Remote Development on AWS: from Cloud9 to VS Code

20 November 2024 - 2 min. read

Cloud9 *Visual Studio Code*

Software development has significantly transformed in recent years, with many developers embracing distributed and collaborative working approaches. With the spread of DevOps methodologies, the adoption of the cloud-native paradigm, and the growing use of automation tools, flexibility and the ability to work on different platforms have become essential. However, this flexibility also introduces challenges, particularly when maintaining the consistency of development environments, which varies greatly depending on the device or operating system used and the projects you are working on.

AWS announced that it has no plans to develop new features for Cloud9, and the service has not been available to new customers for several months. This has led many developers to look for alternatives that offer more customization, flexibility, and a better user experience. **Visual Studio Code (VS Code)**, combined with an EC2 or ECS instance on AWS, is a great alternative. This configuration allows developers to access a robust, unified remote development environment, maintaining the same configurations and tools they are used to, regardless of the client device used.

Using VS Code for remote development ensures a portable workflow, access to on-demand computing resources, and ultra-fast connectivity for downloading libraries, dependencies, and artifacts.

This approach reduces compatibility issues, fully leverages EC2 instances' capabilities, and allows for the integration of custom tools and configurations.

In this article, we will explore how to configure Visual Studio Code to work on an EC2 instance, exploring the necessary settings and their benefits step by step.

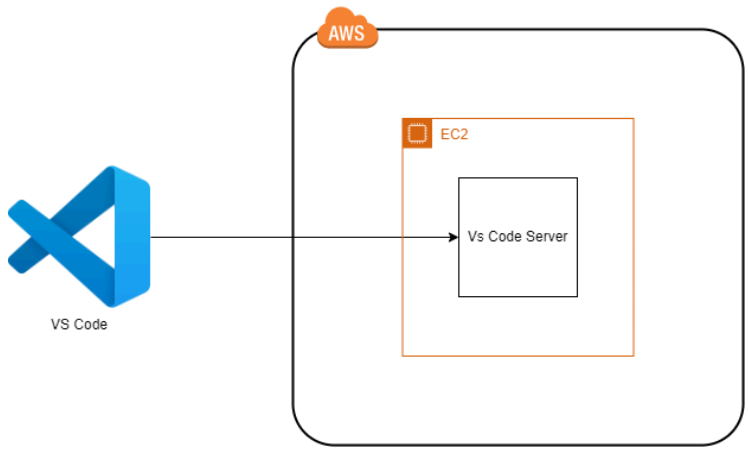
The solution

To obtain an experience similar to that of Cloud9, the code, dependencies, libraries, and specific configurations must physically reside on the remote instance. The local client must, therefore, be wholly integrated with the host, showing its terminal and using the remote environment for library searches, auto-completion, and all functions that require access to the source.

Additionally, running tests on the remote instance or launching a development server to test the software you are working on must be possible.

With Visual Studio Code, we can get everything we need through a single extension, which allows us to connect the client to a Visual Studio Code Server.

Therefore, we will configure an EC2 instance to host the development server and see how to configure the client to connect to the instance and use it for remote development.



Communication between client and server is based on SSH, and it is possible to configure a single instance for multiple developers by taking advantage of the user space authentication and segregation mechanism already present in Linux and supported by SSH.

Each developer can create a user account on the instance, and each user will run a distinct VS Code server process.

Advantages of remote instance development

Developing on a remote instance offers several significant benefits, including the ability to significantly improve a team's workflow and simplify the management of development environments.

For example, one of the most common problems is inconsistency between developer machine configurations. Even slight differences can cause errors that are difficult to diagnose or hinder and slow down collaborative work.

With a remote instance, all of these configurations are centralized. Each developer works in an identical environment, potentially in the same instance, with no specific contraindications. This minimizes compatibility issues and ensures that dependencies are always up-to-date and consistent.

It is possible to configure the host to offer the most commonly used interpreters and libraries, centralizing versioning and distribution.

Where necessary, interpreter version management systems, such as pyenv or nvm, can also be installed and made available to all developers.

A further aspect to be noticed is performance. Many modern frameworks are resource-intensive during the compilation or build phases, not to mention the growing use of machine learning algorithms that require large amounts of CPU time, dedicated GPUs, and large amounts of RAM.

By freeing the IDE from the execution environment using a remote instance, developers can assign hardware resources that are significantly more performing than local devices, increasing or reducing them as needed to optimize costs and speed up work.

Furthermore, if a local device fails or requires maintenance, there are no significant interruptions: the remote environment remains intact and ready for use.

