Assignment 05

1. Setting up Visual Studio involves a few steps.

Here's a concise step-by-step guide:

Download Visual Studio:

* Visit the Visual Studio website and click on "Download Visual Studio."
* Download from <https://code.visualstudio.com/download>.
* Follow the on-screen instructions to download the installer.
* Run the Installer:
* Run the downloaded installer.
* Choose the "Visual Studio" workload during installation, which includes the necessary components for general development.

Select Workloads and Components:

* In the Visual Studio Installer, select the workloads and components you need based on your development requirements. Common workloads include ".NET Desktop Development" or "Web Development."

Install:

* Click the "Install" button to start the installation process.
* This may take some time, as it involves downloading and installing the selected components.

Launch Visual Studio:

* Once the installation is complete, launch Visual Studio.
* Sign in with your Microsoft account or create one if prompted.
* Choose Development Environment:
* On the welcome screen, select your development environment. For example, you can choose "Development Settings" based on your preferred coding style.

Start Coding:

* You're now ready to start coding! Create a new project or open an existing one to begin your development work.

2.

After installing Visual Studio Code (VS Code), there are several initial configurations and settings you should adjust to create an optimal coding environment. Here are some important settings and extensions to consider:

**General Settings**

1. **User Settings**:

* Open settings by clicking on the gear icon (⚙) in the lower left corner and selecting "Settings", or by pressing Ctrl+,.
* Customize the settings by searching for specific settings in the search bar at the top.

1. **Theme and Icon Theme**:

* Choose a theme that is comfortable for your eyes. Go to File > Preferences > Color Theme or press Ctrl+K Ctrl+T.
* You can also change the file icon theme via File > Preferences > File Icon Theme.

1. **Font and Font Size**:

* Set a comfortable font and font size by searching for Editor: Font Family and Editor: Font Size in the settings.

1. **Word Wrap**:

* Enable word wrap to avoid horizontal scrolling. Search for Editor: Word Wrap in the settings and set it to on.

1. **Auto Save**:

* Enable auto-save to save files automatically. Search for Files: Auto Save and set it to afterDelay or another option that suits your workflow.

1. **Minimap**:

* Enable or disable the minimap based on your preference by searching for Editor: Minimap.

**Key Extensions**

1. **Language Support Extensions**:

* Install language-specific extensions for the languages you are working with (e.g., Python, JavaScript, C++, etc.). Popular extensions include:
  + - Python by Microsoft
    - JavaScript (ES6) code snippets
    - C/C++ by Microsoft

1. **Code Formatting**:

* Prettier - Code formatter: An opinionated code formatter that supports many languages.
* ESLint: Integrates ESLint into VS Code for JavaScript and TypeScript projects.

1. **Version Control**:

* GitLens: Supercharges the built-in Git capabilities of VS Code.

1. **Debugger**:

* Debugger for your specific language, such as Python, Node.js, or C++.

1. **IntelliSense and Snippets**:

* Path IntelliSense: Autocompletes filenames.
* IntelliSense for your specific languages, such as Python or TypeScript.

1. **Live Server**:

* Live Server: Launch a local development server with a live reload feature for static and dynamic pages.

1. **Bracket Pair Colorizer**:

* Helps visually distinguish matching brackets.

1. **Remote Development**:

* Remote - WSL: Enables you to use the Windows Subsystem for Linux as your full-time development environment.
* Remote - SSH: Open any folder on a remote machine using SSH and take advantage of VS Code's full feature set.

**Settings Sync**

1. **Settings Sync**:

* Enable Settings Sync to sync your settings, extensions, and keyboard shortcuts across multiple machines. Go to File > Preferences > Settings Sync or press Ctrl+Shift+P and search for Settings Sync.

**Terminal Configuration**

1. **Integrated Terminal**:

* Customize the integrated terminal by going to settings and searching for Terminal Integrated. You can change the shell, font, and appearance.
* Set the default terminal to the one you use most often (e.g., PowerShell, Git Bash).

**Workspace and Project Settings**

1. **Workspace Settings**:

* Configure workspace-specific settings by going to File > Add Folder to Workspace and then adjusting settings in the .vscode/settings.json file for the specific project.

**Miscellaneous Settings**

1. **Auto Close and Auto Rename Tags**:

* For HTML and XML, enable extensions like Auto Close Tag and Auto Rename Tag.

