8.3D: Writing a Multi-Route Template Server Application

Tasks

In this task your objective is to create a server application that can serve *dynamic pages* using a template engine within Express, in this case Encapsulated JS (**EJS**).

The server provides a simple interface to a access and display data on users feedback (comments and rating) on their favourite ice cream flavours. You will be provided with sample (very *simple*) data in the form of `JSON` files and will include:

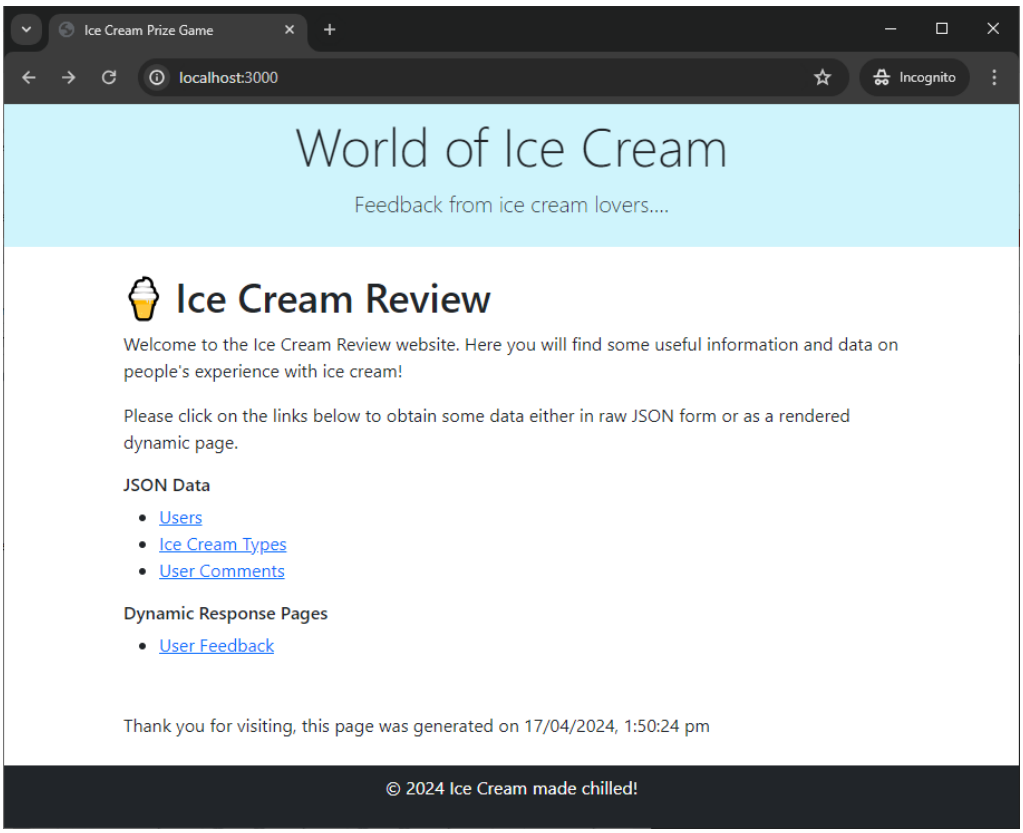
- User details
- Ice Cream Types
- User Comments
    - Ice cream type
    - Rating out of 5
    - Some comments on the taste

The routes your server should support include:

- http://localhost:3000/
- http://localhost:3000/users [returns JSON data]
- http://localhost:3000/icecreamtypes [returns JSON data]
- http://localhost:3000/comments [returns JSON data]
- http://localhost:3000/feedback

