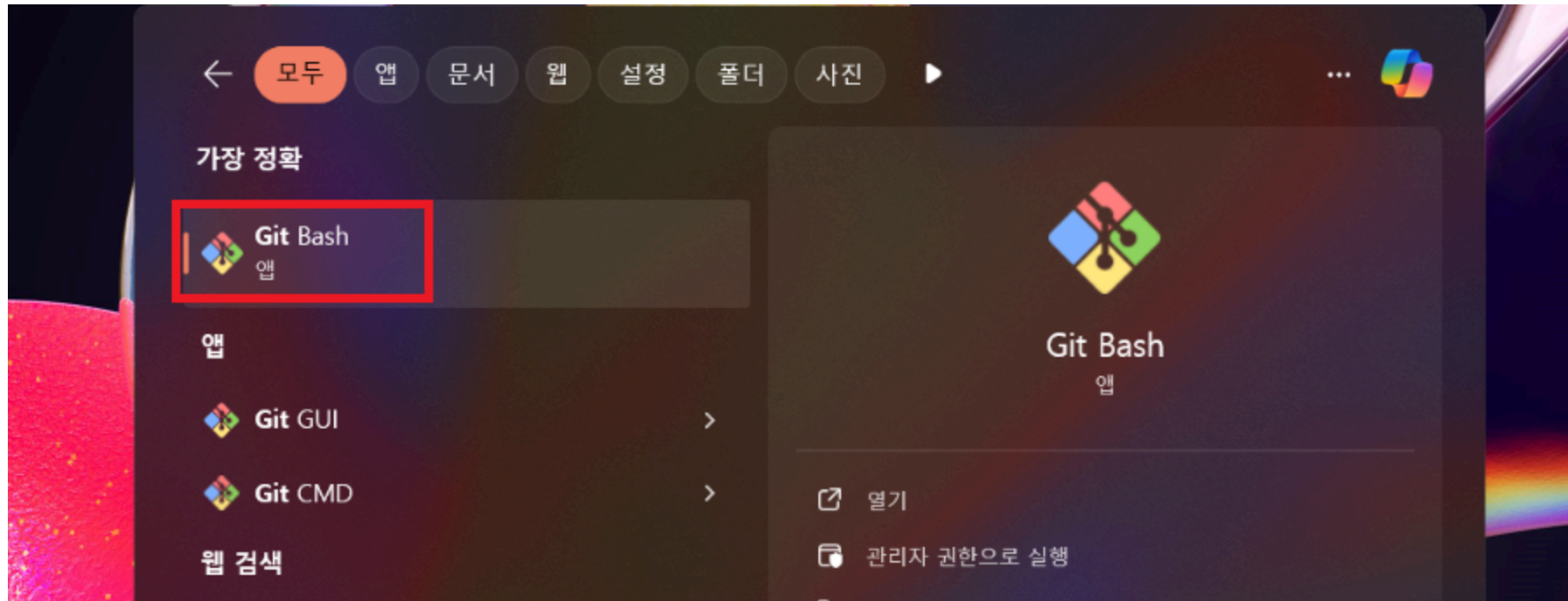


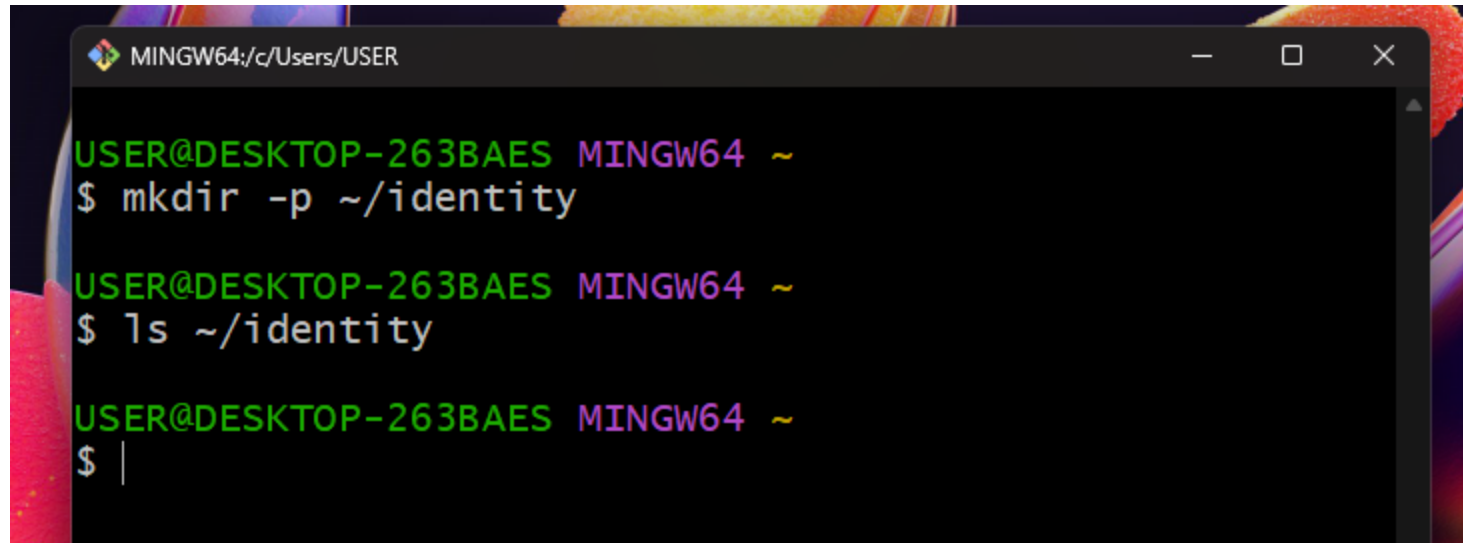
# Git Bash 접속 방법

## 단계1: Git Bash 열기



## 단계2: identity 폴더 생성

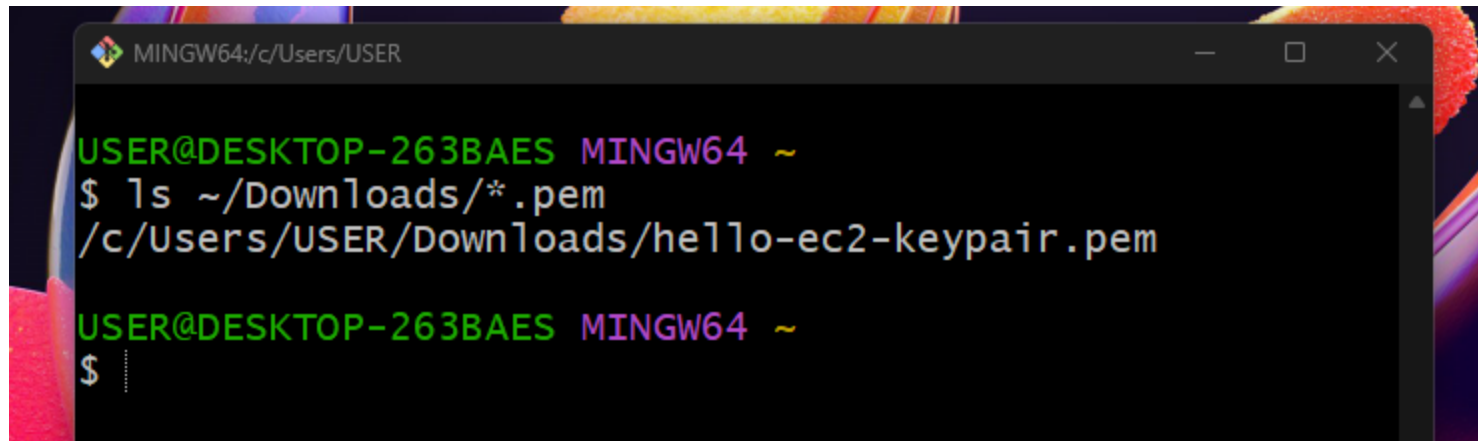
```
mkdir -p ~/identity # 폴더 생성  
ls ~/identity # 폴더 확인
```



```
MINGW64:/c/Users/USER  
  
USER@DESKTOP-263BAES MINGW64 ~  
$ mkdir -p ~/identity  
  
USER@DESKTOP-263BAES MINGW64 ~  
$ ls ~/identity  
  
USER@DESKTOP-263BAES MINGW64 ~  
$ |
```

