Assignment 2 , week 2

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.

Prerequisites

{i}Operating System: Ensure your system is running Windows 11 Operating System.

{ii}internet Connection: An active internet connection to download the installer.

Steps to Download and Install Visual Studio Code

(a)**Visit the Official Website:**

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

(b)**Download the Installer:**

Click on the "Download for Windows" button that automatically detects your operating system.

OR

Navigate to the "Download" section and select "Windows" to download the installer manually.

(c)**Run the Installer:**

Locate the downloaded file on the Downloads folder in your PC and double-click `VSCodeSetup.exe` to run the installer.

(d)**Install Visual Studio Code:**

A setup wizard will open. Follow the prompts:

(i)Accept the license agreement.

(ii)Choose the destination folder (or leave it as default).

(iii) Select any additional tasks (like creating a desktop icon) that you want.

(iv)Click "Next" through the setup screens and finally click "Install".

(e)**Complete Installation:**

You can choose to launch Visual Studio Code immediately by clicking "Finish".

(f)**First Launch:**

- If you didn't launch Visual Studio Code from the installer, you can find it in your Start menu or by searching for "Visual Studio Code".

1 First-time Setup for VS Code

After installing Visual Studio Code (VS Code), you should adjust some initial configurations and settings to optimize your coding environment. Here’s how you get started:

(a) User Settings

1. **Access User Settings:**

   - Press `Ctrl+Shift+P` to open the Command Palette.

   - Type `Preferences: Open Settings (UI)` and press Enter.

2. **Modify User Settings:**

   - You can customize settings such as editor font size, theme, auto-save, etc.

   - you might want to set `"editor.fontSize": 14` to adjust the font size.

(b) Workspace Settings

1. **Create Workspace Settings:**

   - Navigate to your project directory.

   - Create a `.vscode` folder if it doesn’t already exist.

   - Inside the `.vscode` folder, create a `settings.json` file.

2. **Customize `settings.json`:**

   - Add project-specific settings. For example:

```json
{
  "editor.tabSize": 2,
  "files.exclude": {
    "/.git": true,
    "/.DS_Store": true
  }
}
```

Extensions

Extensions are a crucial part of configuring VS Code to suit your development needs. Here are some recommended extensions based on language and functionality:

(a) Language-Specific Extensions

1. **Python:**
   - Microsoft Python Extension:
     - Install by searching for "Python" in the Extensions view (`Ctrl+Shift+X`).
     - This extension provides features like IntelliSense, linting, debugging, and more.

2. **JavaScript:**
   - ESLint:
     - Install by searching for "ESLint".
     - Integrates ESLint into VS Code to provide real-time linting and code quality checks.
   - Prettier:
     - Install by searching for "Prettier - Code formatter".
     - Ensures consistent code styling and formatting.

3. **HTML/CSS:**
   - Live Server:
     - Install by searching for "Live Server".
     - Launches a local development server with live reload feature for static and dynamic pages.

(b) **Code Formatting**

1. **Prettier:**
   - Prettier - Code formatter:
     - Install by searching for "Prettier - Code formatter".
     - Ensures consistent styling across your codebase.
     - Configure in `settings.json`:

```json
{
  "editor.formatOnSave": true,
  "prettier.singleQuote": true,
  "prettier.semi": false
}
```

2. **ESLint:**
   - ESLint Extension:
     - Install by searching for "ESLint".
     - Provides JavaScript linting and integrates with your project’s ESLint configuration.
     - Configure in `settings.json`:

```json
{
  "eslint.alwaysShowStatus": true,
  "editor.codeActionsOnSave": {
    "source.fixAll.eslint": true
  }
}
```

1. User Settings: Access and modify through `Ctrl+Shift+P` -> `Preferences: Open Settings (UI)`.

2. Workspace Settings: Create a `.vscode/settings.json` file in your project directory for project-specific configurations.

3. Extensions:

   - Language-Specific: Python (Microsoft Python), JavaScript (ESLint, Prettier), HTML/CSS (Live Server).

   - Code Formatting: Prettier (consistent styling), ESLint (linting and code quality for JavaScript).

By customizing these settings and installing the right extensions, you can create an optimal coding environment tailored to your needs in VS Code.

User Interface Overview

When going through the Visual Studio Code (VS Code) user interface, it's essential to understand its main components and their purposes. Here's an explanation from my perspective:

Main Components of the VS Code User Interface

1. **Activity Bar:**

The Activity Bar provides access to different views and controls, such as Run and Debug, the Explorer, Source Control, and Extensions. For example, when I want to see my project files, I click on the Explorer icon in the Activity Bar. If I need to debug my code, I can click on the Run and Debug icon.

2. **Side Bar:**

The Side Bar displays various panels depending on the activity selected from the Activity Bar. Common panels include the Explorer, where I can navigate through my project files, and the Search panel, where I can search for specific strings within my project.

3. **Editor Group:**

The Editor Group is the main area where I write and edit my code. I can open multiple files here, each in its tab. VS Code also allows me to split this area into several panes for side-by-side editing, which is super helpful when I need to compare or work on multiple files at once.

4. **Status Bar:**

The Status Bar displays information about the current state of the editor and my project. This includes details like the current file's encoding and line endings, the programming language I'm using, the Git branch, and any errors or warnings. For instance, when I see an error icon in the Status Bar, I know there are issues I need to address in my code. By understanding these components, I can navigate and use VS Code more effectively, making my coding experience smoother and more efficient.

