Registration: _______________ Name: _________________________

May 13, 2025 **Web Application Development** Total Marks: 30

Q2: You are tasked with creating a "Product" section for an e-commerce website (like daraz). This section will display a single product with its image, name, a short description, its price, and a call-to-action button. This question has 2 parts. [CLO 2][5+5=10]

a) HTML Structure (10 points):

- Create a main container for the "Product" section using an appropriate semantic tag.
- Inside, include an image of the product. Use a placeholder image. Provide meaningful alt text.
- Display the product name using an appropriate heading level.
- Include a short descriptive paragraph (2-3 sentences).
- Display the product price.
- Include a button (e.g., "View Details").
- Use appropriate semantic HTML5 tags for structuring this content.

b) CSS Styling (10 points):

- Style the main container to have a light background color and a thin, solid border.
- The container should have 20px of padding internally.
- The product image should be on the left, and the text content (name, description, price, button) should be on the right. The image should not exceed 40% of the container's width. Do not use float.
- The product name should be bold and a larger font size.
- The price should be a distinct color and also bold.
- The button should have a contrasting background color, white text, 10px 15px padding, no text underline, and rounded corners.
- Ensure there's appropriate spacing between the text elements (name, description, price).

Q3: You are building an interactive task list application. This question has 3 parts. [CLO 3][10+5+5=20]

Given HTML:

```html
<div class="task-app">
  <h1>My Tasks</h1>
  <input type="text" id="task-input" placeholder="Add a new task...">
  <select id="priority-select">
    <option value="low">Low</option>
    <option value="medium" selected>Medium</option>
    <option value="high">High</option>
  </select>
  <button id="add-task-btn">Add Task</button>
  <ul id="task-list">
    </ul>
  <p id="stats">Total tasks: 0 | High priority: 0</p>
</div>
```

a) Add Task Functionality:

- Write JavaScript code that runs when the DOMContentLoaded event fires.
- When the "Add Task" button (#add-task-btn) is clicked:
  1. Get the text from #task-input and the selected priority from #priority-select.
  2. If the task input is empty, do nothing.
  3. Otherwise, create a new <li> element.
  4. The <li> should display the task text and its priority (e.g., "Do Homework [High]").
  5. Add a data attribute to the <li> to store its priority (e.g., data-priority="high").
  6. Append the new <li> to the #task-list.
  7. Clear the #task-input field.
  8. Update the statistics in the #stats paragraph (see Part B).

b) Update Statistics:

- Create a function updateStats() that counts the total number of tasks and the number of tasks with "high" priority.
- This function should update the text content of the #stats paragraph accordingly (e.g., "Total tasks: 5 | High priority: 2").
- Call this function whenever a task is added.

c) Event Delegation for Task Removal:

- Modify your "Add Task" functionality: Each new <li> should also contain a "Remove" button (e.g., <button class="remove-btn">Remove</button>).
- Using **event delegation** on the #task-list, make it so that clicking a "Remove" button removes its parent <li> from the list.
- After removing a task, call updateStats() again.