### 단계3: key pair 파일 확인

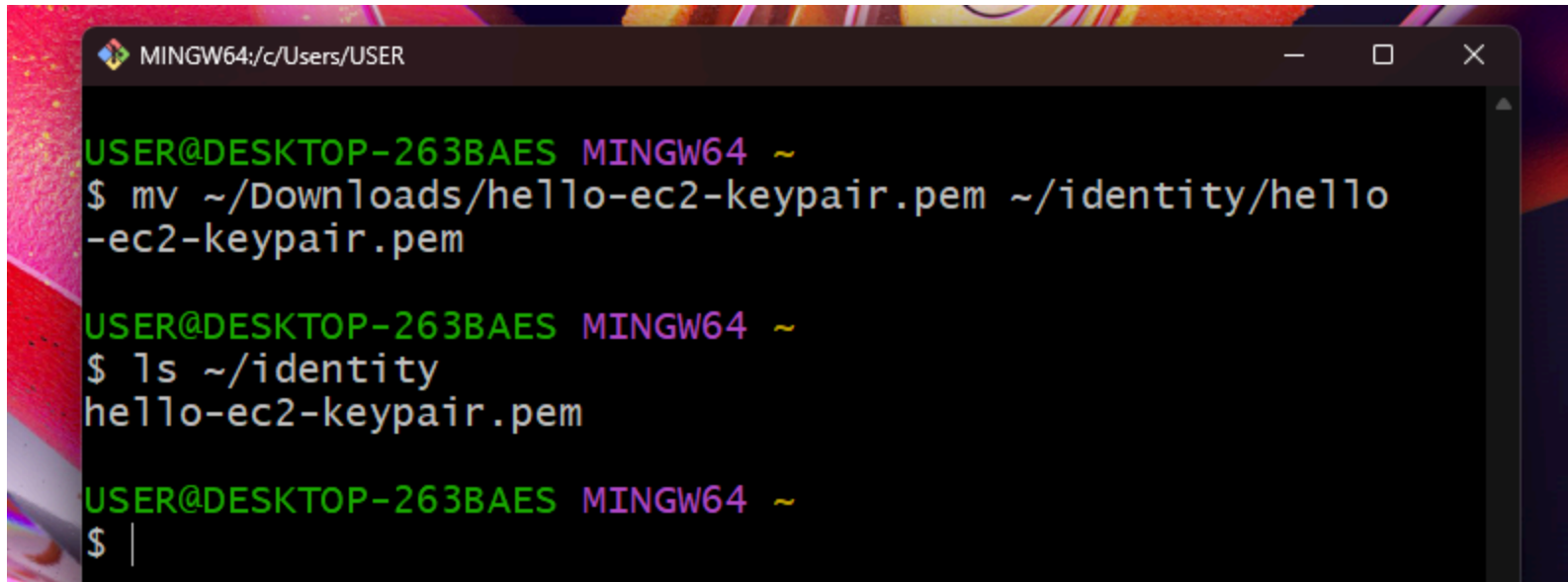
```
ls ~/Downloads/*.pem
```



```
MINGW64:/c/Users/USER  
USER@DESKTOP-263BAES MINGW64 ~  
$ ls ~/Downloads/*.pem  
/c/Users/USER/Downloads/hello-ec2-keypair.pem  
USER@DESKTOP-263BAES MINGW64 ~  
$
```

## 단계4: key pair 파일 이동

```
# 파일 이동
mv ~/Downloads/hello-ec2-keypair.pem ~/identity/hello-ec2-keypair.pem
# 결과 확인
ls ~/identity
```



A screenshot of a Windows command prompt window titled "MINGW64: c:/Users/USER". The window shows the following commands and output:

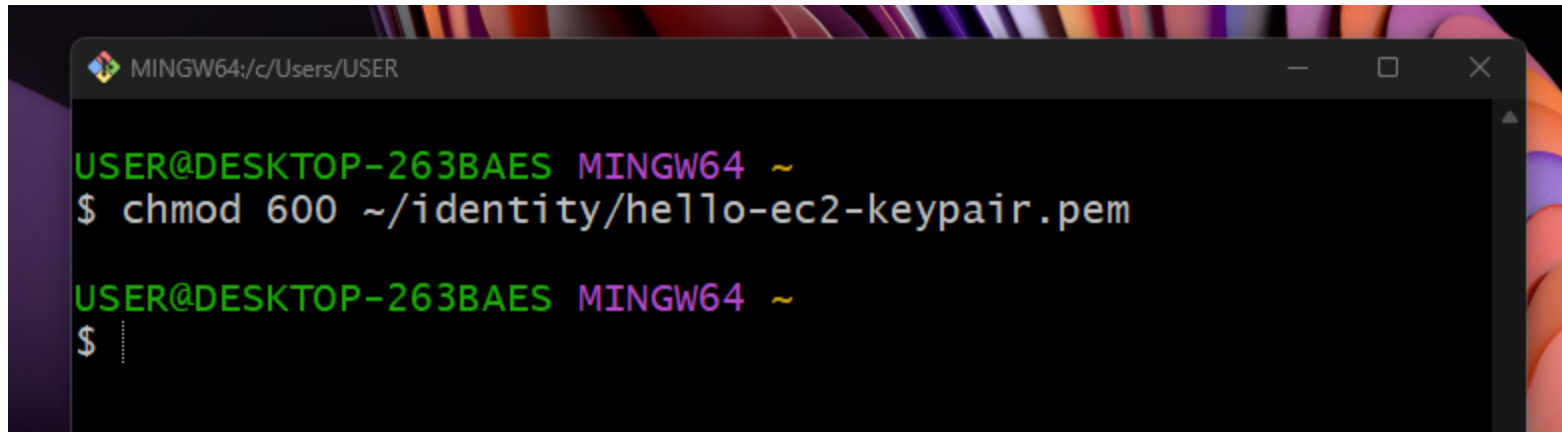
```
USER@DESKTOP-263BAES MINGW64 ~
$ mv ~/Downloads/hello-ec2-keypair.pem ~/identity/hello-ec2-keypair.pem

USER@DESKTOP-263BAES MINGW64 ~
$ ls ~/identity
hello-ec2-keypair.pem

USER@DESKTOP-263BAES MINGW64 ~
$ |
```

