**틀라우드 클라우드 구축**

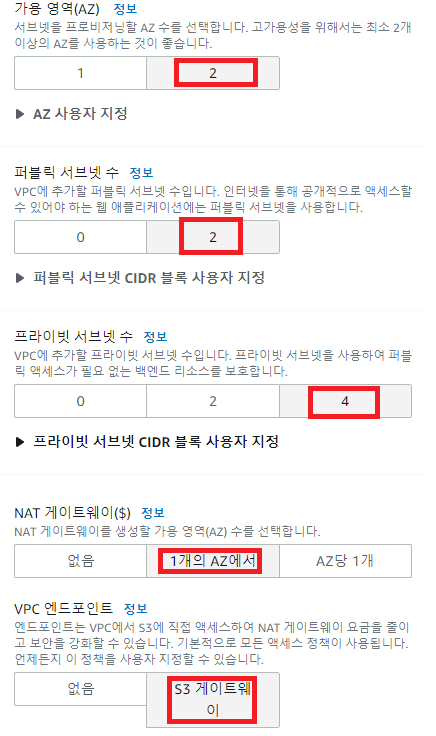
**장수민**

1 . VPC 구축

1. VPC 생성







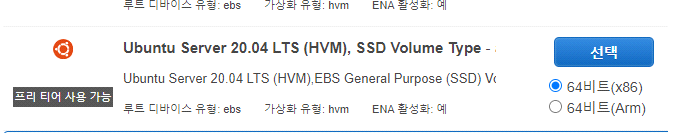
생성 확인

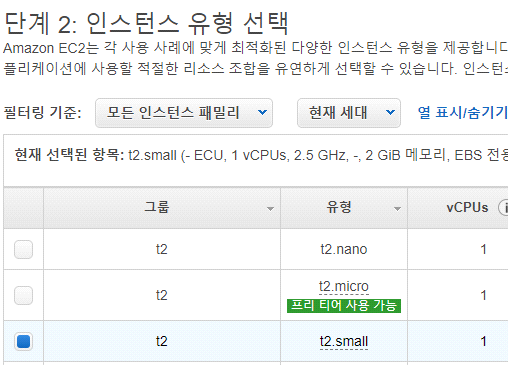


서브넷 확인 ( 각 서브넷을 어디에 쓸것인지 이름을 적어주면 좋다)

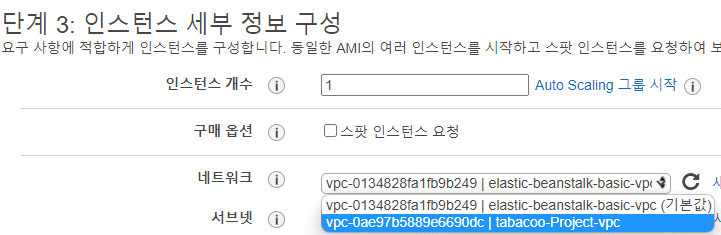


1. EC2 인스턴스 생성
2. 매뉴얼에는 인스턴스 유형을 t2.small로 하였지만 실제 프로젝트에서 돌고있는 젠킨스용 ec2는 t2.midium으로 설정했습니다. 후에 있을 안드로이드 CICD를 하기위해서는 t2.small유형의 용량이 부족하기 때문입니다

