

SE-Assignment-5

1. Installation and Navigation of Visual Studio Code (VS Code) Instructions:
Answer the following questions based on your understanding of the installation and navigation of Visual Studio Code (VS Code). Provide detailed explanations and examples where appropriate.

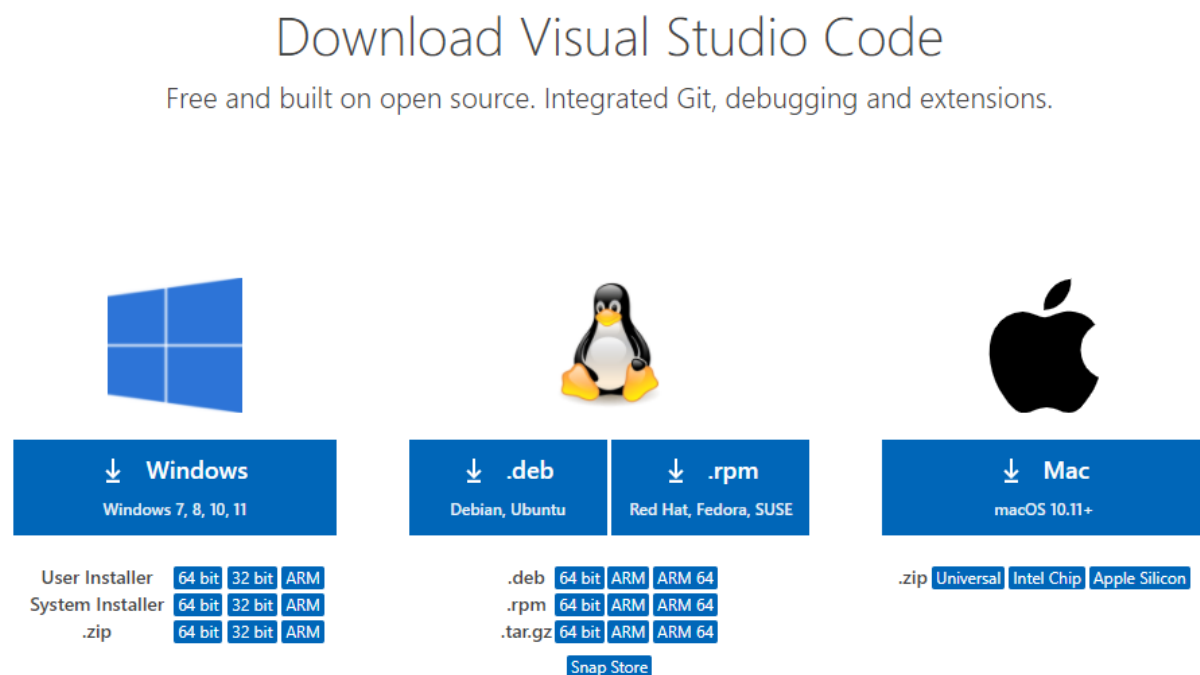
Questions:

Installation of VS Code:

Describe the steps to download and install Visual Studio Code on Windows 11 operating system. Include any prerequisites that might be needed.

Steps to Install Visual Studio Code on Windows


Step 1: Visit the [Official Website](#) of the **Visual Studio Code** using any web browser like [Google Chrome](#), [Microsoft Edge](#), etc.




Step 2: Press the “**Download for Windows**” button on the website to start the download of the Visual Studio Code Application.


Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.


[↓ Windows](#)
Windows 7, 8, 10, 11


[↓ .deb](#)
Debian, Ubuntu

[↓ .rpm](#)
Red Hat, Fedora, SUSE


[↓ Mac](#)
macOS 10.11+

User Installer

System Installer

.zip

64 bit

32 bit

ARM

64 bit

32 bit

ARM

64 bit

32 bit

ARM

.deb

.rpm

.tar.gz

64 bit

ARM

ARM 64

64 bit

ARM

ARM 64

64 bit

ARM

ARM 64

[Snap Store](#)

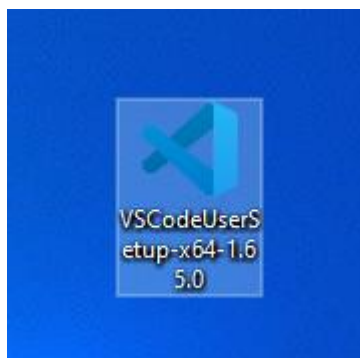
.zip

Universal

Intel Chip

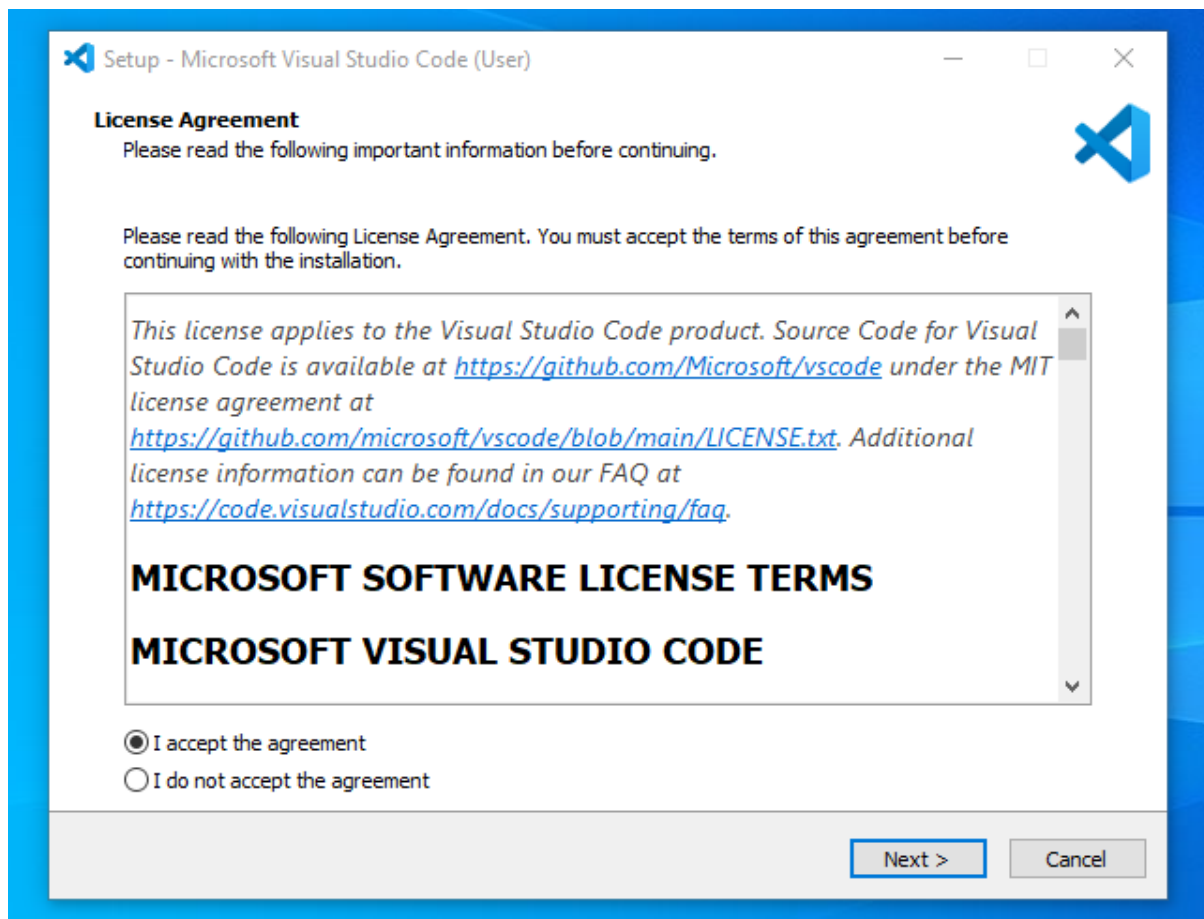
Apple Silicon

Step 3: When the download finishes, then the **Visual Studio Code Icon** appears in the downloads folder.

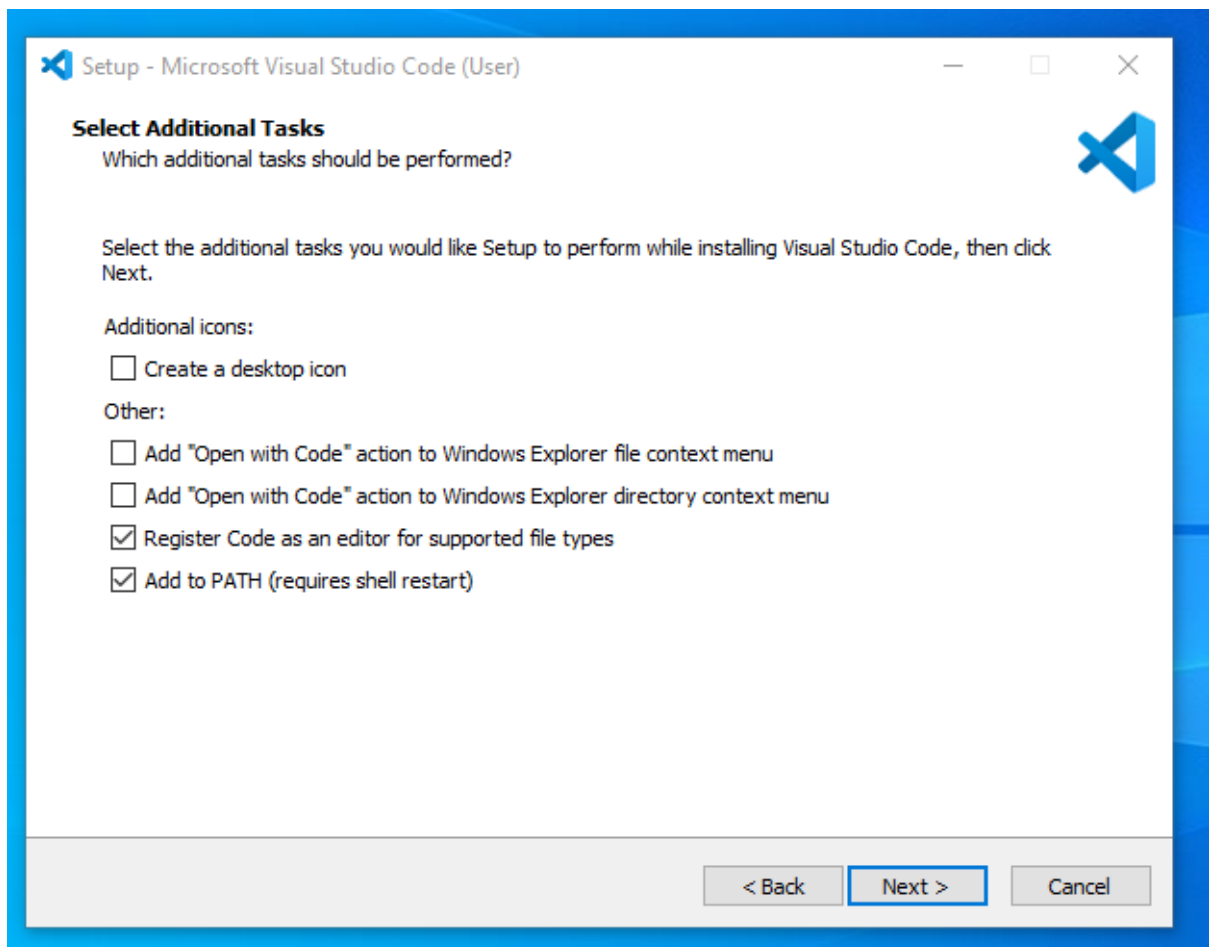


Step 4: Click on the **Installer** icon to start the installation process of the Visual Studio Code.

