

# Secreto 배포 가이드스

## 1) AWS Lightsail 인스턴스 생성

- Ubuntu 20.04.6 LTS 80USD

## 2) MobaXterm 다운로드

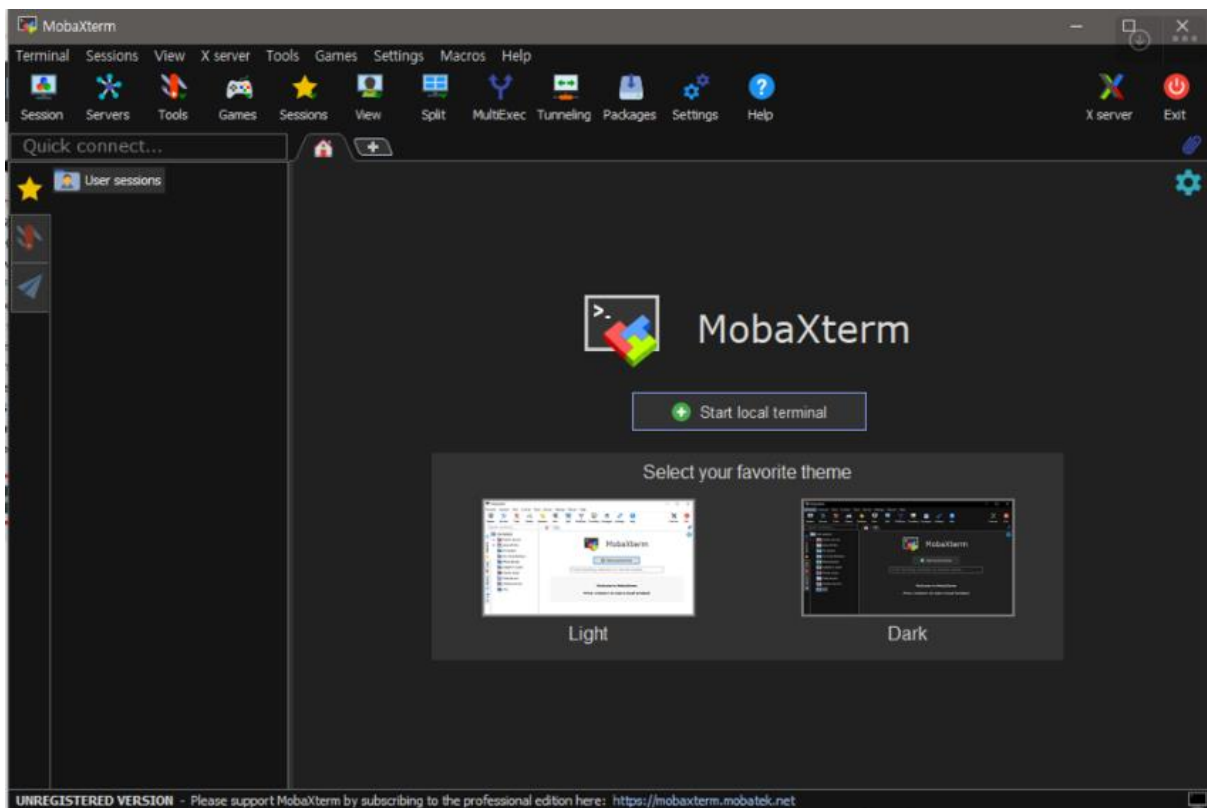
- [MobaXterm free Xserver and tabbed SSH client for Windows \(mobatek.net\)](https://mobatek.net)

【 위 링크 클릭 후 Download – Download Now 클릭 】

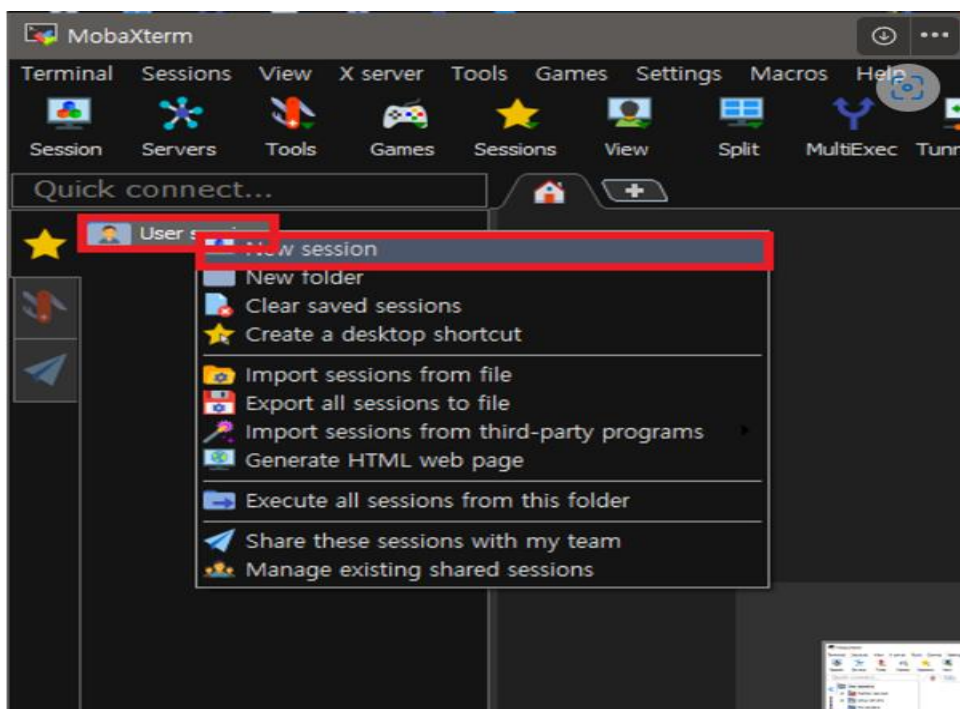
The screenshot shows the MobaXterm website. The navigation bar includes links for Home, Demo, Features, Download (highlighted with a red box), Plugins, Help, and Contact. There are also social media icons and a 'Customer area' button. The main content area is divided into two columns: 'Home Edition' and 'Professional Edition'.

Home Edition	Professional Edition
<b>Free</b>	<b>\$69 / 49€ per user*</b>
<ul style="list-style-type: none"><li>Full <b>X server</b> and <b>SSH</b> support</li><li>Remote desktop (RDP, VNC, Xdmcp)</li><li>Remote terminal (SSH, telnet, rlogin, Mosh)</li><li>X11-Forwarding</li><li>Automatic SFTP browser</li><li>Master password protection</li><li>Plugins support</li><li>Portable and installer versions</li><li>Full documentation</li><li>Max. <b>12</b> sessions</li><li>Max. <b>2</b> SSH tunnels</li><li>Max. <b>4</b> macros</li><li>Max. <b>360</b> seconds for Tftp, Nfs and Cron</li></ul>	<ul style="list-style-type: none"><li>* Excluding tax. Volume discounts <a href="#">available</a></li><li><b>Every feature from Home Edition +</b></li><li>Customize your startup message and logo</li><li>Modify your profile script</li><li>Remove unwanted games, screensaver or tools</li><li>Unlimited number of sessions</li><li>Unlimited number of tunnels and macros</li><li>Unlimited run time for network daemons</li><li>Enhanced security settings</li><li>12-months updates included</li><li>Deployment inside company</li><li>Lifetime right to use</li></ul>
<a href="#">Download now</a> (button highlighted with a red box)	<a href="#">Subscribe online / Get a quote</a>

