1. **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.**

* Download Visual Studio Code

Visit the Official Website:

Open your web browser and go to the official Visual Studio Code website: code.visualstudio.com.

Download the Installer:

Click on the "Download" button on the homepage.

The website should automatically detect your operating system and provide the appropriate download link.

Click on the "Windows" download button to download the installer (VSCodeUserSetup-x64-x.x.x.exe for 64-bit or VSCodeUserSetup-x32-x.x.x.exe for 32-bit).

* Install Visual Studio Code

Run the Installer:

Locate the downloaded installer file (usually in your Downloads folder) and double-click it to run.

Setup Wizard:

The Visual Studio Code Setup Wizard will open. Click "Next" to continue.

Accept the License Agreement:

Read and accept the license agreement by checking the box and clicking "Next."

Select Installation Location:

Choose the destination folder where you want to install VS Code. The default location is typically fine. Click "Next."

Select Additional Tasks:

Choose any additional tasks you want to perform, such as creating a desktop icon, adding VS Code to the PATH (recommended), and adding context menu entries for easier access. Click "Next."

Install:

Click "Install" to begin the installation process. The setup will install Visual Studio Code on your machine.

Finish Installation:

Once the installation is complete, you can choose to launch Visual Studio Code immediately by checking the "Launch Visual Studio Code" box. Click "Finish."

* First Launch and Initial Setup

Launch VS Code:

If you didn't select the option to launch VS Code immediately, you can open it from the Start menu or by double-clicking the desktop icon.

Install Extensions:

VS Code has a rich ecosystem of extensions. To enhance your development experience, you can install extensions by clicking on the Extensions view icon on the Sidebar or by pressing Ctrl+Shift+X.

Popular extensions include Python, C/C++, JavaScript, and Git.

* Configure Visual Studio Code

Configure settings to suit your preferences by navigating to File > Preferences > Settings or using Ctrl+,.

Customize themes, font size, keybindings, and more.

Open a Folder or Workspace:

Open a project folder by going to File > Open Folder or by dragging and dropping a folder into the VS Code window.

For larger projects, consider setting up a workspace.

1. **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.**

### **General Settings**

**Theme and Icon Theme:**

* + Customize the appearance by selecting a theme. Go to ***File > Preferences > Color Theme***
  + Choose an icon theme via ***File > Preferences > File Icon Theme***.

**Font and Size:**

* + Adjust the font size and typeface in ***File > Preferences > Settings***
  + Search for ***Font Size*** and set your preferred size.
  + Set the font family by searching for ***Editor: Font Family.***

**Tab and Indentation:**

* + Configure tab size and indentation in settings. Search for ***Tab Size*** and ***Indent Using Tabs/Spaces.***
  + Enable ***Editor: Detect Indentation*** to automatically detect indentation style.

**Line Numbers and Wrapping:**

* + Enable line numbers and word wrap by searching for ***Editor: Line Numbers*** and ***Editor: Word Wrap*** in settings.

**Auto Save:**

* + Enable auto save by searching for ***Files: Auto Save*** and setting it to ***afterDelay***

### **Key Extensions**

**Python:**

* + Install the Python extension by Microsoft for Python development, which provides IntelliSense, linting, and debugging capabilities.
  + Also, consider installing Pylance for better performance and type checking.

**Prettier - Code Formatter:**

* + Install Prettier for automatic code formatting. Configure it to format on save in settings: search for Editor: Format On Save and enable it.

**GitLens:**

* + Install GitLens to enhance Git capabilities, providing insights into code authorship, history, and more.

**Bracket Pair Colorizer:**

* + Install Bracket Pair Colorizer to help visually distinguish matching brackets with color coding.

**Debugger for Chrome:**

* + For web development, install Debugger for Chrome to enable debugging directly from VS Code.

### **Language-Specific Extensions**

**JavaScript and TypeScript:**

* + Install the JavaScript (ES6) code snippets extension for useful snippets.
  + TypeScript support is built-in, but installing the TypeScript Hero extension can enhance productivity.

**HTML, CSS:**

* + Install the HTML CSS Support extension for enhanced HTML and CSS features.