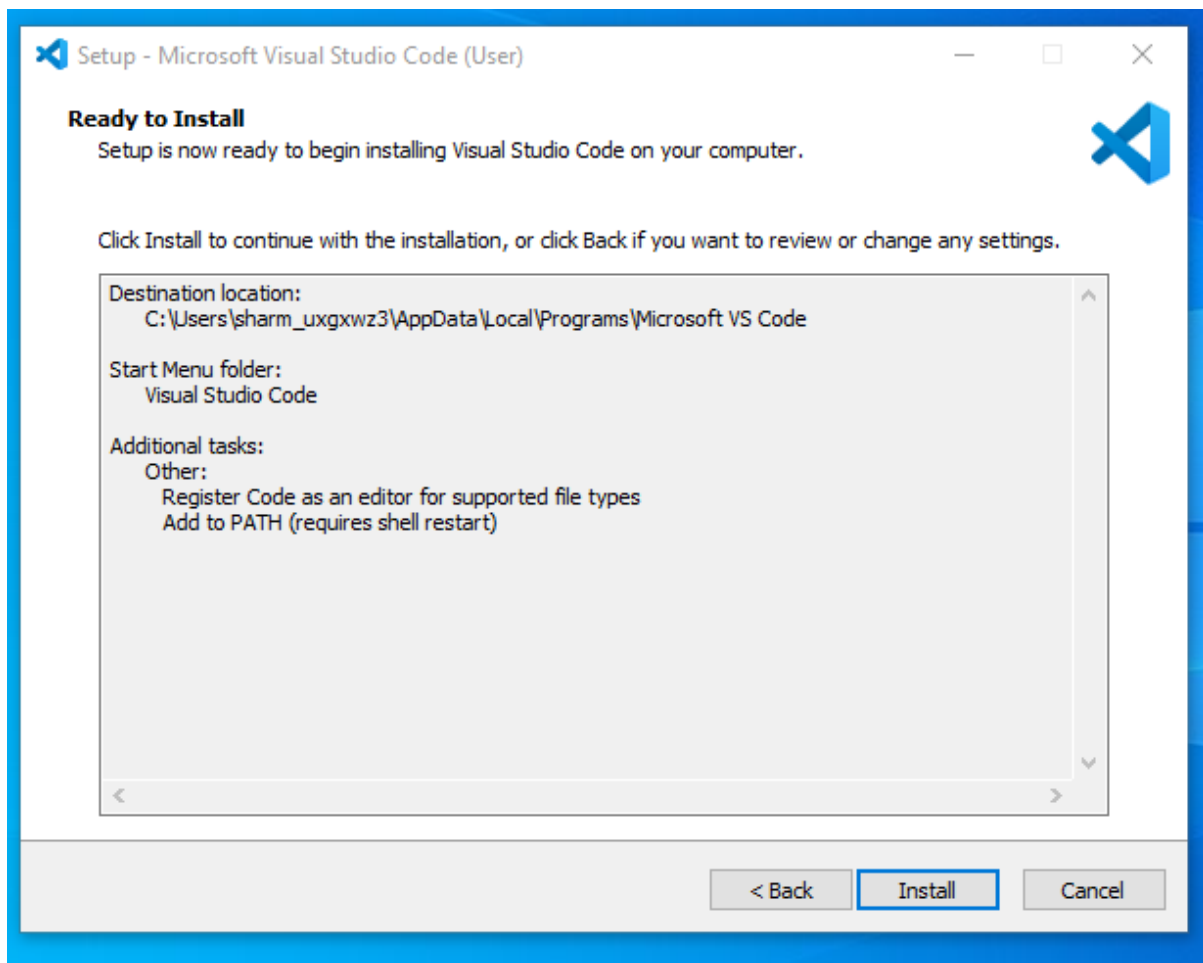
Step 5: After the Installer opens, it will ask you to accept the terms and conditions of the Visual Studio Code. Click on **I accept the agreement** and then click the **Next** button.



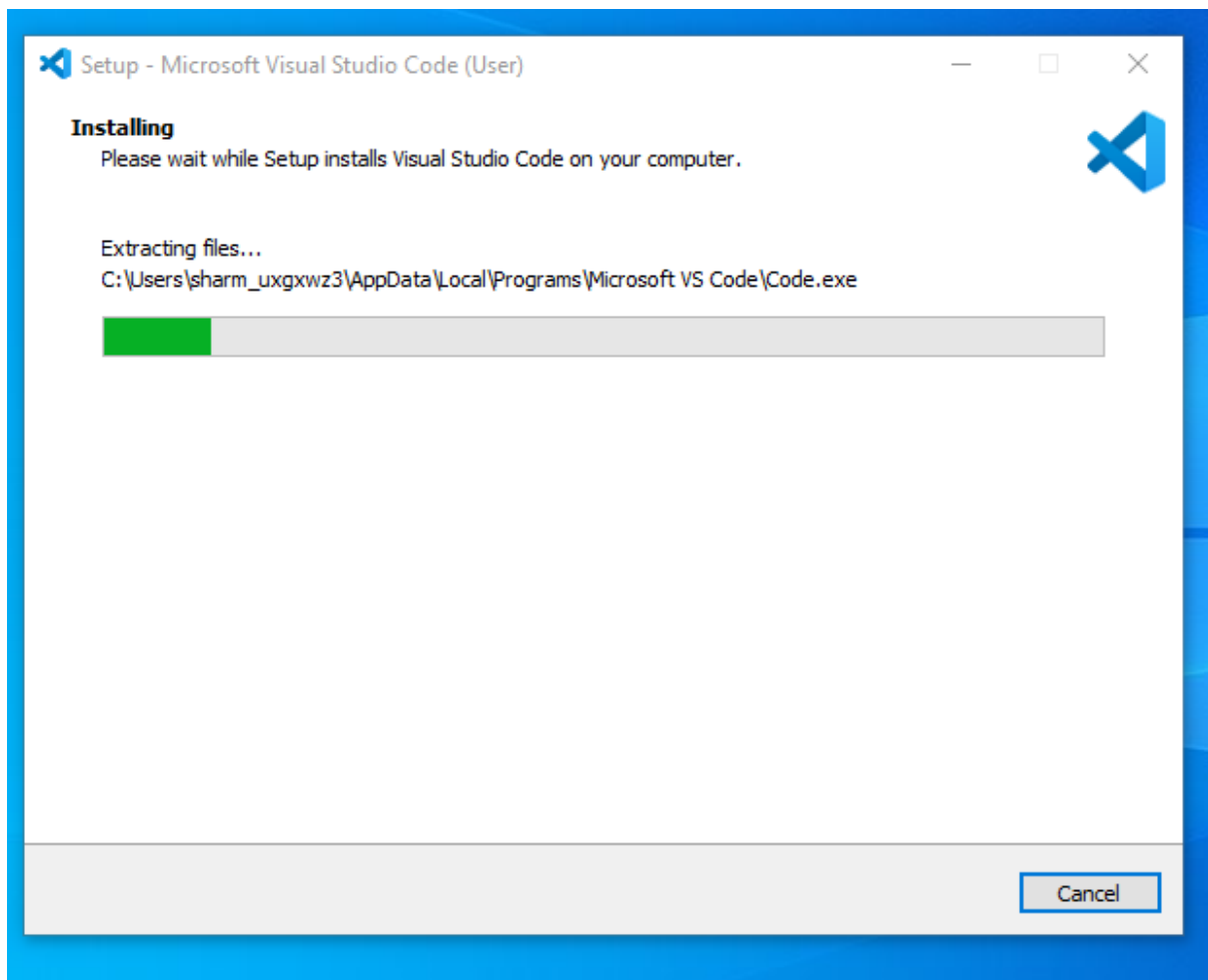
Step 6: Choose the location data for running the Visual Studio Code. It will then ask you to browse the location. Then click on the **Next** button.



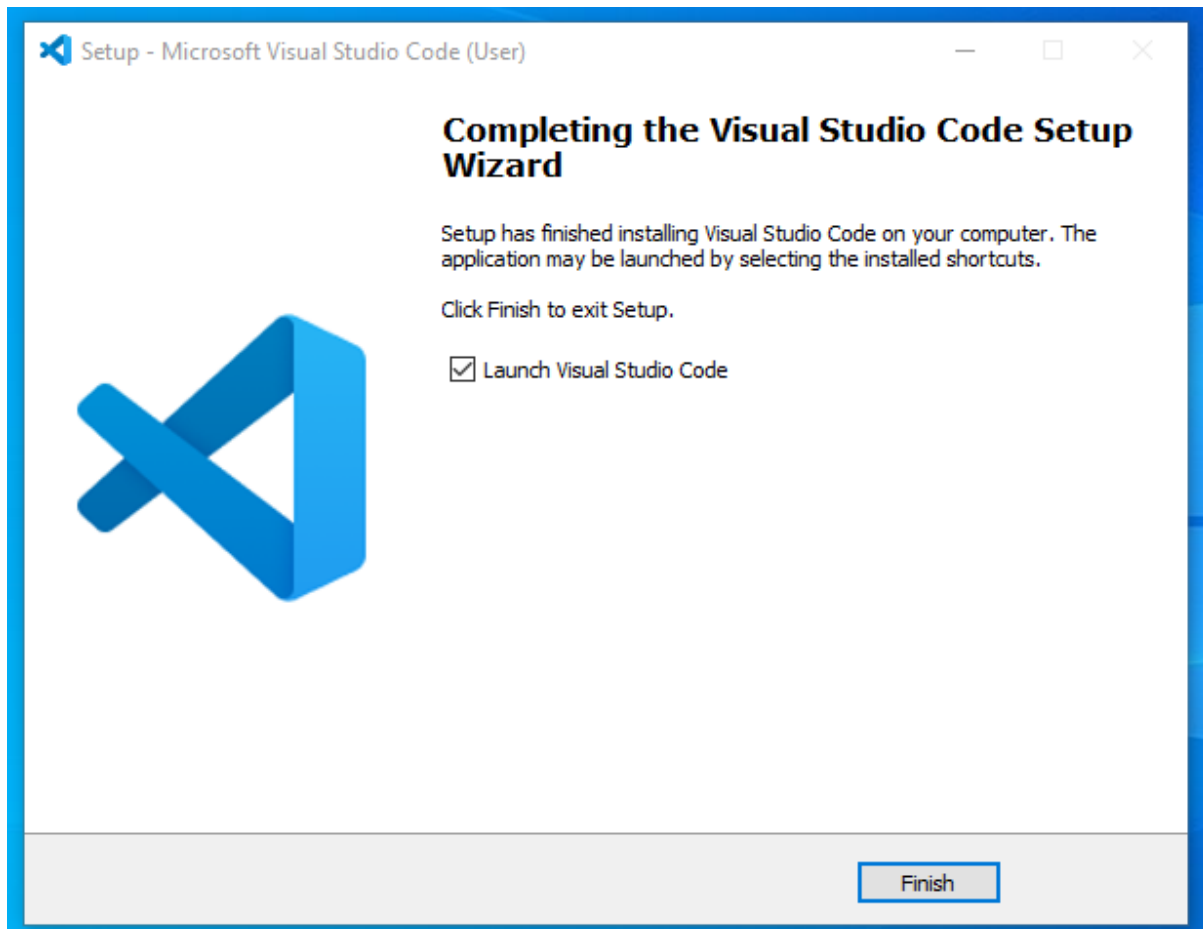
Step 7: Then it will ask to begin the installation setup. Click on the **Install** button.