【 원하는 테마 선택 】

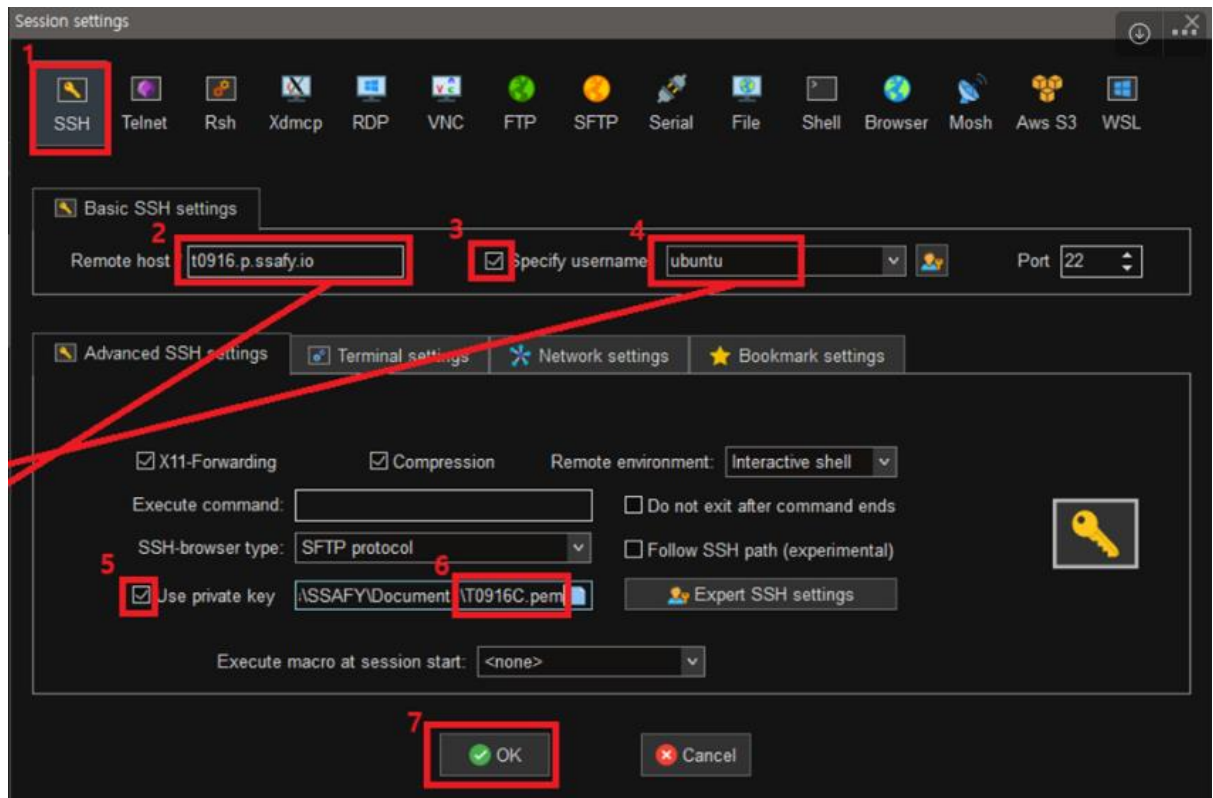


【 SSH 프로토콜로 접속할 서버 정보 입력 】

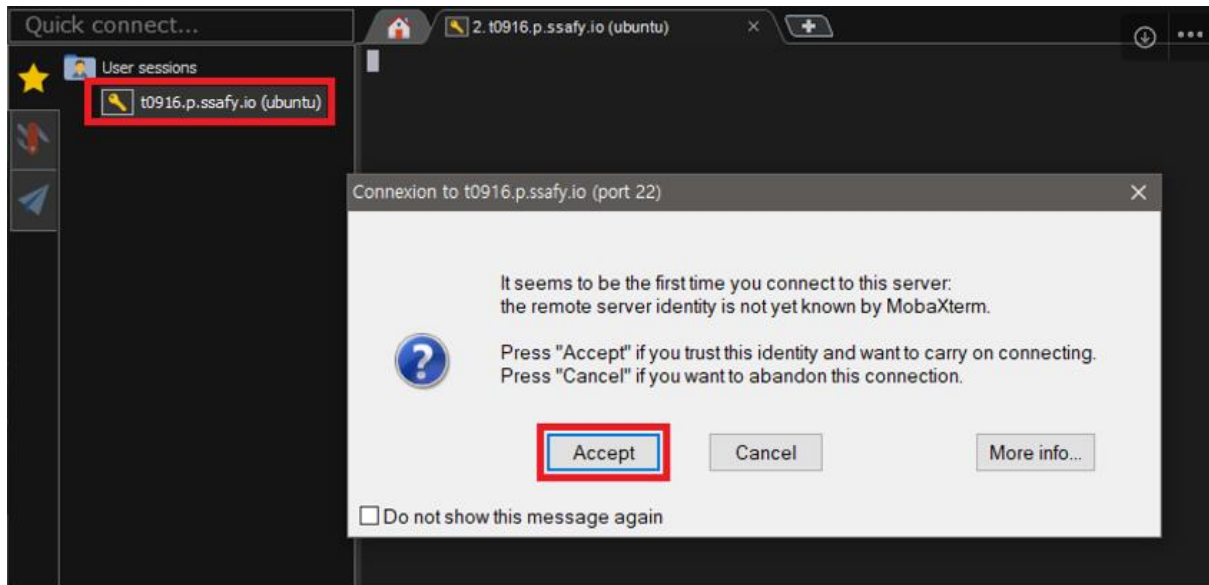


### 【 SSH 접속정보 입력 】

1. SSH 타입 선택
2. 서버 주소 입력
3. Specify username 체크
4. 계정명 입력 (Ubuntu)
5. Use private key 체크
6. 디렉토리에서 \*.pem 키 선택
7. OK 클릭



【 알림창이 뜨면 Accept 클릭 】



### 3) Ubuntu 서버 세팅

【 우분투 서버의 시간을 한국 표준시로 변경 (UTC+9) 】

```
sudo timedatectl set-timezone Asia/Seoul
```

【 미리 서버를 카카오 서버로 변경 】

```
sudo sed -i 's/ap-northeast-2.ec2.archive.ubuntu.com/mirror.kakao.com/g' /etc/apt/sources.list
```

【 패키지 목록 업데이트 및 패키지 업데이트 】

```
sudo apt-get -y update && sudo apt-get -y upgrade
```

【 스왑 영역 4GB 할당 】

```
sudo fallocate -l 4G /swapfile
```

【 swapfile 권한 수정 】

```
sudo chmod 600 /swapfile
```

【 swapfile 생성 】

```
sudo mkswap /swapfile
```

【 swapfile 활성화 】

```
sudo swapon /swapfile
```

【 시스템이 재부팅 되어도 swap 유지할 수 있도록 설정 】

```
sudo echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
```

【 swap 영역이 할당 되었는지 확인 】

```
free -h
```

## 4) Docker 세팅

【 Docker 설치 전 필요한 패키지 설치 】

```
sudo apt-get -y install apt-transport-https ca-certificates curl gnupg-agent software-properties-common
```

【 Docker에 대한 GPG Key 인증 진행 】

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

【 Docker 레포지토리 등록(AMD64 계열) 】

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu  
$(lsb_release -cs) stable"
```

【 패키지 리스트 갱신 】

```
sudo apt-get -y update
```

【 Docker 패키지 설치 】

```
sudo apt-get -y install docker-ce docker-ce-cli containerd.io
```

【 Docker 일반 유저에게 권한 부여 】

```
sudo usermod -aG docker Ubuntu
```

【 Docker 서비스 재시작 】

```
sudo service docker restart
```

## 5) NginX 세팅

【 NginX 설치 (Host OS) 】

```
sudo apt-get -y install nginx
```

【 Nginx 삭제 명령어 】

```
sudo apt-get -y remove --purge nginx nginx-full nginx-common
```

### 5-1) NginX SSL 설정 (CertBot)

【 CertBot 다운로드 】