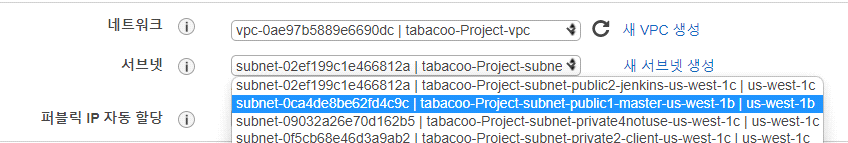




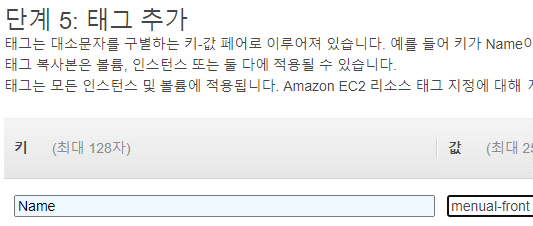
네트워크에 만들었던 vpc 선택

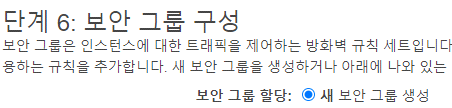


서브넷은 지금 만들고자하는 ec2의 용도에 맞게 public이면 public 서브넷 private이면 private 서브넷을 선택하면 된다

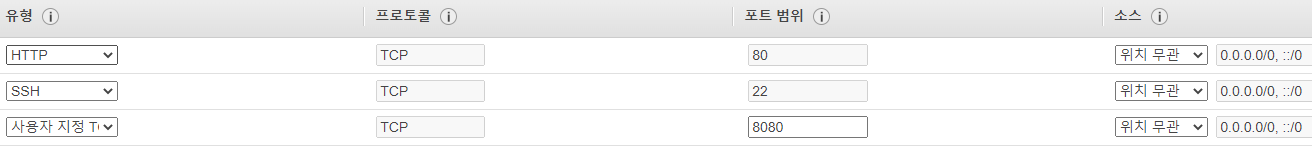


퍼블릭 ip 또한 만들고자하는 ec2의 용도에 맞게 public이면 활성화 private이면 비활성화를 선택하면 된다

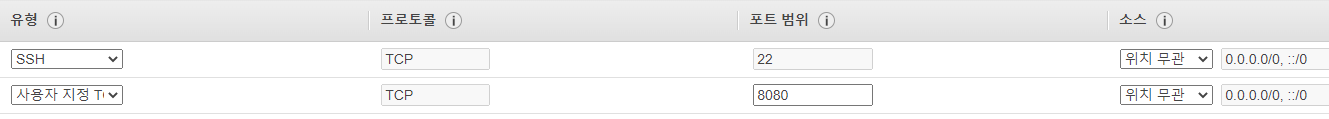




보안그룹 public 용



보안그룹 private 용



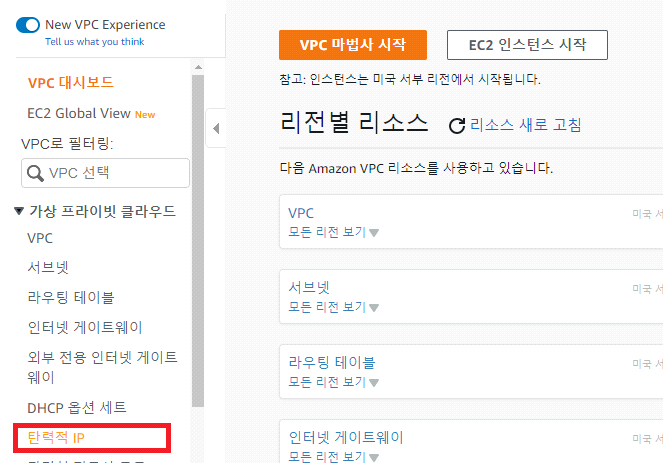
키페어 생성후 다운로드하고 인스턴스 시작

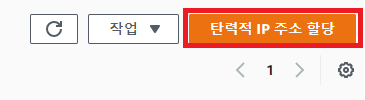




1. 탄력적 ip 생성

먼저 탄력적 IP를 생성한다 (ec2가 중단이 된 후 ip가 자동으로 바뀌게 되는이 이것을 탄력적 ip가 고정시켜준다

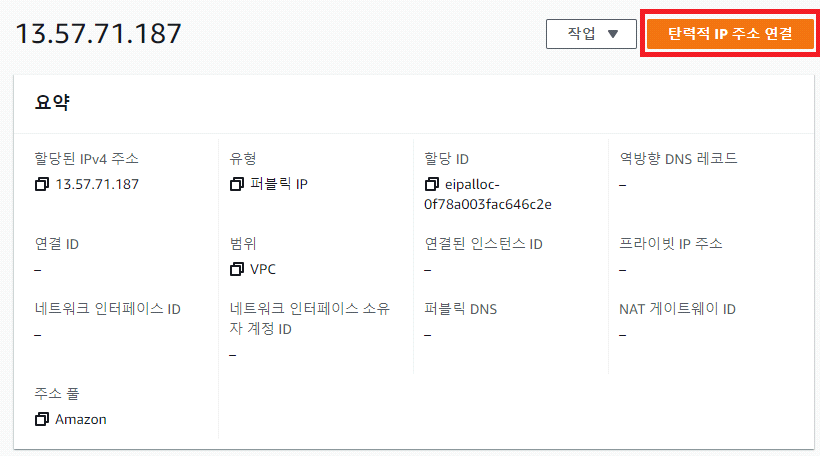




할당 설정 값은 기본값으로 생성

1. 탄력적 ip 연결

EC2에 할당된 탄력적 ip주소 연결

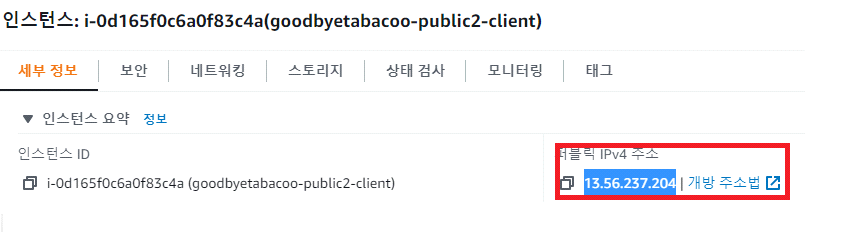


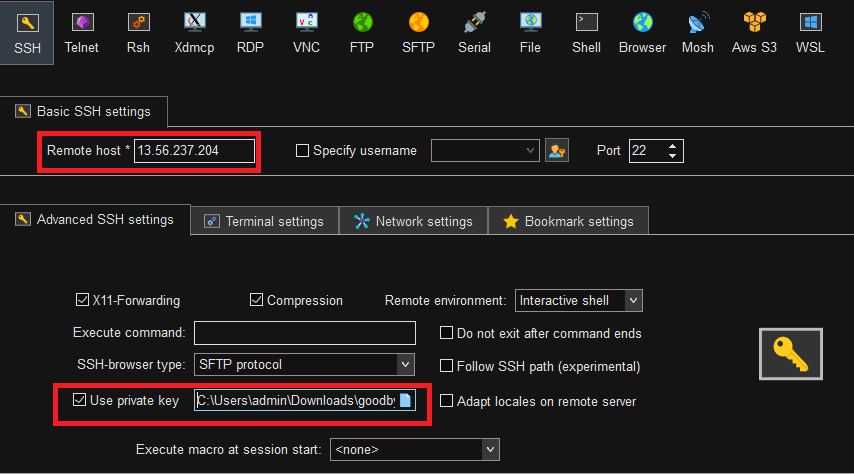
연결할 ec2 선택후 연결하면 ip가 고정이 된다



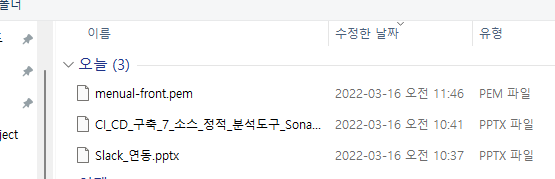
**EC2 public 서버 접속**

MobaXterm의 ssh로 접속

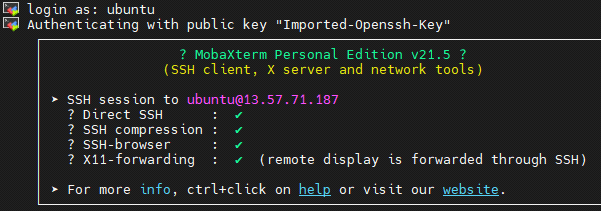
접속할 인스턴스의 퍼블릭ip 주소를 입력



인스턴스에 접속하기 위해 발급받은 키 파일(.pem) 선택



Ubuntu로 접속해준다



jenkins는 java버전 8을 쓰기 때문에 jdk버전 8을 다운받는다

먼저 apt update를 한후에



jdk버전 11을 다운받는다



패스 경로 설정

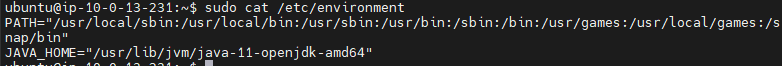
$ sudo nano /etc/environment

JAVA\_HOME="/usr/lib/jvm/java-11-openjdk-amd64"



