PLP SOFTWARE ENGINEERING COURSE 2024

Student Name: Lindelani Xhanti : **Email:**lindelani@agrishelves.co.za | **Cell:**+27767123993

Dated: 21st June 2024

Assignment Name: Installation And Navigation Visual Studio Code (VS Code)

1. Installation of visual studio code

- **Download Visual Studio Code**:
  - Visit the official Visual Studio Code website https.:/code.virtualstudio.com
  - Choose the appropriate version for Windows (x64 or Arm64) and click the download link.
- **Run the Installer**:
  - Once the installer is downloaded, run it (usually named VSCodeUserSetup-{version}.exe).
  - Follow the installation prompts. By default, VS Code will be installed under C:\Users\{Username}\AppData\Local\Programs\Microsoft VS Code.
- **Launch Visual Studio Code**:
  - After installation, click the "Launch" button to open Visual Studio Code.

2. Configuration

- **Extensions**:
  - Install useful extensions from the VS Code Marketplace. Some popular ones include:
    - **Bracket Pair Colorizer**: Helps visualize matching brackets.
    - **GitLens**: Enhances Git integration.
    - **Prettier**: Auto-formats code.
    - **Live Server**: Facilitates web development. and
    - **Python** (which we recommended)
- **User Settings**:
  - Access user settings by clicking on the gear icon in the bottom left corner and selecting "Settings."
  - Customize settings globally for all VS Code instances.
  - Common settings to adjust:
    - "editor.tabSize": Set your preferred tab size (e.g., 2 or 4 spaces).
    - "editor.wordWrap": Choose how lines wrap (e.g., "on" or "off").
    - "editor.fontSize": Adjust font size.
    - "files.autoSave": Enable auto-saving files.
    - "workbench.colorTheme": Pick a color theme you like.
- **Workspace Settings**:
  - Workspace settings apply only to the current project/workspace.

  - Create a .vscode folder in your project root and add a settings.json file.
  - Customize settings specific to your project (e.g., linter rules, formatter settings).
- **IntelliSense**:
  - VS Code provides smart code completions as you type.
  - Ensure IntelliSense is enabled (usually enabled by default).

3. User Interface

- **Activity Bar:**
  - Located on the side (usually on the left), the Activity Bar provides quick access to different functionalities:
    - **Explorer**: Navigates files and folders in your workspace.
    - **Search**: Allows you to search across your codebase.
    - **Source Control**: Manages Git repositories and version control.
    - **Run and Debug**: Helps you run and debug your code.
    - **Extensions**: Accesses installed extensions.
    - **Remote Explorer**: Connects to remote servers or containers.
- **Side Bar:**
  - Adjacent to the Activity Bar, the Side Bar contains various panels:
    - **Explorer**: Displays your project's file structure.
    - **Source Control**: Shows Git changes and commits.
    - **Extensions**: Lists installed extensions.
    - **Outline**: Provides an overview of symbols in the current file.
    - **Debug**: Helps with debugging tasks.
    - **Remote Explorer**: Manages remote connections.
- **Editor Group:**
  - The central area where you edit files.
  - You can split it into multiple columns (using "Split Editor" or dragging files) for side-by-side editing.
  - Each column can have multiple tabs representing open files.
- **Status Bar:**
  - Located at the bottom, the Status Bar displays essential information:
    - **Language Mode**: Indicates the language of the current file.
    - **Line and Column Numbers**: Shows your cursor position.
    - **Git Branch**: Displays the active Git branch.
    - **Errors and Warnings**: Alerts you to issues in your code.
    - **Extensions**: Shows active extensions.

