**Power Learn Project**

**Installing Visual Studio assignment**

**Siyabonga T. Pika**

**Installation of Visual Studio code**

To download and install Visual Studio (VS) Code on a Windows 11 operating system, you must first install Windows 11 and have administrator access to install applications. Start by opening your web browser and going to the official Visual Studio Code page. To install VS Code, click the "Download for Windows" option. After the download is complete, locate the downloaded file (called something like VSCodeSetup-1.2.3.exe, where 1.2.3 denotes the version number) in your Downloads folder and double-click it to launch the installer.

When the setup process appears, acknowledge the license agreement and then click "Next". Choose your chosen installation folder or accept the default option, then click "Next". In the additional tasks area, you may create a desktop icon, add "Open with Code" actions to the Windows Explorer file and directory context menus, register Code as an editor for supported file types, and add VS Code to the PATH for command line access. After you've chosen your choices, click "Next" and then "Install" to start the installation process.

After the installation is finished, you can run Visual Studio Code immediately by checking the "Launch Visual Studio Code" box and clicking "Finish". Upon initial launch, you may be required to install additional components or extensions; follow the prompts to complete these installs as needed.

**First-time setup**

After installation, launch VS Code and select the Extensions view by clicking the Extensions icon in the Activity Bar or hitting Ctrl+Shift+X. Find and install any extensions that are required, such as Python or JavaScript language support, or debuggers. To personalize settings, navigate to File > Preferences > Settings or press Ctrl+, and make the appropriate changes or edit the settings. If you use version control systems like Git, set them up in VS Code by navigating to the Source Control view through the Source Control icon in the Activity Bar or by typing Ctrl+Shift+G. You can configure it for an optimal coding environment by changing some settings and installing important extensions. Initially, customize the theme and font to your liking by going to File > Preferences > Color Theme for theme selection and File > Preferences > Settings for font and size selection, such as Fira Code or Source Code Pro at 14px.  
  
Enabling Settings Sync is quite useful because it synchronizes your settings, extensions, and keybindings throughout numerous devices. You can activate it by going to File > Preferences > Settings Sync > Turn On Settings Sync. To change the tab size in the editor, go to settings and search for Tab Size. To enable auto-formatting on save, search for Format On Save and check the box. To avoid horizontal scrolling, turn on Word Wrap.

Increase efficiency by preventing unneeded files and folders from being indexed. This can be done by searching for files. Exclude in settings and add patterns such as \*\*/node\_modules.

Several plugins can greatly improve your coding experience. Consider adding extensions like ms-python.python for Python, esbenp.prettier-vscode for JavaScript/TypeScript, ecmel.vscode-html-css for HTML/CSS, ms-vscode.cpptools for C/C++, redhat.java for Java, golang.go for Go, and felixfbecker.php-intellisense for PHP. To assure code formatting, esbenp.prettier-vscode and dbaeumer.vscode-eslint are excellent options.

For version control, eamodio.gitlens is an excellent tool. To increase productivity, use christian-kohler.path-intellisense for path suggestions, coenraads.bracket-pair-colorizer-2 for visualizing matching brackets, and ritwickdey.liveserver for a live server. For Docker and Kubernetes development, ms-azuretools.vscode-docker and ms-kubernetes-tools.vscode-kubernetes-tools are extremely useful. Debugging can be improved using msjsdiag.debugger-for-chrome and firefox-devtools.vscode-firefox-debug.

Also, familiarize yourself with the command palette (Ctrl+Shift+P or Cmd+Shift+P on macOS) for easy access to commands. To use the integrated terminal, press Ctrl+ (or Cmd+) and configure keybindings under File > Preferences > Keyboard Shortcuts to meet your workflow. Making these changes and installing these extensions will allow you to create a powerful and efficient coding environment in Visual Studio Code that is suited to your specific needs.

**User interface**

Visual Studio Code's user interface is designed to be both powerful and user-friendly, with numerous key components. The Activity Bar, located on the far-left side, enables easy access to various views and functions within VS Code via a collection of icons. Each icon represents a distinct feature or section, such as Explorer for file navigation and management, Search for file searching, Source Control for integrating with version control systems such as Git, Run and Debug for application testing, and Extensions for installing new functionalities.  
  
The Side Bar, which appears next to the Activity Bar, displays the contents of the currently selected activity. For example, the Explorer View displays the structure of your project's files and directories, enabling file management actions like as opening, creating, deleting, and transferring files. The Search View displays search results and allows for search and replace operations across files. The Source Control View displays changes in your repository, allowing you to stage commits and manage branches. The Run and Debug View offers debugging tools and insights, such as breakpoints, call stacks, and variables.

The Editor Group is the major area where you write and edit your code, and it supports multiple editor tabs so you may work on numerous files at once. You can divide the editor into different groups so that you can view and edit files simultaneously. The Editor Group's key features include syntax highlighting to improve readability, code folding to compact and extend parts of code, a minimap for convenient navigation, and breadcrumbs to display the current location within your project's structure.