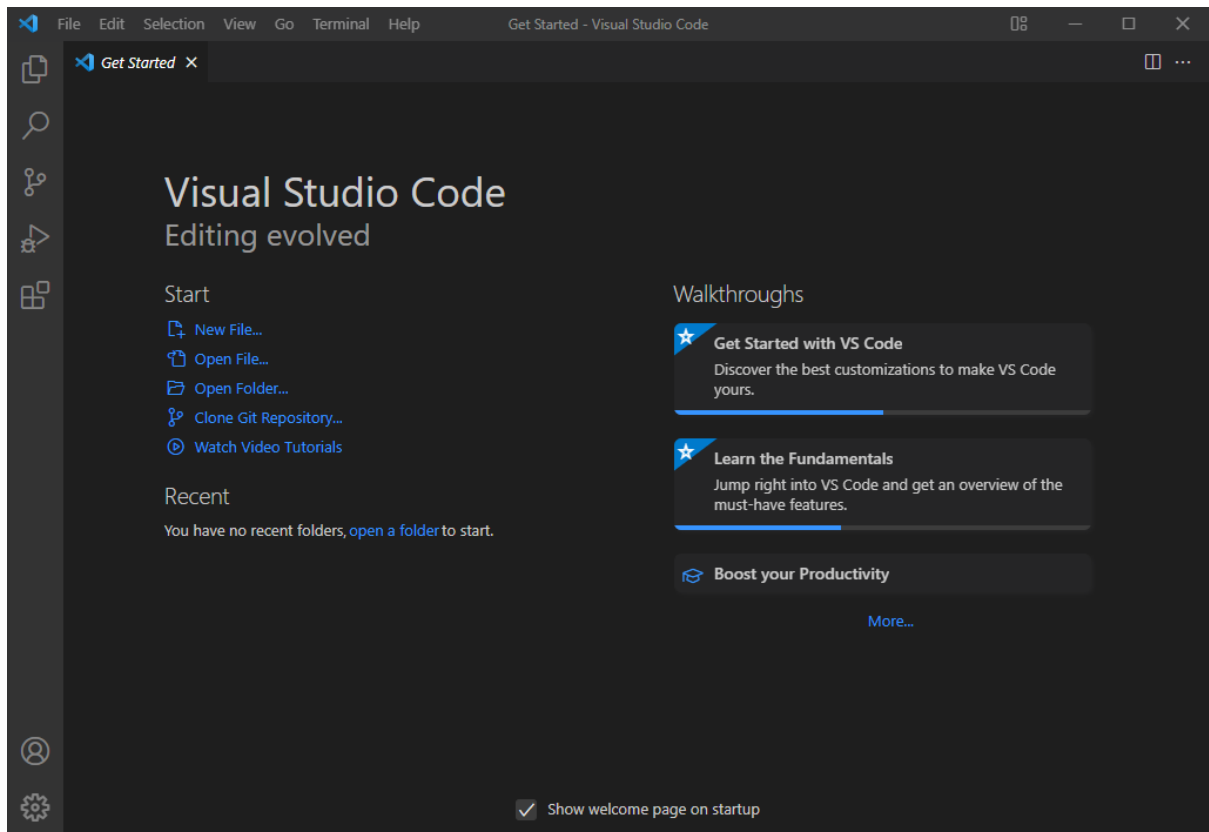
Step 8: After clicking on Install, it will take about 1 minute to install the Visual Studio Code on your device.



Step 9: After the Installation setup for Visual Studio Code is finished, it will show a window like this below. Tick the “**Launch Visual Studio Code**” checkbox and then click **Next**.



Step 10: After the previous step, the **Visual Studio Code window** opens successfully. Now you can create a new file in the Visual Studio Code window and choose a language of yours to begin your programming journey!



So this is how we successfully installed **Visual Studio Code** on our Windows system.

2. First-time Setup:

After installing VS Code, what initial configurations and settings should be adjusted for an optimal coding environment? Mention any important settings or extensions.

Configuring Extensions

The following are the extensions that we need in order to set up VS Code properly. Open VS Code and head over to the Extensions Panel (**Ctrl + Shift + X**) which is on the left toolbar and start downloading the following extensions

1. **Python** by **Microsoft** — You will need to configure Python for this extension to work. Download and install the latest version from [here](#).
2. **Code Runner** — We'll use this extension to run all our programs. This needs some configuration steps to be performed.

Select File -> Preferences -> Settings

Search “code runner run in terminal” in the search box and check the checkbox.

3. User Interface Overview:

Explain the main components of the VS Code user interface. Identify and describe the purpose of the Activity Bar, Side Bar, Editor Group, and Status Bar.

VS Code comes with a simple and intuitive layout that maximizes the space provided for the editor, while leaving ample room to browse and access the full context of your folder or project. The user interface is divided into five main areas:

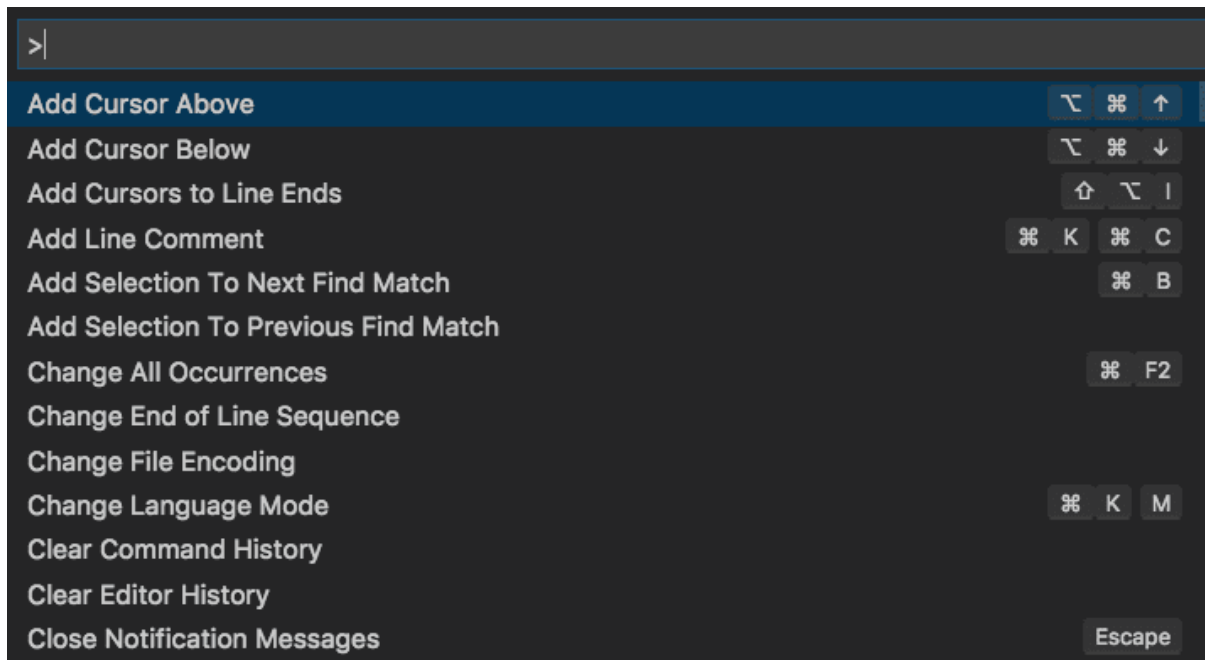
- **Editor** - The main area to edit your files. You can open as many editors as you like side by side vertically and horizontally.
- **Primary Side Bar** - Contains different views like the Explorer to assist you while working on your project.
- **Status Bar** - Information about the opened project and the files you edit.
- **Activity Bar** - Located on the far left-hand side. Lets you switch between views and gives you additional context-specific indicators, like the number of outgoing changes when Git is enabled. You can change the position of the Activity Bar.

4. Command Palette:

What is the Command Palette in VS Code, and how can it be accessed? Provide examples of common tasks that can be performed using the Command Palette.

[Command Palette](#)

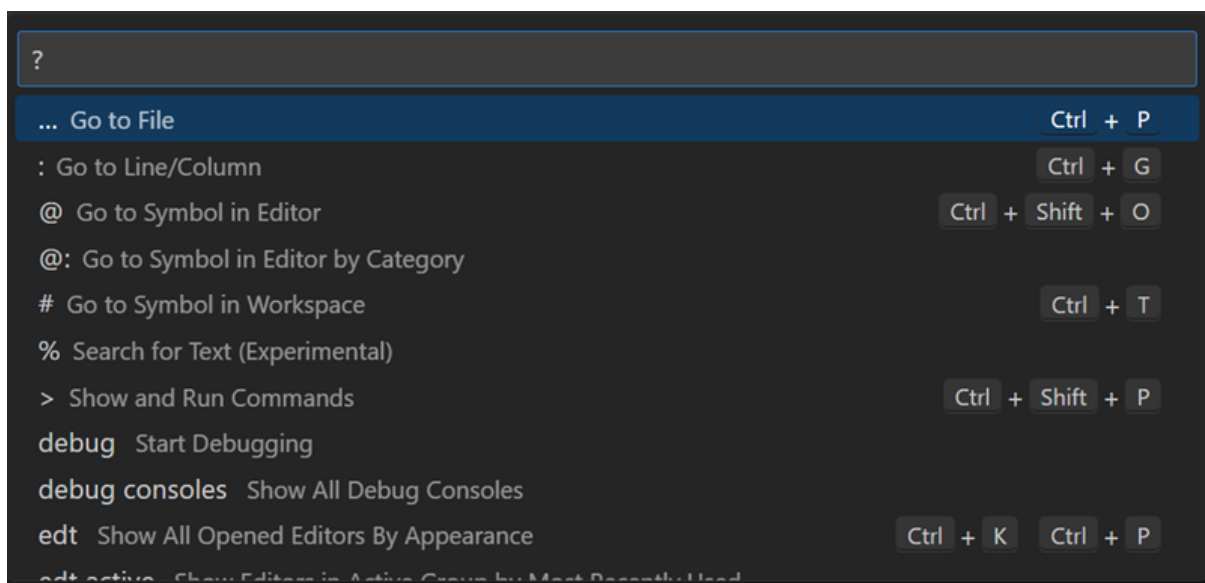
VS Code is equally accessible from the keyboard. The most important key combination to know is Ctrl+Shift+P, which brings up the **Command Palette**. From here, you have access to all functionality within VS Code, including keyboard shortcuts for the most common operations.



The **Command Palette** provides access to many commands. You can run editor commands, open files, search for symbols, and see a quick outline of a file, all using the same interactive window. Here are a few tips:

- Ctrl+P enables you to navigate to any file or symbol by typing its name
- Ctrl+Tab cycles you through the last set of files opened
- Ctrl+Shift+P brings you directly to the editor commands
- Ctrl+Shift+O enables you to navigate to a specific symbol in a file
- Ctrl+G enables you to navigate to a specific line in a file

Type “?” in the input field to get a list of available commands that you can run from the Command Palette.

