# 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**

The Weather Quiz App exercise assesses a student's proficiency in web development, focusing on JavaScript, HTML, and CSS. In this project, students create an app that fetches weather data for a specified city and generates a quiz based on the data. The exercise tests their ability to work with APIs, handle user input, manage application state, create custom objects, and design a responsive user interface. It concisely evaluates JavaScript development skills while promoting creativity and problem-solving.

## Scenario

Build a Weather Quiz App

(Functions & Events, Data Types, Arrays, DOM, Forms, String & Array Manipulation, Custom Objects, State Management, Event Objects, Ajax/Fetch, Third-party APIs)

Students will build a weather quiz app that retrieves data from a third-party weather API, such as OpenWeatherMap, and presents 5 multiple-choice questions based on the fetched data. The app should have the following features:

1. A form that allows users to input a city name or ZIP code.
2. Use event listeners to handle form submissions and fetch weather data from the API.
3. Display a set of multiple-choice questions based on the fetched weather data (e.g., current temperature, humidity, wind speed, etc.). Use arrays to store the questions, answer choices, and the correct answers.
4. Use custom objects to represent each question, with properties like questionText, answerChoices, and correctAnswer, and a method to check if the user's answer is correct.
5. Allow users to submit their answers through a form with radio buttons for each answer choice. Use event listeners to handle form submissions and check the user's selected answer against the correct answer.
6. Keep track of the user's score and display it on the page once they have completed the quiz.
7. Store the user's last search query and score in the browser's local storage so that the app can display their last score when the user revisits the app.
8. Integrate a mapping API, such as Google Maps or Mapbox, to display a map of the user's chosen location.

By combining elements of the other challenges into the Weather App challenge, students will have the opportunity to demonstrate their proficiency in a wide range of JavaScript concepts and skills within a 1-2 hour time frame.

## Provided Assets

* Challenge Instructions
* API Key
* Grading Rubric
* Cheat Sheet
* Video example

## Grading

|  |  |
| --- | --- |
| Weighted Category |  |
| Functionality | 40 points |
| Code Quality | 30 points |
| User Interface & Design | 20 points |
| Extra Features & Creativity | 10 points |

|  |  |
| --- | --- |
| Scale |  |
| Novice | 0 - 79 points  The learner has a limited understanding of exercise requirements and significant issues in the code, documentation, accessibility, and design. The final product does not meet the expected standards. |
| Proficient | 80 - 90 points  The learner understands exercise requirements well and has only minor issues in code, documentation, accessibility, and design. The final product demonstrates proficiency in the expected standards. |
| Exemplary | 90 – 100 points  The learner understands the exercise requirements and has no issues with 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.

## Rubric

|  |  |  |  |
| --- | --- | --- | --- |
| Category | Novice (0-26) | Proficient (27-34) | Exemplary (35-40) |
| Functionality (40) | The app is partially functional or has significant bugs, making it difficult to use. It may not fetch weather data or generate quiz questions correctly. | The app is mostly functional and fetches weather data, generate quiz questions, and handles user input as expected, with minor bugs or issues. | The app is fully functional, free of bugs, and provides a smooth user experience. It fetches weather data, generates quiz questions, and handles user input flawlessly. |
| Code Quality (30) | **Novice (0-19)** | **Proficient (20-25)** | **Exemplary (26-30)** |
|  | The code is disorganized, difficult to read, or lacks proper formatting. It may not follow best practices, and there may be a need for separation of concerns or use of functions/classes. | The code is mostly organized and follows best practices. It uses functions/classes to separate concerns, and a few areas could be improved. | The code is clean, organized, and easy to read. It follows best practices, effectively separates concerns, and demonstrates a thorough understanding of JavaScript concepts. |
| UI & Design (20) | **Novice (0-12)** | **Proficient (13-16)** | **Exemplary (17-20)** |
|  | The user interface is basic or lacks visual appeal. There may be usability issues or unclear elements that make it difficult for users to interact with the app. | The user interface is clear, visually appealing, and easy to use. There may be some areas that could be improved or polished. | The user interface is visually appealing, intuitive, and user-friendly. It demonstrates an excellent understanding of design principles and provides a seamless user experience. |
| Extra Features & Creativity (10) | Novice (0-6) | Proficient (7-8) | Exemplary (9-10) |
|  | The app implements the basic requirements but lacks extra features or creativity. There may be limited or no attempt to extend the functionality beyond the given challenge. | The app includes some extra features or creative elements that enhance the user experience or demonstrate a deeper understanding of the challenge concepts. | The app demonstrates a high level of creativity, with additional features or innovative ideas that **significantly enhance** the user experience and demonstrate a mastery of the challenge concepts. |

### A student must score at least 80 points overall to achieve a Proficient grade.

## Exercise Instructions