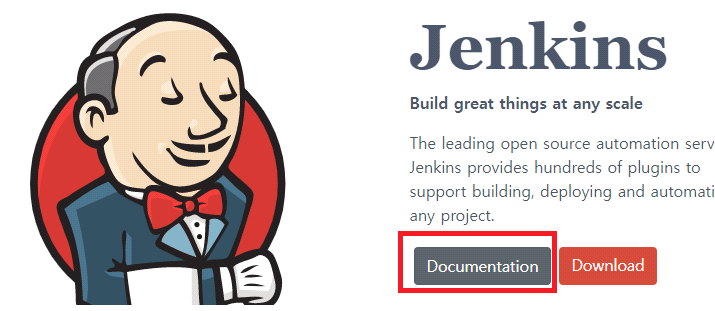
패스 경로 확인

$ sudo cat /etc/environment



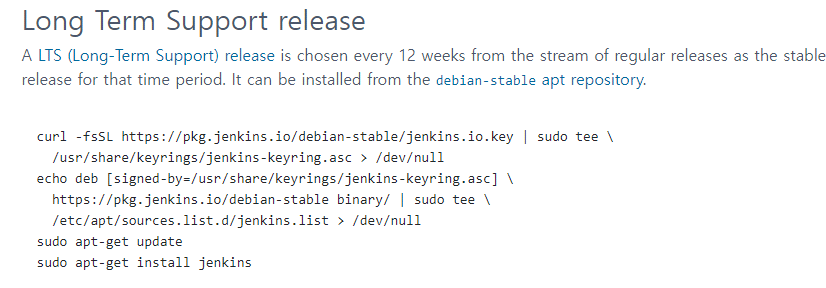
젠킨스 설치

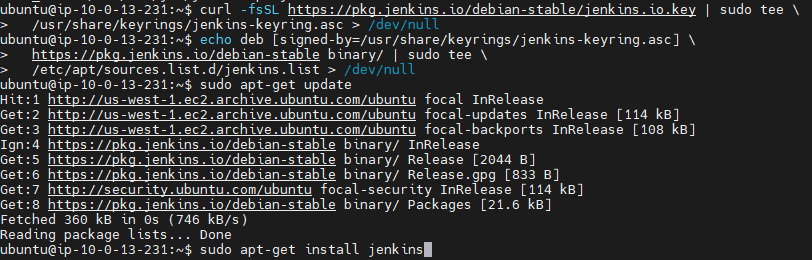
<https://www.jenkins.io/> 홈페이지 들어가서 Documentation 클릭



Installing Jenkins 에 Linux 클릭 하면 나오는 코드로 Linux에 jenkins를 설치한다

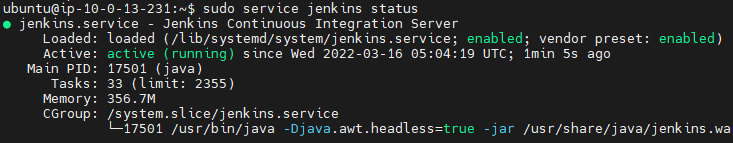






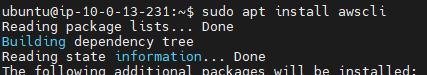
젠킨스 설치확인

sudo service jenkins status



aws명령어 사용을 위한 awscli 설치

Sudo apt install awscli

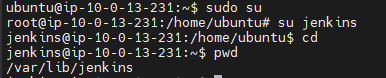


Aws 설치 확인

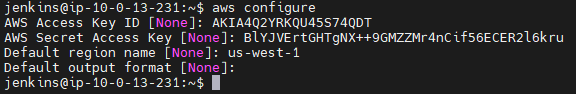
aws —version



Aws 계정 설정

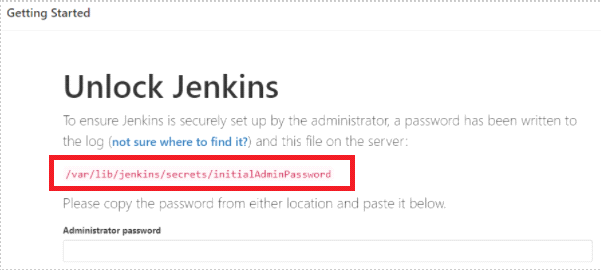


Aws Access key ID , Aws Secret Access Key, region name 은 발급 받은 걸 사용합니다



설치 후 ec2의 {ip주소}:8080을 주소창에 입력하면 아래처럼 뜹니다

아래의 빨간 박스 안에는 최초 실행시 비밀번호 경로 확인하는 코드입니다



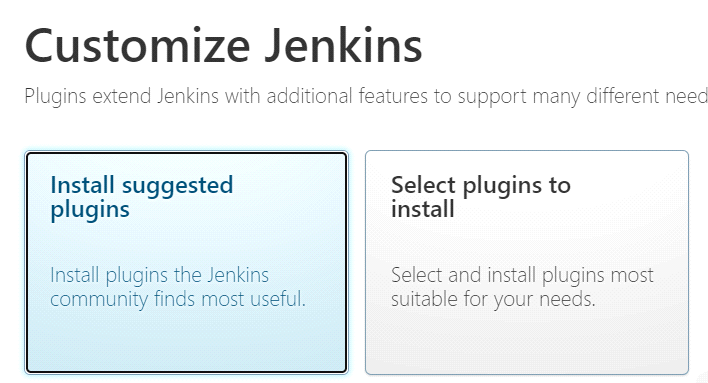
Jenkins 에서 ubuntu로 나온후

최초실행시 비밀번호 경로를 확인하기위해

Sudo cat /var/lib/jenkins/secrets/initialAdminPassword를 입력후 나오는 값을 복사하여 젠킨스 페이지의 Administrator password 칸에 넣어줍니다



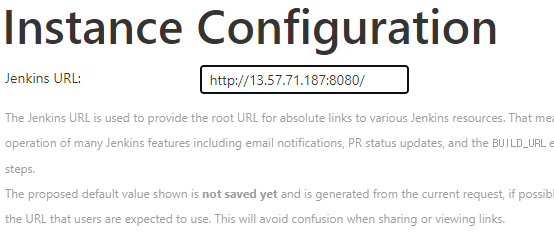
Install suggested plugins 선택

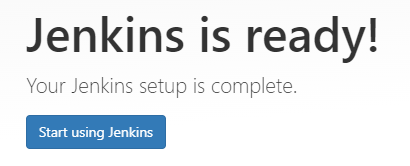


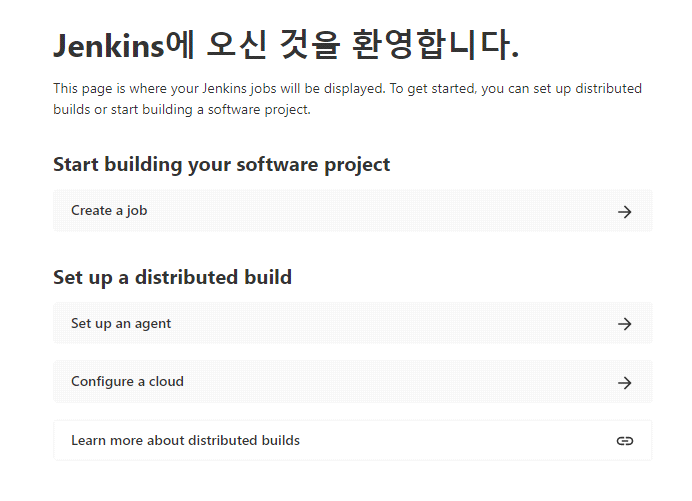
설치후 계정을 생성하는 페이지가 나오는데 작성해줍니다



젠킨스 주소 확인







젠킨스와 Git연동

EC2에 git이 설치되어있는지 먼저 확인(만약 없다면 sudo apt install git)

