

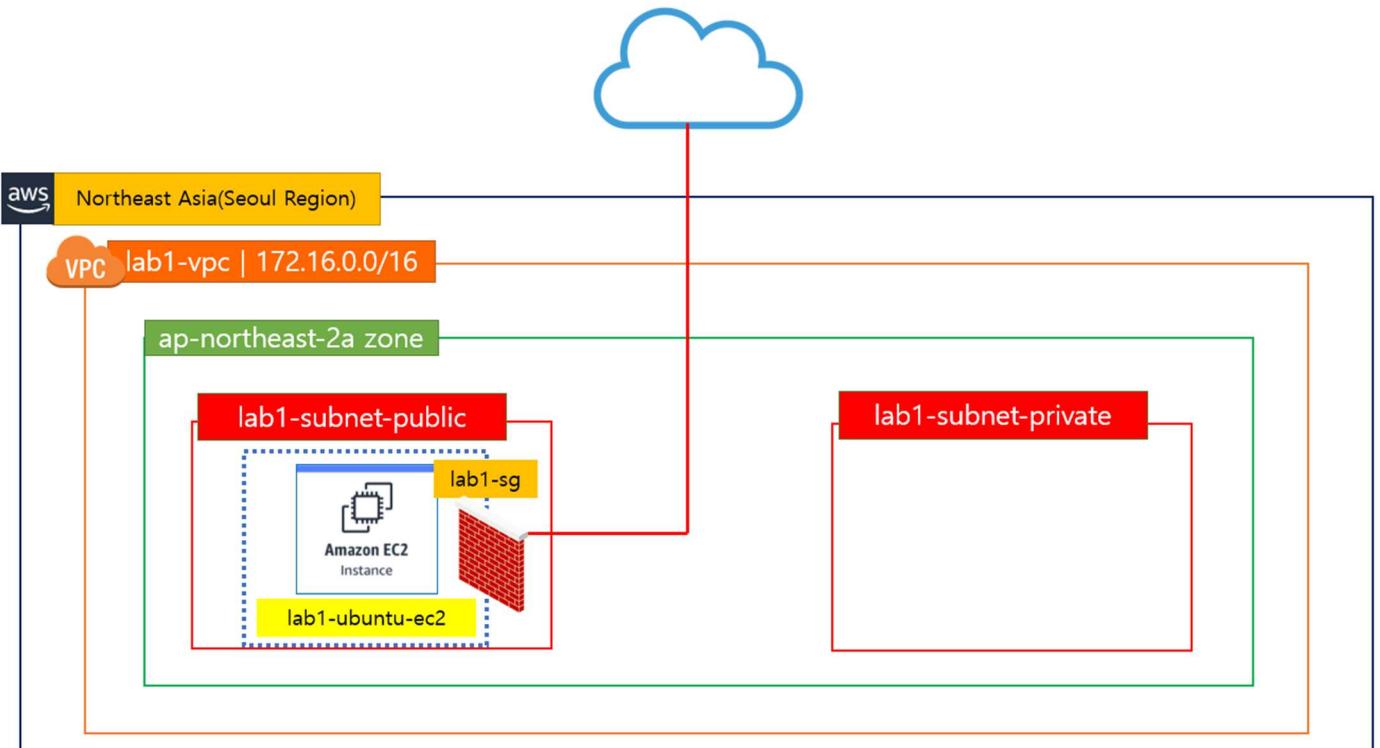
Lab1. Server-Oriented Hello World

목적

Python Flask 라이브러리를 이용한 간단한 웹 애플리케이션을 실행하기 위해 VPC + EC2를 생성하고 Flask 라이브러리를 설치한 후, 간단하게 Hello, World 문자열 출력

사전 준비물

AWS Free-Tier 계정



Create VPC for Lab1

- 로그인 후, **VPC** 페이지로 이동한다. VPC를 새로 생성하기 위해 **[VPC 생성]** 주황색 버튼을 클릭한다.

The screenshot shows the AWS VPC dashboard. On the left, there's a sidebar with various VPC-related options like 'Virtual Private Cloud', '서브넷', '라우팅 테이블', etc. In the center, there's a section titled '리전별 리소스' (Regional Resources) with a grid of buttons for different VPC components. At the top right of this section, there's a large orange button labeled 'VPC 생성' (Create VPC). Below the grid, there's a note: 'Amazon VPC 리소스를 사용하고 있습니다.' (Using Amazon VPC resources).

- [VPC 생성]** 페이지에서 다음과 같이 각각의 값을 설정하여 생성한다.

- [생성할 리소스]** : VPC 등
- [이름 태그]** : lab1
- [IPv4 CIDR]** : 172.16.0.0/16
- [태넌시]** : 기본값

The screenshot shows the 'Create VPC' configuration page. It has several sections:

- VPC 설정** (VPC Settings):
 - 생성할 리소스** (Create Resource): A radio button group where 'VPC 등' (VPC and others) is selected, highlighted with a red box.
 - 이름 태그 자동 생성** (Auto-Tag Name): A checked checkbox '자동 생성' (Auto-create) with the value 'lab1' in the input field, also highlighted with a red box.
- IPv4 CIDR 블록** (IPv4 CIDR Block): Shows '172.16.0.0/16' and '65,536 IPs'.
- IPv6 CIDR 블록** (IPv6 CIDR Block): Shows 'IPv6 CIDR 블록 없음' (No IPv6 CIDR block).
- 테넌시** (Tenancy): Shows '기본값' (Default) in a dropdown menu, highlighted with a red box.

3. [가용 영역(AZ) 수] 섹션에서 1개를 선택한다. 그 아래 [AZ 사용자 지정]을 클릭하여 ap-northeast-2a를 선택한다. [퍼블릭 서브넷 수]와 [프라이빗 서브넷 수]는 각각 1개를 선택한다.

가용 영역(AZ) 수 정보
서브넷을 프로비저닝할 AZ 수를 선택합니다. 고가용성을 위해서는 최소 2개 이상의 AZ를 사용하는 것이 좋습니다.

1	2	3
---	---	---

▼ AZ 사용자 지정

첫 번째 가용 영역

퍼블릭 서브넷 수 정보
VPC에 추가할 퍼블릭 서브넷 수입니다. 인터넷을 통해 공개적으로 액세스할 수 있어야 하는 웹 애플리케이션에는 퍼블릭 서브넷을 사용합니다.

0	1
---	---

프라이빗 서브넷 수 정보
VPC에 추가할 프라이빗 서브넷 수입니다. 프라이빗 서브넷을 사용하여 퍼블릭 액세스가 필요 없는 백엔드 리소스를 보호합니다.

0	1	2
---	---	---

▶ 서브넷 CIDR 블록 사용자 지정

4. [NAT 게이트웨이]는 [1개의 AZ에서]를 선택하고, 나머지는 기본값 그대로 사용하기로 한다. [VPC 생성] 오른쪽 버튼을 클릭한다.

NAT 게이트웨이(\$) 정보
NAT 게이트웨이를 생성할 가용 영역(AZ) 수를 선택합니다. 각 NAT 게이트웨이마다 요금이 부과됩니다.

없음	1개의 AZ에서	AZ당 1개
----	----------	--------

VPC 엔드포인트 정보
엔드포인트는 VPC에서 S3에 직접 액세스하여 NAT 게이트웨이 요금을 줄이고 보안을 강화할 수 있습니다. 기본적으로 모든 액세스 정책이 사용됩니다. 언제든지 이 정책을 사용자 지정할 수 있습니다.

없음	S3 게이트웨이
----	----------

DNS 옵션 정보
 DNS 호스트 이름 활성화
 DNS 확인 활성화

취소 VPC 생성

5. 정상적으로 VPC가 잘 생성되었고 방금 생성한 VPC의 [상태]가 Available임을 확인한다.

VPC > VPC > vpc-0d3d2c1f04857e1c3

vpc-0d3d2c1f04857e1c3 / lab1-vpc

세부 정보		설정	
VPC ID	vpc-0d3d2c1f04857e1c3	상태	Available
데넌시	Default	DNS 호스트 이름	활성화됨
기본 VPC	아니요	기본 라우팅 테이블	rtb-0893f1e6f79bbcd5
네트워크 주소 사용 지표	비활성화됨	IPv6 풀	-
		Route 53 Resolver DNS 방화벽 규칙 그룹	-
		소유자 ID	789534828835

CIDR | 플로우 로그 | 태그

CIDR 정보

주소 유형	CIDR	네트워크 경계 그룹	풀	상태
IPv4	172.16.0.0/16	-	-	Associated

VPC (1) 정보

Name	VPC ID	상태	IPv4 CIDR	DHCP 옵션 세트	기본 라우팅 테이블	기본 네트워크 ACL
lab1-vpc	vpc-0d3d2c1f04857e1c3	Available	172.16.0.0/16	dopt-0a5ca099f137a6...	rtb-0893f1e6f79bbcd5	acl-07e909a7d6c36fa75

Create EC2 with Ubuntu Linux

1. AWS 콘솔에 로그인 후, EC2 페이지로 이동한다.

The screenshot shows the AWS Management Console with the search bar at the top containing "서비스, 기능, 블로그, 설명서 등을 검색합니다." and the keyboard shortcut "[Option+S]". On the left, there is a sidebar with various service icons and names in Korean. The main area is titled "컴퓨팅" (Computing) and lists several services:

- AWS App Runner**: Build and run production web applications at scale
- Batch**: 규모에 상관없는 완전관리형 배치 처리
- EC2** (highlighted with a red box): 클라우드의 가상 서버
- EC2 Image Builder**: OS 이미지 빌드, 사용자 지정 및 배포를 자동화하는 관리형 서비스
- Elastic Beanstalk**: 웹 앱 실행 및 관리
- Lambda**: 서버에 대한 걱정 없이 코드 실행
- Lightsail**: 가상 프라이빗 서버 시작 및 관리
- AWS Outposts**: 온프레미스에서 AWS 서비스 실행

2. 우측 상단의 [인스턴스 시작] 오렌지 색 버튼을 클릭한다.

The screenshot shows the EC2 Instances page. The left sidebar has a collapsed "인스턴스" (Instances) section with a "New" badge. The main area is titled "인스턴스 정보" (Instance Information) and displays a table with columns: Name, 인스턴스 ID, 인스턴스 상태, 인스턴스 유형, 상태 검사, 경보 상태, and 가용 영역. A message at the bottom says "이 리전에는 인스턴스가 없음" (No instances in this region). At the top right, there is a prominent orange button labeled "인스턴스 시작" (Launch Instance), which is highlighted with a red box.

3. [인스턴스 시작] 페이지이다. [이름 및 태그] 섹션에서 [이름]에 lab1-ubuntu-ec2를 입력한다.

EC2 > 인스턴스 > 인스턴스 시작

인스턴스 시작 정보

Amazon EC2를 사용하면 AWS 클라우드에서 실행되는 가상 머신 또는 인스턴스를 생성할 수 있습니다. 아래의 간단한 단계에 따라 빠르게 시작할 수 있습니다.

