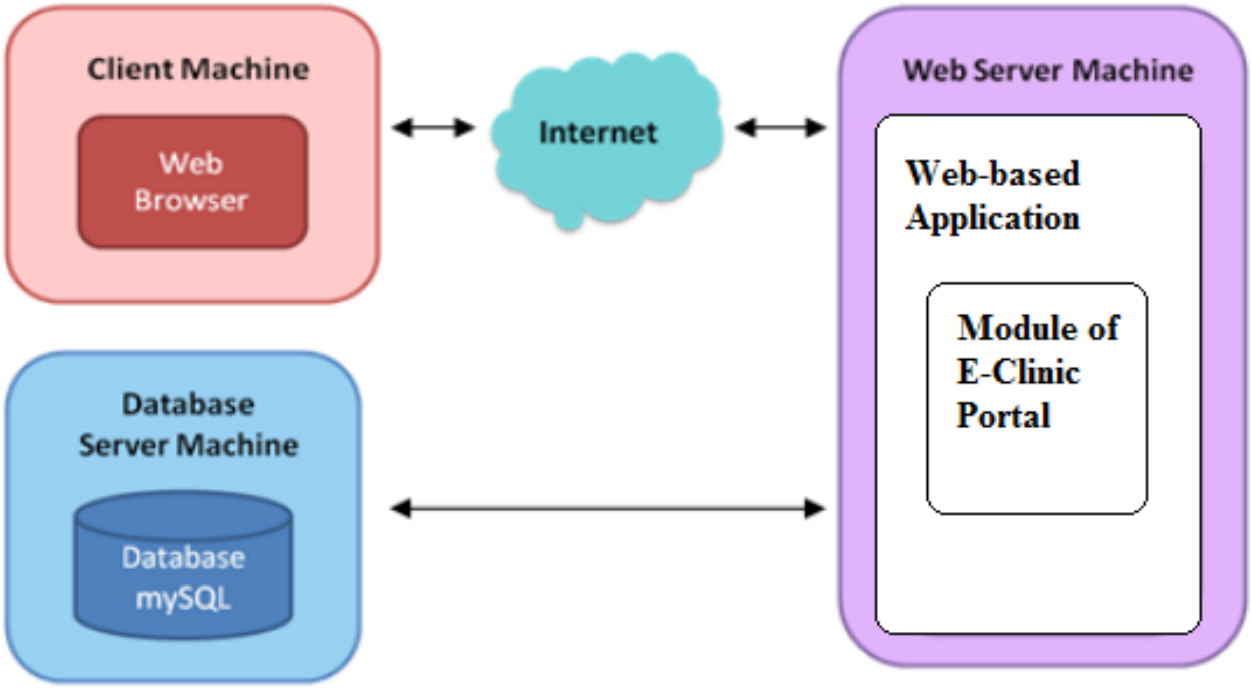
# Cloud Architecture Explained: Types and Use Cases

https://svitla.com/blog/cloud-architecture-explained/

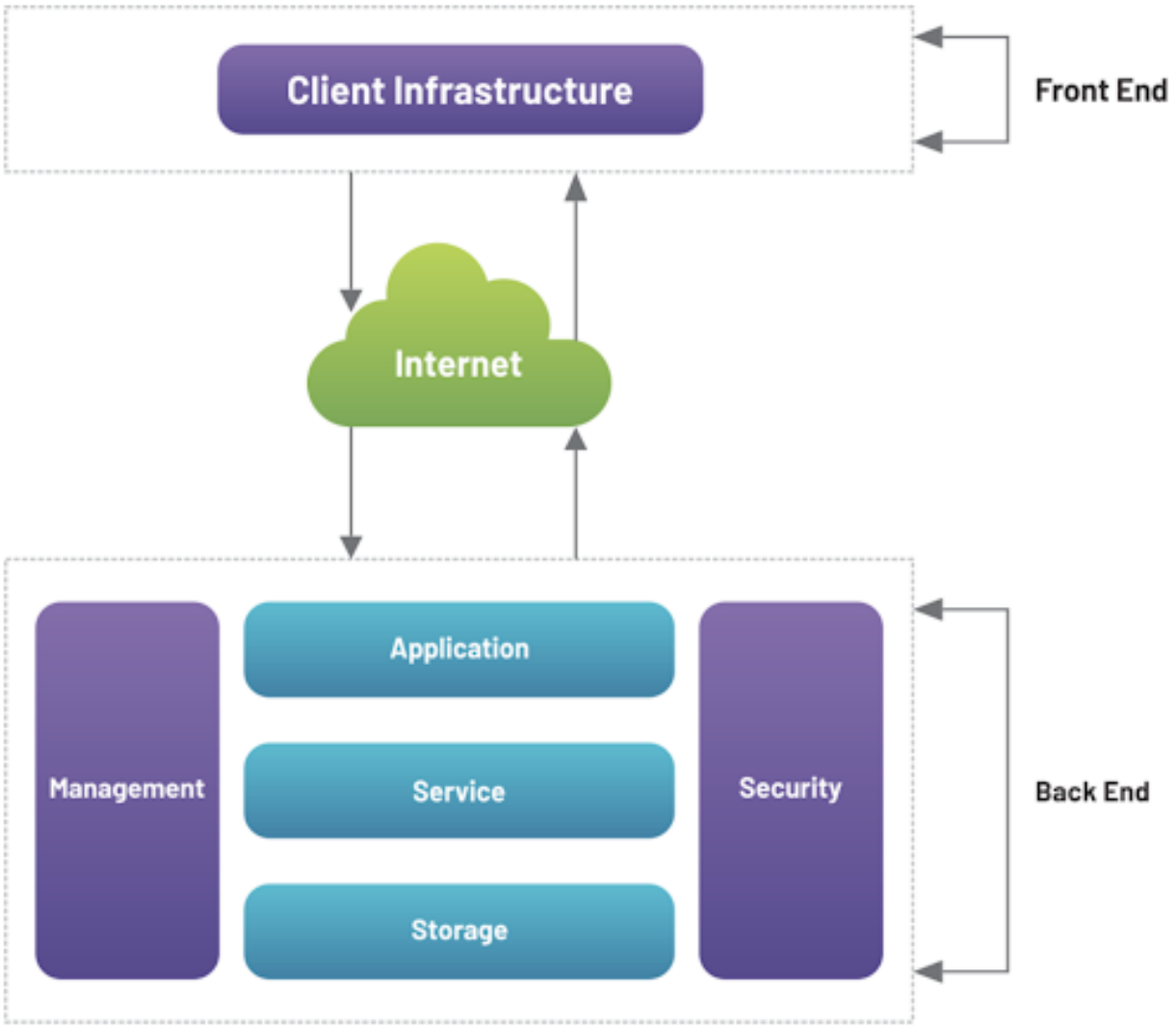
- Every organization now uses cloud services to access or store information that is easily accessible always, from anywhere in the world via the Internet.
- The cloud architecture umbrella pools multiple technologies, making them easily accessible to everyone from everywhere.
- **It can be used in private, public, and hybrid clouds, and it is cheap, flexible, and easy to scale**.

