# Express Shopping List

For this exercise we will be building a simple JSON API application where we will store a shopping list. You should use an **array** to store your items in the shopping list.

Each item should be a JavaScript object with the keys of name, and price.

Remember that since you are using an **array** for storage, this will be cleared each time the server restarts. Create a simple file called ***fakeDb.js*** which contains the following:

```
global.items = []

module.exports = items
```

Use this list of ***items*** in your routes and test files.

Your application should have the following routes:

1. ***GET /items*** - this should render a list of shopping items.

   *Here is what a response looks like:*

   ***[{"name": "popsicle", "price": 1.45}, {"name":"cheerios", "price": 3.40}]***

2. ***POST /items*** - this route should accept JSON data and add it to the shopping list.

   *Here is what a sample request/response looks like:*

   ***{"name":"popsicle", "price": 1.45} => {"added": {"name": "popsicle", "price": 1.45}}***

3. ***GET /items/:name*** - this route should display a single item's name and price.

   *Here is what a sample response looks like:*

   ***{"name": "popsicle", "price": 1.45}***

4. ***PATCH /items/:name***, this route should modify a single item's name and/or price.

   *Here is what a sample request/response looks like:*

   ***{"name":"new popsicle", "price": 2.45} => {"updated": {"name": "new popsicle", "price": 2.45}}***

5. ***DELETE /items/:name*** - this route should allow you to delete a specific item from the array.

   Here is what a sample response looks like:

   ***{message: "Deleted"}***

Please make use of the Express Router.

**Warning: STOP and make sure your routes are tested**

Make sure you have tested all of these routes with Jest and Supertest.

## Further Study

1. Make your own data store! Read/Write to a JSON file instead of using an array for storage.
2. Use OOP to help refactor your logic for reading/writing. This is almost like building your own ORM!
3. Build a front-end that interacts with your API. This will involve figuring out how to load static assets with express. Try doing this in jQuery, or vanilla JavaScript (ultra challenge)!

## Solution

View our solution