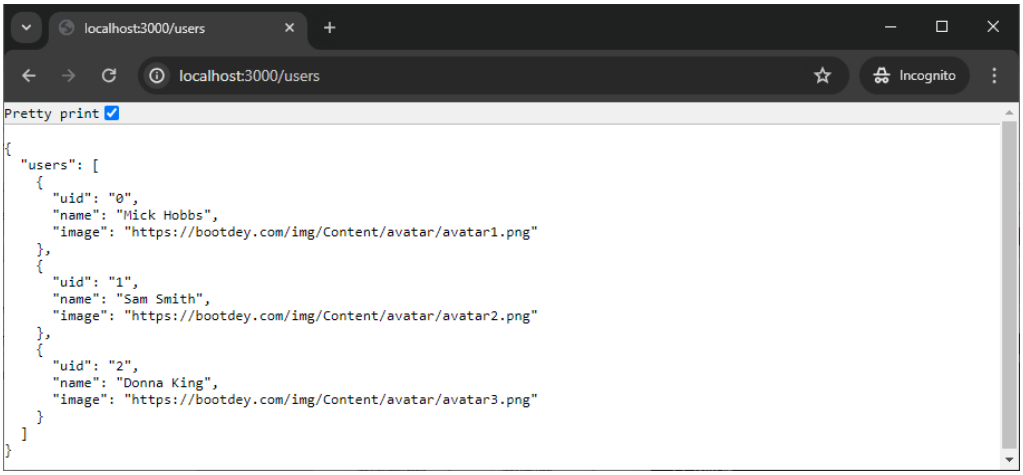
The default `/` (home) and the `/feedback` routes should both render dynamic pages, while the remaining routes all return raw `JSON` data for their respective data files.

An example of the output from the default `/` (home) page is shown below (**NOTE**: The date/time shown on the page is dynamic):



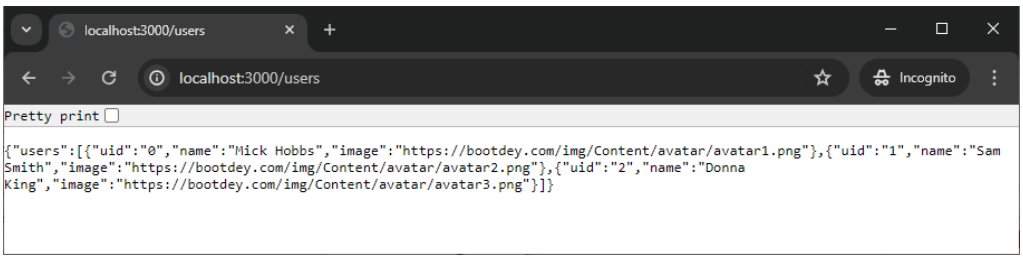
Task8.3.1 Node.js application serving the `/` route

The routes that return raw `JSON` data should display output similar to that shown below:



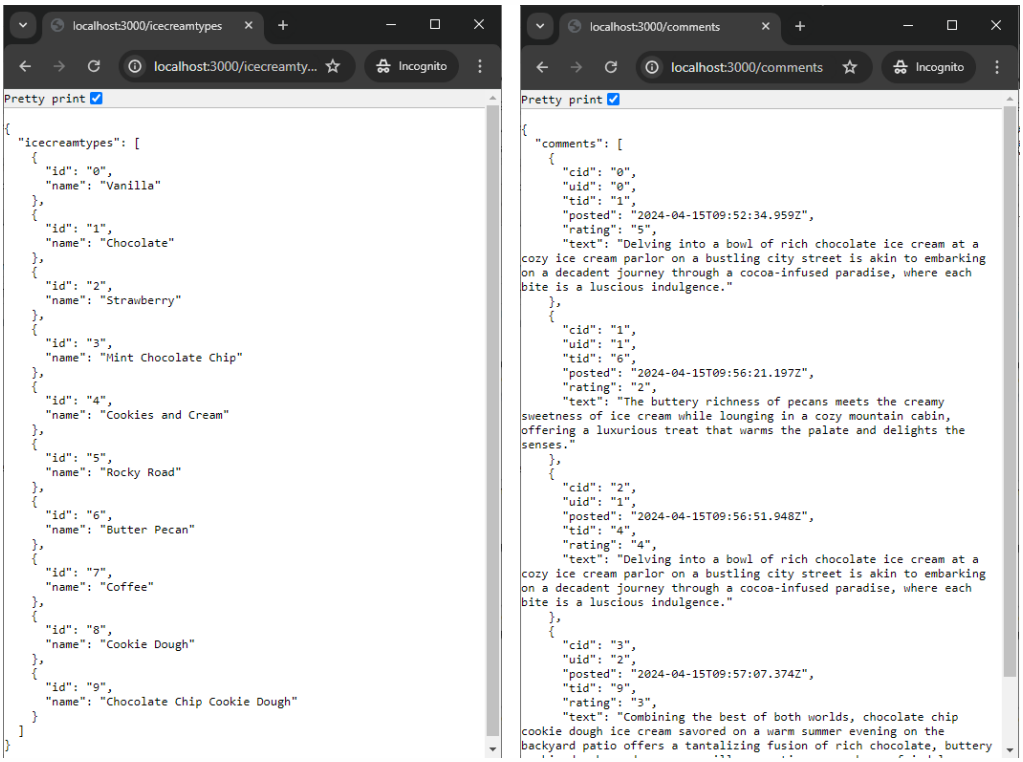
Task8.3.2 JSON repsonse to the `/users` route - formmated output