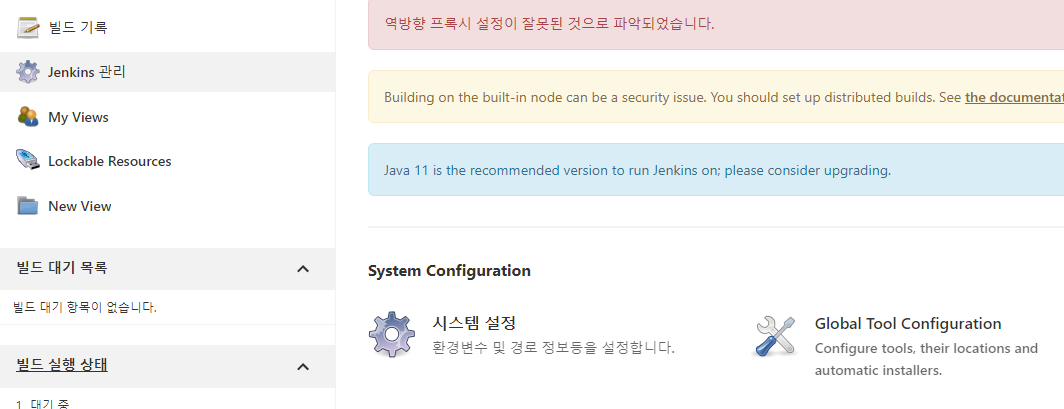
$ sudo git –version



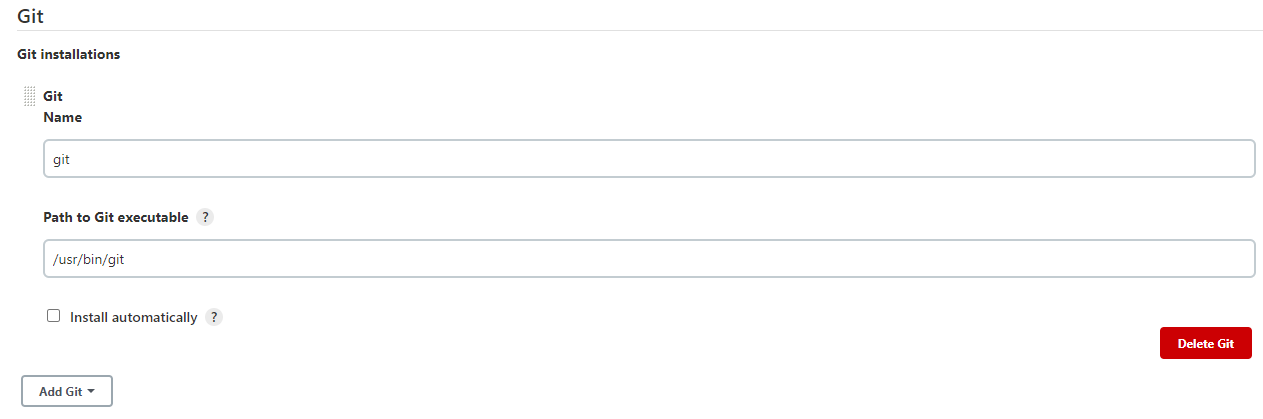
sudo which git으로 git의 설치 경로 확인



젠킨스 관리에서 Global Tool Configuration 클릭



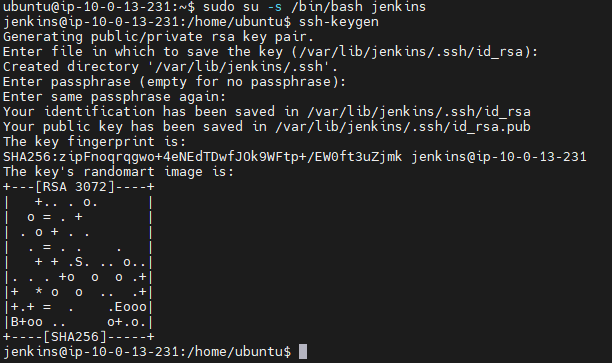
Git의 경로를 확인했던 경로로 수정 후 저장



sudo su 수퍼유저권한으로 젠킨스를 bash형태로 접속하여 keygen 생성

$ sudo su -s /bin/bash Jenkins

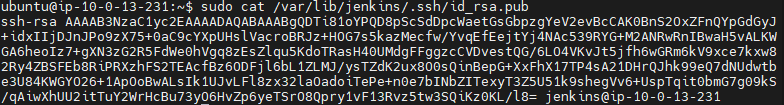
$ ssh-keygen



exit명령어로 jenkins에서 빠져나온 이후 위에서 생성한 퍼블릭 키를 cat 명령어로 조회한다.

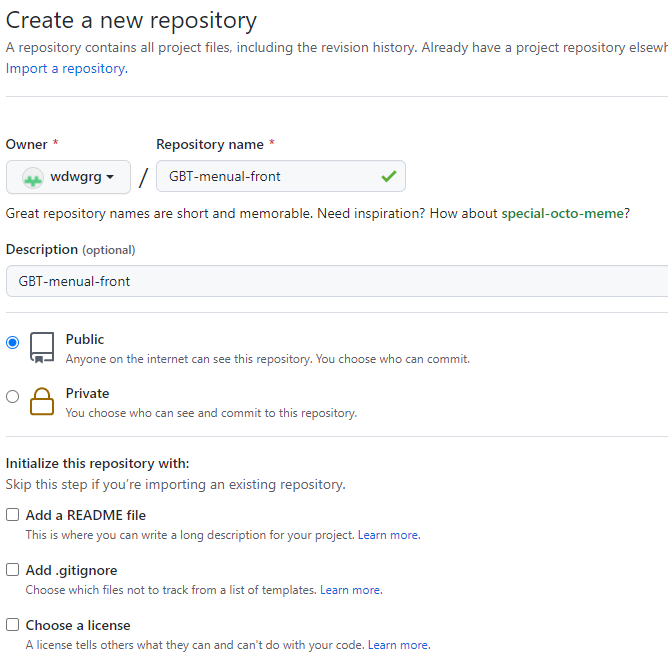
$ sudo cat /var/lib/jenkins/.ssh/id\_rsa.pub



