**Installation and Navigation of Visual Studio Code**

**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.

Visit the Visual Studio Code website (<https://code.visualstudio.com/>), navigate through it till you find the download button. Click the button.

Once downloaded, locate the downloaded file and run the installer.

Choose the destination folder where you want to install VS Code.

Choose the additional tasks you want the installer to perform

Click install to begin the installation process.

Once the installation is complete, you can choose to launch Visual Studio Code then click finish.

**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.

**Theme** choose a color theme that suits your preference, this can either be dark or light

**Icon Theme** select an icon theme to enhance visual differentiation of file types.

**Extensions** install the necessary extensions to assist in the day to day coding and workings using vs code.

They include;

Python

Prettier

ESLint for JavaScript and Typescript projects, it helps to find and fix problems in your code.

Live Server Provides a live preview of your HTML/CSS/JavaScript files with automatic refresh on changes.

Debugger for Chrome for JavaScript development, this extension allows you to debug your code directly in Chrome.

Docker Adds support for Docker containers, allowing you to work with Docker files and Docker Compose.

Remote SSH Allows you to use a remote machine as your development environment.

**Configuration Files Settings** here you can adjust settings like font size, tab size, and auto save preferences

**Git Integration** used to commit and push changes, make sure Git is installed on your system and initialize a repository in your project directory.

**Custom Code Snippets** these are snippets for commonly used code blocks.

**Workspace Configuration** Creates workspace settings for specific projects by saving settings in a code workspace file.

**Configure prettier** these add a prettier file to define formatting rules, they include single quote, trailing comma, print width and font size and styles

**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.

**Activity Bar**

It is the vertical bar on the far left of the interface and provides quick access to different views and functions within VS Code.

It shows your workspace and files, allows you to search across your files, integrates with version control systems like Git., manages debugging sessions, helps you browse and install VS Code extensions.

**Side Bar** it is located directly to the right of the Activity Bar and displays different views and panels depending on the selected activity.

It shows the file structure of your project, displays results from searches, shows changes, staged files and commit history.

**Editor Group** this is the central area where files and code are displayed and allows you to open and edit multiple files simultaneously.

Each open file is displayed as a tab at the top of the editor group, it allows you to split the editor vertically or horizontally to view and edit multiple files side by side.

Right-clicking within the editor provides options like formatting, refactoring, and running code snippets.

**Status Bar** it is the horizontal bar at the bottom of the interface and displays information about the current state of the editor and provides shortcuts to various functionalities.

It shows the current line number and column of the cursor, displays the file’s encoding e.g. UTF-8) and shows the current line-ending format e.g. CRLF

It shows the count of errors and warnings in the code and provides feedback options and shows notifications or alerts.

**Command Palette** it is accessible via Ctrl+Shift+Pand provides a quick way to access all commands and settings.It allows you to execute commands, navigate to files, or open settings without using the mouse.

**Panel** this is the bottom area of the interface, above the Status Bar and displays output, debugging information, integrated terminal and problems found in the code.

It shows output from various extensions and processes, displays debugging information and allows you to interact with the debugger.

**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.

**The Command Palette** is a powerful feature in Visual Studio Code that allows you to access and execute a wide range of commands quickly and efficiently. It provides a unified interface to perform various tasks without needing to navigate through menus.

You can open the Command Palette by selecting view from the top menu and then clicking command Palette or by clicking Ctrl+Shift+P.

Examples of Common Tasks Performed Using the Command Palette

Opening Files and Folders

Running Built-in Commands

Searching and Replacing

Source Control Integration

Debugging

Extension Management

Settings and Preferences

Terminal Management

Running Code Snippets

View and Layout Customization

**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.

Extensions in Visual Studio Code play a crucial role in enhancing its functionality and customizing the development environment to suit various programming needs. They allow users to add features, languages, debuggers and tools that are not included by default in VS Code.

**Language Support** extensions provide syntax highlighting, IntelliSense (code completion), linting, and debugging for various programming languages.

Examples Python, Java, C++, Ruby, and Go.

**Development Tools** they add tools that assist in development, such as linters, formatters and debuggers.

Examples: ESLint, Prettier, Docker, Live Server.

**Version Control Integration** these enhance Git integration with additional features like GitLens, which provides detailed insights into repositories.

Examples: GitLens, Git History.