1. **Linting and Code Analysis**:

* Install and configure linters and code analysis tools for your languages (e.g., Pylint for Python, TSLint for TypeScript).

1. **File Explorer**:

* Customize the file explorer settings to show or hide certain files and folders by searching for Files: Exclude.

1. **Breadcrumbs**:

* Enable breadcrumbs to improve navigation within your code by searching for Breadcrumbs in the settings and turning them on.

By adjusting these settings and installing these extensions, you'll create a more efficient and pleasant coding environment tailored to your specific needs and preferences.

3. Visual Studio Code (VS Code) has a user interface designed to maximize productivity and ease of use. Here are the main components of the VS Code user interface:

* **Activity Bar**: Provides quick access to different views like Explorer, Search, Source Control, Run and Debug, and Extensions.
* **Side Bar**: Displays detailed views and tools related to the selected activity, like file navigation, search results, version control details, and debugging configurations.
* **Editor Group**: The main code editing area where you can open, edit, and manage multiple files with features like tabs and split view.
* **Status Bar**: Displays information about the current workspace and editor state, including file details, Git status, errors and warnings, and running tasks.

4. The Command Palette in Visual Studio Code (VS Code) is a powerful feature that provides quick access to a wide range of commands and settings. It allows you to perform various tasks without navigating through menus or remembering complex keyboard shortcuts.

**Accessing the Command Palette**

You can open the Command Palette in VS Code in two main ways:

1. **Keyboard Shortcut**:

* Press Ctrl+Shift+P (Windows/Linux) or Cmd+Shift+P (Mac).

1. **Menu**:

* Click on the View menu and select Command Palette.

**Common Tasks Using the Command Palette**

Here are examples of common tasks you can perform using the Command Palette:

1. **Open a File**:

* Type >Open File to quickly open a file by navigating through the file explorer.

1. **Change Language Mode**:

* Type >Change Language Mode to switch the syntax highlighting and features for the current file to another programming language.

1. **Run a Task**:

* Type >Run Task to execute predefined tasks such as build scripts or custom tasks defined in tasks.json.

1. **Install Extensions**:

* Type >Extensions: Install Extensions to open the Extensions view and search for new extensions to install.

1. **Git Commands**:

* Type >Git: Clone to clone a repository.
* Type >Git: Commit to commit changes with a message.
* Type >Git: Push to push committed changes to the remote repository.

1. **Debugging Commands**:

* Type >Debug: Start Debugging to start a debugging session.
* Type >Debug: Add Configuration to add a new debug configuration to your project.

1. **Open Settings**:

* Type >Preferences: Open Settings to open the settings editor where you can configure various aspects of VS Code.

1. **View Command History**:

* Type >View: Command History to see a list of recently used commands and execute them again.

1. **Toggle Features**:

* Type >View: Toggle Terminal to show or hide the integrated terminal.
* Type >View: Toggle Sidebar to show or hide the sidebar.

1. **Search and Replace**:

* Type >Search: Find in Files to perform a search across all files in the workspace.
* Type >Replace in Files to perform a search and replace operation across multiple files.

1. **Change Theme**:

* Type >Preferences: Color Theme to switch between different color themes for the editor.

1. **Format Document**:

* Type >Format Document to format the entire document according to the configured formatter.

1. **Show All Commands**:

* Typing > will display a list of all available commands that you can search through.

**Example Usage**

1. **Opening a File**:

* Press Ctrl+Shift+P.
* Type >Open File.
* Select the file from the file explorer.

1. **Changing the Language Mode**:

* Press Ctrl+Shift+P.
* Type >Change Language Mode.
* Select the desired language from the list.

1. **Running a Task**:

* Press Ctrl+Shift+P.
* Type >Run Task.
* Choose the task you want to execute from the list.

1. **Installing an Extension**:

* Press Ctrl+Shift+P.
* Type >Extensions: Install Extensions.
* Search for and install the desired extension.

5

Extensions in Visual Studio Code (VS Code) play a crucial role in enhancing and customizing the development environment. They allow users to add new features, improve productivity, and support additional languages, frameworks, and tools. Extensions can provide functionalities such as syntax highlighting, code completion, debugging tools, version control integrations, linters, and formatters, making VS Code a highly versatile and powerful code editor.

**Finding, Installing, and Managing Extensions**

**Finding Extensions**

1. **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 (Windows/Linux) or Cmd+Shift+X (Mac).
* Use the search bar to find specific extensions or browse through categories like "Trending," "Popular," and "Recommended."

