I.

1. F。alert 語法屬於 window 的 web api，node.js 環境沒有支援。
2. T。
3. T。
4. T。
5. F。undefined 代表未定義，null 則是空值。
6. F。React 是 javascript 的框架，屬於前端應用。Node.js 則是後端技術。
7. T。
8. F。margin 是外邊距，padding 才是內邊距。
9. T。
10. T。

II.

1. C
2. C
3. A
4. A
5. B

III.

1. https://codepen.io/Wang-Wei-Li/pen/GRLdxxw
2. https://codepen.io/Wang-Wei-Li/pen/zYXjjQy
3. 見 III.3 資料夾
4.
    a. 見 /III.4/a 資料夾
    b. 見 /III.4/b 資料夾
    c.
        (1) 差異：jQuery 是一個以 Javascript 來編寫的函式庫，相較於原生 Javascript 提供了更簡潔的 API 和豐富的插件擴展功能。
        (2) 適用情境：只需少量 DOM 操作且無需引入額外函式庫時，使用 Javascript 可減少頁面加載時間和資源消耗；jQuery 則適合需要大量 DOM 操作或動畫效果支持的項目。
5. 見 III.5 資料夾

IV.

1. Console.log 處錯掉了，index + 1 應該括號起來。

   修正版：

   ```
   const fruits = ['Apple', 'Banana', 'Orange'];

   fruits.forEach((fruit, index) => {
     console.log((index + 1) + '. ' + fruit);
   });
   ```

2. div 沒有指定 id，但 el 後面已經指定 #app 了。

   修正版：

   ```
   <div id="app">
     <p>{{ message }}</p>
   </div>
   <script>
     new Vue({
       el: '#app',
       data: {
         message: 'Hello, World!'
       }
     });
   </script>
   ```

3. <h1> 的結束標籤應該是 </h1> 而不是 </h2>

   修正版：

   ```
   import React from 'react';
   import ReactDOM from 'react-dom';

   const element = <h1>Hello, World!</h1>;
   ```

```
ReactDOM.render(element, document.getElementById('root'));
```

V.

1. RESTful API 是一種基於 REST（Representational State Transfer）架構風格設計的 API。它利用 HTTP 協議的方法（如 GET、POST、PUT、DELETE 等）來操作網路資源，實現無狀態的、面向資源的操作。
   a. 特點：
      (1) 無狀態性：每個請求都是獨立的，伺服器不會存儲客戶端的狀態，每個請求應該包含所有必要的信息。
      (2) 面向資源：通過 URL 定位資源，使用 HTTP 方法操作資源。
      (3) 統一接口：通過統一的接口進行操作，簡化系統的設計和實現。
      (4) 可擴展性：通過超連結（Hypermedia as the Engine of Application State, HATEOAS）提供可發現性和擴展性。
   b. 應用場景：
      (1) Web 應用：如前端與後端之間的數據交互。
      (2) 移動應用：如移動應用與伺服器之間的通訊。
      (3) 物聯網：如設備與伺服器之間的數據傳輸。
   c. 示例：
      (1) GET /users：獲取所有用戶信息。
      (2) POST /users：創建新用戶。
      (3) PUT /users/{id}：更新指定 ID 的用戶信息。
      (4) DELETE /users/{id}：刪除指定 ID 的用戶。

2. React：
   a. 優點：
      (1) 靈活性高：提供了一個靈活的框架，允許開發者選擇自己喜歡的工具和庫。
      (2) 虛擬 DOM：通過虛擬 DOM 減少實際 DOM 操作，提高性能。
      (3) 社區支持：擁有強大的社區和豐富的生態系統。
      (4) 組件化：基於組件的開發模式，易於重用和維護。
   b. 缺點：

(1) 學習曲線陡峭：需要掌握 JSX 語法和相關工具（如 Webpack、Babel）。

(2) 過於靈活：可能導致不同開發者之間的代碼風格不一致。

Vue.js：

a. 優點：

(1) 簡單易學：語法簡單，適合新手入門。

(2) 雙向數據綁定：自動同步數據和視圖，提高開發效率。

(3) 組件化：和 React 一樣，基於組件的開發模式，易於重用和維護。

(4) 文檔完善：提供了詳細的官方文檔和示例。

b. 缺點：

(1) 生態系統相對較小：相比 React，其生態系統和社區支持稍弱。

(2) 靈活性稍低：相比 React，Vue 的靈活性稍低，但這也使其更易於上手和統一風格。

3. INNER JOIN：

a. 定義：只返回兩個表中鍵值匹配的行。

b. 使用場景：當需要獲取兩個表中相關聯的數據，並且關注的是兩者都有的數據。

c. 示例：

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

上述查詢返回所有客戶及其對應的訂單，僅當客戶和訂單都有匹配時返回。

LEFT JOIN：

a. 定義：返回左表中的所有行，即使右表中沒有匹配的行，未匹配的右表數據用 NULL 填充。

b. 使用場景：當需要獲取左表的所有數據，並且可能會有右表數據缺失的情況。

c. 示例：

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

上述查詢返回所有客戶及其對應的訂單，即使某些客戶沒有訂單，這些客戶也會被返回，訂單信息用 NULL 填充。

4. 子查詢：
   a. 定義：一個嵌套在另一個查詢中的查詢。通常用括號括起來，可以在 SELECT、INSERT、UPDATE、DELETE 語句中使用。
   b. 作用：用於分解複雜的查詢，進行多步驟的數據過濾和計算。
   c. 使用方法(示例)：
      (1) 在 SELECT 中使用：

```
SELECT CustomerName
FROM Customers
WHERE CustomerID IN (SELECT CustomerID FROM Orders WHERE
OrderDate > '2023-01-01');
```

上述查詢返回所有在 2023 年 1 月 1 日之後有訂單的客戶。

      (2) 在 FROM 中使用：

```
SELECT sub.CustomerName, sub.TotalOrders
FROM (SELECT CustomerID, COUNT(OrderID) AS TotalOrders
        FROM Orders
        GROUP BY CustomerID) AS sub
JOIN Customers ON sub.CustomerID = Customers.CustomerID;
```

上述查詢返回每個客戶的總訂單數量。

      (3) 在 WHERE 中使用：

```
SELECT CustomerName
FROM Customers
WHERE EXISTS (SELECT 1 FROM Orders WHERE
Customers.CustomerID = Orders.CustomerID);
```

上述查詢返回至少有一筆訂單的客戶。

5. GROUP BY：
   a. 定義：將結果分組，對每組應用聚合函數（如 COUNT、SUM、AVG 等）。
   b. 作用：用於按一個或多個欄位對數據進行分組，通常與聚合函數一起使用。
   c. 示例：

```
SELECT CustomerID, COUNT(OrderID) AS OrderCount
FROM Orders
GROUP BY CustomerID;
```

上述查詢按客戶分組，計算每個客戶的訂單數量。

HAVING:

a. 定義：用於過濾 GROUP BY 的結果，作用類似於 WHERE，但 HAVING 是針對聚合結果進行過濾。
b. 作用：在聚合後的數據集上進行條件過濾。
c. 示例：

```
SELECT CustomerID, COUNT(OrderID) AS OrderCount
FROM Orders
GROUP BY CustomerID
HAVING COUNT(OrderID) > 5;
```

上述查詢返回訂單數量大於 5 的客戶。