4. Commands Pallets in VS Code

The **Command Palette** in Visual Studio Code (VS Code) is a powerful tool that allows you to access various features, commands, and settings directly via keyboard shortcuts or a menu. To open the Command Palette, use the keyboard shortcut **Ctrl+Shift+P** (or **Cmd+Shift+P** on Mac). Once opened, you can start typing to search for specific commands by name. Here are some common tasks you can perform using the Command Palette:

- **Open File**:
  - Quickly open a specific file in the editor.
  - Type "Open File" in the Command Palette and select the desired file.
- **Search Symbols**:
  - Find specific symbols (functions, classes, variables) within your codebase.
  - Type "Search Symbols" and enter the symbol name to navigate to its location.
- **Run Task**:
  - Execute predefined tasks (e.g., build, linting, testing) related to your project.
  - Type "Run Task" and choose from available tasks (e.g., TypeScript compiler, linters).
- **Manage Extensions**:
  - Install, update, or remove VS Code extensions.
  - Type "Manage Extensions" to access extension-related commands.

Great examples are:

1. **Switch between open tabs**:
   - Windows/Linux: **Ctrl + Tab**
   - Mac: **Command + Option + Right Arrow**
2. **Move to the next open tab**:
   - Windows/Linux: **Ctrl + Page Down**
   - Mac: **Control + Tab**
3. **Move to the previous open tab**:
   - Windows/Linux: **Ctrl + Page Up**
   - Mac: **Control + Shift + Tab**
4. **Close the current tab**:
   - Windows/Linux: **Ctrl + W**
   - Mac: **Command + W**

5. Extensions in VS Code

Visual Studio Code (VS Code) extensions play an important role in enhancing my development experience. They allow me to add languages, debuggers, and tools to my installation, tailoring it to my specific needs. Here's how I find, install, and manage extensions:

- **Finding Extensions:**
  - Open VS Code and click on the **Extensions** icon in the Activity Bar (or use the shortcut **Ctrl+Shift+X**).
  - Browse the VS Code Marketplace to discover extensions. Each extension includes a brief description, publisher details, download count, and rating.
  - Search for extensions by typing keywords (e.g., "todo," "Python," "Git") in the search box.
- **Installing Extensions:**
  - Once I find an extension, click the **Install** button.
  - VS Code will download and install the extension.
  - After installation, the button changes to a **Manage gear** icon.
- **Managing Extensions:**
  - To disable or uninstall an extension, I go to the Extensions view and click the gear icon next to the installed extension.
  - Use the Command Palette (**Ctrl+Shift+P**) with commands like "Extensions: Disable" or "Extensions: Uninstall."
- Now, let's explore some essential extensions for web development:
- **Live Server**:
  - Launches a local development server for web projects, automatically refreshing the browser.
  - Great for HTML, CSS, and JavaScript development.
- **Prettier**:
  - Automatically formats your code to maintain consistent styling.
  - Supports various languages, including JavaScript, TypeScript, and CSS.
- **ESLint**:
  - Integrates ESLint for static code analysis and enforcing coding standards.
  - Essential for maintaining clean and error-free code.
- **Debugger for Chrome**:
  - Debugs JavaScript code directly in Chrome.
  - Useful for frontend development and debugging.

6. Integrated Terminal

The **integrated terminal** in Visual Studio Code (VS Code) is a powerful tool that streamlines your coding workflows. Here's how to use it and its advantages:

- **Opening the Integrated Terminal**:
  - To open the integrated terminal, use the keyboard shortcut **Ctrl+`** (backtick) or go to the menu: **View** → **Terminal**.
  - It starts at the root of your workspace, allowing you to execute commands directly within VS Code.
- **Advantages of the Integrated Terminal**:
  - **Seamless Integration**: Execute Git commands, manage branches, and perform version control operations directly within the editor environment.
  - **Convenience**: No need to switch between VS Code and an external terminal; everything happens in one place.
  - **Multiple Instances**: You can open multiple terminal instances for different tasks simultaneously.
  - **Error Detection**: The integrated terminal detects errors and provides clickable links for easy navigation.

