Assignment Instructions: Implementing CSS-Border Styling

Objective

In this assignment, you are required to implement CSS rules to apply different types of borders (`solid`, `dotted`, and `dashed`) to HTML elements.

You will also be tested using a custom JavaScript test case to ensure that your CSS meets the required styles.

Instructions

1. Implementing the CSS-Border Styling

Implementing the CSS Border Styling

You will start with a partially completed `index.html` file. Your task is to write the CSS rules to implement the following styles for the `.solid-border`, `.dotted-border`, and `.dashed-border` classes:

1. CSS Structure: The CSS rules should be written inside the `<style>` tag, which is already provided in the `<head>` section of the HTML file.

2. The HTML structure is already present, and you only need to implement the CSS rules for the border styles.

3. The CSS file should implement the following styling:

- The `.solid-border` class should have a solid black border.

- The `.dotted-border` class should have a blue dotted border.

- The `.dashed-border` class should have a red dashed border.

- Ensure that each class also includes padding inside the border to space the content away from the edges.

2. Code: CSS-Border Styling Implementation

Here is the exact CSS code you need to implement inside your index.html file:

3. Explanation of the CSS Code

CSS Structure and Explanation:

- In the provided code, the CSS rules should be written inside the `<style>` tag in the `<head>` section.

- For the `.solid-border` class:

    - Set the `border` property to `5px solid black` to create a solid black border.

    - Add `padding: 10px` to provide space inside the border.

- For the `.dotted-border` class:

    - Set the `border` property to `3px dotted blue` to create a blue dotted border.

    - Add `padding: 10px` for the content's inner spacing.

- For the `.dashed-border` class:

    - Set the `border` property to `3px dashed red` to create a red dashed border.

    - Add `padding: 10px` to create space inside the border.

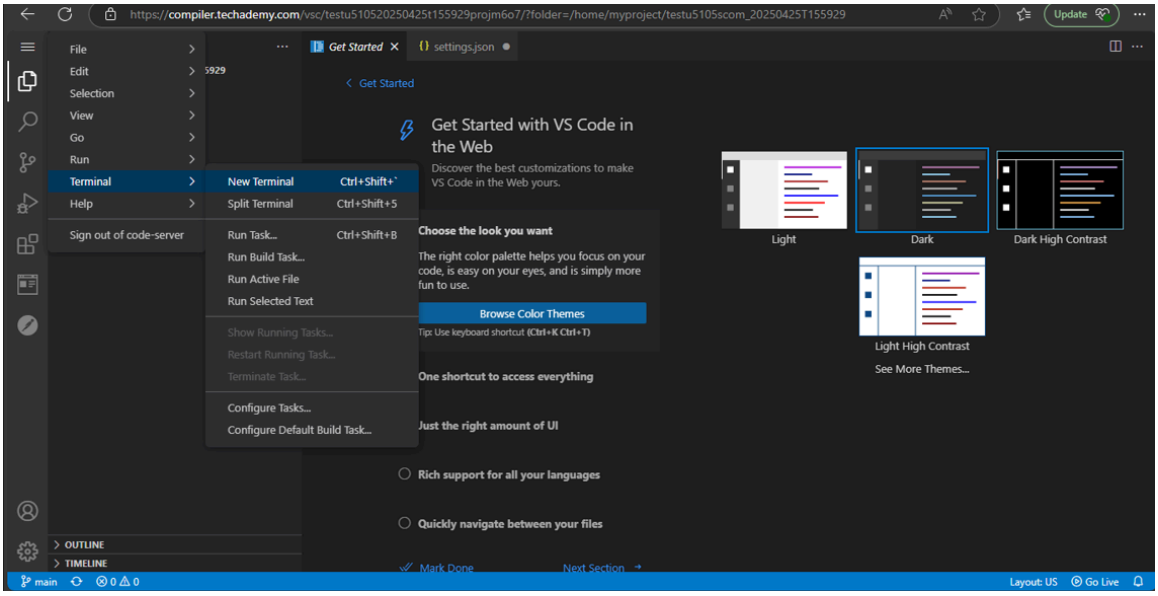
Assessment Guidelines

Step 1:

- Once the VS Code interface loads in the browser, wait until you see the workspace and left sidebar.
- To open the command terminal the test takers, need to go to

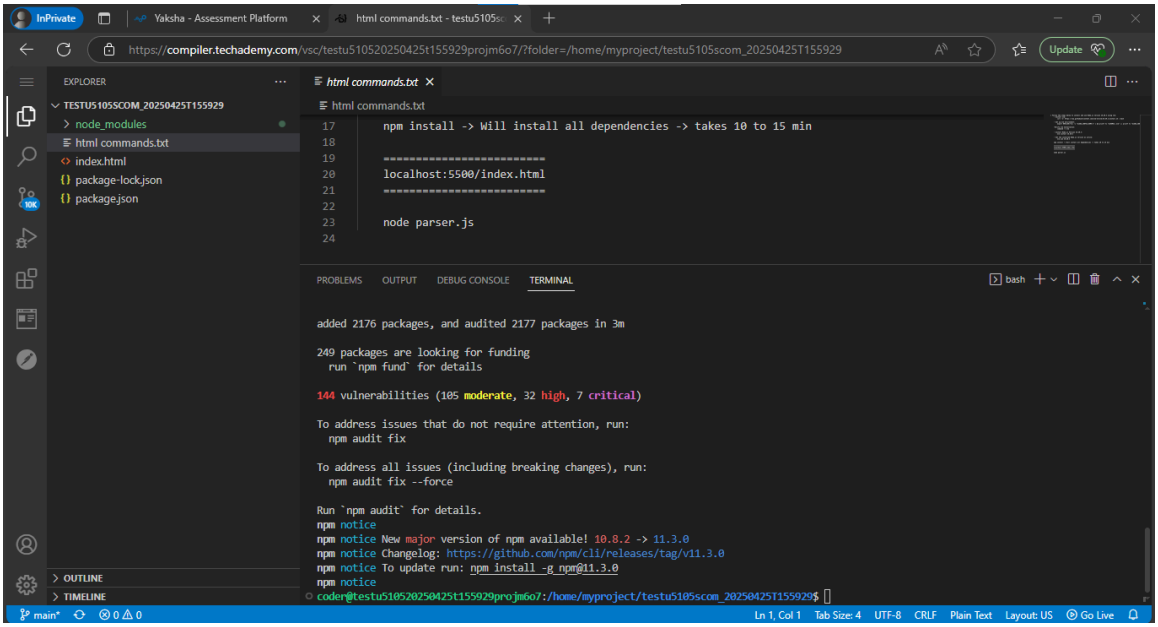
  Application menu (Three horizontal lines at left top) -> Terminal ->New Terminal.