```
sudo snap install --classic certbot
```

【 NginX 다운로드에 필요한 플러그인 설치 】

```
sudo apt-get install python3-certbot-nginx
```

【 SSL 인증서 발급 】

```
sudo certbot --nginx -d i10a805.p.ssafy.io
```

## 5-2) NginX로 vue.js 프로젝트 배포 설정

- UBUNTU에서 /etc/nginx/sites-available/default 파일의 내용을 아래와 같이 설정한다.

【 Default server configuration 설정 】

```
# Default server configuration
#
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    # SSL configuration
    #
    # listen 443 ssl default_server;
    # listen [::]:443 ssl default_server;
    #
    # Note: You should disable gzip for SSL traffic.
    # See: https://bugs.debian.org/773332
    #
    # Read up on ssl_ciphers to ensure a secure configuration.
    # See: https://bugs.debian.org/765782
    #
    # Self signed certs generated by the ssl-cert package
    # Don't use them in a production server!
    #
    # include snippets/snakeoil.conf;

    root /var/www/html;

    # Add index.php to the list if you are using PHP
    index index.html index.htm;

    server_name _;

    location / {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        try_files $uri $uri/ /index.html;
    }

    # pass PHP scripts to FastCGI server
    #
    #location ~ \.php$ {
    #    include snippets/fastcgi-php.conf;
    #
    #    # With php-fpm (or other unix sockets):
    #    fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
    #    # With php-cgi (or other tcp sockets):
    #    fastcgi_pass 127.0.0.1:9000;
    #}

    # deny access to .htaccess files, if Apache's document root
    # concurs with nginx's one
    #
    #location ~ /\.ht {
    #    deny all;
    #}
}
```



## 【 Virtual Host configuration 설정 】

```
server {
    # SSL configuration
    #
    # listen 443 ssl default_server;
    # listen [::]:443 ssl default_server;
    #
    # Note: You should disable gzip for SSL traffic.
    # See: https://bugs.debian.org/773332
    #
    # Read up on ssl_ciphers to ensure a secure configuration.
    # See: https://bugs.debian.org/765782
    #
    # Self signed certs generated by the ssl-cert package
    # Don't use them in a production server!
    #
    # include snippets/snakeoil.conf;

    root /var/www/html/dist;

    # Add index.php to the list if you are using PHP
    index index.html index.htm;
    server_name i10a805.p.ssafy.io; # managed by Certbot


    location / {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        try_files $uri $uri/ /index.html;
    }

    #location @rewrites {
    #    rewrite ^(.+)$ /index.html last;
    #}

    # pass PHP scripts to FastCGI server
    #
    #location ~ \.php$ {
    #    include snippets/fastcgi-php.conf;
    #
    #    # With php-fpm (or other unix sockets):
    #    fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
    #    # With php-cgi (or other tcp sockets):
    #    fastcgi_pass 127.0.0.1:9000;
    #}

    # deny access to .htaccess files, if Apache's document root
    # concurs with nginx's one
    #
    #location ~ /\.ht {
    #    deny all;
    #}


    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/i10a805.p.ssafy.io/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/i10a805.p.ssafy.io/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}
```

## 【 nginx 서비스 재시작 】

**sudo service nginx restart**

## 6) MariaDB 세팅

【 MariaDB 이미지 받기 】

```
sudo docker pull mariadb:latest
```

【 MariaDB 컨테이너 실행 】

```
docker run -d -p 3306:3306 -e MYSQL_ROOT_PASSWORD=비밀번호 -v /var/lib/mysql:/var/lib/mysql --name mariadb mariadb
```

【 MariaDB OS 재시작 후 자동실행 설정 】

```
sudo vim /etc/systemd/system/docker-mariadb.service
```

- docker-mariadb.service 등록

[Unit]

Description=docker-mariadb

Wants=docker.service

After=docker.service

[Service]

RemainAfterExit=yes

ExecStart=/usr/bin/docker start mariadb

ExecStop=/usr/bin/docker stop mariadb

[Install]

WantedBy=multi-user.target

【 docker 서비스 활성화 】

**sudo systemctl enable docker**

【 docker 서비스 시작 】

**sudo systemctl start docker**

【 docker-mariadb 서비스 활성화 】

**sudo systemctl enable docker-mariadb.service**

【 docker-mariadb 서비스 시작 】

**sudo systemctl start docker-mariadb.service**

【 데이터베이스 접속 】

**sudo mysql -u MariaDB계정이름 -p**

【 데이터베이스 생성 】

**create database 데이터베이스이름;**

【 생성한 데이터베이스 사용 】

**use 데이터베이스이름;**

【 MariaDB 사용자 목록 조회 】

**use mysql;**

**select host, user, password from user;**

【 사용자 계정 생성 】

```
create user '계정이름'@'%' identified by '비밀번호';
```

【 사용자 계정 삭제 】

```
drop user 계정이름@아이피주소;
```

【 사용자에게 데이터베이스 사용 권한 부여 】

```
grant all privileges on 데이터베이스이름.* to '계정이름'@'%';
```

【 변경한 환경 설정 반영 】

```
flush privileges;
```

## 7) Jenkins 세팅

【 Jenkins 이미지 받기 】

```
docker pull jenkins/jenkins:jdk17
```

【 Jenkins 컨테이너 실행 】

```
docker run -d --env JENKINS_OPTS=--httpPort=8081 -v /etc/localtime:/etc/localtime:ro -e  
TZ=Asia/Seoul -p 8081:8081 -v /jenkins:/var/jenkins_home -v  
/var/run/docker.sock:/var/run/docker.sock -v /usr/local/bin/docker-  
compose:/usr/local/bin/docker-compose --name jenkins -u root jenkins/jenkins:jdk17
```

【 Jenkins OS 재시작 후 자동실행 설정 】

```
sudo vim /etc/systemd/system/docker-jenkins.service
```

**[Unit]**

**Description=docker-jenkins**

**Wants=docker.service**

**After=docker.service**

**[Service]**

**RemainAfterExit=yes**

**ExecStart=/usr/bin/docker start jenkins**

**ExecStop=/usr/bin/docker stop jenkins**

**[Install]**

**WantedBy=multi-user.target**

【 docker 서비스 활성화 】

**sudo systemctl enable docker**

【 docker 서비스 시작 】

**sudo systemctl start docker**

【 docker-jenkins 서비스 활성화 】

**sudo systemctl enable docker-jenkins.service**

【 docker-jenkins 서비스 시작 】

**sudo systemctl start docker-jenkins.service**

【 Jenkins 접속 】

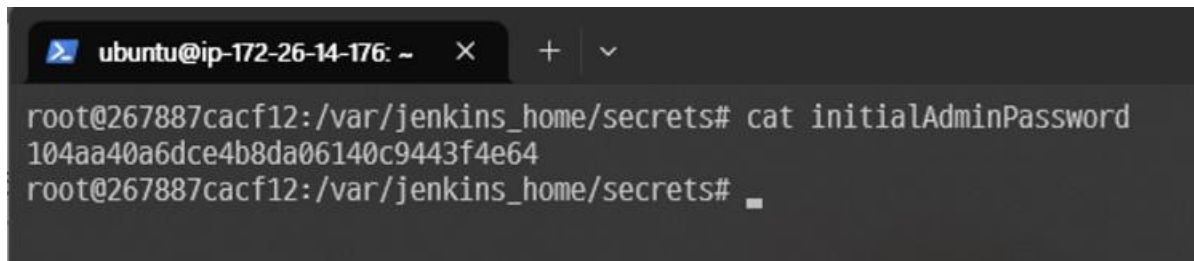
```
docker exec -it jenkins /bin/bash
```

【 해당하는 디렉토리로 이동 】