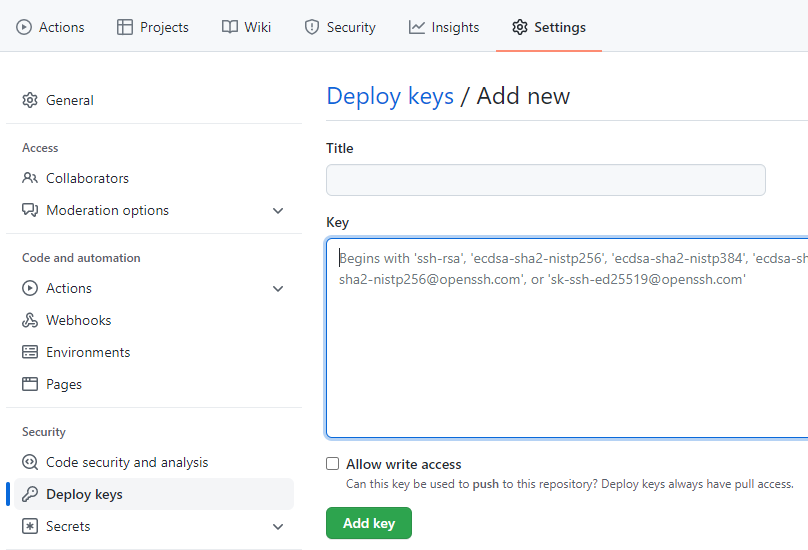


jenkins와 연동할 레포지토리 생성

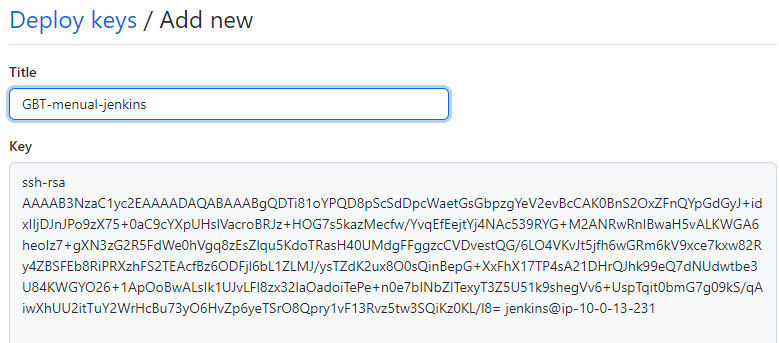
github 에서 젠킨스와 연동할 레포지토리를 생성한다



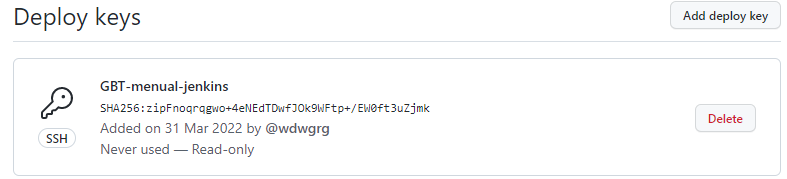
생성한 레포지토리의 설정에서 Deploy keys에서 add deploy key 선택



key 부분에 위에서 cat 명령어로 조회했던 jenkins 키 입력후 add key 버튼을 누른다

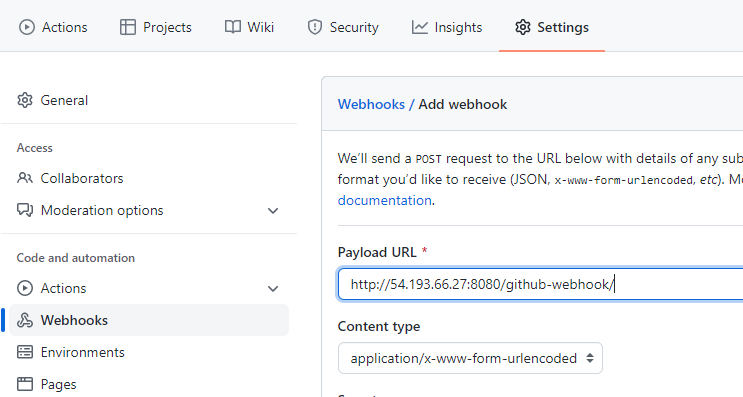


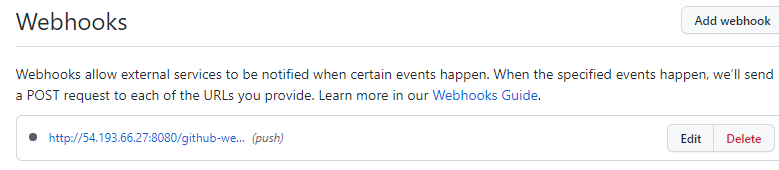
발급된 키를 확인할 수 있다



push를 했을 때 jenkins에서 인식할 수 있도록 webhook 설정을 합니다

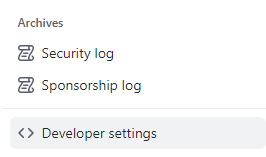
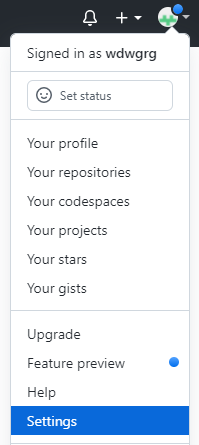
EC2 주소와 젠킨스 포트를 입력하고 github-webhook/ 을 씁니다( 맨마지막 “ / ” 반드시 넣을것)

