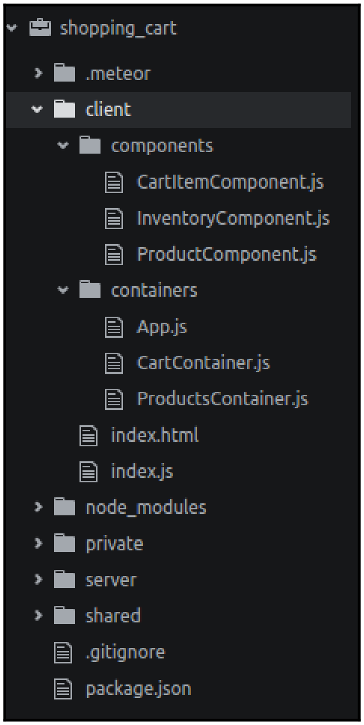
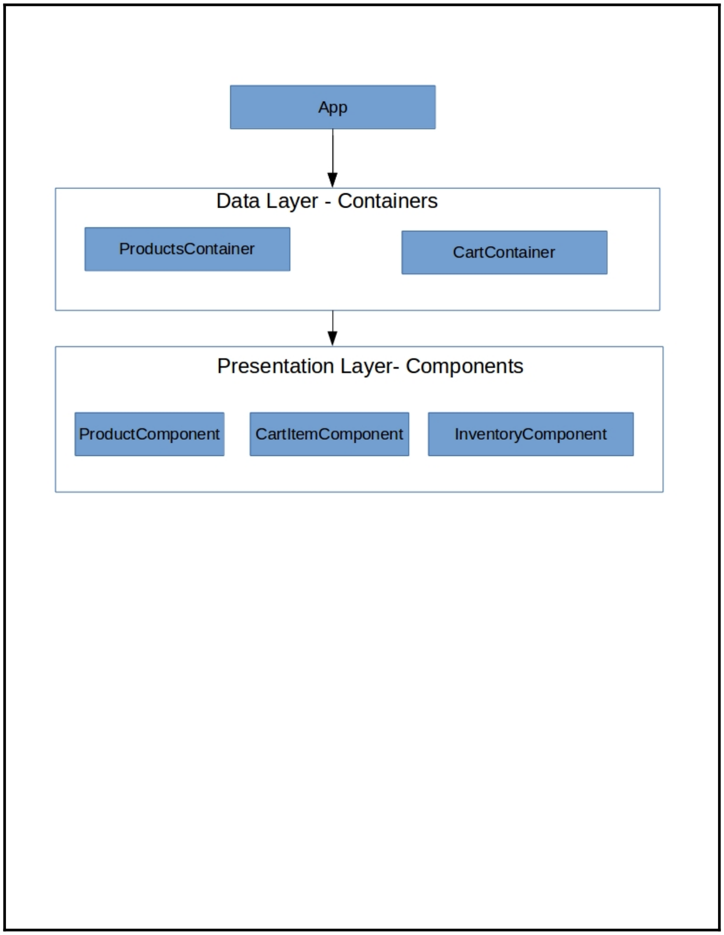
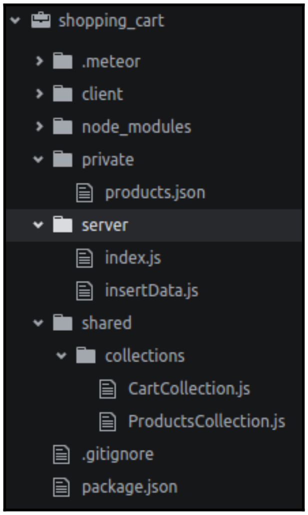
Chapter 2: Building a Shopping Cart

