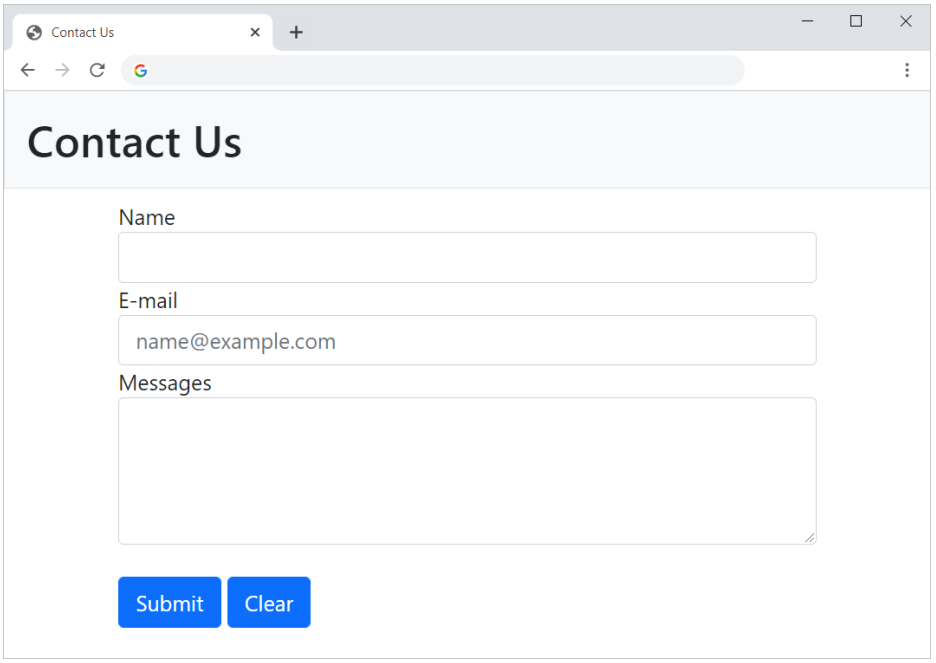
ITCS212 - Web Programming
Lab 7: Node.js Part II

MUST READ:

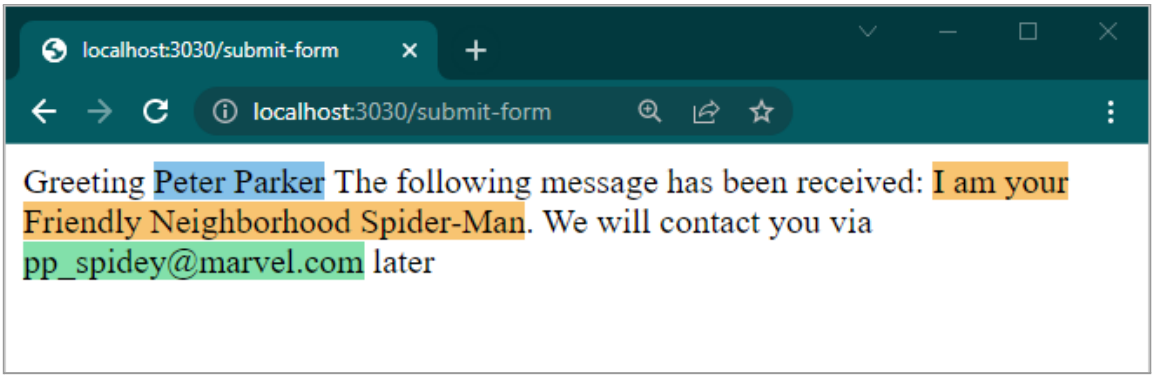
- There are 2 questions to be completed: Q1 and Q2
- Q1 is required to be submitted on MyCourses following the naming convention **by today's class time (Bangkok time)**. You must notify LAs **via the SOS** sheet to grade your work.
- The SOS sheet will be opened by the instructor or LA (approximately the beginning of the lab). The student who wants to submit or ask questions shall put their name on the list. Do not put your name for reservation ahead of time.
- The SOS sheet are here: [sec1][sec2][sec3]
- **No late submission allowed**

Q1: [To Submit] Form Handling: Your task is to implement form handling using Node.js Express Framework.

