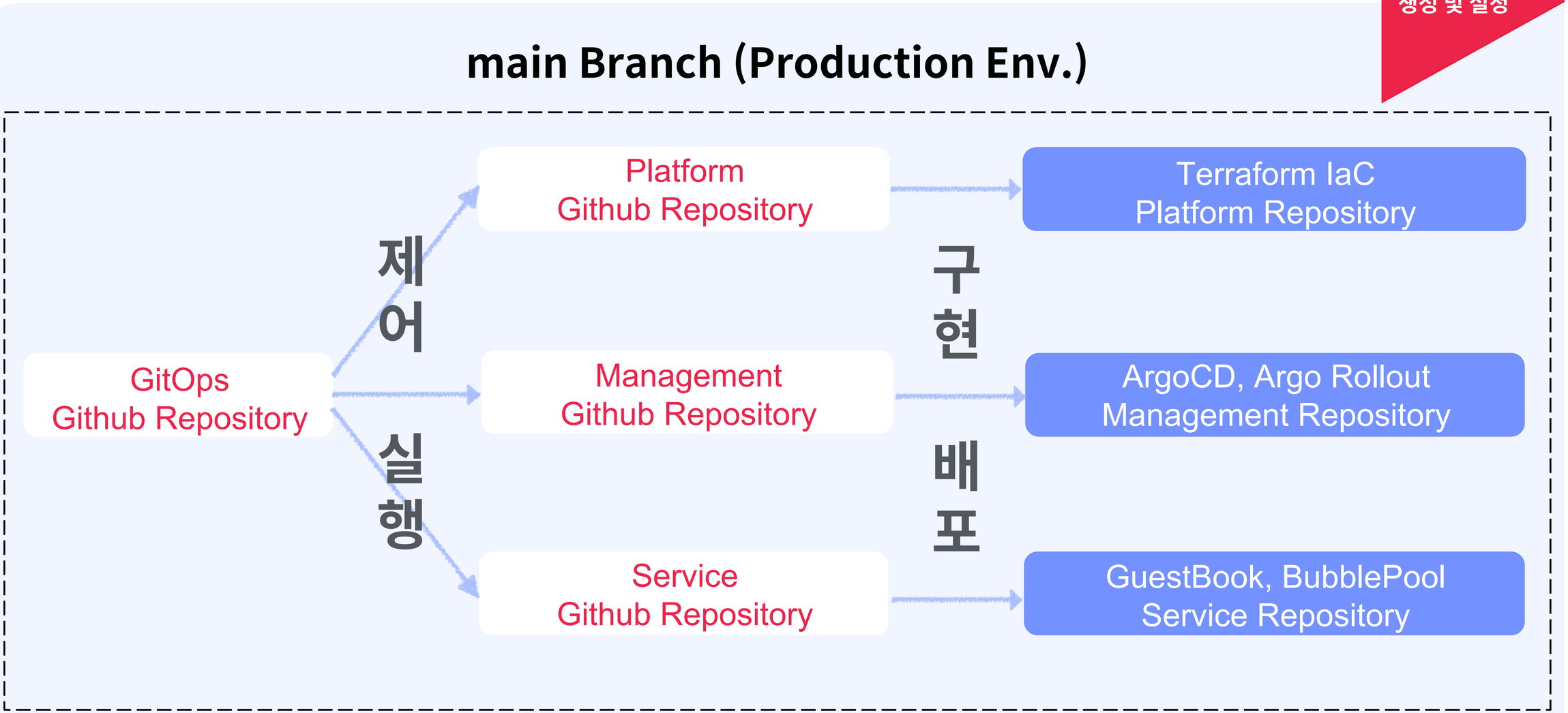
1. Github Repository 구성 설계
2. SSH 관리를 위한 Key 생성 및 Github 등록
3. gitops-repository 구성 확인
4. platform-repository 구성 확인
5. GitOps를 통한 Terraform으로 AWS 인프라/플랫폼 프로비저닝