이름 및 태그 정보

이름
lab1-ubuntu-ec2

추가 태그 추가

4. [애플리케이션 및 OS 이미지(Amazon Machine Image)] 섹션에서 [Quick Start] 항목 중 [Ubuntu]를 선택하고 그 아래 목록에서 [Ubuntu Server 22.04 LTS(HVM), SSD Volume Type] 서버를 찾은 후 [아키텍처]는 [64비트(x86)]이 선택되어 있는 것을 확인한다.

▼ 애플리케이션 및 OS 이미지(Amazon Machine Image) 정보

AMI는 인스턴스를 시작하는 데 필요한 소프트웨어 구성(운영 체제, 애플리케이션 서버 및 애플리케이션)이 포함된 템플릿입니다. 아래에서 찾고 있는 항목이 보이지 않으면 AMI를 검색하거나 찾아보십시오.

Search our full catalog including 1000s of application and OS images

최근 사용 Quick Start

Amazon Linux macOS Ubuntu Windows Red Hat ... 더 많은 AMI 찾아보기 AWS, Marketplace 및 커뮤니티의 AMI 포함

Amazon Machine Image(AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type
 ami-0f0646a5f59758444 (64비트(x86)) / ami-076c814e42cb96d58 (64비트(Arm))
 가상화: hvm ENA 활성화됨: true 루트 디바이스 유형: ebs

설명
Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2022-12-06

아키텍처
64비트(x86)

AMI ID
ami-0f0646a5f59758444

확인된 공급 업체

5. [인스턴스 유형] 섹션에서, [t2.micro]를 선택한다.

The screenshot shows the 'Instance Type' section of the AWS Lambda configuration page. A red box highlights the dropdown menu where 't2.micro' is selected. The dropdown also lists 't2.micro' (1 vCPU, 1 GiB memory), '프리 티어 사용 가능' (Free Tier usage available), and prices for '온디맨드 Linux 요금: 0.0144 USD 시간당' (On-Demand Linux price: 0.0144 USD per hour) and '온디맨드 Windows 요금: 0.019 USD 시간당' (On-Demand Windows price: 0.019 USD per hour). A blue link '인스턴스 유형 비교' (Compare instance types) is visible on the right.

6. [Key pair (login)] 섹션에서, 새 key pair를 생성하기 위해 [Create new key pair] 링크를 클릭한다.

The screenshot shows the 'Key pair (login)' section. A red box highlights the 'Create new key pair' button, which is labeled with a circular icon containing a 'C' and the text 'Create new key pair'.

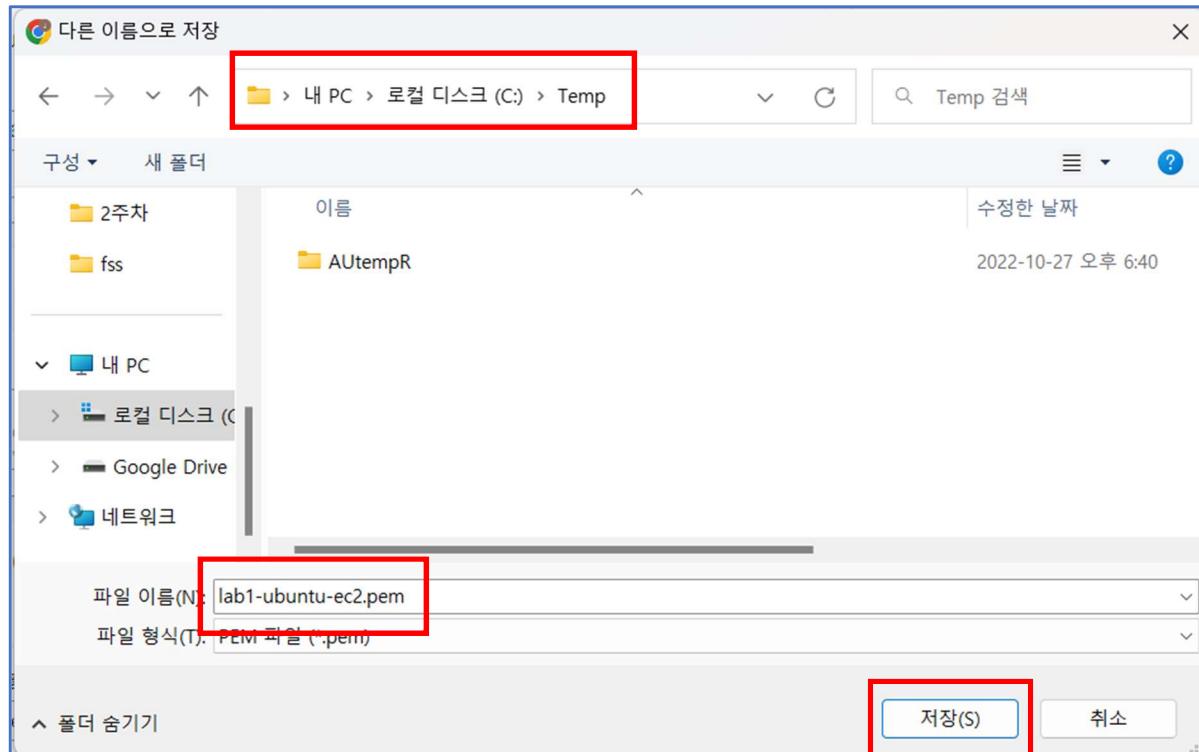
7. [키 페어 생성] 창에서, [키 페어 이름]에 lab1-ubuntu-ec2를 입력한다. 그리고 [키 페어 유형]은 RSA로, [프라이빗 키 파일 형식]은 .pem으로 선택되어 있음을 확인하고, [키 페어 생성] 버튼을 클릭하여 창을 닫는다.

The screenshot shows the 'Key Pair Creation' dialog box. Several fields are highlighted with red boxes:

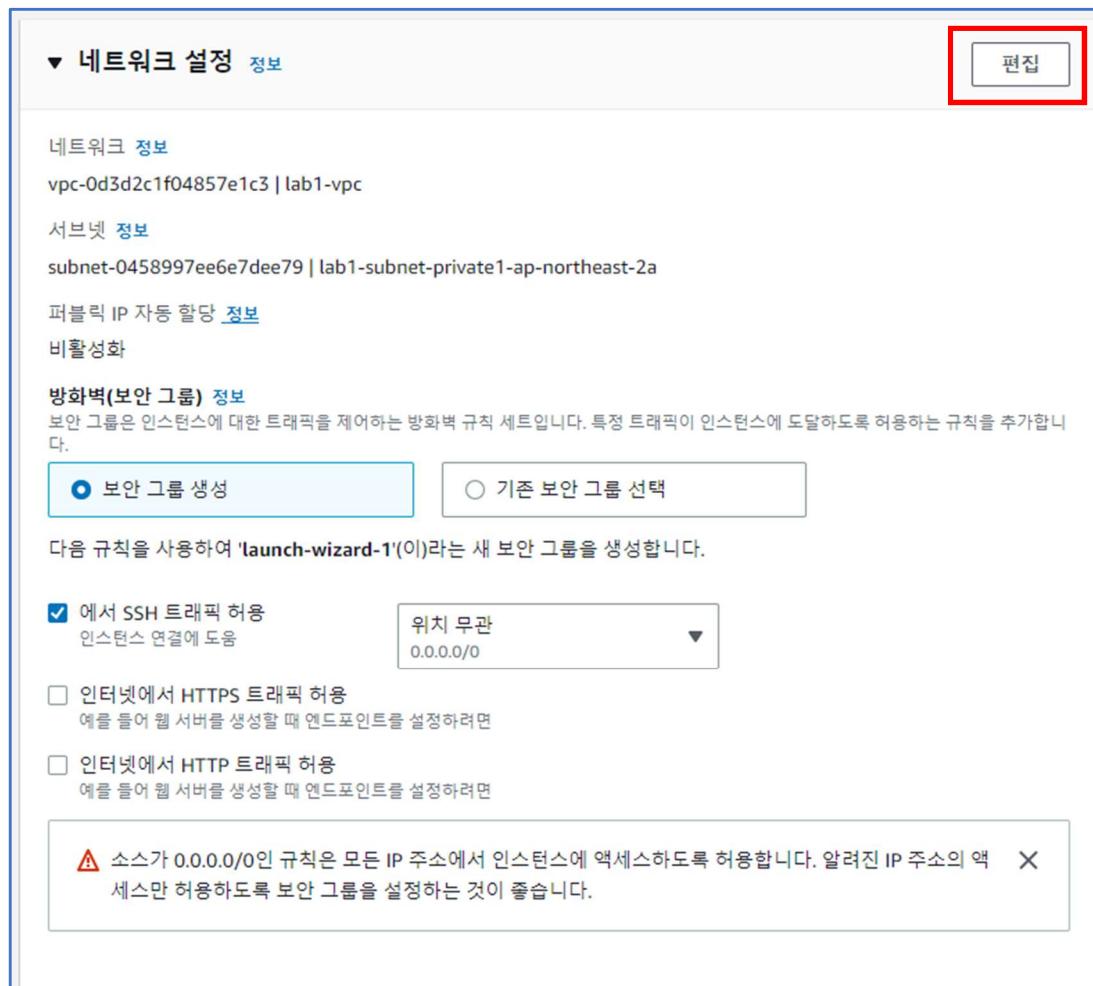
- [Key Pair Name]: Contains 'lab1-ubuntu-ec2'.
- [Key Pair Type]: Shows 'RSA' selected (radio button).
- [Private Key File Format]: Shows '.pem' selected (radio button).
- [Create New Key Pair] Button: This button is also highlighted with a red box.

The dialog also contains descriptive text and links for additional information.

8. 위에서 생성한 pem 파일을 찾기 쉬운 C:/Temp에 저장하기로 한다. 파일 이름을 확인하고 [저장]을 클릭한다.



9. [네트워크 설정] 섹션에서 네트워크 정보를 수정하기 위해 [편집] 버튼을 클릭한다.



10. [VPC]는 lab1-vpc로, [서브넷]은 lab1-subnet-public1-ap-northeast-2a로, [퍼블릭 IP 자동 할당]은 활성화를 각각 선택한다.

11. [방화벽(보안 그룹)]은 [보안 그룹 생성]을 선택하고, [보안 그룹 이름]은 lab1-sg로, [설명]에는 Security Group for Lab1 Web Server로 입력한다. 그 아래 [인바운드 보안 그룹 규칙]은 기본값이 ssh 22번 포트가 설정되어 있다. 보안 그룹 규칙을 추가하기 위해 [보안 그룹 규칙 추가]를 클릭한다.

12. 다음과 같이 HTTP 규칙을 추가한다.