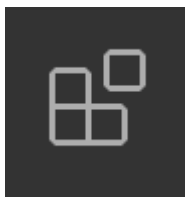


5. Extensions in VS Code:

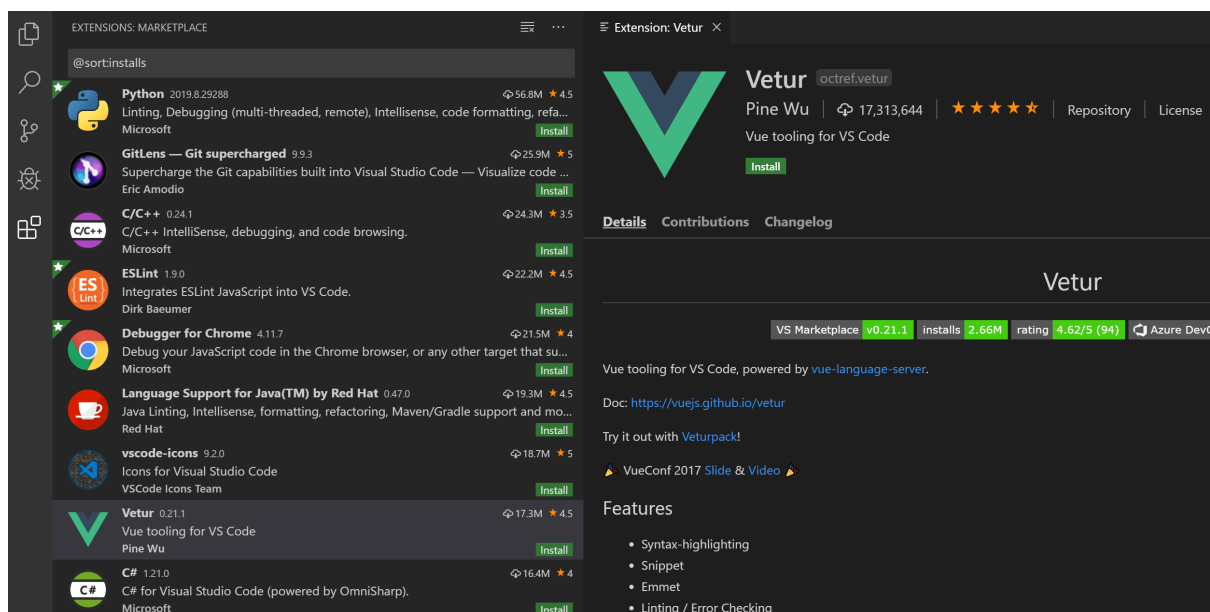
Discuss the role of extensions in VS Code. How can users find, install, and manage extensions? Provide examples of essential extensions for web development.

[Browse for extensions](#)

You can browse and install extensions from within VS Code. Bring up the Extensions view by clicking on the Extensions icon in the **Activity Bar** on the side of VS Code or the **View: Extensions** command (Ctrl+Shift+X).



This will show you a list of the most popular VS Code extensions on the [VS Code Marketplace](#).



Each extension in the list includes a brief description, the publisher, the download count, and a five star rating. You can select the extension item to display the extension's details page where you can learn more.

[Install an extension](#)

To install an extension, select the **Install** button. Once the installation is complete, the **Install** button will change to the **Manage** gear button.

[Manage extensions](#)

VS Code makes it easy to manage your extensions. You can install, disable, update, and uninstall extensions through the Extensions view, the **Command Palette** (commands have the **Extensions:** prefix) or command-line switches.

[List installed extensions](#)

By default, the Extensions view will show the extensions you currently have installed, and all extensions that are recommended for you. You can use the **Extensions: Focus on Installed View** command, available in the **Command Palette** (Ctrl+Shift+P) or in the **More Actions (...)** dropdown menu > **Views** > **Installed**, to clear any text in the search box and show the list of all installed extensions, which includes those that have been disabled.

[Uninstall an extension](#)

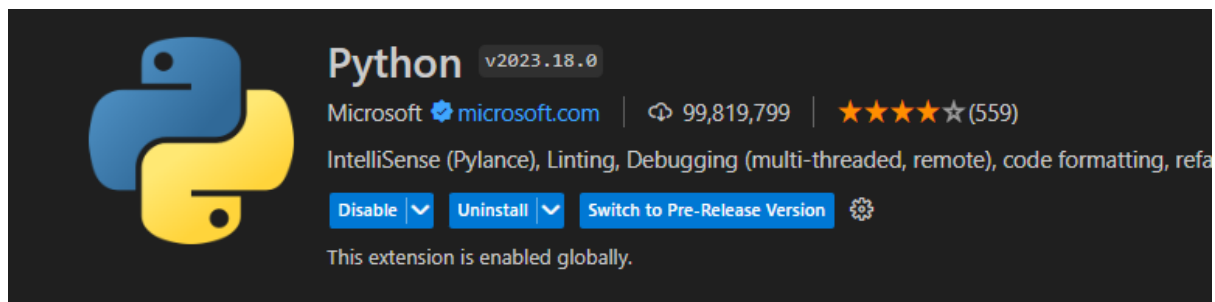
To uninstall an extension, select the **Manage** gear button at the right of an extension entry and then choose **Uninstall** from the dropdown menu. This will uninstall the extension and prompt you to restart the extension host (**Restart Extensions**).

[Find and install an extension](#)

You can search extensions in the extension tab and install them. Examples of essential extensions for web development are python, dart, flutter and Django.

Python

The **Python extension is an essential tool for Python development**, including Django. It delivers extensive support for Python, offering functionalities like code completion, debugging, linting, and code formatting



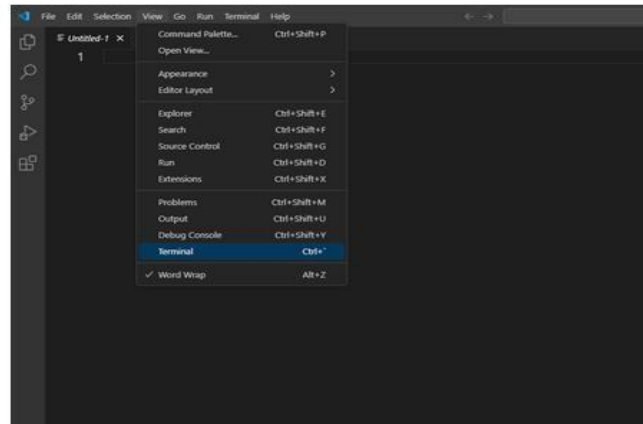
6. Integrated Terminal:

Describe how to open and use the integrated terminal in VS Code. What are the advantages of using the integrated terminal compared to an external terminal?

How do I open a terminal in VS Code on Microsoft Windows?

Using the Menu Bar

To begin, open VS Code on Windows 10/11 and navigate to the menu bar at the top. From there, select the "View" option and then click on "Terminal".



YoungWonks

Using Keyboard Shortcut

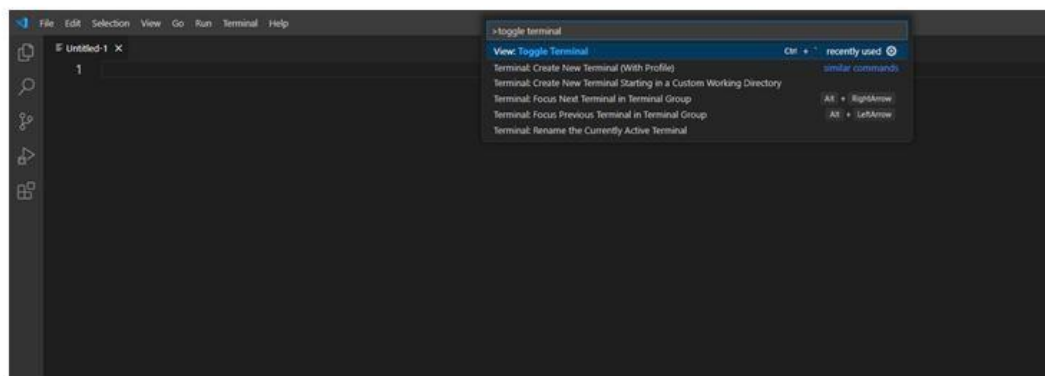
The quickest way to open the terminal in VS Code on Windows is by using the keyboard shortcut:

Ctrl + ` : Pressing Ctrl and the backtick key simultaneously opens the terminal.

Using Command Palette

Another method is through the command palette:

- Press Ctrl + Shift + P to open the command palette.
- Type "toggle terminal" and press Enter to open the terminal.

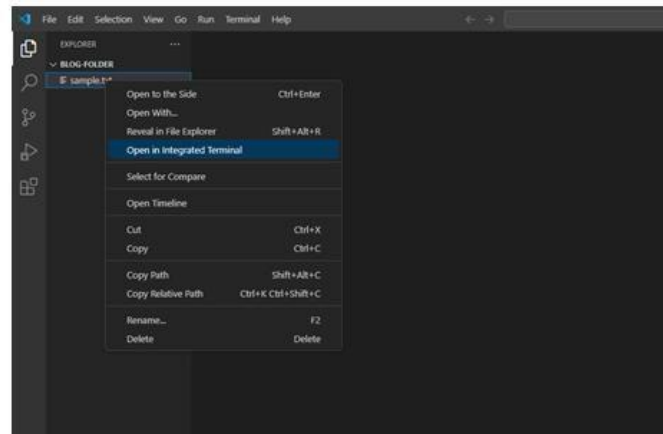


YoungWonks

Using Right-Click Menu

You can also access the terminal via the right-click menu within the workspace:

- In the file explorer or editor, right-click on any file or empty space.
- Select "Open in Integrated Terminal" to open the terminal in the current workspace.



YoungWonks

Once the terminal window appears, you can start using it to run commands and scripts in various languages such as Bash, Python, JavaScript, Java, TypeScript, and more.

One of the key advantages of VS Code's integrated terminal is **its seamless integration with Git commands and version control operations**.

7. File and Folder Management:

Explain how to create, open, and manage files and folders in VS Code. How can users navigate between different files and directories efficiently?

Creating Files and Folders

1. Creating a New File:
 - To create a new file, you can use the following methods:
 - Click on the Explorer icon in the Activity Bar on the side (or use Ctrl+Shift+E or Cmd+Shift+E).
 - Right-click on the parent folder where you want to create the file and select New File.
 - Use the keyboard shortcut Ctrl+N to create a new file directly.

2. Creating a New Folder:

- Similarly, to create a new folder:
 - Right-click on the parent folder in the Explorer and select New Folder.
 - Use the command palette (Ctrl+Shift+P or Cmd+Shift+P) and type "New Folder" to create one.

Opening Files and Folders

1. Opening Files:

- To open an existing file in VS Code:
 - Click on the Explorer icon in the Activity Bar.
 - Navigate to the file you want to open and double-click on it.
 - Alternatively, use Ctrl+P (Cmd+P on macOS) to open the Command Palette and type the name of the file you want to open.

2. Opening Folders:

- To open an entire folder (project) in VS Code:
 - Use File > Open... from the menu bar.
 - Click on Open Folder in the Welcome page or in the File Explorer.
 - Drag and drop a folder into the VS Code window.

Managing Files and Folders

1. Renaming Files and Folders:

- Right-click on the file or folder in the Explorer and choose Rename, or press F2 when the file or folder is selected.

2. Deleting Files and Folders:

- Right-click on the file or folder in the Explorer and select Delete, or press Delete or Backspace when the file or folder is selected. Confirm the deletion if prompted.

3. Moving/Copying Files and Folders:

- To move a file or folder, drag it to the desired location within the Explorer.
- To copy a file or folder, hold down Ctrl (Cmd on macOS) while dragging it to the desired location.