```
cd /var/jenkins_home/secrets
```

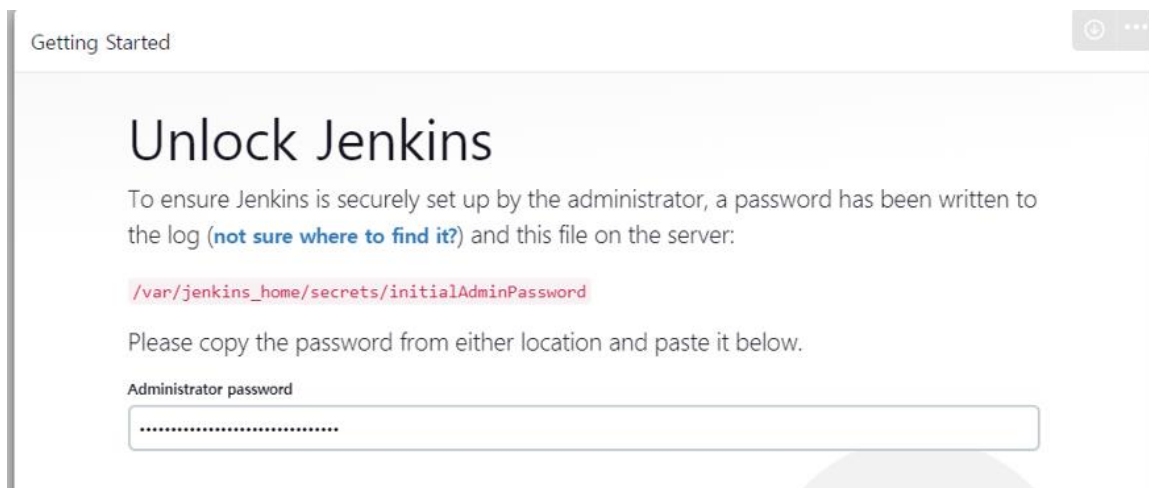
【 cat 명령어를 이용하여 초기 비밀번호를 확인 】

```
cat initialAdminPassword
```

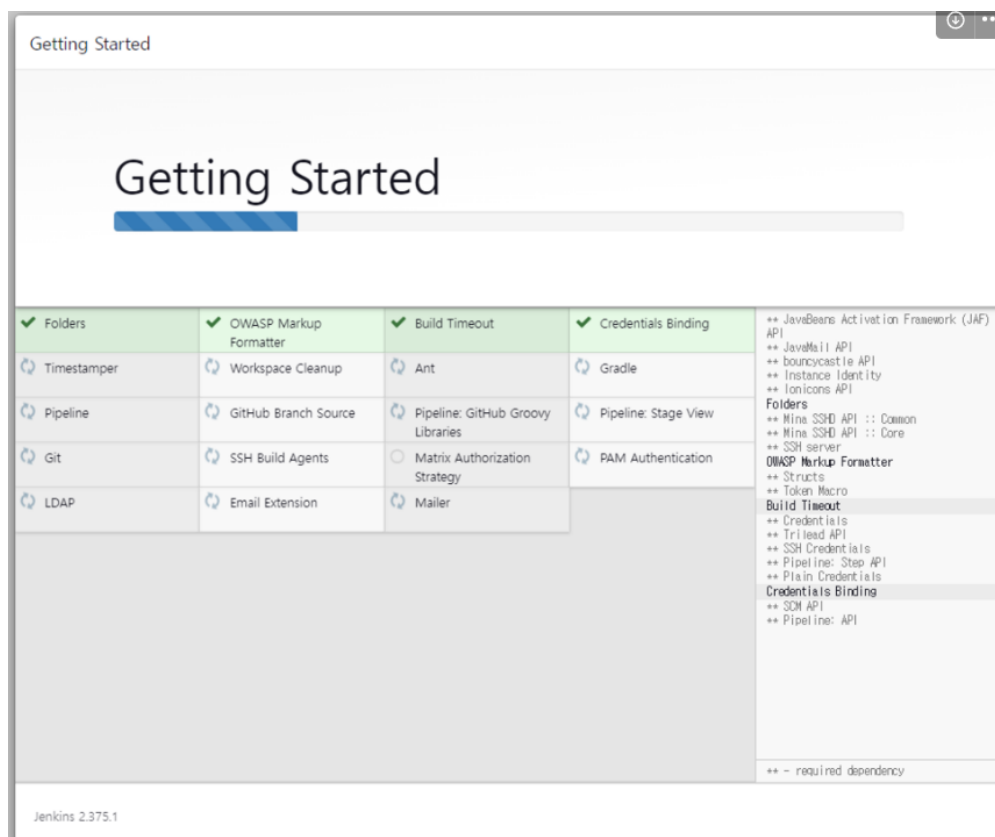
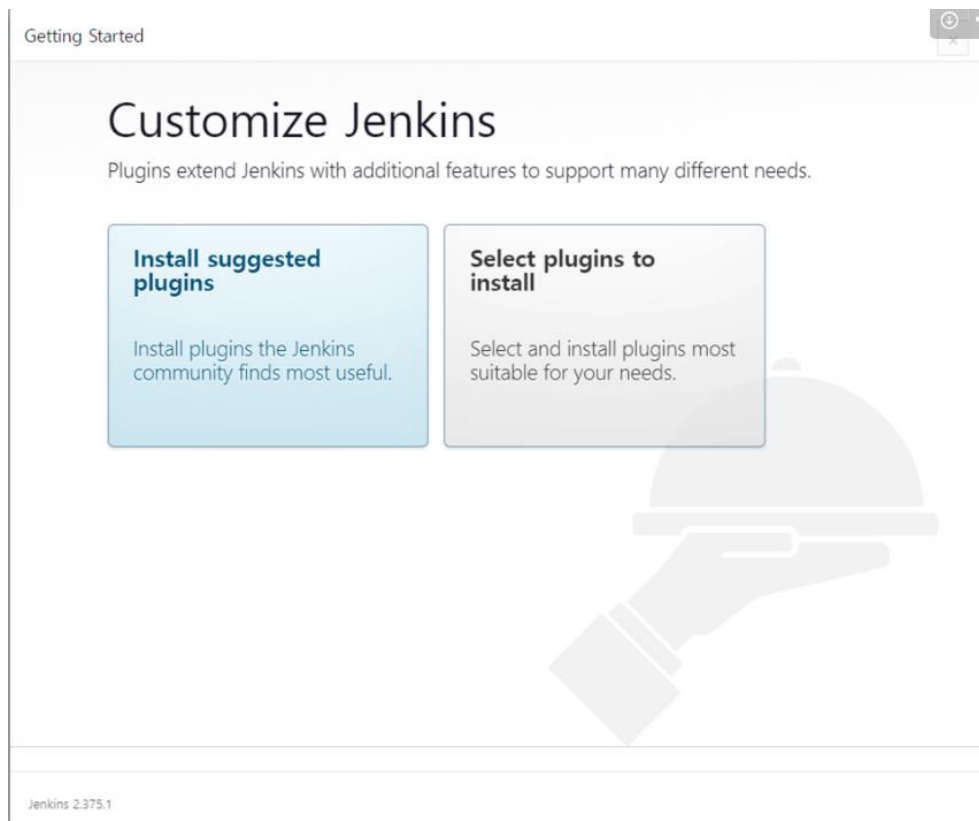