**C/C++:**

* + Install the C/C++ extension by Microsoft for IntelliSense, debugging, and code browsing.

**Java:**

* + Install the Java Extension Pack which includes tools for Java development like IntelliSense, project management, and debugging.

### **Additional Configurations**

**Integrated Terminal:**

* + Customize the integrated terminal by setting the default shell in settings: search for ***Terminal: Integrated: Shell*** and specify your preferred shell (e.g., PowerShell, Bash).

**Code Snippets:**

* + Create or import custom code snippets for repetitive code structures. Go to ***File > Preferences > User Snippets.***

**Workspace Settings:**

* + Customize settings per project by modifying workspace settings. Go to ***File > Preferences > Settings***, then switch to the workspace tab.

**Version Control:**

* + Configure Git by setting your user name and email in the integrated terminal

1. **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.**

**Activity Bar**

The Activity Bar is located on the far left side of the VS Code window. It serves as the main navigation tool and provides quick access to different views and features. The default icons include:

* Explorer: Manages files and folders within your workspace.
* Search: Performs text searches across your files.
* Source Control: Manages version control operations like commits and branches, primarily for Git.
* Run and Debug: Starts debugging sessions and manages debug configurations.
* Extensions: Installs and manages VS Code extensions.

You can customize the Activity Bar by right-clicking to hide or show icons based on your workflow preferences.

**Side Bar**

The Side Bar is adjacent to the Activity Bar and changes content based on the Activity Bar selection. Its primary purpose is to provide detailed views and functionalities. For instance:

* Explorer View: Displays the file and folder structure of your workspace.
* Source Control View: Shows changes, commit history, and branch management.
* Search View: Displays search results with options to replace text across files.
* Extensions View: Lists installed extensions and provides options to search and install new ones.

The Side Bar enhances navigation and control, making it easier to manage various aspects of your project.

**Editor Group**

The Editor Group is the central part of the interface where you write and edit code. It can host multiple editors in a tabbed layout. Key features include:

* Tabs: Each open file is represented by a tab. You can switch between tabs easily.
* Split Editors: Allows you to divide the Editor Group into multiple panes, enabling side-by-side file editing.
* IntelliSense: Provides code completion, parameter info, quick info, and member lists.
* Code Navigation: Features like go-to-definition, find references, and symbol search improve code navigation and understanding.

The Editor Group is designed for efficient code writing and navigation, supporting various programming languages and configurations.

**Status Bar**

The Status Bar is located at the bottom of the VS Code window. It provides essential information about your current workspace and open files. Key elements include:

* Language Mode: Displays the language of the current file and allows you to change it.
* Git Branch: Shows the current Git branch and provides quick access to branch and repository commands.
* Encoding and Line Endings: Displays the current file's encoding and line ending type.
* Errors and Warnings: Shows the number of errors and warnings in the current file and workspace.
* Live Server Status: For those using Live Server, it shows the server status and provides quick access to start or stop the server.

1. **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.**

The Command Palette is a powerful feature that provides quick access to a wide range of commands and functions without needing to navigate through menus or use the mouse. It serves as a central command hub, allowing you to efficiently perform various tasks through keyboard commands.

* **Accessing the Command Palette**

Keyboard Shortcut: Press ***Ctrl+Shift+P***

Menu Bar: Navigate to ***View > Command Palette..***..

Once the Command Palette is open, you can start typing the name of the command or function you want to execute. VS Code will provide autocomplete suggestions based on your input.

* **Common Tasks Using the Command Palette**

Here are some examples of common tasks that can be performed using the Command Palette:

*Opening Files and Folders:*

***File: Open File..***.: Quickly open a file.

***File: Open Folder***...: Open a folder as a workspace.

***File: Save All***: Save all open files.

*Navigating and Editing Code:*

***Go to File...***: Open a specific file by typing its name.

***Go to Symbol in File..***.: Jump to a specific symbol (function, variable, etc.) within the current file.

***Go to Definition***: Navigate to the definition of a symbol.

*Source Control Management:*

***Git: Clone.***..: Clone a repository from a URL.

***Git: Commit..***.: Commit changes to the local repository.

***Git: Push***: Push commits to the remote repository.

*Debugging:*

***Debug: Start Debugging:*** Start a debugging session.

***Debug: Add Configuration***: Add a new debug configuration.

***Debug: Step Over/Into/Out***: Control the flow during debugging.

*Extensions and Settings:*

***Extensions: Install Extensions***: Open the Extensions view to browse and install extensions.

***Preferences: Open Settings (UI)***: Open the settings UI to customize VS Code.

***Preferences: Open Keyboard Shortcuts***: Modify keyboard shortcuts.

*Terminal Operations:*

***Terminal: Create New Integrated Terminal:*** Open a new integrated terminal.

***Terminal: Run Task***: Run a predefined task.

Customization and Appearance:

***View: Toggle Terminal:*** Show or hide the integrated terminal.

***View: Toggle Side Bar Visibility***: Show or hide the Side Bar.

***View: Toggle Zen Mode***: Enter or exit Zen Mode for a distraction-free coding environment.

1. **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.**

**Finding and Installing Extensions**

* Using the Extensions View:

Open the Extensions view by clicking the Extensions icon in the Activity Bar on the side of the window or by pressing Ctrl+Shift+X.

In the Extensions view, you can browse, search, and install extensions from the Visual Studio Code Marketplace.

To find a specific extension, type its name or keywords related to its functionality in the search bar.

* Installing Extensions:

Once you find an extension you want to install, click the Install button next to it.

After installation, some extensions may require a reload of VS Code. If prompted, click the Reload button.

**Managing Extensions**

* Enabling/Disabling Extensions:

In the Extensions view, installed extensions have additional options. You can enable or disable an extension by clicking the gear icon (⚙) next to it and selecting the appropriate option.

Uninstalling Extensions:

To uninstall an extension, click the gear icon (⚙) next to the extension in the Extensions view and select Uninstall.

* Extension Settings:

Many extensions come with customizable settings. To access these, click the gear icon (⚙) next to the extension and select Extension Settings.

* Updates:

Extensions are updated regularly. VS Code will notify you of available updates, which you can install via the Extensions view.

**Essential Extensions for Web Development**

* Prettier - Code Formatter:

Automatically formats your code according to a consistent style. Supports JavaScript, CSS, HTML, and more.

Install: Search for Prettier - Code formatter and click Install.

* ESLint:

Integrates ESLint into VS Code to provide real-time linting for JavaScript and TypeScript, helping you maintain code quality.

Install: Search for ESLint and click Install.

* Live Server:

Launches a local development server with live reload capability for static and dynamic pages.

Install: Search for Live Server and click Install.

* Debugger for Chrome:

Allows you to debug your JavaScript code running in Google Chrome directly from VS Code.

Install: Search for Debugger for Chrome and click Install.

* HTML CSS Support:

Enhances VS Code’s built-in HTML and CSS capabilities by adding IntelliSense and validation for HTML and CSS.

Install: Search for HTML CSS Support and click Install.

* JavaScript (ES6) code snippets:

Provides a collection of useful JavaScript code snippets for faster coding.

Install: Search for JavaScript (ES6) code snippets and click Install.

* Path Intellisense:

Auto-completes filenames when typing paths in your code.

Install: Search for Path Intellisense and click Install.

* GitLens:

Enhances Git capabilities by providing insights into code authorship, commit history, and more.

Install: Search for GitLens and click Install.

* REST Client:

Allows you to send HTTP requests and view responses directly within VS Code, useful for API testing.

Install: Search for REST Client and click Install.

1. **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?**

**Opening the Integrated Terminal:**

Keyboard Shortcut: Press ***Ctrl+***

Menu Bar:Navigate to ***View > Terminal*** to open the terminal.

Activity Bar:Click the terminal icon in the Activity Bar on the side of the window.

**Using the Integrated Terminal**

* Creating New Terminals:

To create a new terminal instance, click the + icon in the terminal tab or use the keyboard shortcut ***Ctrl+Shift+***

* Splitting Terminals:

You can split the terminal window into multiple panes by clicking the split icon or using ***Ctrl+\***

* Switching Between Terminals:

Switch between terminal instances by clicking on the terminal tabs or using the drop-down menu in the terminal tab bar.

