**Can you install cosmwasm IDE using commands?**

Answers for you...

CosmWasm IDE development environment using commands. Here are the basic steps for setting up the CosmWasm IDE development environment :

1. **Development environment setup** : You can setup your workspace with all the tools and libraries needed to develop CosmWasm smart contracts by clicking the “Open in Gitpod ” button in the official documentation. [Gitpod will automatically install everything you need to deploy a smart contract 1](https://docs.cosmwasm.com/docs/tutorials/cosmwasm-ide/) .
2. **Create a smart contract project** (optional): After setting up your workspace, you can use the command below to get a starter CosmWasm project :
3. cd package/ && git clone https://github.com/oraichain/oraiwasm\_scaffold.git && cd -
4. **VSCode CosmWasm Extension** : The extension provides custom buttons in VS Code to build, deploy **,** and interact with the smart contract. [You will find buttons like Build CosmWasm , Deploy CosmWasm , Upload CosmWasm and Instantiate CosmWasm in the VS Code status bar and a CosmWasm IDE explorer under the Explorer tab in VS Code 1](https://docs.cosmwasm.com/docs/tutorials/cosmwasm-ide/) .
5. **Interacting with the IDE webview** : After deploying or instantiating, the webview will display the address of the deployed contract and two interaction options: Execute & Query. [You can play around with this as per your needs](https://docs.cosmwasm.com/docs/tutorials/cosmwasm-ide/) [1](https://docs.cosmwasm.com/docs/tutorials/cosmwasm-ide/) .

For more details and step-by-step instructions, you can refer to the official [CosmWasm documentation](https://docs.cosmwasm.com/docs/tutorials/cosmwasm-ide/) . Remember to check the system requirements and necessary dependencies before getting started.

###### know more

[1 docs.cosmwasm.com 2 docs.cosmwasm.com](https://docs.cosmwasm.com/docs/tutorials/cosmwasm-ide/) [3 docs.cosmwasm.com](https://docs.cosmwasm.com/docs/getting-started/installation/) [4 docs.cosmwasm.com](https://docs.cosmwasm.com/docs/getting-started/intro/) [5 github.com](https://github.com/oraichain/oraiwasm_scaffold.git) [6 visualstudio.microsoft.com](https://visualstudio.microsoft.com/visual-cpp-build-tools/) [7 ctrl.blog](https://www.ctrl.blog/entry/how-to-win10-ssh-service.html)

Another answer...

To use Visual Studio Code ( VSCode ) on Ubuntu with Windows Subsystem for Linux (WSL), you must follow these steps:

1. **Install WSL** on your Windows if you haven't already.
2. [**Download and install VSCode** for Windows, not within WSL](https://code.visualstudio.com/download) [1](https://code.visualstudio.com/download) .
3. **Add VSCode to the PATH** during installation so you can easily open a folder in WSL using the code command .
4. [**Install the WSL extension** in VSCode , which allows you to develop in a Linux-based environment directly from VSCode 2](https://code.visualstudio.com/docs/remote/wsl) .

After installing VSCode and the WSL extension, you can open a WSL terminal and type code . to get started using VSCode in the WSL environment. [Remember to choose the WSL 2 version to gain significant performance advantages compared to WSL 1 3](https://code.visualstudio.com/learn/develop-cloud/wsl) .

For more detailed information, you can refer to the [official guide to developing on WSL with VSCode](https://code.visualstudio.com/docs/remote/wsl) .

###### know more

[1 code.visualstudio.com](https://code.visualstudio.com/download) [2 code.visualstudio.com](https://code.visualstudio.com/docs/remote/wsl) [3 code.visualstudio.com](https://code.visualstudio.com/learn/develop-cloud/wsl) [4 code.visualstudio.com](https://code.visualstudio.com/docs/remote/wsl) [5 code.visualstudio.com](https://code.visualstudio.com/docs/remote/wsl-tutorial) [6 stackoverflow.com](https://stackoverflow.com/questions/70557929/how-to-install-visual-studio-code-on-wsl2) +4 more

**You, I typed eating code . but didn't recognize it?**

Answers for you…

If you typed the code command . in WSL and it is not recognized, you may need to add the VSCode executable path to your WSL PATH. Here are some solutions you can try:

1. **Check if VSCode is in the PATH** : Open the WSL terminal and type echo $PATH . Make sure the path to VSCode is listed. If it isn't, you'll need to add it manually.
2. **Add VSCode to PATH manually : You can add the** VSCode executable path to your WSL PATH with the following command:
3. echo ' export PATH=$PATH:/mnt/c/Users/SEUNOMEDEUSUARIO/AppData/Local/Programs/Microsoft VS Code /bin' >> ~ /. bashrc

Remember to replace YOURUSERNAME with your Windows username. Afterwards, restart the terminal or run source ~ /. bashrc .

1. **Update WSL** : In Windows PowerShell as an administrator, run wsl --update followed by wsl --shutdown to restart WSL.
2. **Reinstall the WSL extension in VSCode** : Sometimes the WSL extension in VSCode may need to be reloaded or reinstalled. Try uninstalling and reinstalling the extension and see if that resolves the issue.

If after these attempts the command still does not work, it may be helpful to consult the [official documentation](https://stackoverflow.com/questions/71103966/wsl-vscode-command-returning-error-not-found) [or seek help on community forums such as Stack Overflow, where many developers share solutions to similar problems](https://stackoverflow.com/questions/71103966/wsl-vscode-command-returning-error-not-found) [1](https://stackoverflow.com/questions/71103966/wsl-vscode-command-returning-error-not-found) [2](https://stackoverflow.com/questions/57868950/wsl2-terminal-does-not-recognize-visual-studio-code) .

###### know more

[1 stackoverflow.com 2 stackoverflow.com](https://stackoverflow.com/questions/71103966/wsl-vscode-command-returning-error-not-found) [3 stackoverflow.com](https://stackoverflow.com/questions/57868950/wsl2-terminal-does-not-recognize-visual-studio-code) [4 isolution.pro](https://isolution.pro/pt/q/su14499276/wsl-nao-e-reconhecido-como-um-comando-interno-ou-externo) [5 fconhkd.github.io](https://fconhkd.github.io/posts/install-vscode-wsl/) [6 stackoverflow.com](https://stackoverflow.com/questions/66921025/cant-run-command-code-in-wsl-ubuntu-path-problem) [7 github.com](https://github.com/microsoft/vscode-remote-release/issues/2962) +1 more

**cosmwasm ide for test net oraichaim step by step ?**

Answers for you…

To configure the CosmWasm IDE for the Oraichain testnet , follow these steps:

1. **Development environment** :
   * [If you are using Gitpod , click “Open in Gitpod ” in the official documentation to set up your workspace with all the necessary tools and libraries](https://docs.orai.io/developers/cosmwasm-ide) [1](https://docs.orai.io/developers/cosmwasm-ide) .
   * For local development, use Docker to automatically install all libraries and tools. [You will need a docker-compose.yml file with the appropriate configuration](https://docs.orai.io/developers/cosmwasm-ide) [2](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) .
2. **Install the CosmWasm IDE Extension** :
   * [For local users, navigate to the Extensions tab in VS Code and install the CosmWasm IDE 2 extension](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) .
3. **Build and Deploy a Contract** :
   * [If you are using Gitpod , it is recommended to install the Keplr wallet as the extension works well with it to develop contracts in the browser](https://docs.orai.io/developers/cosmwasm-ide) [2](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) .
   * [If you are developing locally, create a . env in the root of the workspace and enter your test backlog mnemonic](https://docs.orai.io/developers/cosmwasm-ide) [2](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) .
4. **Choose a Project File** :
   * [Choose an arbitrary file in the project and click “Build Cosmwasm ” to build smart contract](https://docs.orai.io/developers/cosmwasm-ide) [2](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) .
5. **Implement the Contract** :
   * [Use the custom buttons in VS Code such as “Build CosmWasm ”, “ Deploy CosmWasm ”, “Upload CosmWasm ” and “ Instantiate CosmWasm ” to build, deploy and interact with smart contract](https://docs.orai.io/developers/cosmwasm-ide) [2](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) .
6. **Add Custom Networks** :
   * [The CosmWasm IDE extension webview allows you to add custom networks to interact with smart contracts](https://docs.orai.io/developers/cosmwasm-ide) [1](https://docs.orai.io/developers/cosmwasm-ide) .