**User Interface Enhancements** they improve the user interface and user experience with themes, icon packs, and custom layouts.

Examples: Material Theme, vs code icons.

**Productivity Boosters** this increase productivity with snippets, keybinding enhancements, and task runners.

Examples: Auto Close Tag, Path Intellisense

**Testing and Debugging** Offer testing frameworks and debugging tools tailored for specific languages and environments.

Examples: Jest, Debugger for Chrome.

To find extensions, click the Eextensions icon in the Activity Bar on the side of the window.

Use the search bar within the Extensions view to find specific extensions or browse through categories like popular, recommended and themes.

Through the Extensions View search for the desired extension, click on the extension in the list to open its details page then click the Install button.

**Essential Extensions for Web Development**

HTML Snippets provides snippets for HTML code.

CSS Peek allows you to quickly jump to a CSS definition in a separate file.

JavaScript code snippets offers useful snippets for JavaScript development.

React Native Tools provides debugging and intellisense for react native projects.

Vue.js Extension Pack collection of extensions for vue.js development.

Angular Essentials set of extensions for angular development.

ESLint integrates ESLint into VS Code for linting JavaScript and TypeScript code.

Prettier Code formatter automatically formats your code according to predefined rules.

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

Auto Rename Tag automatically renames paired HTML/XML tags.

Path Intellisense autocompletes filenames in paths.

Git history views git log, file history, compare branches or commits.

Material Theme a theme that provides a visually appealing UI.

**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?

The integrated terminal in Visual Studio Code (VS Code) allows you to run command-line tasks from within the editor. This is useful for compiling code, running scripts and using version control.

**Workspace Awareness** the integrated terminal opens in the context of the current workspace, automatically setting the working directory to the root of the project. This eliminates the need to navigate to the project directory manually.

**Project specific Configuration** environment variables and project-specific settings can be easily managed and applied within the integrated terminal.

**Single Interface** by keeping the terminal within the same window as the code editor, you avoid switching back and forth between different applications. This seamless integration saves time and reduces context-switching.

**Multitasking** you can run multiple terminals simultaneously, allowing you to run servers, scripts and other command-line tools concurrently without leaving the editor.

**Shortcuts and Commands** VS Code’s keyboard shortcuts and Command Palette make it easy to open, close, and manage terminals quickly.

**Integrated Tools extensions** that you use within VS Code can interact with the terminal, enhancing your workflow. For example, tasks and scripts defined in index. json can be executed directly from the terminal.

**Theming and Customization** the integrated terminal inherits the editor’s theme and settings, providing a consistent look and feel. You can customize the terminal’s appearance to match your preferences.

**Persistent Layout** the terminal layout and state are preserved across sessions, allowing you to resume work without reconfiguring your environment.

**Debugging Tools** when debugging, VS Code can launch terminals with debugging context, making it easier to run and inspect processes with debugging information directly accessible.

**Enhanced Functionality** many extensions enhance the capabilities of the integrated terminal, such as GitLens for Git operations or Docker for container management, providing a richer development environment.

**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?

**To create a new file** open the Explorer pane by clicking the Explorer icon in the Activity Bar on the left or by pressing Ctrl + Shift + E.Right-click in the Explorer pane and select new file, or click the new file icon a sheet of paper with a plus sign.Enter the name of the new file and press enter.

**Create a New Folder** right click in the Explorer pane and select new folder, or click the new folder icon a folder with a plus sign.Enter the name of the new folder and press enter.

**Managing Files and Folders**

**Renaming files and Folders** right click on the file or folder in the Explorer pane and select Rename. Enter the new name and press Enter.

**Moving files and folders** drag and drop files or folders within the Explorer pane to move them to a different location. Alternatively, right click and select cut or copy then navigate to the desired location, right click and select paste.

**Deleting Files and Folders** right-click on the file or folder and select delete. Confirm the deletion when prompted.

**Efficient Navigation Between Files and Folders\**

**Explorer Pane** Use the Explorer pane to quickly navigate and open files and folders in the current workspace.

**Quick Open** Press Ctrl + P to open the Quick Open menu. Start typing the name of the file you want to open. Quick Open will display a list of matching files in your workspace. Use the arrow keys to navigate and press enter to open the selected file.

**Go to Definition** place the cursor on a function, variable, or class name and press F12 to navigate to its definition. Alternatively, right-click and select go to definition.