* Running Commands:

Type and run commands just as you would in any other terminal. This can include running build scripts, executing code, managing version control with Git, etc.

* Customizing the Terminal:

You can customize the integrated terminal's appearance and behavior through settings. For instance, you can set the default shell by going to ***File > Preferences > Settings*** and searching for ***Terminal: Integrated: Shell.***

**Advantages of Using the Integrated Terminal**

* Seamless Integration:

The integrated terminal is part of the VS Code interface, allowing you to stay within the editor without needing to switch to an external terminal application. This helps maintain focus and reduces context switching.

* Efficient Workflow:

Running commands, viewing output, and making code changes all within the same window streamlines the development process. You can quickly fix issues and see the results without leaving the editor.

* Context Awareness:

The integrated terminal operates within the context of the current workspace. It automatically starts in the root directory of your project, saving you from manually navigating to your project's directory each time you open a terminal.

* Multiple Terminals:

You can have multiple terminal instances and easily switch between them. This is particularly useful for managing different tasks concurrently, such as running a development server, executing build commands, and using version control.

* Customization:

The integrated terminal can be customized to fit your preferences, including setting the default shell (e.g., Bash, PowerShell, Command Prompt), changing font size, and configuring keybindings for terminal actions.

* Consistent Environment:

Since the terminal is integrated into VS Code, you benefit from a consistent environment across different operating systems and setups. This uniformity can help reduce configuration issues and improve portability.

* Extension Integration:

Many VS Code extensions enhance the integrated terminal’s functionality, such as providing additional commands, integrating with tools like Docker, and displaying inline output for tasks like testing and linting.

1. **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**

* Using the Explorer View:

Open the Explorer view by clicking the Explorer icon in the Activity Bar or pressing Ctrl+Shift+E.

To create a new file, right-click on the desired folder or workspace in the Explorer and select New File. You can also press Ctrl+N to create a new untitled file.

To create a new folder, right-click on the desired location in the Explorer and select New Folder.

* Using the Command Palette:

Open the Command Palette with Ctrl+Shift+P (Windows/Linux) or Cmd+Shift+P (macOS).

Type New File or New Folder and select the corresponding command.

Opening Files and Folders

* Using the File Menu:

Go to File > Open File... or File > Open Folder... to browse and open files or folders.

You can also use File > Open Recent to quickly access recently opened files and folders.

* Using the Explorer View:

Double-click on a file in the Explorer to open it.

Single-clicking a file will preview it without opening it in a tab, allowing you to quickly browse files.

* Drag and Drop:

Drag files or folders from your file explorer (e.g., Windows Explorer, Finder on macOS) into the VS Code window to open them.

**Managing Files and Folders**

* Renaming Files and Folders:

Right-click on the file or folder in the Explorer and select Rename, or use F2 to rename the selected item.

* Deleting Files and Folders:

Right-click on the file or folder and select Delete, or use the Delete key. Confirm the deletion if prompted.

* Moving Files and Folders:

Drag and drop files or folders within the Explorer to move them to a new location. Alternatively, use cut (Ctrl+X) and paste (Ctrl+V) commands.

**Navigating Between Files and Directories Efficiently**

* Quick Open:

Press Ctrl+P (Windows/Linux) or Cmd+P (macOS) to open the Quick Open dialog. Start typing the name of the file you want to open, and VS Code will provide a list of matching files. This is useful for quickly jumping to specific files.

* Go to Definition:

Use F12 to go to the definition of a symbol under the cursor, which helps navigate to the source of functions, variables, etc.

* Breadcrumbs:

Enable breadcrumbs by going to View > Show Breadcrumbs. This provides a navigation bar at the top of the editor, showing the path of the current file and allowing easy navigation within the file hierarchy.

* Side Bar Navigation:

Use the Explorer view to navigate through your file structure. You can expand and collapse folders to manage your view efficiently.

* Search:

Use the search feature (Ctrl+Shift+F or Cmd+Shift+F) to search for text within files in your workspace. This helps locate files containing specific code or text quickly.