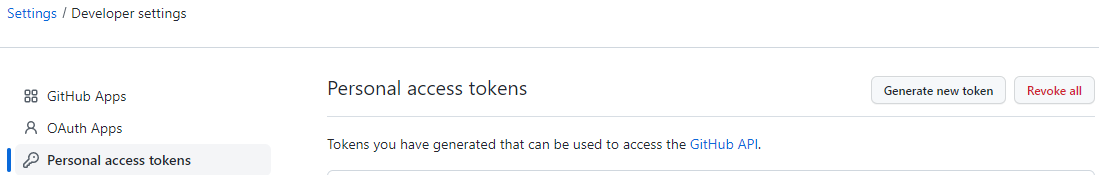




jenkins에서 github로 접근할 수 있도록 토큰을 생성합니다

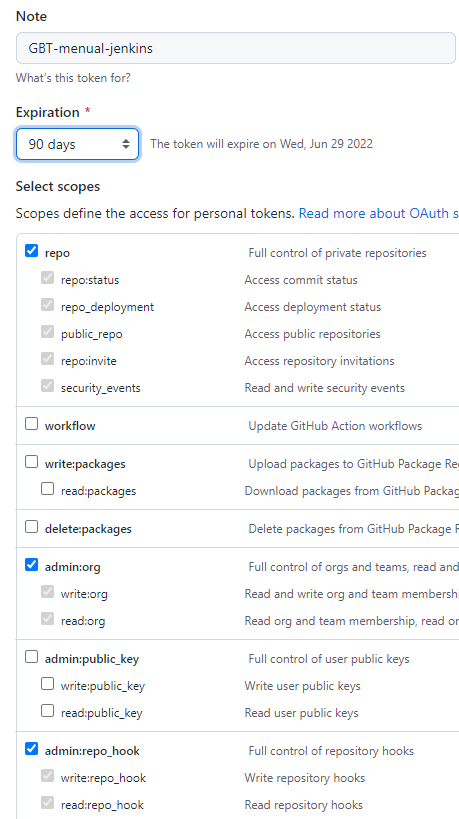
자신의 계정의 setting에 Develop setting에 personal access token 클릭





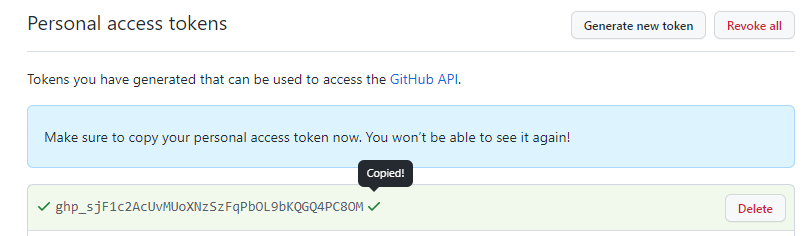
Generate new token 클릭을 한 후

토큰 이름을 설정해주고 아래의 권한을 허용해 줍니다



토큰을 만들면 토큰 키가 나오는데 토큰 키는 지금이 아니면 볼 수 없으므로 다른곳에 저장을 하는 것이 좋습니다

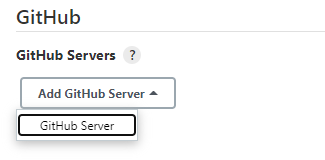
토큰 : ghp\_sjF1c2AcUvMUoXNzSzFqPbOL9bKQGQ4PC8OM



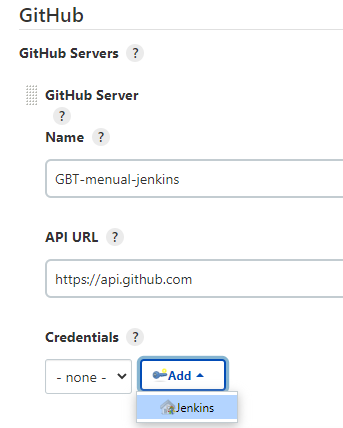
토큰을 발급 받은 후 젠킨스로 돌아와 젠킨스 설정에서 시스템 설정으로 들어갑니다



Github를 찾아서 add github를 해줍니다

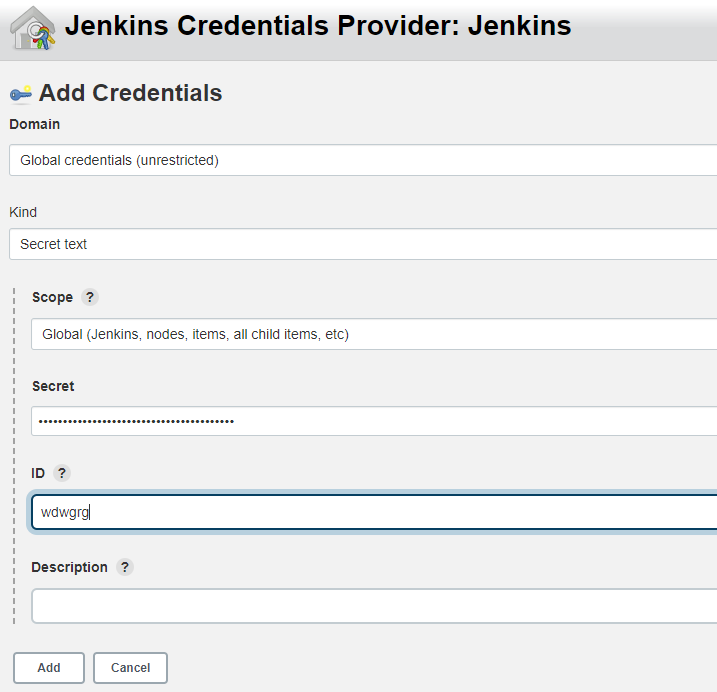


Credential을 추가하는데 Credential 항목에 Add부분에 jenkins클릭합니다



Kind 부분을 secret text로 바꾸고 Secret 부분에는 아까 발급 받은 key값을 넣습니다

ID부분에는 github Id를 적으면됩니다

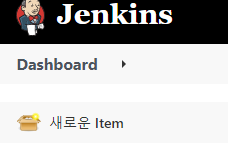


**Credentials 에서 위에서 만든 Credentials을 선택 후 test connection을 실행**

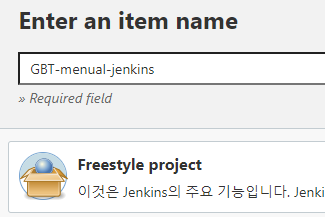
Credentials verified for user YeongSeon1, rate limit: 4999 이렇게 뜨면 성공한것이기 때문에 성공한 것을 확인하고 저장을 해줍니다



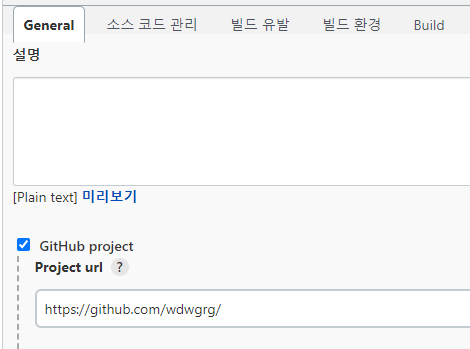
**Credentials 설정 후 새로운 Item 선택**



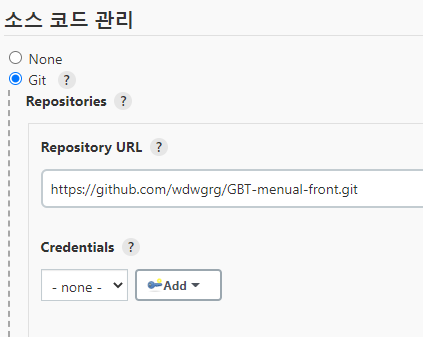
이름을 입력하고 Freestyle project 선택



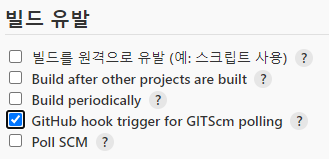
Github project를 선택후 github주소와 자신의 닉네임을 입력하고 끝에 “ / “ 를 입력합시다