```
ubuntu@ip-172-26-14-176: ~  
root@267887cacf12:/var/jenkins_home/secrets# cat initialAdminPassword  
104aa40a6dce4b8da06140c9443f4e64  
root@267887cacf12:/var/jenkins_home/secrets#
```

【 확인한 방법으로 얻어낸 관리자 비밀번호를 입력 】



## 【 기본 플러그인 설치 】



## 【 관리자 계정 설정 】

Getting Started

# Create First Admin User

계정명

암호

암호 확인

이름

이메일 주소

Jenkins 2.375.1Skip and continue as adminSave and Continue



## 【 서버 인스턴스 확인 】

Getting Started

Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

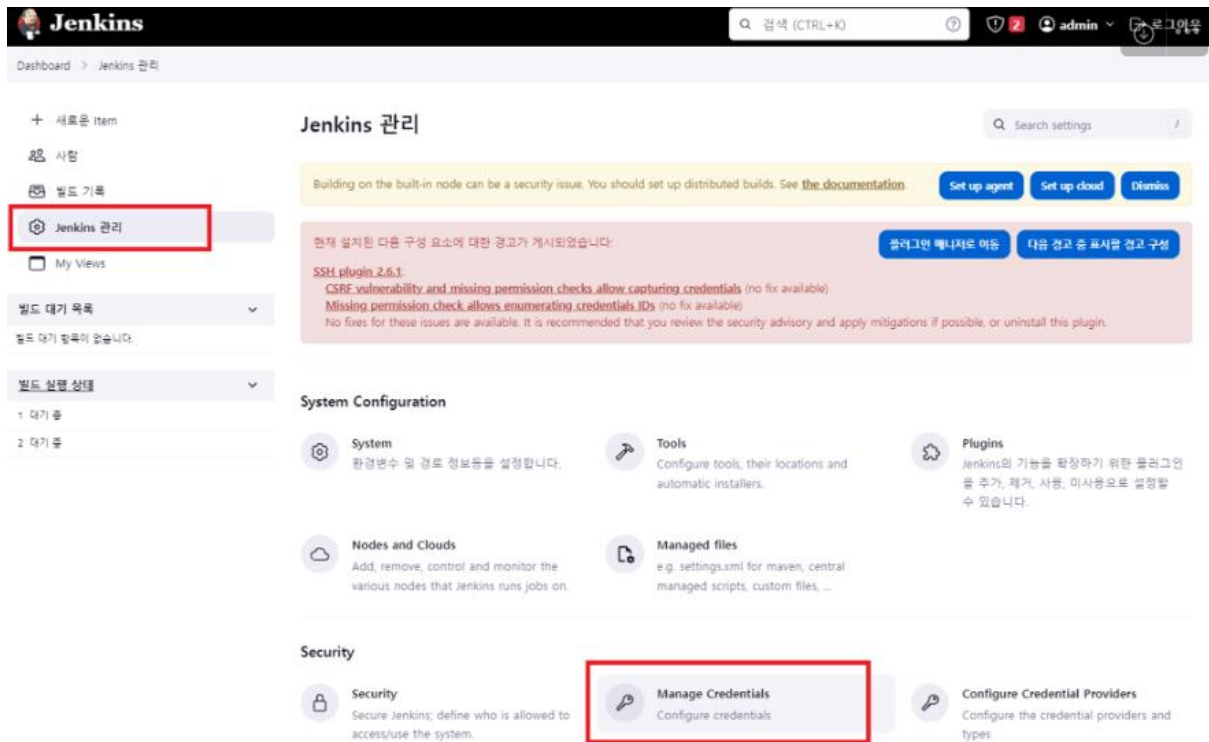
The proposed default value shown is not saved yet and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.375.1

Not now

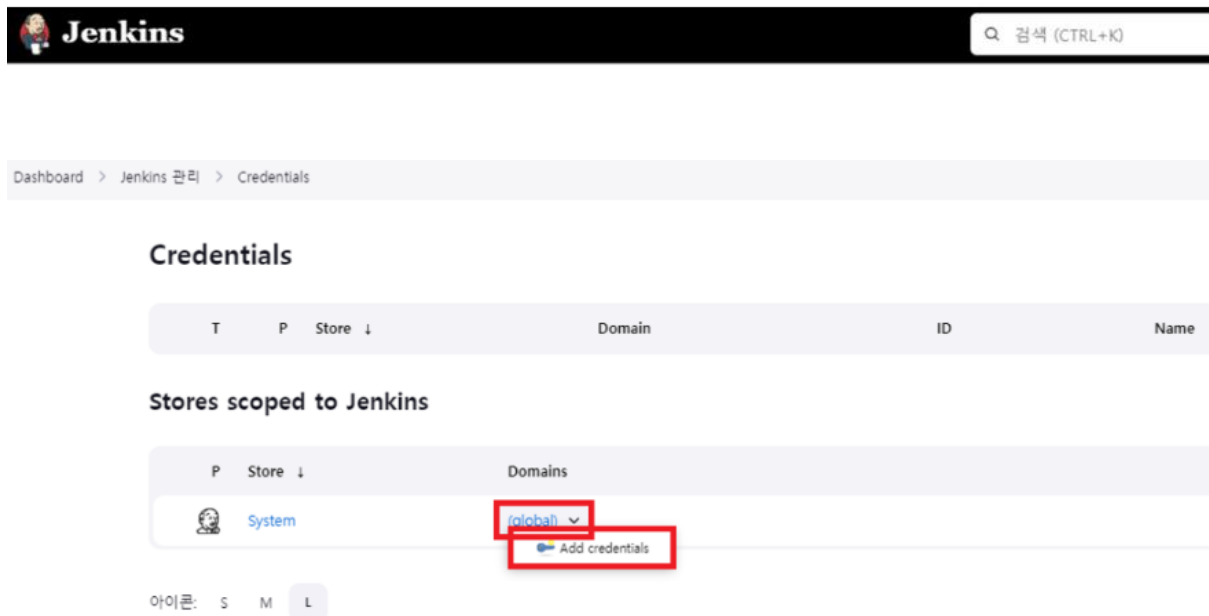
Save and Finish

## 【 GitLab Credential 등록 (Username with password) 】

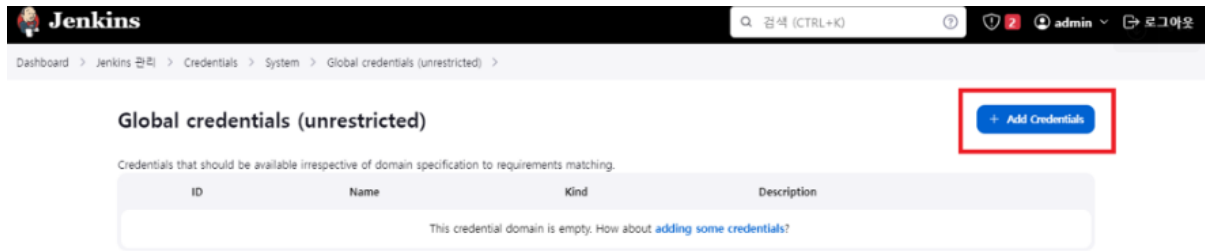


The screenshot shows the Jenkins Dashboard. On the left sidebar, 'Jenkins 관리' (Jenkins Management) is highlighted with a red box. The main content area is titled 'Jenkins 관리' and contains several sections. A yellow banner at the top mentions security issues with distributed builds. Below that, a red banner displays a security warning about the SSH plugin 2.6.1. The 'System Configuration' section includes links for 'System', 'Tools', 'Plugins', 'Nodes and Clouds', and 'Managed files'. The 'Security' section is highlighted with a red box and contains links for 'Security', 'Manage Credentials' (which is also highlighted with a red box), and 'Configure Credential Providers'.

- Stores scoped to Jenkins - Domains - (global) - Add credentials 클릭

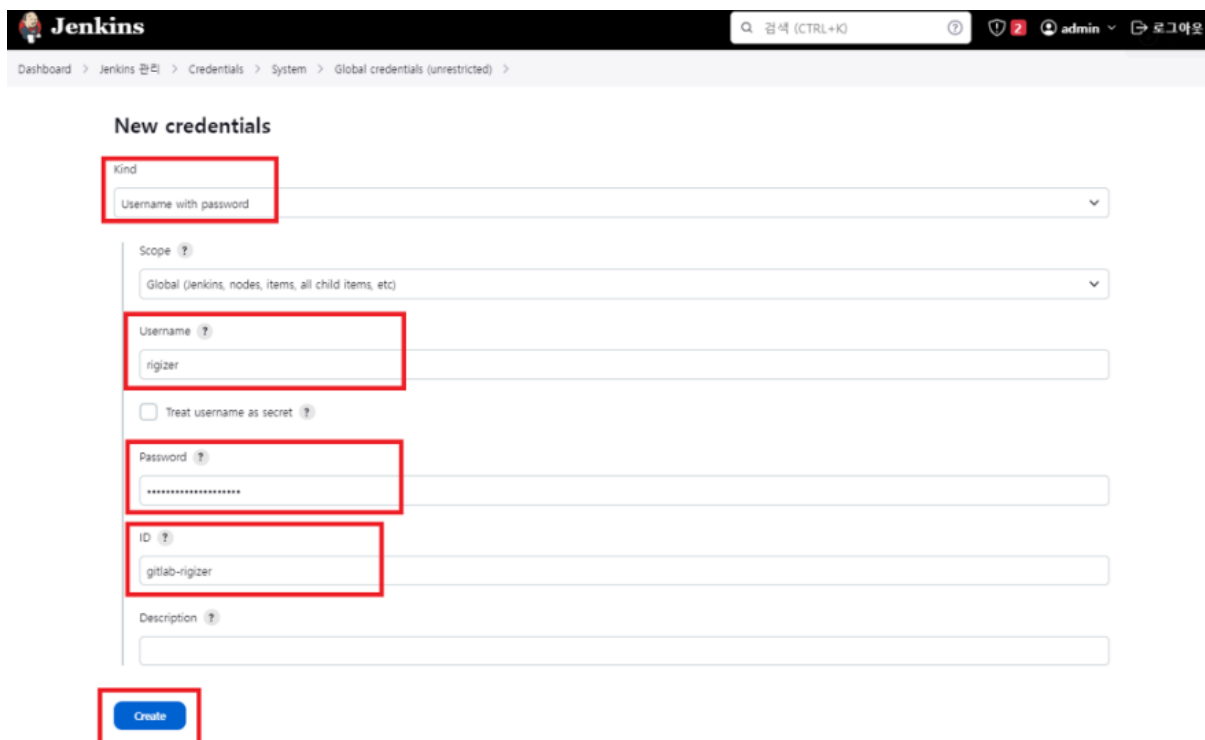


The screenshot shows the 'Credentials' page in Jenkins. The breadcrumb trail at the top reads 'Dashboard > Jenkins 관리 > Credentials'. The page title is 'Credentials'. Below the title, there is a table with columns 'T', 'P', 'Store', 'Domain', 'ID', and 'Name'. Under the 'Stores scoped to Jenkins' section, there is a table with columns 'P', 'Store', and 'Domains'. The 'System' store is listed, and the 'Domains' column shows a dropdown menu with 'global' selected and highlighted with a red box. Below the dropdown, there is a red box containing the 'Add credentials' button.

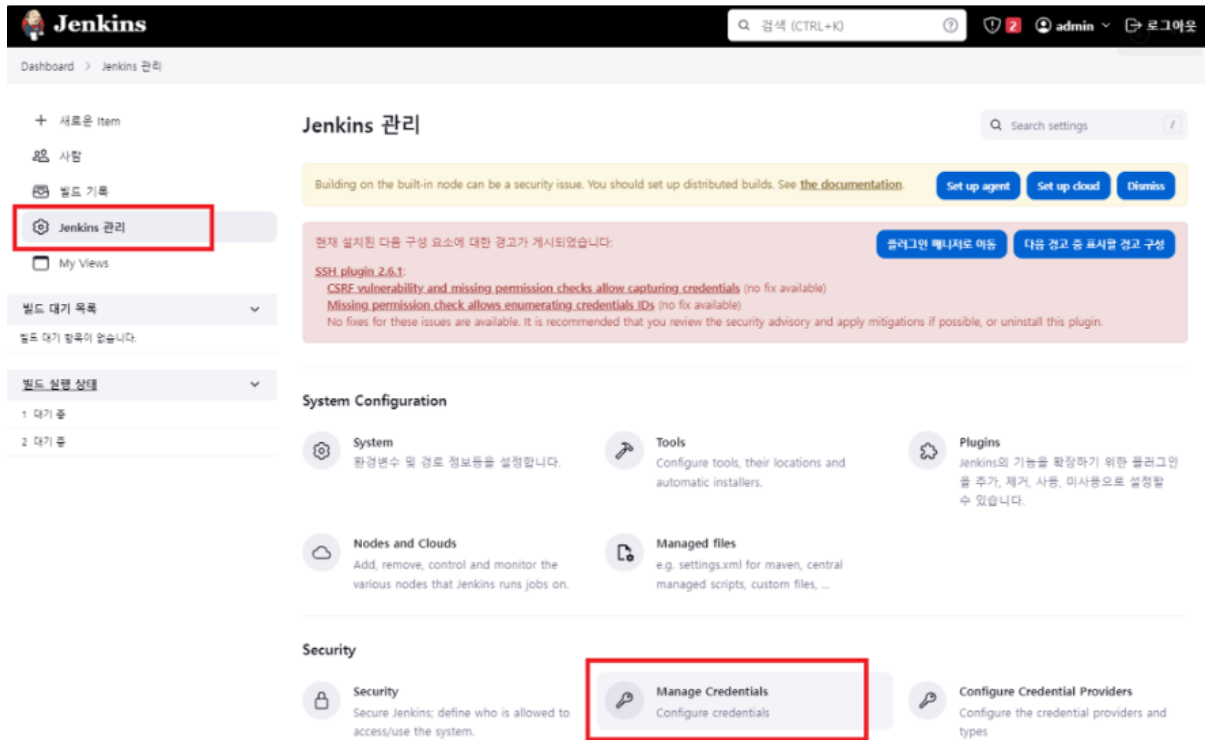


【 정보 입력 후 **Create** 클릭 】

- Kind : Username with password 선택
- Username : Gitlab 계정 아이디 입력
- Password : Gitlab 계정 비밀번호 입력 (토큰 발행시, API 토큰 입력)
- ID : Credential에 대한 별칭



## 【 GitLab Credential 등록 (API Token) 】



The screenshot shows the Jenkins Dashboard. In the left sidebar, the 'Jenkins 관리' (Jenkins Management) option is highlighted with a red box. In the main content area, under the 'System Configuration' section, the 'Manage Credentials' option is highlighted with a red box. A security warning banner is visible at the top of the main content area, mentioning SSH plugin 2.6.1 vulnerabilities.

**Jenkins 관리**

Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#). [Set up agent](#) [Set up cloud](#) [Dismiss](#)

현재 설치된 다음 구성 요소에 대한 경고가 게시되었습니다: [플러그인 해시지도 이동](#) [다음 경고 중 표시를 열고 구성](#)

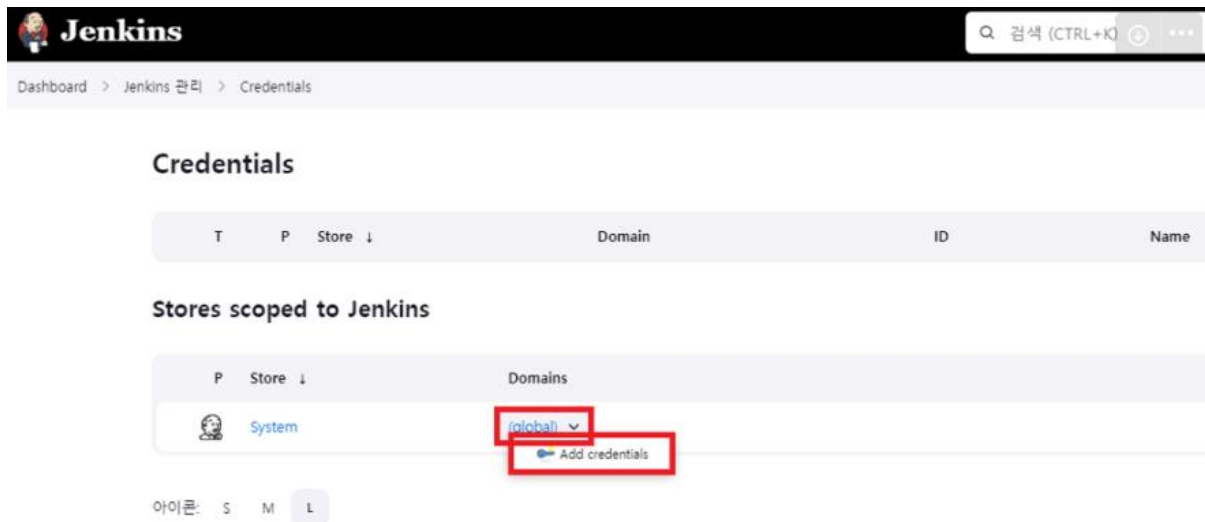
**SSH plugin 2.6.1:**  
[CSRF vulnerability and missing permission checks allow capturing credentials](#) (no fix available)  
[Missing permission check allows enumerating credentials IDs](#) (no fix available)  
No fixes for these issues are available. It is recommended that you review the security advisory and apply mitigations if possible, or uninstall this plugin.

**System Configuration**

- System**  
환경변수 및 경로 정보들을 설정합니다.
- Tools**  
Configure tools, their locations and automatic installers.
- Plugins**  
Jenkins의 기능을 확장하기 위한 플러그언을 추가, 제거, 사용, 미사용으로 설정할 수 있습니다.
- Nodes and Clouds**  
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- Managed files**  
e.g. settings.xml for maven, central managed scripts, custom files, ...

**Security**

- Security**  
Secure Jenkins; define who is allowed to access/use the system.
- Manage Credentials**  
Configure credentials
- Configure Credential Providers**  
Configure the credential providers and types



The screenshot shows the Jenkins Credentials page. The breadcrumb trail is 'Dashboard > Jenkins 관리 > Credentials'. Under the 'Stores scoped to Jenkins' section, the 'System' store is selected. The 'Domains' dropdown menu is open, and 'global' is selected, highlighted with a red box. Below the dropdown, the 'Add credentials' button is also highlighted with a red box.

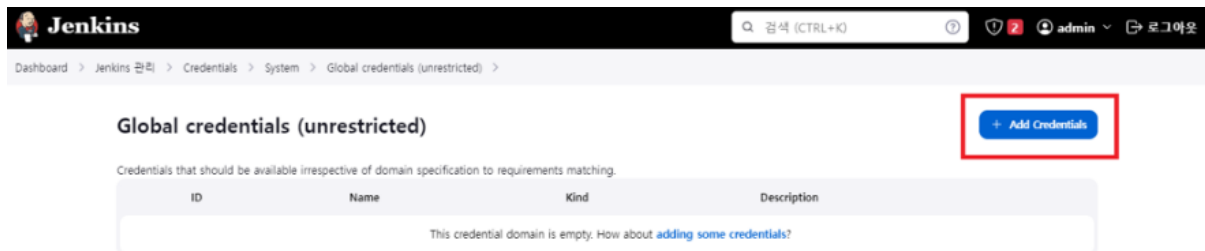
**Credentials**

T	P	Store ↓	Domain	ID	Name
---	---	---------	--------	----	------

**Stores scoped to Jenkins**

P	Store ↓	Domains
	System	<b>global</b> ↓ <a href="#">Add credentials</a>

아이콘: S M L



The screenshot shows the 'Global credentials (unrestricted)' page. The breadcrumb trail is 'Dashboard > Jenkins 관리 > Credentials > System > Global credentials (unrestricted)'. The '+ Add Credentials' button is highlighted with a red box. Below the button, there is a table with columns 'ID', 'Name', 'Kind', and 'Description'. The table is currently empty, and a message states: 'This credential domain is empty. How about [adding some credentials](#)?'.

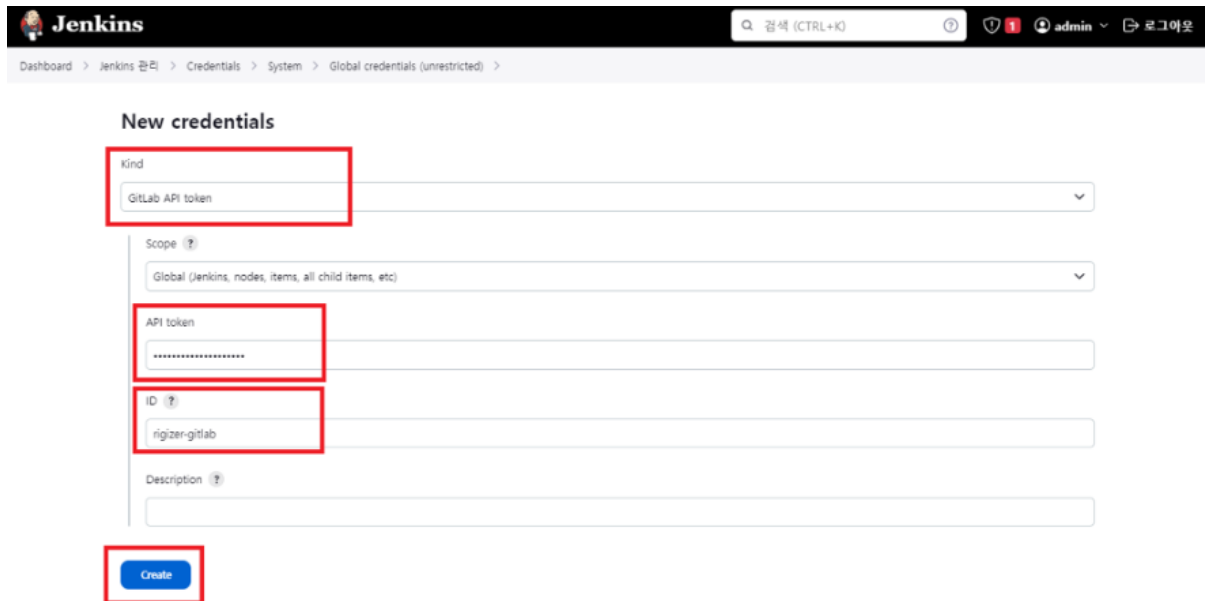
**Global credentials (unrestricted)**

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
This credential domain is empty. How about <a href="#">adding some credentials</a> ?			

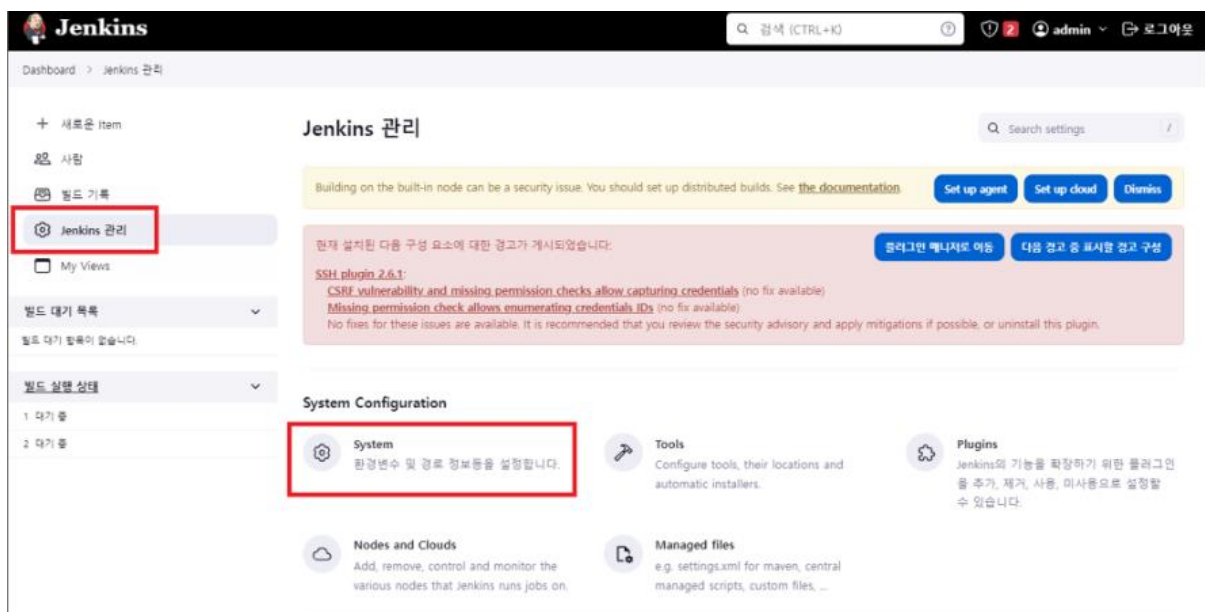
## 【 정보 입력 후 Create 클릭 】

- Kind : Gitlab API token 선택
- API tokens : Gitlab 계정 토큰 입력
- ID : Credential에 대한 별칭



The screenshot shows the 'New credentials' form in Jenkins. The 'Kind' dropdown is set to 'GitLab API token'. The 'Scope' dropdown is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'API token' field contains a masked string '.....'. The 'ID' field is set to 'rigizer-gitlab'. The 'Description' field is empty. The 'Create' button is highlighted with a red box.

## 【 GitLab 커넥션 추가 】



The screenshot shows the 'Jenkins 관리' (Jenkins Management) page. The 'Jenkins 관리' link in the left sidebar is highlighted with a red box. The main content area shows the 'System Configuration' section, which includes 'System', 'Tools', 'Plugins', 'Nodes and Clouds', and 'Managed files'. The 'System' link is highlighted with a red box. A security warning banner is visible at the top of the main content area.

### 【 Gitlab의 **Enable authentication for '/project' end-point** 체크】

- Connection name : Gitlab 커넥션 이름 지정
- Gitlab host URL : Gitlab 시스템의 Host 주소 입력
- Credentials : 조금 전 등록한 **Jenkins Credential (API Token)**을 선택
- 이후, **Test Connection**을 눌러 Success가 뜨면 **저장** 클릭

Gitlab

☒ Enable authentication for '/project' end-point

GitLab connections

Connection name  
A name for the connection

rigizer

Gitlab host URL  
The complete URL to the Gitlab server (e.g. http://gitlab.mydomain.com)

https://lab.ssafy.com

Credentials  
API Token for accessing Gitlab

GitLab API token

Add +

그룹 ▾

Success

Test Connection

추가

저장 Apply