Finally, centralizing development on a remote server makes it easier to implement and monitor security policies and control user activity.

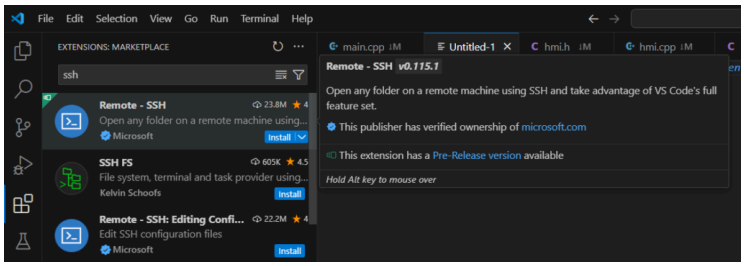
Client Configuration: "Remote - SSH" Extension

The first step is to install and configure the "Remote—SSH" extension in Visual Studio Code. This extension is the heart of remote connection, allowing you to work on a remote instance as if it were local while maintaining the power and flexibility of your development environment.

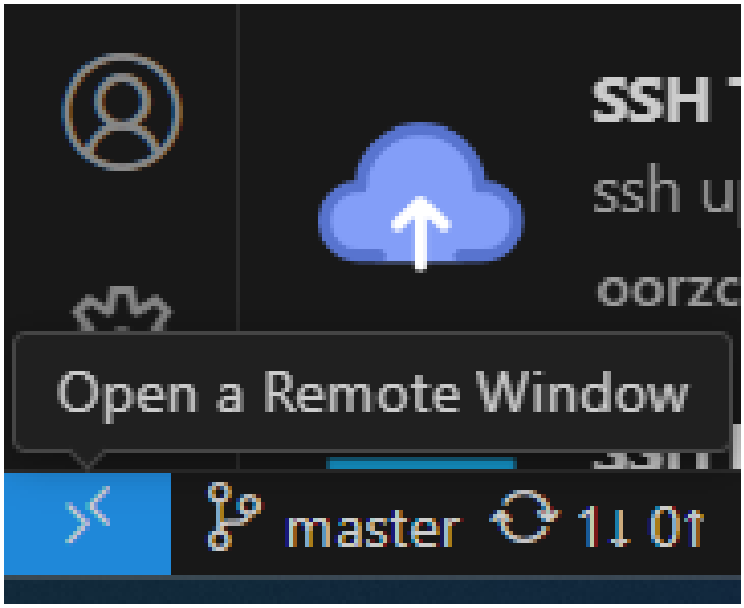
This add-on for VS Code allows you to open a folder on a remote server directly from the editor using an SSH tunnel. With this extension, files are viewed, edited, and managed within VS Code but remain physically on the remote server.

The commands written on the integrated terminal, which is a remote terminal, are also conveyed inside the SSH tunnel, as are all the operations that VS Code performs on the FS to provide support for the desired language and frameworks.

Installing the extension is very easy. In the left sidebar of VS Code, click on the extensions icon and search for "Remote—SSH." The first result should be the official Microsoft extension.



Let's proceed with the extension installation. Once the process is completed, a new option will be available in the control panel that allows you to connect to a remote host.



Instance configuration

No unique configurations are necessary on the remote instance. Once the SSH tunnel is established, VS Code will install the server automatically.

The server component is installed in user space and does not require elevated privileges.

The client will then directly update the server-side component, proposing updates to the user just as it does for the client part.

Therefore, we need to start an EC2 instance, taking care to choose the most suitable resources based on the use case.

Suppose you're testing or the use case needs more horsepower. In that case, the t family gets the best value and usually fits nicely into the CPU usage pattern of a developer who doesn't compile intensively.

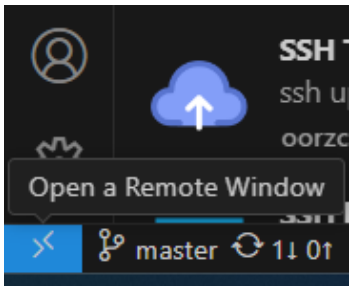
However, if the remote instance is used to accelerate the development and testing of ML algorithms, there are particularly high-performance instances with powerful GPUs available. The family Inf2 is the best choice if the workload is really important and represents the maximum performance currently available for compute-intensive AI projects.

When starting the instance, associate it to a security group that allows connections on port 22 from your IP or the company IP range, and select or generate an SSH key for access.