- A. [유형] : HTTP
- B. [프로토콜] : TCP
- C. [포트 범위] : 80
- D. [소스] : 위치 무관

▼ 보안 그룹 규칙 2 (TCP, 80, 0.0.0.0/0)

제거

유형 정보	프로토콜 정보	포트 범위 정보
HTTP	TCP	80
소스 유형 정보	원본 정보	설명 - optional 정보
위치 무관	<input type="text"/> CIDR, 접두사 목록 또는 보안 그룹	예: 관리자 데스크톱용 SSH
	<input type="text"/> 0.0.0.0/0	X

⚠ 소스가 0.0.0.0/0인 규칙은 모든 IP 주소에서 인스턴스에 액세스하도록 허용합니다. 알려진 IP 주소의 액세스만 허용하도록 보안 그룹을 설정하는 것이 좋습니다. X

보안 그룹 규칙 추가

▶ 고급 네트워크 구성

13. [스토리지 구성] 섹션에서, 기본 사이즈 8GiB를 지우고 최대 용량 30GiB를 설정한다.

▼ 스토리지 구성 정보

어드밴스드

1x	<input type="text"/> 30	GiB	gp2	▼	Root volume (암호화되지 않음)
----	-------------------------	-----	-----	---	------------------------

ⓘ 프리 티어를 사용할 수 있는 고객은 최대 30GB의 EBS 범용(SSD) 또는 마그네틱 스토리지를 사용할 수 있습니다. X

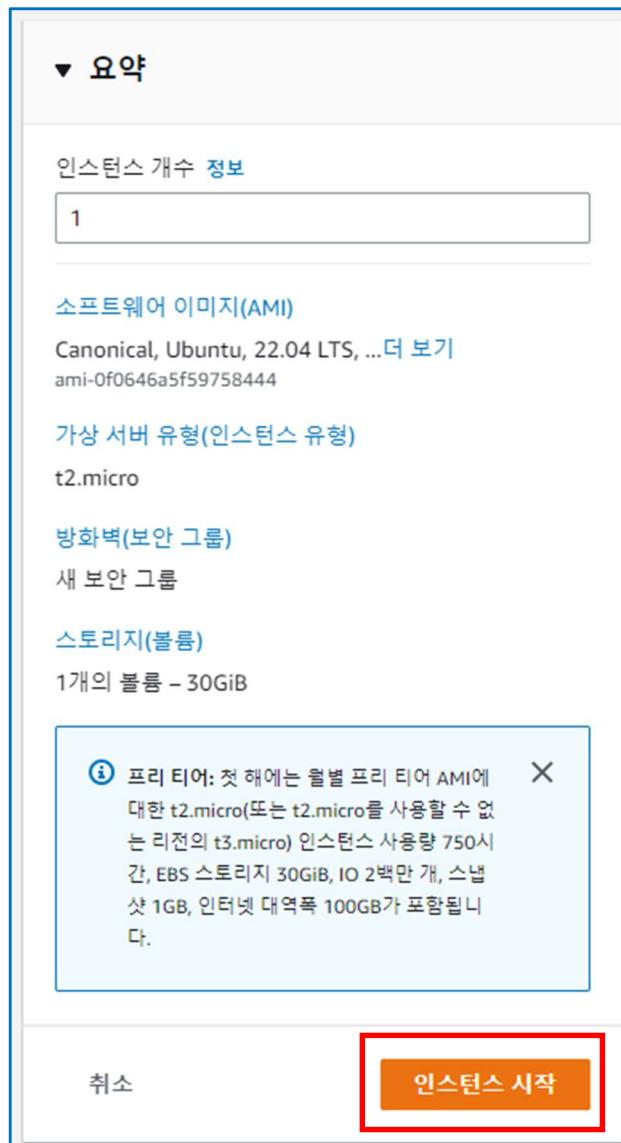
새 볼륨 추가

선택한 AMI에 인스턴스가 허용하는 것보다 많은 인스턴스 스토어 볼륨이 포함되어 있습니다. AMI에서 처음 0개의 인스턴스 스토어 볼륨에만 액세스할 수 있습니다.

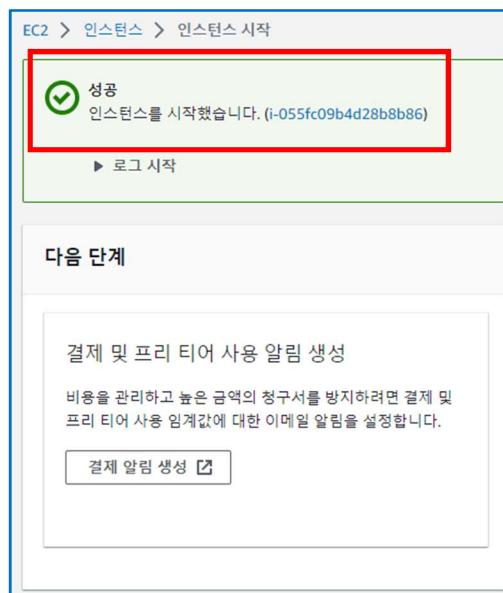
0 x 파일 시스템

편집

14. 페이지 왼쪽에는 지금까지 설정한 내용에 대한 [요약] 섹션이 있다. 확인 후, [인스턴스 시작] 버튼을 클릭하여 새 인스턴스를 생성한다.



15. 인스턴스가 생성되면 해당 링크를 클릭하여 인스턴스 페이지로 이동한다.



16. 인스턴스 목록에서 방금 생성한 **lab1-ubuntu-ec2**를 확인할 수 있다. 현재 [인스턴스 상태]는 실행 중으로, [상태 검사]는 2/2개 검사 통과임을 확인한다.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with options like 'New EC2 Experience', 'EC2 대시보드', 'EC2 글로벌 보기', '이벤트', '태그', '제한', and a expanded '인스턴스' section with '인스턴스' (selected), '인스턴스 유형', '시작 템플릿', '스팟 요청', 'Savings Plans', '예약 인스턴스', '전용 호스트', and '용량 예약'. The main area is titled '인스턴스 (1) 정보' and contains a table with one row. The table columns are 'Name', '인스턴스 ID', '인스턴스 상태', '인스턴스 유형', '상태 검사', and '경고 상태'. The row shows 'lab1-ubuntu-ec2', 'i-055fc09b4d28b8b86', 'Running', 't2.micro', '2/2개 검사 통과', and '경고 없음'. A red box highlights the search bar at the top and the entire row for the instance. Below the table, there's a section titled '인스턴스 선택'.

Name	인스턴스 ID	인스턴스 상태	인스턴스 유형	상태 검사	경고 상태
lab1-ubuntu-ec2	i-055fc09b4d28b8b86	Running	t2.micro	2/2개 검사 통과	경고 없음

Python Flask 설치 및 Hello, World 출력하기

- Linux Server 인스턴스가 정상적으로 생성된 후, 해당 인스턴스 요약 페이지로 이동한다. [인스턴스 상태] 가 실행 중임을 확인한다. 해당 인스턴스와 연결하기 위해 [연결]을 클릭한다.

i-055fc09b4d28b8b86 (lab1-ubuntu-ec2)에 대한 인스턴스 요약 정보
less than a minute 전에 업데이트됨

인스턴스 ID: i-055fc09b4d28b8b86 (lab1-ubuntu-ec2)

IPv6 주소: -

호스트 이름 유형: IP 이름: ip-172-16-14-144.ap-northeast-2.compute.internal

프라이빗 리소스 DNS 이름 등록: IPv4(A)

자동 할당된 IP 주소: 3.34.141.10 [프라이빗 IP]

IAM 역할: -

퍼블릭 IPv4 주소: 3.34.141.10 | 개방 주소법

인스턴스 상태: 실행 중

프라이빗 IP DNS 이름(IPv4만 해당): ip-172-16-14-144.ap-northeast-2.compute.internal

인스턴스 유형: t2.micro

VPC ID: vpc-0d3d2c1f04857e1c3 (lab1-vpc)

서브넷 ID: subnet-04ab0c1e84467e180 (lab1-subnet-public1-ap-northeast-2a)

프라이빗 IPv4 주소: 172.16.14.144

퍼블릭 IPv4 DNS: ec2-3-34-141-10.ap-northeast-2.compute.amazonaws.com | 개방 주소법

탄력적 IP 주소: -

AWS Compute Optimizer 찾기: 권장 사항을 위해 AWS Compute Optimizer에 옵트인합니다. | 자세히 알아보기

Auto Scaling 그룹 이름: -

- [인스턴스에 연결] 페이지에서 [SSH 클라이언트] 탭을 선택한다. 아래 순서 목록 중 4번의 [퍼블릭 DNS 을(를) 사용하여 인스턴스에 연결] 값을 복사한다.

EC2 > 인스턴스 > i-055fc09b4d28b8b86 > 인스턴스에 연결

인스턴스에 연결 정보

다음 옵션 중 하나를 사용하여 인스턴스 i-055fc09b4d28b8b86 (lab1-ubuntu-ec2)에 연결

EC2 인스턴스 연결 Session Manager **SSH 클라이언트** EC2 직렬 콘솔

인스턴스 ID: i-055fc09b4d28b8b86 (lab1-ubuntu-ec2)

1. SSH 클라이언트를 엽니다.

2. 프라이빗 키 파일을 찾습니다. 이 인스턴스를 시작하는 데 사용되는 키는 lab1-ubuntu-ec2.pem입니다.

3. 필요한 경우 이 명령을 실행하여 키를 공개적으로 볼 수 없도록 합니다.
 chmod 400 lab1-ubuntu-ec2.pem

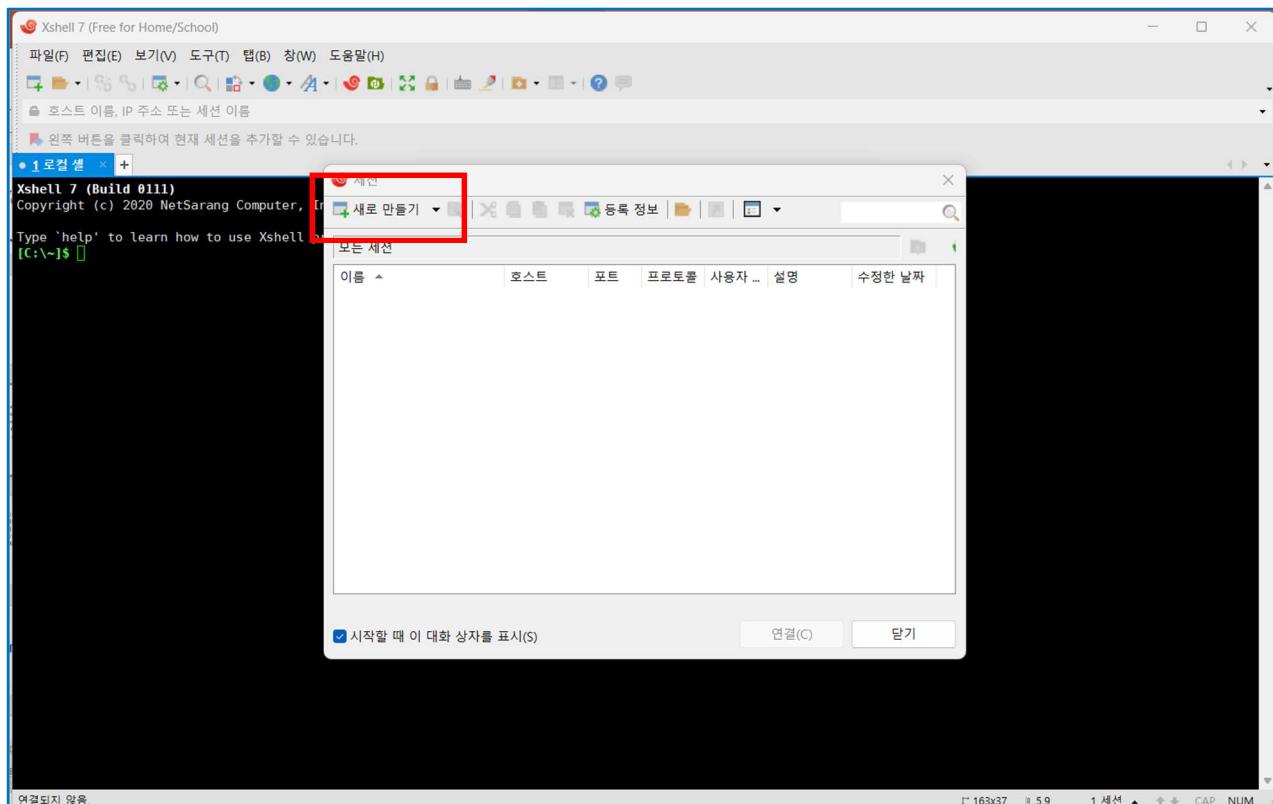
4. 퍼블릭 DNS을(를) 사용하여 인스턴스에 연결:
 ec2-3-34-141-10.ap-northeast-2.compute.amazonaws.com