Given the following **`contact_us.html`** (available in MyCourse), as followed



The form takes all input information from users. **Note**: all information must be filled in before submission. After hitting "Submit", the information will be sent to Node.js server at "/submit-form" via **HTTP POST method**. The server then sends the response about the submitted information as follows.



Write the code to create a server running at the port 3030. The server must handle the following routes

- http:///localhost:3030/ → Show `contact_us.html` `(GET)`
- http:///localhost:3030/submit-form → Show result from the form `(POST)`

To start with Express, follow these steps to initialize your project:

1. use **`npm init`** to initialize your project. Note: set `L07-SecY-XX88XXX-q1.js` as the entry point (Note: `XXX` is your ID and `Y` is your section) and also put your name, your section and student ID as the author
2. Install `express` using **`npm install express --save`** and `nodemon` using **`npm install nodemon --save`**. **Note** ** Installing `nodemon` is optional. This is for your convenience only
3. In the `script` part in `package.json`, set up **"start"** to "**nodemon L07-SecY-XX88XXX-q1.js**". If you do not install **nodemon** in the previous step, set the value of "`start`" to "`node L07-SecY-XX88XXX-q1.js`" instead. **Note**: you can remove the "`test`" script.

After these steps, your `package. json` file and the directory should look like this.

```
{
  "name": "l07-q1",
  "version": "1.0.0",
  "description": "",
  "main": "L07-SecY-XX88XXX-q1.js",
  "scripts": {
    "start": "nodemon

L07-SecY-XX88XXX-q1.js"
  },
  "author": "[Your Name], Section
Y,
                6x88xxx",
  "license": "ISC",
  "dependencies": {
    "express": "^4.17.2",
    "nodemon": "^2.0.15"
  }
```

```
(root)
  > node_modules
  <> contact_us.html
  |_ L07-SecY-XX88XXX-q1.js.js
  |_ package-lock.json
  |_ package.json
```

}	

Here are the example of outputs from Q1 program:

#1 Start your server with **`npm start`**

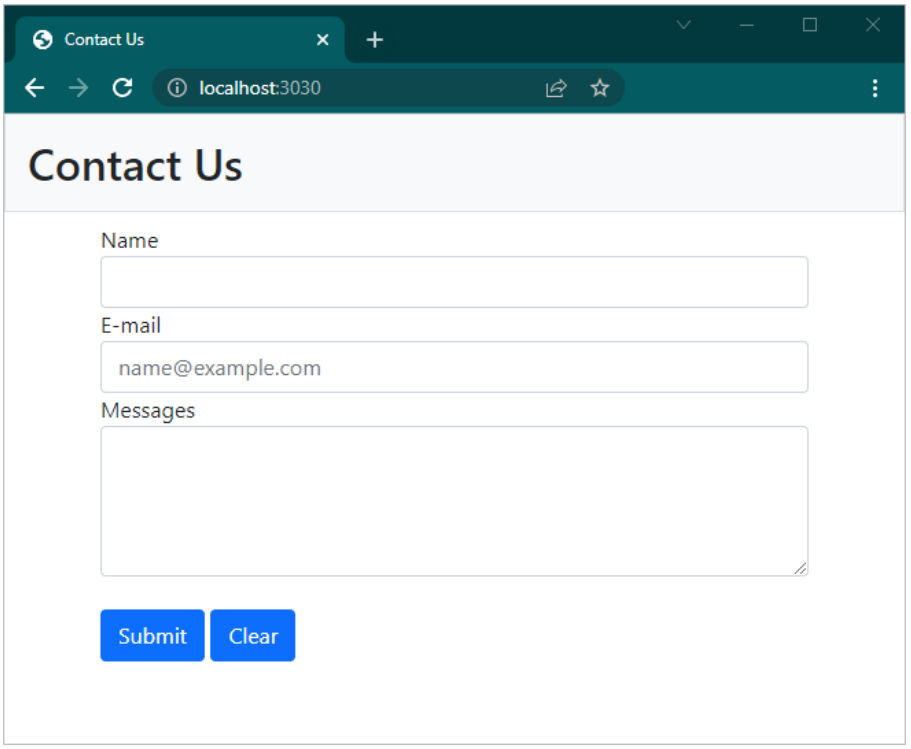
```
C:\[PATH]\L07-q1>npm start
> l07-q1@1.0.0 start
> nodemon L07-SecY-XX88XXX-q1.js

[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node L07-SecY-XX88XXX-q1.js`
Server listening at Port 3030
_
```