Useful shortcuts

- **Ctrl + A**: Move to the start of the line.
- **Ctrl + E**: Move to the end of the line.
- **Ctrl + U**: Delete from the cursor to the start of the line.
- **Ctrl + K**: Delete from the cursor to the end of the line.
- **Ctrl + W**: Delete the word before the cursor.
- **Ctrl + L**: Clear the terminal screen.
- **Ctrl + C**: Stop the current process/command.
- **Ctrl + D**: Log out or exit the terminal.
- **Ctrl + Z**: Pause the current process (can be resumed).
- **Ctrl + R**: Search command history (backward search).
- **Up Arrow**: Show the previous command (from history).
- **Down Arrow**: Show the next command (from history).
- **!!**: Repeat the last command.
- **!n**: Repeat the nth command from history.
- **Tab**: Auto-complete commands, files, or directories.
- **Ctrl + Shift + C**: Copy selected text or command.
- **Ctrl + Shift + V**: Paste copied text or command.
- **Ctrl + Shift + N**: Open a new terminal window.
- **Ctrl + Shift + T**: Open a new tab in the terminal.
- **Ctrl + Tab** or **Ctrl + PageDown**: Switch between terminal tabs.

7. File and Folder Management

- **Creating Files and Folders**:
    - To create a new file, open VS Code and select **File** > **New File**.
    - To create a new folder, select **File** > **New Folder**.
    - Alternatively, use the keyboard shortcuts: **Ctrl+N** for a new file and **Ctrl+Shift+N** for a new folder.
- **Opening Files and Folders**:
    - Open a folder as a workspace by selecting **File** > **Open Folder…** and choosing a directory.
    - You can also launch VS Code from a terminal with the path to a folder: code . (current folder).
- **Managing Files and Folders**:
    - Use the Explorer (Side Bar) to view and manage files and directories.
    - Right-click to create, rename, delete, or move files/folders.
    - Organize logically to find files quickly.
- **Efficient Navigation**:
    - Learn basic terminal commands (e.g., cd, ls, dir) for command-line navigation.
    - Consider creating a logical folder structure to avoid clutter.