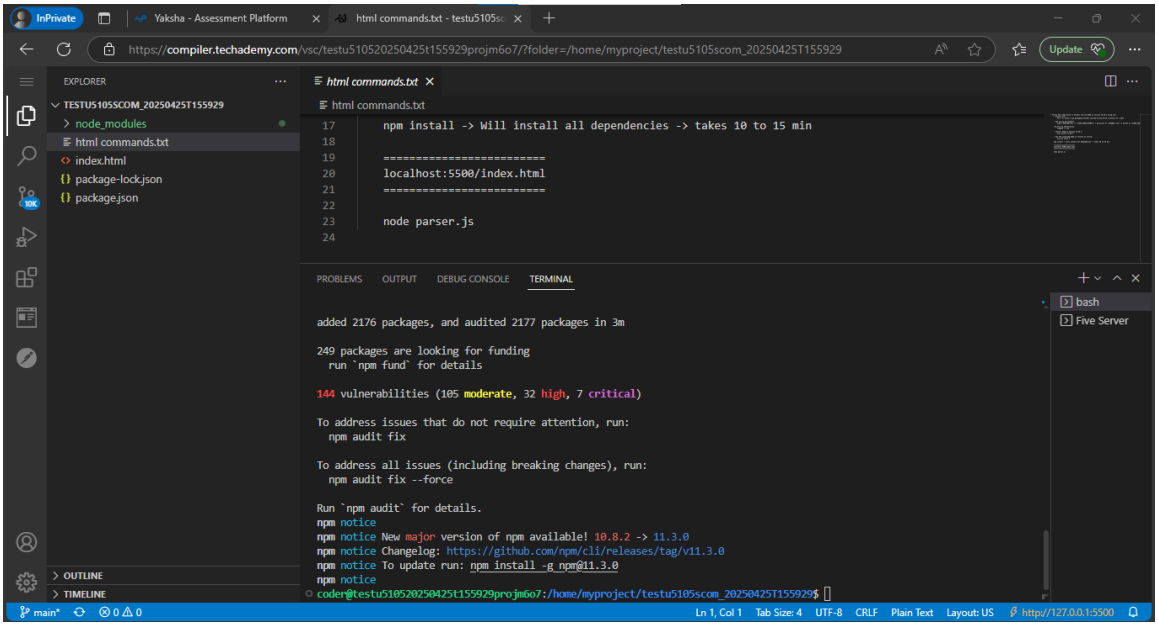
Now in the terminal you need to install all dependencies using the “npm install” command.



Step 2:

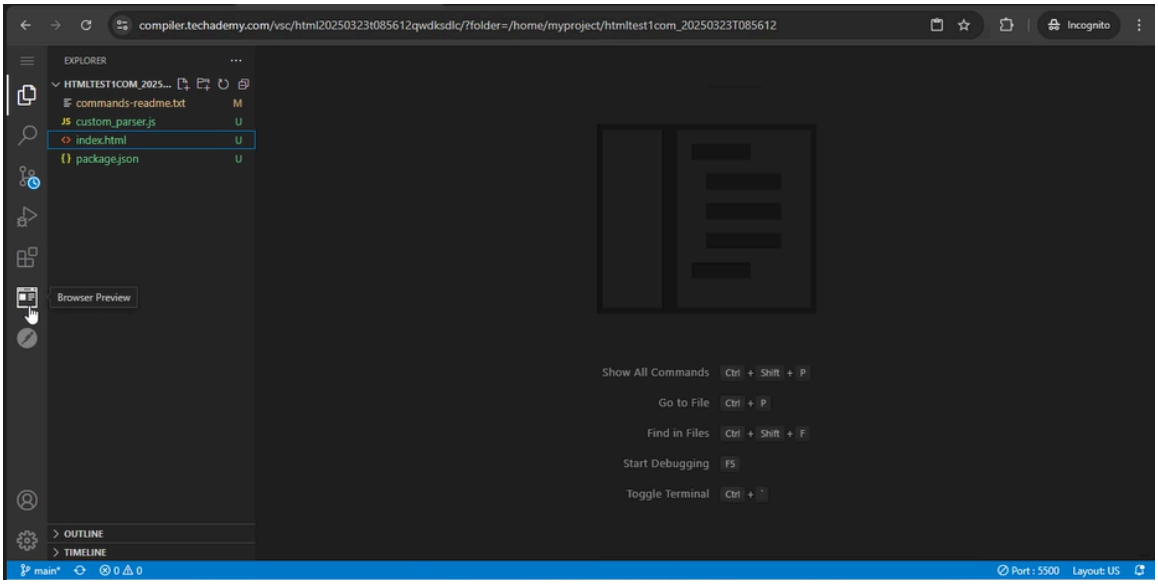
- Once installation completes, go to the **bottom right corner** of the VS Code screen.
- Click the **"Go Live"** button – This will start a **live server**, The server will run at port `5500` (e.g., `http://localhost:5500/`)