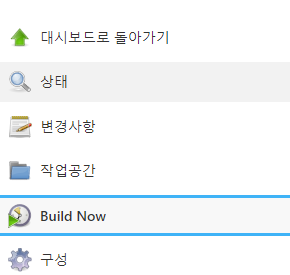
소스 코드에는 git을 선택한후 연결할 레포지토리의 주소를 입력합니다



webhook에서 이벤트가 발생하면 빌드를 할것이기 때문에 빌드 유발에서 GitHub hook 체크하고 저장을 합니다

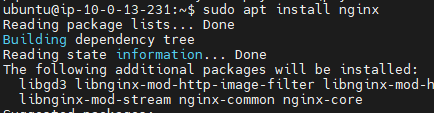


저장한후 build now 를 눌러 workspace가 만들어 지도록 합니다



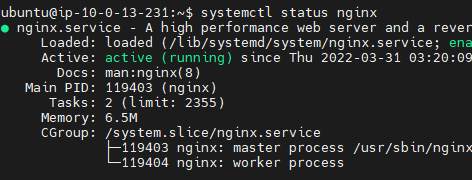
Ec2에 Nginx 설치

$ sudo apt install nginx



Nginx 설치후 상태 확인

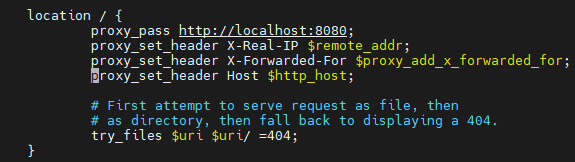
$ systemctl status nginx



경로 추가

$sudo nano /etc/nginx/sites-available/default



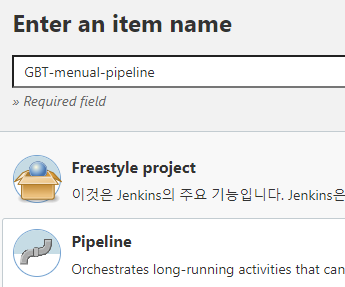


설정을 다했으면 nginx를 재시작 해줍니다

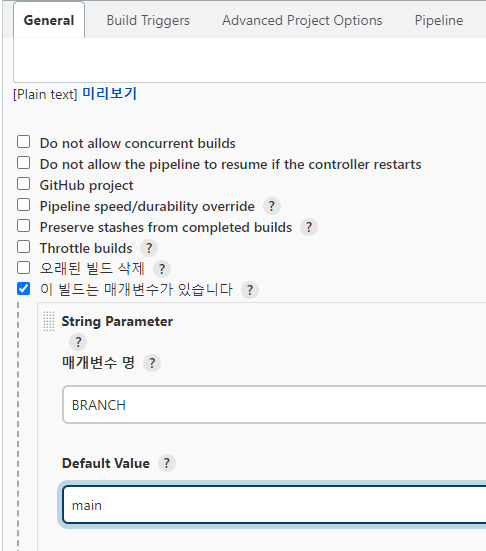
$ sudo systemctl restart nginx

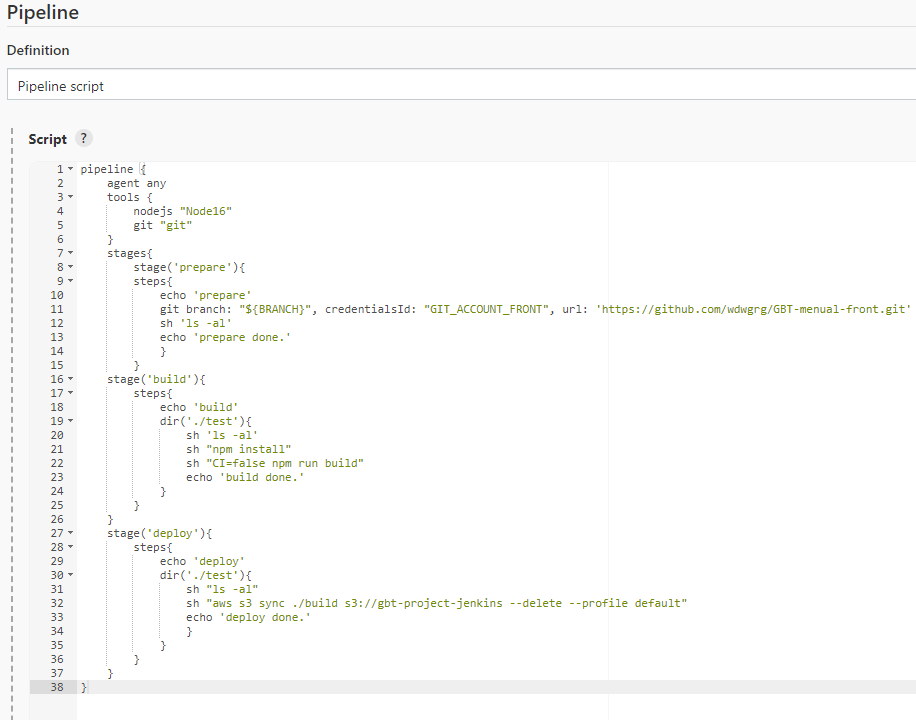


다음은 젠킨스에서 파이프 라인을 생성 합니다



매개변수를 체크하고 String Parameter를 선택한뒤 매개변수 명과 Default Value에는 branch이름을 적어줍니다





S3를 생성 합니다

먼저 버킷 엑세스 차단 설정에 차단을 풀어줍니다



정적 웹 사이트 호스팅을 활성해주고

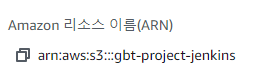
인덱스 문서와 오류 문서를 둘다 index.html로 설정해줍니다



S3를 생성한후 권한의 버킷 정책에 이렇게 입력합니다