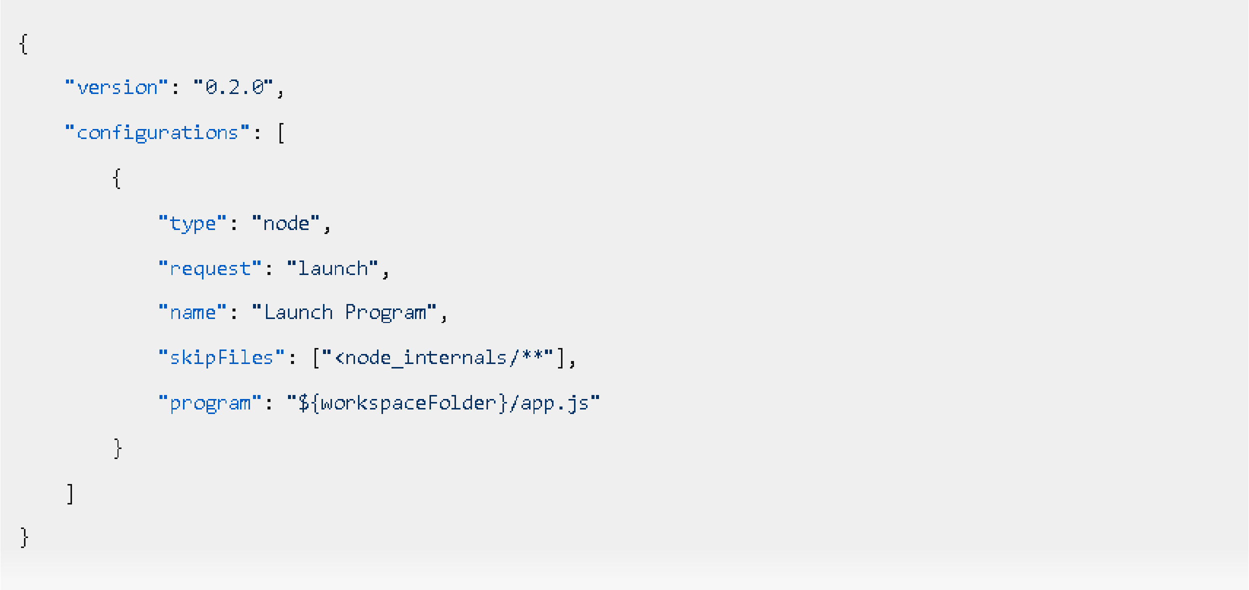
8. Setting and Preferences

- **Open Settings**:
    - Press **Ctrl + ,** (Windows) or **Cmd + ,** (Mac) to open the Settings menu.
    - Alternatively, go to **File** > **Preferences** > **Settings**.
- **Change Theme**:
    - Search for "Color Theme" in the search bar.
    - Click the edit icon next to the theme you want (e.g., "Dark+").
    - Choose your preferred theme from the dropdown.
- **Adjust Font Size**:
    - Search for "Font Size" in the search bar.
    - Edit the font size value (e.g., "14px").
- **Modify Keybindings**:
    - Search for "Keybindings" in the search bar.
    - Click "Edit Keybindings.json" to customize shortcuts.

9. Debugging In VS Code

1. **Create a Sample Application**:

    - First, create a sample Node.js application (e.g., a "Hello World" JavaScript file named app.js).
2. **Open the Run and Debug View**:
    - Click the **Run and Debug** icon in the Activity Bar (or use the shortcut **Shift+Cmd+D** on Mac or **Ctrl+Shift+D** on Windows/Linux).
    - The Run and Debug view displays relevant information and commands.
3. **Configure Launch Settings**:
    - For most debugging scenarios, create a launch.json file (located in a .vscode folder in your workspace).
    - To create it, select "Create a launch.json file" in the Run start view.
    - VS Code will try to detect your debug environment automatically, but you can choose it manually if needed.
4. **Sample launch.json Configuration for Node.js**:



```
{
    "version": "0.2.0",
    "configurations": [
        {
            "type": "node",
            "request": "launch",
            "name": "Launch Program",
            "skipFiles": ["<node_internals/**"],
            "program": "${workspaceFolder}/app.js"
        }
    ]
}
```

5. **Run and Debug**:
    - Press **F5** or select **Run and Debug** on the Debug start view.
    - VS Code will try to run your currently active file based on the configuration.
6. **Debugging Features**:
    - **Breakpoints**: Toggle breakpoints by clicking the editor margin.
    - **Data Inspection**: Examine variables, call stack, and watch expressions.
    - **Debug Console**: Evaluate expressions and interact with the debugger.

10. Source Control

- **Initialize a Repository**:
    - Open your project folder in VS Code.
    - Click the **Source Control** icon in the Activity Bar (or use the shortcut **Ctrl+Shift+G**).
    - Initialize a new Git repository by clicking the **Initialize Repository** button.
    - Alternatively, you can open an existing Git repository by clicking **Open Repository**.
- **Stage and Commit Changes**:
    - In the Source Control view, you'll see three sections:
        - **CHANGES**: Unstaged changes (modified files).
        - **STAGED CHANGES**: Files ready for commit.
        - **MERGE CHANGES**: Incoming changes from other branches (if any).
    - Click the "+" icon next to a file to stage it for commit.
    - Enter a commit message in the text box above the changes.
    - Press **Ctrl+Enter** (or **Cmd+Enter** on Mac) to commit the changes.
- **Configure Git Username and Email**:
    - Ensure your Git username and email are set. If not, configure them using:
    - git config --global user.name "Lindelani Xhanti"
    - git config --global user.email onebiz.innovation@gmail.com

- **Create a Remote Repository on GitHub**:
    - Go to GitHub and create a new repository.
    - Copy the repository URL (e.g., https://github.com/yourusername/your-repo.git).
- **Link Local Repository to Remote**:
    - In VS Code, click the **...** menu in the Source Control view.
    - Choose **Publish to GitHub** and paste the repository URL.
    - Follow the prompts to authenticate and link your local repo to the remote.
- **Push Changes to GitHub**:
    - After committing changes, click the **...** menu again.
    - Select **Push** to push your commits to GitHub.
    - Provide your GitHub credentials if prompted.

Disclaimer.

The data contained from this assignment was sourced from the PLP study guides, VS Code website and W3sschools.com