1. **VS Code Marketplace**:

* Visit the [Visual Studio Code Marketplace](https://marketplace.visualstudio.com/vscode) to browse and search for extensions online.

**Installing Extensions**

1. **From Extensions View**:

* In the Extensions view, find the extension you want to install.
* Click the Install button for the selected extension.
* Once installed, the extension may require you to reload VS Code. Click the Reload button if prompted.

1. **From VS Code Marketplace**:

* On the VS Code Marketplace website, find the extension you want to install.
* Click the Install button, which will open VS Code and initiate the installation process.

**Managing Extensions**

1. **Enable/Disable Extensions**:

* In the Extensions view, right-click on the installed extension and select Disable or Enable as needed.

1. **Uninstall Extensions**:

* In the Extensions view, find the extension you want to remove, right-click on it, and select Uninstall.

1. **Update Extensions**:

* VS Code will notify you when updates are available for installed extensions. You can update them directly from the Extensions view by clicking the Update button.

**Essential Extensions for Web Development**

Here are some must-have extensions for web development in VS Code:

1. **Live Server**:

* Launches a local development server with a live reload feature for static and dynamic pages.
* Ideal for HTML, CSS, and JavaScript development.

1. **Prettier - Code Formatter**:

* An opinionated code formatter that supports multiple languages and integrates with VS Code's formatting capabilities.
* Helps maintain consistent code style across a project.

1. **ESLint**:

* Integrates ESLint into VS Code for JavaScript and TypeScript projects.
* Provides linting and error checking to improve code quality and maintain coding standards.

1. **Debugger for Chrome**:

* Allows debugging of JavaScript code running in Google Chrome directly from VS Code.

1. **IntelliSense for CSS class names in HTML**:

* Provides CSS class name completion for the HTML class attribute based on the CSS files in your workspace.
* Enhances productivity by reducing typos and improving code completion.

1. **JavaScript (ES6) code snippets**:

* Adds a collection of JavaScript and TypeScript code snippets for commonly used patterns and features.
* Speeds up coding by providing ready-to-use code snippets.

1. **Path IntelliSense**:

* Autocompletes file paths in your project.
* Helps avoid path-related errors and speeds up file referencing.

1. **GitLens**:

* Enhances Git capabilities in VS Code by providing rich git history, blame annotations, and more.
* Useful for tracking changes and collaborating with others.

1. **Bracket Pair Colorizer 2**:

* Colors matching brackets to make nested code blocks easier to read and understand.
* Improves readability, especially in complex code.

1. **HTML Snippets**:

* Provides code snippets for HTML, including HTML5 elements and attributes.
* Enhances productivity by offering quick access to common HTML structures.

6**.** Opening and Using the Integrated Terminal in VS Code

**Opening the Integrated Terminal**

1. **Keyboard Shortcut**:

* Press Ctrl+ (backtick) on Windows/Linux or Cmd+ (backtick) on Mac.

1. **Menu Option**:

* Go to View > Terminal in the menu bar.

1. **Command Palette**:

* Press Ctrl+Shift+P (Windows/Linux) or Cmd+Shift+P (Mac) to open the Command Palette, then type Terminal: Create New Terminal and select it.

**Using the Integrated Terminal**

1. **Creating and Managing Terminals**:

* **New Terminal**: Click the plus icon (+) in the terminal tab bar to open a new terminal instance.
* **Split Terminal**: Click the split terminal icon to split the terminal pane, allowing you to work with multiple terminal instances side by side.
* **Switch Terminals**: Use the dropdown menu in the terminal tab bar to switch between different terminal instances.

1. **Customizing the Terminal**:

* **Change Shell**: You can change the default shell by opening the Command Palette (Ctrl+Shift+P or Cmd+Shift+P) and typing Terminal: Select Default Profile. Choose from available shells like PowerShell, Command Prompt, Git Bash, or any other installed shell.
* **Resize Terminal**: Drag the top border of the terminal pane to resize it.
* **Toggle Panel**: Show or hide the terminal panel by clicking the terminal icon in the Activity Bar or using the keyboard shortcut Ctrl+ (backtick) or Cmd+ (backtick).