# System Architecture Diagram of the Web-based System




System Architecture Diagram of the Web-based System | Download Scientific Diagram (researchgate.net)

# A traditional cloud framework



| Feature | Cloud Software | Web Software |
|---|---|---|
| **Definition** | Applications hosted on cloud infrastructure, accessed via the internet | Applications accessed via a web browser, typically hosted on web servers |
| **Hosting** | Managed by cloud service providers like AWS, Azure 아/져, or Google Cloud | Can be hosted on traditional servers or cloud servers |
| **Scalability** | Highly scalable, can adjust resources dynamically based on demand | Generally less scalable compared to cloud-native apps |
| **Accessibility** | Accessible from any device with an internet connection | Requires a web browser and internet connection |
| **Maintenance** | Providers handle updates, security, and maintenance | Managed by the service provider or IT team |
| **Examples** | Google Drive, Dropbox, Salesforce | Gmail, Facebook, LinkedIn |

- **Front-end**.

- The collection of elements a client interacts with is the user interface and application a person uses to get to the cloud services.

- **Back-end**.

- The cloud that’s leveraged by the cloud provider that holds, manages, and secures resources. In addition to this, it contains storage, virtual machines, ways to control traffic, deployment models, and so on.
-

Python for the back-end, import flask framework.

Flask is an open-source web framework for Python that is designed to make it easier to build web applications.

## Application coding

- import the necessary libraries and modules.
- set up an application by initializing the Flask app.
- use flask to manage sessions securely.

## Code for cloud security

- **Practice and tools to employ diverse security mechanisms to keep cloud resources, systems, and files, safe for end users**

**User Authentication and JWT**

Users log in by providing a username and password

On successful login, a JWT token is generated and returned

The token includes user data and expiration time ensuring it is used within a secure timeframe

**Token verification**

The token_required decorator checks if the token is present and valid

It decodes the token to verify the user's identity and role

**Role-Based Access Control (RBAC)**:

The role_required decorator checks user roles before

allowing access to specific endpoints.

**Secure Password Management**:

Passwords are hashed using bcrypt to enhance security.

Building Applications in the Cloud: A Step-By-Step Guide (codewave.com)

**https://codewave.com/insights/application-development-for-the-cloud-guide/**

# Types of Cloud Applications

Understanding the different types of cloud applications is key to selecting the right solution for your needs. Here's a breakdown of the main types:

## 1. Software-as-a-Service (SaaS)

SaaS delivers software applications over the internet. Users can access these applications through a web browser, eliminating the need for local installation and maintenance.

**Examples:**

- **Google Workspace:** Provides a suite of productivity tools such as Gmail, Google Docs, and Google Drive.
- **Salesforce:** A comprehensive CRM platform offering tools for sales, customer service, and marketing.

## 2. Platform-as-a-Service (PaaS)

PaaS offers a platform that allows developers to build, deploy, and manage applications without worrying about the underlying infrastructure. It provides development tools, database management, and middleware.

**Examples:**

- **Heroku:** Facilitates building, running, and scaling applications with ease.
- **Google App Engine:** Enables developers to build and deploy applications on Google’s infrastructure.

## 3. Infrastructure-as-a-Service (IaaS)

IaaS provides virtualized computing resources over the internet, including virtual machines, storage, and networking. It offers the fundamental building blocks for computing.

- **Amazon Web Services (AWS) EC2:** Provides scalable virtual servers.
- **Microsoft Azure 아/져 Virtual Machines:** Delivers scalable computing resources in the cloud.

## 4. Serverless Architecture

Serverless architecture, also known as Function-as-a-Service (FaaS), allows developers to execute code in response to events without managing servers. It abstracts the underlying infrastructure and automatically scales with usage.

- **AWS Lambda:** Runs code in response to events such as database changes or HTTP requests.
- **Azure Functions:** Provides event-driven compute services with automatic scaling.

# How to use cloud for developing application

## Develop the Application with the Right Tools

Use appropriate programming languages like **JavaScript, Python, or Java, and frameworks** such as React or Node.js to build your application's front-end and back-end components

## Connect your existing web system to the AWS cloud

The process of connecting an existing web system to the AWS cloud and installing it consists of several steps, and each step requires the correct settings.

## Software-as-a-Service (SaaS)

SaaS delivers software applications over the internet. Users can access these applications through a web browser, eliminating the need for local installation and maintenance.

**Examples:** Use current cloud service

- **Google Workspace:** Provides a suite of productivity tools such as Gmail, Google Docs, and Google Drive.
- **Salesforce:** A comprehensive CRM platform offering tools for sales, customer service, and marketing.

# Connect your existing web system to the AWS cloud

The process of connecting an existing web system to the AWS cloud and installing it consists of several steps, and each step requires the correct settings.