Finally, the Status Bar is positioned at the bottom of the interface and displays information about the editor and workspace's current state, as well as shortcuts to various settings and operations. It shows the line and column numbers showing the current cursor location, file encoding and line endings, the language mode of the current file, the current Git branch and any outstanding changes, and notifications from extensions or the editor. These components work together to create a consistent and efficient coding environment, allowing you to manage projects, edit code, and use version control and debugging tools with ease.

**Command Palette**

The Command Palette in VS Code is a powerful feature that enables users to easily access and execute various commands and functions without having to navigate menus. It can be accessed by pressing Ctrl+Shift+P on Windows/Linux or Cmd+Shift+P on Mac. This opens a searchable prompt where you can enter the command you want to execute.

The Command Palette can handle a wide variety of activities, making it an indispensable tool for increasing efficiency. For example, you may open and switch between files by typing part of the file name, or you can install and manage extensions by searching for the "view and install extensions" command. It also lets you conduct version control operations like commit, pull, and push to a Git repository. You can also use it to run and debug your code by selecting the proper command to begin debugging or to do a certain operation. You may also customize the editor's settings and themes, format the code, browse to specific lines or symbols within your code, and much more. The Command Palette's versatility makes it a useful tool for optimizing workflows and efficiently utilizing all of VS Code's functions.

**Extensions in VS code**

Extensions are important in VS Code because they enhance its functionality and allow you to customize the editor to meet your individual development needs. They enable users to add features that support many programming languages, tools, and frameworks, turning VS Code into a diverse and powerful development environment. Users may easily search, install, and manage extensions using the built-in Extensions view, which can be accessed by clicking the Extensions icon in the Activity Bar or hitting Ctrl+Shift+X on Windows/Linux or Cmd+Shift+X on macOS. Users can use this view to search for extensions by name or functionality, read reviews, and see installation data. Installing an extension is as simple as clicking the "Install" button, and once installed, extensions may be controlled, activated, disabled, or deleted using the same interface.

Several important web development extensions can dramatically improve productivity and code quality. Prettier - Code Formatter (esbenp.prettier-vscode) is a popular tool for maintaining uniform code formatting. ESLint (dbaeumer.vscode-eslint) identifies and resolves JavaScript and TypeScript code errors based on predefined linting rules. Live Server (ritwickdey.liveserver) allows you to run a local development server with live reloading for static and dynamic pages. Path Intellisense (christian-kohler.path-intellisense) autocompletes file paths, making file imports more efficient. HTML CSS Support (ecmel.vscode-html-css) improves HTML and CSS editing by providing intellisense and validation. GitLens (eamodio.gitlens) provides advanced Git integration, letting developers see code authorship and repository history. Using these extensions, web developers can create a more efficient and productive working environment that is personalized to their individual workflows and project needs.

**Integrated terminal**

The integrated terminal in VS code enables developers to access a command-line interface right from within the editor, speeding the development workflow. To access the integrated terminal, use the keyboard shortcut Ctrl+ (Windows/Linux) or Cmd+ (macOS), or travel through the menu by selecting View > Terminal. When you open the terminal, it appears at the bottom of the VS Code window, where you may perform shell commands just like in an external terminal. The integrated terminal supports multiple instances, so you may create, navigate, and manage numerous terminal sessions at the same time by clicking the "+" icon or switching between them via the dropdown menu.

Using an integrated terminal has various advantages over an external terminal. It creates a streamlined workflow by maintaining everything in a one window, eliminating the need to switch between the editor and a separate terminal application. This connection improves productivity since it allows you to edit code, run build scripts, maintain version control, and perform other operations without leaving the editor. Furthermore, the integrated terminal has capabilities such as syntax highlighting and shell integration, which can increase the readability and usability of command output. By embedding the terminal into VS Code, you get consistent theming and keyboard shortcuts, resulting in a more coherent and efficient developer environment.

**File and folder management**

Creating, opening, and managing files and folders in VS code is simple and efficient, which improves the development workflow. To create a new file or folder, use the Explorer view, which is accessible from the Activity Bar. Right-click the target directory in Explorer and choose "New File" or "New Folder" from the context menu. Alternatively, to create a new file, use the keyboard shortcut Ctrl+N (or Cmd+N on macOS). Existing files can be opened by either clicking on them in Explorer or selecting File > Open File from the menu. To open an entire folder, choose File > Open Folder and navigate to the desired directory.

VS Code includes various tools that make it easier to navigate between different files and directories. The Command Palette (Ctrl+Shift+P or Cmd+Shift+P on macOS) provides easy access to files and commands by inputting their names. The Quick Open function (Ctrl+P or Cmd+P on macOS) allows you to open files quickly by inputting a section of the file name. Furthermore, the breadcrumb navigation at the top of the editing pane displays the current file path and allows you to swiftly go to parent directories or sibling files. The integrated search function (Ctrl+Shift+F, or Cmd+Shift+F on macOS) allows you to find and open files based on their content, which is especially handy for large projects. Using these functionalities, users may efficiently navigate and manage their project files and directories, resulting in a seamless and productive workflow.

