# Caremerge Test

This test consists of three parts. You need to implement the same task three times using different control flow strategies.

The purpose of this test is to measure your understanding of control flow in an asynchronous environment.

All tasks should be implemented using Node.js . We would prefer raw Node but you may use express.js if you want.

## Problem Statement

Create a node server which responds to one and only one route : `GET /I/want/title`

This route expects a list of websites addresses in query string format e.g.

1. `/I/want/title/?address=google.com`
2. `/I/want/title/?address=http://yahoo.com`
3. `/I/want/title/?address=google.com&address=www.dawn.com/events/`

etc.

The number of addresses can be more than one.

The route will make request to each of the `address` passed to it. It will parse out the `<title></title>` tags, render them in html and send back the html in response. e.g. the response to above #3 example would be:

```
<html>
<head></head>
<body>

    <h1> Following are the titles of given websites: </h1>

    <ul>
       <li> google.com - "Google" </li>
       <li> www.dawn.com/events/ - "Events - DAWN.COM" </li>
    </ul>
</body>
</html>
```

For all other routes, the server should return HTTP code `404` .

Please make sure to add error handling e.g.

`/I/want/title/?address=asdasdasd` should return `<li> asdasdasd - NO RESPONSE </li>`

## Tasks

1. Implement the above task using plain node.js callbacks (you can use `express` or `http` or any other helper module but nothing which absracts control flow).
2. Implement the above using some kind of flow library e.g. async.js or step.js
3. Implement the above using Promises. You could use any library e.g. RSVP or Q

BONUS: Implement the above using Streams e.g. bacon.js or RxJs