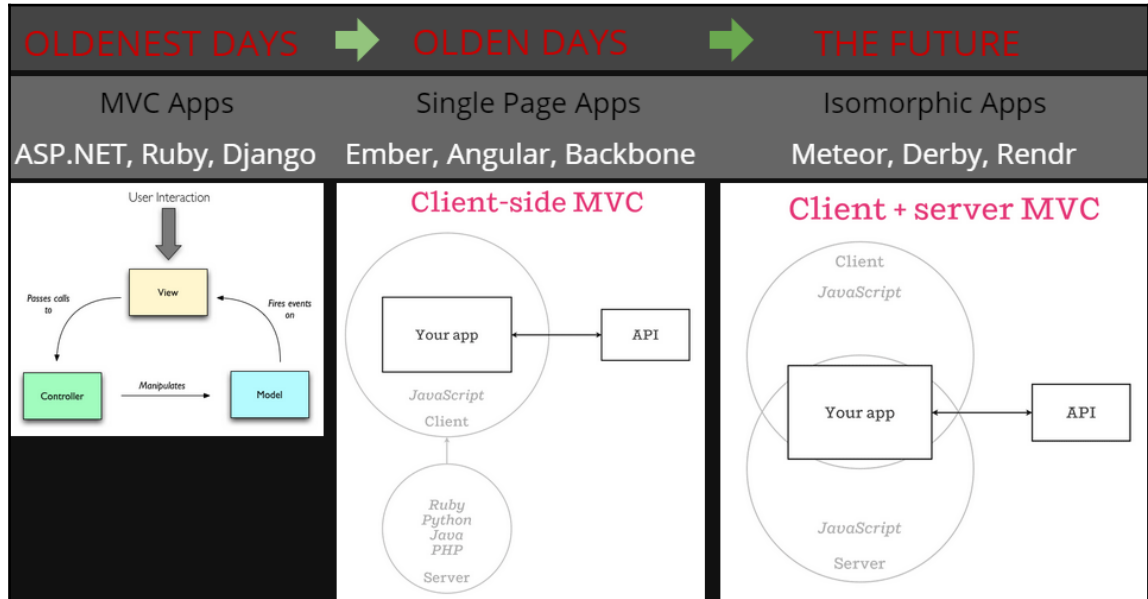
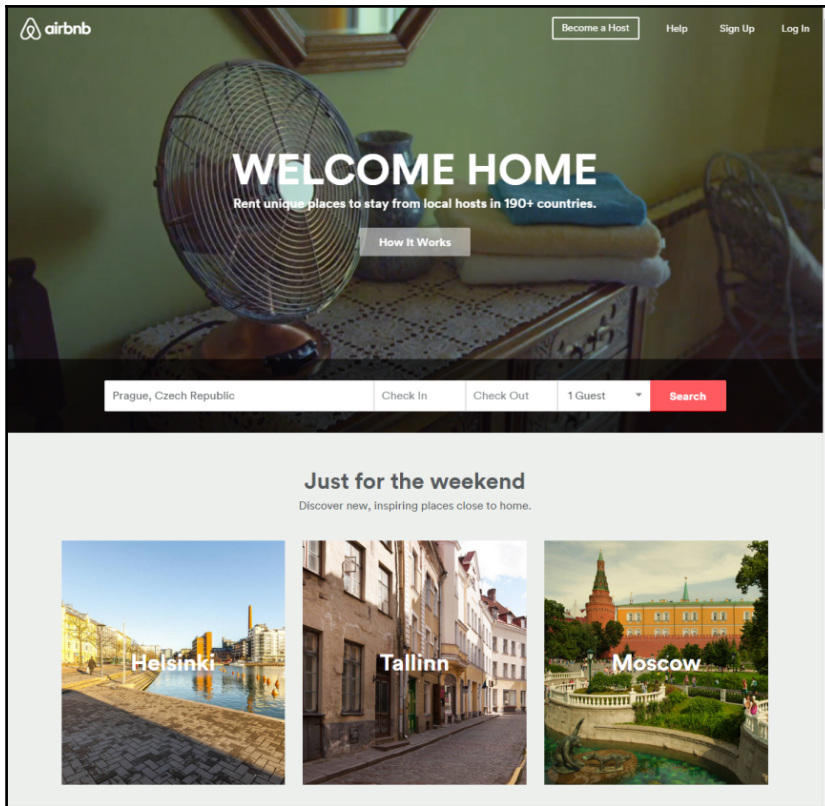


Chapter 1: Getting Started with Isomorphic Web Apps

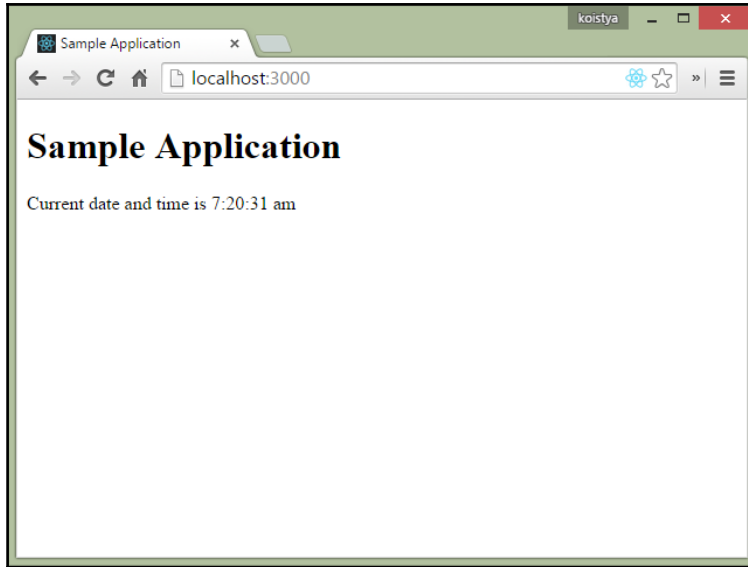


 **Konstantin** @koistya · Jan 15
Hello, world!
...

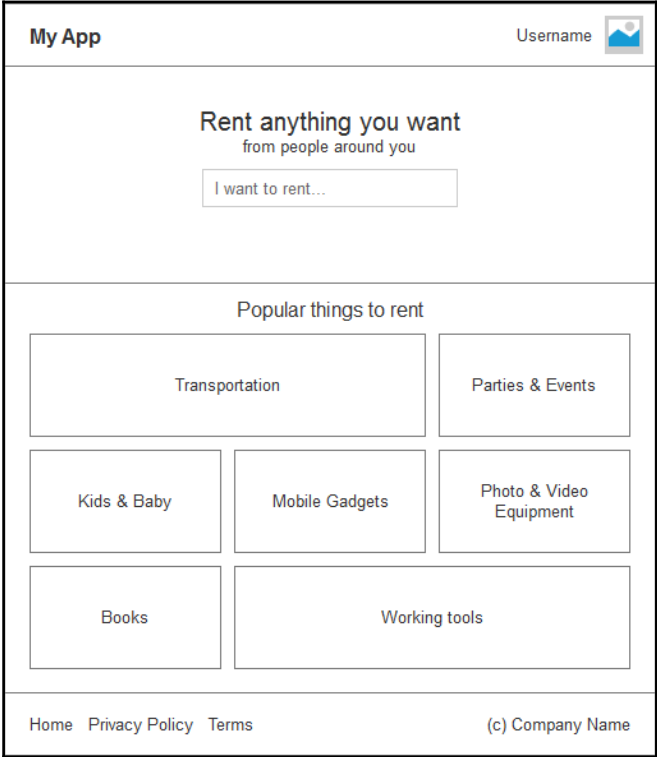



```
npm
> babel-node tools/run serve

[01:19:26] Starting 'serve'...
[01:19:26] Starting 'build'...
[01:19:26] Starting 'clean'...
[01:19:26] Finished 'clean' after 248 ms
[01:19:26] Starting 'copy'...
[01:19:26] Finished 'copy' after 39 ms
[01:19:26] Starting 'bundle'...
Hash: 5b1d7889daf2c795cd33
Version: webpack 1.12.2
Time: 744ms
  Asset      Size  Chunks             Chunk Names
server.js    8.69 kB          0 [emitted]   main
Hash: 8ba90a717eb0a14b6dc6
Version: webpack 1.12.2
Time: 2768ms
  Asset      Size  Chunks             Chunk Names
app.js      1.07 MB          0 [emitted]   main
[01:19:29] Finished 'bundle' after 3016 ms
[01:19:29] Finished 'build' after 3305 ms
Node.js server is listening at http://localhost:3000/
[01:19:32] Finished 'serve' after 6069 ms
```

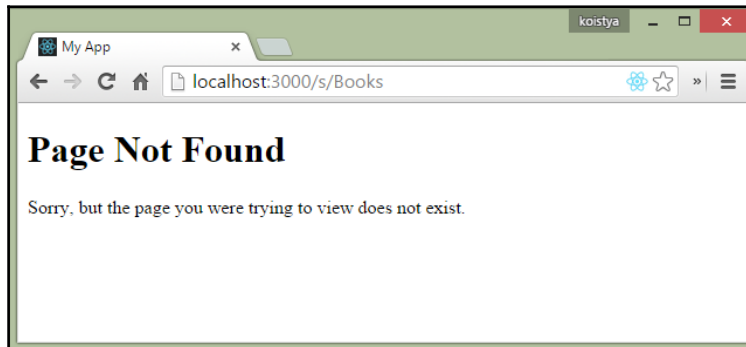


Chapter 2: Creating a Web UI with React

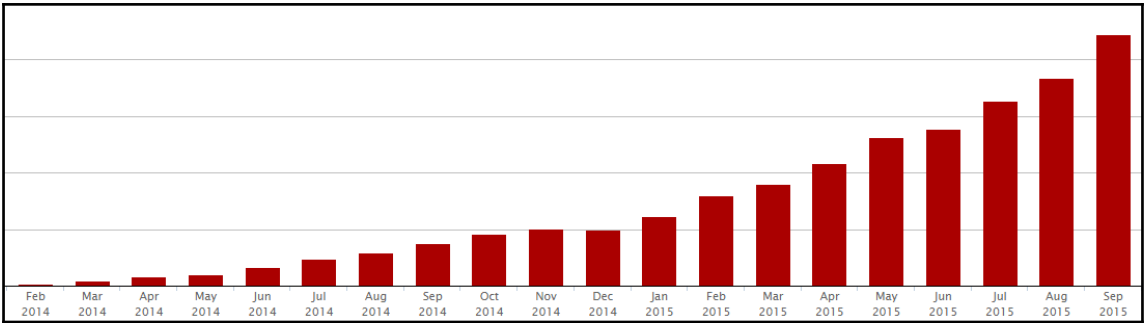
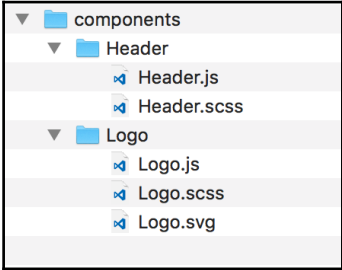



My App Username 

| Pick-up date | Return date | Map |
|-----------------------|-------------------------|-----|
| item name \$15/day | item name \$12/day | |
| item name \$2/day | item name \$300/week | |
| item name \$15/day | item name \$12/day | |
| item name \$2/day | item name \$300/week | |
| | | |
| | | |
| | | |



Chapter 3: Working with CSS and Media Assets

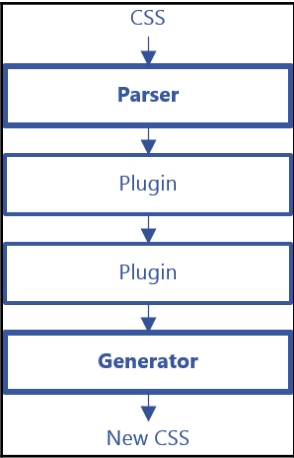


 **Mark Otto**
@mdo [Follow](#)

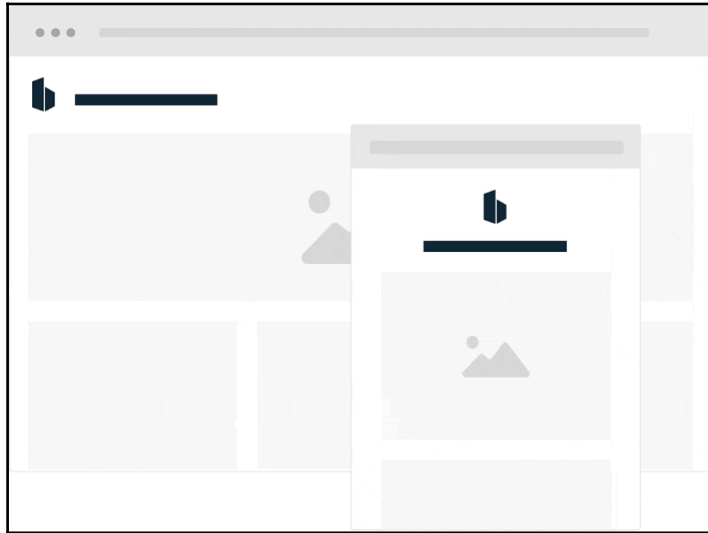
Oh, btw—Bootstrap 4 will be in SCSS. And if you care, v5 will likely be in PostCSS because holy crap that sounds cool.

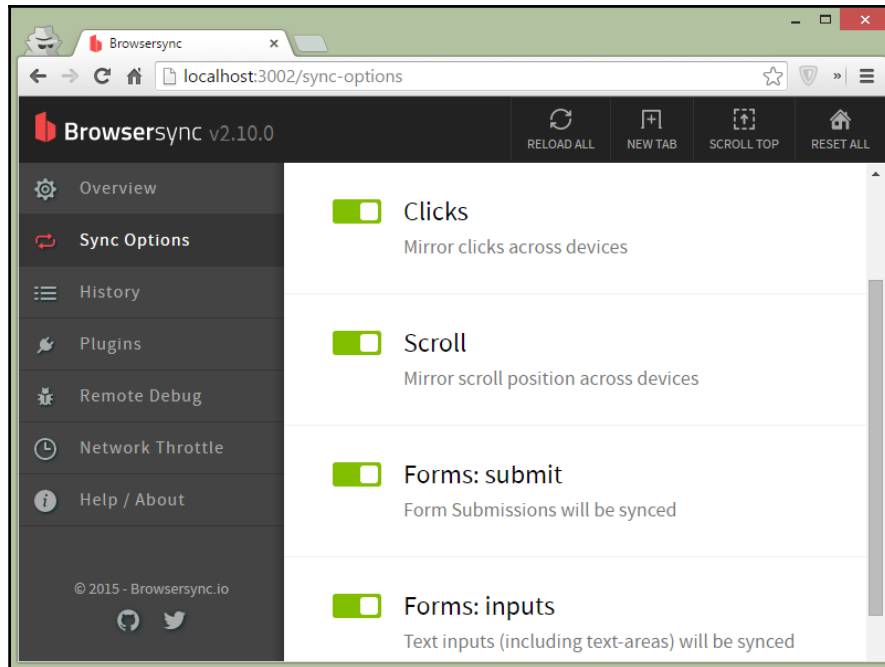
1:13 AM - 24 Apr 2015

↩️ ↻️ 1,218 ❤️ 1,026

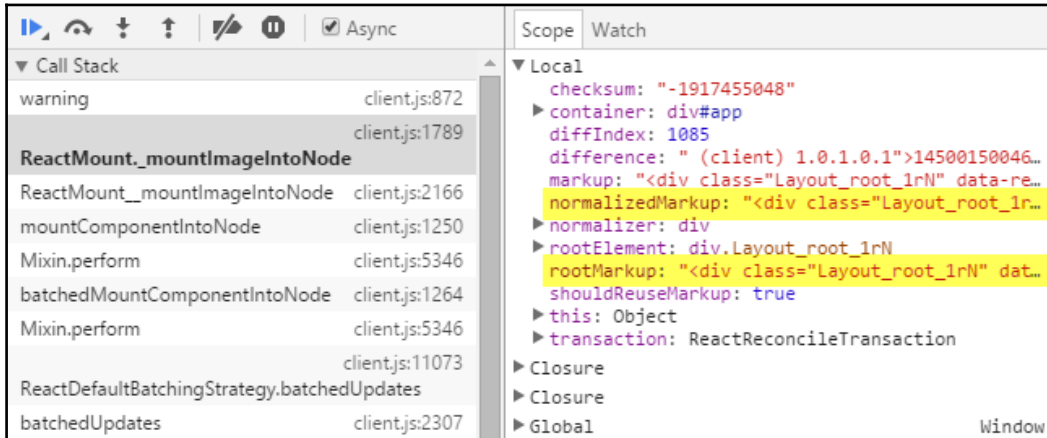


Chapter 4: Working with Browsersync and Hot Module Replacement





Chapter 5: Rendering React Components on the Server

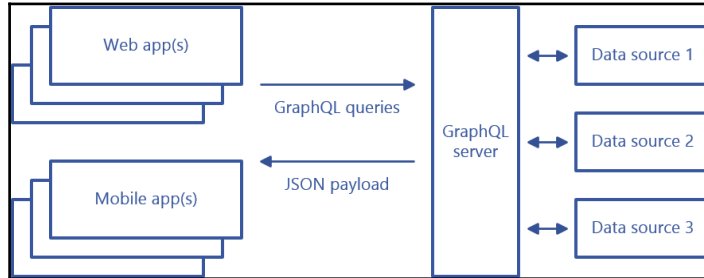


The image shows a browser's developer console with the 'Call Stack' and 'Local' scopes expanded. The 'Call Stack' shows a sequence of function calls, with 'ReactMount._mountImageIntoNode' highlighted. The 'Local' scope shows the following variables:

- checksum: "-1917455048"
- container: div#app
- diffIndex: 1085
- difference: " (client) 1.0.1.0.1">14500150046..
- markup: "<div class="Layout_root_1rN" data-re...
- normalizedMarkup: "<div class="Layout_root_1r...
- normalizer: div
- rootElement: div.Layout_root_1rN
- rootMarkup: "<div class="Layout_root_1rN" dat...
- shouldReuseMarkup: true
- this: Object
- transaction: ReactReconcileTransaction

The 'Global' scope is also visible, showing 'Window'.

Chapter 6: Creating Data API with GraphQL



The screenshot shows the GraphQL Playground interface. The query editor contains the following query:

```
1 query {
2   person (id: "cGVvcGx1OjE=") {
3     name,
4     gender
5   }
6   height
```

A tooltip is displayed over the `gender` field, containing the text: "String The gender of this person. Either "Male", "Female" or "unknown", "n/a" if the person does not have a gender."

The right sidebar shows the schema for the `Root` type, listing various fields such as `allFilms`, `film`, `allPeople`, `person`, `allPlanets`, and `planet`.

The screenshot shows the GraphQL Playground interface with the raw JSON response displayed. The query is: `query { person(id: "cGVvcGx1OjE=") { name } }`. The response is:

```
{
  "data": {
    "person": {
      "name": "Luke Skywalker",
      "gender": "male"
    }
  }
}
```

The interface includes a "Raw" button and a "Parsed" button to toggle between the raw JSON and the parsed JSON.

Developer Tools - http://graphql-swapi.parseapp.com/

Elements Console Sources Network Timeline Profiles Resources Security Audits PageSpeed >>

top Preserve log

```
> fetch('http://graphql-swapi.parseapp.com/?query=query{person(id:"cGVvcGx10jE="){name,gender}}').then(resp =>
  resp.json()).then(data => console.log(JSON.stringify(data)))
< Promise {[[PromiseStatus]]: "pending", [[PromiseValue]]: undefined}
{"data":{"person":{"name":"Luke Skywalker","gender":"male"}}} VM197:2
>
```

graphql-swapi.parseapp.com

graphql-swapi.parseapp.com/?query=query%20PersonQuery(%24id%3A%20ID)%

GraphQL < Docs

```
1 query PersonQuery($id: ID) {
2   person(id: $id) {
3     name,
4     gender,
5     homeworld {
6       name
7     }
8   }
9 }
```

QUERY VARIABLES

```
1 {
2   "id": "cGVvcGx10jE="
3 }
```

```
{
  "data": {
    "person": {
      "name": "Luke Skywalker",
      "gender": "male",
      "homeworld": {
        "name": "Tatooine"
      }
    }
  }
}
```

graphql-swapi.parseapp.com

graphql-swapi.parseapp.com/?query=%7B%0A%20%20person(id%3A%20%22cGVvc%20%22%7D

GraphiQL

Prettify Docs

```
1 {
2   person(id: "cGVvcGx10jE=") {
3     name,
4     ...PersonInfo,
5     ...PersonHomeworld
6   }
7 }
8
9 fragment PersonInfo on Person {
10  gender,
11  birthYear
12 }
13
14 fragment PersonHomeworld on Person {
15  homeworld {
16    name
17  }
18 }
```

```
{
  "data": {
    "person": {
      "name": "Luke Skywalker",
      "gender": "male",
      "birthYear": "1989",
      "homeworld": {
        "name": "Tatooine"
      }
    }
  }
}
```

QUERY VARIABLES

localhost:3000/graphql?query=query%20%7B%0A%20%20greeting(n)%20%7D

GraphiQL

Schema Query

```
1 query {
2   greeting(n)
3 }
```

```
{
  "data": {
    "greeting": "Hello, Tarkus!"
  }
}
```

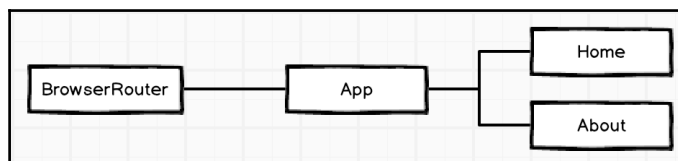
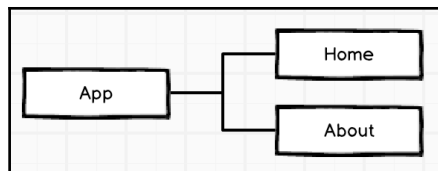
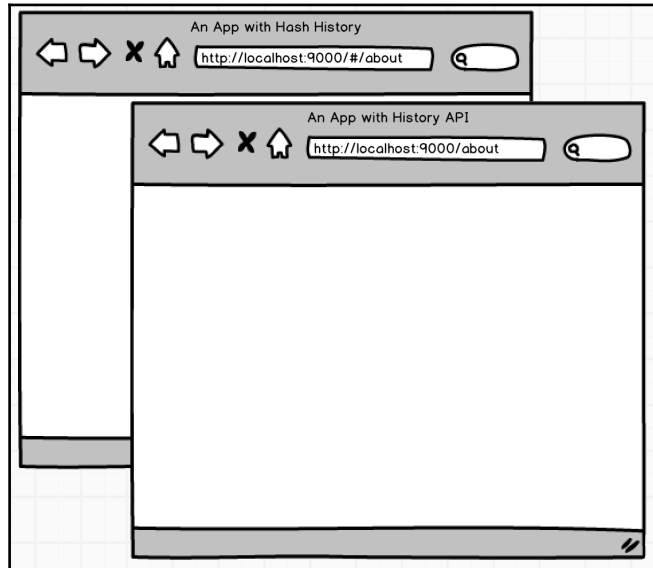
No Description

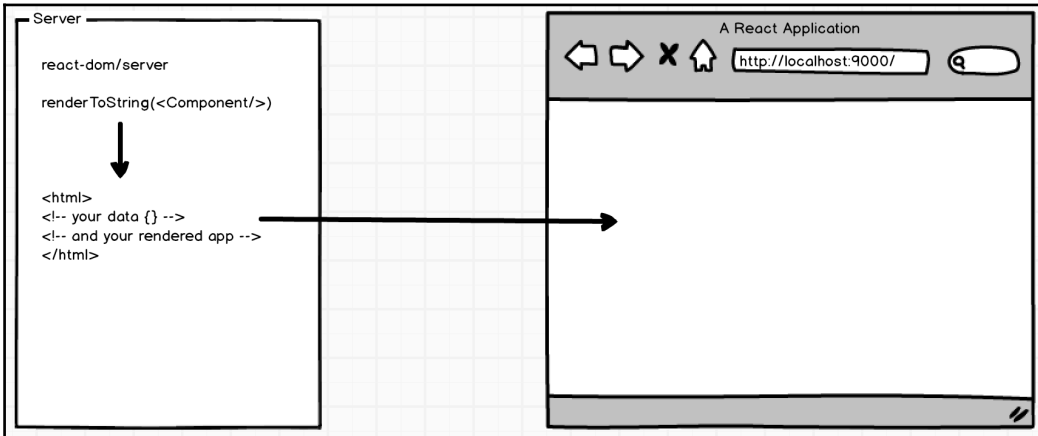
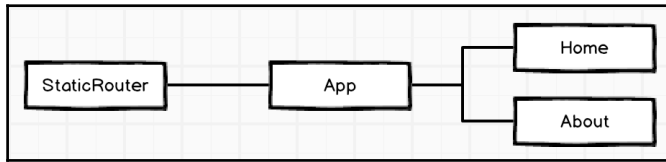
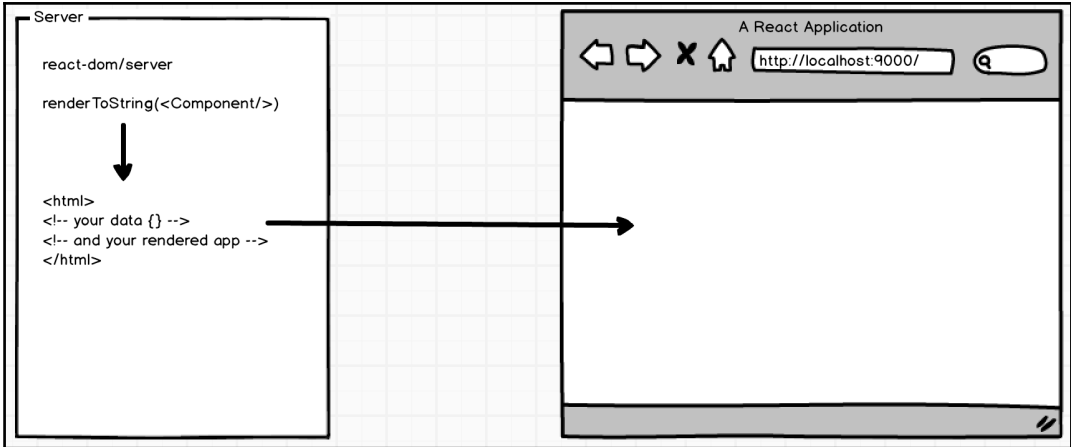
FIELDS

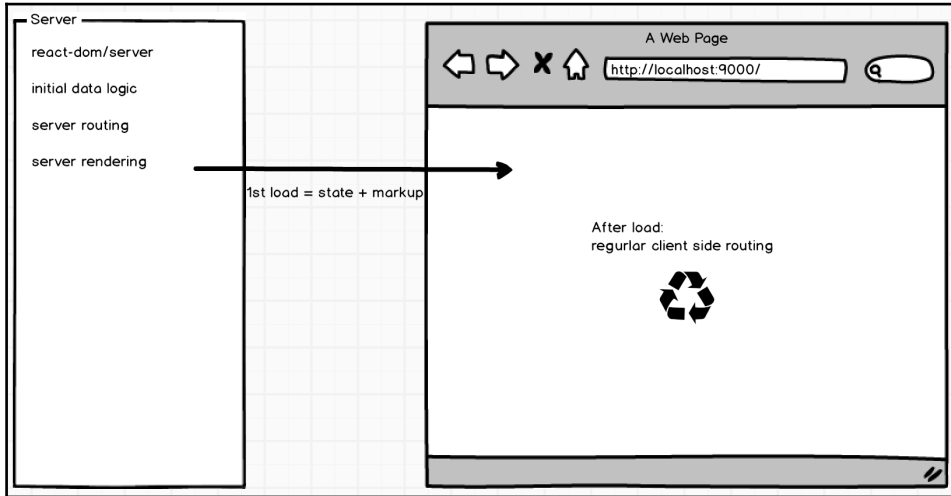
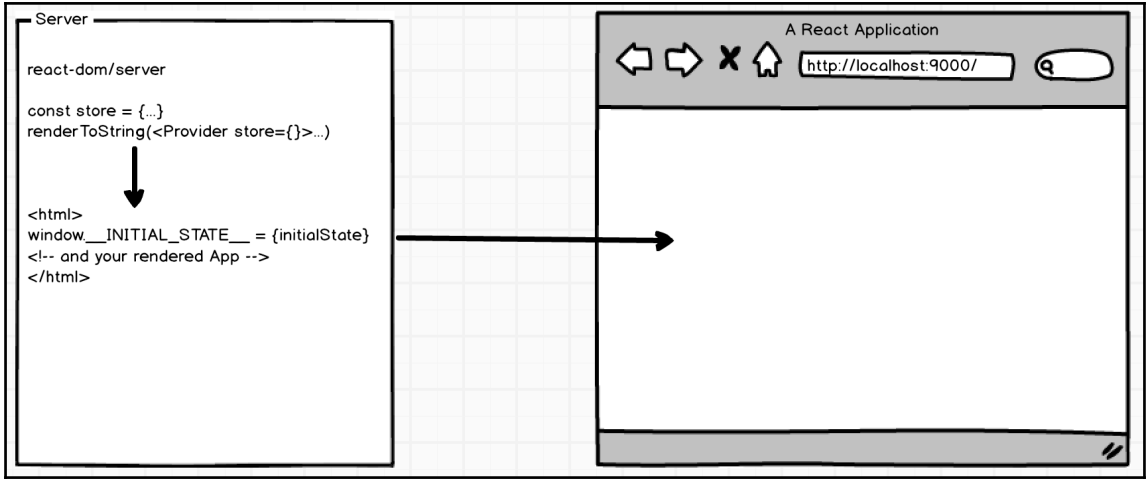
`greeting(name: String): String`

QUERY VARIABLES

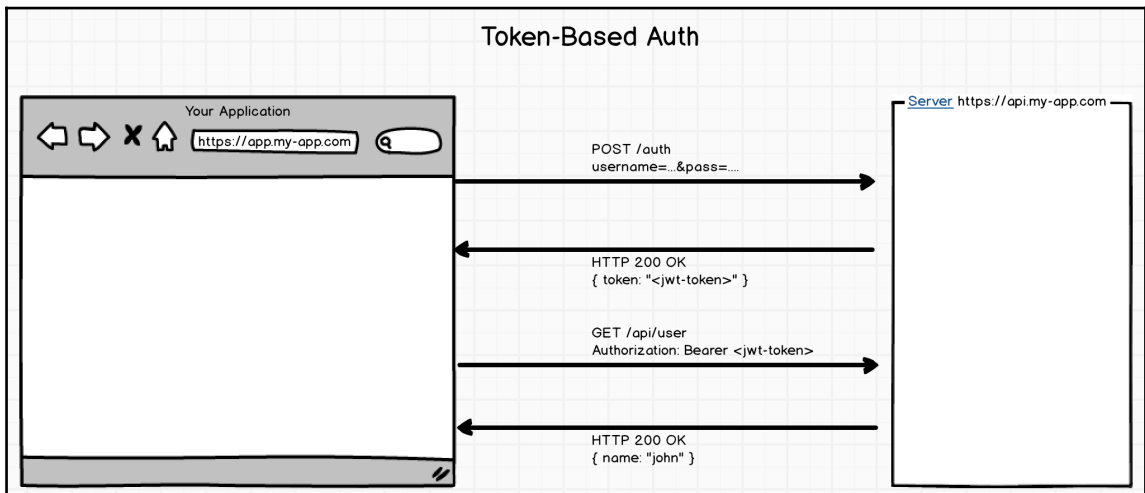
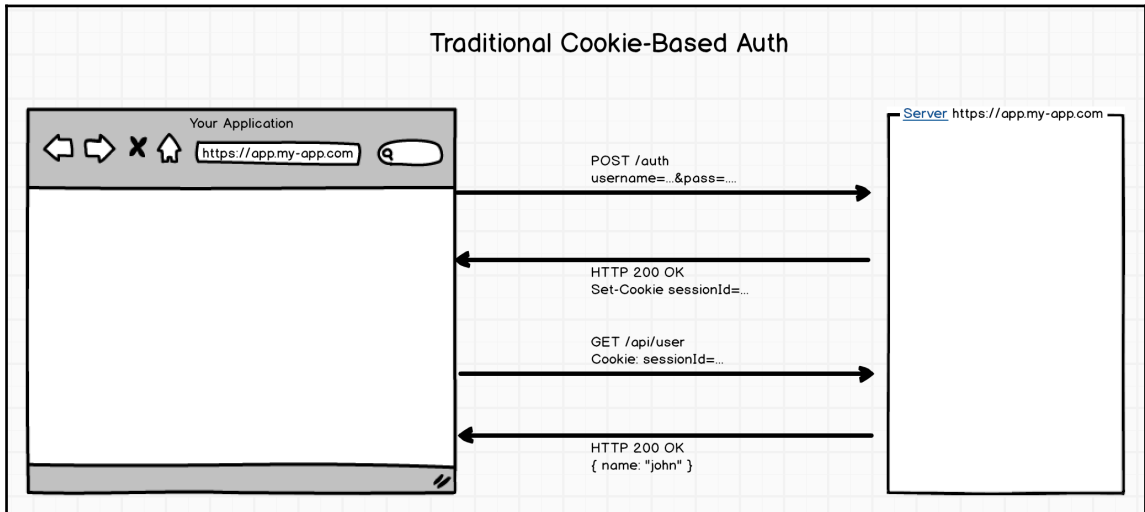
Chapter 7: Implementing Routing and Navigation

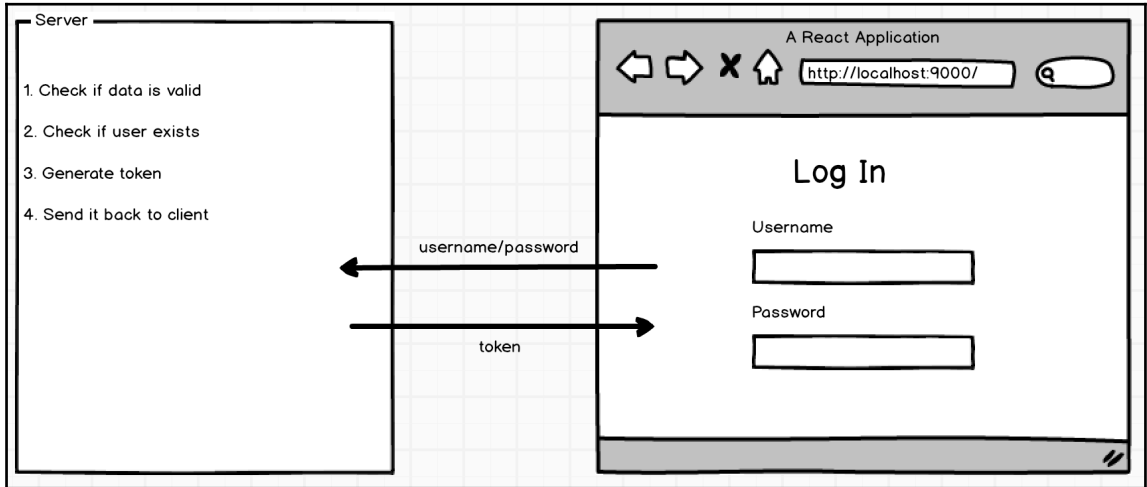
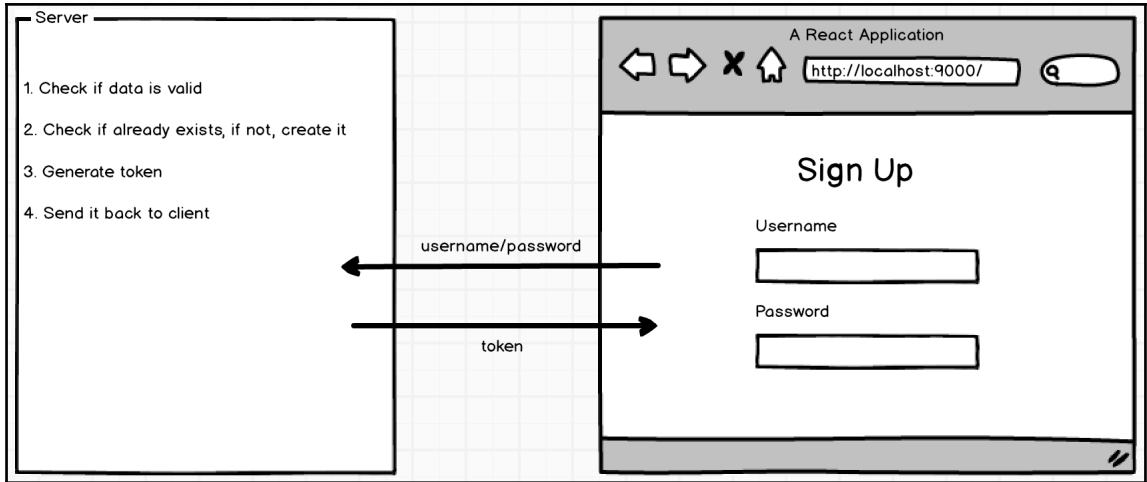


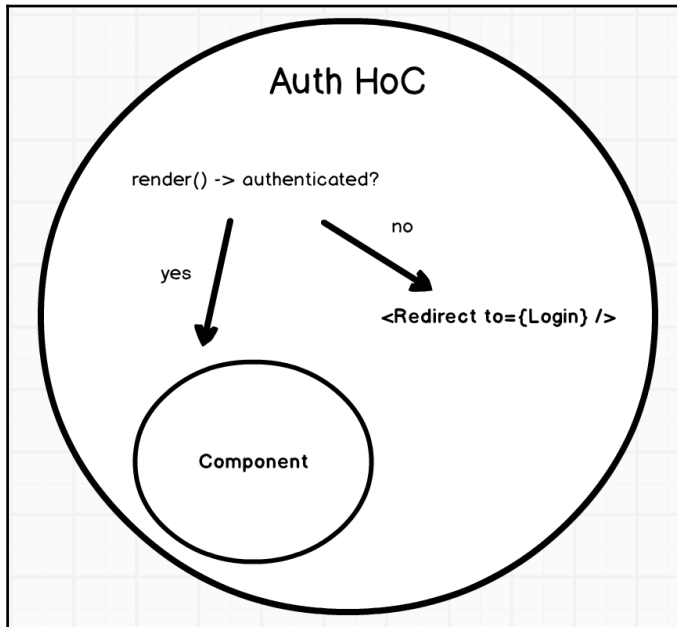
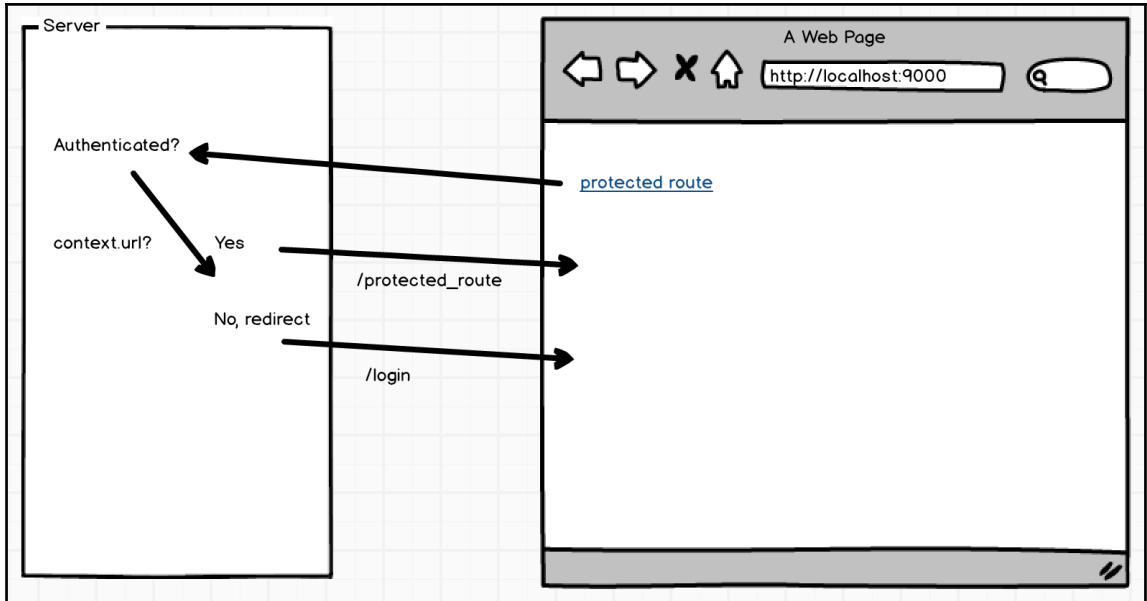


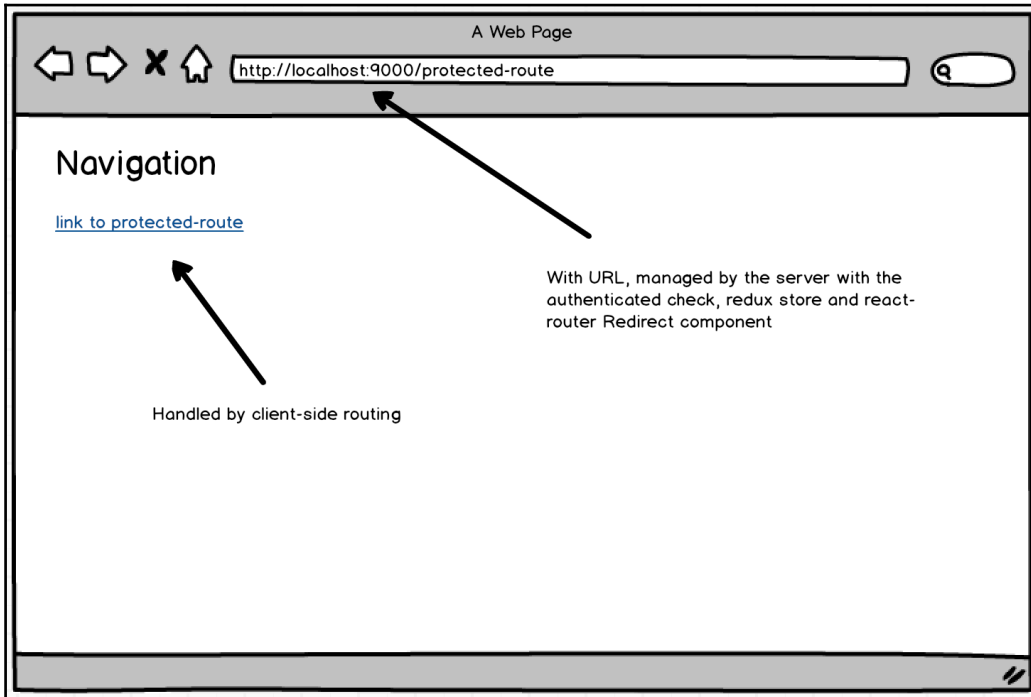
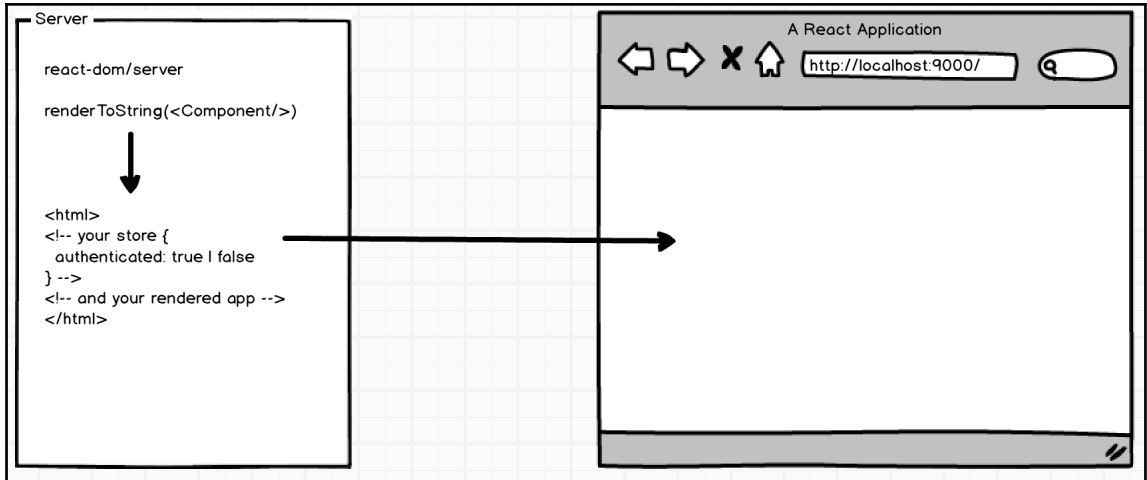


Chapter 8: Authentication and Authorization

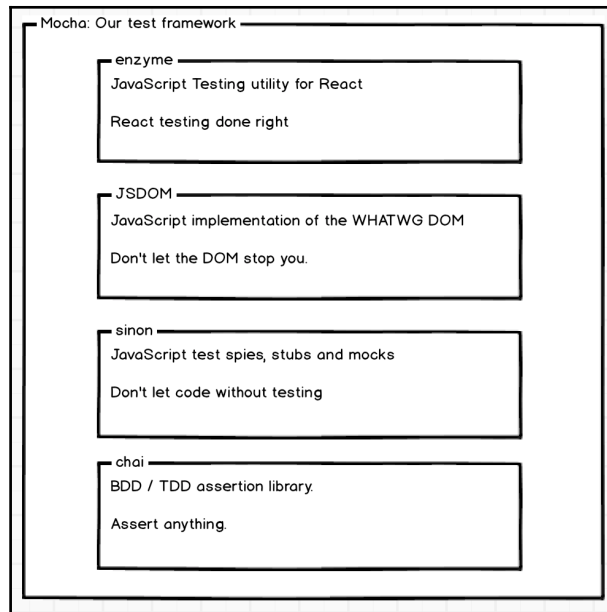
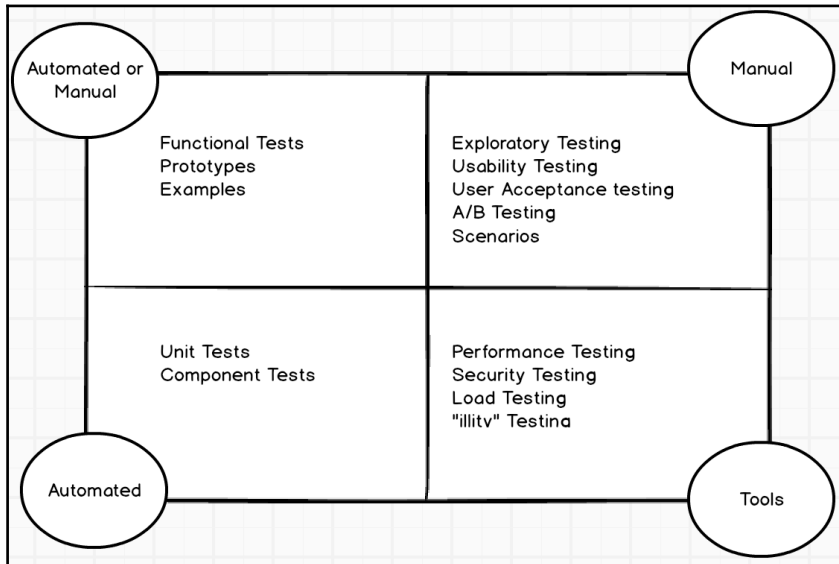








Chapter 9: Testing and Deploying Your App



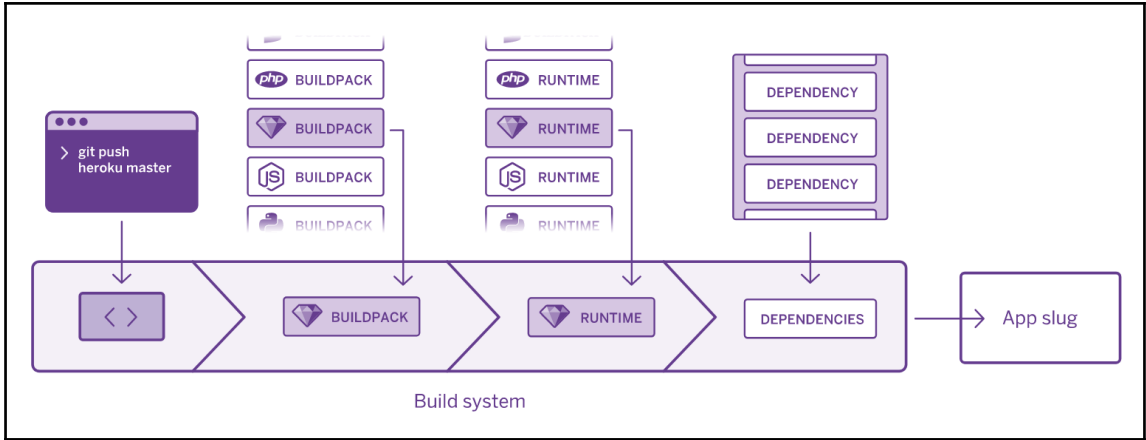


Table of Contents

Index

2

Index