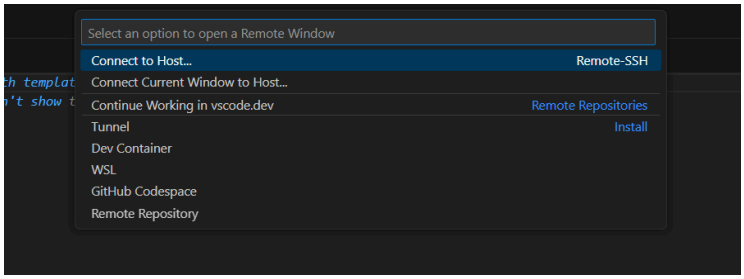
Further, it is possible to connect and create an ad hoc user to avoid using the default user. However, we recommend using SSH key-based authentication and avoiding password-only authentication.

Link to the instance

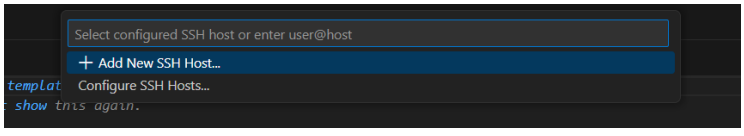
Once the instance is active and reachable, simply click on the extension icon in the control panel at the bottom to connect.



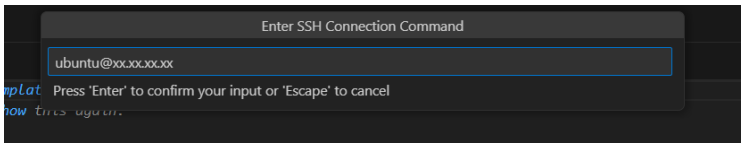
And then select “Connect to host.”



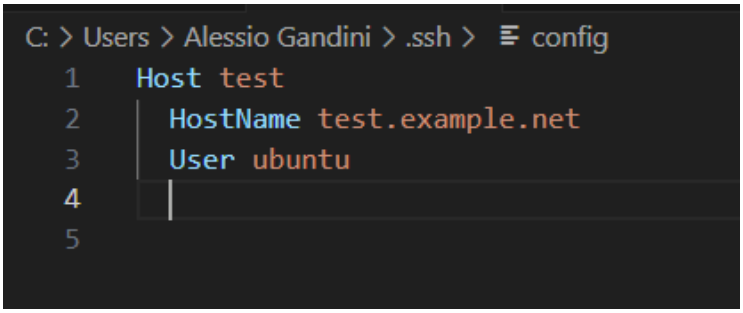
We can then add a new host, specifying our instance's username and IP or domain name.



The format to use is user@ip or fqdn



Next, you will be asked where to save the SSH configuration file. This file will contain the hosts' connection information, so you don't have to rewrite it every time.



Once you have selected the configuration file, you can view and edit it to configure additional options.

During access, VS Code will try to guess the authentication mechanism. If you have used an SSH key, you will have to tell VS Code the location, and this will be saved in the configuration file for subsequent connections.

If you use a passphrase to protect your SSH private key, you must enter it the first time you log in.

After a successful connection, VS Code will ask you which directory to open on the remote server. Once the directory is selected, VS Code will automatically configure the remote development environment, installing the necessary extensions directly from the active SSH tunnel.

Conclusions

We have seen how to develop on a remote instance quickly and cheaply. The main benefits of this development solution are the ability to centralize environment configuration and access powerful hardware resources on demand.

If you want to optimize your development environment and discover new strategies to work more efficiently, continue following our blog.

You will also find other insights, guides, and tips on how to make the most of the latest tools and technologies and the Cloud-Native paradigm.

About Proud2beCloud

Proud2beCloud is a blog by beSharp, an Italian APN Premier Consulting Partner expert in designing, implementing, and managing complex Cloud infrastructures and advanced services on AWS. Before being writers, we are Cloud Experts working daily with AWS

services since 2007. We are hungry readers, innovative builders, and gem-seekers. On Proud2beCloud, we regularly share our best AWS pro tips, configuration insights, in-depth news, tips&tricks, how-tos, and many other resources. Take part in the discussion!

Alessio Gandini

Cloud-native Development Line Manager @ beSharp, DevOps Engineer and AWS expert.Since I was still in the Alfa version, I'm a computer geek, a computer science-addicted, and passionate about electronics all-around.At this moment, I'm hanging out in the IoT world, also exploring the voice user experience, trying to do amazing (Io)Things.Passionate about cinema and great TV series consumer, Sunday videogamer

---