예:
 ssh -i "lab1-ubuntu-ec2.pem" ubuntu@ec2-3-34-141-10.ap-northeast-2.compute.amazonaws.com

참고: 대부분의 경우 추정된 사용자 이름은 정확합니다. 하지만 AMI 사용 지침을 읽고 AMI 소유자가 기본 AMI 사용자 이름을 변경했는지 확인하십시오.

취소

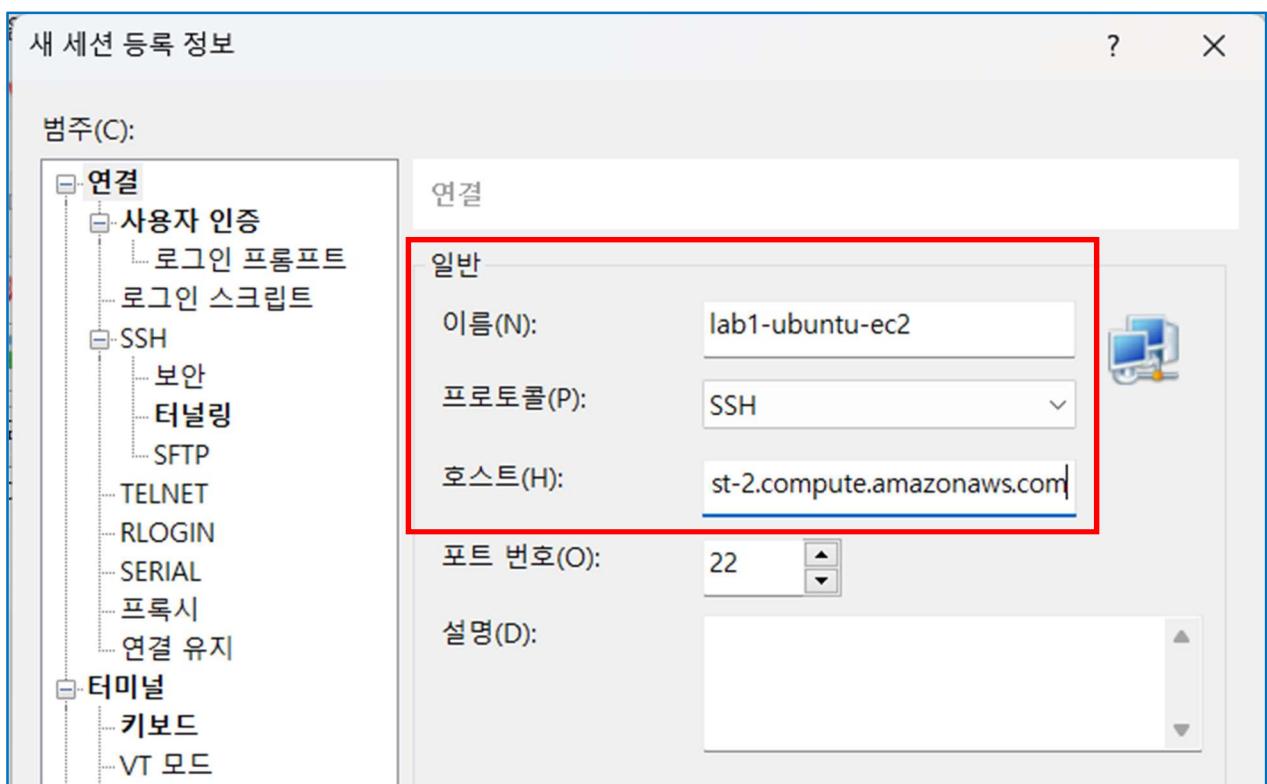
3. XSHELL 프로그램을 시작하면 아래와 같이 [세션] 창이 열린다. [새로 만들기]를 클릭한다.



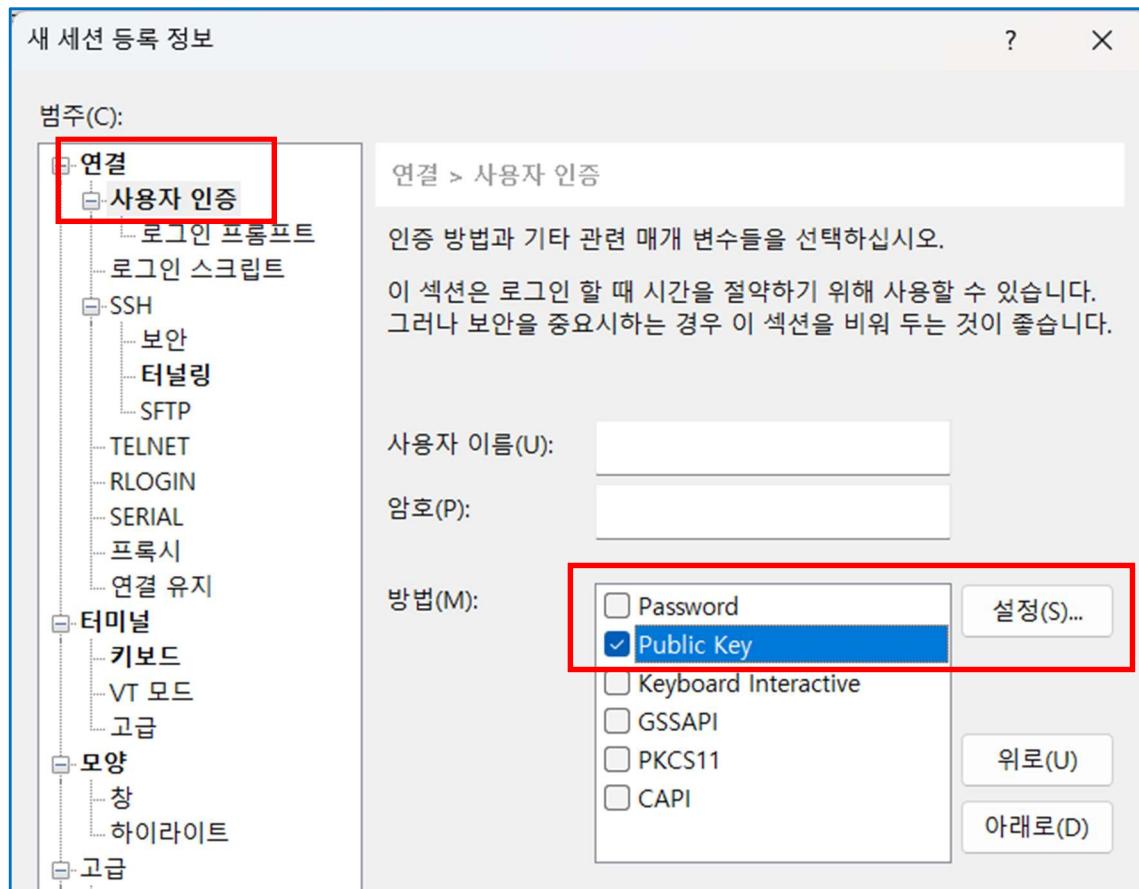
4. [새 세션 등록 정보]창에서 아래와 같이 입력한다.

A. 이름 : lab1-ubuntu-ec2

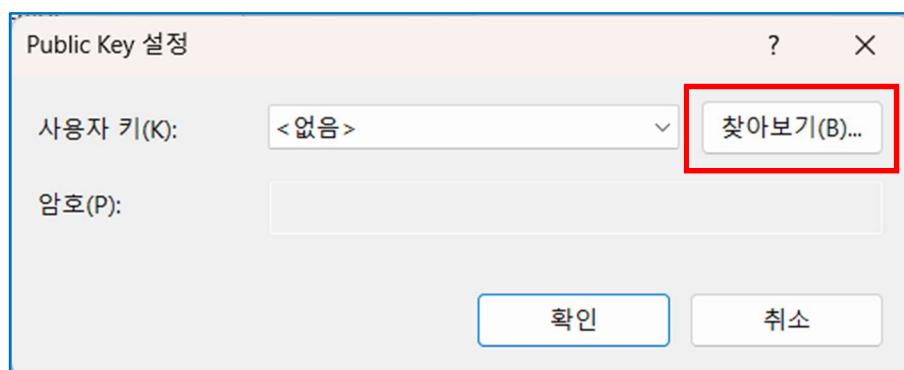
B. 호스트 : 위 2번에서 복사한 퍼블릭 DNS 연결 값



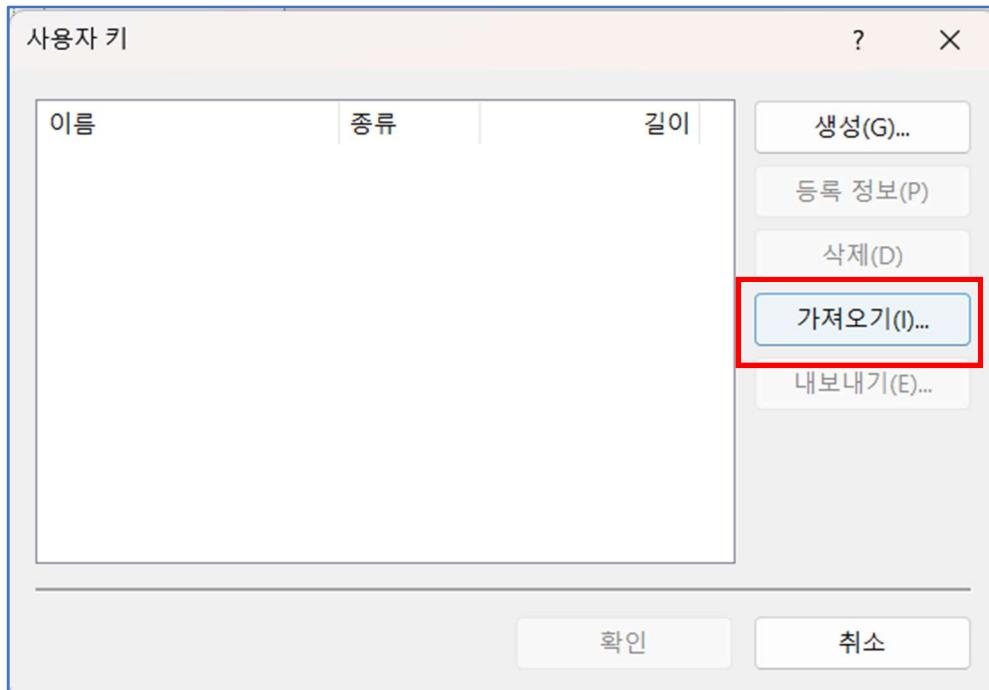
5. 좌측 메뉴 중 [연결] > [사용자 인증]을 선택한다. [방법]에 Public Key를 체크 및 선택하면, [설정] 버튼이 활성화된다. [설정] 버튼을 클릭한다.