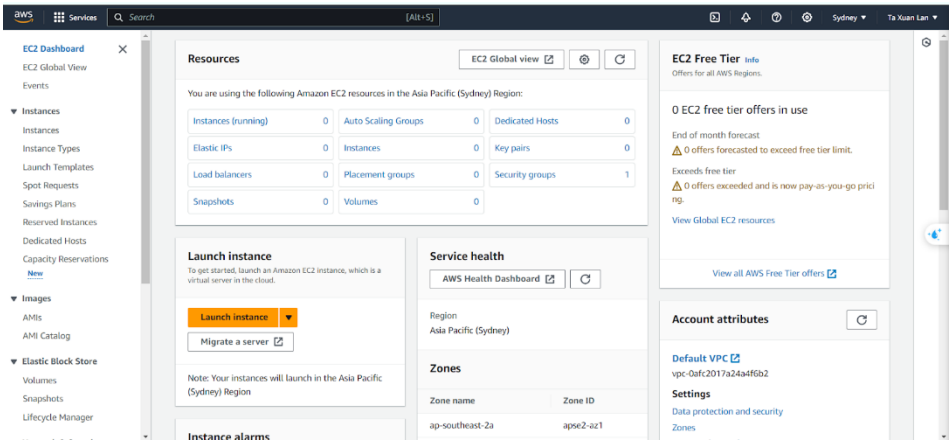
# Example How to connect to AWS

[Type2] Install your cloud environment

[Step 1]: Set up your AWS account

**Step 1: Launch an instance**

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/




2. From the EC2 console dashboard, in the Launch instance box, choose Launch instance.

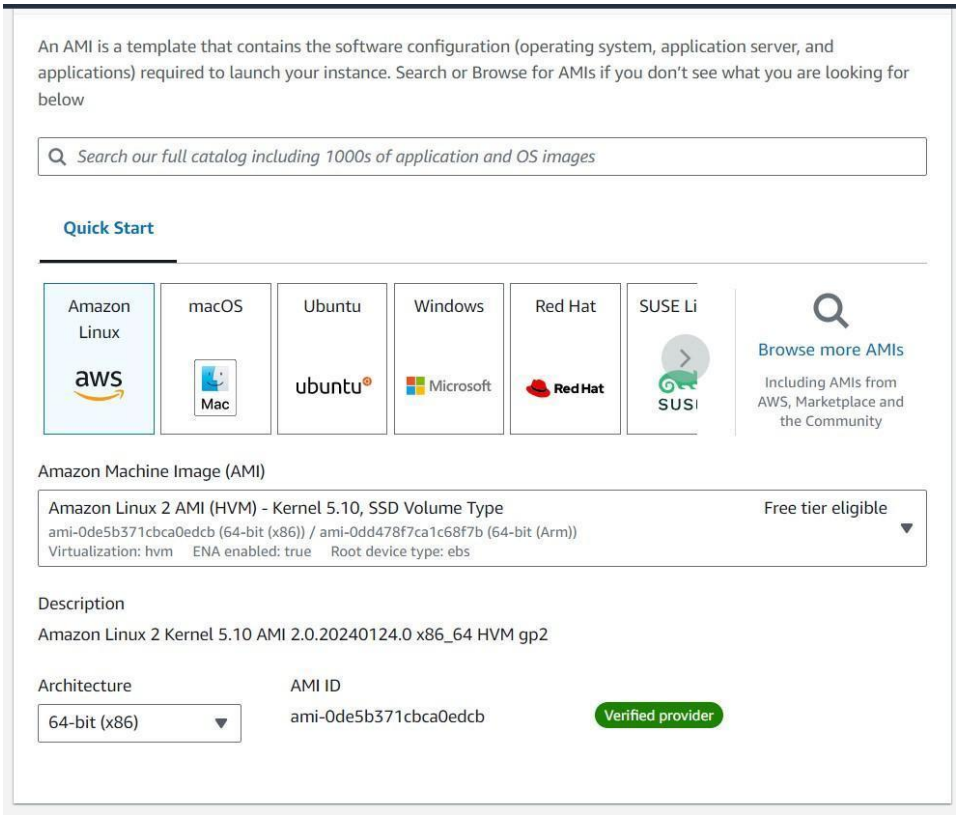
3. Under Name and tags, for Name, enter a descriptive name for your instance.


4. Under Application and OS Images (Amazon Machine Image), do the following:

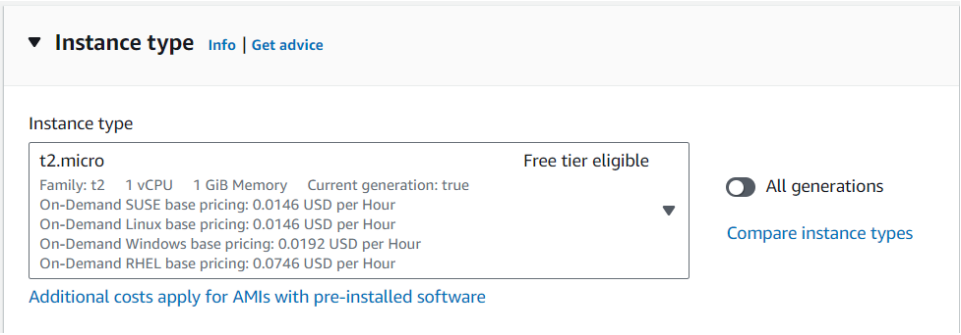
   a. Choose Quick Start, and then choose Amazon Linux. This is the operating system (OS) for your instance.

   b. From Amazon Machine Image (AMI), select an HVM version of Amazon Linux 2. Notice that these AMIs are marked Free Tier eligible. An Amazon Machine Image (AMI)

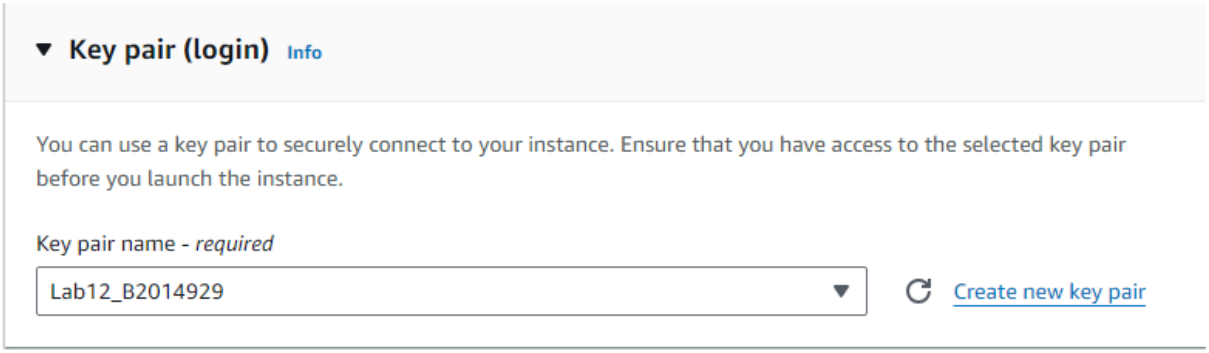
is a basic configuration that serves as a template for your instance.