**Go to File** Press Ctrl + T to open the go to file menu. This is similar to Quick Open and allows you to navigate to files quickly.

**Open Editors** Use the open editors section at the top of the Explorer pane to view and switch between currently open files.

**Recent Files** Press Ctrl + R to open the Recent Files menu, showing a list of recently opened files for quick access.

**Side-by-Side Editing** Split the editor by clicking the split editor icon in the top-right corner of the editor or by pressing Ctrl + \. This allows you to view and edit multiple files simultaneously.

**Integrated Terminal** Use the integrated terminal for file operations like navigating directories, creating, moving or deleting files and folders using command-line commands.

**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.

In Visual Studio Code, you can customize settings to tailor the environment to your preferences. The settings can be accessed and modified through the user interface or by editing the settings JSON file directly.

Go to file Preferences settings. This opens the Settings tab where you can search for and modify settings.

**Changing the Theme**

Via the Settings UI, open the Settings tab as described above, In the search bar type theme. Under color theme, click on the dropdown menu and select your preferred theme from the list.

**Changing the Font Size**

Via the Settings UI, open the Settings tab. In the search bar type font size. Under editor: Font Size, enter your desired font size.

**Editing the settings.jsn File**

Open the Command Palette, type preferences: open settings (JSON) and select it. Add or modify the following line to set the font size:

**Changing Keybindings**

Via the Keybindings UI, go to file, preferences, Keyboard Shortcuts. This opens the keyboard shortcuts editor. Find the command you want to change, right click on it and select change Keybinding. Press the keys you want to assign and press `Enter`.

**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?

Open VS Code and navigate to the folder containing your project.

Click on the debug icon in the Activity Bar on the side. Click on the gear icon to create a launch.json file if it doesn’t already exist, which holds your debug configurations. VS Code provides templates for common languages e.g. Python, JavaScript to help you get started with basic configurations.

Edit launch.json to specify how VS Code should launch your program. Define attributes such as the type of debugger, the program’s entry point, arguments, environment variables.

Set breakpoints in your code by clicking in the gutter next to the line number where you want to pause execution. Press F5 or click the green play button in the debug view to start debugging. This will launch your program in debug mode.

Once the program hits a breakpoint, you can, Execute the current line and move to the next line, If the current line calls a function, step into that function, Finish executing the current function and return to the line where it was called, Resume execution until the next breakpoint, Stop the program execution.

**Key Debugging Features in VS Code:**

**Breakpoints** Set breakpoints to pause execution at specific lines of code.

**Variable Inspection** Hover over variables to see their current values or use the Variables pane for more detailed inspection.

**Call Stack** View the function call hierarchy with the ability to navigate and inspect frames.

**Watch Expressions** Monitor specific variables or expressions continuously.

**Debug Console** Execute commands or evaluate expressions in the context of the current execution.

**Multi-threaded Debugging** Support for debugging applications that use multiple threads.

**Conditional Breakpoints** Set breakpoints that only trigger when specified conditions are met.

**Exception Handling** Configure VS Code to break execution when exceptions are thrown.

**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

Integrating Git with VS Code for version control involves several steps, including initializing a repository, making commits and pushing changes to GitHub.

**Install Git** First, ensure Git is installed on your system.

**Install** VS Code

**Open VS Code** Launch VS Code and open the folder or workspace where you want to initialize a Git repository.

**Initialize Git** Repository Open the Command Palette and type Git: Initialize Repository.

Select the folder you want to initialize as a Git repository.

**Add Files to the Repository** Start by creating or adding files to your project within the opened folder.

**Making Commits** in the Source Control view you’ll see a list of files with changes. Click the + icon next to each file you want to stage for the commit. Alternatively, stage all changes by clicking the + button at the top of the list.

**Commit Changes** after staging your changes, enter a commit message in the text box at the top of the Source Control view.

**Linking to a GitHub Repository** If you haven’t already set up a GitHub repository for your project, create one on GitHub.com.

**Add Remote Repository** in VS Code, go to the Command Palette, type Git: Add Remote and provide the URL of your GitHub repository.

**Push Changes** after committing your changes locally, push them to GitHub. Open the Command Palette, type Git: Push and select the remote branch to push either origin/main or origin/master.

If prompted, **authenticate** with your GitHub credentials username and password or token.

**Verify on GitHub** go to your GitHub repository page to verify that your changes have been successfully pushed.

