Challenge No.7 - Specification

Objects in JavaScript

Description

After the presentation of the challenge and receiving the starter files, each student is required to complete the assignment within the given deadline. The solution should be uploaded to GitLab, with the assessor added as a maintainer to the repository. Additionally, the student is instructed to add the solution link to the learning platform.

Create a simple weather dashboard application to display weather data from the database (array of objects). The application will allow the users to search for cities, display current weather conditions, a 5-day forecast and store recent searches in the local storage. Complete this challenge using JavaScript objects and classes. Refer to the screenshots and the dmeo video for the visual layout of the application.

Level 1: Beginner

Student should understand the basics of JavaScript syntax and data types.
Student should be familiar with JavaScript objects and arrays to store and manipulate data.
Student should have basic understanding of classes in JavaScript, including constructor functions and the this keyword.
Student should have the ability to write simple functions and understand function execution flow.
Student should know basic DOM manipulation: querying DOM elements (document.getElementById, document.querySelector) and modifying their content (element.innerHTML).
Student should have basic event handling: attaching event listeners to DOM elements (element.addEventListener) to respond to user actions like clicks.
Student should have understanding of conditionals (if-else statements) to handle different outcomes, such as displaying an alert if a city is not found.

Level 2: Intermediate

Student should have proficiency in using classes and static methods in JavaScript to organize code logically and maintain state.
Student should have enhanced DOM manipulation skills, including appending elements dynamically and managing element classes and attributes.
Student should have the ability to implement more complex logic for data manipulation and display, such as simulating a 5-day forecast based on existing data.
Student should have understanding of loops (for loop) to iterate over data arrays.
Student should have basic knowledge of using browser local storage (localStorage) to save and retrieve data.

Level 3: Advanced

Student should have a deep understanding of JavaScript classes.
Student should have advanced DOM manipulation techniques, creating interactive and dynamic web pages that respond to user inputs and actions.
Student should have an understanding of localStorage to manage application state, including adding, removing, or updating stored data.

Exercise 01
Create a function to get from the database the weather reports and implement a search functionality to display the current weather data for the searched city.

1. Use the JavaScript object *weatherData* which has the following information: city, temperature, humidity, windSpeed.
2. Create a function *fetchWeather* to display weather data based on the search input
3. Create a function *displayCurrentWeather* to display in the DOM the current weather details

Exercise 02
Create a function to get the latest 5-day forecast for the city that the user searched. There is no data for 5-day forecasts for a given city, so in this case you can simulate a 5-day forecast by getting the temperature for a given city from the current database, and just add 1 more in a loop. For example, if current temperature in Berlin is 14°C, you can simulate a 5-day forecast by showing 15°C, 16°C, 17°C, 18°C, 19°C for each upcoming day.

1. Create a function *fetchForecast* to display a 5 day weather forecast data based on the search input.
2. Create a function *displayForecast* to display in the DOM the current weather forecast.

Exercise 03
Save recent searches in the local storage and show them in the DOM under the weather forecast.

1. Create a separate custom function *saveRecentSearch* to save searches in local storage.
2. Create a separate custom function *displayRecentSearches* to get the data from local storage and display them correctly in the DOM.

Exercise 04: BONUS
Make the recent searches list items clickable so that it allows the users to re-run a search by clicking on a city name from this recent searches list.

Evaluation system

Criteria	Excellent 2.5p	Proficient 2.0p	Good 1.5p	Fair 1.0p	Poor 0.5p
Solution Correctness	*The solution is flawless, meeting all requirements and*	*The solution is correct for the most part, with minor errors that do not*	*The solution is mostly correct, but there are some major errors affecting*	*The solution has several major errors, making it partially functional, but*	*The solution is incorrect, incomplete and lacks understanding of*

	functionalities.	*significantly impact the functionality.*	*the overall functionality.*	*shows a basic understanding of the problem.*	*the problem.*
Code Quality	*Code is exceptionally well-organized, follows best practices, and demonstrates a deep understanding of all required concepts.*	*Code is mostly clear and well-organized and follows good practices, with only minor improvements needed.*	*Code is mostly organized, but there are notable areas that could be improved for better readability and maintainability.*	*Code lacks organization, readability, and structure, and could be significantly improved.*	*Code is messy, unreadable, poorly organized, and does not adhere to coding standards.*
Adherence to Specifications	*The solution precisely follows all specified challenge requirements.*	*The solution mostly adheres to the specifications, with only minor deviations or oversights.*	*The solution deviates from specifications in some aspects but still fulfills the main requirements.*	*The solution deviates significantly from the specifications, but some elements are implemented correctly.*	*The solution disregards most or all of the specified requirements.*
Documentation, Comments, Cleanliness	*The code is well-documented, clear and exceptionally clean. Comments explain the logic behind any complex parts.*	*Good documentation and comments. Code is generally clean with a few areas for improvement.*	*Basic documentation and comments, with room for improvement in clarity. Code cleanliness is acceptable but needs improvement.*	*Limited documentation and comments. Code lacks clarity and cleanliness and is not well-organized.*	*No documentation or comments. Code is disorganized and challenging to understand.*

Deadline

1 week after its presentation, at 23:59 (end of the day).

Assessment Rules

- Fair Assessment: ethical considerations
  - Assessors should ensure that the assessments are conducted in a fair and ethical manner, respecting the principles of academic integrity and honesty.
- Reliability and Validity: enabling consistency
  - Assessments should be consistent and reliable, meaning that they yield consistent results when applied repeatedly to the same task or performance.
  - Assessments must accurately gauge the knowledge, skills, or abilities they are intended to evaluate, ensuring their validity as indicators.
- Feedback: as a method for continuous improvement
  - Assessors should offer constructive feedback that identifies strengths and areas for improvement. The Feedback should be specific and actionable, it should include thought provoking guides and should challenge the student to become better at a specific task.

- Late Submission Policy: assessing assignments after their deadline
  - Students should be allowed a grace period of 3 days (72 hours) to make a late submission on any assignment, with the notice that they will be deducted 20% from the total possible points.
- Plagiarism Policy: assessing assignments with matching solutions
  - In the event of suspected plagiarism, the assessor is required to promptly collaborate with the Student Experience Coordinator/Team as the initial step. Together, they will draft a notice to remind students of the strict prohibition against plagiarism, with potential repercussions for recurrent violations. The following actions will be considered in cases of repeated plagiarism:
    - If a submitted solution exhibits substantial similarity, exceeding 60%, with another student's work (individually or within a group), the respective challenge will incur a 50% reduction from the maximum points attainable.
    - In cases where a solution is identified as more than 90% identical to another student's work (individually or within a group), the challenge in question will receive a score of 0 points.
  - The use of generative AI is encouraged as a learning tool in our educational programs; however, students must engage with the material and contribute with original thought. Reliance on AI for complete content generation is strictly prohibited and will result in point deductions, official warning, or other academic penalties.
  - Upon completion of the assessment process for each challenge/project, the assessor is tasked with selecting the most complete, optimal, and creative solution and to showcase it by publishing it on the platform together with the assessment results
- Timeliness: timeframe for delivering results and feedback to students
  - Feedback on challenges should be provided within 7 days after the deadline has passed.