5. Under **Instance type**, from the **Instance type** list, you can select the hardware configuration for your instance. Choose the t2.micro instance type, which is selected by default.
The t2.micro instance type is eligible for the Free Tier. In Regions where t2.micro is unavailable, you can use a t3.micro instance under the Free Tier. For more information, see [AWS Free Tier]



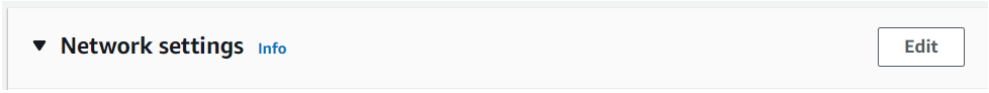
6. Under **Key pair (login)**, for **Key pair name**, choose the key pair that you created when getting set up

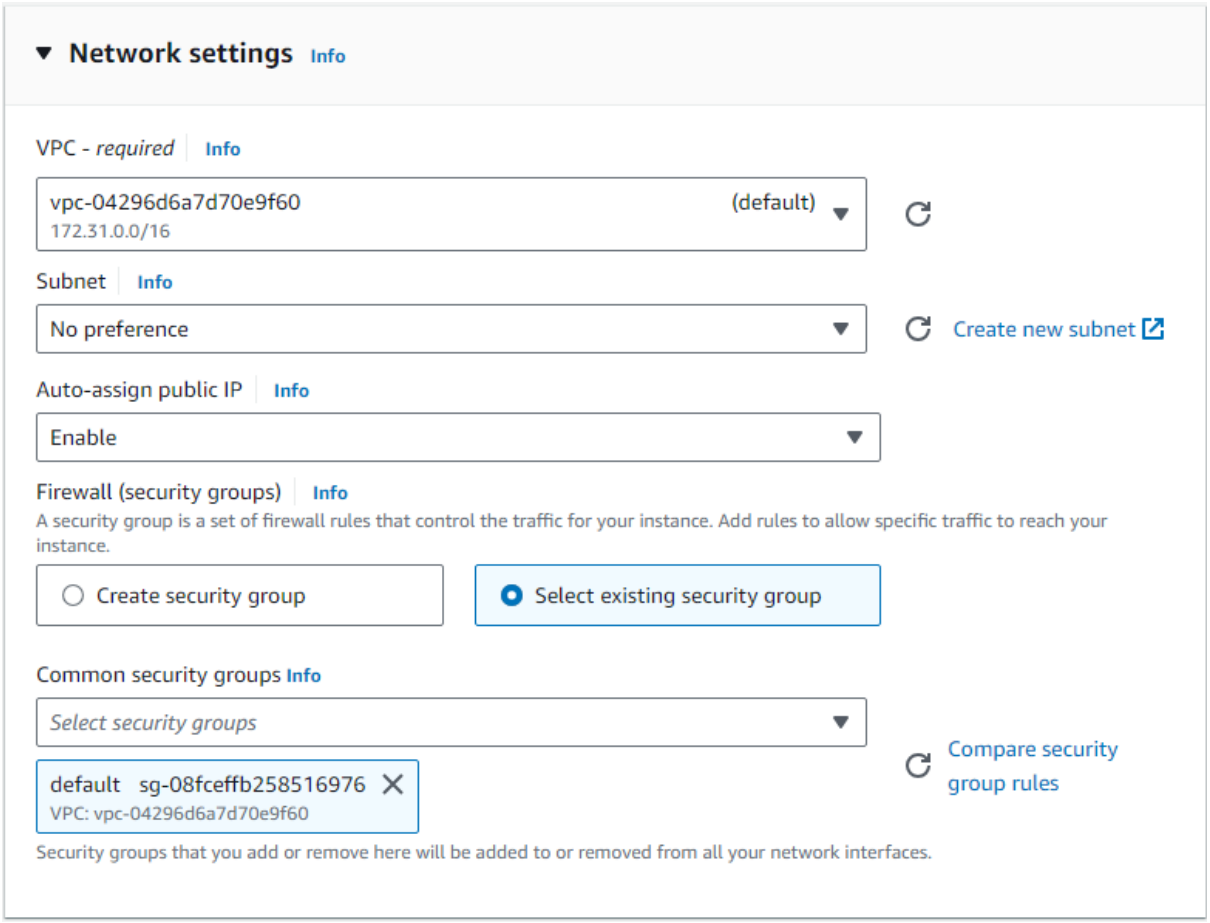



7. Next to **Network settings**, choose **Edit**. For **Security group name**, you'll see that the wizard created and selected a security group for you. You can use this security group, or alternatively you can select the security group that you created when getting set up using the following steps:

a. Choose Select existing security group.

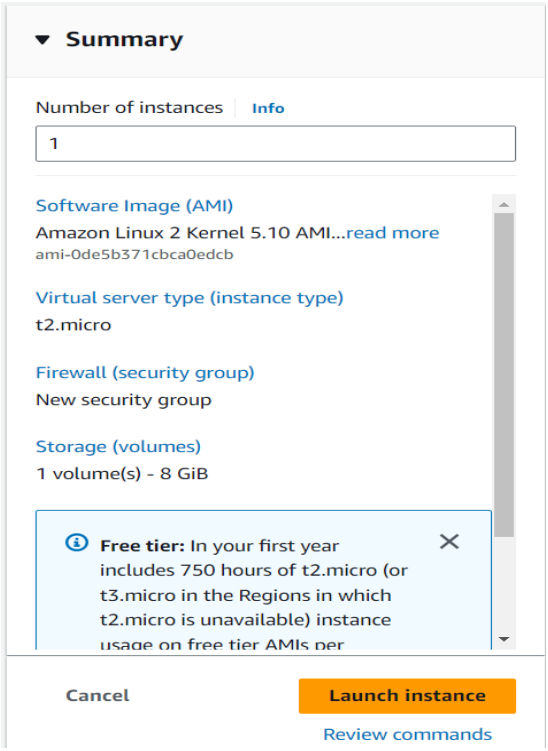
b. From Common security groups, choose your security group from the list of existing security groups.