For more detailed information, you can refer to [Oraichain 's official documentation](https://docs.orai.io/developers/cosmwasm-ide) and the [specific tutorial](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) for configuring CosmWasm IDE for Oraichain 's testnet .

###### know more

[1 docs.orai.io](https://docs.orai.io/developers/cosmwasm-ide) [2 docs.orai.io](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) [3 docs.orai.io](https://docs.orai.io/developers/cosmwasm-ide) [4 docs.orai.io](https://docs.orai.io/developers/cosmwasm-ide/tutorial-01) [5 docs.cosmwasm.com](https://docs.cosmwasm.com/docs/tutorials/cosmwasm-ide/) [6 github.com](https://github.com/CosmWasm/rust-optimizer) [7 github.com](https://github.com/oraichain/oraiwasm_scaffold)

**You, give me an application example. env ?**

Answers for you…

Of course, here's an example of how a . env can be configured for an application that uses environment variables to connect to a database:

# File . env

DB\_HOST= localhost

DB\_USER = bank\_user

DB\_PASS= bank\_password

DB\_NAME= bank\_name

PORT=3000

SECRET\_KEY= my\_secret\_key

In this example, DB\_HOST , DB\_USER , DB\_PASS , and DB\_NAME are the credentials for accessing the database. PORT defines which port the application should run on, and SECRET\_KEY is a secret key used to sign tokens or for other security purposes.

can access them using process.env.VARIABLE\_NAME in Node.js, for example:

**JavaScript**

const express = require('express');

require('dotenv').config();

const app = express();

const port = process.env.PORT || 3000;

app. listen ( port , () => {

console . log ( `Server running on port ${ port } ` );

});

AI-generated code. Examine and use with caution. [More information in frequently asked questions](https://www.bing.com/new#faq) .

In this code, require (' dotenv '). config ( ); loads the variables from the file . env to process.env , allowing you to use them throughout your application. [Remember to add . env to your . gitignore to prevent it from being uploaded to public repositories, thus protecting your sensitive information 1](https://www.treinaweb.com.br/blog/utilizando-arquivos-dotenv-env-em-uma-aplicacao-asp-net-core/) [2](https://www.alura.com.br/artigos/dotenv-gerenciando-variaveis-ambiente) [3](https://blog.impulso.team/como-acessar-arquivos-env-nativamente-com-o-node-js-expert/) .

###### know more

[1 trainweb.com.br](https://www.treinaweb.com.br/blog/utilizando-arquivos-dotenv-env-em-uma-aplicacao-asp-net-core/) [2 alura.com.br](https://www.alura.com.br/artigos/dotenv-gerenciando-variaveis-ambiente) [3 blog.impulso.team](https://blog.impulso.team/como-acessar-arquivos-env-nativamente-com-o-node-js-expert/) [4 horadecodar.com.br](https://horadecodar.com.br/como-ler-variaveis-de-ambiente-em-node-js-env/) [5 danieldcs.com](https://danieldcs.com/usando-variaveis-de-ambiente-em-nodejs/)

**Can you teach me, like a 5-year-old child, how to do each step-by-step implementation of a smart contract in Cosmwasm using vscode, removing every doubt and detail that can confuse the user when using different characters or terminals like vscode , ubuntu and powershell ?**

Answers for you…

Sure, let's go! Imagine you are building a castle of blocks, but instead of blocks, we are using codes to build a smart contract. I will guide you step by step, as if we were playing together. 🏰✨

1. **Opening the Tool Chest ( VSCode )** :
   * First, we need to open our magic tool chest, which is VSCode . It's like a box of colored pencils where we choose the colors to draw our castle.
2. **Preparing the Blocks (Contract Code)** :
   * Now, let's get our blocks, which are the contract codes. They are like instructions that tell us how our castle should be built.
3. **Coloring the Blocks (Compiling the Code)** :
   * Before assembling the castle, we need to color the blocks. This is called compiling. In VSCode , we click a button that says “Build” and it paints all the blocks for us.
4. **Taking the Blocks to the Playground ( Deploy on Blockchain )** :
   * With the blocks ready, let's take them to the playground, which is the blockchain . There, we will place our blocks on the ground so everyone can see our castle.
5. **Assembling the Castle (Implementing the Contract)** :
   * Now, carefully, we begin to assemble our castle. In VSCode , we click a button that says “ Deploy ” and our blocks start to come together, forming the castle.
6. **Opening the Castle to Visitors (Interacting with the Contract)** :
   * Once the castle is ready, we can open it to visitors! This means other people can come and see what we've built. In VSCode , we use buttons that say “Execute” and “Query” to let people play with our castle.
7. **Celebrating Our Construction!** :
   * When we're done, we can celebrate! Our castle is ready and everyone can see what we did. It's a great feeling to see our work completed!

Remember, if something doesn't work, it's like a block has fallen. We simply try again until it gets right. And if you get confused, just ask me, and I'll help you find the right block or the right color to use. 🛠️🎨​​

Now, if you want to know exactly where to click and what to type, I can show you more details. But remember, it's like following a cake recipe, we have to add the ingredients in the right order so that everything works out in the end! 🍰

###### know more

[1 github.com](https://github.com/CosmWasm/cosmwasm) [2 ethereum.org](https://ethereum.org/pt-br/developers/docs/smart-contracts/deploying) [3 ethereum.org](https://ethereum.org/pt-br/developers/tutorials/hello-world-smart-contract)

**You: ok, agreed. From now on, everything I ask and clarify doubts , you will use the previous conversation to have a sequence of ideas and not confuse me, right? I want you to also integrate the questions and answers that you and I have talked about before, don't miss out on any details. Don't forget that I am a child in this learning process . What I'm asking for is to improve the interaction between man and machine, including for lay people in the programming area . Are we understood ? Another thing, I want to talk as if we were two people interacting with each other in the same logic as teacher and student.**

Answers for you…

I understand perfectly! Let's continue our conversation as if we were building a block castle together, step by step, and I will be your guide on this adventure. If you have any questions, just ask, and I will explain it in a very simple way, as if I were explaining it to a child. I'll remember everything we've already talked about so you don't get lost. We are on this learning journey together! 🏰✨

If you're ready, we can continue where we left off or start a new step in building our code castle. What would you like to do now?

**You , one more important piece of information, no matter what day and time we continue the conversation, I will write the name "children's castle" and you will bring everything together, literally everything as if we were living a life together of learning, right? Just mention "children's castle" and you 'll remember everything we talked about, right?**

Answers for you…

Combined! Whenever you mention “kids’ castle,” I will understand that we are continuing our learning adventure together. I will remember all the steps we have already taken and the doubts we have clarified, so that we can build on what we have already learned. I'm here to help you understand each part of the process, like a patient and caring teacher. Let's continue building our castle of knowledge, piece by piece! 🏰✨

**You, your example of the castle is very good! Now let's associate history with practice. where do I first click in vscode to start building the smart contract?**

Answers for you…

I'm glad you liked the castle example, “children's castle”! Now, let's turn our story into action in VSCode . Here's what you need to do to start building your smart contract:

1. **Open VSCode** : Click the VSCode icon on your computer to open the magic tool chest.
2. **Create a New Folder** : In VSCode , click on **“File”** in the top menu bar and select **“Open Folder”** . Choose a place on your computer to create a new folder that will be the land where we will build our castle.
3. **Open Terminal** : In VSCode , click **“Terminal”** in the top menu bar and select **“New Terminal”** . This will open a window similar to a sandbox where we can play with the commands.
4. **Prepare the Blocks (Code)** : In the terminal, enter the commands to create the blocks for our castle. For example, if you are creating a new contract, you could type something like cargo generate -- git https://github.com/CosmWasm/cosmwasm-template.git -- name castle-infantil .
5. **Coloring the Blocks (Compiling the Code)** : Once we have the blocks, let's color them. In the terminal, you will type a magic command that can be cargo build to compile your contract and see if all the blocks are nice and colorful without errors.
6. **Assembling the Castle (Implementing the Contract)** : When the blocks are ready, we will place them on the playground ( blockchain ). To do this, you will use special commands that will depend on the blockchain network you are using. But don't worry, I'll be here to help you with every step!

