Aim

The aim of this lab is to create a simple, interactive web application simulating a basic bank account interface. The application will display an account balance and feature "Deposit" and "Withdraw" buttons.

Objectives

1. **Structure the application:** Use HTML to create a well-structured page that includes a main container, a heading, a display for the balance, and a section for action buttons.
2. **Style the interface:** Use CSS to design a visually appealing and organized interface, including styling the background, the main card, text, and buttons.
3. **Ensure responsiveness:** Use CSS media queries to adjust the layout and font sizes for optimal viewing on smaller screens.
4. **(Implicit) Prepare for functionality:** Create a foundation with HTML and CSS that is ready for future JavaScript implementation to handle deposit and withdrawal logic.

Theory

This project focuses on the fundamental concepts of front-end development:

- **HTML Structure:** The div tag is used to create a "bank-container" that holds all the other elements. An <h2> tag provides a clear heading, and other div elements are used to group the balance display and the action buttons.
- **CSS Styling:** The styles are applied to make the interface look like a modern card. This involves setting background colors, using a responsive layout (even for a simple card), and styling buttons with distinct colors to represent their actions. A key part of the styling is using media queries to ensure the design adapts to mobile devices, preventing elements from becoming too large or small.
- **User Interface (UI) Design:** The use of green for "Deposit" and red for "Withdraw" is a common UI practice to visually communicate the actions of adding and subtracting, respectively.

Procedure

1. **HTML Structure:** Create the index.html file with a <head> section containing a <title> and a <style> block.
2. **CSS Styling:**
    - In the <style> block, define a background color for the <body>.
    - Style the .bank-container to create a card-like appearance with a white

background, padding, and a box shadow.
    - Style the <h2> and .balance-box for clear typography.
    - Style the .btn class for the buttons, giving them rounded corners and padding.
    - Apply unique background colors and hover effects for the .btn.deposit and .btn.withdraw buttons.
    - Add a @media query for screens with a maximum width of 500px to make font sizes and padding more suitable for mobile devices.
3. **Testing:** Open the index.html file in a web browser. Verify the visual layout is correct. Test the hover effect on the buttons and resize the browser window to see the responsive adjustments.

Code

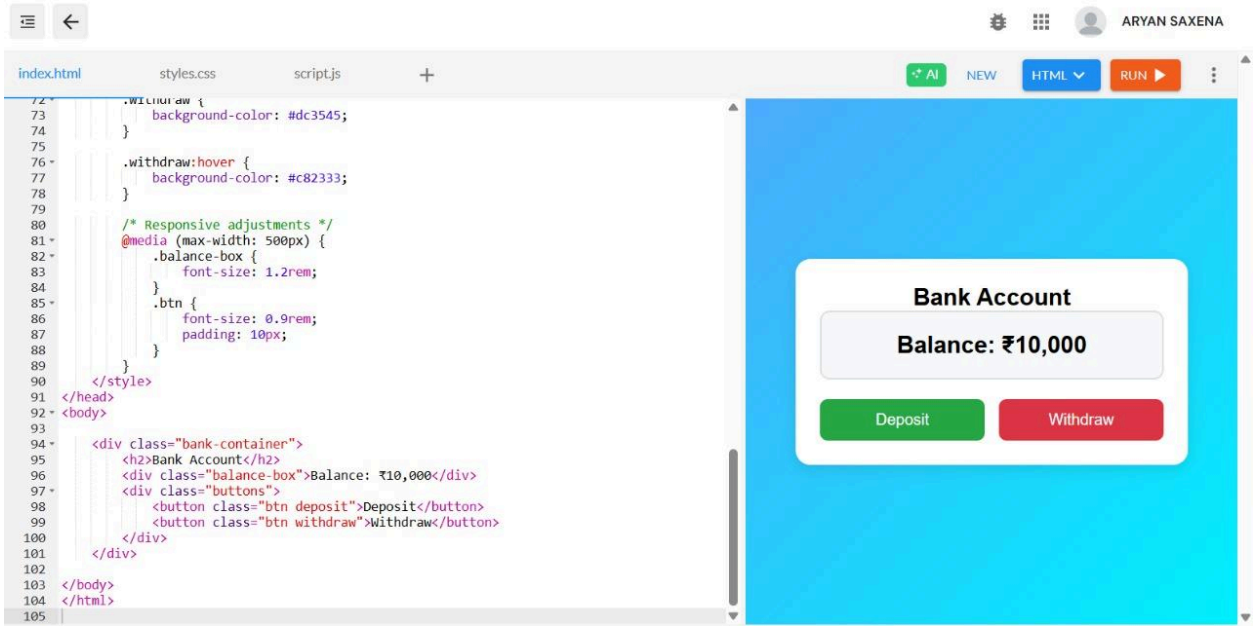
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Bank Account</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: linear-gradient(to bottom right, #a0d4ff, #8ab5d8);
      display: flex;
      justify-content: center;
      align-items: center;
      min-height: 100vh;
      margin: 0;
    }

    .bank-container {
      background-color: #ffffff;
      padding: 30px;
      border-radius: 15px;
      box-shadow: 0 10px 20px rgba(0, 0, 0, 0.1);
      text-align: center;
      width: 90%;
      max-width: 400px;
    }
```

```
h2 {
    font-size: 2em;
    margin-bottom: 20px;
    color: #333;
}

.balance-box {
    background-color: #f1f1f1;
    padding: 20px;
    border-radius: 10px;
    margin-bottom: 25px;
    font-size: 1.5em;
    font-weight: bold;
    color: #555;
    box-shadow: inset 0 2px 4px rgba(0,0,0,0.05);
}

.buttons {
    display: flex;
    justify-content: space-around;
    gap: 15px;
}

.btn {
    border: none;
    padding: 15px 30px;
    font-size: 1em;
    font-weight: bold;
    color: #ffffff;
    border-radius: 8px;
    cursor: pointer;
    transition: background-color 0.3s ease;
    flex-grow: 1;
}

.btn.deposit {
    background-color: #4CAF50;
}
```

```
        .btn.deposit:hover {
            background-color: #45a049;
        }

        .btn.withdraw {
            background-color: #f44336;
        }

        .btn.withdraw:hover {
            background-color: #da3329;
        }

        /* Responsive adjustments */
        @media (max-width: 500px) {
            h2 {
                font-size: 1.5em;
            }
            .balance-box {
                font-size: 1.2em;
            }
            .btn {
                padding: 12px 20px;
                font-size: 0.9em;
            }
            .bank-container {
                padding: 20px;
            }
        }
    </style>
</head>
<body>
    <div class="bank-container">
        <h2>Bank Account</h2>
        <div class="balance-box">Balance: ₹10,000</div>
        <div class="buttons">
            <button class="btn deposit">Deposit</button>
            <button class="btn withdraw">Withdraw</button>
        </div>
    </div>
```

</body>
</html>

Output



Learning Outcomes

Upon successful completion of this lab, the student will be able to:

1. **Develop a functional layout:** Create a structured and well-organized HTML layout for an application interface, using appropriate semantic tags.
2. **Apply theme-based styling:** Style a web component using CSS to create a professional and intuitive user experience, including color choices that align with button actions.
3. **Implement responsive design:** Use CSS media queries to ensure the application's interface remains usable and visually appealing across different device screen sizes.