8. Keep the default selections for the other configuration settings for your instance.

9. Review a summary of your instance configuration in the Summary panel, and when you're ready, choose Launch instance.



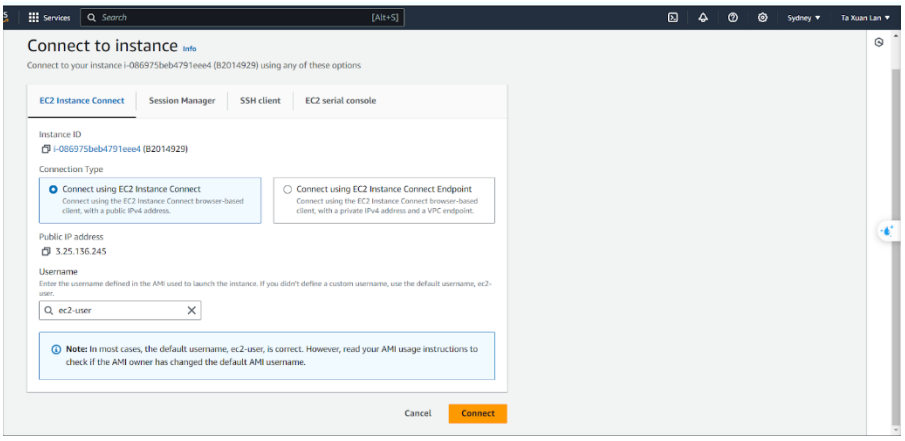

10. A confirmation page lets you know that your instance is launching. Choose View all instances to close the confirmation page and return to the console.

11. On the Instances screen, you can view the status of the launch. It takes a short time for an instance to launch. When you launch an instance, its initial state is pending. After the instance starts, its state changes to running and it receives a public DNS name. If the Public IPv4 DNS column is hidden, choose the settings icon ( Settings icon. ) in the top-right corner, toggle on Public IPv4 DNS, and choose Confirm.

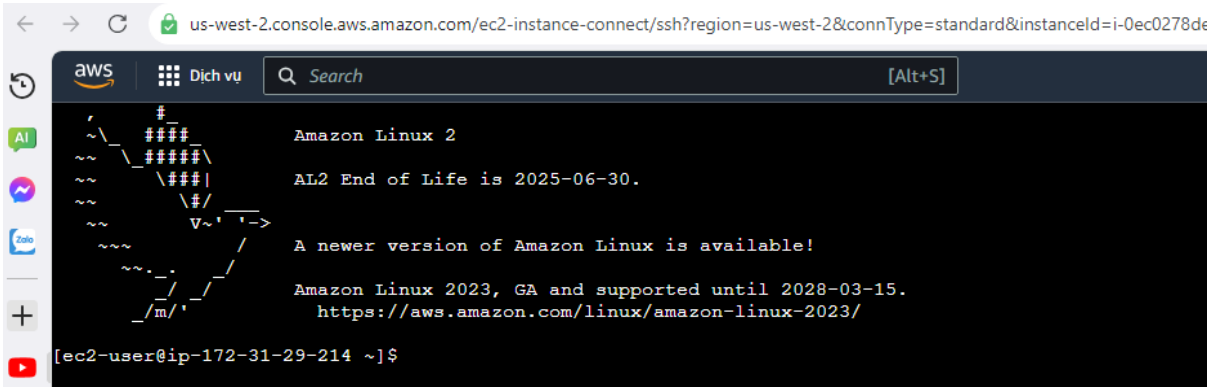
12. It can take a few minutes for the instance to be ready for you to connect to it. Check that your instance has passed its status checks; you can view this information in the Status check column.

