# Course Outcomes

1. Use JavaScript basic syntax, object-oriented programming, functions, variables, events, arrays, validation, accessibility, and web standards.
2. Write, debug, and maintain well-formed, well-documented interactive web content using JavaScript code.
3. Validate HTML forms using JavaScript.

## **Exercise Description**

This exercise assesses a learner's knowledge of JavaScript basics, object-oriented programming, functions, variables, events, arrays, validation, accessibility, and web standards. The learner will create an interactive web form for Byte Forge Academy, validate user inputs, and provide feedback using JavaScript.

## Scenario

Byte Forge Academy is a rapidly growing online coding school offering programming courses for beginners and experienced developers. They aim to provide a seamless and user-friendly experience for students who want to enroll in their courses.

## Provided Assets

* Byte Forge Logo (2)
* Home page Image (1)
* Social media icons (2)
* ReadMe.txt file containing HEX/RGB color codes and brand information

## Grading

|  |  |
| --- | --- |
| Category |  |
| HTML Structure | 15 points |
| Javascript Code | 60 points |
| CSS Styling | 15 points |
| Code Quality and Documentation | 10 points |

|  |  |
| --- | --- |
| Scale |  |
| Novice | 0 - 79 points  Learner has limited understanding of exercise requirements and significant issues in the code, documentation, accessibility, and design. The final product does not meet expected standards. |
| Proficient | 80 - 90 points  Learner demonstrates good understanding of exercise requirements and has only minor issues in code, documentation, accessibility, and design. The final product demonstrates proficiency of expected standards. |
| Exemplary | 90 – 100 points  Learner demonstrates excellent understanding of the exercise requirements and has no issues in the code, documentation, accessibility, and design. The final product fully meets or exceeds the expected standards. |

80% is required to pass this exercise. More information is found within the grading rubric.

## Exercise Instructions

### HTML Structure (15 points)

1. Create an HTML file with a basic structure, including a doctype declaration, HTML, head, and body tags. Add a title and a descriptive heading for your form.
2. Add the Byte Forge Academy logo (provided) at the top of the HTML file.
3. Add the company tagline and call to action – styling and placement is developers choice
4. Tagline: Unlock Your Coding Potential
5. Call to Action: Register today to transform your future!
6. Inside the body tag, include create a simple form with the following fields:
7. Full Name (text input, required)
8. Email Address (email input, required)
9. Password (password input, required, minimum 8 characters)
10. Confirm Password (password input, required, should match password)
11. Age (number input, required, minimum 18 years)
12. Favorite Programming Language (select input with at least 4 options)
13. A "Submit" button
14. Add the image provided next to the form.
15. Add a footer at the bottom of the HTML file containing:
16. ByteForge Academy copyright information
17. Social media icons (provided) that link to social media sites (i.e. Twitter links to [www.twitter.com](http://www.twitter.com) and Facebook links to www. Facebook.com)
18. Add comments throughout your HTML code to explain each section's purpose and functionality.

### Javascript Code (60 points)

1. Use JavaScript to perform the following tasks:
2. Create an object constructor named "User" with properties for full name, email address, password, age, and favorite programming language.
3. Write a function named "validateForm" that takes the form as an input and checks if the form fields meet the requirements stated in step 3. If a field does not meet the requirements, display an error message next to the input field.
4. In the validateForm function, use an array to store error messages for each field. If the form is valid, create a new instance of the User object with the provided information and display a success message on the screen.
5. Add an event listener to the form's submit event that calls the validateForm function, and use the "preventDefault" method to stop the form from submitting if it is not valid.
6. Add comments throughout your JavaScript code to explain each section's purpose and functionality.

### CSS Styling (15 points)

1. Style your form using CSS to make it visually appealing, easy to use, and follow web standards. Your design should be responsive and accessible to all users. At minimum, CSS styling should include:
2. Changing the font family and size
3. Adding background color and border to form fields
4. Styling the submit button
5. Changing the colors of the error messages
6. Making the form responsive using media queries
7. Add comments throughout your CSS code to explain each section's purpose and functionality.
8. Make sure your form is accessible and follows web standards:
9. Add appropriate labels to form fields.
10. Use semantic HTML elements.
11. Ensure proper keyboard navigation and focus order.

### Code Quality and Documentation (10 points)

1. Test your form by entering different inputs, checking if the error messages display correctly, and if the form submits when all fields are valid.

### Submission

Provide the HTML, CSS, and JavaScript files you created during this exercise. Ensure your code is well-formatted, well-documented, and free of syntax errors or bugs. Make sure to include the logo, footer, social icons, and image as part of your submission.

## Grading Rubric for Byte Forge Academy Registration Form Exercise (100 points)

1. **HTML Structure (15 points)**
   * Basic structure (doctype, html, head, body) (3 points)
   * Title and heading (2 points)
   * Form with all required fields (7 points)
   * Logo, footer, and image properly added (3 points)
2. **JavaScript Code (60 points)**
   * User object constructor (5 points)
   * validateForm function (10 points)
   * Properly getting form elements (5 points)
   * Error messages array (4 points)
   * Validation for each field (20 points)
     + Full Name (3 points)
     + Email (3 points)
     + Password (3 points)
     + Confirm Password (3 points)
     + Age (3 points)
     + Favorite Programming Language (5 points)
   * Displaying error messages or creating a new User instance (6 points)
   * Event listener for the form's submit event (5 points)
   * Comments throughout JavaScript code (2 points)
3. **CSS Styling (15 points)**
   * Visual appeal and consistency (5 points)
   * Responsiveness (5 points)
   * Accessibility (3 points)
   * Web standards compliance (2 points)
4. **Code Quality and Documentation (10 points)**
   * Well-formatted and well-documented code (6 points)
   * No syntax errors or bugs (4 points)

**Total: 100 points**

### Scale:

1. **Novice (0-79 points):** The learner has limited understanding of the exercise requirements and has significant issues in the code, documentation, accessibility, and design. The final product does not meet the expected standards.
2. **Proficient (60-89 points):** The learner demonstrates a good understanding of the exercise requirements and has only minor issues in the code, documentation, accessibility, and design. The final product demonstrates proficiency of expected standards.
3. **Exemplary (90-100 points):** The learner demonstrates an excellent understanding of the exercise requirements and has no issues in the code, documentation, accessibility, and design. The final product fully meets or exceeds the expected standards.