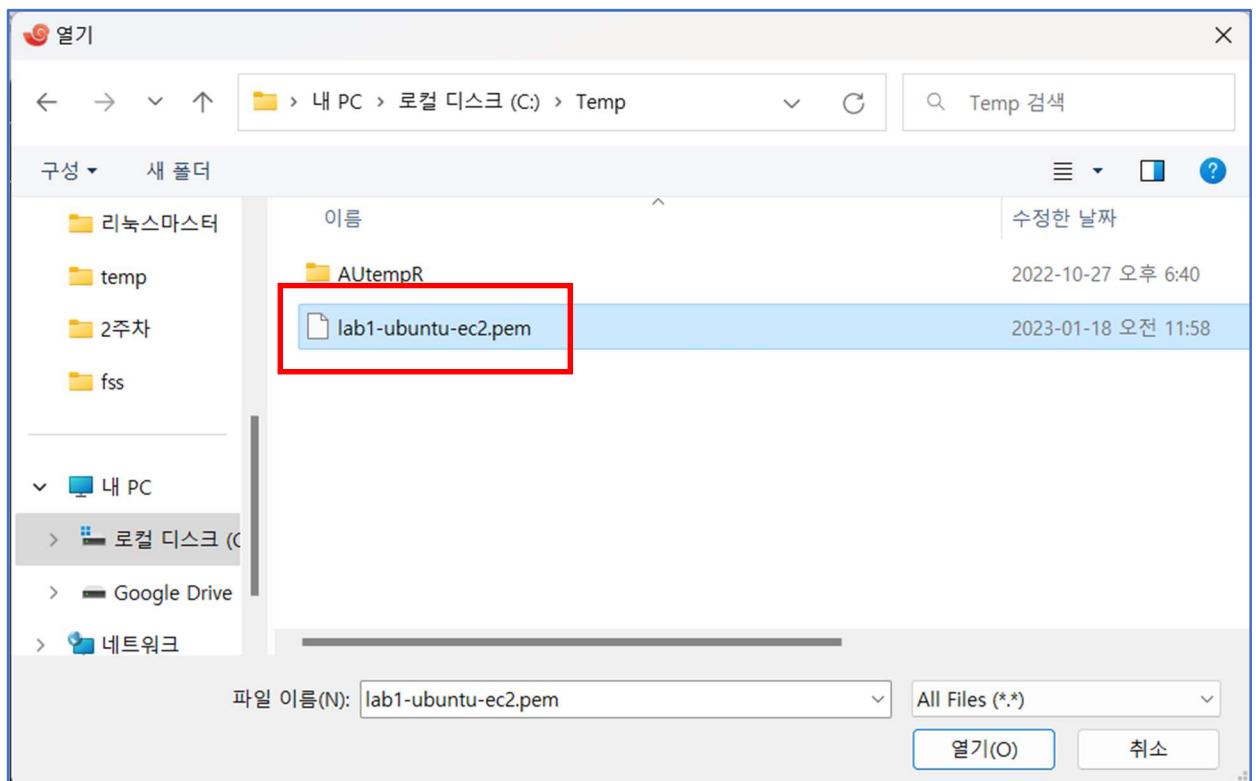
6. [Public Key 설정] 창에서, [찾아보기]를 클릭한다.



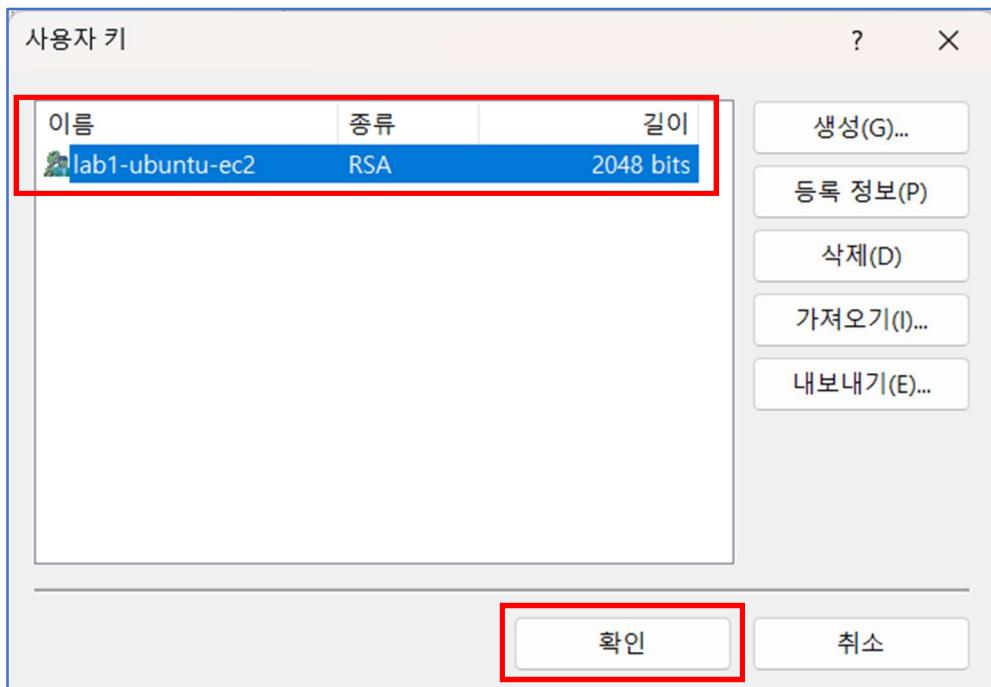
7. [사용자 키] 창에서, [가져오기]를 클릭한다.



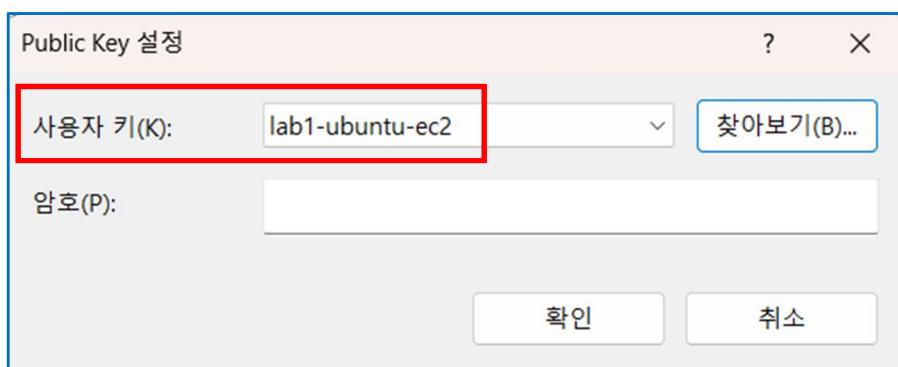
8. [열기] 창에서, 앞에서 이미 .pem 파일을 C:/Temp 폴더에 저장했기 때문에, 해당 폴더로 이동하여 lab1-ubuntu-ec2.pem 파일을 선택하고 [열기]를 클릭한다.



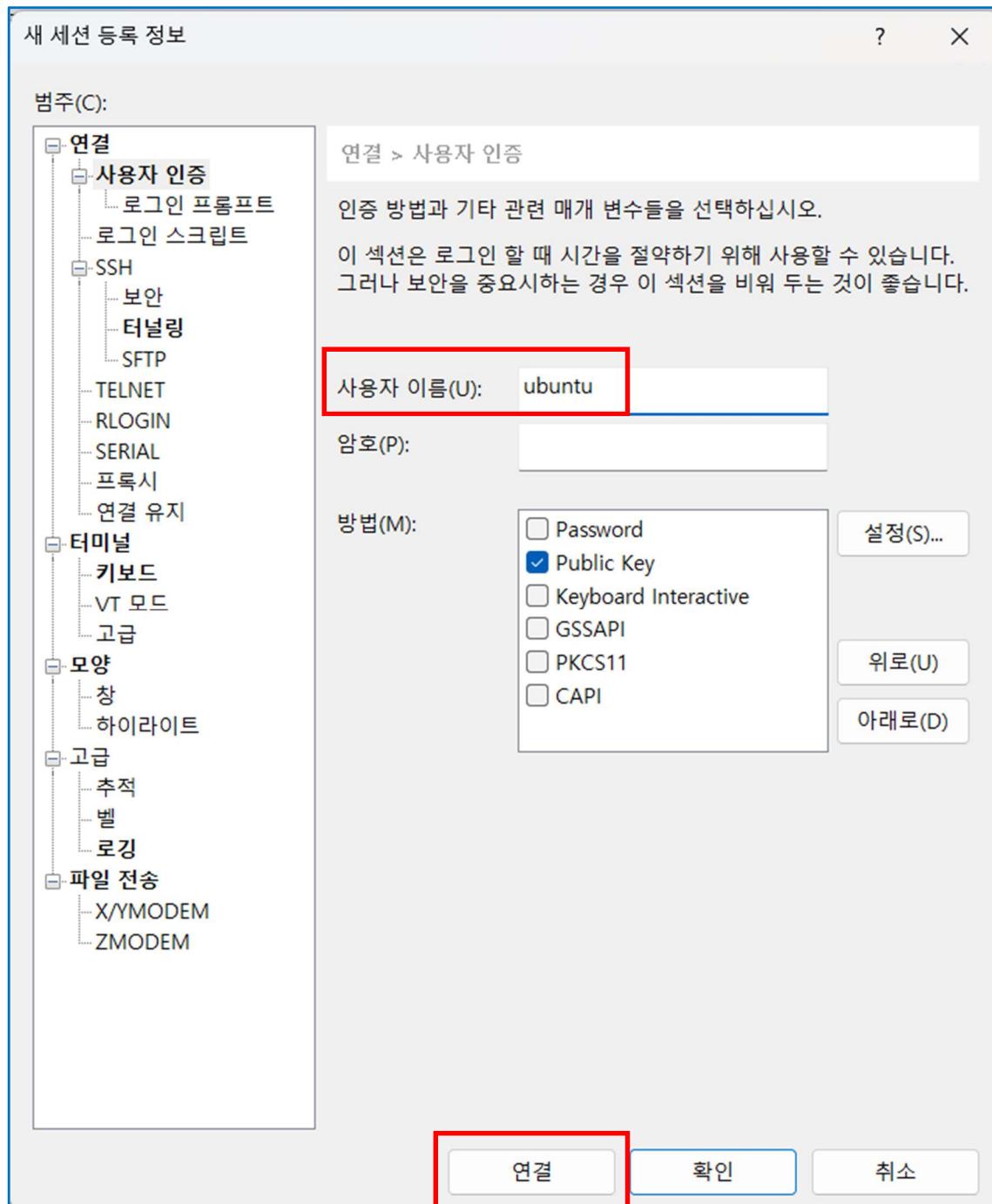
9. [사용자 키] 창에 방금 선택한 lab1-ubuntu-ec2가 보인다. 선택한 후, [확인] 버튼을 클릭한다.