## 8) Jenkins Webhook Integration 설정

- Jenkins 관리 - Plugins 클릭
- Available plugins에서 Generic Webhook Trigger 플러그인 설치

【 Pipeline 아이템에 다음과 같은 설정 추가 】

The screenshot shows the 'Build Triggers' section of the Jenkins Pipeline configuration. The 'General' tab is selected. The 'Build when a change is pushed to GitHub' checkbox is checked. The 'Push Events' checkbox is checked. The 'Opened Merge Request Events' checkbox is checked. The 'Approved Merge Requests (EE-only)' checkbox is checked. The 'Comments' checkbox is checked. The 'Comment (regex) for triggering a build' field contains 'jenkins please retry a build'. The 'Generic Webhook Trigger' checkbox is unchecked. The 'GitHub hook trigger for GITSCM polling' checkbox is unchecked.

【 고급 - Generate 클릭 후 발행된 Secret token 복사해두고 저장 클릭 】

The screenshot shows the 'Advanced' section of the Jenkins Pipeline configuration. The 'Generate' button is highlighted. The 'Secret token' field contains the value '1630e8910192dbff2615102e897fc17e'. The 'Generate' button is highlighted. The 'Save' button is highlighted.

## 【 Gitlab Webhook 지정 】

한재용(실습코치) > gerrit-coach > Webhook Settings



Search page

### Webhooks

Webhooks enable you to send notifications to web applications in response to events in a group or project.

Project Hooks 0

#### URL

http://develop.code-speed.com:9090/project/codespeed

URL must be percent-encoded if it contains one or more special characters.

☒ Show full URL

☐ Mask portions of URL

Do not show sensitive data such as tokens in the UI.

#### Secret token

.....

Used to validate received payloads. Sent with the request in the `X-Gitlab-Token` HTTP header.

#### Trigger

☒ Push events

☐ All branches

☐ Wildcard pattern

☒ Regular expression

develop/be

Regular expressions such as `^(feature|hotfix)/` are supported.

## 【 추가한 화면 】

### Webhooks

Webhooks enable you to send notifications to web applications in response to events in a group or project. We recommend using an [integration](#) in preference to a webhook.

Project Hooks 2

Add new webhook

http://i10a805.p.ssafy.io:8081/project/front/

Test

Edit

Delete

Push events Comments Merge request events SSL Verification: enabled

http://i10a805.p.ssafy.io:8081/project/back/

Test

Edit

Delete

Push events Comments Merge request events SSL Verification: enabled



## 【 프론트엔드 설정 】

### Build Triggers

- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ Build when a change is pushed to GitLab. GitLab webhook URL: `http://i10a805.p.ssafy.io:8081/project/front` ?

Enabled GitLab triggers

- ☒ Push Events ?
- ☐ Push Events in case of branch delete ?
- ☒ Opened Merge Request Events ?
- ☐ Build only if new commits were pushed to Merge Request ?
- ☐ Accepted Merge Request Events ?
- ☐ Closed Merge Request Events ?

Rebuild open Merge Requests ?

Never

- ☒ Approved Merge Requests (EE-only) ?
- ☒ Comments ?

고급 ^

- ☒ Enable [ci-skip] ?
- ☒ Ignore WIP Merge Requests ?

Labels that launch a build if they are added (comma-separated) ?

- ☒ Set build description to build cause (eg. Merge request or Git Push) ?
- ☐ Build on successful pipeline events

Pending build name for pipeline ?

- ☐ Cancel pending merge request builds on update ?

Allowed branches

- ☒ Allow all branches to trigger this job ?
- ☐ Filter branches by name ?
- ☐ Filter branches by regex ?
- ☐ Filter merge request by label

Secret token ?

f9529f48883d30ae778fba21850b7579

Generate

【 프론트엔드 pipeline script 】