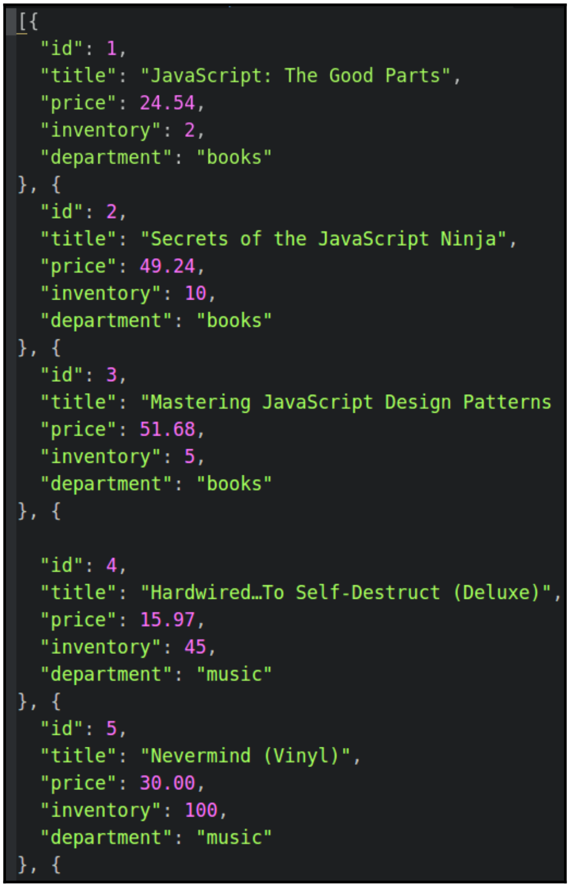


```
shopping_cart
  .meteor
  client
    components
      CartItemComponent.js
      InventoryComponent.js
      ProductComponent.js
    containers
      App.js
      CartContainer.js
      ProductsContainer.js
    index.html
    index.js
  node_modules
  private
  server
  shared
  .gitignore
  package.json
```

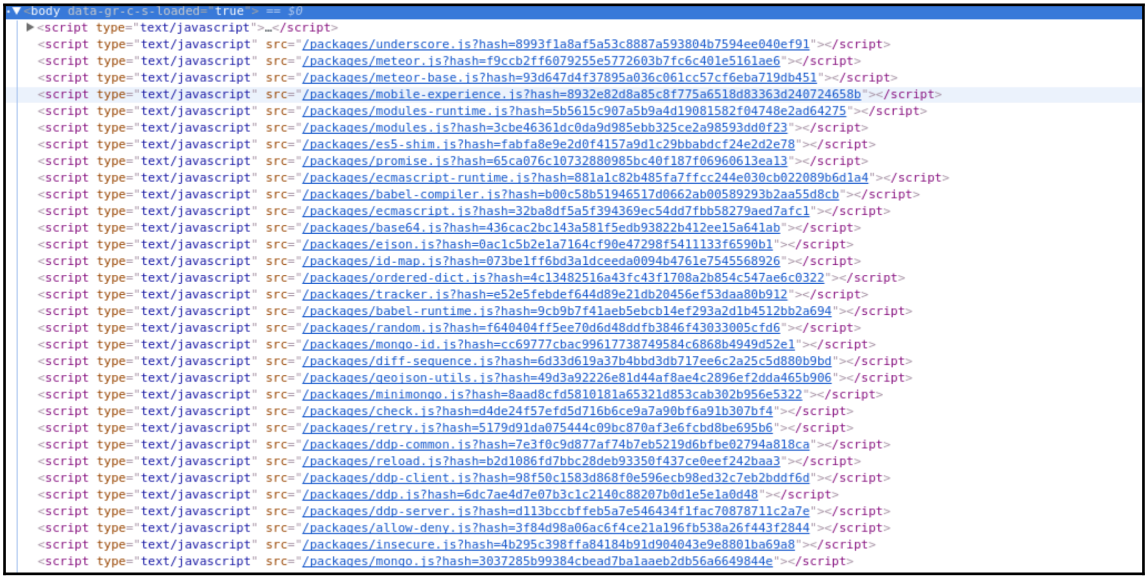

App
Data Layer - Containers
ProductsContainer
CartContainer
Presentation Layer- Components
ProductComponent
CartItemComponent
InventoryComponent