실습 예제코드 경로

Chapter05

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

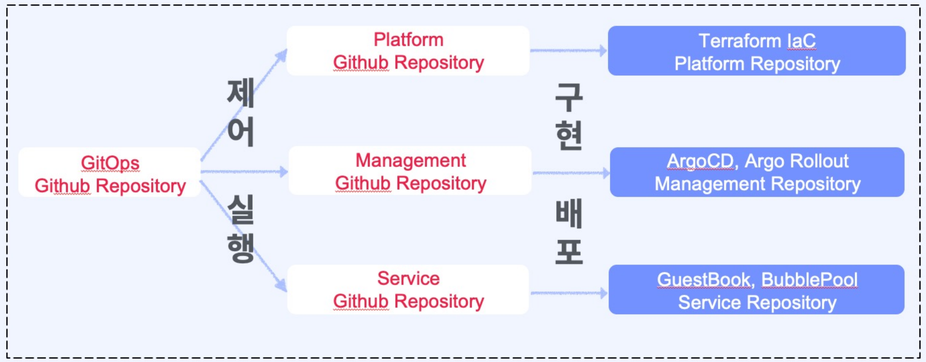
1. Github Repository 구성 설계 #1



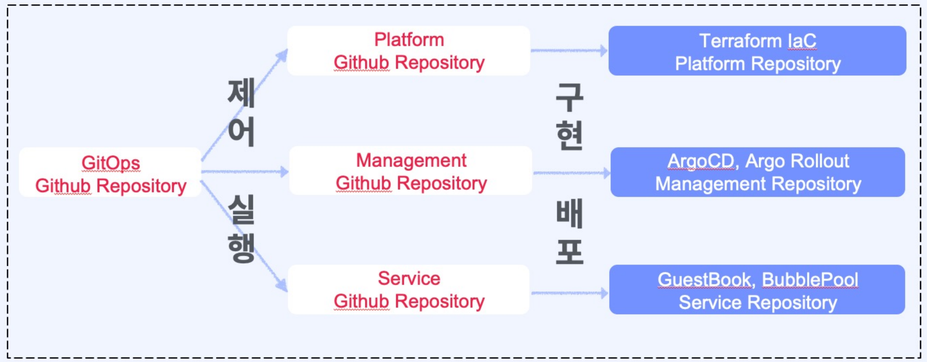
Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

1. Github Repository 구성 설계 #2

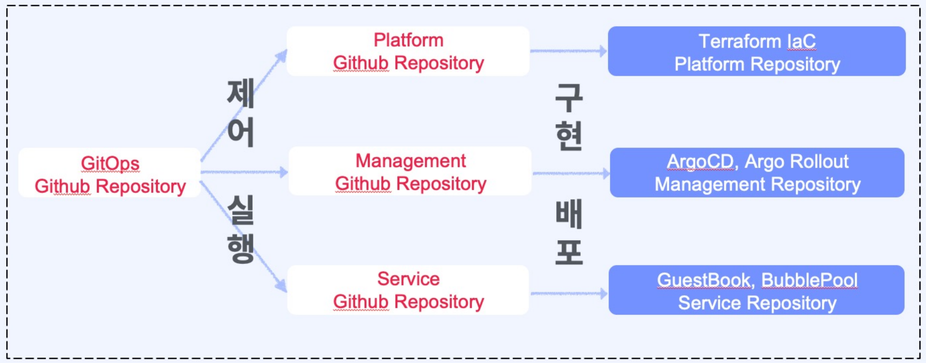
main Branch (Prod Env.)



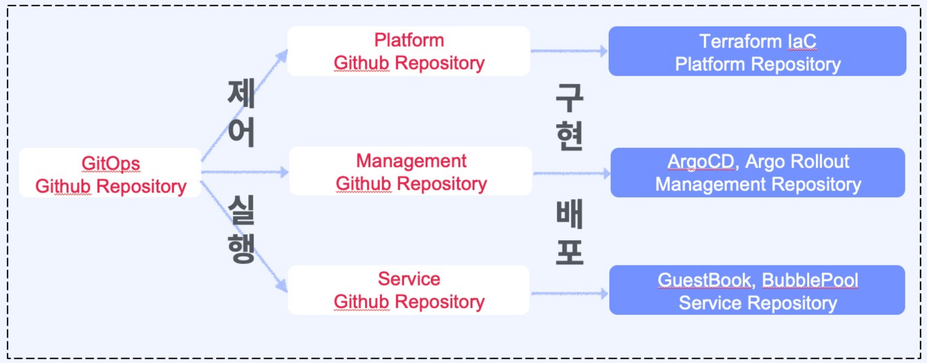
staging Branch (QA Env.)



develop Branch (Dev Env.)



sandbox Branch (Test Env.)



Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

2. SSH 관리를 위한 Key 생성 및 Github 등록 #1

SSH 관리를 위한 Key 생성

```
$ ssh-keygen -q -t rsa -N '' -m PEM -t rsa -b 4096 -C test -f ./id_rsa <<<y >/dev/null 2>&1
```

생성후 OS별 SSH를 Agent에 등록 방법 (등록이 되어야 항상 SSH로 Git 접근이 가능)

- **MAC OS**

```
$ ssh-add id_rsa
```

- **Linux**

1. `$ eval `ssh-agent``
2. `$ ssh-add id_rsa`

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

2. SSH 관리를 위한 Key 생성 및 Github 등록 #2

Github 등록 방법

1. Github 로그인 후 > 우측 상단 아이콘 클릭 > 맨 아래서 2번째 위 "Settings" 클릭
2. 좌측 메뉴 중 중간에 Access > SSH and GPG keys 클릭
3. 우측 상단 "New SSH key" 클릭 > Title은 식별가능한 이름으로, Key는 위에서 생성한 id_rsa.pub의 Key 내용을 복사, 붙여넣기 > "Add SSH key" 버튼을 눌러 생성 완료
4. 우측 상단 클릭 > "Your repositories" 클릭 > 특정 Repository 클릭 > 중간 우측에 있는 "Code" 버튼 클릭 > Clone > SSH 내역 복사 (git@github.com:본인의 repository 주소)
5. 터미널에서 특정 디렉토리로 이동 > git clone (git@github.com:본인의 repository 주소)
6. 정상적으로 git clone이 되었는지 확인

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

3. gitops-repository 구성 확인 #1

```
.
├── README.md
├── management
│   ├── argo-cd
│   │   ├── manifests
│   │   │   └── kustomization.yaml
│   │   └── scripts
│   │       └── setup.sh
│   └── argo-rollout
│       ├── manifests
│       │   └── kustomization.yaml
│       └── scripts
│           └── setup.sh
├── platform
│   └── aws
│       └── ap-northeast-2
│           ├── terraform-backend
│           │   └── iac.tf
│           └── terraform-codes
│               └── iac.tf
└── service
    ├── bubblepool
    │   └── application.yaml
    └── guestbook
        └── application.yaml
```

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

3. gitops-repository 구성 확인 #2

Service Github Repository 참조 방식 (ArgoCD > application.yaml)

git@github.com:<사용자 Org명>/service-repository.git

Management Github Repo 참조 방식 (Kustomize > Kustomization.yaml)

https://github.com/<사용자 Org명>/management-repository//argo-cd?ref=main

Platform Github Repository 참조 방식 (Terraform > iac.yaml)

git@github.com:<사용자 Org명>/platform-repository.git//aws/ap-northeast-2/terraform-backend?ref=main

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

4. platform-repository 구성 확인

```
.
├── README.md
└── aws
    └── ap-northeast-2
        ├── terraform-backend
        │   ├── providers.tf
        │   ├── terraform-backend.tf
        │   └── variables.tf
        └── terraform-codes
            ├── eks-cluster.tf
            ├── eks-nodegroup.tf
            ├── iam-roles.tf
            ├── internet_gateway.tf
            ├── providers.tf
            ├── route_table.tf
            ├── route_table_association.tf
            ├── security-group.tf
            ├── subnet.tf
            ├── variables.tf
            └── vpc.tf
```

Fast campus Copyright FASTCAMPUS Corp.All Rights Reserved

5. GitOps를 통한 Terraform으로 AWS 인프라/플랫폼 프로비저닝

terraform 초기화 명령어 (Init)

$ terraform init

terraform 코드 문법 확인 및 실행 검증(Dry-Run)

$ terraform plan

terraform 프로비저닝 수행 및 리소스 생성 (Run)

$ terraform apply