INDEX.HTML

The code provided is an HTML document that creates a simple web page structure. Here's a breakdown of what the code does:

1. The `<!DOCTYPE html>` declaration defines this document as an HTML5 document.

2. The `<html>` element is the root element of the HTML document.

3. The `<head>` section contains metadata about the document, such as the character encoding, title, description, and viewport settings.

4. The `<link>` tag includes an external CSS file (`main.css`) to style the page.

5. The `<body>` element contains the main content of the web page.

6. The `<nav>` element defines a navigation bar with a logo, brand text, and a toggler button.

7. The `<header>` section includes a header with a title, description, a button to create a resume, and an image.

8. The page includes a `<footer>` element with a text placeholder.

9. Various classes are used to style the elements, such as `bg-white`, `bg-bright`, and `bg-dark` for background colors, text styles like `text-uppercase`, `text-center`, `fs-15` for font sizes, and positioning styles like `fw-6` for font weight.

Overall, this HTML code creates a basic web page structure with a navigation bar, header section, and footer. It uses classes for styling elements and includes placeholders for various content like text and images.

RESUME.HTML

This is an HTML document that represents a resume-building web page. Here are some key points about the code:

1. The document structure: The document starts with the `<!DOCTYPE html>` declaration, followed by the `<html>`, `<head>`, and `<body>` elements.

2. Metadata: In the `<head>` section, metadata like character set, viewport settings, and the page title are defined. Custom CSS is linked using the `<link>` tag.

3. Navigation Bar: The page contains a navigation bar (`<nav>`) with a logo and a toggler button for mobile devices.

4. Resume Form: The main content of the page is a form (`<form>`) where users can input their personal information, achievements, experiences, education, projects, and skills. The form includes various input fields for different details like name, image, designation, address, email, phone number, summary, etc.

5. Repeater Functionality: The form sections for achievements, experience, education, projects, and skills include repeater functionality. Users can dynamically add multiple sections for each category by clicking on the "+" button and remove them using the "-" button.

6. Preview Section: There is a preview section (`<section id="preview-sc">`) where the entered information is displayed in a structured format for the user to review before printing.

7. Print Button: At the end of the page, there is a print button (`<button>`) that triggers a JavaScript function (`printCV()`) to print the resume.

8. Script tags: The page includes script tags to load jQuery, jQuery repeater, custom JavaScript files (`assets/js/script.js` and `assets/js/app.js`) that handle the form interactions and functionalities.

Overall, this code creates a web page where users can input their resume details, preview the formatted resume, and print it.

JAVASCRIPT

This code contains functions for validating form data inputs, fetching user inputs from form elements, displaying user data in a CV preview, generating a CV, previewing an image, and printing a CV.

1. The code defines regular expressions for validating strings, emails, phone numbers, and digits.

2. It sets up constants for different input types and user input elements.

3. It defines a function `fetchValues` to gather data from multiple input fields into an array of objects.

4. The `getUserInputs` function fetches user inputs from various form elements and sets up event listeners for form validation.

5. The `validateFormData` function validates form inputs based on the type of input field (text, email, phone number, or any).

6. The `addErrMsg` function adds error messages for invalid form fields.

7. The `removeErrMsg` function clears error messages for valid form fields.

8. The `showListData` function displays list data in a specified container by creating HTML elements dynamically.

9. The `displayCV` function populates the CV preview with user data.

10. The `generateCV` function collects user inputs, displays the CV preview, and logs the user data to the console.

11. The `previewImage` function reads and displays the selected image file for preview.

12. The `printCV` function triggers the browser's print functionality to print the CV.

Overall, the code helps in creating a CV form with validation and dynamically updating a preview based on the user input.