Front-end Basics

1. Understand and Apply Basic JavaScript Programming

Objective: The student demonstrates proficiency in using JavaScript, including variables, data types, functions, loops, and conditional statements.

Criteria	Poor - 0	Fair - 1	Good - 2	Excellent - 3
Variables and Data Types	Incorrect use or misunderstanding of variables and data types.	Basic use, but frequent errors or confusion with data types.	Mostly correct use, with minor mistakes in variable declaration or type handling.	Proficient use of variables and correct handling of multiple data types with no errors.
Functions	Functions are incorrect or missing.	Functions work but lack clarity or appropriate structure.	Functions are mostly correct with minor inefficiencies.	Functions are well-structured, reusable, and adhere to JavaScript best practices.
Loops	Loops are missing or incorrectly implemented.	Basic loop structures with notable inefficiencies or errors.	Correct use of loops, with occasional logic errors.	Efficient and correct use of loops with no errors.
Conditionals	Conditionals are missing or incorrectly used.	Inconsistent or unclear use of conditionals.	Mostly correct use of conditionals, with minor logic issues.	Proper use of conditionals with clear and error-free logic.

2. Develop Responsive Web Pages Using HTML and CSS

Objective: The student can build responsive web pages using semantic HTML and advanced CSS techniques for responsive design.

Criteria	Poor - 0	Fair - 1	Good - 2	Excellent - 3
HTML Structure	HTML is disorganised,	Basic HTML structure, with notable	Mostly correct HTML structure, with minor	Well-organized, fully semantic HTML with a

	non-semantic, or incorrect.	semantic or organizational issues.	semantic or organizational issues.	clear and logical structure.
Page Layout and Flow	The layout is non-responsive or breaks significantly on different screen sizes.	The layout is mostly functional, but has notable responsiveness or flow issues on some devices.	The layout is responsive on most devices, with minor issues in visual flow or spacing.	The page layout adapts fluidly across all devices, with smooth flow and spacing across screen sizes.
Navigation Design	Navigation is non-functional or breaks on smaller screens.	Navigation is functional, but not optimized for responsiveness.	Navigation works across devices, but with minor layout or usability issues.	Navigation is fully responsive, user-friendly, and adapts perfectly across all screen sizes.
Typography and Readability	Text size and spacing do not adapt for readability on different screens.	Text is somewhat readable, but with issues in font size or spacing on certain devices.	Text is mostly readable, with minor issues in sizing or spacing on some devices.	Text is fully responsive and consistently readable on all devices, with well-optimized font sizes and spacing.
Accessibility	No consideration for accessibility (e.g., missing alt-tags, poor colour contrast).	Basic accessibility features are present, but many best practices are missing.	Most accessibility features are implemented, with minor improvements needed (e.g., colour contrast, screen reader compatibility).	Fully accessible design, following best practices for alt text, keyboard navigation, colour contrast, and ARIA attributes.

3. Integrate JavaScript with External Data Sources

Objective: The student can work with JavaScript to handle objects, arrays, and JSON data, as well as interact with external APIs.

Criteria	Poor - 0	Fair - 1	Good - 2	Excellent - 3

Objects and Arrays	Objects and arrays are incorrectly used or missing.	Basic use of objects and arrays, but with frequent errors.	Mostly correct use of objects and arrays, with occasional logic mistakes.	Efficient and correct use of objects and arrays with no errors.
JSON Handling	Incorrect or incomplete handling of JSON data.	Basic understanding of JSON, but frequent errors in parsing or using the data.	Mostly correct use of JSON, with minor parsing or logic errors.	Proficient handling of JSON data with no parsing or usage errors.
API Interaction	Unable to correctly interact with external APIs.	Basic API calls made, but with errors or missing data handling.	Correct interaction with APIs, but with minor logic or error-handling issues.	Efficient interaction with APIs, including proper error handling and data usage.
Asynchronous Programming	No use or incorrect use of asynchronous JavaScript.	Basic use of asynchronous JavaScript with notable issues.	Mostly correct use of asynchronous JavaScript with minor issues.	Proficient use of asynchronous JavaScript, ensuring smooth interaction with APIs.

4. Build Responsive and Mobile-Friendly Web Pages with Bootstrap

Objective: The student can use Bootstrap's grid system, components, and utilities to build responsive and mobile-friendly web pages.

Criteria	Poor - 0	Fair - 1	Good - 2	Excellent - 3
Grid System	No use or incorrect use of Bootstrap’s grid system.	Basic use of the grid system, but with notable issues in layout.	Mostly correct use of the grid system, with minor layout issues.	Efficient and correct use of the grid system, ensuring responsive layouts across devices.
Bootstrap Components	No use or incorrect implementation	Basic use of components, but with issues	Mostly correct use of components	Proper and efficient use of Bootstrap

	of Bootstrap components.	in functionality or layout.	with minor styling or functionality issues.	components, ensuring consistent styling and functionality.
Mobile-First Design	Web pages are not mobile-friendly or break on mobile devices.	Limited mobile responsiveness; layout issues on mobile devices.	Mostly mobile-friendly, with minor issues on some devices.	Fully mobile-friendly design, adapting seamlessly to all screen sizes.
Utility Classes	Bootstrap utility classes are not used or incorrectly applied.	Basic use of utility classes, but with inconsistencies.	Mostly correct use of utility classes, with minor inconsistencies.	Efficient use of utility classes, ensuring clean and responsive design.