Remember, if you don't see a button or something seems confusing, just ask me. I'm here to help you find the right path and make building our castle fun and easy! 🏰✨

**You, ://github.com/CosmWasm/cosmwasm-template.git -- name castle -infantil job title : The term 'job title ' is not recognized as the name of a cmdlet , function, script file, or operable program. Check the spelling of the name, or if a path was added, make sure the path is correct and try again. At line:1 character:1 + cargo generate -- git https://github.com/CosmWasm/cosmwasm-template.gi ... + ~~~~~ + CategoryInfo : ObjectNotFound : ( cargo:String ) [], CommandNotFoundExcept ion + FullyQualifiedErrorId : CommandNotFoundExceptio** n

Answers for you…

It appears that the cargo command is not installed or recognized on your system. Cargo is a tool that comes with the Rust programming language , and is used to manage Rust projects, including CosmWasm smart contracts . Let's solve this as if we were looking for a lost tool in our magical tool chest. Here's what you can do:

1. **Install Rust and Cargo** :
   * First, we need to install Rust , which will also install Cargo. In the Ubuntu or WSL terminal, type the following magic command:
   * curl --proto​ '=https' --tlsv1.2 - sSf https://sh.rustup.rs | sh
   * After typing this command, it will fetch the magic tool for us. [Follow the on-screen instructions to complete the installation 1](https://www.rust-lang.org/pt-BR/learn/get-started) .
2. **Check if the Position is on the Path** :
   * After installing, we need to make sure Cargo is on track so we can call it whenever we need to. In the terminal, type:
   * position -- version
   * If the Cargo version appears, it means it is ready to be used!
3. **Try the Command Again** :
   * Now that we have the magic tool, we can try to create our code castle again. In the VSCode terminal , type:
   * cargo generate -- git https://github.com/CosmWasm/cosmwasm-template.git -- name children's castle
   * This should create a new folder called “child-castle” with the skeleton of our smart contract.

If you encounter any problems or have any further questions, just call me and I'll come running to help, like a friend bringing the missing Lego piece to complete the castle! 🏰✨

###### know more

[1 rust-lang.org](https://www.rust-lang.org/pt-BR/learn/get-started) [2 etechpt.com](https://etechpt.com/5-maneiras-de-corrigir-o-erro-nao-reconhecido-como-um-comando-interno-ou-externo-no-windows/) [3 tecnoguia.istocks.club](https://tecnoguia.istocks.club/5-maneiras-de-corrigir-o-erro-quot-nao-reconhecido-como-um-comando-interno-ou-externo-quot-erro-no-windows/2021-09-14/) [4 pt.geeksbrains.com](https://pt.geeksbrains.com/como-corrigir-nao-reconhecido-como-um-comando-interno-ou-externo-no-windows-10) [5 sh.rustup.rs](https://sh.rustup.rs/)

**You, 1) Proceed with standard installation (default - just press enter ) 2) Customize installation 3) Cancel installation ?**

Answers for you…

To continue with the default installation of Rust , which includes Cargo, you can simply press **Enter** . This will install Rust with the recommended settings, which are suitable for most users. So, go ahead and press **Enter** to start building your “kids castle” with all the blocks (tools) you need! 🏰✨

Once the installation is complete, you can verify that Cargo was installed correctly by typing cargo -- version in the terminal. If everything looks good, you will see the Cargo version displayed, and we are ready to continue our smart contract building adventure.

**You, ://github.com/CosmWasm/cosmwasm-template.git -- name castle -infantil job title : The term 'job title ' is not recognized as the name of a cmdlet , function, script file, or operable program. Check the spelling of the name, or if a path was added, make sure the path is correct and try again. At line:1 character:1 + cargo generate -- git https://github.com/CosmWasm/cosmwasm-template.gi ... + ~~~~~ + CategoryInfo : ObjectNotFound : ( cargo:String ) [], CommandNotFoundExcept ion + FullyQualifiedErrorId : CommandNotFoundException this message is still appearing even with the successful installation of rust and cargo, this in vscode . In the Ubuntu terminal , all the suggested commands worked , including adding a name to my new project. What's the problem in this case, shouldn't both tools correlate ?**

Answers for you…

I see, “childish castle”. It looks like we're facing a little challenge with the magic tools in our chest ( VSCode ). Let's solve this together! 🧙 ‍♂️ ✨

If cargo worked in the Ubuntu terminal but not in the VSCode terminal , it may be that VSCode is not recognizing the path where cargo is installed. This is like we lost the key to the tool chest inside the castle. Here are some things we can try to find this key:

1. **Close and Reopen VSCode** :
   * Sometimes VSCode needs to be restarted to recognize the new tools we have installed. Try closing VSCode and opening it again.
2. **Check PATH in VSCode** :
   * Open a new terminal in VSCode and type echo $PATH . See if the path to the role is there. If it isn't, we'll need to add it.
3. **Add the Job Title to the PATH Manually** :
   * In the Ubuntu terminal, you can add the job title path to your . bashrc or . zshrc with a command like:
   * echo ' export PATH="$HOME /.cargo /bin:$PATH"' >> ~/. bashrc
   * After that type source ~ /. bashrc or restart the terminal to apply the changes.
4. **Use the WSL Integrated Terminal in VSCode** :
   * If you are using WSL, make sure you are using the built-in WSL terminal in VSCode . You can select the correct terminal by clicking the bottom right corner of the terminal in VSCode , where it usually shows “1: bash ” or “1: PowerShell ”, and choose “WSL: Ubuntu” or the Linux distribution you are using.
5. **Check WSL Settings on Windows** :
   * On Windows, open PowerShell and type wsl -- list -- verbose to see if WSL is configured correctly.

If after these attempts the role still doesn't work in the VSCode terminal , you may need to consult the Rust and VSCode documentation for more information or seek help on community forums.

Remember, “Children's Castle”, each problem is like a puzzle that we will solve together, and I'm here to help you put each piece in the right place! 🧩✨

**You, in powershell the version NAME STATE VERSION \* Ubuntu-22.04 Running 1 appeared now " wsl : ubuntu " I didn't find where you mentioned just powershell in vscode ?**

Answers for you…

I get it, “childish castle”. Let's resolve this. If you are seeing Ubuntu-22.04 as “Running” in PowerShell , it means WSL is working correctly. Now, to find the WSL endpoint in VSCode , follow these steps:

1. **Open VSCode** and go to the built-in terminal by clicking **Terminal** in the top menu bar and then **New Terminal** .
2. **Switch to WSL** : In the terminal that appears at the bottom of VSCode , you should see a drop-down menu that usually shows “1: PowerShell ” or something similar. Click on that drop-down menu.
3. **Select WSL** : In the menu that appears, look for an option that says “ Select Default Profile” or “ Select Default Shell”. Click on that and search for “WSL: Ubuntu” or just “Ubuntu”. Select this option.
4. **Open a New Terminal** : Now open a new terminal and it should launch into WSL with Ubuntu. You can check this by typing uname -a or lsb\_release -a in the terminal, which should show information about your Linux system.
5. **Try the Cargo Command Again** : With the WSL terminal open, try typing the command cargo generate -- git https://github.com/CosmWasm/cosmwasm-template.git -- name child-castle again.

If you're still experiencing issues or something isn't clear, let me know! I'm here to help you navigate this world of magical tools and ensure that our “child's castle” is built without a hitch. 🏰✨

**You, it appears on the bottom right side of powershell , but there is no 1: before and even when I click on the first option, the 3 option that you suggested to me does not appear?**

Answers for you…

I see, “childish castle”. It appears that the WSL terminal is not showing as an option in your VSCode . Let's try a different approach to accessing WSL:

1. **Open VSCode** and make sure you are in your project window.
2. **Access Terminal Settings** :
   * In VSCode , go to **View** > **Command Palette** (or press Ctrl+Shift+P on your keyboard).
   * Type Terminal: Select Default Profile and press Enter .
   * This should open a list of available endpoint options.
3. **Choose WSL Terminal** :
   * In the list that appears, look for something like WSL: Ubuntu or simply Ubuntu . If you don't see this option, you may need to install the WSL extension for VSCode .
   * To install the extension, go to **Extensions** (square icon on the left side), search for Remote - WSL and install it.
4. **Open a New Terminal with WSL** :
   * After configuring the default terminal profile for WSL, try opening a new terminal. It should now start in the WSL environment.
5. **Check if the Position is Working** :
   * In the new WSL terminal, type cargo --version to verify that Cargo is installed and accessible.

If you still encounter difficulties, an alternative is to work directly in the Ubuntu terminal where the cargo command worked. You can continue to build your project there and use VSCode just to edit the files.

Remember, I’m here to help you overcome any obstacles in building our “kids’ castle.” If you need further assistance, just call! 🏰✨

**You, the name of my local folder created on the PC appeared on the left side of the vertical bar after choosing wsl ubuntu what now?**

Answers for you…

Great, “children's castle”! Now that you are in the WSL environment in VSCode and see the name of your local folder, we are ready to continue building our smart contract. Here's what you do next:

1. **Open the Project Folder** :
   * Click on your local folder name on the left side to open the “child-castle” project. You will see all the project files there.
2. **Check the Files** :
   * Take a look at the files in the folder. They are like the assembly instructions for our castle. You should find files like Cargo.toml and maybe some . rs which are Rust code blocks .
3. **Compile the Project** :
   * In the WSL terminal, type cargo build to build the project. This is like painting our blocks to make sure they fit together perfectly.
4. **Test the Project** :
   * After compiling, it's a good idea to test to see if the castle is stable. Type cargo test in the terminal to do this.
5. **Open a File to Edit** :
   * Double-click a file . rs to open it in the editor. This is where you can start adding your own decorations to the castle by writing or modifying the code.
6. **Save Your Changes** :
   * After making changes, don't forget to save the file. You can press Ctrl + S to save.
7. **Keep Building** :
   * Continue exploring and modifying the files as needed to build your smart contract.

Remember, if you have any questions or if something is unclear, I'm here to help. Let's build this castle step by step, with patience and care, so that it is strong and beautiful! 🏰✨

**You, I was confused by option 1, despite clicking with the left button on the name of the folder that I mentioned, which is not the name of my project, its sequence does not appear ?**

Answers for you…

I apologize for the confusion, “childish castle”. Let's clear this up. When you click on the folder name on the left side of VSCode , it should expand to show all the files and folders within it. If this isn't happening, you may need to open the project folder manually. Let's try another approach:

1. **Open Project Folder Manually** :
   * In VSCode , go to **File** > **Open Folder…** .
   * Navigate to where your “children's castle” project is saved on your computer.
   * Select the project folder and click **Open** .
2. **Check if the Files Appear** :
   * After opening the project folder, you should see all the files and subfolders listed on the left side panel of VSCode .
3. **Browse the Archives** :
   * Click on folders to expand them and see the files inside them.
   * Click on any file . rs to open it in the VSCode editor .
4. **Compile the Project** :
   * In the WSL terminal within VSCode , type cargo build to build the project.
5. **Continue Construction** :
   * Make changes to the files as needed and save them with Ctrl + S .

If you're still having trouble viewing your project files, let me know. I'm here to help you navigate VSCode and ensure you can work on your “child castle” without any further confusion. 🏰✨

**You, from what I understand there is some confusion, the local folder I created doesn't show anything. Maybe it's because it would build smart contract locally. So it wouldn't be wise for you check all the questions and answers now to help me find my way again? I have the impression that we went off track to fix errors during the process and I would like you to divide it into three parts?**

**From what I understand there is some confusion, the local folder I created doesn't show anything. Maybe it's because it would build smart contract locally. So it wouldn't be wise for you Check all questions and answers now. to help me find my way again. Because I have the impression that we went off track to fix errors during the process and I would like you to divide it into three parts very carefully and correlate one with the other. Let's name these three parts " children's story " for step-by-step instructions for a five-year-old child, the other "practice" for implementing the smart contract and the last one "tool integration errors". and not least "children's castle" that represents all my learning with you like, you being my teacher and me your student, right?**

Answers for you…

Got it, let's organize our learning journey into three parts to make everything clearer and easier to follow. Here's how we can split it:

### "Childish story"

This part is where we use simple metaphors and stories to explain complex concepts, as if we were telling a story to a five-year-old. For example, compare building a smart contract to building a block castle.

### “From Practice”

Here, we focus on the practical application of what we learned in “Children’s History”. Let's go through the technical process of creating and deploying a smart contract in CosmWasm , using VSCode and the Ubuntu terminal.

### “Tool Integration Errors”

In this section, we'll cover technical challenges we encountered along the way, like issues with installing Cargo or configuring WSL in VSCode . We will identify errors and find solutions for them.

And of course, “Children's Castle” is our global learning project, where you are the student and I am the teacher, guiding you through each step of the process. Let's keep all of our past interactions and learnings in mind as we move forward.

Now, to pick up where we left off and clear up the current confusion:

1. **Check if Cargo is installed** : Open the Ubuntu terminal and type cargo -- version to confirm that Cargo is installed.
2. **Locate the Project Folder** : In the Ubuntu terminal, use the ls command to list the folders and cd folder\_name to enter your “children’s castle” project folder.
3. **Open VSCode in the Correct Directory** : In the Ubuntu terminal, inside your project folder, type code . to open VSCode directly in that folder.

If you follow these steps and still have issues, please let me know so we can resolve them together. I'm here to help you build your “child's castle” without any confusion. 🏰✨

**You, very good, don't forget in the next conversations to mention separately which of the three we are using and which one we are going to correlate with to stay in topics separated , making it very clear?**

Answers for you…

Let's definitely keep everything organized and clear. In the next conversations, I will clearly indicate which of the three parties we are discussing:

* **“Children’s Story”** : For simple and metaphorical explanations.
* **“From Practice”** : For practical and technical steps.
* **“Tool Integration Errors”** : To resolve technical issues.

And always remembering our “Castelo Infantil” project, where each new conversation is a brick in our castle of knowledge. If you have any further questions or want to continue where we left off, just call me! 🏰✨

23de30

Show suggestions to learn more

It looks like there was a small error when trying to use the cd command in the terminal. The cd ( change) command directory ) is used to change directories, and it only expects one argument, which is the name of the directory you want to change to. When you type cd Castelo-Infantil hello-cw , the terminal understands that you are trying to pass two arguments, “Castelo-Infantil” and “ Hello-CW ”, which causes the error “too many arguments ”.

To fix this, you must type the cd command followed by just a directory name. If “castelo-infantil” is the name of your project folder and “ hello-cw ” is a subfolder within it, you must first change to the “castelo-infantil” directory and then to “ hello-cw ”. Here is how you can do this:

children's castle cd

CD hello-cw

Or, if “ hello-cw ” is not a subfolder and you were trying to type something after entering the “child-castle” directory, make sure you are typing commands separately. If “ hello-cw ” is a command or a file you want to execute, first change to the directory and then execute the command or file.