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 in VS Code is a powerful tool that allows you to access all available commands and perform various tasks without leaving the keyboard.

Accessing the Command Palette:

- Press `Ctrl+Shift+P` (or `F1`).

Common Tasks:

- Opening Settings: Type "Preferences: Open Settings" to access and modify user or workspace settings.

- Installing Extensions: Type "Extensions: Install Extensions" to open the Extensions view and search for new extensions.

- Git Commands: Type "Git: Commit" to commit changes or "Git: Push" to push commits to a remote repository.

- Formatting Code: Type "Format Document" to format the entire file according to the configured formatter.

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 enhance the functionality of VS Code by adding features, tools, and integrations that support various programming languages and workflows.

Finding, Installing, and Managing Extensions:

- Finding Extensions: Click on the Extensions icon in the Activity Bar or press `Ctrl+Shift+X`.

- Installing Extensions: Search for the desired extension and click "Install".

- Managing Extensions: View installed extensions, disable, enable, or uninstall them from the Extensions view.

Essential Extensions for Web Development:

- Python: Microsoft Python extension for Python development.

- JavaScript: ESLint for linting and Prettier for code formatting.

- HTML/CSS: Live Server for launching a development server with live reload.

6. <u>Integrated Terminal</u>

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 VS Code allows you to run command-line tasks directly within the editor.

Opening the Integrated Terminal:

- Press `Ctrl+`` (backtick) or go to `View` > `Terminal`.

Using the Integrated Terminal:

- Execute commands, run scripts, and manage your project without leaving VS Code.

Advantages of the Integrated Terminal:

- Convenience: Access terminal commands within the same window as your code.
- Workspace Context: The terminal starts in the context of your workspace directory.
- Split Terminals: Open multiple terminal instances in different panes for better multitasking.

7. <u>File and Folder Management</u>

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:

- Create File: Right-click in the Explorer panel and select `New File` or press `Ctrl+N`.
- Create Folder: Right-click in the Explorer panel and select `New Folder`.

Opening Files and Folders:

- Open File: Press `Ctrl+O` and browse to the file.
- Open Folder: Go to `File` > `Open Folder` and select the folder.

Managing Files and Folders:

- Rename: Right-click on a file or folder and select `Rename`.
- Delete: Right-click and select `Delete`.

Efficient Navigation:

- Quick Open: Press `Ctrl+P` and start typing the file name to quickly open it.

- File Explorer: Use the Explorer panel to browse and manage files.

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.

Finding and Customizing Settings:

- Press `Ctrl+Shift+P` and type `Preferences: Open Settings` or go to `File` > `Preferences` > `Settings`.

Examples of Customizations:

- Change Theme: Search for "Color Theme" in the Command Palette and select a theme.

- Change Font Size: Add `"editor.fontSize": 14` to your settings.

- Change Keybindings: Go to `File` > `Preferences` > `Keyboard Shortcuts` and modify keybindings as needed.

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?

Steps to Set Up and Start Debugging:

1. Open your project: Ensure the code file you want to debug is open.

2. Set Breakpoints: Click in the gutter next to the line number to set breakpoints.

3. Open Debug View: Click on the Debug icon in the Activity Bar.

4. Configure Launch.json: If prompted, create a `launch.json` file to configure the debugger.

5. Start Debugging: Press `F5` or click the green play button in the Debug view.

Key Debugging Features:

- Breakpoints: Pause execution at specific lines to inspect variables and program state.

- Watch Variables: Monitor specific variables for changes.

- Call Stack: View the call stack to understand the sequence of function calls.

- Step Through Code: Use step over, step into, and step out to navigate through your code line by line.

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 allows you to manage version control directly within the editor, making it easier to track changes, collaborate with others, and maintain your project's history.

Integrating Git with VS Code:

1. **Install Git:**

   - Ensure Git is installed on your system. You can download and install it from git-scm.com.

2. **Initialize a Repository:**

   - Open your project folder in VS Code.

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

   - Click on the "Initialize Repository" button. This will create a new Git repository in your project folder.

3. **Making Commits:**

   - Stage Changes: In the Source Control view, you'll see a list of changes. Click the `+` icon next to each file to stage changes or click the `+` next to "Changes" to stage all changes.

   - Commit Changes: Enter a commit message in the message box at the top of the Source Control view, then click the checkmark icon or press `Ctrl+Enter` to commit the changes.

4. **Pushing Changes to GitHub:**

   - Create a GitHub Repository: Go to GitHub, create a new repository, and copy the repository URL.

   - Add Remote: In VS Code, open the integrated terminal (`Ctrl+``) and run:

     ```bash
     git remote add origin <repository-url>
     ```

     Replace `<repository-url>` with your GitHub repository URL.

   - Push Changes: To push your commits to GitHub, run:

     ```bash
     git push -u origin master
     ```

     The `-u` flag sets the upstream branch, so you can just use `git push` in the future.

1. Install Git: Download and install Git if not already installed.
2. Initialize a Repository: Use the Source Control view to initialize a Git repository in your project folder.
3. Make Commits: Stage changes and commit them with meaningful messages.
4. Push to GitHub: Add a remote repository on GitHub and push your changes.

If you follow these steps, you can effectively manage version control in your projects using Git and VS Code, allowing you to collaborate and maintain your project's history seamlessly.