# Search For To-Do Items

⏱ an hour

## Search For To-Do Items

What may be the most complex set of tests to write (except that weird event emitter thing), search makes you think though what it should do in a variety of cases.

Here's the relevant part of the server that handles a search query.

```
else if (req.url.startsWith('/search') && req.method === 'GET') {
  const [_, query] = req.url.split('?', 2);
  const { term } = querystring.parse(query);
  const filePath = path.join(__dirname, 'search-items-screen.html');
  const template = await fs.promises.readFile(filePath, 'utf-8');
  let foundItems = [];
  if (term) {
    foundItems = searchItems(items, term);
  }
  const html = mergeItems(template, foundItems);
  res.setHeader('Content-Type', 'text/html');
  res.writeHead(200);
  res.write(html);
}
```

You've already tested `mergeItems`, so that's not needed, again. The only method that you will need to test is `searchItems`.

Open **search-items.js** and review how that code is working. It takes a list of `items` and a search `term`. The first thing it does is force the term to lower case.

```
term = term.toLowerCase();
```

Then, it uses the `filter` function on the array to create a new array of items that meet the comparison in the function. The comparison function makes the title lower case and checks to see if the term is contained in that string.

```
return items.filter(x => {
  const title = x.title.toLowerCase();
  return title.indexOf(term) >= 0;
});
```

If the term *is* in the title, then the comparison returns `true` and the `filter` function will add it to the new array. If the term is *not* in the title, the comparison returns `false` and it is not added to the new array.

Here is an example. Supposed you have the following items in your array.

```
[
  { title: 'Go grocery shopping', category: 'Home' },
  { title: 'Play with my puppy', category: 'Pet' },
  { title: 'Shop for a puppy bed', category: 'Pet' },
]
```

Now, say the search term someone entered is "SHOP". This is what happens in the function.

```
Convert "SHOP" to "shop"
Filter the array of items based on the term "shop":
  Item 1:
    Convert "Go grocery shopping" to "go grocery shopping"
    Does it contain the term "shop"? YES
    Add it to the new array
  Item 2:
    Convert "Play with my puppy" to "play with my puppy"
    Does it contain the term "shop"? NO
  Item 3:
    Convert "Shop for a puppy bed" to "shop for a puppy bed"
    Does it contain the term "shop"? YES
    Add it to the new array
Return the new array that contains items 1 and 3
```

So, that's what you want to test for.

Open **search-items-spec.js**. You'll see three tests.

In the first test, you are asked to fix the *arrange* step to declare `items` and `term` given the directions. This is not a trick. It's just declaring those two variables that it's asking you to create.

In the second test, fix the *assert* step to assert the proper length of the result by completely replacing the `expect.fail` line.

In the third test, you are asked to fix the *arrange* step by choosing a string value for `term` that makes the rest of the test pass.

## What have you done?

Now that you've done that, you've won the entire game! All of the meaty logic of the game is now well tested. If someone were to come along and try to change the code, when the tests ran, it would check to make sure they didn't accidentally break something in their earnest to add new functionality!

Here's what you did:

- You've looked at, read, and understood other people's code
- You've seen and used a variety of assertions
- You've seen how to do real (not fake) asynchronous testing using the `done` method
- You've invested time in hardening the maintainability of an application

Here's a link to a solution. https://appacademy-open-assets.s3-us-west-1.amazonaws.com/Module-JavaScript/testing/projects/testing-an-existing-project-solution.zip

In the next step, you're going to use the fact that you have tests to radically change the code.

**Did you find this lesson helpful?**

No 👎 | 👍 Yes

✓ Mark As Complete

Finished with this task? Click **Mark as Complete** to continue to the next page!