4. Searching within Files:

- Use the search functionality (Ctrl+Shift+F or Cmd+Shift+F) to search for specific text within files in your project.

5. Saving Files:

- Files are saved automatically in VS Code. However, you can use Ctrl+S (Cmd+S on macOS) to manually save changes if needed.
6. Closing Files and Folders:
- To close a file, click on the x next to the file tab at the top of the editor.
 - To close a folder, right-click on the folder in the Explorer and select Remove Folder from Workspace.
- Use the Explorer icon in the Activity Bar (Ctrl+Shift+E or Cmd+Shift+E) to view and navigate through your project's folder structure.
 - You can expand and collapse directories by clicking on the arrows next to folder icons.

8. Settings and Preferences:

Where can users find and customize settings in VS Code? Provide examples of how to change the theme, font size, and keybindings.

[Settings editor](#)

Use the Settings editor to review and change VS Code settings. To open the Settings editor, navigate to **File > Preferences > Settings**. Alternately, open the Settings editor from the **Command Palette** (Ctrl+Shift+P) with **Preferences: Open Settings** or use the keyboard shortcut (Ctrl+,).

When you open the Settings editor, you can search and discover the settings you are looking for.

1. Changing the Theme:

Visual Studio Code comes with several built-in themes, and you can also install additional themes from the VS Code Marketplace.

- **To change the theme:**
 1. Open VS Code.
 2. Go to File > Preferences > Color Theme (on Windows/Linux).
 3. A list of installed themes will appear. Select the theme you want to use. You can preview each theme by clicking on them.
 4. To install new themes, click on the Install Additional Color Themes... link at the bottom of the list and follow the instructions.

2. Changing Font Size and Family:

- **To change the font size and family:**
 1. Open VS Code.
 2. Go to **File > Preferences > Settings** (on Windows/Linux) to open the Settings tab.

3. In the search bar at the top of the Settings tab, type font.
4. You'll see options for Editor: Font Size and Editor: Font Family.
5. Adjust Editor: Font Size to change the size of the text in the editor.
6. Adjust Editor: Font Family to change the font family. You can specify a list of preferred fonts separated by commas.

3. Changing Keybindings:

- **To change keybindings:**
 1. Open VS Code.
 2. Go to **File > Preferences > Keyboard Shortcuts** (on Windows/Linux) or use the shortcut Ctrl+K Ctrl+S to open keyboard shortcut.
 3. This opens the Keyboard Shortcuts tab where you can search for specific commands or keybindings.
 4. To change a keybinding, click on the pencil icon next to the keybinding you want to modify and enter your desired key combination.
 5. You can also override keybindings by copying them to your keybindings.json file. To do this, click on the {} icon in the upper right corner of the Keyboard Shortcuts tab to open keybindings.json.

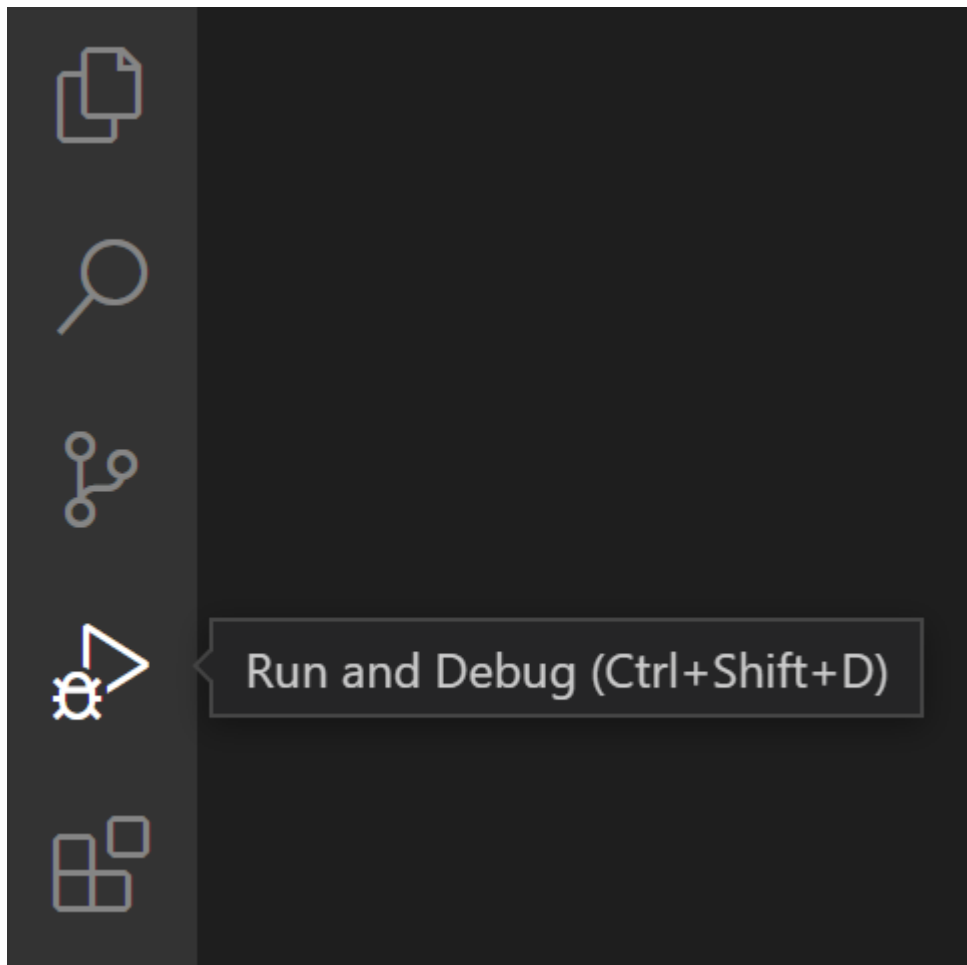
9. Debugging in VS Code:

Outline the steps to set up and start debugging a simple program in VS Code.

What are some key debugging features available in VS Code?

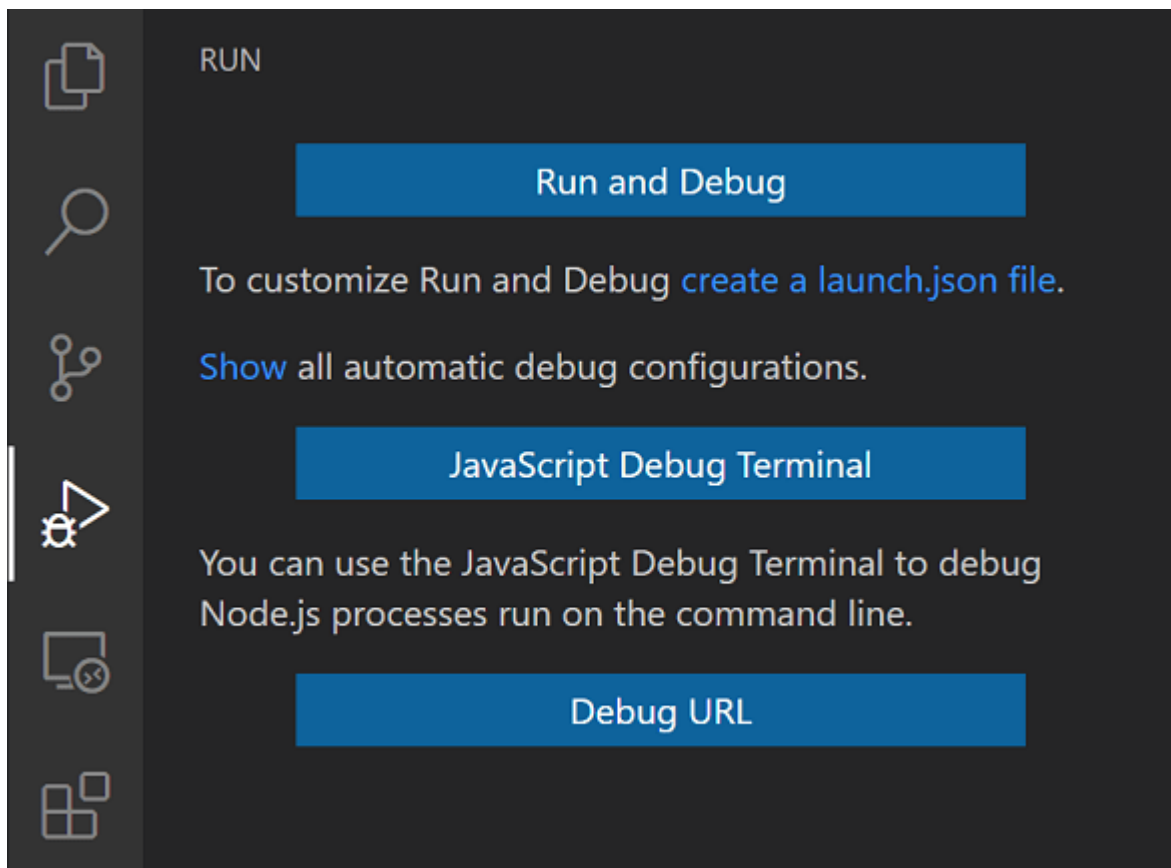
[Run and Debug view](#)

To bring up the **Run and Debug** view, select the **Run and Debug** icon in the **Activity Bar** on the side of VS Code. You can also use the keyboard shortcut Ctrl+Shift+D.



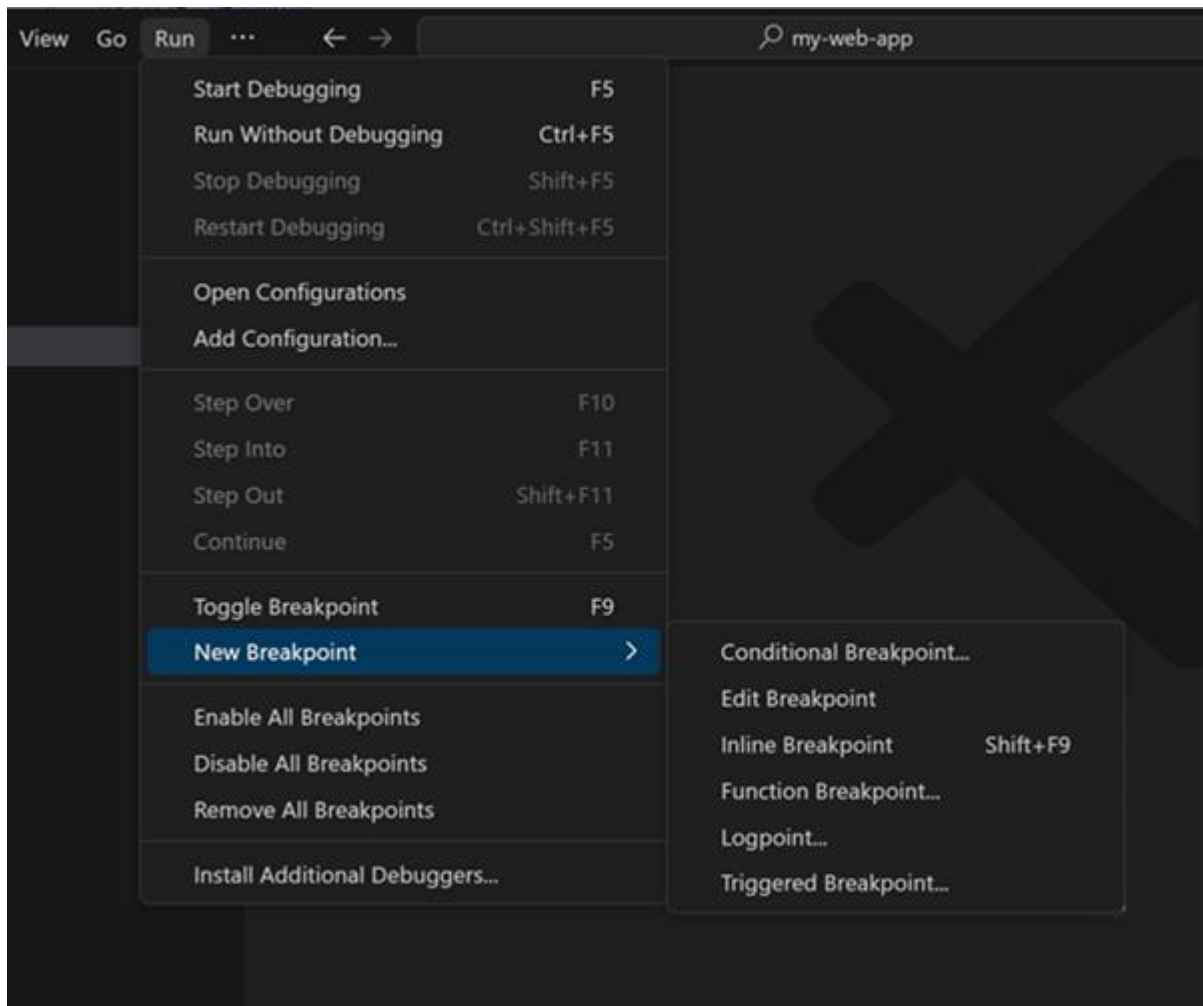
The **Run and Debug** view displays all information related to running and debugging and has a top bar with debugging commands and configuration settings.