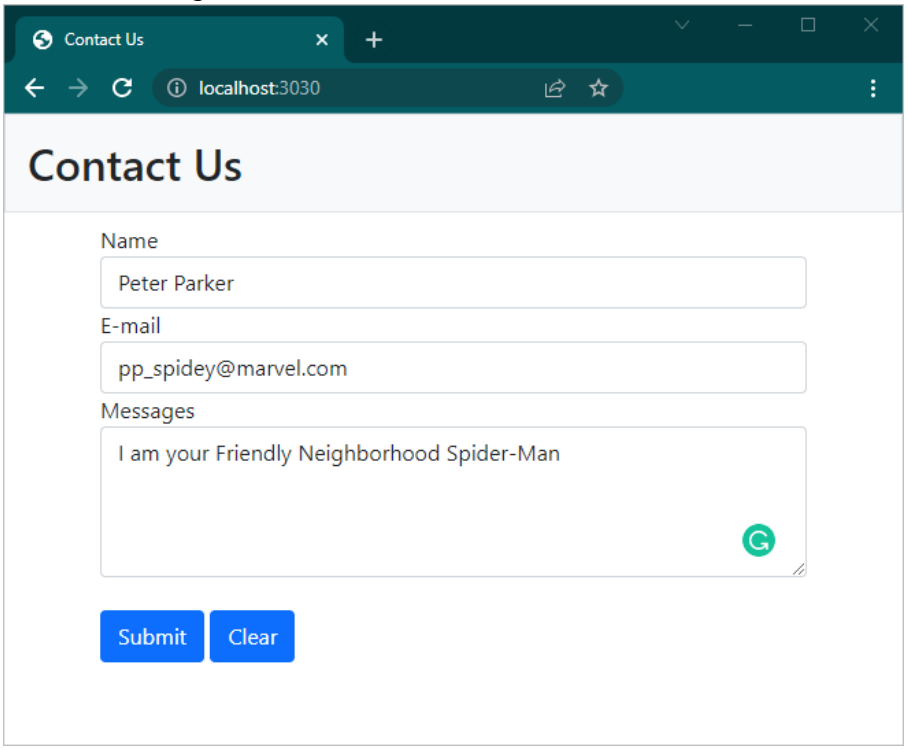
#2 Type http://localhost:3030/ in the browser, the terminal must show

```
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node 6x88xxx_secY_q1.js`
Server listening at Port 3030
Accessed Contact Us
_
```

The browser must shows



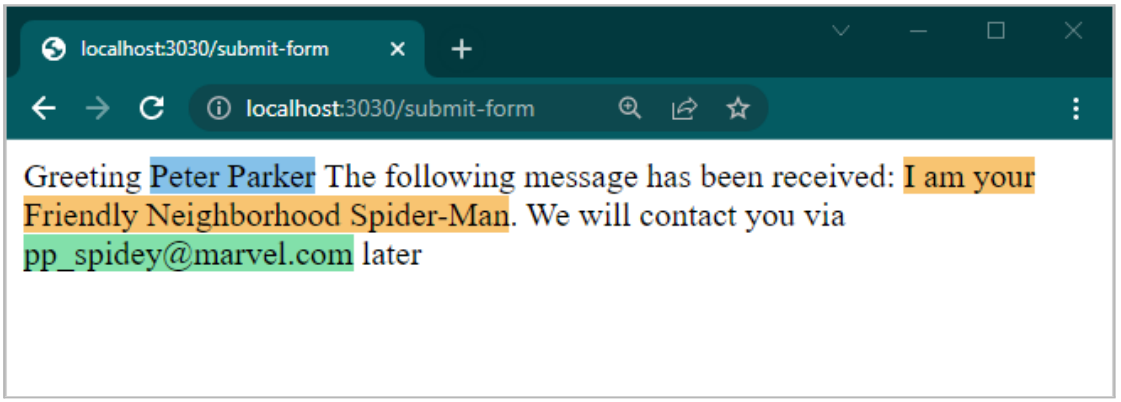
Example of form filling



#4 After user click a “submit” button, the form should be sent to the server at "`/submit-form`" via `POST` and show the following information in the terminal.
Hint: you need to modify `contact_us.html` partially.

```
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node 6x88xxx_secY_q1.js`
Server listening at Port 3030
Accessed Contact Us
Form submitted by Peter Parker
_
```

