WEB APPLICATION DEVELOPMENT USING NODEJS - 502070

LAB SESSION 2

By Mai Van Manh

OBJECTIVES

1. Install NodeJS and become familiar with the npm module management tool.
2. Create a web server and develop a simple web application using NodeJS.
3. Explore fundamental concepts in NodeJS, such as REPL, callbacks, event emitters, streams, and modules.
4. Handle requests and responses in NodeJS.
5. Manage files, process URLs, and query strings using built-in modules like fs, url, querystring, and path.
6. Reinforce knowledge related to the HTTP Protocol, including headers, status codes, and HTTP methods.
7. Handle basic HTML forms.

The exercises below require utilizing only the built-in modules within NodeJS, without the use of any additional modules installed from package managers (npm, yarn).

Exercise 1. Creating a Basic Website using the http Module in NodeJS

Utilize the http module to build a basic web page with simple calculations as illustrated below. When accessing http://localhost, the webpage will display an HTML form as shown in the image below. Upon clicking the "**Calculate**" button, the result will be displayed at http://localhost/result.

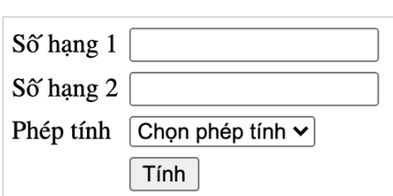


Image 1: Homepage interface displaying the form

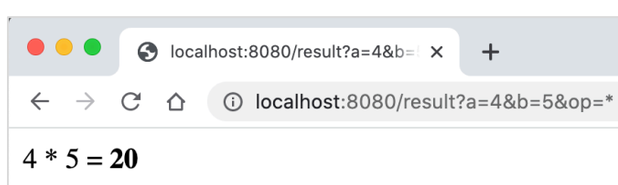


Image 2: the result when all required information is entered.

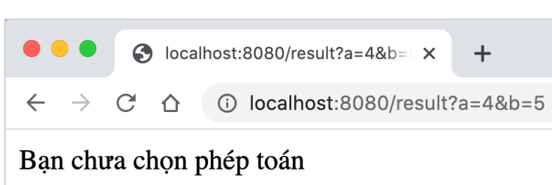


Image 3: displaying an error when information is missing

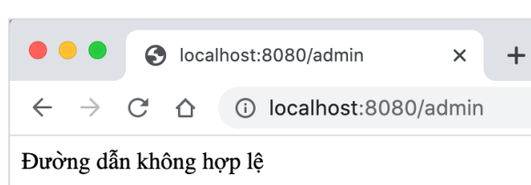


Image 4: Error notification when accessing an unsupported URL

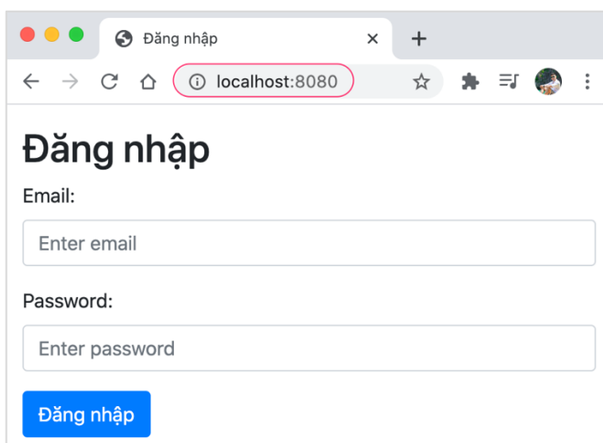
Requirements:

- The entire program should be written in a single file named main.js.
- Perform error checks if necessary and display appropriate error messages.

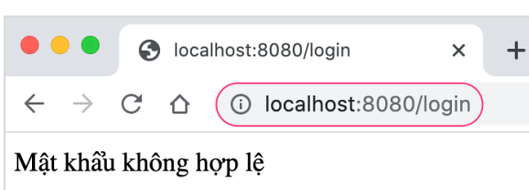
Exercise 2. Handling POST Requests using the http module

Complete an exercise similar to Exercise 1.

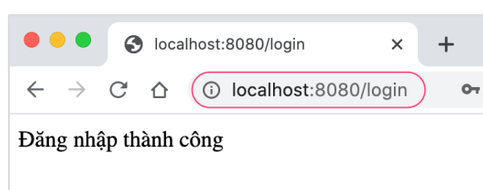
- http://localhost: Display a login form that accepts two pieces of information: email and password, then redirect to /login using the POST method.
- http://localhost/login: Receive the login information sent via the POST method, process the login, and display an error message if necessary. Upon successful login, display a 'login successful' message.
- Additional Requirements:
    - The content of the web pages should be retrieved from HTML files using the file reading mechanism of the fs module.



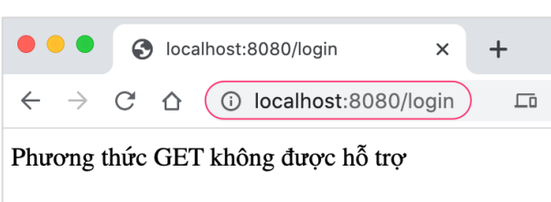
Homepage interface displaying the login form



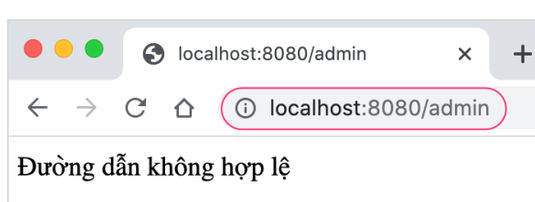
Error notification when the password is not entered



Success notification upon successful login



Error notification when accessing /login using GET



Error notification when accessing other URLs

Exercise 3. Create a Rest API with NodeJS that provides basic student information as follows.

- http://localhost/students:
  - GET Method: Returns a JSON Array of student lists.
  - POST Method: Adds a new student.
- http://localhost/students/{id}:
  - GET Method: Returns information about a specific student.
  - PUT Method: Updates information for a student.
  - DELETE Method: Deletes a student.
- Accessing other endpoints: Returns an error message in JSON format.

Note:

- Sample data is to be initialized by the student.
- Create the API and then use REST client tools to verify the API.