# PPA4: Enhancing Client-Side Interaction

## 1 Purpose of This Assignment

In PPA4 you will build on your PPA3 Node server. The server remains simple and synchronous. The primary goal this week is to significantly improve the graphical user interface and client-side using statements.

This assignment emphasizes control flow, conditional logic, and DOM manipulation. You must use only synchronous, vanilla JavaScript. No frameworks. No async or fetch.

## 2 Folder Structure Requirement

Your project must follow this exact structure.

```
ppa4/
│
├── server.js
├── package.json
├── public/
│     ├── index.html
│     ├── style.css
│     └── script.js
│
└── README.txt
```

Your server.js must serve the files inside the public folder exactly as you did in PPA3. The focus of grading is on `index.html`, `style.css`, and `script.js`.

## 3 Server-Side Expectations

You may reuse your server.js from PPA3. It should:

- Serve static files using the built in http and fs modules.
- Remain fully synchronous in logic.
- Not introduce new asynchronous features.

## 4 Client-Side Requirements

Your interface must include at least:

- Three input elements.
- Two buttons that trigger different logic paths.
- One dynamic output area that updates without page reload.
- You must demonstrate multiple `if` and `else` statements, nested conditionals, and clear block structure.

## 5 Using `document.getElementById()`

The browser builds a Document Object Model, or DOM, when it loads your HTML. The method `document.getElementById` allows you to directly access an element by its id attribute, and returns a reference to the element. Because it is synchronous, execution continues step by step. You can then read properties such as value or modify properties such as innerText or style.

## 6 Example Starter script.js

```
const input1  = document.getElementById("input1");
const input2  = document.getElementById("input2");
const input3  = document.getElementById("input3");
const buttonA = document.getElementById("buttonA");
const buttonB = document.getElementById("buttonB");
const output  = document.getElementById("output");

buttonA.onclick = function() {
    const value1 = Number(input1.value);
    const value2 = Number(input2.value);
    const value3 = Number(input3.value);

    if (value1 > value2) {
        if (value1 > value3) {
            output.innerText = "Input1 is largest";
        } else {
            output.innerText = "Input3 is largest";
        }
    } else {
        output.innerText = "Input2 may be largest";
    }
};

buttonB.onclick = function() {
    if (input1.value === "" || input2.value === "" || input3.value === "") {
        output.innerText = "All fields must be filled";
    } else {
        output.innerText = "All fields contain values";
    }
};
```

## 7 GUI Improvement Expectations

Compared to PPA3, your interface must demonstrate thoughtful layout and usability improvements. Use CSS to improve spacing, alignment, and visual clarity.

Your GUI must clearly communicate what the user should do. Labels must be explicit. Error messages must be informative.

## 8 Reflection Prompts

In your README.txt include answers to the following:

1. Explain how JavaScript statements control execution flow in your program.
2. Identify where nested conditionals occur and why they are necessary.
3. Explain how document.getElementById connects user interface elements to program logic.
4. Describe how your GUI design decisions improve usability compared to PPA3.
5. Connect your implementation to this week's reading on statements. What concepts did you apply directly?

## 9 Submission Checklist

1. Project runs with node server.js.
2. All logic is synchronous.
3. Uses only vanilla JavaScript.
4. Includes nested conditionals.
5. Includes improved GUI styling.