## 단계5: (윈도우 생략) 권한 변경

```
# pem 키 권한 변경 -> 600이 아닐 경우 보안 취약으로 판단  
chmod 600 ~/identity/hello-ec2-keypair.pem
```

A screenshot of a Windows command prompt window titled "MINGW64: c:/Users/USER". The window has a black background with green and white text. The prompt shows the user "USER@DESKTOP-263BAES" in a "MINGW64" environment. The command "\$ chmod 600 ~/identity/hello-ec2-keypair.pem" has been entered and executed. The prompt then shows the user at the "\$" prompt, ready for the next command.

```
MINGW64: c:/Users/USER  
  
USER@DESKTOP-263BAES MINGW64 ~  
$ chmod 600 ~/identity/hello-ec2-keypair.pem  
  
USER@DESKTOP-263BAES MINGW64 ~  
$
```

## 단계6: 인스턴스 Public IP 복사

- 인스턴스 Public IP는 서버를 stop할때마다 변경됨!!

The screenshot shows the AWS Management Console interface. On the left, the navigation menu includes 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Instances', 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', 'Capacity Reservations', 'Images', and 'AMIs'. The main content area displays the 'Instances (1/1)' page. A table lists the instance 'hello-ec2' with ID 'i-0e0a30e99c1f7e3a4', state 'Running', type 't2.micro', and status '2/2 checks passed'. The instance is highlighted with a red box. Below the table, the 'Details' tab is selected for the instance 'i-0e0a30e99c1f7e3a4 (hello-ec2)'. The 'Instance summary' section shows the 'Public IPv4 address' as '43.202.33.26', which is also highlighted with a red box and pointed to by a red arrow. Other details include the 'Instance ID' 'i-0e0a30e99c1f7e3a4 (hello-ec2)' and 'Private IPv4 addresses' '172.31.14.95'.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
hello-ec2	i-0e0a30e99c1f7e3a4	Running	t2.micro	2/2 checks passed	View alarms	ap-no