1. **Terminal Settings**:

* Customize terminal settings by going to File > Preferences > Settings or using the keyboard shortcut Ctrl+,. Search for Terminal to find settings like font size, cursor style, shell integration, and more.

**Advantages of Using the Integrated Terminal**

1. **Convenience and Workflow Integration**:

* **Seamless Integration**: The integrated terminal is built directly into VS Code, providing a seamless workflow without the need to switch between different applications.
* **Context Awareness**: The terminal automatically opens in the context of your current project, saving you time navigating to the project directory manually.

1. **Multitasking and Efficiency**:

* **Multiple Terminals**: Easily manage multiple terminal instances within the same window, allowing you to run various tasks concurrently (e.g., running a development server, executing build commands, and monitoring logs).
* **Split View**: The ability to split the terminal pane and view multiple terminal instances side by side enhances multitasking and efficiency.

1. **Customization and Consistency**:

* **Custom Profiles**: Configure different terminal profiles for different tasks and switch between them as needed.
* **Consistent Environment**: The integrated terminal uses the same settings and environment variables as VS Code, providing a consistent and predictable development experience.

1. **Feature Rich**:

* **Command Palette Integration**: Access terminal commands and create new terminal instances directly from the Command Palette.
* **Keyboard Shortcuts**: Utilize keyboard shortcuts for quick actions like opening, closing, and switching between terminals.
* **Terminal Settings**: Customize terminal behavior and appearance to match your preferences and workflow.

1. **Visual Integration**:

* **Panels and Views**: The terminal is part of the VS Code interface, allowing you to see terminal output alongside your code and other tools like the editor, file explorer, and debugger.
* **Persistent Sessions**: Terminals persist across VS Code sessions, so you can pick up where you left off after closing and reopening VS Code.

7. Creating, Opening, and Managing Files and Folders in VS Code

**Creating Files and Folders**

1. **Creating a New File**:

* **Using File Menu**: Go to File > New File or use the keyboard shortcut Ctrl+N (Windows/Linux) or Cmd+N (Mac).
* **Using Explorer**: Right-click on a folder in the Explorer sidebar and select New File. Enter a name for the new file.

1. **Creating a New Folder**:

**Using Explorer**: Right-click on a location in the Explorer sidebar where you want to create the folder and select New Folder. Enter a name for the new folder.

**Opening Files and Folders**

1. **Opening a File**:

* **Using File Menu**: Go to File > Open File and select the file from the dialog that appears.
* **Using Explorer**: Click on a file in the Explorer sidebar to open it in the editor.
* **Using Command Palette**: Press Ctrl+P (Windows/Linux) or Cmd+P (Mac) to open the Quick Open dialog, start typing the name of the file, and select it from the list.

1. **Opening a Folder**:

* **Using File Menu**: Go to File > Open Folder and select the folder from the dialog that appears.
* **Using Explorer**: Click the Open Folder button in the Explorer sidebar when no folder is open, or use File > Open Folder.

1. **Opening Recent Files/Folders**:

* **Using File Menu**: Go to File > Open Recent to see a list of recently opened files and folders.

**Managing Files and Folders**

1. **Renaming Files and Folders**:

* **Using Explorer**: Right-click on the file or folder you want to rename and select Rename. Enter the new name.

1. **Deleting Files and Folders**:

* **Using Explorer**: Right-click on the file or folder you want to delete and select Delete. Confirm the deletion.

1. **Moving Files and Folders**:

* **Using Drag and Drop**: Drag the file or folder from one location in the Explorer sidebar to another.
* **Using Cut/Copy and Paste**: Right-click on the file or folder, select Cut or Copy, right-click on the destination folder, and select Paste.

**Efficient Navigation Between Files and Directories**

1. **Quick Open**:

* Press Ctrl+P (Windows/Linux) or Cmd+P (Mac) to open the Quick Open dialog. Start typing the name of the file, and select it from the list.
* You can also type @ to go to a symbol in the file, # to search within files, and : to go to a line number.

1. **Explorer Sidebar**:

* Use the Explorer sidebar to navigate the folder structure of your project. Click on files and folders to expand and open them.
* Use the arrow keys to navigate through the tree structure quickly.

1. **Breadcrumbs**:

* Breadcrumbs show the current location in the file and allow quick navigation to parent elements. Enable breadcrumbs from the View > Show Breadcrumbs menu or by searching for breadcrumbs in the Command Palette (Ctrl+Shift+P or Cmd+Shift+P).