Although, if your browser doesn't automatically display formatted/indented JSON, it may appear as plain text, as below:



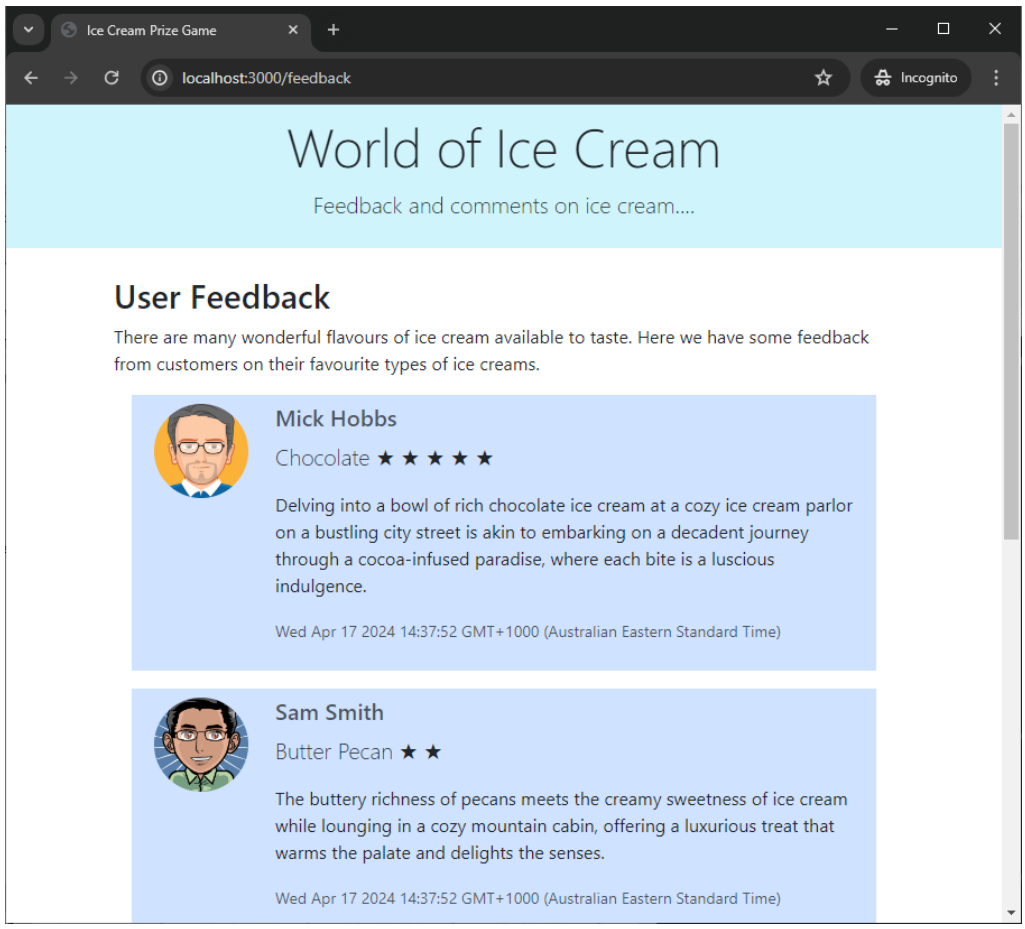
Task8.3.2-b JSON repsonse to the `/users` route - unformatted output

Sample output for the two remaining JSON data routes `/icecreamtypes` and `/comments` are:

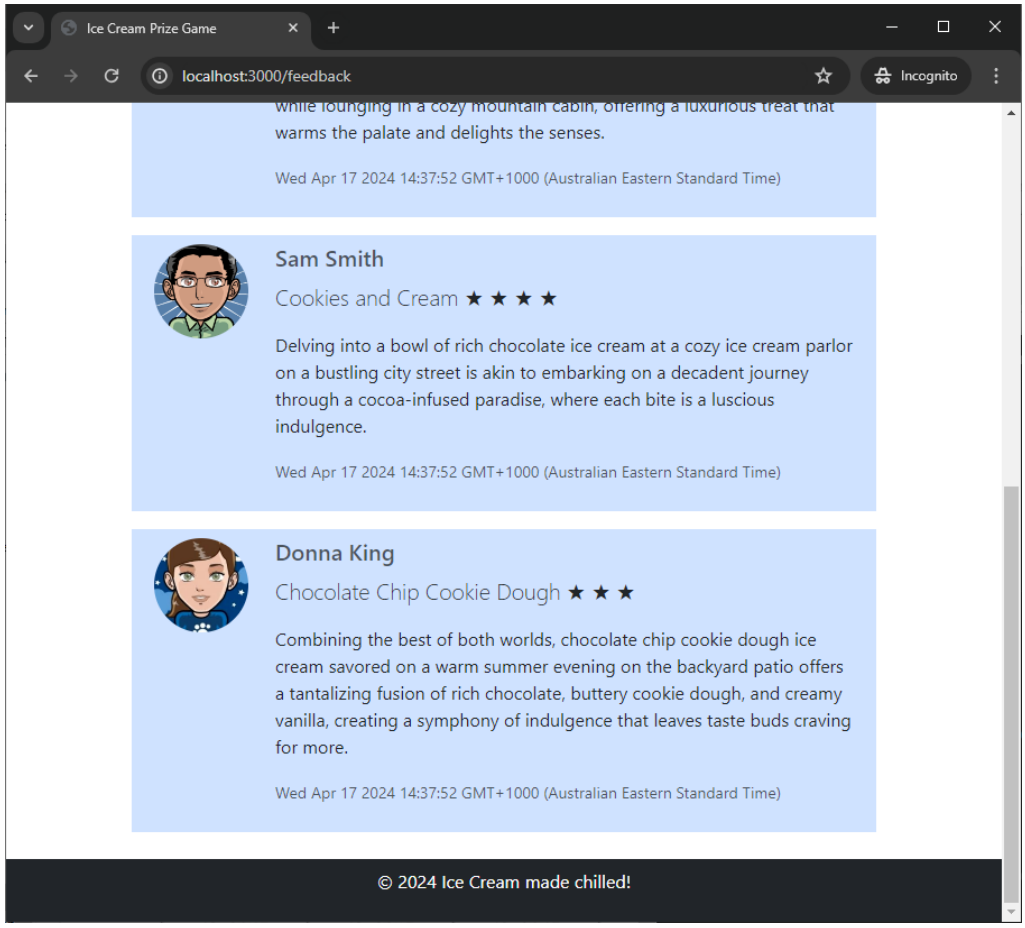


Task8.3.3 JSON repsonse to the `/icecreamtypes` & `/comments` route - formatted output

The final route `/feedback` should render a dynamic page that combines all three data resources into a list of feedback and presents these neatly on the page. A sample of how this may be presented is shown in the screenshot below:



Task8.3.4 Dynamically rendered page to the `/feedback` route



Task8.3.5 Dynamically rendered page to the `/feedback` route (page 2)

Resources Provided

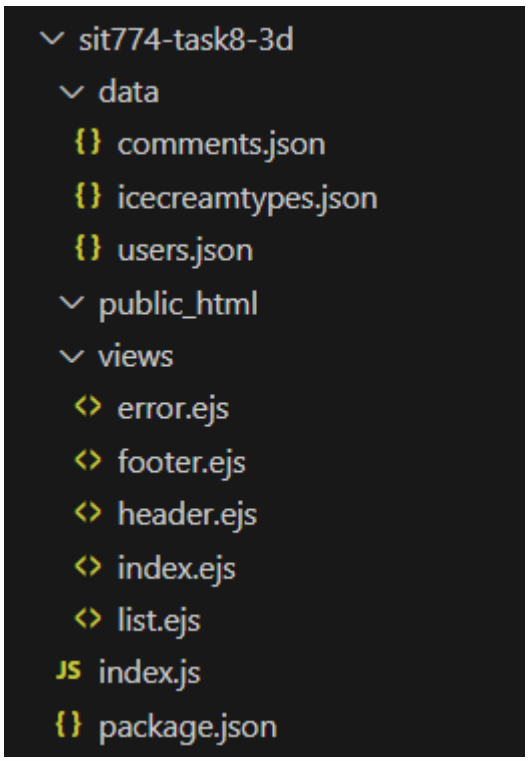
The following resources are provided for this task:

- The EJS template files used to render the default `/` (home) route:
  - `index.ejs`
  - `header.ejs` - The *include* files used in `index.ejs`
  - `footer.ejs` - The *include* files used in `index.ejs`
  - `error.ejs` - Skeleton error template file
- The raw `JSON` data file for:
  - `users.json` - User details
  - `icecreamtypes.json` - Ice Cream Types
  - `comments.json` - User Comments

Steps

Follow the steps below to complete this task:

1. Create your own local directory (or you can use the same local directory created in Task 8.1).
2. Add a new Node Module package to install the template engine for express supporting *Encapsulated JS* or `ejs`
   - `npm install ejs`
3. Create a new folder in your project called `data`, to hold the `JSON` data files
4. Create a new folder in your project called `views`, to hold the `ejs` template file to render your output pages
5. Your directory structure should be as shown below:



Task8.3.6 Project directory structure

6. Use express module to create a Node.js application (i.e., `index.js`) that will set a local web server. The server listens to the port **3000**.

7. Add to your `index.js` server the access to the `path` module (already installed along with the `express` module):

```
const path = require('path');     // Added to support access to file system paths
```

8. Add variables (constants) to access the `JSON` data provided for the users, icecreamtypes and comments:

```
const jsonIcecreamTypeData = require(path.join(__dirname, 'data/icecreamtypes'));
const jsonCommentData = require(path.join(__dirname, 'data/comments'));
const jsonUserData = require(path.join(__dirname, 'data/users'));
```

9. Add and configure the EJS template engine:

```
// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');
```

10. Add a route handler for a `GET` request on the path `/` to respond with a rendered page from the *EJS template* `index.ejs` using the `res.render()` command. Note the arguments provided include the name of the template to use and the data to use inside the template as the second parameter, such as:

```
app.get('/', (req, res, next) => {
    res.render('index', { title: 'Ice Cream Review' });
});
```

11. Add the route handlers to respond with the `JSON` data files using the `res.json()`

```
app.get('/users', (req, res, next) => {
    res.json(jsonUserData);
});

... // add others for `/icecreamtypes` and `/comments`
```

12. Create and build a new template for the page to display the list of feedback comments showing the users and their ratings. This should be held in a `list.ejs` file.

13. Add the route handler for `/feedback` to render a template `list.ejs` using the data from the `JSON` files. It could be called with multiple parameters, such as:

```
res.render('list', {
    title: 'User Feedback',
    types: jsonIcecreamTypeData.icecreamtypes,
    comments: jsonCommentData.comments,
    users: jsonUserData.users
    });
```

14. Run your project `npm run start:dev` in a Command Prompt (Windows) or Terminal (Mac OS) within your local directory to start the server.

15. Open a web browser and use the address to test the routes you have implemented:

    - `http://localhost:3000/`
        - Should return a rendered web page
    - `http://localhost:3000/users`
        - Should return JSON data
    - `http://localhost:3000/icecreamtypes`
        - Should return JSON data
    - `http://localhost:3000/comments`
        - Should return JSON data
    - `http://localhost:3000/feedback`
        - Should return a rendered web page

Hints

To complete this task, review the **EJS Documentation** (https://www.npmjs.com/package/ejs) on using template elements to display content; especially that on `loops`.

What will you submit?

You should submit:

- Source code of the **server** file `index.js`
- Source code of the **template** file `list.ejs`
- Screenshot (1 or 5) of the response to: `http://localhost:3000/`
- Screenshot (2 or 5) of the response to: `http://localhost:3000/users`
- Screenshot (3 or 5) of the response to: `http://localhost:3000/icecreamtypes`
- Screenshot (4 or 5) of the response to: `http://localhost:3000/comments`
- Screenshot (5 or 5) of the response to: `http://localhost:3000/feedback`