**Step 2: Connect to your instance**

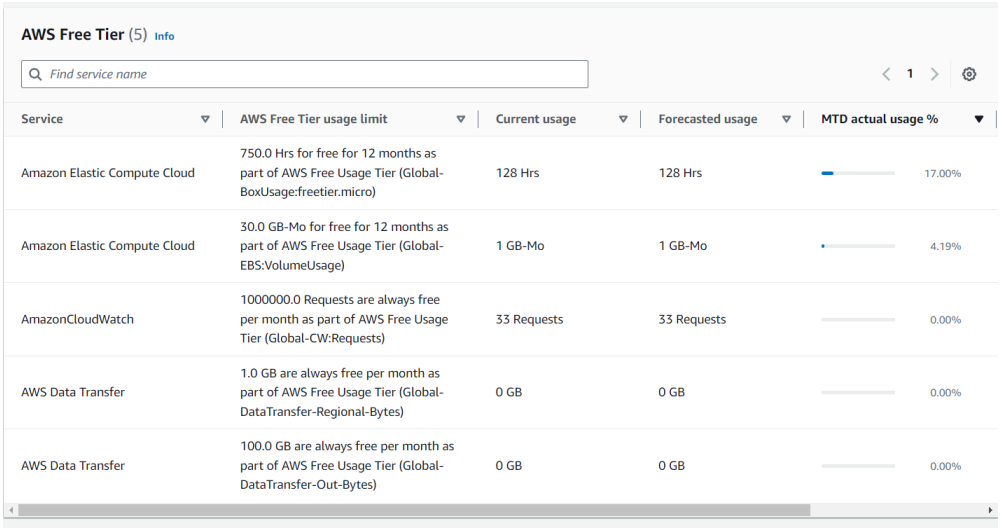
- Connect to instance with EC2



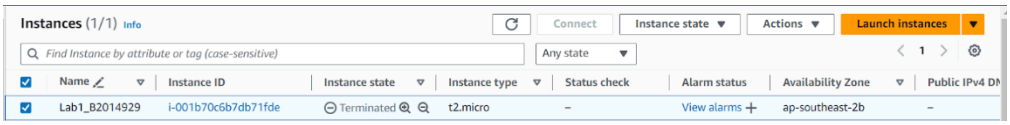

- Connect to instance with SSH client

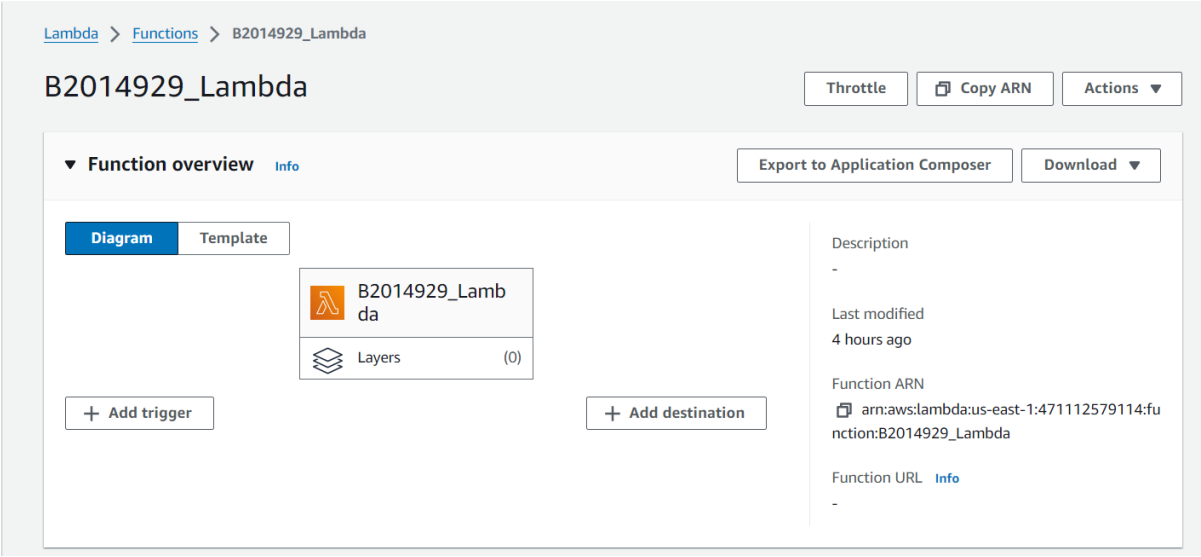

## Step 3: Track your Free Tier usage




### Step 4: Clean up your instance

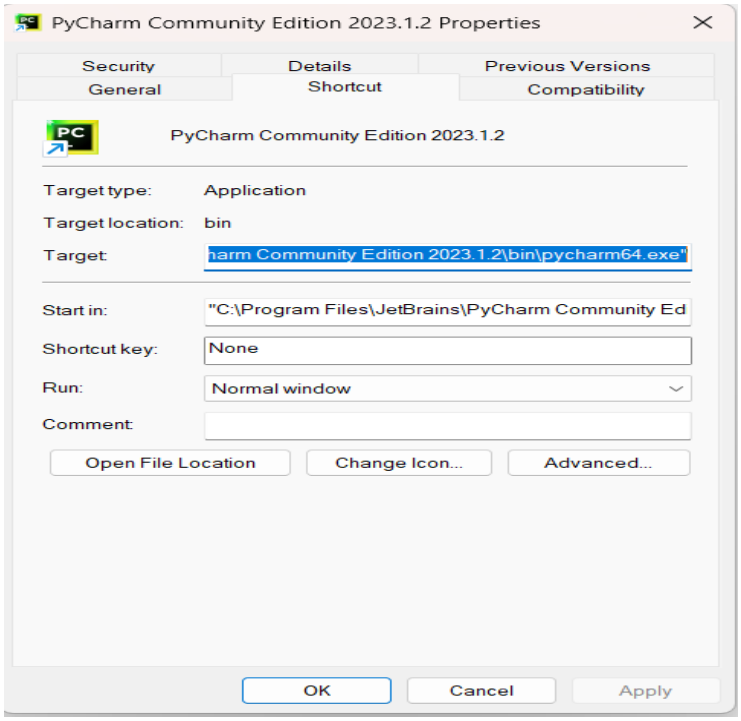



[Step 2]: Select and Configure AWS Services

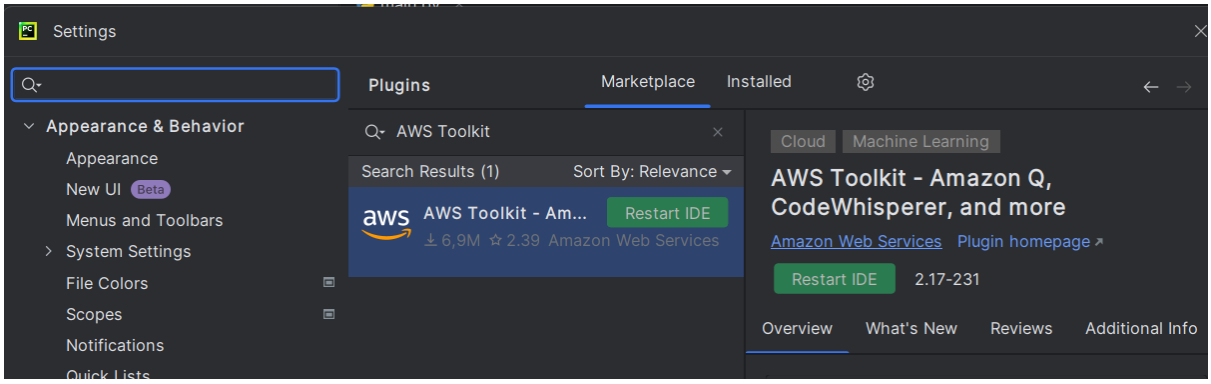



[Step 3]: Set up and use PyCharm

1. Install PyCharm: Download PyCharm from the JetBrains website. The Community version is free, and the Professional version provides additional features.