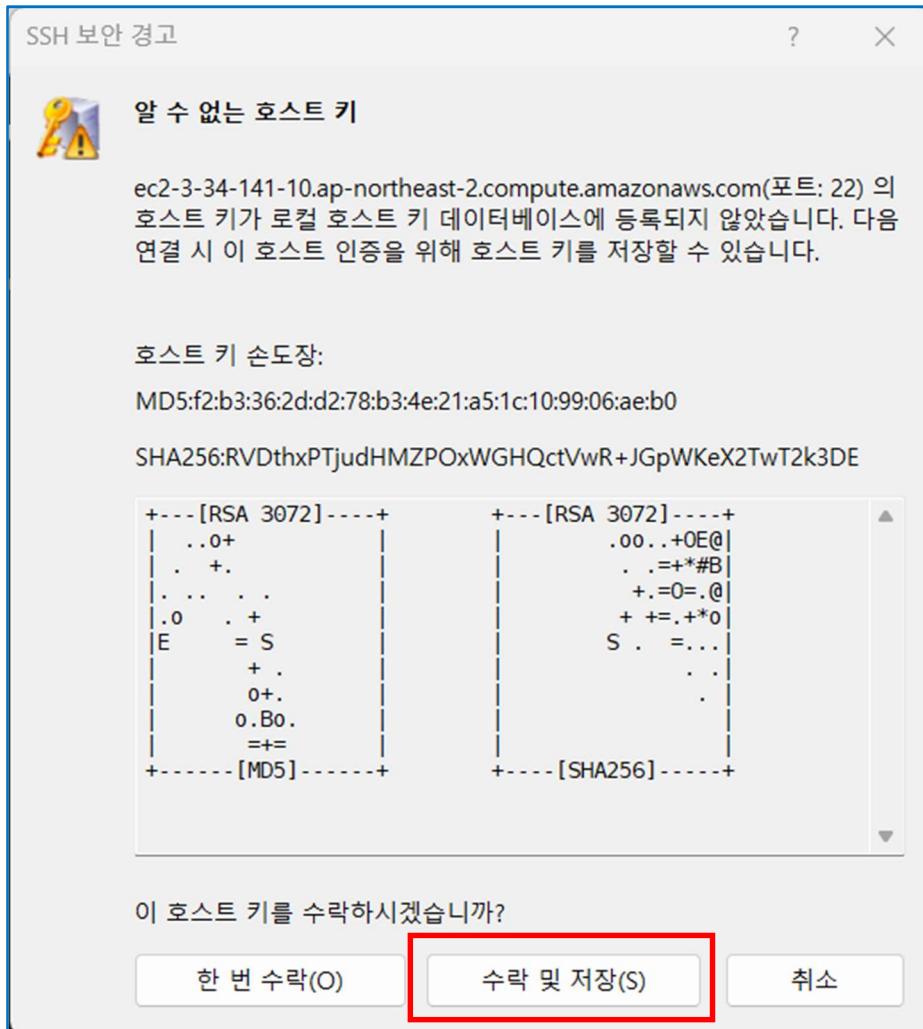
10. [Public Key 설정] 창에서, [사용자 키]를 확인하고 [확인] 버튼을 클릭한다.



11. 다시 [새 세션 등록 정보] 창으로 돌아왔다. [사용자 이름]에 ubuntu를 입력하고 [연결] 버튼을 클릭하여 서버와 연결하도록 한다.



12. 연결이 성공하고, 해당 Key가 맞으면 다음과 같이 [SSH 보안 경고] 창이 나타난다. [수락 및 저장]을 클릭한다.



13. AWS에 위에서 생성한 EC2 인스턴스에 성공적으로 연결되었다.

```
● 1 lab1-ubuntu-ec2 × +  
Host 'ec2-3-34-141-10.ap-northeast-2.compute.amazonaws.com' resolved to 3.34.141.10.  
Connecting to 3.34.141.10:22...  
Connection established.  
To escape to local shell, press 'Ctrl+Alt+]'.  
  
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1026-aws x86_64)  
  
* Documentation: https://help.ubuntu.com  
* Management: https://landscape.canonical.com  
* Support: https://ubuntu.com/advantage  
  
System information as of Wed Jan 18 04:12:46 UTC 2023  
  
System load: 0.080078125 Processes: 97  
Usage of /: 5.2% of 28.89GB Users logged in: 0  
Memory usage: 22% IPv4 address for eth0: 172.16.14.144  
Swap usage: 0%  
  
0 updates can be applied immediately.  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
/usr/bin/xauth: file /home/ubuntu/.Xauthority does not exist  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-16-14-144:~$ [red box]  
ssh://ubuntu@ec2-3-34-141-10.ap-northeast-2.compute.amazonaws.com:22
```

14. 다음과 같은 명령어로 인터넷 연결 및 Cache 업데이트를 수행한다.

```
$ sudo apt update
```

```
1 lab1-ubuntu-ec2 × +  
ubuntu@ip-172-16-14-144:~$ sudo apt update  
  
Get:21 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [2448 B]  
Get:22 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [432 B]  
Get:23 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [3324 B]  
Get:24 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [1580 B]  
Get:25 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [272 B]  
Get:26 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-backports/restricted amd64 c-n-f Metadata [116 B]  
Get:27 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [6744 B]  
Get:28 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe Translation-en [9460 B]  
Get:29 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 c-n-f Metadata [352 B]  
Get:30 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy-backports/multiverse amd64 c-n-f Metadata [116 B]  
Get:31 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [582 kB]  
Get:32 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [122 kB]  
Get:33 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [7588 B]  
Get:34 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [518 kB]  
Get:35 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [79.7 kB]  
Get:36 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [556 B]  
Get:37 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [627 kB]  
Get:38 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [83.9 kB]  
Get:39 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [11.0 kB]  
Get:40 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [4268 B]  
Get:41 http://security.ubuntu.com/ubuntu jammy-security/multiverse Translation-en [972 B]  
Get:42 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 c-n-f Metadata [228 B]  
Fetched 25.4 MB in 4s (6023 kB/s)  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
48 packages can be upgraded. Run 'apt list --upgradable' to see them.  
ubuntu@ip-172-16-14-144:~$
```

15. 먼저 **Flask Library**를 설치하기 전에 **Ubuntu 22.04** 인스턴스에 설치되어 있는 **Python**버전을 확인해본다.

현재 3.10.6 버전이 설치되어 있음을 확인할 수 있다.

```
$ python -V
```

```
$ python3 --version
```

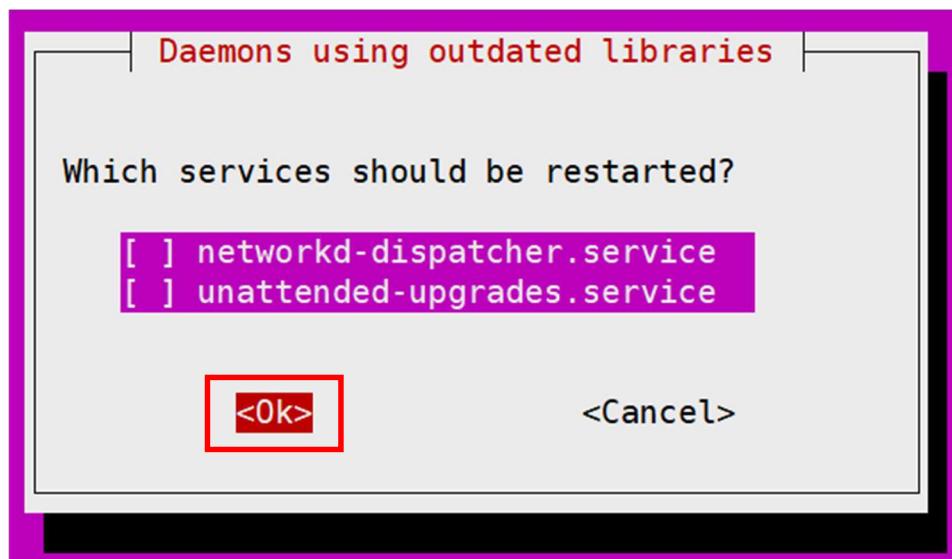
```
1 lab1-ubuntu-ec2 × +  
ubuntu@ip-172-16-14-144:~$ python -V  
Command 'python' not found, did you mean:  
  command 'python3' from deb python3  
  command 'python' from deb python-is-python3  
ubuntu@ip-172-16-14-144:~$ python3 --version  
Python 3.10.6  
ubuntu@ip-172-16-14-144:~$
```

16. **pip**와 **Python** 가상환경을 설치한다.

```
$ sudo apt install python3-venv pip -y
```

```
● 1 lab1-ubuntu-ec2 x +  
ubuntu@ip-172-16-14-144:~$ sudo apt install python3-venv pip -y  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
Note, selecting 'python3-pip' instead of 'pip'  
The following additional packages will be installed:  
  build-essential bzip2 cpp cpp-11 dpkg-dev fakeroot fontconfig-config fo  
  gcc-11-base javascript-common libalgorithm-diff-perl libalgorithm-diff-  
  libatomic1 libc-dev-bin libc-devtools libc6-dev libcc1-0 libcrypt-dev l  
  libfakeroot libfile-fcntllock-perl libfontconfig1 libgcc-11-dev libgd3  
  libjpeg-turbo8 libjpeg8 libjs-jquery libjs-sphinxdoc libjs-underscore l
```

17. 혹시 다음 그림과 같은 창이 나타나면 **Tab** 키를 눌러서 <Ok>로 이동한 후 엔터키를 누른다.