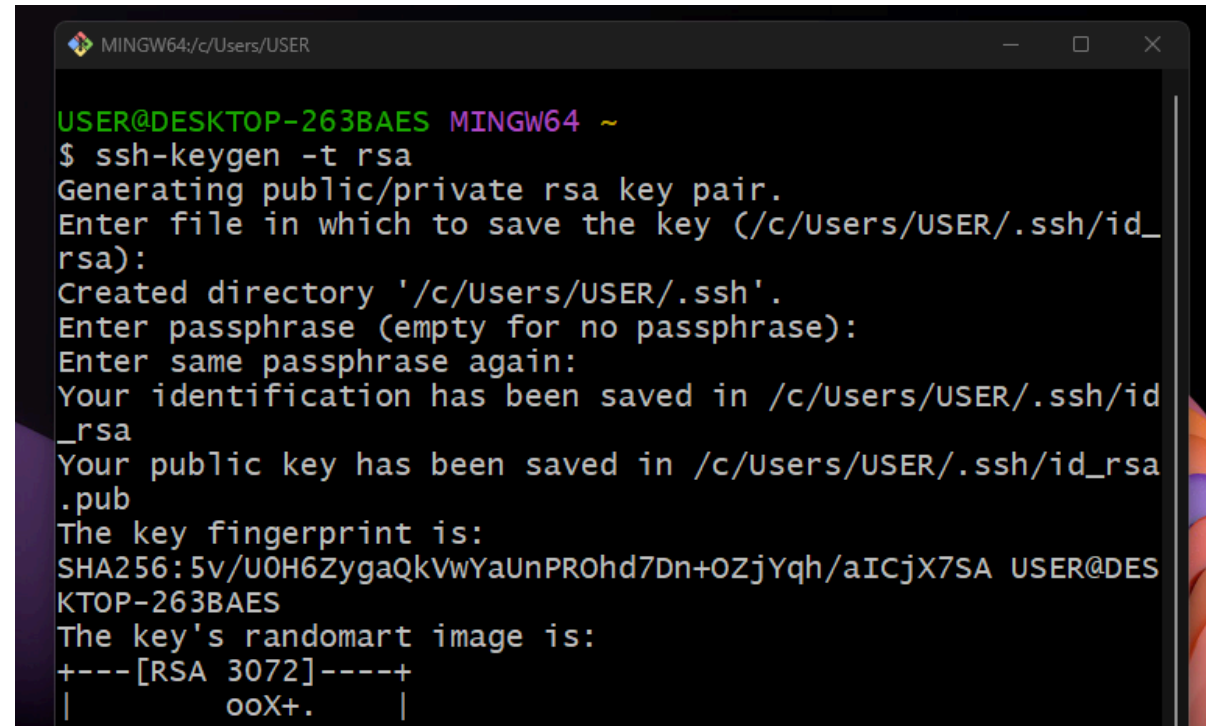
**i-0e0a30e99c1f7e3a4 (hello-ec2)**

**Instance summary**

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0e0a30e99c1f7e3a4 (hello-ec2)	43.202.33.26   <a href="#">open address</a>	172.31.14.95

## 단계7: ssh key 만들기

```
ssh-keygen -t rsa  
# enter 세 번 탁! 탁! 탁!
```

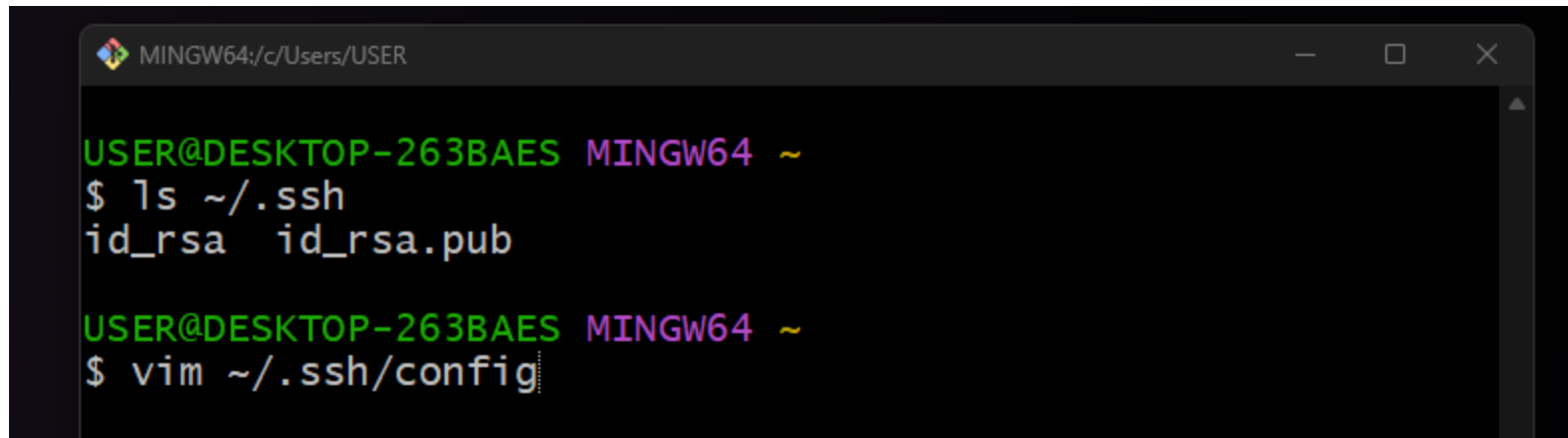
A terminal window titled 'MINGW64; c/Users/USER' showing the execution of the 'ssh-keygen -t rsa' command. The output shows the generation of an RSA key pair, the creation of the '.ssh' directory, and the saving of the public key 'id\_rsa.pub' and private key 'id\_rsa'. It also displays the key's fingerprint and a randomart image.

```
MINGW64; c/Users/USER  
USER@DESKTOP-263BAES MINGW64 ~  
$ ssh-keygen -t rsa  
Generating public/private rsa key pair.  
Enter file in which to save the key (/c/Users/USER/.ssh/id_rsa):  
Created directory '/c/Users/USER/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /c/Users/USER/.ssh/id_rsa  
Your public key has been saved in /c/Users/USER/.ssh/id_rsa.pub  
The key fingerprint is:  
SHA256:5v/U0H6ZygaQkVwYaUnPROhd7Dn+OZjYqh/aICjX7SA USER@DESKTOP-263BAES  
The key's randomart image is:  
+---[RSA 3072]---+  
|                 |  
|                ooX+. |
```