2. Install AWS Toolkit for PyCharm: A plugin that helps you use AWS services more easily in PyCharm. In PyCharm's Settings or Preferences menu, find the Plugins section, search for the AWS Toolkit, and install it.




3. Setting up the AWS Toolkit: After installing the AWS Toolkit, link your AWS account in PyCharm. For this, you need the access key and secret access key of the IAM user you created.

User name

B2014929_TaXuanLan

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

☑ Provide user access to the AWS Management Console - *optional*

If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.

**Are you providing console access to a person?**

User type

○ Specify a user in Identity Center - Recommended

We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

◉ I want to create an IAM user

We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keyspaces, or a backup credential for emergency account access.

Console password

◉ Autogenerated password

You can view the password after you create the user.

○ Custom password

Enter a custom password for the user.

☐ Show password

☑ Users must create a new password at next sign-in - Recommended

Users automatically get the IAMUserChangePassword policy to allow them to change their own password.

## Create user group

Create a user group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. Learn more

User group name

Enter a meaningful name to identify this group.

B2014929_TaXuanLan

Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.

**Permissions policies** (1/923)

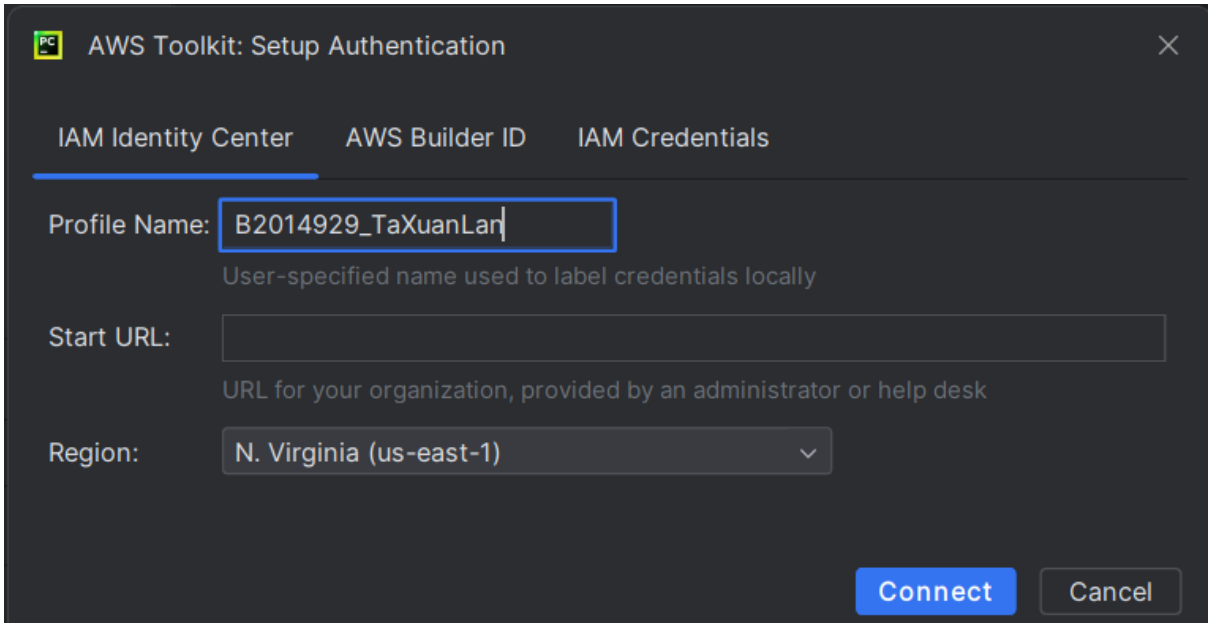
Create policy

Filter by Type

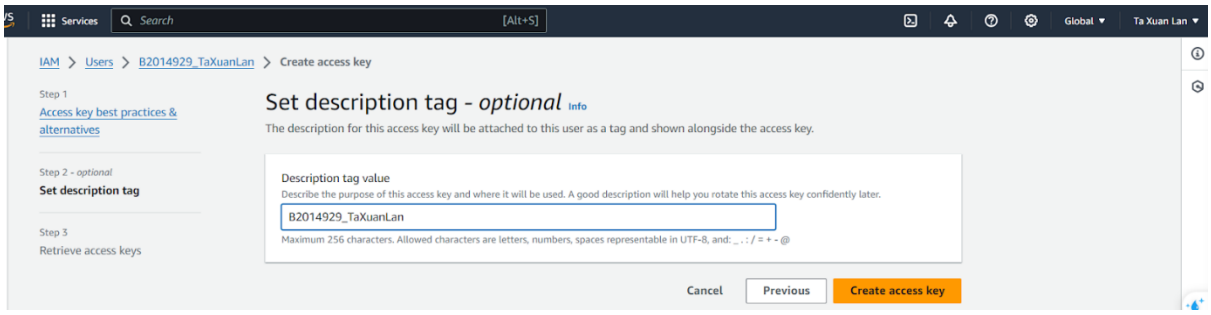
Search | All ty...