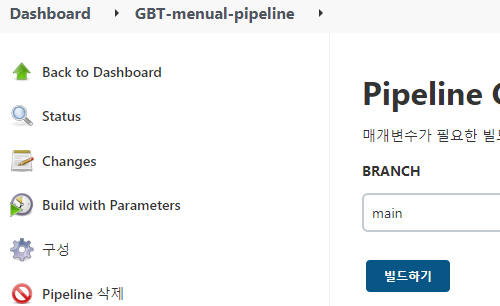


S3 속성에 Amazon 리소스 이름이 있는데 이부분을 복사한후 pipeline에 수정합니다

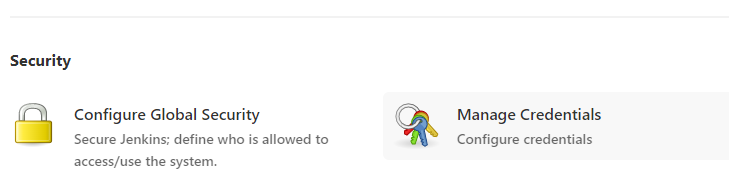


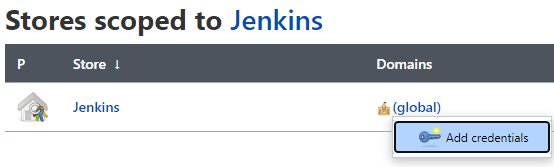


파이프 라인 실행

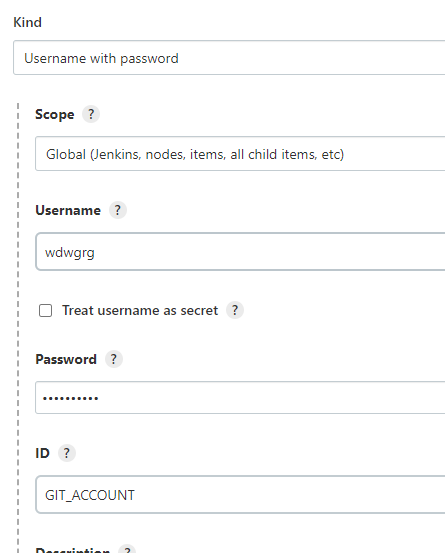


실행후 Jenkins 관리에 Manage Credentials클릭

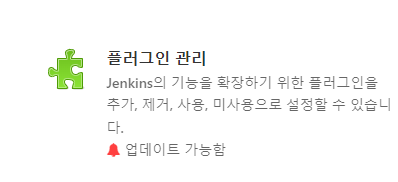


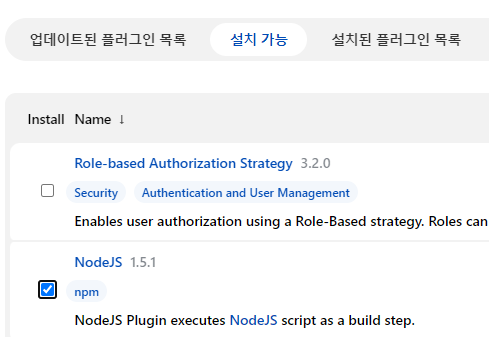


username에는 깃허브 아이디 password에는 깃허브 비밀번호, ID에는 GIT\_COUNT를 넣는다

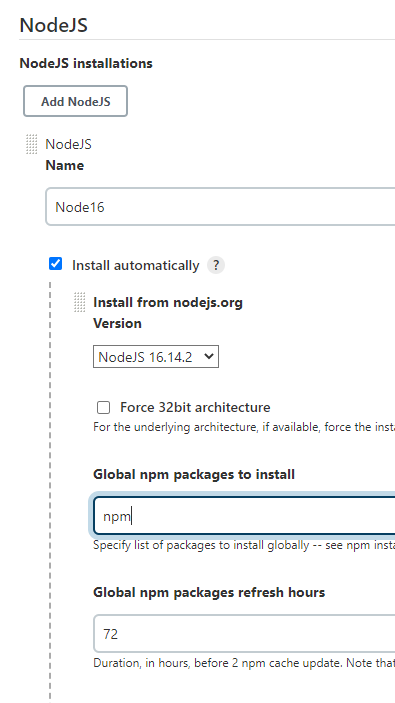


jenkins관리에 플러그 인에서 node검색후 설치





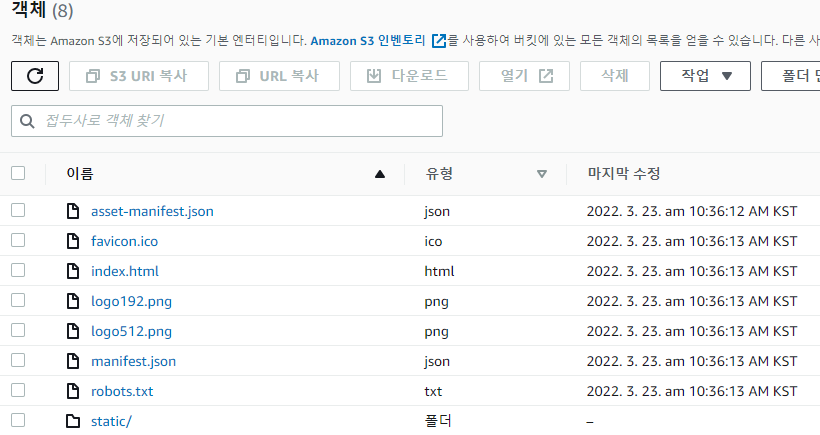
설치후 Global Tool Configuration에서 Add NodeJS 클릭



Name칸에 입력한 Node16 기억할것

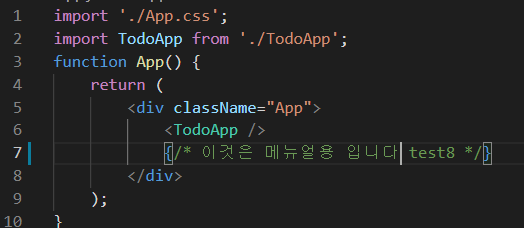
S3에 파일 추가

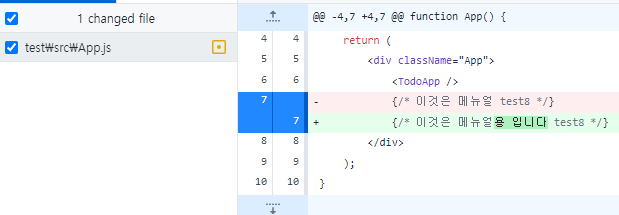
로컬의 build폴더 안에 있는 파일 및 폴더를 넣어준다

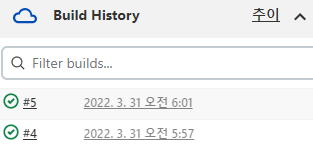


그런 다음 올린파일에 해당하는 것을 test용으로 주석을 달고 github에 commit, push를 하면

젠킨스가 github의 웹훅을 감지 하여 파이프라인으로 인해 자동빌드가되고 S3에 다시 배포하게됩니다







엘라스틱 빈스톡을 이용한 Backend 구축