## 단계8: ssh config 생성

```
# ssh key 확인  
ls ~/.ssh  
# ssh config 생성 및 작성  
vim ~/.ssh/config
```



A screenshot of a Windows command prompt window titled "MINGW64: c:/Users/USER". The window shows the following commands and output:

```
USER@DESKTOP-263BAES MINGW64 ~  
$ ls ~/.ssh  
id_rsa  id_rsa.pub  
  
USER@DESKTOP-263BAES MINGW64 ~  
$ vim ~/.ssh/config
```

## 단계9: ssh config 작성 후 저장

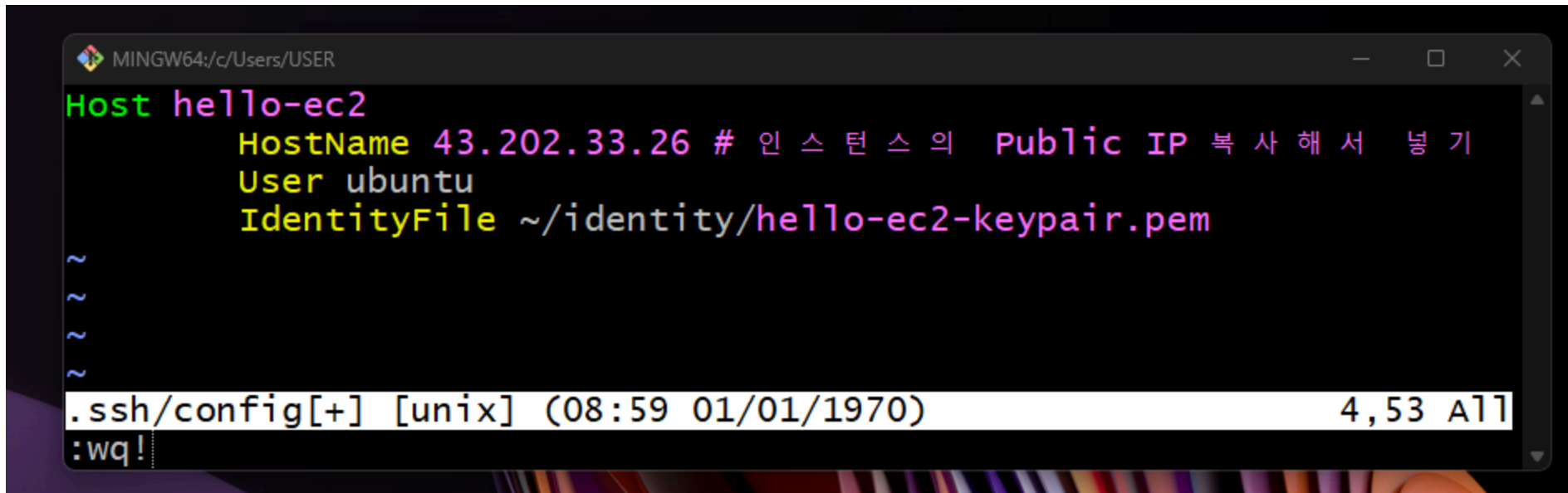
# 아래 내용 추가 후 저장

Host hello-ec2

HostName 43.202.33.26 # 인스턴스의 Public IP 복사해서 넣기

User ubuntu

IdentityFile ~/.identity/hello-ec2-keypair.pem



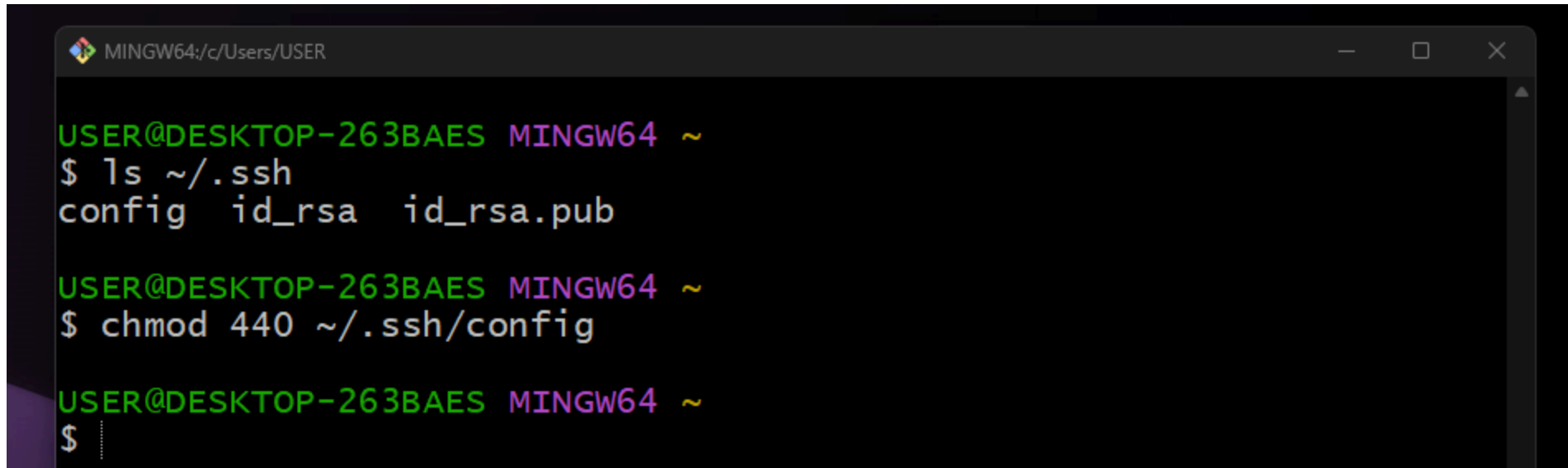
The screenshot shows a Windows command prompt window titled "MINGW64: c:/Users/USER". The user has entered the following commands to save the SSH configuration:

```
Host hello-ec2
    HostName 43.202.33.26 # 인스턴스의 Public IP 복사해서 넣기
    User ubuntu
    IdentityFile ~/.identity/hello-ec2-keypair.pem
```