* File Tabs:

Open multiple files in tabs and switch between them by clicking the tabs or using Ctrl+Tab to cycle through them.

* Split Editor:

Split the editor to view multiple files side by side by right-clicking a tab and selecting Split Right or Split Down. You can also drag a tab to the right or bottom edge of the editor to split it.

* Navigation History:

Use Alt+Left Arrow to go back and Alt+Right Arrow (Windows/Linux) or Ctrl+Shift+- (macOS) to go forward through your navigation history.

1. **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.**

**Accessing Settings**

* Settings UI:

Open the settings UI by navigating to File > Preferences > Settings or pressing Ctrl+, (Windows/Linux) or Cmd+, (macOS).

The settings UI provides a user-friendly interface where you can search for and modify settings.

* Settings JSON:

For more advanced customization, you can edit the settings.json file directly. Open it by clicking the {} icon in the top-right corner of the settings UI, or by using the Command Palette (Ctrl+Shift+P or Cmd+Shift+P) and typing Preferences: Open Settings (JSON).

**Changing the Theme**

* Using the Settings UI:

Go to File > Preferences > Color Theme or use the Command Palette (Ctrl+Shift+P or Cmd+Shift+P) and type Color Theme.

Browse through the available themes and click on a theme to apply it.

* Using the Command Palette:

Open the Command Palette with Ctrl+Shift+P (Windows/Linux) or Cmd+Shift+P (macOS).

Type Preferences: Color Theme and select it.

Choose a theme from the list.

**Changing the Font Size**

* Using the Settings UI:

Open the settings UI (Ctrl+, or Cmd+,).

In the search bar, type Font Size.

Locate the Editor: Font Size setting and adjust the value to your desired font size. This setting changes the font size in the editor.

* Using the Settings JSON:

Open the settings.json file as described above.

Add or modify the following entry to change the editor font size:

***"editor.fontSize":***

Replace with your preferred font size.

**Changing Keybindings**

* Using the Settings UI:

Navigate to File > Preferences > Keyboard Shortcuts or press Ctrl+K Ctrl+S.

This opens the keybindings editor where you can search for commands and modify their keybindings.

* Customizing Keybindings:

In the keybindings editor, search for the command you want to change.

Click on the pencil icon next to the command to edit its keybinding.

Press the new key combination you want to assign and press Enter to apply it.

* Using the Keybindings JSON:

For advanced customization, you can edit the keybindings.json file directly.

Open the Command Palette (Ctrl+Shift+P or Cmd+Shift+P), type Preferences: Open Keyboard Shortcuts (JSON), and select it.

Add or modify entries to customize keybindings. For example, to change the keybinding for opening a new terminal:

***{***

***"key": "ctrl+shift+t",***

***"command": "workbench.action.terminal.new"***

***}***

1. **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?**

**Setting Up and Starting Debugging**

1. Install the Necessary Extensions

Depending on the programming language you are using, you might need to install a specific extension. For example, for Python, you need the Python extension, and for JavaScript/TypeScript, the built-in Node.js debugger is sufficient.

Open the Extensions view by pressing Ctrl+Shift+X.

Search for and install the relevant extension (e.g., Python).

2. Open or Create Your Project

Open your project folder by selecting File > Open Folder... and navigating to your project directory.

3. Configure the Debugger

Open the debug view by clicking the Debug icon in the Activity Bar or by pressing Ctrl+Shift+D.

Click on the gear icon to open the launch.json file. This file contains configuration settings for debugging. If you don't have a launch.json file, VS Code will prompt you to create one.

4. Set Breakpoints

Open the file you want to debug.

Click in the gutter to the left of the line numbers to set a breakpoint. A red dot will appear to indicate the breakpoint.

5. Start Debugging

In the Debug view, select the configuration you want to use from the drop-down menu.

Click the green play button (Start Debugging) or press F5.

**Key Debugging Features in VS Code**

* Breakpoints:

Set breakpoints to pause program execution at specific lines.

Conditional breakpoints can be set to pause execution only when certain conditions are met.

