Lab 16.3 - Test a Promise-Based async/await API

The labs-3 folder contains a `package.json` file and a `store.js` file.

The `package.json` file contains the following:

```
{
  "name": "labs-3",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "UNLICENSED"
}
```

The `store.js` file contains the following:

```
'use strict'
const { promisify } = require('util')
const timeout = promisify(setTimeout)
module.exports = async (value) => {
  if (Buffer.isBuffer(value) === false) {
    throw Error('input must be a buffer')
  }
  await timeout(300)
  const id = Math.random().toString(36).split('.')[1].slice(0, 4)
```



```
  return { id }
}
```

This API mimics some kind of async storage mechanism, such as to a database. In some circumstances it is infeasible to check for a specific value (for instance an ID returned from a database). For those cases, we can check for the presence of an ID, or apply some validation. In our case we can at least check that the length of the ID is 4.

Write tests for the `store.js` file. Ensure that when `npm test` is executed with the labs-2 folder as the current working directory the `store.js` file is fully tested.

Any additional dependencies, such as a test harness, may be additionally installed.

Also in the labs-3 folder is a `validate.js` file. The implementation can be checked with `node validate.js`. The implementation is successful if the final output of `node validate.js` is `passed!`.

For completeness the following is the `validate.js` file contains the following, but it is not necessary to understand it for the purposes of this exercise:

```
'use strict'
const assert = require('assert').strict
const { spawnSync } = require('child_process')
const { writeFileSync } = require('fs')
const store = require.resolve('./store')
const storeCode =
Buffer.from('27757365207374726963742 70a636f6e7374207b2070726f6d6973696
679207d203d207265717569726528277574696c27290a636f6e73742074696d656f757
4203d2070726f6d697369667928736574546 96d656f7574290a6d6f64756c652e65787
06f727473203d206173796e63202876616c756529203d3e207b0a20206966202842756
66665722e697342756666665722876616c756529203d3d3d2066616c736529207b0a202
020207468726f77204572726f722827696e707574206d75737420626520612061206275666
6657227290a20207d0a2020617761697420746 96d656f757428333030290a2020636f6
e7374206964203d204d6174682e72616e646f6d28292e746f537472696e67283336292
e73706c697428272e27295b315d2e736c69636528302c2034290a202072657475726e2
07b206964207d0a7d0a', 'hex')

try {
  {
    writeFileSync(store, storeCode)
```



```
    const sp = spawnSync('npm', ['test'], {
      env: { ...process.env, NODE_OPTIONS:
'--unhandled-rejections=strict' },
      stdio: 'ignore'
    })

    assert.equal(sp.status, 0, 'tests should be successful (is
package.json test script configured?)')
  }

  {
    const badOutput = `'use strict'
      const { promisify } = require('util')
      const timeout = promisify(setTimeout)
      module.exports = async (value) => {
        if (Buffer.isBuffer(value) === false) {
          throw Error('input must be a buffer')
        }
        await timeout(300)
        const id = Math.random().toString(36).split('.')[1].slice(0,
2)
        return { id }
      }
    `

    writeFileSync(store, badOutput)

    const sp = spawnSync('npm', ['test'], {
      env: { ...process.env, NODE_OPTIONS:
'--unhandled-rejections=strict' },
      stdio: 'ignore'
    })
    assert.equal(sp.status, 1, 'output should be tested (id length)')
  }

  {
    const badValidation = `'use strict'
      const { promisify } = require('util')
      const timeout = promisify(setTimeout)
      module.exports = async (value) => {
        await timeout(300)
```



```
        const id = Math.random().toString(36).split('.')[1].slice(0,
4)
        return { id }
      }
    `

    writeFileSync(store, badValidation)

    const sp = spawnSync('npm', ['test'], {
      env: { ...process.env, NODE_OPTIONS:
'--unhandled-rejections=strict' },
      stdio: 'ignore'
    })

    assert.equal(sp.status, 1, 'error case should be tested')
  }

  console.log('passed!')
} finally {
  writeFileSync(store, storeCode)
}
```