If you need more help or have other questions, I'm here to help! 🏰✨

**You, ~/child-castle$ cd hello-cw - bash : cd : hello-cw : No such file or directory ?**

Answers for you…

It appears that the “ hello-cw ” directory does not exist within the “infantil-castle” directory. This can happen if the directory name is incorrect or if it actually does not exist. Let's check this carefully:

1. **List the Directories and Files** : In the “castelo-infantil” directory, use the ls command to list all directories and files present. This will show us what is inside the “children's castle”.
2. ls
3. **Check the Names** : Look at the names of the directories and files that appear after you type ls . If you find a directory that is “ hello- cw ”, make sure you type the name exactly as it appears, respecting upper and lower case letters.
4. **Change to the Correct Directory** : If you find the directory you are looking for, use the cd command followed by the correct name to enter it.
5. CD correct\_directory\_name

If “ hello-cw ” is the name of a project or directory you want to create, you will need to create that directory first using the mkdir command :

mkdir hello-cw

CD hello-cw

This will create a new directory called “ hello-cw ” inside “child-castle” and then switch to it.

If you continue to have issues or have other questions, I'm here to help! 🏰✨

**You, ~/child-castle$ ls Cargo.toml LICENSE NOTICE README.md src ?**

Answers for you…

Perfect, now that you used the ls command and listed the files inside the “children's-castle” directory, we can see what we have here:

* **Cargo.toml** : This is the configuration file for your Rust project . It contains information about your project's dependencies and other settings.
* **LICENSE** : This file contains your project's license, which tells people how they can use your code.
* **NOTICE** : This file often contains legal notices or acknowledgments for third parties.
* **README.md** : This is a Markdown file that generally explains what your project is, how to configure it, and how to use it.
* **src** : This is a folder ( source ) that contains the source code of your project.

Now that we know what each file and folder does, you can start working on your smart contract. If you want to start writing or editing code, you should go to the src folder :

CD src

Inside the src folder, you will find the Rust code files , where you can start programming your smart contract. If you want to create a new code file, you can use a text editor in the terminal or open the files directly in VSCode to edit.

If you need more help or have other questions, I'm here to help! 🏰✨

**You, so we fixed the error in vscode we were looking for?**

Answers for you…

Yes, it looks like we fixed the bug in VSCode . You are now in the correct directory of your “children's castle” project and can see the files and folders that are part of it. With that, you are ready to start working on your smart contract.

If you need more guidance on how to edit the files or the next steps to develop your smart contract, I'm here to help. Remember, if you have any questions, just ask me. Let's continue building your “children's castle” together! 🏰✨

**You, so that I don't get confused, what name do I give to my folder created locally on the PC ?**

Answers for you...

cosmwasm SMART CONTRACT : We can continue the previous topic called " casteli infantil" with these last conversations Perfect, now that you used the ls command and listed the files within the “casteli-infantil” directory, we can see what we have here: Cargo.toml :

This is the configuration file for your Rust project . It contains information about your project's dependencies and other settings.

LICENSE: This file contains your project's license, which tells people how they can use your code.

NOTICE: This file often contains legal notices or acknowledgments for third parties.

README.md: This is a Markdown file that generally explains what your project is, how to configure it, and how to use it. src : This is a folder ( source ) that contains the source code of your project.

Now that we know what each file and folder does, you can start working on your smart contract. If you want to start writing or editing code, you should go to the src folder : cd src Inside the src folder , you will find the Rust code files , where you can start programming your smart contract.

If you want to create a new code file, you can use a text editor in the terminal or open the files directly in VSCode to edit. If you need more help or have other questions, I'm here to help! 🏰✨

**You, so we fixed the error in vscode we were looking for?**

Answers for you…

Yes, it looks like we fixed the bug in VSCode . You are now in the correct directory of your “children's castle” project and can see the files and folders that are part of it. With that, you are ready to start working on your smart contract.

If you need more guidance on how to edit the files or the next steps to develop your smart contract, I'm here to help. Remember, if you have any questions, just ask me. Let's continue building your “children's castle” together! 🏰✨

**So that I don't get confused, what name do I give to my folder created locally on the PC ?**

Answers for you...

I understand that you want to choose a suitable name for your project folder to avoid confusion. Here are some tips for naming your project folder locally on your PC:

1. **Simplicity** : Choose a name that is easy to remember and makes sense for the project context.
2. **Specificity** : Use a name that clearly describes the content or purpose of the project.
3. **Consistency** : Maintain a naming pattern that you are already using for other related projects or folders.
4. **No Spaces** : Avoid using spaces in folder names. Use hyphens (-) or underscores (\_) to separate words if necessary.
5. **No Special Characters** : Avoid using special characters like ! @ # $ % ^ & \* ( ) as they may cause problems with some systems or tools.

Based on these tips, here are some name suggestions for your folder:

* children's\_castle
* intelligent\_contract\_castelo
* castle\_project
* cosmowasm\_castelo

Choose the name that you think is most appropriate and will help you easily identify the project in the future. If you need more help or have other questions, I'm available! 🚀✨

**You, where do I type in vscode CD src ?**

Answers for you...

In Visual Studio Code ( VSCode ), you don't need to type cd src in the text editor. Instead, you will use the **Integrated Terminal** to navigate between folders. Here is how you can do this:

1. Open VSCode and make sure the children's castle project is open.
2. In the top menu, click on **Terminal** and then **New Terminal** . This will open the built-in terminal at the bottom of the VSCode window .
3. In the terminal you will type cd src to change to your project's src folder .

Here is an example of how the command should be typed in the terminal:

CD src

After typing this command and pressing **Enter** , you will be in the src folder where you can start editing your smart contract code files.

If you need anything else, I'm here to help! 🚀✨

**You, where in vscode do I see children's castle project?**

Answers for you...

To preview your children's castle project in Visual Studio Code ( VSCode ), you can follow these steps:

1. Open **VSCode** .
2. Select **File** from the top menu and click **Open Folder…** .
3. Navigate to the location where your children's castle folder is saved on your computer.
4. Select the children's castle folder and click **Open** .

After opening the folder, you will see all the files and subfolders of your project in the **Sidebar** on the left, in the **Explorer section** . If the sidebar is not visible, you can activate it by clicking the two overlapping squares ( 📁 ) icon in the activity bar, which is on the far left of the VSCode window .

There, you can browse your project's files and folders, such as Cargo.toml , LICENSE , NOTICE , README.md , and the src folder .

If you need further assistance or have any other questions, please feel free to ask. I am here to help! 🏰✨

**You, CD src - bash : cd : src : No such file or directory this was the answer to cd src ?**

Answers for you...

The error message - bash : cd : src : No such file or directory indicates that the cd command src was run in a directory where there is no subfolder called src . This can happen for a few reasons:

1. **You may be in the wrong directory** : Make sure you are in the root directory of your castle-child project before trying to access the src subfolder .
2. **The** src folder **may not exist** : Check whether the src folder actually exists within the project directory.
3. **Typo** : A typo may have occurred when entering the command.

To resolve this, you can follow these steps in VSCode 's built-in terminal :

1. Check the current directory with the pwd (print working) command directory ).
2. If you are not in the project directory, navigate to it using the cd command followed by the correct path.
3. Once in the project directory, list the files and folders with ls to confirm the existence of the src folder .
4. If the src folder is present, try accessing it again with cd src .

Here is an example of how you can check and navigate to the src folder :

pwd

# Expected output: /path/to/the/project/children's-castle

ls

# Expected output should list ' src ' among other files and folders

CD src

If the src folder does not exist, you will need to create it with the mkdir command src .

If you continue to experience issues or have other questions, I'm here to help! 🛠️✨​​

**You, this was the way out / mnt /c/ Users /Hearthz Gaming/OneDrive/Área de Trabalho/CONTRATO INTELLIGENTE?**

Answers for you...

The output / mnt /c/ Users /Hearthz Gaming/OneDrive/Desktop/SMART CONTRACT indicates that you are in the SMART CONTRACT directory on your Desktop. If the src folder is inside this directory, you can access it with the cd command src . Otherwise, you will need to navigate to the correct directory where the src folder is located.