1. **Go to Definition**:

* Right-click on a symbol (like a function or variable) and select Go to Definition, or press F12. This navigates directly to the symbol's definition.

1. **Peek Definition**:

* Right-click on a symbol and select Peek Definition, or press Alt+F12. This opens a small inline window showing the definition, allowing you to view it without leaving your current context.

1. **Navigate Back and Forward**:

* Use Ctrl+- (Windows/Linux) or Cmd+- (Mac) to navigate back, and Ctrl+Shift+- (Windows/Linux) or Cmd+Shift+- (Mac) to navigate forward through your edit locations.

1. **Explorer Shortcuts**:

* Ctrl+B (Windows/Linux) or Cmd+B (Mac) to toggle the Explorer sidebar visibility.
* Ctrl+Shift+E (Windows/Linux) or Cmd+Shift+E (Mac) to focus on the Explorer sidebar.

1. **Search Across Files**:

* Press Ctrl+Shift+F (Windows/Linux) or Cmd+Shift+F (Mac) to open the search panel. Enter your search query to find occurrences across all files in the workspace.

1. **Open Editors View**:

* Use the Open Editors view at the top of the Explorer sidebar to see all currently open files. Click on a file to switch to it.

1. **Keyboard Shortcuts**:

* Learn and use keyboard shortcuts for common navigation tasks to speed up your workflow. VS Code provides a list of default shortcuts which can be customized in the settings.

**8. Finding and Customizing Settings in VS Code**

VS Code allows extensive customization of its settings to tailor the development environment to your preferences. Users can find and modify these settings in several ways.

**Accessing Settings**

1. **Settings Menu**:

* Go to File > Preferences > Settings (Windows/Linux) or Code > Preferences > Settings (Mac).

1. **Keyboard Shortcut**:

* Press Ctrl+, (Windows/Linux) or Cmd+, (Mac).

1. **Command Palette**:

* Press Ctrl+Shift+P (Windows/Linux) or Cmd+Shift+P (Mac), then type Preferences: Open Settings and select it.

**Settings UI vs. Settings JSON**

* **Settings UI**: A graphical interface where you can search and modify settings.
* **Settings JSON**: The raw JSON file (settings.json) where you can directly edit settings.

**Customizing Settings**

**Changing the Theme**

1. **Using Settings UI**:

* Open the settings as described above.
* Search for "Color Theme."
* Select your preferred theme from the dropdown list.

1. **Using Command Palette**:

* Press Ctrl+Shift+P (Windows/Linux) or Cmd+Shift+P (Mac).
* Type Preferences: Color Theme and select it.
* Choose your desired theme from the list.

**9. Setting Up and Starting Debugging in VS Code**

**Step 1: Install Necessary Extensions**

Depending on the language you are working with, you might need to install specific extensions. For example:

* **Python**: Install the Python extension.
* **JavaScript/Node.js**: VS Code has built-in support, but you can install additional extensions like ESLint.
* **C++**: Install the C++ extension.

**Step 2: Open Your Project**

Open your project folder in VS Code by going to File > Open Folder and selecting the folder that contains your project files.

**Step 3: Configure Debugging**

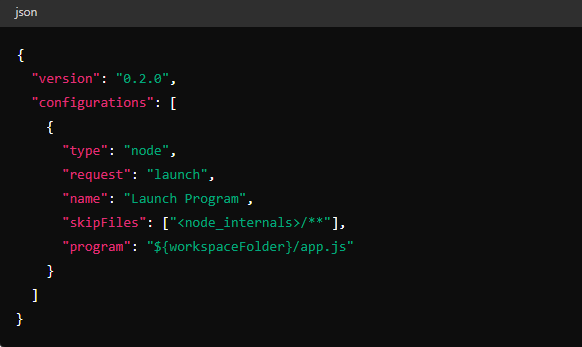
1. **Open Debug View**:

* Click the Debug icon in the Activity Bar on the side of the window or press Ctrl+Shift+D (Windows/Linux) or Cmd+Shift+D (Mac).

1. **Create a launch.json File**:

* Click the gear icon at the top of the Debug view and select Add Configuration.
* This opens a launch.json file where you can configure your debugging settings. VS Code usually suggests configurations based on the type of project you have opened.

For a simple Node.js application, the configuration might look like this:



**Step 4: Set Breakpoints**

* Open the file you want to debug.
* Click in the gutter to the left of the line number where you want to set a breakpoint. A red dot will appear to indicate the breakpoint.