If running and debugging is not yet configured (no `launch.json` has been created), VS Code shows the Run start view.



[Run menu](#)

The top-level **Run** menu has the most common run and debug commands:

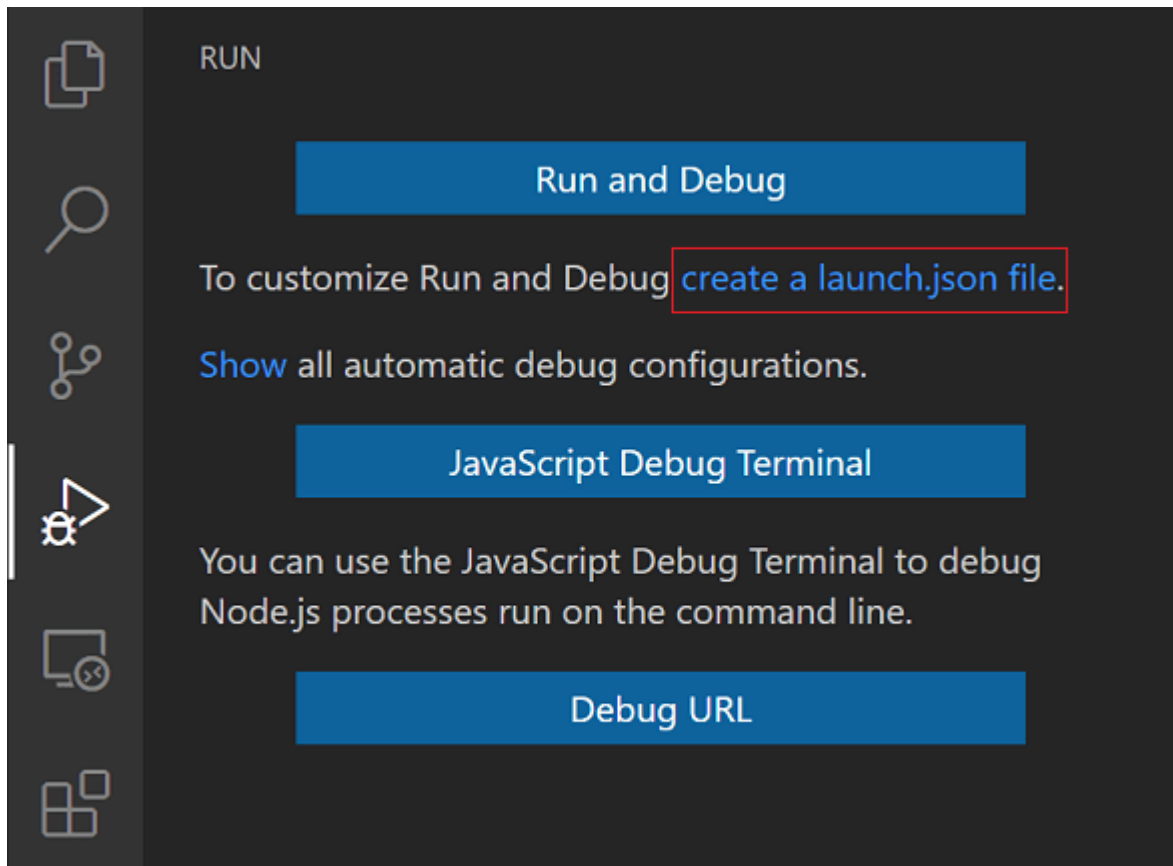


[Launch configurations](#)

To run or debug a simple app in VS Code, select **Run and Debug** on the Debug start view or press F5 and VS Code will try to run your currently active file.

However, for most debugging scenarios, creating a launch configuration file is beneficial because it allows you to configure and save debugging setup details. VS Code keeps debugging configuration information in a `launch.json` file located in a `.vscode` folder in your workspace (project root folder) or in your [user settings](#) or [workspace settings](#).

To create a `launch.json` file, select **create a launch.json file** in the Run start view.



COMMON DEBUGGING FEATURES

Step in / Step over

This is the most basic process of debugging, it's not really a unique feature nor worth mentioning. BUT! I've seen a lot of people using them **without** the hotkeys, which really impacts your debugging speed. So, short tip: **use the hotkeys (F10/F11)!**

Manually changing values in runtime

By hovering the mouse over properties and variables, we can check their current value. But it becomes even better: we can change the value to whatever we want, as long as it is of the same type, otherwise Visual Studio will kindly complain.

Immediate window

Another great tool, we can also use it to change variables and properties values. But it's more powerful since we can use it to call other methods, which may or may not change a value. You can even create and populate new variables, so it is very useful in order to investigate different scenarios while debugging.

Watcher

This one is really useful to keep track of variables, as well as dissect them. You can add any variable to it, and it will automatically load all its values. And as you progress through your debugging journey, it will mark in red the ones that have changed. Oh, you can also add expressions to it.

Conditional breakpoints

Breakpoints are incredibly useful, but sometimes, when we are iterating through a giant *for* loop, it can be pretty boring to skip until we find what we're looking for. So, we have the conditional breakpoints to save us.

Yellow arrow dragging

Yellow arrow indicates which is the next piece of code that is going to be executed, right? It was pretty shocking to me to learn that we can drag it around. We can use it to skip a line, run a line again, and so on.

There are some limitations to it, of course. You can't drag it to another method, you can't drag it inside or outside a *finally* statement, and so on. You will figure them out as you go.

Call stack

It enables us to know how did we end up where we are. And by double clicking on the lines, we are able to get the current context from that method's perspective. For example, we can check the current values of all variables related to that method.

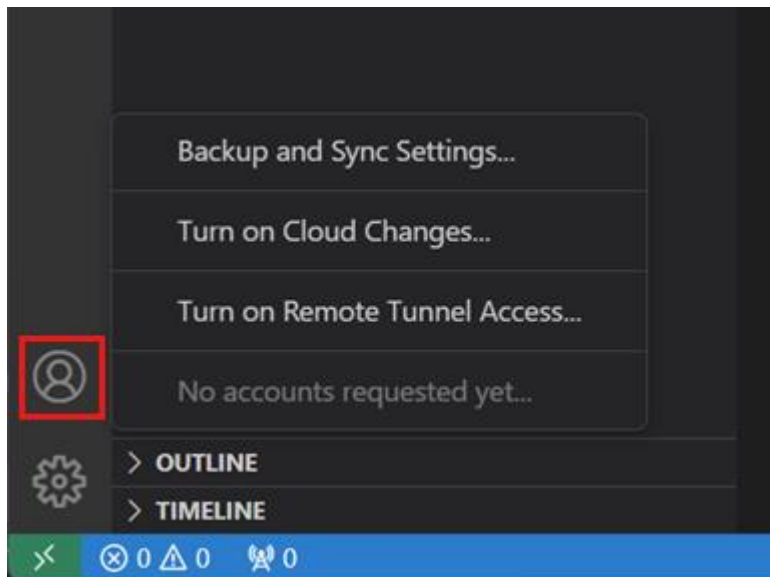
10. Using Source Control:

How can users integrate Git with VS Code for version control? Describe the process of initializing a repository, making commits, and pushing changes to GitHub.

[Set up Git in VS Code](#)

To use Git and GitHub in VS Code, first make sure you [have Git installed on your computer](#). If Git is missing, the **Source Control** view shows instructions on how to install it. Make sure to restart VS Code afterwards.

Additionally you can sign into VS Code with your GitHub account in the **Accounts** menu in the lower right of the Activity bar to enable additional features like [Settings Sync](#), but also cloning and publishing repositories from GitHub.



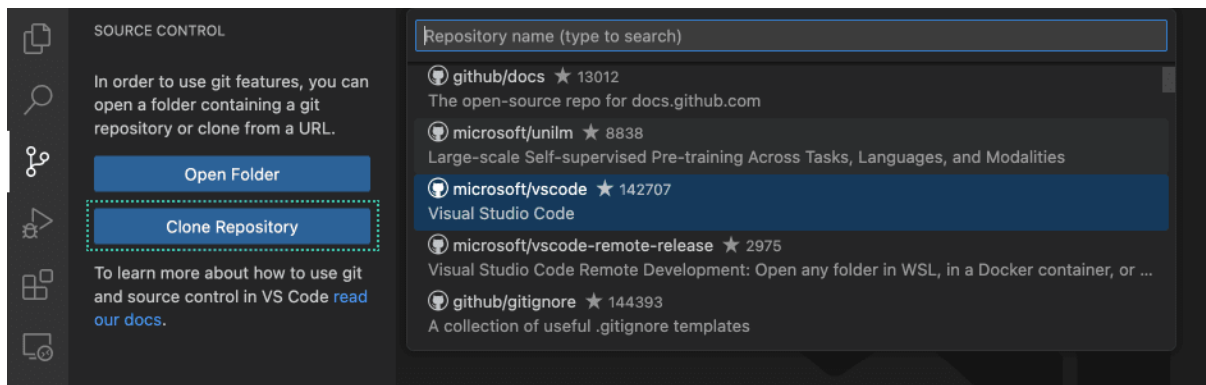
[Open a Git repository](#)

VS Code provides several ways to get started in a Git repository, from local to remote cloud-powered environments like [GitHub Codespaces](#).