The prompt shows the user is in the `~` directory. The command prompt shows the user is in the `.ssh/config` file, and the status bar indicates the file is in `[unix]` mode, with a timestamp of `(08:59 01/01/1970)` and a cursor position of `4,53 A11`. The user has entered `:wq!` to save and quit the editor.

## 단계10: ssh config 생성

```
# ssh config 확인  
ls ~/.ssh  
# config 파일 권한 수정  
chmod 440 ~/.ssh/config
```

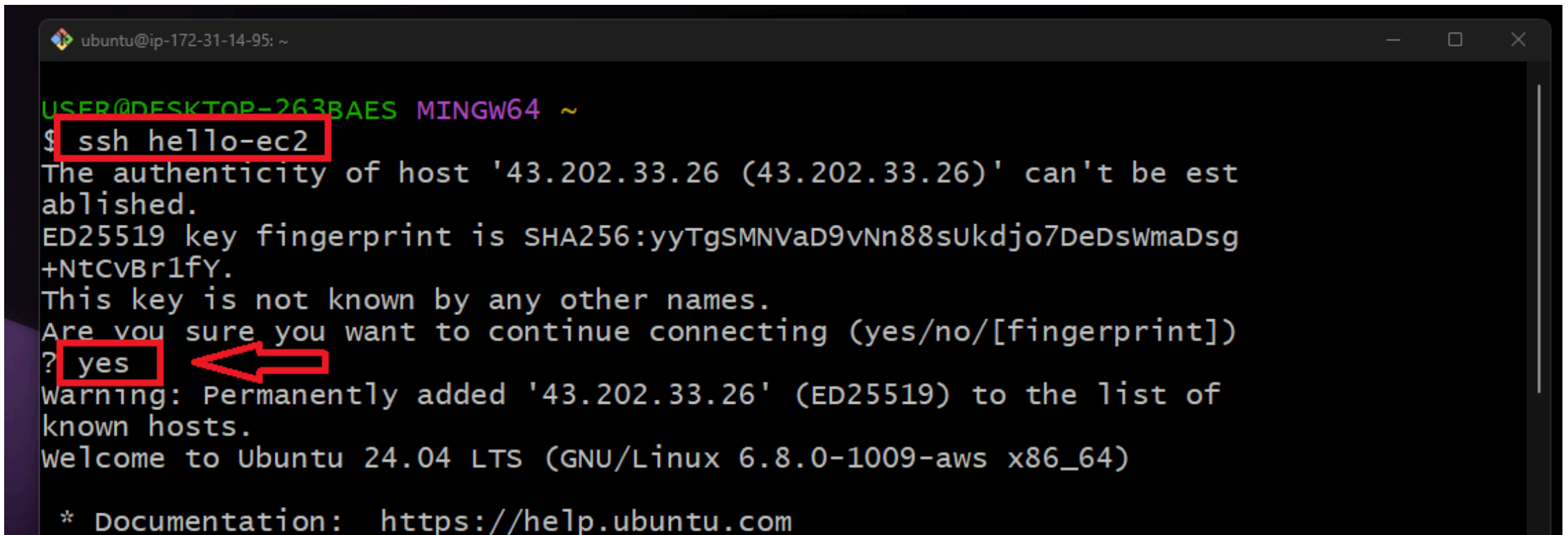


A screenshot of a Windows command prompt window titled "MINGW64: c:/Users/USER". The window shows the following commands and their outputs:

```
USER@DESKTOP-263BAES MINGW64 ~  
$ ls ~/.ssh  
config  id_rsa  id_rsa.pub  
  
USER@DESKTOP-263BAES MINGW64 ~  
$ chmod 440 ~/.ssh/config  
  
USER@DESKTOP-263BAES MINGW64 ~  
$
```