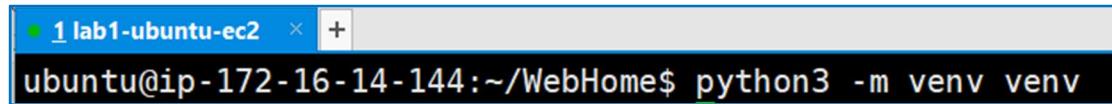
18. **WebHome** 디렉토리를 생성하고, 해당 디렉토리로 이동한다 .

```
$ mkdir WebHome && cd WebHome
```

```
● 1 lab1-ubuntu-ec2 x +  
ubuntu@ip-172-16-14-144:~$ mkdir WebHome && cd WebHome  
ubuntu@ip-172-16-14-144:~/WebHome$
```

19. 다음의 명령으로 새 가상 환경을 생성한다.

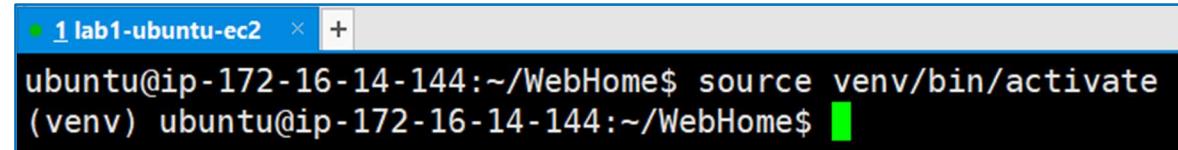
```
$ python3 -m venv venv
```



```
● 1 lab1-ubuntu-ec2 ✘
ubuntu@ip-172-16-14-144:~/WebHome$ python3 -m venv venv
```

20. 가상 환경이 생성되면 다음과 같은 명령으로 활성화를 한다.

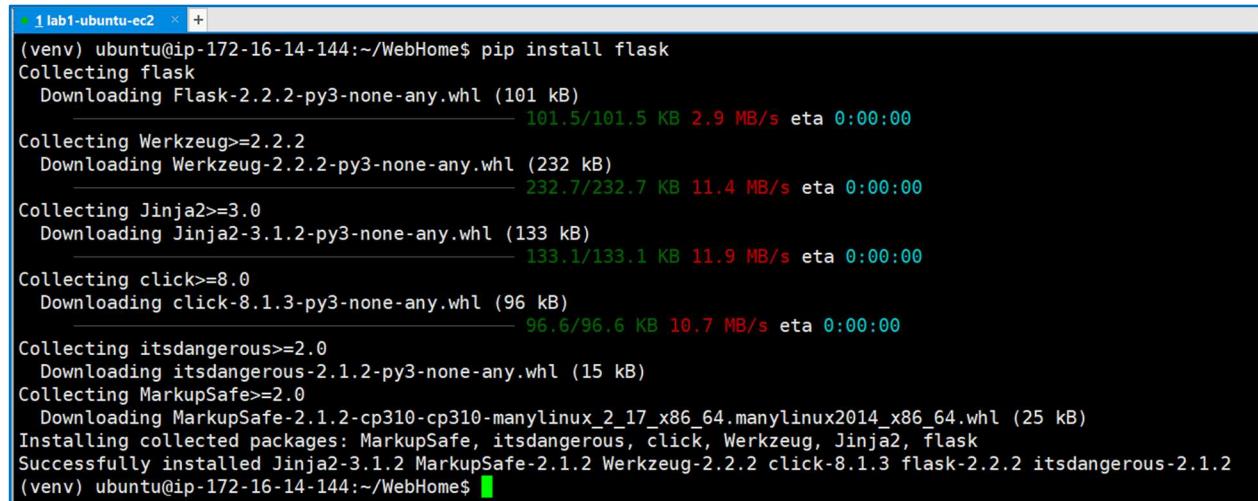
```
$ source venv/bin/activate
```



```
● 1 lab1-ubuntu-ec2 ✘
ubuntu@ip-172-16-14-144:~/WebHome$ source venv/bin/activate
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ █
```

21. 가상 환경에서 Flask Library를 설치한다.

```
$ pip install flask
```

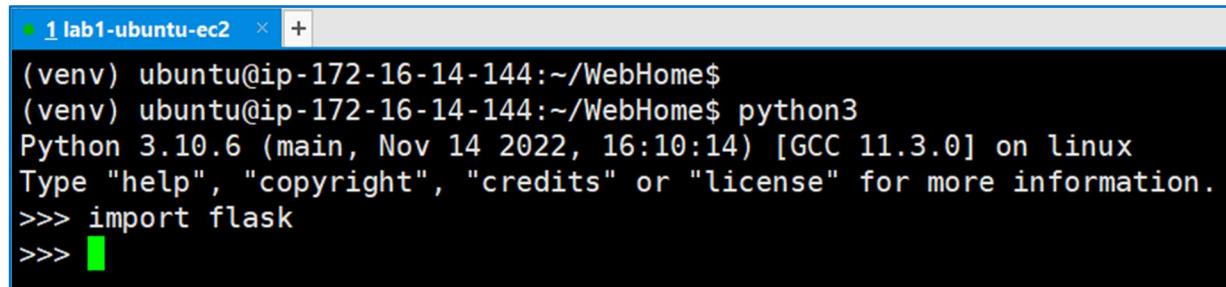


```
● 1 lab1-ubuntu-ec2 ✘
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ pip install flask
Collecting flask
  Downloading Flask-2.2.2-py3-none-any.whl (101 kB)
    101.5/101.5 KB 2.9 MB/s eta 0:00:00
Collecting Werkzeug>=2.2.2
  Downloading Werkzeug-2.2.2-py3-none-any.whl (232 kB)
    232.7/232.7 KB 11.4 MB/s eta 0:00:00
Collecting Jinja2>=3.0
  Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
    133.1/133.1 KB 11.9 MB/s eta 0:00:00
Collecting click>=8.0
  Downloading click-8.1.3-py3-none-any.whl (96 kB)
    96.6/96.6 KB 10.7 MB/s eta 0:00:00
Collecting itsdangerous>=2.0
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-2.1.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25 kB)
Installing collected packages: MarkupSafe, itsdangerous, click, Werkzeug, Jinja2, flask
Successfully installed Jinja2-3.1.2 MarkupSafe-2.1.2 Werkzeug-2.2.2 click-8.1.3 flask-2.2.2 itsdangerous-2.1.2
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ █
```

22. Flask Library 설치를 마치면 잘 설치되었는지 확인하기 위해 **Python shell**을 시작한다. 그리고 다음과 같이 **flask**를 **import**한다. 아무 에러메시지가 출력되지 않음을 확인한다.

```
$ python3
```

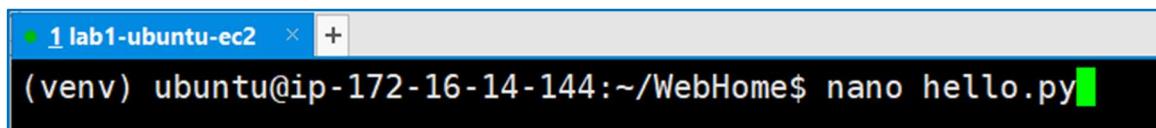
```
>>> import flask
```



```
● 1 lab1-ubuntu-ec2 ✘
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ 
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ python3
Python 3.10.6 (main, Nov 14 2022, 16:10:14) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import flask
>>> █
```

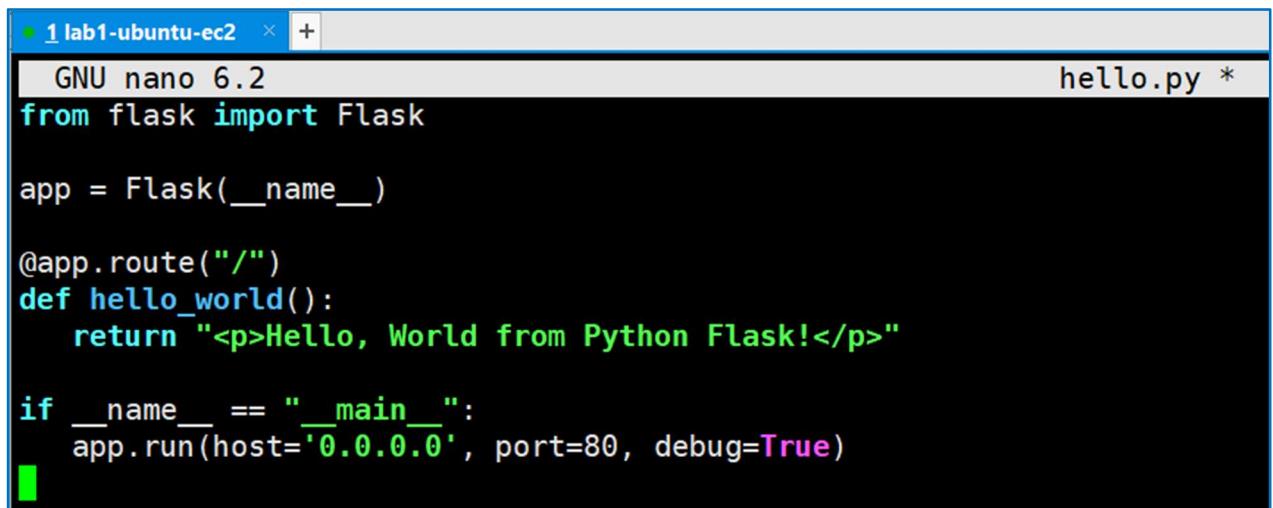
23. Ubuntu의 파일 편집기인 **nano**를 이용해서 **hello.py** 파일을 생성한다.

```
$ nano hello.py
```



```
● 1 lab1-ubuntu-ec2 × +  
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ nano hello.py
```

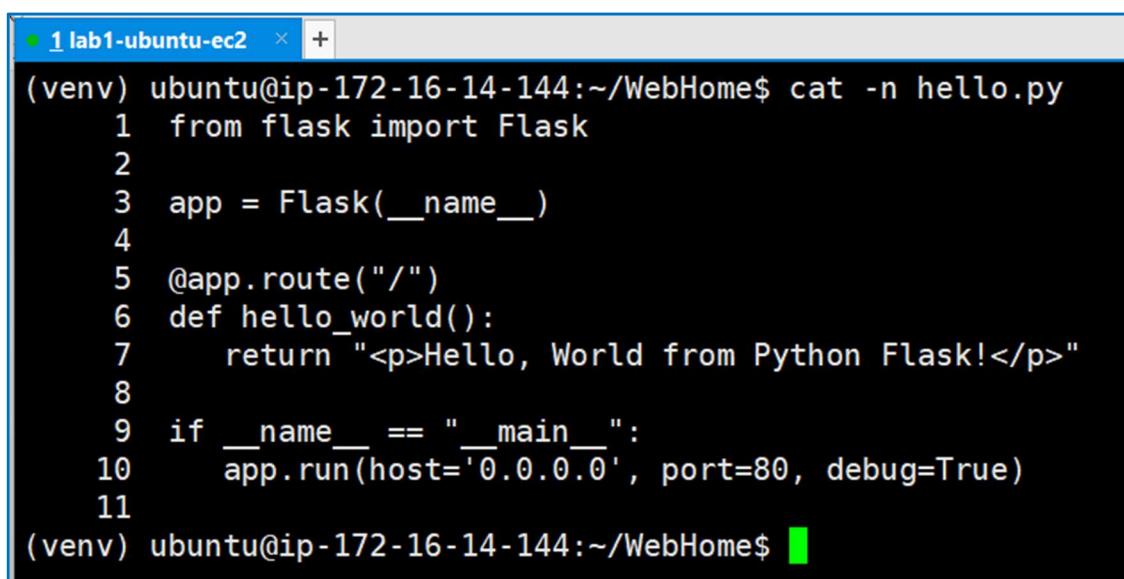
24. 다음과 같이 입력하고 **Ctrl + O**를 눌러 저장하고 **Ctrl + X**를 눌러 **nano** 편집기를 닫는다.



```
● 1 lab1-ubuntu-ec2 × +  
GNU nano 6.2                                     hello.py *  
from flask import Flask  
  
app = Flask(__name__)  
  
@app.route("/")  
def hello_world():  
    return "<p>Hello, World from Python Flask!</p>"  
  
if __name__ == "__main__":  
    app.run(host='0.0.0.0', port=80, debug=True)
```