The server should respond with the following message on the browser. **Hint:** `res.send()` can contain some HTML tags and (inline) CSS.



Submit your `contact_us.html`, your JS file: `L07-SecY-XX88XXX-q1.js`, and `package.json` by class time via MyCourses (Note: `XXX` is your ID and `Y` is your section). You must notify LAs to grade your work before submitting to MyCourses.

Q2: [No submission] Express Node.js with DB: Your task is to implement an Express Node.js server connected to the `tinycollege` database using `mysql2`.

Note: You are required to have the `tinycollege` database in your local machine. The SQL script to install the database can be found in MyCourses. You will need a user account to access the `tinycollege` database with the proper permissions.

Write the code to create a server running at the port 3030 and connect to the `tinycollege` database. The server must handle the following routes

- GET: "**`/cis-students`**" to retrieve CIS student name (first name and last name) and gpa from the database in the JSON format
- GET: "**`/cis-courses`**" to retrieve CIS course code and name (fi from the database in the JSON format

You should follow the steps for initialization of the project as usual (see steps in Q1) Name your main `.js` files as `L07-SecY-XX88XXX-q2.js`

The server port and database connection data (e.g. host, username, password, and

database name) must be stored in the `.env` file, i.e., you also need to install `dotenv` by `npm install dotenv --save`. Finally, install `mysql2` for MySQL connection with Node.js by `npm install mysql2 --save`

After the initialization, your `package.json` file and the directory should look like this

```
{
  "name": "l07-q2",
  "version": "1.0.0",
  "description": "",
  "main": "L07-SecY-XX88XXX-q2",
  "scripts": {
    "start": "nodemon L07-SecY-XX88XXX-q2"
  },
  "author": "[Your Name], Section Y,
                  6x88xxx",
  "license": "ISC",
  "dependencies": {
    "dotenv": "^16.0.0",
    "express": "^4.17.2",
    "mysql2": "^2.3.3",
    "nodemon": "^2.0.15"
  }
}
```

```
(root)
  > node_modules
  |_ .env
  |_ L07-SecY-XX88XXX-q2.js
  |_ package-lock.json
  |_ package.json
```

Here are the example of outputs from Q2 program:

#1 Start your server with **`npm start`**

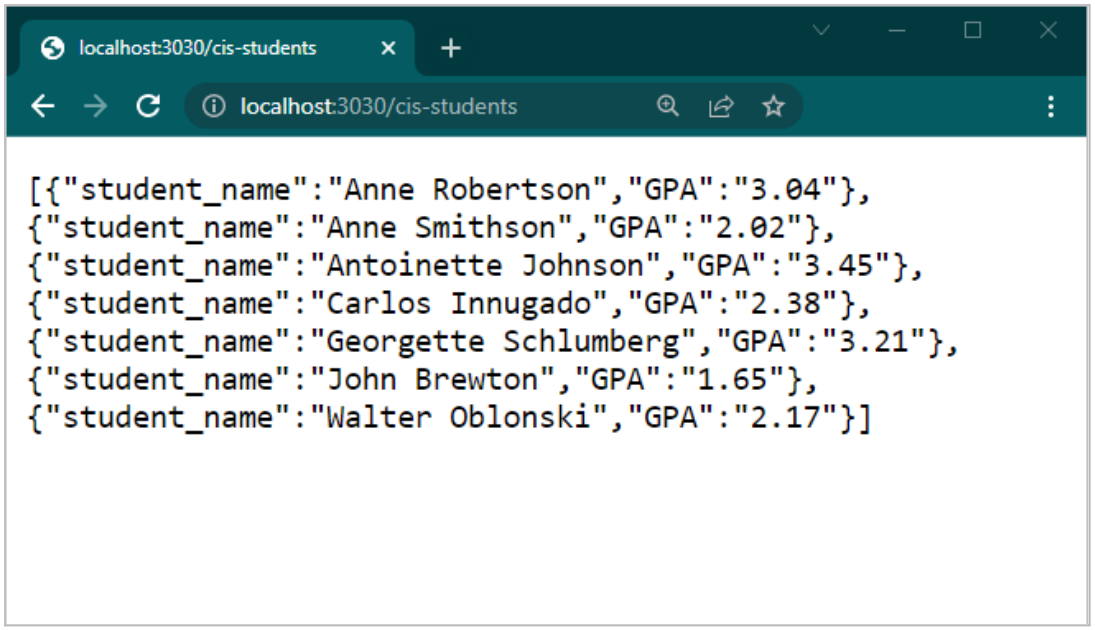
```
C:\[PATH]\L07-q2>npm start
> l07-q1@1.0.0 start
> nodemon L07-SecY-XX88XXX-q2.js

[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node L07-SecY-XX88XXX-q2.js`
Server listening at Port 3030
Connected DB: tinycollege
_
```