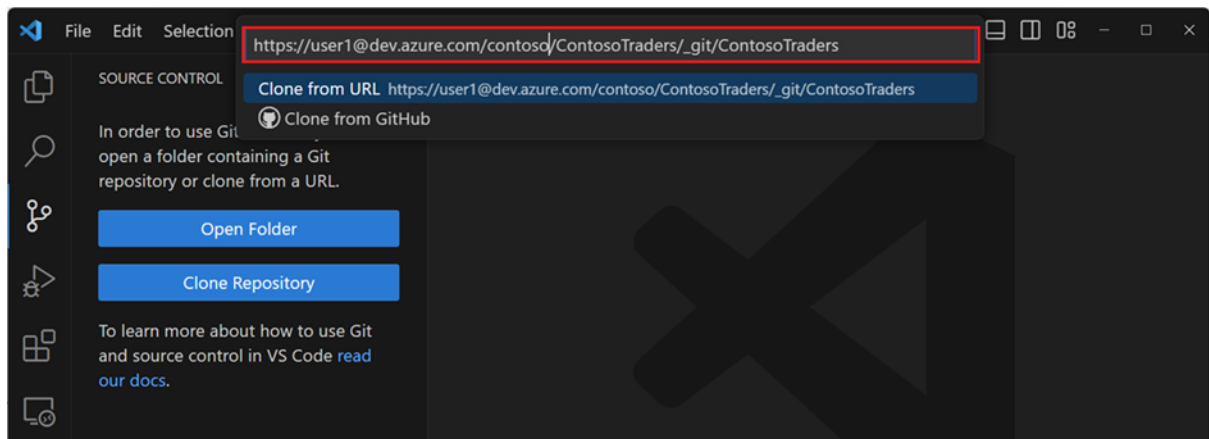
[Clone a repository locally](#)

To clone a repository, run the **Git: Clone** command in the Command Palette (Ctrl+Shift+P), or select the **Clone Repository** button in the **Source Control** view.

If you clone from GitHub, VS Code prompts you to authenticate with GitHub. Then, select a repository from the list to clone to your machine. The list contains both public and private repositories.



For other Git providers, enter the repository URL, select **Clone**, and pick a folder on your local machine to clone the files into. VS Code opens the folder once the repository is cloned on your local machine.



[Initialize a repository in a local folder](#)

To initialize a new local Git repository:

1. Pick an existing or new folder on your computer and open it in VS Code.
2. In the **Source Control** view, select the **Initialize Repository** button.

This creates a new Git repository in the current folder, allowing you to start tracking code changes.

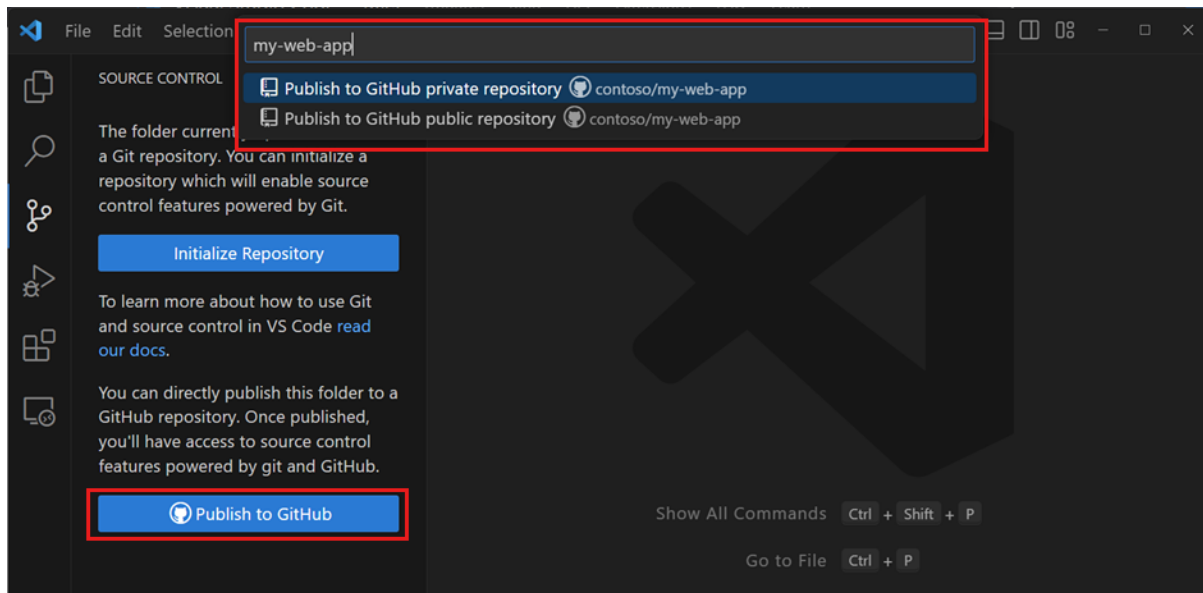
This action is equivalent to running `git init` on the command-line.



[Publish local repository to GitHub](#)

You can also initialize a local repository and publish it directly to GitHub. This creates a new repository on your GitHub account, and pushes your local code changes to the remote repository. Having your source code on a remote repository is a great way to back up your code, collaborate with others, and automate your workflow with [GitHub Actions](#).

Use the **Publish to GitHub** command button in the **Source Control** view. You can then choose a name and description for the repository, and whether to make it public or private.



Once the repository has been created, VS Code pushes your local code to the remote repository. Your code is now backed up on GitHub, and you can start collaborating with others with commits and pull requests.

[Open a GitHub repository in a codespace](#)

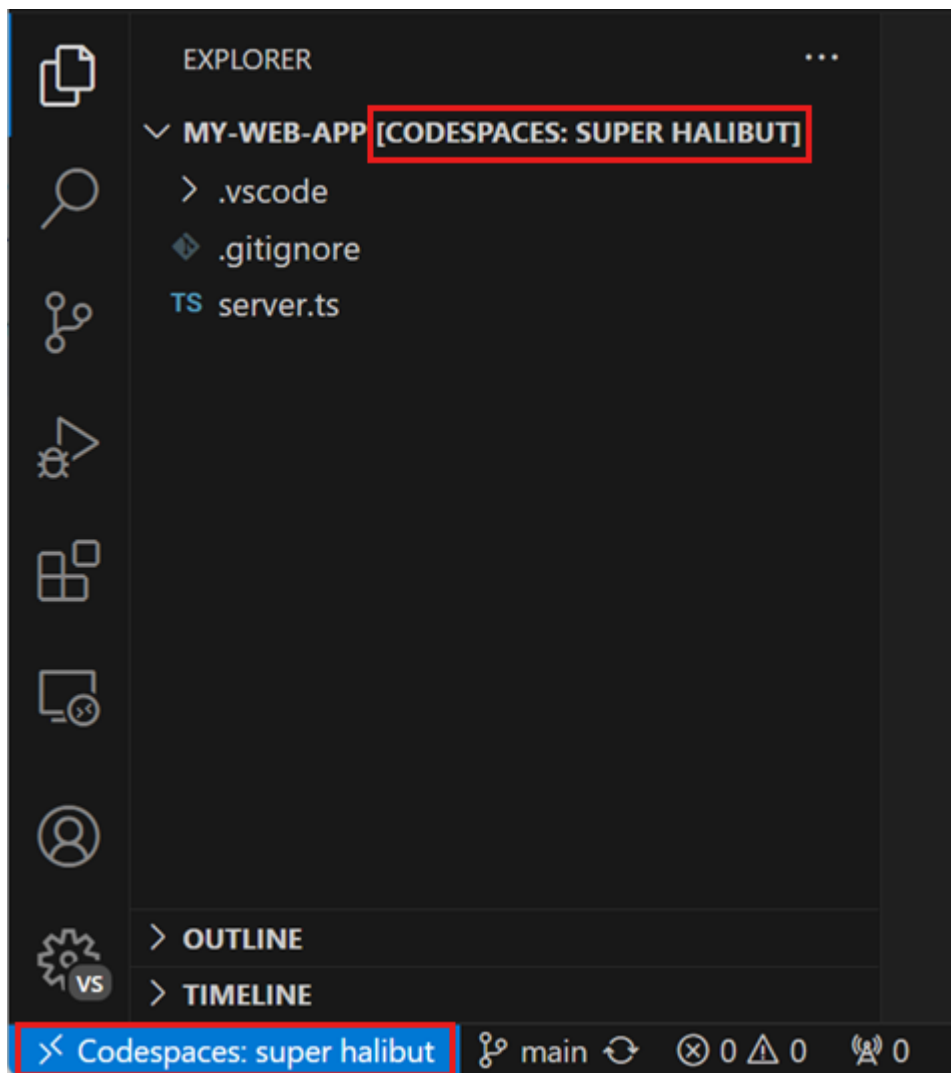
[GitHub Codespaces](#) lets you open a GitHub repository in a fully configured cloud-based development environment, enabling you to develop in a browser without having to install any software on your local computer. GitHub Codespaces allows free usage for individuals, which makes it easy to get started working on open source projects.

To create a codespace for your GitHub repository:

1. Install the [GitHub Codespaces](#) extension in VS Code and sign in with your GitHub account.
2. Run the **Codespaces: Create New Codespace** command.
3. Select the repository and branch you want to open.

VS Code opens a new window, which is connected to the codespace. The source code, terminal, and running and debugging are hosted in the remote cloud-based development environment.

Notice that the File Explorer and Status Bar indicate that the workspace is opened in a codespace.



Alternatively, you can also start from a codespace template on the [GitHub Codespaces website](https://github.com/codespaces).

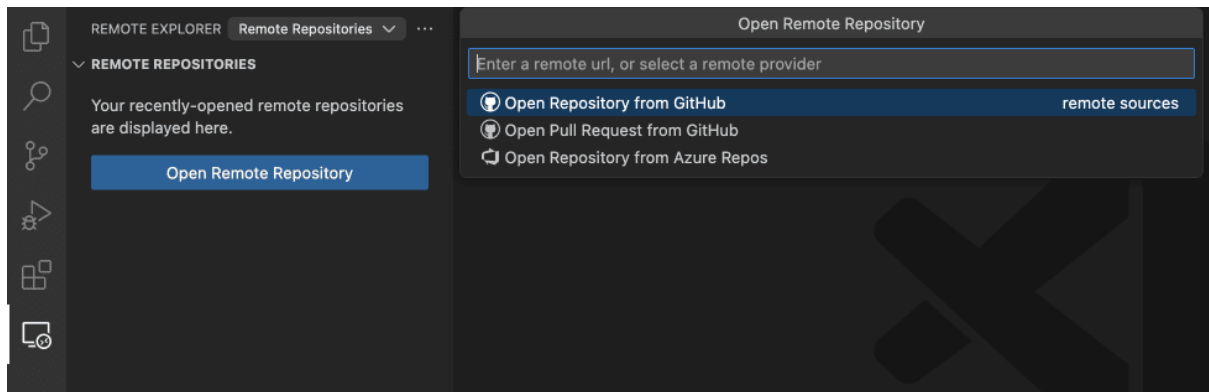
If you already have a codespace open in your browser, run the **Codespaces: Open in VS Code Desktop** command in the browser to connect to the codespace from your local VS Code Desktop.

You can learn more about GitHub Codespaces, including customization such as forwarding ports, in the [Developing in a codespace](https://docs.github.com/en/codespaces) documentation.

[Open a GitHub repository remotely](#)

VS Code's remote repository support allows you to browse and edit a GitHub repository without cloning it to your local computer. This is useful for quickly making changes to a remote repository without having to clone the entire codebase to your machine.

1. First install the [GitHub Repositories](#) extension.
2. Run the command **Remote Repositories: Open Remote Repository...** or use the **Open Remote Repository** button the Explorer view.
3. Search and select the GitHub repository that you want to open.