25. 잘 작성되었는지 확인한다.

```
$ cat -n hello.py
```



```
● 1 lab1-ubuntu-ec2 × +  
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ cat -n hello.py  
1  from flask import Flask  
2  
3  app = Flask(__name__)  
4  
5  @app.route("/")  
6  def hello_world():  
7      return "<p>Hello, World from Python Flask!</p>"  
8  
9  if __name__ == "__main__":  
10     app.run(host='0.0.0.0', port=80, debug=True)  
11  
(venv) ubuntu@ip-172-16-14-144:~/WebHome$
```

26. 하지만 **Flask**는 기본포트가 **5000**번이기 때문에 **80**번으로 포트를 변경하려면 몇 가지 추가 작업을 해야 한다. 먼저 가상환경을 중지한다.

```
$ deactivate
```

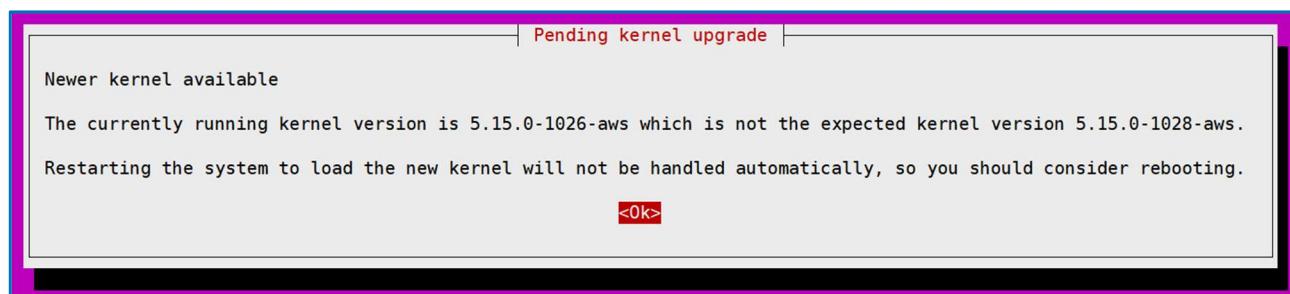
```
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ deactivate  
ubuntu@ip-172-16-14-144:~/WebHome$ █
```

27. **Flask** 포트를 **80**번으로 오픈하기 위해 다음과 같이 **authbind** 패키지를 설치한다.

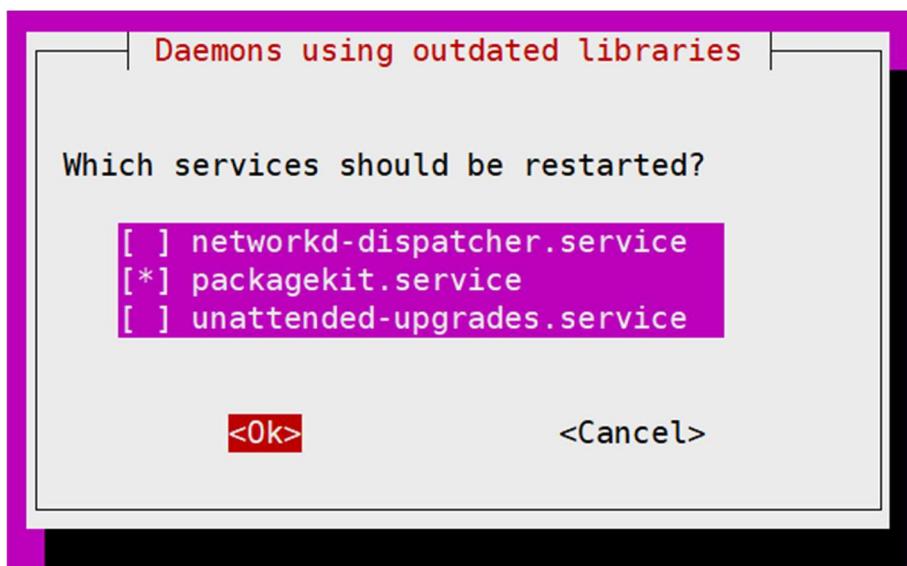
```
$ sudo apt install authbind
```

```
ubuntu@ip-172-16-14-144:~/WebHome$ sudo apt install authbind  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following NEW packages will be installed:  
  authbind  
0 upgraded, 1 newly installed, 0 to remove and 28 not upgraded.  
Need to get 18.7 kB of archives.  
After this operation, 78.8 kB of additional disk space will be used.  
Get:1 http://ap-northeast-2.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 authbind amd64 2.1.3build1 [18.7 kB]  
Fetched 18.7 kB in 0s (71.5 kB/s)  
Selecting previously unselected package authbind.
```

28. 설치 도중 다음과 같은 창이 나타나면 <Ok>를 선택되어 있음을 확인하고 엔터키를 누른다.



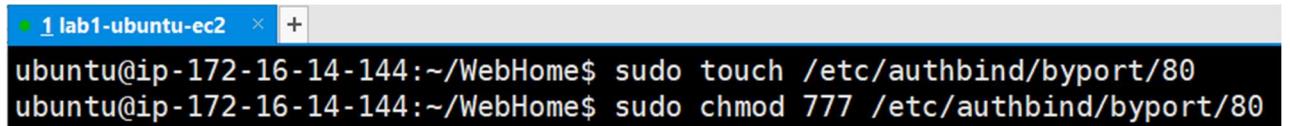
29. 다음과 같은 창이 나타나면 **Tab** 키를 이용하여 <Ok>에 맞춘 후, 엔터키를 누른다.



30. **authbind** 패키지 설치 후 다음과 같은 작업을 수행한다.

```
$ sudo touch /etc/authbind/byport/80
```

```
$ sudo chmod 777 /etc/authbind/byport/80
```

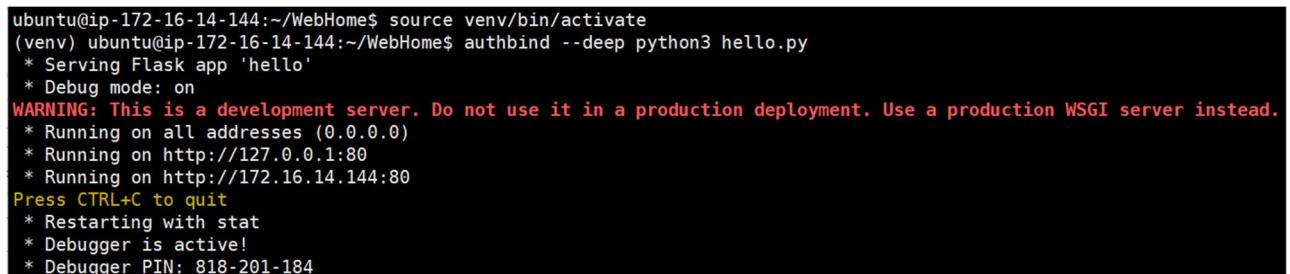


```
ubuntu@ip-172-16-14-144:~/WebHome$ sudo touch /etc/authbind/byport/80
ubuntu@ip-172-16-14-144:~/WebHome$ sudo chmod 777 /etc/authbind/byport/80
```

31. 다시 **Python** 가상환경을 시작한다. 그리고 **authbind**를 이용하여 **hello.py**를 실행한다.

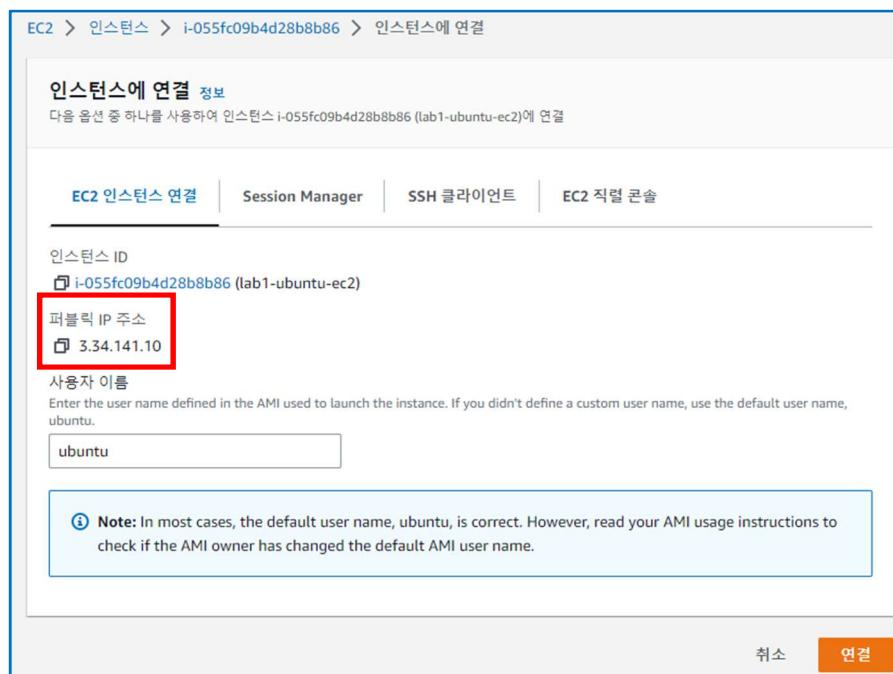
```
$ source venv/bin/activate
```

```
$ authbind --deep python3 hello.py
```



```
ubuntu@ip-172-16-14-144:~/WebHome$ source venv/bin/activate
(venv) ubuntu@ip-172-16-14-144:~/WebHome$ authbind --deep python3 hello.py
* Serving Flask app 'hello'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:80
* Running on http://172.16.14.144:80
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 818-201-184
```

32. **Flask Web Server**가 정상적으로 실행되었다. Ubuntu 인스턴스의 [퍼블릭 IP 주소]를 복사한다.



33. 이제 복사한 [퍼블릭 IP 주소]를 브라우저에 붙여 넣는다. `hello.py`에 코딩한 대로, 해당 글자가 제대로 출력되는 것을 확인할 수 있다.