**Settings and preferences**

Users can find and customize settings in VS Code through the Settings menu, which offers a wide range of configuration options to tailor the development environment to their preferences. To access the settings, navigate to File > Preferences > Settings or use the shortcut Ctrl+, (or Cmd+, on macOS). This opens the Settings editor, where you can browse or search for specific settings.

To change the theme, users can go to File > Preferences > Color Theme or use the Command Palette (Ctrl+Shift+P or Cmd+Shift+P on macOS) and type "Color Theme" to select from a list of available themes. This allows you to quickly switch between light and dark modes or choose a theme that suits your visual preferences.

Adjusting the font size is equally straightforward. In the Settings editor, search for "Font Size" and modify the value to your desired size. This change will immediately reflect in the editor, enhancing readability and comfort during coding sessions.

Customizing keybindings can be done by navigating to File > Preferences > Keyboard Shortcuts or using the Command Palette and typing "Keyboard Shortcuts." This opens the Keyboard Shortcuts editor, where you can search for specific commands and reassign their keybindings. For example, to change the keybinding for saving a file, you can search for "Save" and click on the current keybinding to set a new one.

**Debugging in VS code**

To set up and debug a basic program in VS Code, first open your project folder. Check that you have the appropriate language extension installed for your project, such as the Python extension for Python programs or the JavaScript/TypeScript extension for JavaScript applications. Next, open the file you wish to debug and create breakpoints by clicking in the gutter to the left of the line numbers where you want the execution to stop. Then, access the Run and Debug view by clicking the play icon in the Activity Bar or pressing Ctrl+Shift+D (or Cmd+Shift+D on macOS).

In case this is your first time configuring debugging for the project, select "create a launch.json file" from the Run and Debug view interface. The debugger's configuration is specified in this file. Depending on the sort of project you're working on, VS Code typically offers auto-detected debug setups. Select the suitable setup or alter it as necessary.

Click the green play button after choosing the preferred setup from the dropdown menu to begin debugging. At the breakpoints you've placed, the debugger will launch and the program will pause. The debugging session can then be managed using the debugging toolbar that shows at the top of the window. Important operations include continuing execution, ending or restarting the debug session, and stepping over, into, and out of functions.

VS Code provides a number of useful debugging tools to facilitate the process. You can better grasp the execution flow by examining the Call Stack panel, which displays the current call stack. You can view and change the variables as needed by using the Variables panel, which shows the variables and their values as of right now. You can track particular expressions or variables over time with the Watch panel. To further give additional control over the debugging process, Conditional Breakpoints can be configured to interrupt execution only when specific circumstances are fulfilled. These characteristics, along with VS Code's user-friendly interface, make debugging easier and more informative.

**Using source control**

Git's version control integration with VS Code is smooth and improves code management and teamwork. Make sure Git is installed and available via the command line before proceeding. Open the project folder in VS Code to get started. The Git icon in the Activity Bar or the shortcut Ctrl+Shift+G (or Cmd+Shift+G on macOS) can be used to access the Source Control view. Click the "Initialize Repository" button or type git init in the terminal to create a new Git repository for your project. Git can now track changes in your project folder because this step generates a.git directory there.

VS Code instantly detects changes to your files once it has been setup. To get ready for committing, click the "+" icon next to each file in the Source Control window to stage these changes. In the text box at the top of the view, type a commit message that explains the changes you're committing. In order to commit your changes locally, click the checkmark icon.  
  
Make sure you have a GitHub repository setup and initialized with a remote URL before pushing your modifications to a remote repository on GitHub. Use the command git remote add origin in the terminal to add the remote URL in VS Code; replace with the URL of your GitHub repository. Use the command git remote add origin in the terminal to add the remote URL in VS Code; replace with the URL of your GitHub repository. Next, in the Source Control view, click the ellipsis (...) next to the commit message and choose "Push" from the dropdown menu to submit your committed changes to GitHub. As an alternative, you can push changes to your remote repository's master branch by typing the command "git push origin master" into the console.

Throughout the process, VS Code shows the progress of commits and pushes as well as the status of files (untracked, updated, and staged) in the Source Control view. Version control activities are made easier by this integration, which enables developers to work together more easily and effectively with Git and GitHub right inside the comfortable VS Code environment.

**References**

[The Complete Guide to Installing VS Code | by Jack Fields | OrdinaryIndustries | Medium](https://medium.com/ordinaryindustries/the-complete-guide-to-installing-vs-code-6e03181b0554) <https://medium.com/ordinaryindustries/the-complete-guide-to-installing-vs-code-6e03181b0554>

[2 Exploring the User Interface - Visual Studio Code [Book] (oreilly.com)](https://www.oreilly.com/library/view/visual-studio-code/9781119588184/c02.xhtml) <https://www.oreilly.com/library/view/visual-studio-code/9781119588184/c02.xhtml>

[Get Started with Visual Studio Code](https://code.visualstudio.com/learn/) <https://code.visualstudio.com/learn/>