**Exercise 2: Using ES6**

# Title

Building a Todo List Application (Todo List)

# Object

Build a simple console application to manage todo lists using Node.js and ES6 features like classes, modules, arrow functions, promises or async/await, template literals, and destructuring.

# Requirement

## Application structure:

* Use **class** to construct a **TodoList** class and a **TodoItem** class.
* Use modules (ES6 Modules or CommonJS) to separate logic into different files (e.g. todoList.js, todoItem.js, and app.js).

Example:

TodoItem.js

class TodoItem {

    constructor(title) {

        this.title = title;

        this.completed = false;

    }

    complete() {

        this.completed = true;

    }

}

export default TodoItem;

TodoList.js

class TodoList {

    constructor() {

        this.items = [];

    }

    addItem(item) {

        this.items.push(item);

    }

    getItemTitles() {

        return this.items.map(item => item.title);

    }

    displayItemsWithStatus() {

        this.items.forEach((item, index) => {

            console.log(`${index + 1}. [${item.completed ? 'X' : ' '}] ${item.title}`);

        });

    }

    completeItem(index) {

        if (index >= 0 && index < this.items.length) {

            this.items[index].complete();

        }

    }

}

export default TodoList;

## Function:

* **Add new job**: User can enter the name of the new job and add it to the list.
* **Display job list**: Print to the console a list of all jobs, including status (completed/incomplete).
* **Mark tasks as completed**: Users can mark a task in the list as completed.
* Use **readline** or an external library like **inquirer** to interact with users via the console.

Use ES6 features like import/export, async/await to handle application logic:

app.js

import TodoList from './TodoList.js';

import TodoItem from './TodoItem.js';

import readline from 'readline';

const rl = readline.createInterface({

    input: process.stdin,

    output: process.stdout

});

const todoList = new TodoList();

const askQuestion = (query) => new Promise(resolve => rl.question(query, resolve));

async function addNewItem() {

    const title = await askQuestion("Enter the title of the new todo item: ");

    const item = new TodoItem(title);

    todoList.addItem(item);

    console.log("Added new item successfully.\n");

}

async function markItemCompleted() {

    const index = await askQuestion("Enter the index of the item you wish to mark as completed: ");

    const indexInt = parseInt(index, 10) - 1; // Adjust index because user input starts from 1

    if (indexInt >= 0 && indexInt < todoList.items.length) {

        todoList.completeItem(indexInt);

        console.log("Item marked as completed successfully.\n");

    } else {

        console.log("Invalid item index.\n");

    }

}

async function main() {

    let running = true;

    while (running) {

        console.log("1. Add a new todo item");

        console.log("2. Display all todo items with their status");

        console.log("3. Mark an item as completed");

        console.log("4. Exit");

        const choice = await askQuestion("Choose an action: ");

        switch (choice) {

            case '1':

                await addNewItem();

                break;

            case '2':

                todoList.displayItemsWithStatus();

                break;

            case '3':

                await markItemCompleted();

                break;

            case '4':

                running = false;

                console.log("Exiting...");

                break;

            default:

                console.log("Invalid choice, please try again.\n");

        }

    }

    rl.close();

}

main().catch(err => console.error(err));

**Note:** Convert app.js file and other modules to ES Module by adding "type": "module" to package.json.

Result:







