

Back-end



Javascript Modules

- To structure an application coherently, the backend consists of separate Javascript files.
- Objects declared in these files must be
 - exported by one file
 - imported by another
- In order to keep each module focused on a specific responsibility

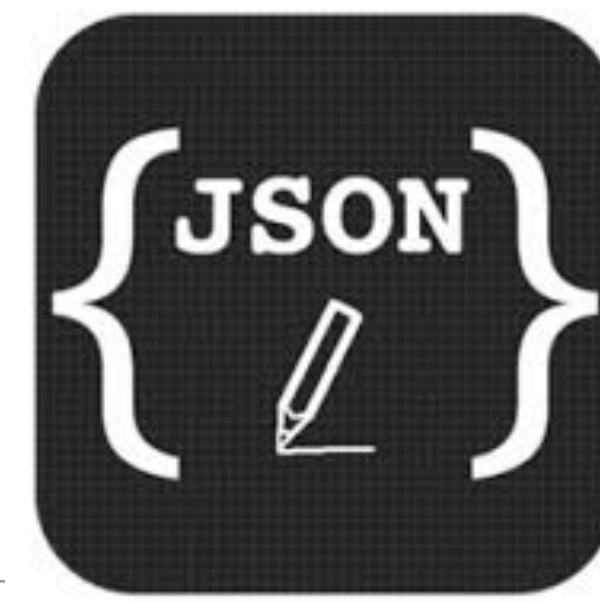
Application Structure

```
📁 assets
  controllers/about.js
  controllers/dashboard.js
  utils/logger.js
  views/layouts/main.hbs
  views/partials/mainpanel.hbs
  views/partials/menu.hbs
  views/about.hbs
  views/dashboard.hbs
🔑 .env
.gitignore
README.md
package.json
routes.js
server.js
```

- App implements Routes + Model/View/Controller Architecture
- These objects collaborate to support structured, predictable application workflow

Back-end

JS



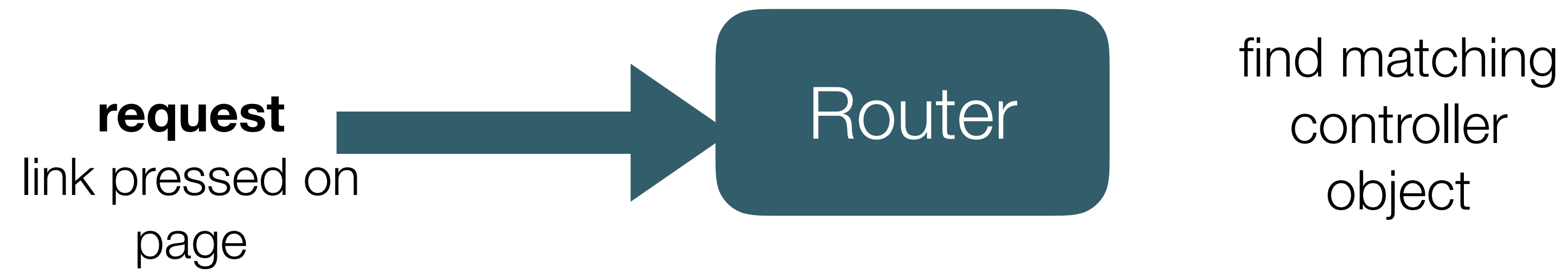
```
controllers/about.js
controllers/dashboard.js
utils/logger.js
⌘ .env
.gitignore
README.md
package.json
routes.js
server.js
```

- All written in Javascript + JSON
- Consists of:
 - **Server** - main entry point
 - **Routes** - supported urls
 - **Controllers** - objects to handle the routes
 - **Config** - .gitignore, .jscsrc, env, package.json, readme.md
- Will include **Models** later...

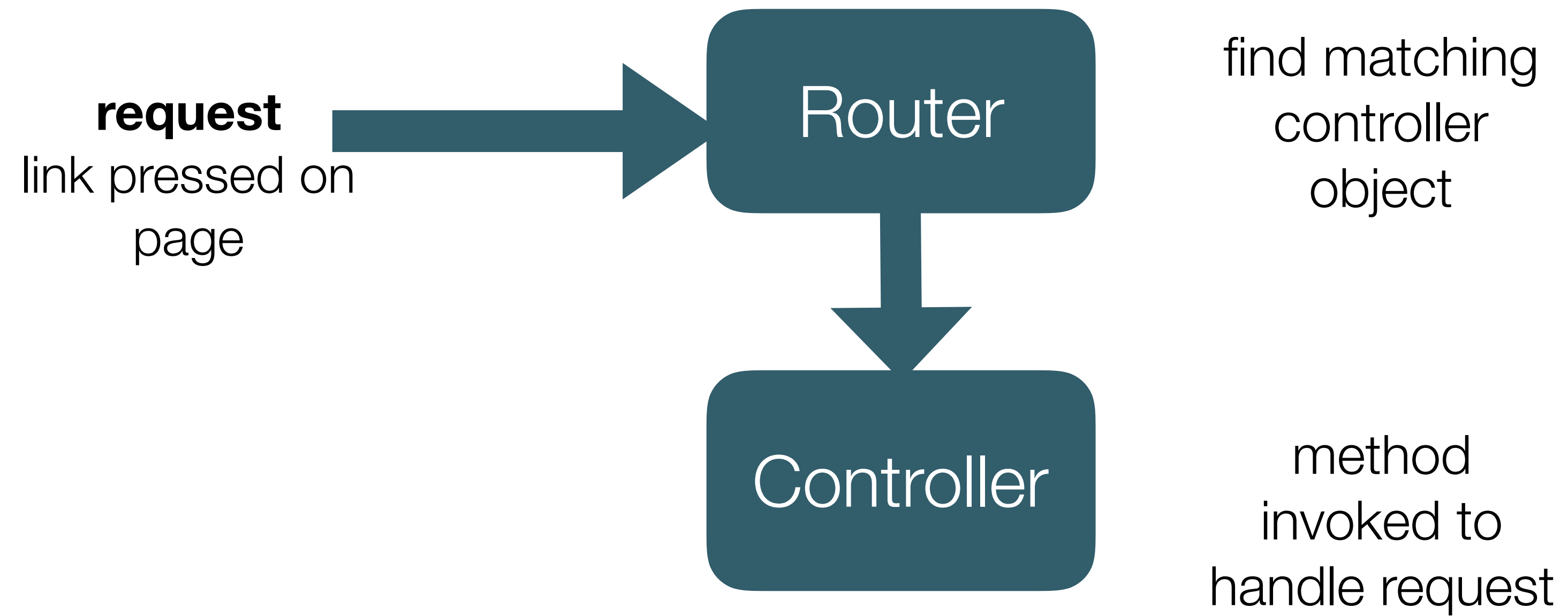
Request/Response Lifecycle

1. **Request** - link pressed on page
2. **Router** - find matching controller object
3. **Controller** - method invoked to handle request
4. **View** - data sent from controller to view to construct response
5. **Response** - complete page rendered into browser

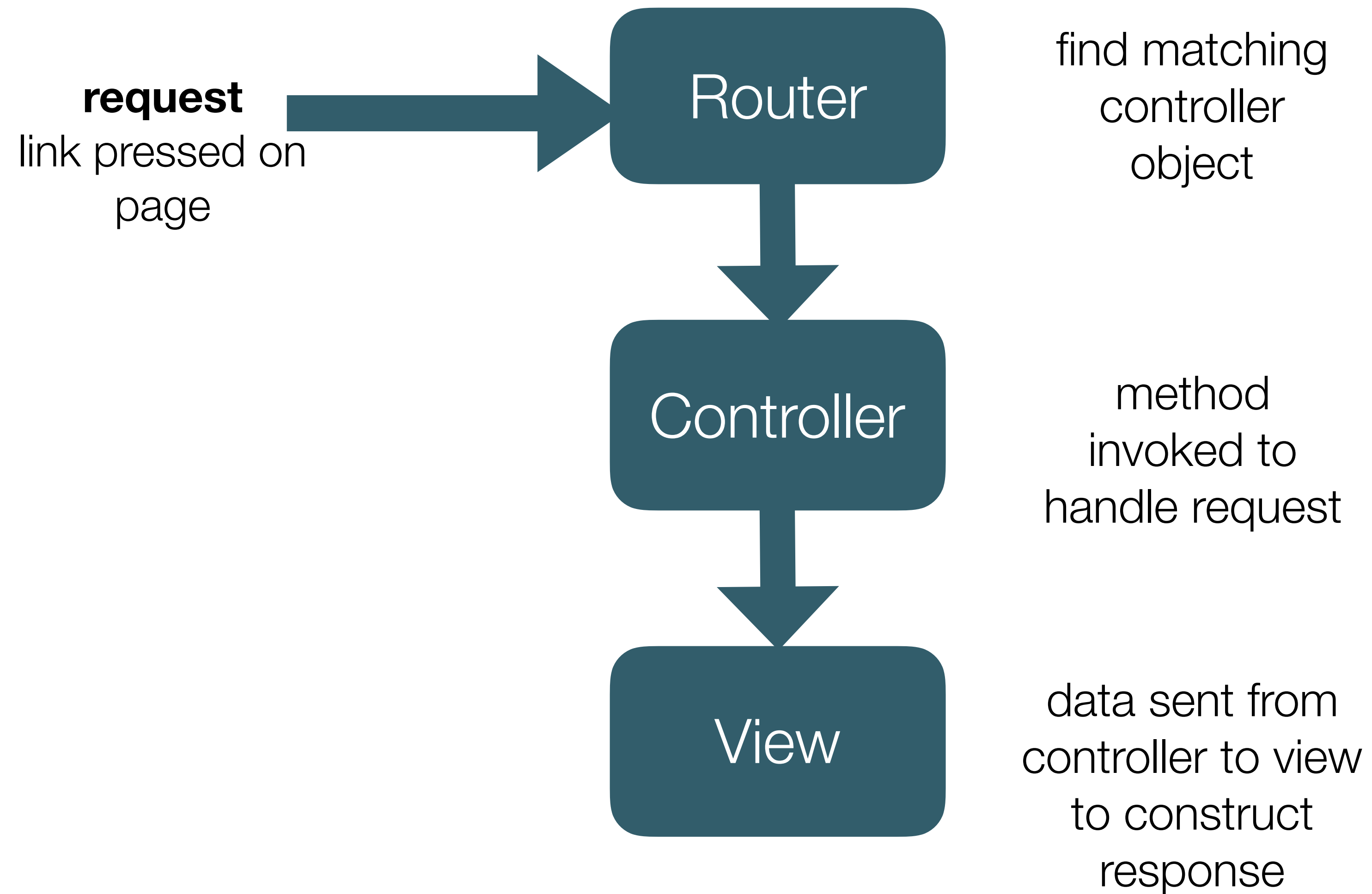
Router/Controller/View



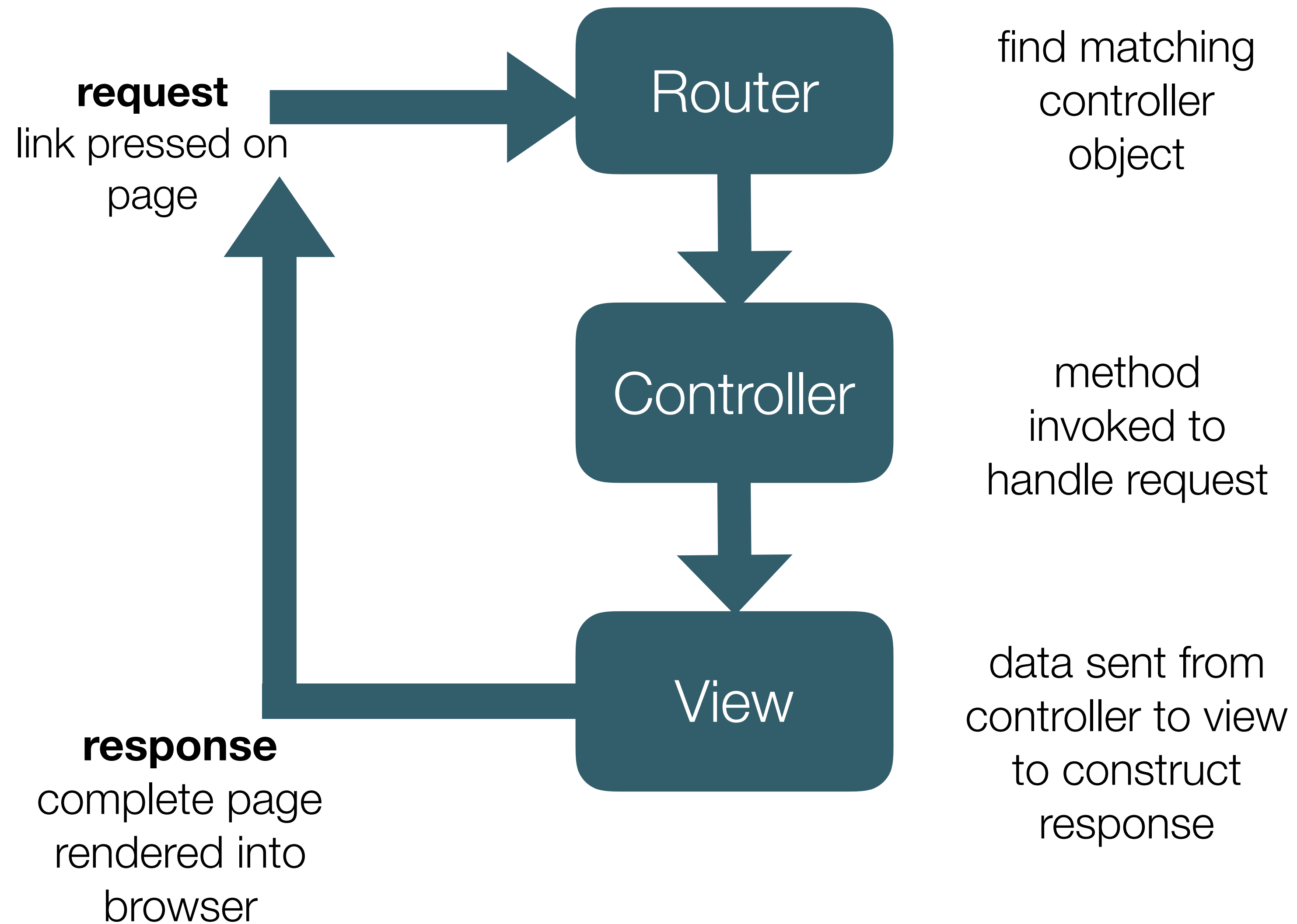
Router/Controller/View



Router/Controller/View

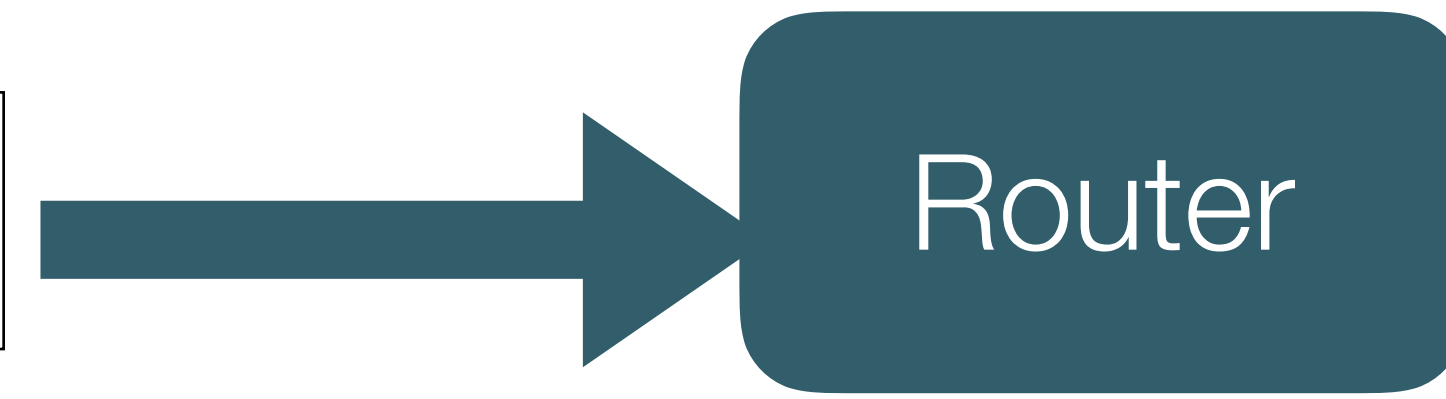


Router/Controller/View



Request - link pressed on page

```
...  
<a id="dashboard" class="item" href="/dashboard"> Dashboard </a>  
<a id="about" class="item" href="/about"> About </a>  
...
```

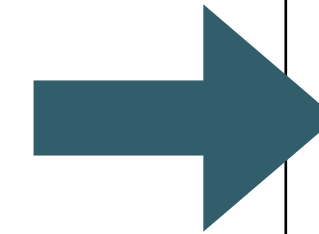


- Requests defined in links in views:
 - href in <a> tags
 - href in Menus
 - href in Buttons
 - action links in forms

Router - find matching controller object

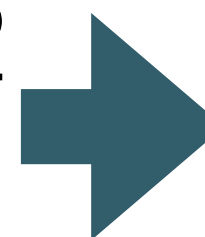
```
controllers/about.js  
controllers/dashboard.js  
utils/logger.js
```

```
⌘ .env  
.gitignore  
README.md  
package.json  
routes.js  
server.js
```



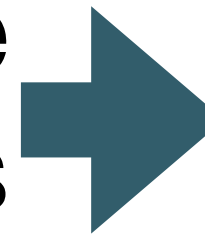
routes.js

Import 2
objects:



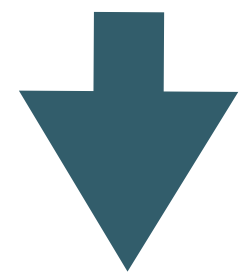
```
...  
const dashboard = require('./controllers/dashboard.js');  
const about     = require('./controllers/about.js');  
  
router.get('/',          dashboard.index);  
router.get('/dashboard', dashboard.index);  
router.get('/about',     about.index);  
...
```

Match these
two objects
with each of
these 'links'



Router Behaviour

If user selects these links...



'/'



dashboard.index

/'dashboard'



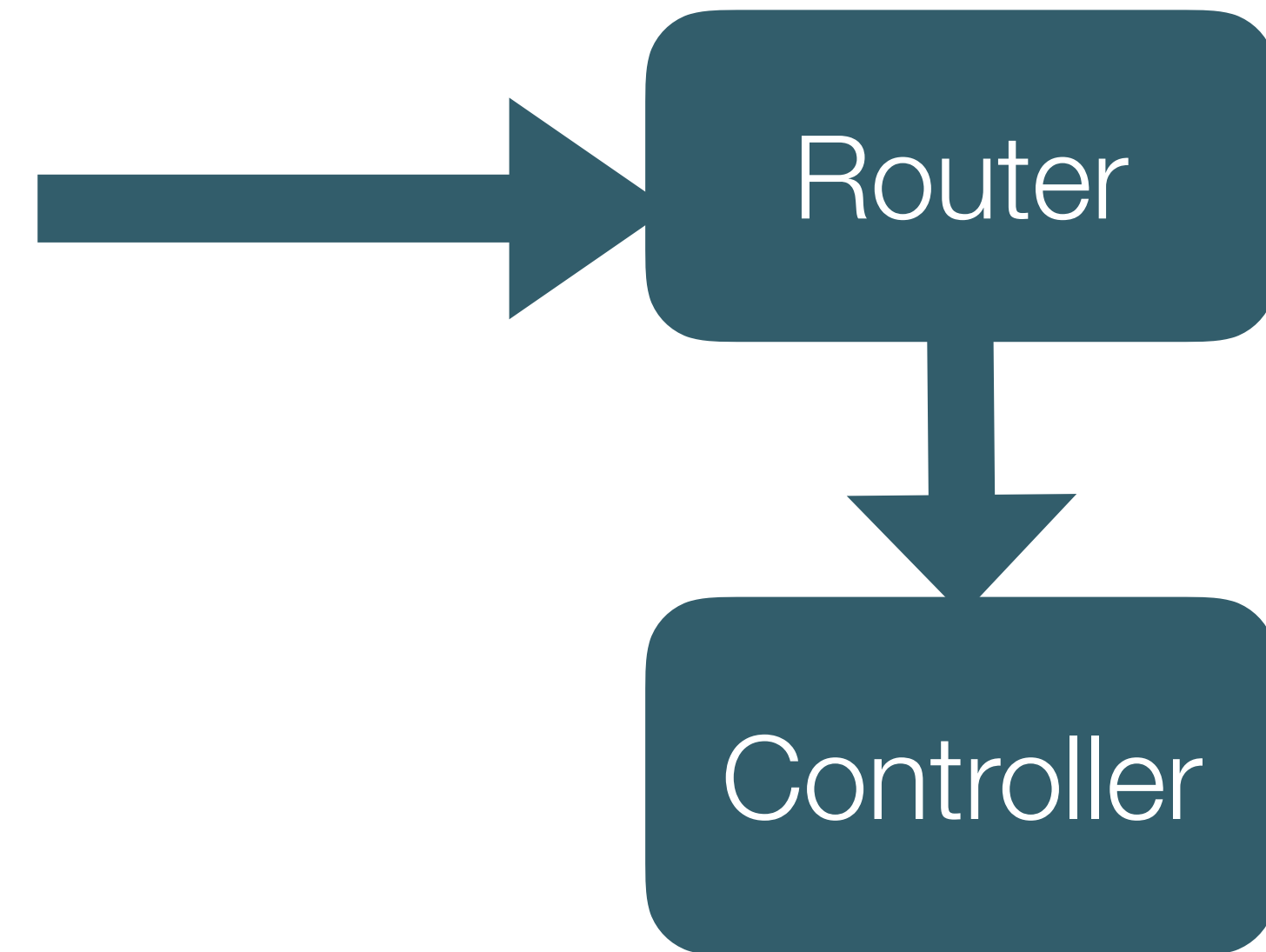
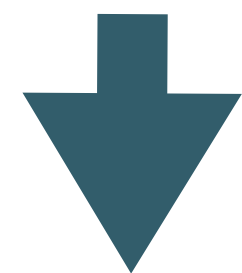
dashboard.index

/'about'



about.index

...then call the corresponding controller methods

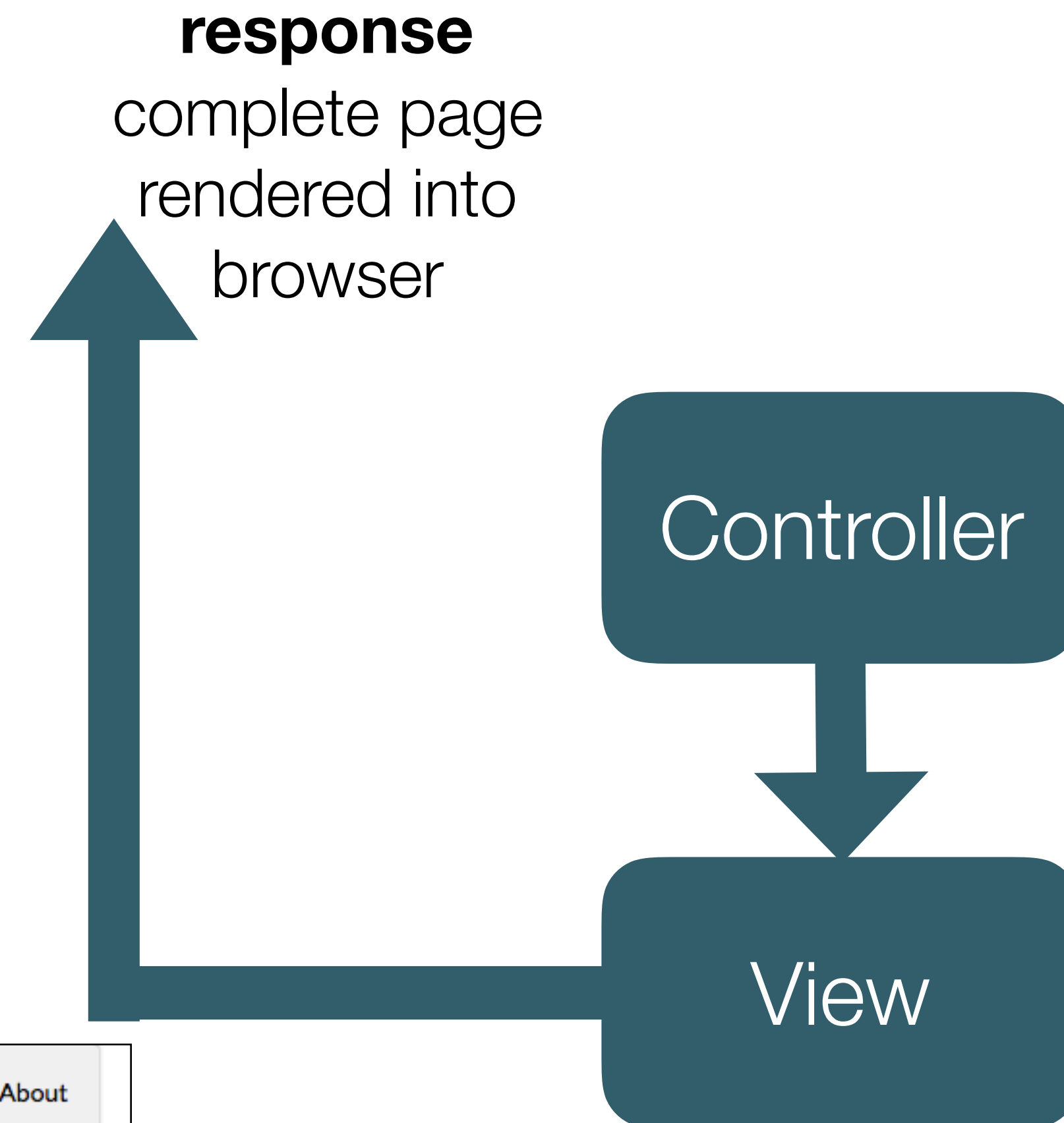
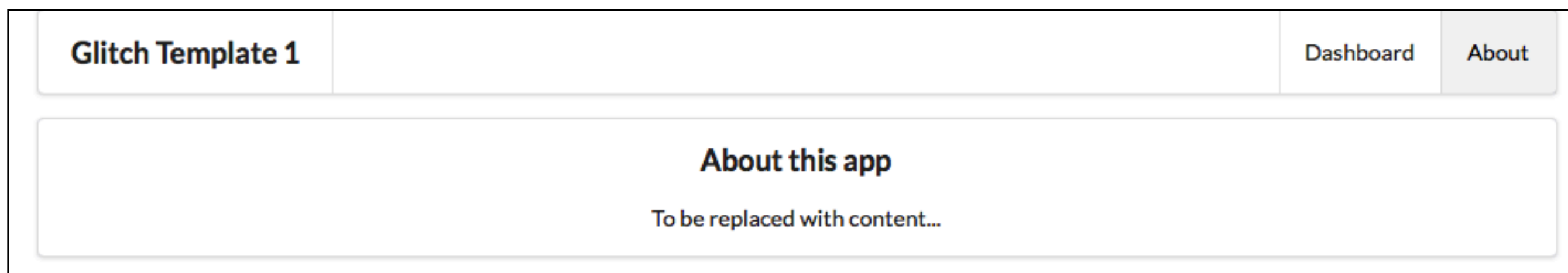


Controller method invoked to handle request

about.js

```
const about = {  
  index(request, response) {  
    logger.info('about rendering');  
    const viewData = {  
      title: 'About Template 1',  
    };  
    response.render('about', viewData);  
  },  
};
```

The About controller



The 'About' controller object -

index method parameters

- Has a single method - index, which has 2 parameters:
- **request** : object containing details of the user request
- **response**: object to be used to send response back to browser

```
const about = {  
  index(request, response) {  
    logger.info('about rendering');  
  
    const viewData = {  
      title: 'About Template 1',  
    };  
  
    response.render('about', viewData);  
  },  
};
```

The 'About' controller **index function body**

logs a message to the console (gomix console, not chrome console)

Create an object called **viewData**, containing a single property: **title**

```
const about = {  
  index(request, response) {  
    logger.info('about rendering');  
    const viewData = {  
      title: 'About Template 1',  
    };  
    response.render('about', viewData);  
  },  
};
```

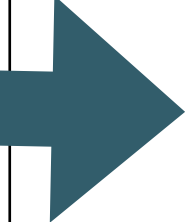
Data sent from controller to view to construct response

Calls **render** method on **response** with 2 parameters:

name of view to render
(about)

object to inject into the view prior to rendering it
(viewData)

```
const about = {  
  index(request, response) {  
    logger.info('about rendering');  
    const viewData = {  
      title: 'About Template 1',  
    };  
    response.render('about', viewData);  
  },  
};
```



The About Controller - Complete

strict mode
javascript for safety

import the **logger**
so we can use it

Export the **about**
object to it can be
used by the router

```
'use strict';  
  
const logger = require('../utils/logger');  
  
const about = {  
  index(request, response) {  
    logger.info('about rendering');  
    const viewData = {  
      title: 'About Template 1',  
    };  
    response.render('about', viewData);  
  },  
};  
  
module.exports = about;
```

about.js

Back-end +
Front-End

```
'use strict';

const logger = require('../utils/logger');

const about = {
  index(request, response) {
    logger.info('about rendering');
    const viewData = {
      title: 'About Template 1',
    };
    response.render('about', viewData);
  },
};

module.exports = about;
```

about.hbs

```
{{> menu id="about"}}

<section class="ui center aligned middle aligned segment">
  <p>
    About this app
  </p>
</section>
```

menu.hbs

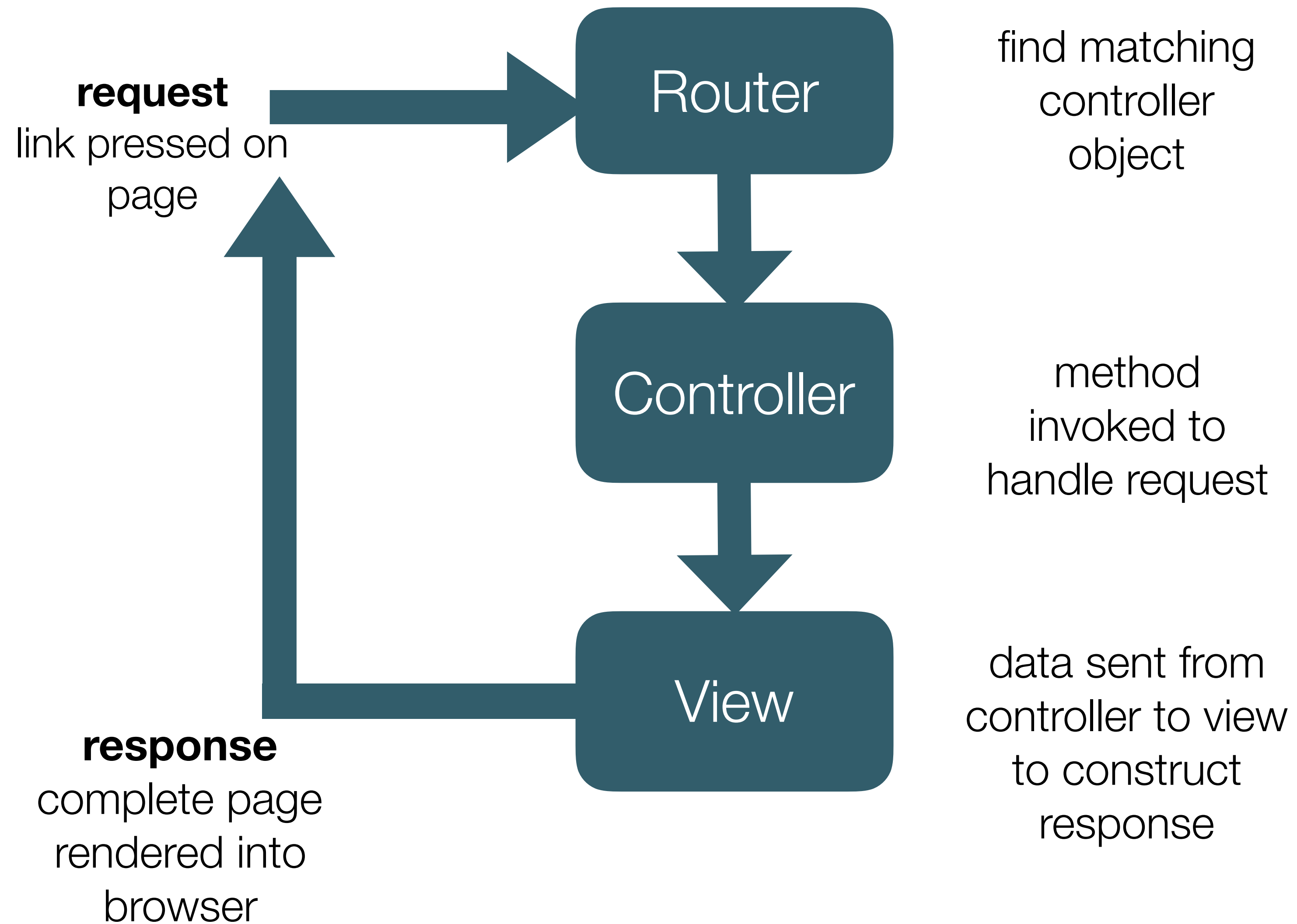
main.hbs

```
<nav class="ui menu">
  <header class="ui header item"> <a href="/"> Template 1 </a></header>
  <div class="right menu">
    <a id="dashboard" class="item" href="/dashboard"> Dashboard </a>
    <a id="about" class="item" href="/about"> About </a>
  </div>
</nav>

<script>
  $("#{{id}}").addClass("active item");
</script>
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> {{title}} </title>
    <meta charset="UTF-8">
    <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.2.8/semantic.min.css">
    <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.2.8/semantic.min.js"></script>
    <link rel="stylesheet" type="text/css" href="/stylesheets/style.css">
  </head>
  <body>
    <section class="ui container">
      {{{body}}}
    </section>
  </body>
</html>
```

Router/Controller/View



Dashboard Controller

```
'use strict';

const logger = require('../utils/logger');

const dashboard = {

  index(request, response) {

    logger.info('dashboard rendering');
    const viewData = {
      title: 'Template 1 Dashboard',
    };
    response.render('dashboard', viewData);

  },

};

module.exports = dashboard;
```

Application Structure

2 Controllers

which will render

2 matching views

```
📁 assets
  controllers/about.js
  controllers/dashboard.js
  utils/logger.js
  views/layouts/main.hbs
  views/partials/mainpanel.hbs
  views/partials/menu.hbs
  views/about.hbs
  views/dashboard.hbs
🔑 .env
.gitignore
README.md
package.json
routes.js
server.js
```

dashboard.js

```
'use strict';

const logger = require('../utils/logger');

const dashboard = {
  index(request, response) {
    logger.info('dashboard rendering');
    const viewData = {
      title: 'Template 1 Dashboard',
    };
    response.render('dashboard', viewData);
  },
};

module.exports = dashboard;
```

routes.js

```
...
router.get('/dashboard', dashboard.index);
router.get('/about', about.index);
...
```

about.js

```
'use strict';

const logger = require('../utils/logger');

const about = {
  index(request, response) {
    logger.info('about rendering');
    const viewData = {
      title: 'About Template 1',
    };
    response.render('about', viewData);
  },
};

module.exports = about;
```

dashboard.js

```
'use strict';  
  
const logger = require('../utils/logger');  
  
const dashboard = {  
  index(request, response) {  
    logger.info('dashboard rendering');  
    const viewData = {  
      title: 'Template 1 Dashboard',  
    };  
    response.render('dashboard', viewData);  
  },  
};  
  
module.exports = dashboard;
```

controllers/views

dashboard.hbs

```
{{> menu id="dashboard"}}  
  
<section class="ui segment">  
  {{> mainpanel}}  
</section>
```

about.js

```
'use strict';  
  
const logger = require('../utils/logger');  
  
const about = {  
  index(request, response) {  
    logger.info('about rendering');  
    const viewData = {  
      title: 'About Template 1',  
    };  
    response.render('about', viewData);  
  },  
};  
  
module.exports = about;
```

about.hbs

```
{{> menu id="about"}}  
  
<section class="ui center aligned middle aligned segment">  
  <p>  
    About this app  
  </p>  
</section>
```