* Watch Variables:

Monitor variables by adding them to the Watch panel. This allows you to see how values change over time.

* Call Stack:

View the call stack to understand the sequence of function calls that led to the current state.

* Variables:

Inspect variables in the Variables panel to see their current values.

* Debug Console:

Execute commands and evaluate expressions while debugging in the Debug Console.

* Step Commands:

Step Over (F10): Move to the next line of code, but don’t step into functions.

Step Into (F11): Step into functions to debug inside them.

Step Out (Shift+F11): Step out of the current function to the caller.

Continue (F5): Resume program execution until the next breakpoint or the end of the program.

* Exception Handling:

Configure VS Code to break on exceptions by setting up exception breakpoints. This helps identify and troubleshoot errors.

* Integrated Terminal:

Use the integrated terminal for debugging tasks that require command-line interaction without leaving the editor.

* Logpoints:

Use logpoints to log messages to the Debug Console without stopping the execution. This is useful for tracing code execution flow without interrupting it.

Set a breakpoint on the line result = add(num1, num2).

* Configure Debugging:

Ensure your launch.json is correctly set up for Python.

* Start Debugging:

Press F5 to start debugging.

* Inspect Variables:

When execution pauses at the breakpoint, inspect the values of num1, num2, and result in the Variables panel.

* Step Through Code:

Use the Step Over, Step Into, and Step Out commands to navigate through your code.

1. **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.**

**Initialize a Git Repository**

* Open Your Project in VS Code:

Open VS Code and open the folder containing your project files (***File > Open Folder..***.).

* Initialize the Repository:

Open the Source Control view by clicking the Source Control icon in the Activity Bar or pressing ***Ctrl+Shift+G.***

If your folder is not already a Git repository, you will see an option to Initialize Repository. Click this button to initialize a new Git repository in your project folder.

**Make Your First Commit**

* Stage Changes:

After initializing the repository, VS Code will detect changes in your files. You will see a list of these changes under the Source Control view.

To stage changes, click the ***+*** icon next to each file you want to include in your commit, or click the + icon at the top to stage all changes.

* Commit Changes:

Once your changes are staged, enter a commit message in the text box at the top of the Source Control view.

Click the checkmark icon or press ***Ctrl+Enter*** to commit the changes.

**Push Changes to GitHub**

* Create a GitHub Repository:

Go to GitHub and create a new repository for your project. Copy the URL of your new GitHub repository.

* Add the Remote Repository:

Open the integrated terminal in VS Code by pressing ***Ctrl+***

Add the remote repository using the following command, replacing URL with the URL of your GitHub repository:  ***git remote add origin URL***

* Push Your Changes:

To push your committed changes to GitHub, use the following command in the terminal:

***git push -u origin master***

This command pushes the changes to the master branch on the remote repository. If you are using a different branch, replace master with the appropriate branch name.

**Key Features for Git Integration in VS Code**

* Source Control View:

Access the Source Control view by clicking the Source Control icon in the Activity Bar or pressing Ctrl+Shift+G.

This view provides a visual representation of your repository's status, including changes, branches, and stashes.

* Branch Management:

You can create, switch, and delete branches directly from the Source Control view. Click the branch icon at the bottom-left corner of the window to open the branch menu.

* Diff View:

VS Code allows you to view diffs of your changes. Click on a modified file in the Source Control view to see a side-by-side comparison of changes.

* Merge and Resolve Conflicts:

When conflicts occur, VS Code provides an interface to help you resolve them. Conflicted files will be highlighted, and you can use the inline editor to choose which changes to keep.

* Built-in Terminal:

The integrated terminal can be used to run Git commands, providing a seamless experience without leaving the editor.

**SOURCES**

1. [Visual Studio Code - Version Control with Git](https://code.visualstudio.com/docs/editor/versioncontrol)
2. [GitHub Guides](https://guides.github.com/)
3. [Git Documentation](https://git-scm.com/doc)
4. FreeCodeCamp - Git and GitHub for Beginners
5. [Stack Overflow - Git tag](https://stackoverflow.com/questions/tagged/git)
6. [VS Code Extensions Marketplace](https://marketplace.visualstudio.com/)