Here are the steps you can follow:

1. List the files and folders in the current directory with the ls command to check if the src folder is present.
2. If the src folder is present, type cd src to enter it.
3. If the src folder is not present, you will need to check whether you are in the correct project directory or whether you need to navigate to another location where the src folder may be.

src folder is still not appearing, it may not have been created or has been deleted. In this case, you can create the src folder again with the mkdir command src .

Here is an example of how to list contents and change to the src folder :

ls

# Expected output: list of files and folders

CD src

src ' folder if it exists

If you need further assistance or have other questions, I'm happy to help! 🚀✨

**Do you, remember the conversations about the children's castle earlier today?**

Answers for you...

Yes, I remember our previous conversations about the children's castle project . We discuss your project's directory structure and how to navigate and edit files in VSCode . If you have more questions or need help with something specific related to your smart contract project,

**You, when it's time to make adjustments again, you you already asked me to check src and got the response that it does not exist or not found. in the ls command in vscode it still outputs / mnt / c/ Users /Hearthz Gaming/OneDrive/Área de Trabalho/CONTACTINTELLIGENT so be careful when carrying out the previous checks so as not to repeat the same paths and mistakes, respecting the three rules I mentioned, to stay Of course?**

Answers for you...

Got it, let's proceed carefully to avoid repeating the same mistakes. If the ls command is showing / mnt /c/ Users /Hearthz Gaming/OneDrive/Desktop/SMART CONTRACT as output, it means you are in the SMART CONTRACT directory on your Desktop.

Here are the steps you should follow to check if the src folder is present and to avoid the above errors:

1. **Check Current Directory** : Make sure you are in the correct directory where the children's castle project should be. Use the pwd command to print the current working directory.
2. **List Directory Contents : Use the** ls command to list all files and folders in the current directory. This will help confirm whether the src folder is present or not.
3. **Check the Path** : If the src folder is not present, you may have saved the children's castle project in a different location. Check other directories where the project may be located or if you have multiple copies of the project.
4. src **Folder if Necessary** : If the src folder does not really exist in the project directory, you can create it with the mkdir command src .
5. **Use Absolute Paths if Necessary** : If you are unsure about the relative path, you can navigate using the absolute path to the project folder.

Here's an example of how you can use the commands in VSCode 's built-in terminal :

pwd

# Expected output: / mnt /c/ Users /Hearthz Gaming/OneDrive/Desktop/SMART CONTRACT

ls

# Check if ' src ' is listed

mkdir src

# Create the ' src ' folder if it does not exist

Remember the **three rules** you mentioned:

* **Simplicity** : Keep names simple and clear.
* **Specificity** : Make sure the names are specific to your project.
* **Consistency** : Use a consistent naming pattern.

Thank you for reiterating the rules and focus of our Children's Castle project . It's great to see that you are committed to a clear and structured learning process. I will summarize our principles to ensure we are aligned:

1. **Children's Story** : We will continue to use **simple analogies** to make complex concepts more understandable.
2. **From Practice** : We will detail the **practical steps** so that each action is understood without ambiguity.
3. **Tool Integration Errors** : We will address errors in a **structured way** , identifying the cause and proposing clear solutions.
4. **Children's Castle** : Each new interaction is a chance to **build on existing knowledge** , strengthening the foundation of learning.

To make learning even easier, creating **additional resources** , such as videos and explanatory texts, is an excellent idea. They can serve as visual and textual guides that complement the information we discuss here.

As for **automating complex processes** , this can be incredibly useful for beginners, allowing them to focus on learning programming logic without worrying excessively about setting up the environment.

If you have a specific idea for a feature or an automation suggestion, or if you need help with a specific step in the project, please share it with me. I'm here to support you every step of the way. 🚀✨

**You, one more question before continuing because in the Ubuntu terminal the output of my project name is like children's castle and in the vscode terminal Is it giving another output name like this / mnt /c/ Users /Hearthz Gaming/OneDrive/Área de Trabalho/CONTRATO SMARTINTE?**

Answers for you...

The difference in directory names between the Ubuntu terminal and the built-in VSCode terminal could be due to several factors. I'll explain what could be happening:

1. **Different Terminal Instances** : The Ubuntu terminal and the VSCode built-in terminal can be open in different directories. Each can start in the directory that has been set as the default for that terminal instance.
2. **WSL (Windows Subsystem for Linux) Integration** : If you are using WSL on Windows, the path / mnt /c/ Users /Hearthz Gaming/OneDrive/Desktop/SMART CONTRACT is the Windows file system path as seen from the Ubuntu. This means that the Ubuntu terminal is accessing Windows files through WSL, which mounts the Windows file system to / mnt /c .
3. **Open Project in VSCode** : VSCode can be configured to open directly in the SMART CONTRACT project directory , which is on your Desktop. This would explain why VSCode 's built-in terminal shows this path.

To ensure that both endpoints are in the same project directory, you can:

* In the Ubuntu terminal, navigate to the children's castle project directory using the cd command followed by the correct path.
* In the integrated VSCode terminal, check that the children's castle project is open and that the terminal is pointing to the correct directory.

VSCode 's built-in terminal to always open in the castle-child project directory , you can configure this in VSCode **Settings** by setting the default directory for the terminal.

I hope this clarifies the situation. If you need more help configuring the terminals or have other questions, I'm here to help! 🛠️✨​​

**You, does this interfere with the src command you is it guiding me or not . They are just different names and what precautions should I take?**

Answers for you...

Different directory names between the Ubuntu terminal and the built-in VSCode terminal should not mess up the cd command src that I'm guiding, as long as you're in the correct directory in your project. Here are the precautions you should take:

1. **Check Current Directory** : Before running cd command src , use the pwd command to ensure you are in the root directory of your castle-child project .
2. **Directory Consistency** : Make sure that both the Ubuntu terminal and the built-in VSCode terminal are pointing to the same project directory.
3. **Correct Navigation** : If you are in a different directory, navigate to the project directory before trying to access the src subfolder .
4. **Absolute Paths vs. Relative** : If there is confusion with paths, consider using absolute paths to avoid errors.
5. **Folder Creation** : If the src folder does not exist, you can create it with the mkdir command src in the correct project directory.
6. **Using the Built-in Terminal** : When using VSCode , remember that the built-in terminal reflects the project directory that is currently open in the editor.

Here is an example of how to check and navigate to the src folder in VSCode 's built-in terminal :

pwd

# Expected output: path to the 'children's castle' project directory

ls

# Check if ' src ' is listed

CD src

src ' folder if it exists

Remember that directory names are just references to the locations where your files are stored. The important thing is to ensure that you are working within your project directory so that the commands work as expected.

If you need more help or have other questions, I'm here to help! 🚀✨

**You, this is where the problem goes back to some previous conversations, if each different terminal points to an exit , wouldn't that be a problem, as the directory can be the same with different names?**

Answers for you...

You are correct in pointing out that different terminals may show different paths to the same directory due to the way operating systems and environments, such as WSL (Windows Subsystem for Linux), map and access files. This isn't necessarily a problem as long as you are aware of how paths are represented in each environment and know how to navigate to the desired location.

Here are some tips for managing this:

1. **Understand Paths** : Familiarize yourself with your operating system's directory structure and how paths are displayed in the Ubuntu terminal and VSCode 's built-in terminal .
2. **Relative vs. Relative Paths Absolutes** : Use relative paths when navigating within the same directory or subdirectories. Use absolute paths to navigate between directories that are not directly related.
3. **Synchronize Environments** : If possible, synchronize the working directory between endpoints to avoid confusion. For example, if you are using WSL, you can set the same working directory for both the Ubuntu terminal and the built-in VSCode terminal .
4. **Double Check** : Always check the current directory with pwd before running path-dependent commands like cd src .
5. **Documentation** : Keep clear documentation of your project, including the directory structure and where important files are located, for easy navigation.

