TPO

Class 5
Full-time studies

If you're not familiar with HTML and CSS, check out these tutorials first: `https://www.w3schools.com/html/` and `https://www.w3schools.com/css/`

I. (3 points) Create a project with a controller that handles `GET` requests only and returns information related to time. Implement a `TimeController` that exposes the following endpoints:

- `GET /current-time` - after sending a `GET` request without parameters to this address, we should see the current time relative to the default time zone displayed in the format `hh:mm:ss.SSSS YYYY/MM/DD` (e.g. `14:32:08.1234 2025/03/31`). The endpoint should also support the following optional query parameters:
  - `timezone` - allows the user to specify a time zone (e.g. `UTC`, `Europe/Warsaw`, `AST`, `GMT` etc.). The response should reflect the current time in the specified zone.
  - `format` - allows the user to specify a custom format for displaying time (e.g. `DD-MM-YYYY hh:mm`). If the format is invalid or unsupported, fall back to the default format and include a warning message.
- `GET /current-year` - returns the current year as a four-digit number (e.g. `2025`).

In case of any exceptions, display the default time relative to the default time zone, along with an appropriate informative error message indicating what went wrong (e.g., `"Invalid time zone provided. Defaulting to system time zone."`)

Prepare an aesthetically pleasing appearance for the server response (formatted in a clean, readable, and visually pleasing way).

II. (3 points) Add a second controller to the previous task, that handles `POST` requests and performs number base conversions using user input from a form. Retrieve from the user the following data:

- `value` - the numeric value to convert
- `fromBase` - information about the number system in which the value is stored (integer from 2 to 100)
- `toBase` - information about the number system to which we want to convert this value (integer from 2 to 100)

Use only characters from the basic ASCII table to represent digits and letters in higher numeral systems (e.g., 0–9, A–Z, and other allowed characters for bases above 36).

Also, for each conversion result, show additionally how this value is represented in the BIN, OCT, DEC, and HEX systems.

Display conversion results in a readable manner. Prepare an aesthetically pleasing appearance for the server response.

Validate the inputs (ensure the value is valid for the provided `fromBase`, and that bases are within the valid range).

In case of any exceptions, display the appropriate informative error message indicating what went wrong (e.g., `"Invalid numeric value for the given base"`, `"fromBase outside the allowed range"` etc.). Return only a clear explanation and do not perform the conversion.

Display all conversion results in a well-structured and readable layout, ensure all responses are aesthetically pleasing and user-friendly.

Send the solutions as a one ZIP archive called *sXXXXX-TPO05.zip*, where *sXXXXX* is your index number.

Send a short (2 minutes) video presentation of your solution (how it works) as a MP4 file called *sXXXXX-TPO05.mp4*, where *sXXXXX* is your index number.