Step 3: Preview Output in Browser

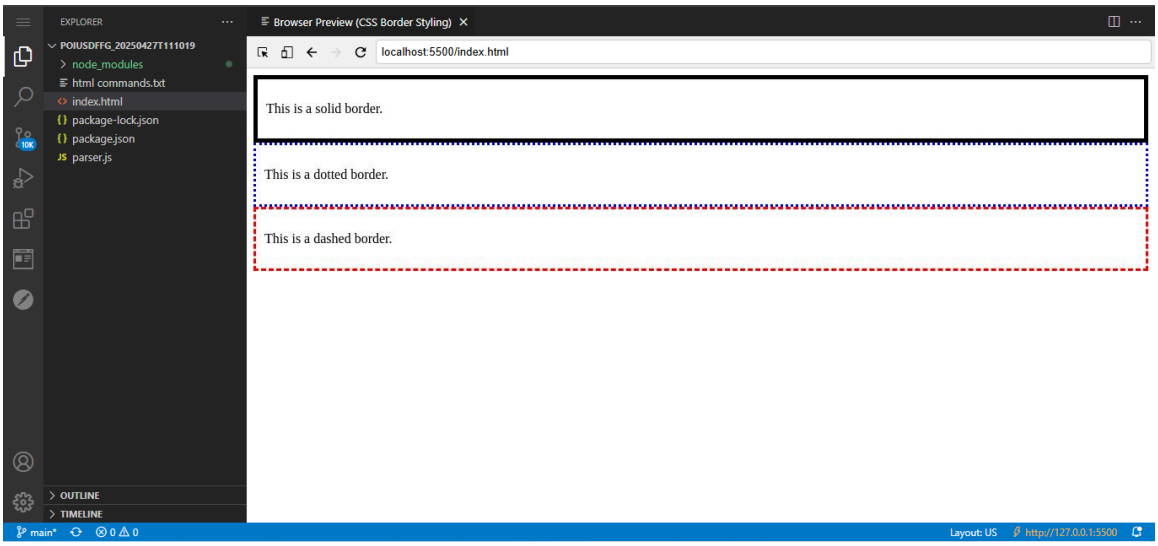
- This is a **web-based application**, so to view it in a browser, use the **internal browser inside the workspace**.
- Click on the **second last icon on the left panel** (the one labeled **"Browser Preview"**). This will open a tab within VS Code where you can **launch and view your application**.
- Note: The application **will not open in your system's local browser** — it must be viewed using the internal browser.



In the **Browser Preview tab**, type the following URL in the address bar and press **Enter**:

Your file is being served on: `localhost:5500/index.html`

This will load your HTML file and display the output of your web page **inside the internal browser**.



Step 4:

- Go back to the **terminal** and type the following command, then press **Enter**:

  node parser.js

- This command will **execute the validation script** and display the test results for your HTML file in the terminal.

Mandatory Assessment Guidelines:

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) -> Terminal ->New Terminal.**
3. **This editor Auto Saves the code.**
4. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**
5. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**
6. **This is a web-based application, to run the application on a browser, use the internal browser in the workspace. Click on the second last option on the left panel of IDE, you can find Browser Preview, where you can launch the application.**

   Note: The application will not run in the local browser
7. **You can follow series of command to setup HTML environment once you are in your project-name folder:**
   a. **npm install -> Will install all dependencies -> takes 10 to 15 min.**

b. **localhost:5500/index.html -> This will load your HTML file and display the output of your web page inside the internal browser.**

c. **node parser.js -> to run all test cases. It is mandatory to run this command before submission of workspace -> takes 5 to 6 min.**

8. **Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on "Submit Assessment" after you are done with code.**
9. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**
10. **If `Ctrl + Shift + B` doesn't work, then manually run the following commands one by one in the terminal:**
    - ➔ **git add .**
    - ➔ **git commit -m "Final commit"**
    - ➔ **git push**