#2 Type http://localhost:3030/cis-students, the terminal should show

```
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node L07-SecY-XX88XXX-q2.js`
Server listening at Port 3030
Connected DB: tinycollege
Retrieved all CIS students in tinycollege...
7 CIS students returned
_
```

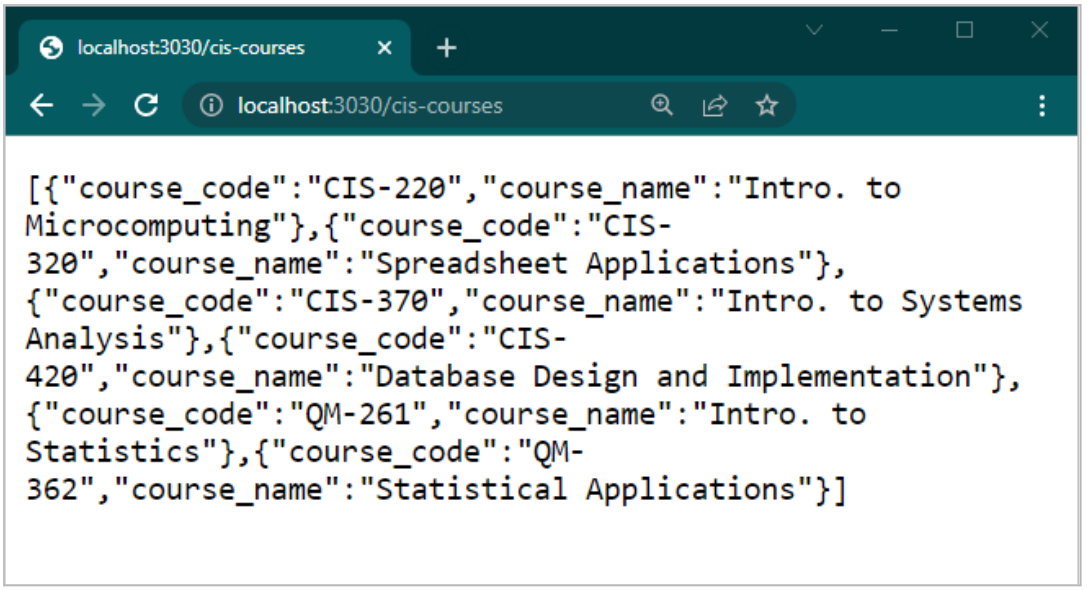
The browser should show the following result in the JSON format and the result should show `student_name` (first name and last name) and their GPA of the student in the CIS department, sorted by the name alphabetically



#3 Type http://localhost:3030/cis-courses, the terminal should show

```
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node L07-SecY-XX88XXX-q2.js`
Server listening at Port 3030
Connected DB: tinycollege
Retrieved all CIS students in tinycollege...
7 CIS students returned
Retrieved all CIS courses in tinycollege...
6 CIS courses returned
```

The browser should show the following result in the JSON format and the result should show `course_code` and `course_name` of the CIS department.



No need to submit this question