Remember, the important thing is that you know where your project files are located and how to access them from whatever terminal you are using. If you have more questions or need help setting up your development environment, I'm here to help. 🛠️✨​​

**You, come on, now I found src to start deploying the smart contract. What next step? should I worry about this too cw -tools: An npm package that helps us create our smart contract in wasm files and even generates schematics for us?**

Answers for you...

Great, now that you've found the src folder , you're ready to start implementing the smart contract. Here are the next steps:

1. **Project Structure : Inside the** src folder , create the necessary files for your smart contract. Typically this includes a lib.rs file for the contract logic and possibly other modules depending on the complexity of your project.
2. **Write the Code** : Start writing Rust code for your smart contract. Define input functions (such as init , handle , and query ) and associated business logic.
3. **Tests** : Write tests to verify the behavior of your contract. This can be done using Rust 's built-in testing framework .
4. **Compilation** : Compile your smart contract for WebAssembly ( Wasm ) using the cargo build --release command . This will generate a file . wasm in the target/wasm32-unknown-unknown/release folder .

As for cw -tools , it is a useful tool that can help in your smart contract development process. Here are some reasons to consider using cw -tools :

* **Wasm File Generation** : Facilitates compilation of your smart contract into Wasm files , which are required to deploy to the blockchain .
* **Schema Generation** : Automates the creation of JSON schemas for your messages and states, which is useful for interacting with the contract and generating compatible clients.
* **Code Check** : Can help verify that your code is following CosmWasm best practices and standards .

To use cw -tools you will need to install it via npm with the npm command install -g cw -tools . Once installed, you can use its commands to assist in developing your contract.

Here is an example of how you might structure the lib.rs file :

**Rust**

#[ cfg ( not ( feature = " library " ))]

use cosmwasm\_std::entry\_point;

use cosmwasm\_std::{to\_binary, Binary, Deps, DepsMut, Env, MessageInfo, Response, StdResult};

#[entry\_point]

pub fn instantiate(

deps : DepsMut ,

\_ env : Env ,

\_info : MessageInfo ,​

\_msg: InstantiateMsg ,

) -> StdResult <Response> {

// Code for the instantiate function

Ok ( Response:: default ())

}

#[ entry\_point ]

pub fn execute (

deps : DepsMut ,

\_ env : Env ,

\_info : MessageInfo ,​

\_msg: ExecuteMsg ,

) -> StdResult <Response> {

// Code for the execute function

Ok ( Response:: default ())

}

#[ entry\_point ]

pub fn query ( deps : Deps , \_env : Env , \_msg: QueryMsg ) -> StdResult < Binary > {

// Code for the query function

to\_ binary ( & "Query Response" )

}

AI-generated code. Examine and use with caution. [More information in frequently asked questions](https://www.bing.com/new#faq) .

Remember to replace the placeholders ( InstantiateMsg , ExecuteMsg , QueryMsg ) with the message types you will define for your contract.

If you need more help or have other questions, I'm here to help! 🚀✨

**Você, npm install -g cw-tools npm ERR! code E404 npm ERR! 404 Not Found - GET https://registry.npmjs.org/cw-tools - Not found npm ERR! 404 npm ERR! 404 'cw-tools@\*' is not in this registry. npm ERR! 404 npm ERR! 404 Note that you can also install from a npm ERR! 404 tarball, folder, http url, or git url. npm ERR! A complete log of this run can be found in: /home/rain/.npm/\_logs/2024-04-24T18\_17\_54\_072Z-debug-0.log?**

Answers for you...

It looks like you encountered an error when trying to install the cw -tools package using npm , and the package was not found in the npm registry . This can happen for a number of reasons, such as the package being removed or renamed.

However, when researching cw -tools , I found that there may be a misunderstanding. The [cwtools package available on npm is a tool for node development and is not related to the development of CosmWasm](https://www.npmjs.com/package/cwtools) [1 smart contracts](https://www.npmjs.com/package/cwtools) . [Additionally, there is a library called cwtools that is used for parsing, editing, and validating Paradox Interactive](https://www.npmjs.com/package/cwtools) [2](https://github.com/cwtools/cwtools) [script files](https://www.npmjs.com/package/cwtools) .

For development of CosmWasm smart contracts , you may need specific tools from the CosmWasm ecosystem . Typically, the process involves:

1. **Compile the Contract** : Use cargo build to compile the smart contract into a Wasm file .
2. **Test the Contract** : Write and run tests for your smart contract.
3. **Optimize Wasm** : After compilation, use tools like wasm-opt to optimize the Wasm file .
4. **Generate Schemas** : Use tools like cargo schema to generate schemas for contract messages.

CosmWasm smart contracts , I recommend checking the official CosmWasm documentation or looking for specific packages that are recommended for this purpose.

If you need further assistance with developing your smart contract or have other questions, I'm here to help! 🚀✨

###### know more

[1 npmjs.com](https://www.npmjs.com/package/cwtools) [2 github.com](https://github.com/cwtools/cwtools) [3 cwtools.github.io](https://cwtools.github.io/) [4 github.com](https://github.com/draconas1/stellaris-tech-tree)

**You, how to check this To install the cwtools package it is necessary to have node js ≥16.0.0 installed on your system?**

Answers for you…

To check if you have Node.js version 16.0.0 or higher installed on your system, you can follow these steps:

1. Open **Terminal** or **Command Prompt** .
2. Type the following command and press **Enter** :

node --version

1. The terminal will display the version of Node.js installed on your system. For example, you will see something like v16.13.0 if you have version 16.13.0 installed.

If the version displayed is 16.0.0 or higher, then you meet the requirement to install the cwtools package . Otherwise, you will need to update Node.js to a newer version.

Remember, it's important to have the correct version of Node.js to ensure compatibility with the packages you want to install and use. If you need more help or have other questions, I'm here to help! 🚀✨

###### know more

[1 stackoverflow.com](https://stackoverflow.com/questions/44515772/how-do-i-know-which-version-of-node-js-i-am-currently-running) [2 masteringjs.io](https://masteringjs.io/tutorials/node/version) [3 foreignerds.com](https://foreignerds.com/how-to-check-if-node-js-is-installed-on-windows/) [4 7.dev](https://7.dev/how-to-check-the-installed-version-of-node-js/) [5 github.com](https://github.com/Homebrew/brew/blob/master/docs/FAQ.md)

**You, did this version come out v21.7.3?**

Answers for you…

The v21.7.3 version you mentioned is a newer version of Node.js than needed to install the cwtools package . This means your system is up to date and you can proceed with installing packages that require Node.js version 16.0.0 or higher.

With this version, you are well equipped to work with the most modern tools and packages available for smart contract development on CosmWasm or other platforms using Node.js.

If you have more questions or need assistance with next steps, I'm here to help! 🚀✨

**You, yarn global add @ oraichain / cwtools Command ' yarn ' not found , but can be installed with : sudo apt install cmdtest help me see this?**

Answers for you…

The message you received indicates that **Yarn** is not installed on your Ubuntu system. cmdtest is a different package and should not be confused with Yarn . To install Yarn , which is a package manager for JavaScript projects , you can follow the steps below:

1. First, update your system's package index with the command:
2. sudo apt update
3. Then install Yarn using the following command:
4. sudo apt install yarn

If you want to install a specific version of Yarn or ensure you are getting the latest version, you can add the official Yarn repository to your system and install from there. Here are the steps for this:

1. Import the GPG key from the Yarn repository and add the repository to your system:
2. curl - sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -
3. echo " deb https://dl.yarnpkg.com/debian/ stable main " | sudo tee /etc/apt/sources.list.d/yarn.list​​​​​​​
4. Update the package index again and install Yarn :
5. sudo apt update
6. sudo apt install yarn