## 단계11: hello-ec2 접속

```
# hello-ec2 접속
ssh hello-ec2
# 진짜 연결하시겠습니까? yes 입력 후 Enter
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

A terminal window titled 'ubuntu@ip-172-31-14-95: ~' displays the command 'ssh hello-ec2' being executed. The terminal output shows a warning about the authenticity of the host '43.202.33.26' and its ED25519 key fingerprint. It asks for confirmation to continue connecting. The user enters 'yes', which is highlighted with a red box and a red arrow. The terminal then shows the warning that the host has been added to the list of known hosts and displays the Ubuntu 24.04 LTS welcome message and documentation link.

```
ubuntu@ip-172-31-14-95: ~
USER@DESKTOP-263BAES MINGW64 ~
$ ssh hello-ec2
The authenticity of host '43.202.33.26 (43.202.33.26)' can't be est
ablished.
ED25519 key fingerprint is SHA256:yyTgSMNVaD9vNn88sUkdjo7DeDswmaDsg
+NtCvBr1fY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])
? yes
Warning: Permanently added '43.202.33.26' (ED25519) to the list of
known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1009-aws x86_64)

* Documentation:  https://help.ubuntu.com
```

## 단계12: hello-ec2 접속 확인

- Private IP와 Git Bash에 표시된 IP가 같은지 확인

The screenshot shows the AWS Management Console interface for the 'Instances' page. The instance 'hello-ec2' is listed with ID 'i-0e0a30e99c1f7e3a4', state 'Running', and type 't2.micro'. Below the table, the 'Details' tab is selected for instance 'i-0e0a30e99c1f7e3a4 (hello-ec2)'. The 'Instance summary' section displays the following information:

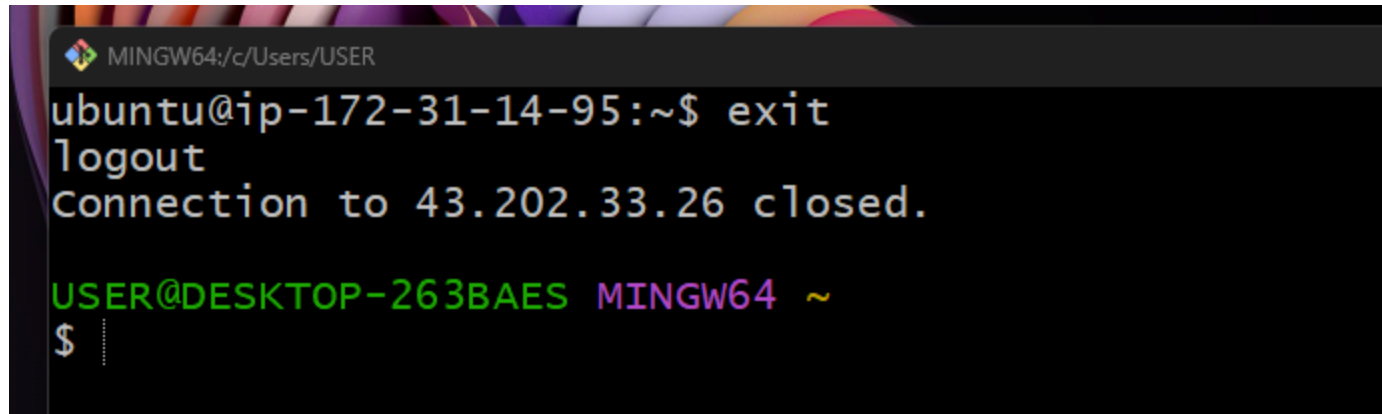
Instance ID	Public IPv4 address	Private IPv4 addresses
i-0e0a30e99c1f7e3a4 (hello-ec2)	43.202.33.26   <a href="#">open address</a>	172.31.14.95

Other fields shown include 'IPv6 address', 'Instance state', and 'Public IPv4 DNS'.

The screenshot shows a terminal window with the prompt 'ubuntu@ip-172-31-14-95: ~\$'. The IP address 'ip-172-31-14-95' in the prompt is highlighted with a red box, and a red arrow points to it from the left. The terminal shows the command 'pwd' and the output '/home/ubuntu'.

## 단계13: 접속 종료

exit



```
MINGW64:/c/Users/USER
ubuntu@ip-172-31-14-95:~$ exit
logout
Connection to 43.202.33.26 closed.

USER@DESKTOP-263BAES MINGW64 ~
$
```