Question 5

* The process of making API requests in JavaScript:
  + Step 1: Make an API call.
    - Using fetch() method to fetch resources from a server:
    - Using HTTP libraries like jQuery, AJAX,…
  + Step 2: Build the request:
    - Determine the URL of API endpoint.
    - Choose HTTP Method
    - Declare headers such as authorization or content-type (application/json, text/plain, …)
  + Step 3: Send request: Handle asynchronous behavior appropriately using await, or callbacks, ...
  + Step 4: Handle response:
    - Once the server processes the request, it sends back a response.
    - Parse the response data.
    - Check the status code of response.
    - Handle the response data based on the status code
* Detailing key concepts:
  + Asynchronous Programming: JavaScript is single-threaded, so async allows you to perform operations without blocking other tasks. Using await, callbacks, …
  + HTTP methods:
    - GET: Retrieves data from the server.
    - POST: Sends data to the server to create a new resource.
    - PUT: Updates an existing resource on the server.
    - DELETE: deletes the specified resource from the server
    - Other: HEAD, CONNECT, …
  + Status Code:
    - 2xx: Success ( 200 OK).
    - 3xx: Redirection (301 Moved Permanently).
    - 4xx: Client errors ( 404 Not Found).
    - 5xx: Server errors (500 Internal Server Error)
  + Handling Responses:
    - After sending an API request, you receive a response from the server.
    - You need to handle the response data based on the format (e.g., JSON, XML) and status code.
    - For successful responses, you may extract and process the data. For errors, you might handle them gracefully or display error messages to the user.
* Frameworks simplify API interactions by providing abstractions, utilities, and conventions that streamline common tasks such as making HTTP requests, handling responses, and managing state, reducing the complexity of writing API-related code.
* To effectively use frameworks for API data retrieval:
  + Choose a suitable framework aligned with project needs and language.
  + Understand the framework's HTTP request, response handling, and data management APIs.
  + Utilize built-in utilities and abstractions for streamlined HTTP requests, response parsing, and error handling.