Assignment 2.A (3%) - Single Widget JavaScript Functionality

T-213-VEFF, Web programming I, 2026-1

Reykjavik University - Department of Computer Science, Menntavegi 1, IS-101 Reykjavík, Iceland

Deadline: Friday, February 20th 2026, 23:59

This is Part A of the second assignment in Web Programming I. In this part, you will add JavaScript functionality to one widget on the provided Student Dashboard by connecting it to a provided API.

This assignment has to be done individually, no group work is permitted.

1. Daily Quote Widget

In Part A, you will implement the functionality of the *Daily quote* widget. You are not expected to implement the task list or quick notes widgets yet, this will be done in Part B.

Your task is to write JavaScript code that retrieves a quote from an API based on a selected category, displays the quote text and author in the dashboard, and updates the quote when the category changes or when the user clicks the "New quote" button.

The focus of this part is correct communication with an API and correct manipulation of the DOM using JavaScript. Visual styling and layout are **not** part of this assignment.

You may need to look up JavaScript features that have not yet been covered in the lectures. Helpful resources include developer.mozilla.org.

Starter Pack

You are provided with a starter project (assignment_2_starterPack) containing the following files:

- `index.html`
- `style.css`
- `app.js`
- `oops.jpg`
- `man_3728711.png`
- `package.json`

All implementation must be done in `app.js` The HTML and CSS files are provided and must **not** be modified.

The `app.js` file already contains a predefined structure with function declarations and exports. This structure is required for the autograder and must not be modified. In particular:

- Do not rename or remove any existing functions
- Do not remove the `export` statement
- Do not remove the `init()` call at the bottom of the file

The project is configured as a JavaScript module. This is necessary for the autograder and requires the project to be run using the included Vite development server:

- Run `npm install`
- Run `npm run dev`
- Open the local development URL shown in the terminal

Do not open the `index.html` file directly in the browser. Since the project uses JavaScript modules, it must be served through the development server.

2. Requirements

The following requirements must be fulfilled in your solution. Failure to meet certain requirements may result in severe point deduction or a score of 0. Therefore, make sure to follow them closely and check the grading criterias in this file before your submission.

1. **Required files and file names**
   (a) The HTML file must be named `index.html`.
   (b) The JavaScript file must be named `app.js`.
   (c) The CSS file must be named `style.css`.
   (d) The image files must be named `oops.jpg` and `man_3728711.png`.
   (e) All files must be located in the root of the project (no subfolders).
   (f) `package.json` must be included.
   (g) Submissions containing incorrectly named required files will receive a score of 0.
   (h) Submissions containing additional files will receive a score of 0.
2. **HTML and JavaScript separation**
   (a) All JavaScript code must be placed in `app.js`.
   (b) No inline JavaScript is allowed.
3. **Restrictions on provided files**
   (a) The provided `style.css` file must not be modified.
   (b) The provided `index.html` file must not be modified.
4. **Daily quote functionality**
   (a) A quote must be retrieved from the provided API endpoint for quotes.
   (b) The selected quote category must be sent to the API using a query parameter.
   (c) The retrieved quote text must be displayed in the `blockquote` element.
   (d) The retrieved author must be displayed in the `figcaption` element.
   (e) A new quote must be fetched from the API when the user changes the category.
   (f) A new quote must be fetched from the API when the user clicks the "New quote" button.
5. **API communication**
   (a) The quote must be fetched from the endpoint:
   `https://veff-2026-quotes.netlify.app/api/v1/quotes`
   (b) The selected category must be sent as a query parameter named `category`.
   (c) Example request:
   `https://veff-2026-quotes.netlify.app/api/v1/quotes?category=study`
   (d) The request must be made using **Axios**.
   (e) Asynchronous code for the API request must be written using `async/await`.

6. **JavaScript usage**
   (a) You must use `const` and `let` instead of `var`.
   (b) You must use arrow functions when defining functions.
   (c) Asynchronous code must be written using `async/await`.

3. Submission

The project is submitted via Gradescope. Submit the following files:

- index.html
- app.js
- style.css
- oops.jpg
- man_3728711.png
- package.json

Submissions will not be accepted after the deadline.

4. Grading and point deduction

Assignment 2.A is graded fully automatically using Gradescope. The total score for Assignment 2.A is **30 points**. Submissions and the autograder will open on Wednesday, February 18th 2026. After each submission, you will receive a grade and detailed feedback from Gradescope. You are encouraged to use this feedback to improve your solution and may submit as many times as needed before the deadline.

Certain violations result in a score of **0 points** for this part, in which case grading stops immediately.

Criteria	Point deduction
Required files and file names: All required files are present and correctly named.	Missing or incorrectly named required files: **-30 points (score reduced to 0; grading stops)**.
No extra files: Only the expected files are submitted.	Any extra/unexpected file(s) included: **-30 points (score reduced to 0; grading stops)**.
Restrictions on provided files: The CSS file must remain completely unchanged. The HTML file must remain unchanged, except for adding a <script> reference.	Any modification to the CSS file, or any change to the HTML file other than adding the script reference: **-30 points (score reduced to 0; grading stops)**.
Daily quote API usage (10 points): The quote is retrieved from the provided API using the selected category.	Missing or incorrect API call: up to **-10 points**, depending on severity.
Quote rendering (8 points): The retrieved quote text and author are displayed in the correct elements.	Quote text or author missing or incorrect: up to **-8 points**.
Quote update behavior (6 points): The quote updates when the category is changed and when the "New quote" button is clicked.	Missing or incorrect update behavior: up to **-6 points**.
JavaScript best practices (6 points): Use of `const`/`let`, arrow functions, and `async/await`.	Significant deviation from required practices: up to **-6 points**.