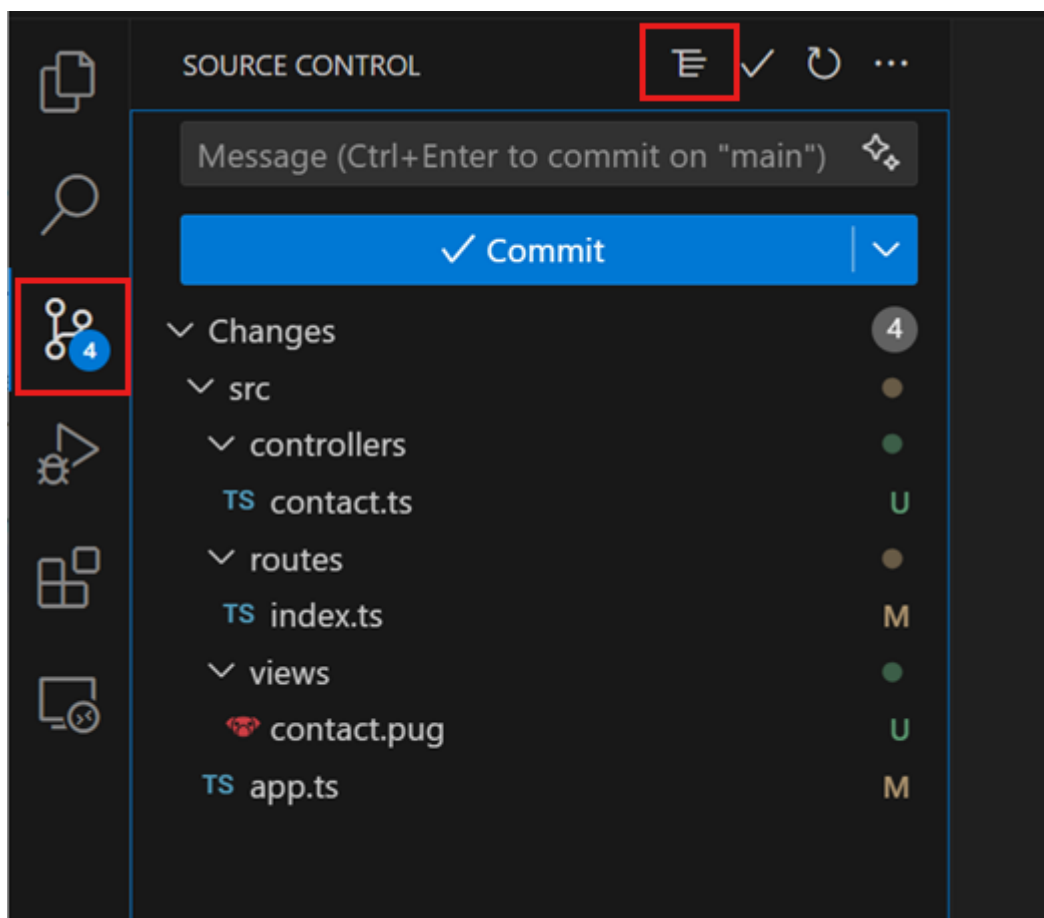
Tip: If you need to execute code or run terminal commands, you can seamlessly switch from a remote repository to a codespace with the command **Continue Working on**.

[Staging and committing code changes](#)

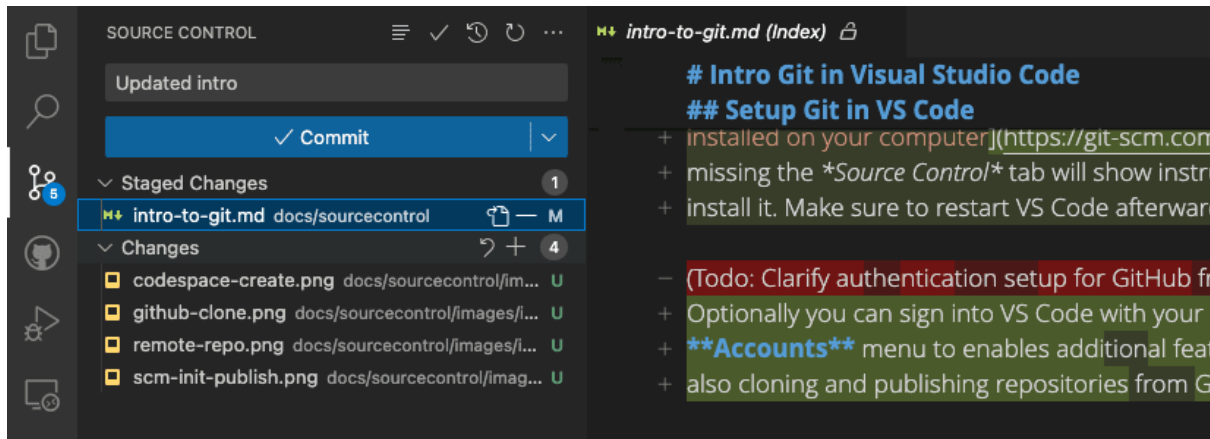
Once you have a Git repository set up, you can start tracking code changes by [staging and committing](#) your newly created and edited code.

Tip: Commit your changes early and often. This makes it easier to revert back to previous versions of your code if needed.

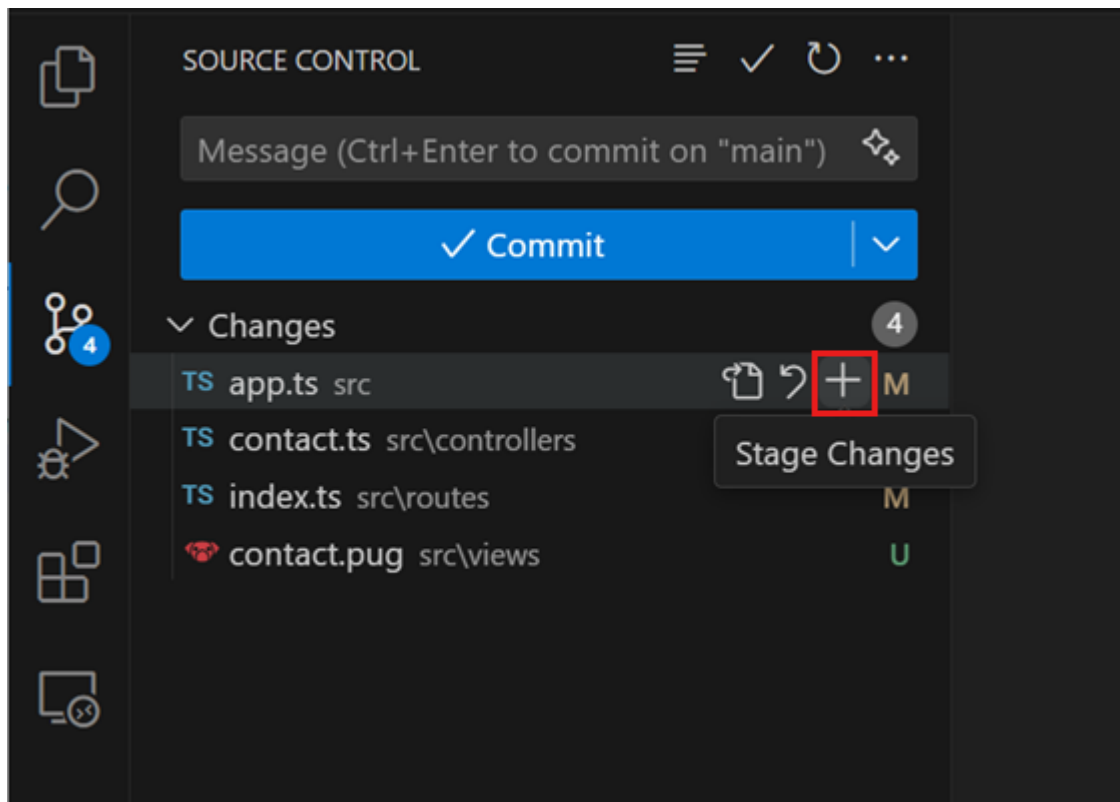
You can access the **Source Control** view from the Activity Bar to list all changed files in your workspace. You can toggle between a tree view or list view by using the tree/list icon in the Source Control view header.



When you select a file in the Source Control view, the editor shows a diff view that highlights the file changes, compared to the previously committed file.

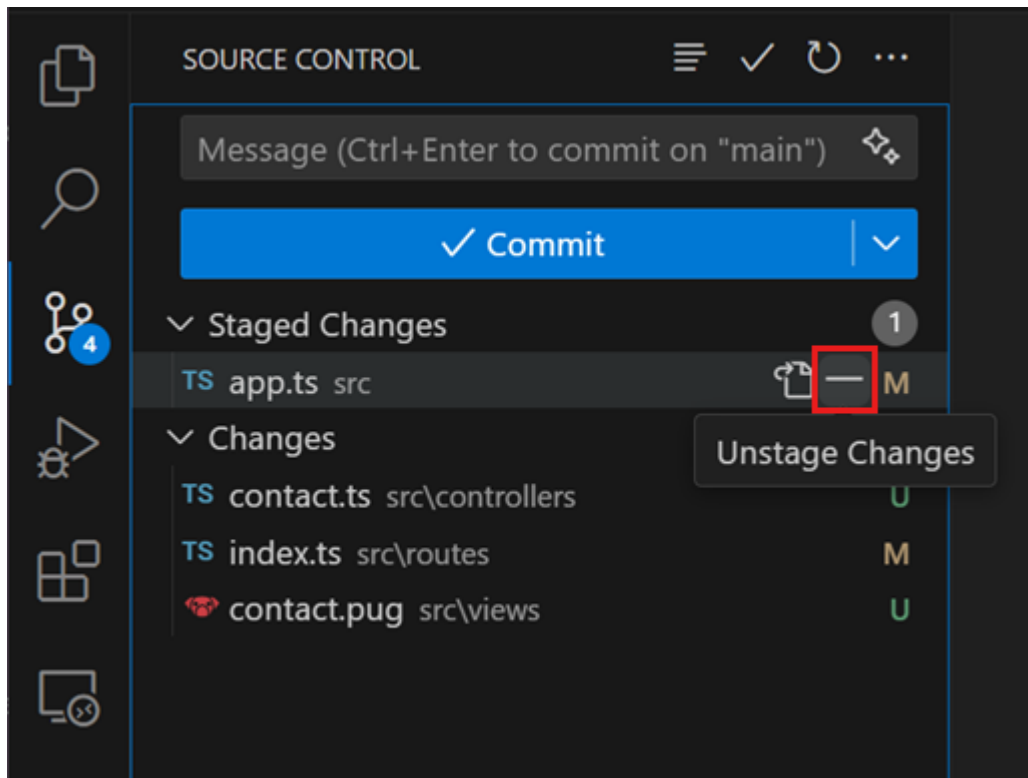


To stage a file, select the + (plus) icon next to the file in the **Source Control** view. This adds the file to the **Staged Changes** section, indicating that it will be included in the next commit.



You can also stage all pending changes at once by selecting the + (plus) icon next to **Changes** in the Source Control view.

Staged changes can also be discarded by selecting the - (minus) icon next to the file. Similarly, you can discard all staged changes by selecting the - (minus) icon next to **Staged Changes** in the Source Control view.



To commit your staged changes, type a commit message in the upper text box, and then select the **Commit** button. This saves your changes to the local Git repository, allowing you to revert to previous versions of your code if needed.

[Pushing and pulling remote changes](#)

Once you have made commits to your local Git repository, you can push them to the remote repository. The **Sync Changes** button indicates how many commits are going to be pushed and pulled. Selecting the **Sync Changes** button downloads (pull) any new remote commits and uploads (push) new local commits to the remote repository.