**Step 5: Start Debugging**

* In the Debug view, select your configuration from the dropdown menu.
* Click the green play button or press F5 to start debugging.

**Key Debugging Features in VS Code**

1. **Breakpoints**:

* Set breakpoints by clicking in the gutter next to the line numbers.
* Conditional breakpoints can be set by right-clicking on a breakpoint and selecting "Edit Breakpoint," then entering the condition.

1. **Watch Variables**:

* In the Debug view, go to the Watch section and click the + icon to add expressions you want to watch.

1. **Call Stack**:

* View the call stack to see the sequence of function calls that led to the current point in execution.

1. **Variable Inspection**:

* Hover over variables in the editor to see their current values.
* Use the Variables section in the Debug view to inspect and modify variable values.

1. **Step Through Code**:

* Use the toolbar in the Debug view to step over, step into, step out, and continue execution:
  + - **Step Over** (F10): Execute the next line of code, but don't step into functions.
    - **Step Into** (F11): Step into the function on the current line.
    - **Step Out** (Shift+F11): Step out of the current function.
    - **Continue** (F5): Continue execution until the next breakpoint.

1. **Debug Console**:

* Use the Debug Console to evaluate expressions and interact with the debug session. This is particularly useful for testing quick code snippets and inspecting variable values.

1. **Exception Handling**:

* Configure exception breakpoints to break on exceptions. This can be done by clicking the gear icon in the Breakpoints section and selecting "Add Configuration."

1. **Integrated Terminal**:

* Use the integrated terminal (Ctrl+ or Cmd+) to run commands without leaving VS Code. This is useful for running build scripts, running tests, or any other terminal commands.

10. **Integrating Git with VS Code for Version Control**

VS Code offers built-in Git support, making it easy to manage your version control directly from the editor. Below are the steps to initialize a repository, make commits, and push changes to GitHub.

**Step 1: Install Git**

Ensure Git is installed on your machine. You can download it from the [official Git website](https://git-scm.com/).

**Step 2: Initialize a Repository**

1. **Open Your Project Folder**:

* Open VS Code and load your project folder by going to File > Open Folder.

1. **Initialize Git Repository**:

* Open the Source Control view by clicking the Source Control icon in the Activity Bar on the side of the window or press Ctrl+Shift+G (Windows/Linux)

**Step 3: Configure Git**

1. **Set Up User Details**:

* Open the terminal in VS Code by pressing Ctrl+ (backtick) or going to View > Terminal.
* Configure your Git username and email:

**Step 4: Make Commits**

1. **Stage Changes**:

* Make changes to your files.
* Open the Source Control view (Ctrl+Shift+G or Cmd+Shift+G).
* The changes will be listed under the "Changes" section. Click the + icon next to the files you want to stage or click the + icon next to "Changes" to stage all changes.

1. **Commit Changes**:

* After staging your changes, enter a commit message in the input box at the top of the Source Control view.
* Click the checkmark icon (✔) or press Ctrl+Enter (Windows/Linux) to commit the changes.

**Step 5: Push Changes to GitHub**

1. **Create a Repository on GitHub**:

* Go to [GitHub](https://github.com/) and log in.
* Click the + icon in the upper-right corner and select New repository.
* Fill in the repository details and click Create repository.

1. **Add Remote Repository**:

* Copy the repository URL from GitHub.
* In the VS Code terminal, add the remote repository:

1. **Push Changes**:

* Push your commits to the remote repository on GitHub:

**Additional Git Features in VS Code**

1. **Branch Management**:

* Create, switch, and delete branches from the Source Control view or using the Command Palette (Ctrl+Shift+P or Cmd+Shift+P).

1. **Pull and Fetch**:

* Pull the latest changes from the remote repository using the git pull command in the terminal or from the Source Control view.
* Fetch updates from the remote repository without merging them into your local branch using the git fetch command.

1. **Merge and Rebase**:

* Use the terminal to merge or rebase branches as needed

1. **Conflict Resolution**:

* When conflicts occur, VS Code highlights the conflicting sections and provides options to accept incoming or current changes.

1. **Viewing History and Diffs**:

* View the commit history and file diffs directly in VS Code. Right-click on a file in the Source Control view and select "Open Changes" to see differences between the current file and the last commit.