After installation, you can verify that Yarn was installed correctly with the command:

yarn --version​

With Yarn installed, you will be able to add the @ oraichain / cwtools package globally using the command you mentioned earlier:

yarn global add @ oraichain / cwtools

If you need further assistance or have other questions, I'm here to help! 🚀✨

###### know more

[1 youtube.com](https://www.youtube.com/watch?v=S1Q59xxoMWw) [2 youtube.com](https://www.youtube.com/watch?v=pyvurlqIMMM) [3 youtube.com](https://www.youtube.com/watch?v=26h4Ct6jou0) [4 classic.yarnpkg.com](https://classic.yarnpkg.com/lang/en/docs/install/) [5 yarnpkg.com](https://yarnpkg.com/getting-started/install) [6 digitalocean.com](https://www.digitalocean.com/community/tutorials/js-yarn-package-manager-quick-intro) [7 yarnpkg.com](https://yarnpkg.com/install.sh) [8 github.com](https://github.com/oraichain/cosmwasm-tools) [9 github.com](https://github.com/oraichain) [10 github.com](https://github.com/lievelfilanchris/cosmwasm-tools/blob/main/README.md) [11 orai.io](https://orai.io/) [12 linuxize.com](https://linuxize.com/post/how-to-install-yarn-on-ubuntu-20-04/) [13 linuxcapable.com](https://www.linuxcapable.com/how-to-install-yarn-on-ubuntu-linux/) [14 webhi.com](https://www.webhi.com/how-to/how-to-install-yarn-ubuntu-linux/) [15 itsfoss.com](https://itsfoss.com/install-yarn-ubuntu/) [16 itslinuxfoss.com](https://itslinuxfoss.com/install-yarn-ubuntu-22-04/) [17 dl.yarnpkg.com](https://dl.yarnpkg.com/debian/pubkey.gpg) [18 dl.yarnpkg.com](https://dl.yarnpkg.com/debian/) [19 deb.nodesource.com](https://deb.nodesource.com/setup_current.x) [20 deb.nodesource.com](https://deb.nodesource.com/setup_lts.x) [21 github.com](https://github.com/CodeSigils/webpack4-react16-setup/tree/cc299b2b534851024e0ea549d9e129f432d07fe8/README.md) +15 more

**You, after this operations , 4875 kB of additional disk space will be used . Do you want to continue? [Y/n] which option?**

Answers for you…

To continue with the installation, you must choose option **Y** (yes). This will confirm that you are happy to use the additional disk space to install Yarn . Here's what you should type into the terminal:

Y

After pressing **Enter** , the installation will proceed. If you do not wish to continue, you can choose option **n** (no), and the installation will be canceled.

If you need further assistance or have other questions, I'm here to help! 🚀✨

**You, command executed in vscode yarn global add @ oraichain / cwtools 00h00m00s 0/0 : : ERROR: [ Errno 2] No such file or directory : 'global' after checking all the steps you suggested previously that worked?**

Answers for you…

It looks like you encountered an error when trying to use the yarn global add command . [According to the information I found, starting with Yarn version 2.x, the global command was removed](https://stackoverflow.com/questions/40317578/yarn-global-command-not-working) [1](https://stackoverflow.com/questions/40317578/yarn-global-command-not-working) [2](https://stackoverflow.com/questions/70756959/global-parameter-does-not-work-with-yarn-3) . Instead, you can use the yarn command dlx to run packages globally without installing them.

Here is how you can use yarn dlx to run the @ oraichain / cwtools package :

yarn dlx @ oraichain / cwtools

This command will run the package directly, without the need to install it globally. If you still want to install the package globally, you can consider using npm or check if there is a third-party plugin for Yarn that allows global installation.

If you need further assistance or have other questions, I'm here to help! 🚀✨

###### know more

[1 stackoverflow.com](https://stackoverflow.com/questions/40317578/yarn-global-command-not-working) [2 stackoverflow.com](https://stackoverflow.com/questions/70756959/global-parameter-does-not-work-with-yarn-3) [3 dev.to](https://dev.to/monierate/yarn-command-not-found-on-windows-10-36pe) [4 bobbyhadz.com](https://bobbyhadz.com/blog/npm-command-not-found-yarn) [5 cli.vuejs.org](https://cli.vuejs.org/guide/installation.html)

**You, 00h00m00s 0/0 : : ERROR: [ Errno 2] No such file or directory : ' dlx '?**

Answers for you…

It looks like you are facing an issue with the yarn command dlx , which is resulting in a “No such file or directory ”. This can occur if the installed version of Yarn does not support the dlx command . To resolve this issue, you can follow these steps:

1. **Uninstall Current Yarn** : If you installed Yarn using brew or npm , uninstall it.
2. npm uninstall -g yarn
3. # or
4. brew uninstall yarn
5. **Enable Corepack** : Corepack is a tool that allows you to manage Yarn and pnpm versions . Enable Corepack with the following command:
6. corepack enable
7. **Install the Desired Version of Yarn** : With Corepack enabled, install the latest version of Yarn or the specific version you want.
8. corepack install --global yarn@latest
9. # or for a specific version
10. corepack install --global yarn@xyz
11. **Yarn Version** : After installation, verify that the correct version of Yarn is installed.
12. yarn --version​
13. **Try the Command Again** : With the correct version of Yarn installed, try adding the @ oraichain / cwtools package again.
14. yarn global add @ oraichain / cwtools

[If you have a recent installation of Node.js, Corepack should already be included, but if not, you can install it manually with npm install -g corepack](https://stackoverflow.com/questions/78161928/yarn-dlx-not-found-why-how-to-run-yarn-dlx-command) [1](https://stackoverflow.com/questions/78161928/yarn-dlx-not-found-why-how-to-run-yarn-dlx-command) .

I hope these steps help resolve the issue. If you need further assistance or have other questions, I'm here to help! 🚀✨

###### know more

[1 stackoverflow.com](https://stackoverflow.com/questions/78161928/yarn-dlx-not-found-why-how-to-run-yarn-dlx-command) [2 yarnpkg.com](https://yarnpkg.com/cli/dlx) [3 bobbyhadz.com](https://bobbyhadz.com/blog/npm-command-not-found-yarn) [4 yarnpkg.com](https://yarnpkg.com/en) [5 nodejs.org](https://nodejs.org/api/corepack.html) [6 yarnpkg.com](https://yarnpkg.com/corepack)

**You, verse yarn is this the one that installed 0.32+git?**

Answers for you…

The 0.32+git version of Yarn you mentioned appears to be a very old version or a specific version of Git , which may not have all current features such as the dlx command . To use the latest Yarn features , including the dlx command , you will need to install a newer version of Yarn .

Here are the steps to install the latest version of Yarn :

1. First, remove the current version of Yarn :
2. sudo apt remove yarn
3. Then install Yarn by following the official instructions, which usually involve adding the Yarn repository to your system and installing from there:
4. curl - sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -
5. echo " deb https://dl.yarnpkg.com/debian/ stable main " | sudo tee /etc/apt/sources.list.d/yarn.list​​​​​​​
6. sudo apt update
7. sudo apt install yarn
8. Check the installed version of Yarn to ensure the installation was successful:
9. yarn --version​

With the latest version of Yarn installed, you should be able to use the yarn command dlx and other modern Yarn features without any problems.

If you need further assistance or have other questions, I'm here to help! 🚀✨

###### know more

[1 yarnpkg.com](https://yarnpkg.com/cli/dlx) [2 classic.yarnpkg.com](https://classic.yarnpkg.com/en/package/@yarnpkg/plugin-dlx) [3 stackoverflow.com](https://stackoverflow.com/questions/78161928/yarn-dlx-not-found-why-how-to-run-yarn-dlx-command) [4 github.com](https://github.com/CodeSigils/webpack4-react16-setup/tree/cc299b2b534851024e0ea549d9e129f432d07fe8/README.md) [5 github.com](https://github.com/pandu-49/clearstack-task/tree/7889826317937785fd7417f7a4efa74a7d8a6853/README.md)

It is important to highlight that I had already resolved this issue of choosing between npm and yarn . Giving command only to npm , there is no need for yarn .