Tutorial 08: Express Framework and EJS template

NOTE: You may refer to sample codes shared on lecture notes for more details to create a new express project.

Task: EJS for Expense transactions

1. Create a new project directory named "**tutorial8express**" and change to the new directory.
   ```
   mkdir tutorial8express
   cd tutorial8express
   ```

2. Create new package.json file:
   ```
   npm init
   ```

3. Create a new main app.js file:
   ```
   fsutil file createnew app.js 0
   ```

   (**NOTE:** `app.js` file contains the gateway to your project, it contains the global project and express settings, it also has the code to connect to your database.)

4. Create a new directory called "**views**" inside your "**tutorial8express**" directory

5. Install express and EJS packages:

   ```
   $npm install express
   $npm install ejs
   ```

6. In your main `app.js` file, add a new route to `transaction.ejs`
   a. Define a route called "/transaction/:month" to response by rendering the `transaction.ejs` file.
   b. In the route, define an array of '**transactions**', and each expense transaction object will have 'date', type' and 'amount' properties
7. Create a new EJS file called "`transaction.ejs`" in the 'views' directory
   a. Inside the EJS file, write **HTML and JavaScript codes** to display the list of expense transactions with date, type and amount using loops
   b. The list should be displayed as a **HTML Table**. The table should contain at least 3 columns: date, type and amount. Each row represents a transaction object with date, type and amount.
   c. Add an "if-else statements" to evaluate the amount of transaction, if any transaction amount is more than 100, display a warning message under the HTML Table ***"Be careful with your expenses! You have spent <amount> in***

<type>”. Replace the <amount> and <type> with the data passed from the ‘**transactions**’ array.

8. Start and restart your web server using `$node app.js` command to test the routes that you defined in app.js file and test the codes that you write in the `transaction.ejs` file.
9. Create a new directory called ‘**public**’’.
10. In the ‘public’ directory, create a new stylesheet called ‘`app.css`’ in the ‘public directory to apply some styles in your `transaction.ejs` file.
11. Create a new directory called ‘**partials**’ inside the ‘view’ directory.
12. In the ‘partial’ directory, create two partials: `header.ejs` and `footer.ejs` to define common header and footer for all web pages in your application.
13. Include the partials in your `transaction.ejs` file.
14. Start and restart your web server using `$node app.js` command to test your styles and partials.