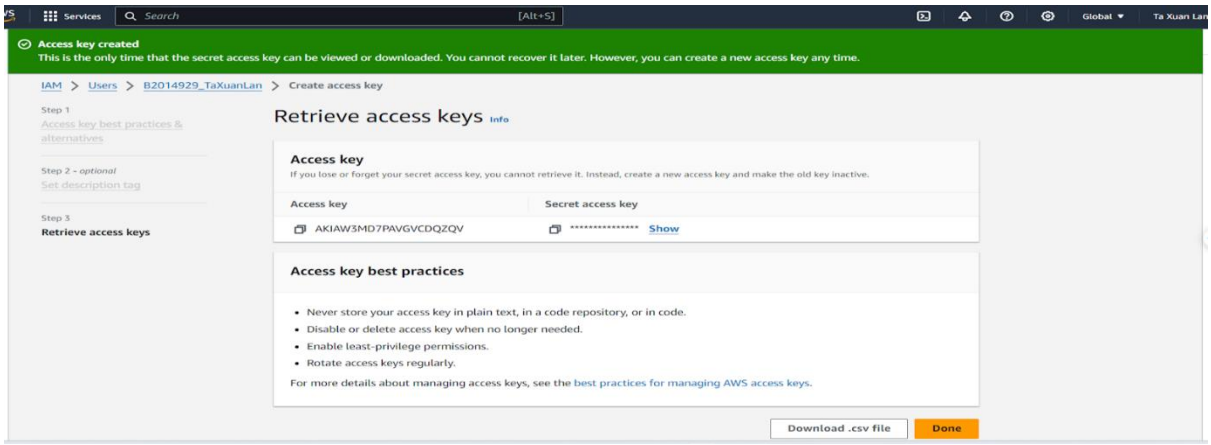
1 2 3 4 5 6 7 ... 47

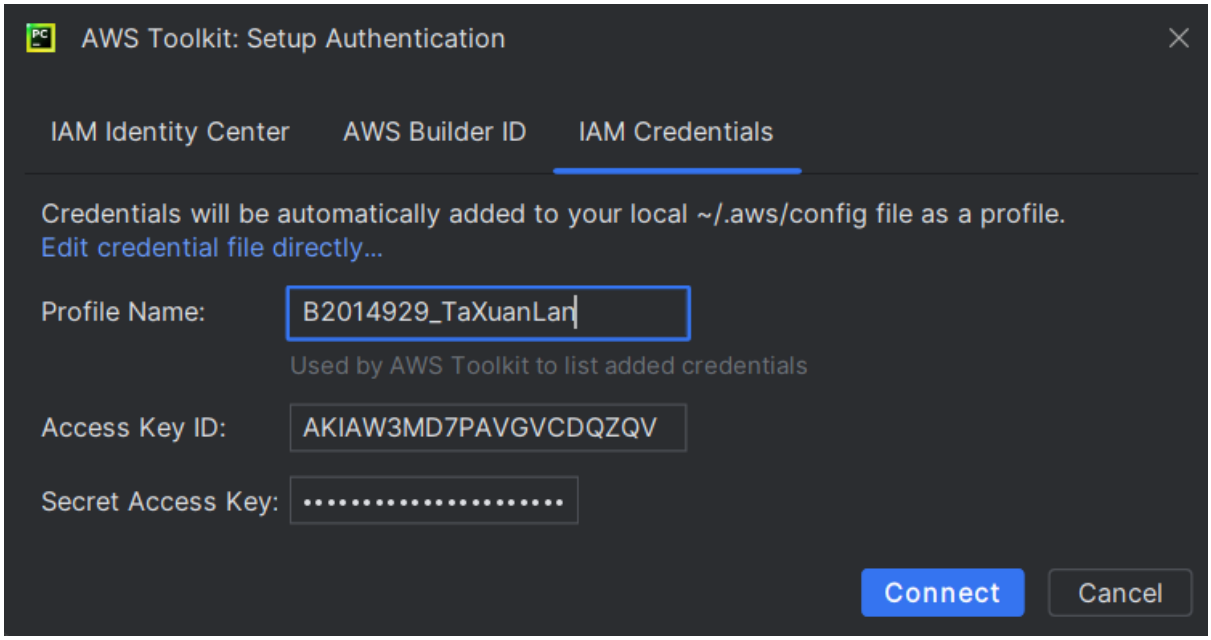
| Policy name | Type | Use... | Description |
|---|---|---|---|
| ☑ AdministratorAccess | AWS managed ... | None | Provides full access to AWS service |

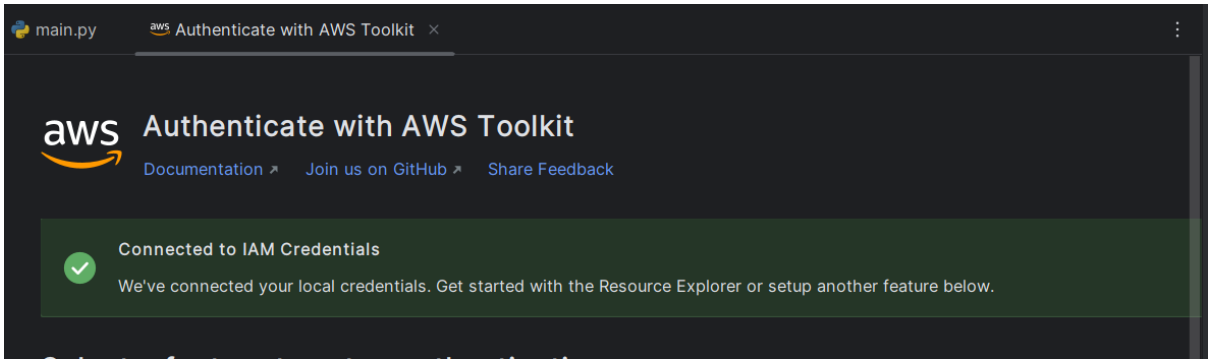

AWS Toolkit: Setup Authentication
IAM Identity Center
AWS Builder ID
IAM Credentials
Profile Name:
B2014929_TaXuanLan
User-specified name used to label credentials locally
Start URL:
URL for your organization, provided by an administrator or help desk
Region:
N. Virginia (us-east-1)
Connect
Cancel


Services
Search
[Alt+S]
Global
Ta Xuan Lan
IAM > Users > B2014929_TaXuanLan > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Access key best practices & alternatives Info
Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.
Use case
Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.


Services
Search
[Alt+S]
Global
Ta Xuan Lan
IAM > Users > B2014929_TaXuanLan > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Set description tag - optional Info
The description for this access key will be attached to this user as a tag and shown alongside the access key.
Description tag value
Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.
B2014929_TaXuanLan
Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @
Cancel
Previous
Create access key


Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.
IAM > Users > B2014929_TaXuanLan > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Retrieve access keys Info
Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.
Access key
Secret access key
AKIAW3MD7PAVGVCDQZQV
Show
Access key best practices
Never store your access key in plain text, in a code repository, or in code.
Disable or delete access key when no longer needed.
Enable least-privilege permissions.
Rotate access keys regularly.
For more details about managing access keys, see the best practices for managing AWS access keys.
Download .csv file
Done




4. Project creation and AWS service integration: Create a new Python project within PyCharm and write code in conjunction with the necessary AWS services (e.g. Lambda, DynamoDB).