```
pipeline{
    agent any

    stages {
        stage('gitlab Connect'){
            steps{
                git branch: 'frontend-develop',
                credentialsId: 'secreto',
                url: 'https://lab.ssafy.com/s10-webmobile1-sub2/S10P12A805.git'
            }
        }

        stage('build'){
            steps{
                sh 'cd /var/jenkins_home/workspace/front/frontend/'
                dir('frontend'){
                    // sh 'npm install -g vite'
                    // sh 'vite build'
                    sh 'npm install'
                    sh 'npm run build'
                }
            }
        }

        stage('deploy'){
            steps{
                dir('frontend'){
                    sh 'cp -r ./dist /var/www/html/'
                }
            }
        }
    }
}
```

```
    }  
  }  
}  
}  
}
```

【 백엔드 pipeline script 】

```
pipeline{  
  agent any  
  
  stages {  
    stage('gitlab Connect'){  
      steps{  
        git branch: 'backend-develop',  
        credentialsId: 'secreto',  
        url: 'https://lab.ssafy.com/s10-webmobile1-sub2/S10P12A805.git'  
      }  
    }  
  
    stage('build'){  
      steps{  
        sh 'cd /var/jenkins_home/workspace/back/backend/'  
        dir('backend'){  
          sh 'chmod +x gradlew'  
          sh './gradlew clean build'  
        }  
      }  
    }  
  }  
}
```

```
    }  
  }  
  
  stage('deploy'){  
    steps{  
      sh 'docker stop secreto && docker rm secreto && docker rmi backend'  
      dir('backend'){  
        sh 'docker build -t backend ./'  
        sh 'docker run --restart=on-failure -p 8080:8080 -d --name secreto  
backend'  
      }  
    }  
  }  
}
```