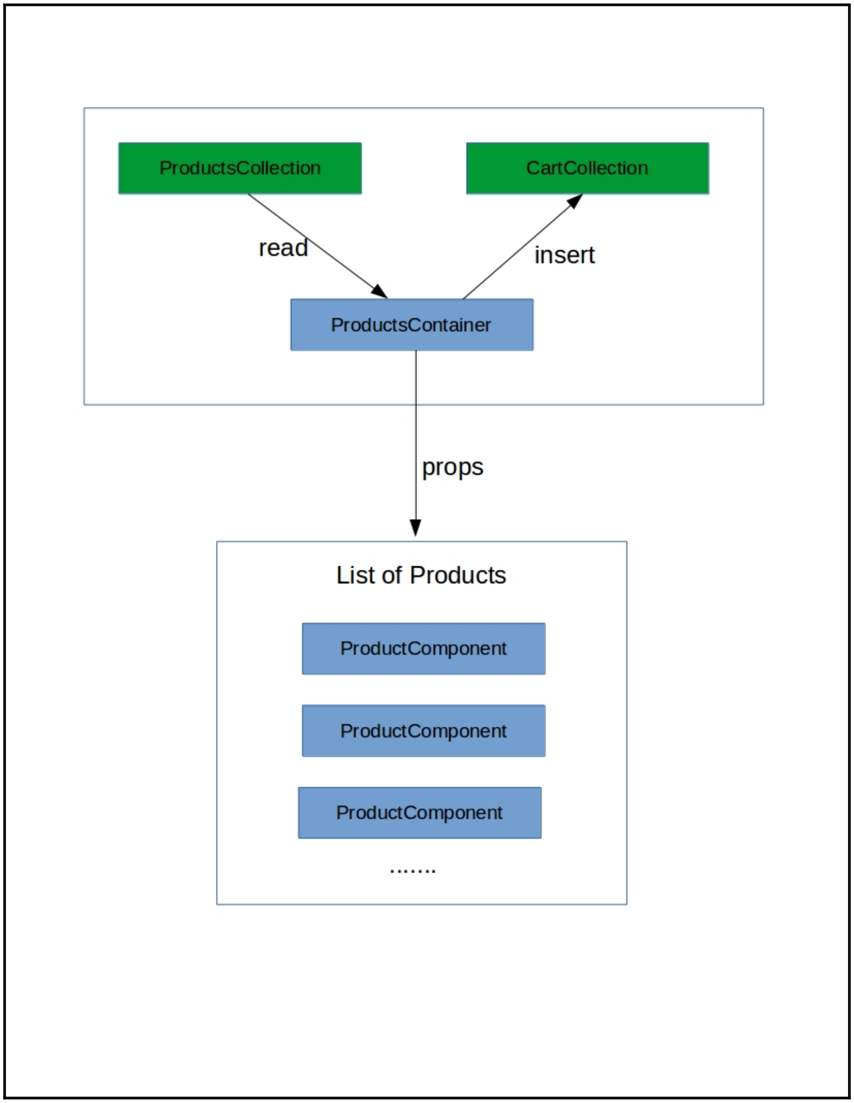
```
shopping_cart
  .meteor
  client
  node_modules
  private
    products.json
  server
    index.js
    insertData.js
  shared
    collections
      CartCollection.js
      ProductsCollection.js
  .gitignore
  package.json
```

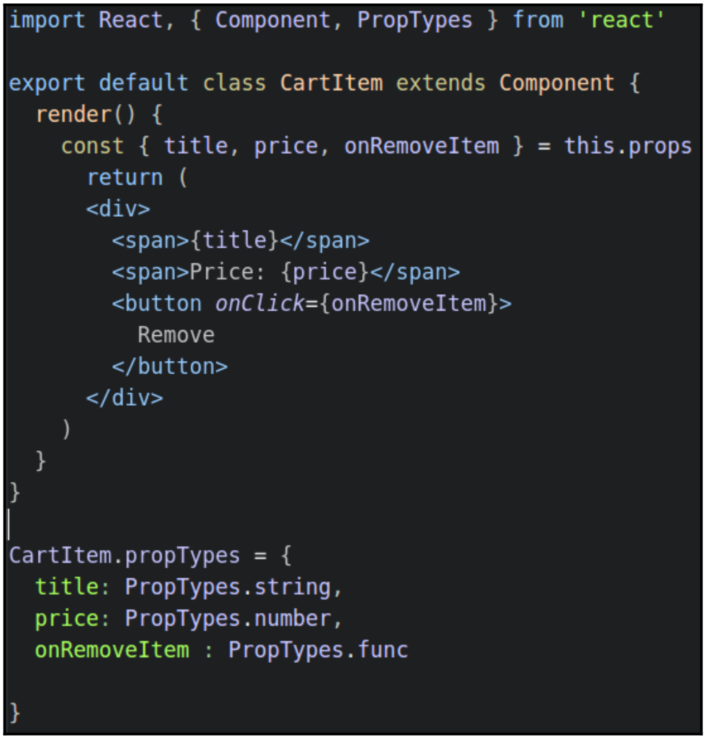


```
[{
  "id": 1,
  "title": "JavaScript: The Good Parts",
  "price": 24.54,
  "inventory": 2,
  "department": "books"
}, {
  "id": 2,
  "title": "Secrets of the JavaScript Ninja",
  "price": 49.24,
  "inventory": 10,
  "department": "books"
}, {
  "id": 3,
  "title": "Mastering JavaScript Design Patterns
  "price": 51.68,
  "inventory": 5,
  "department": "books"
}, {

  "id": 4,
  "title": "Hardwired…To Self-Destruct (Deluxe)",
  "price": 15.97,
  "inventory": 45,
  "department": "music"
}, {
  "id": 5,
  "title": "Nevermind (Vinyl)",
  "price": 30.00,
  "inventory": 100,
  "department": "music"
}, {
```

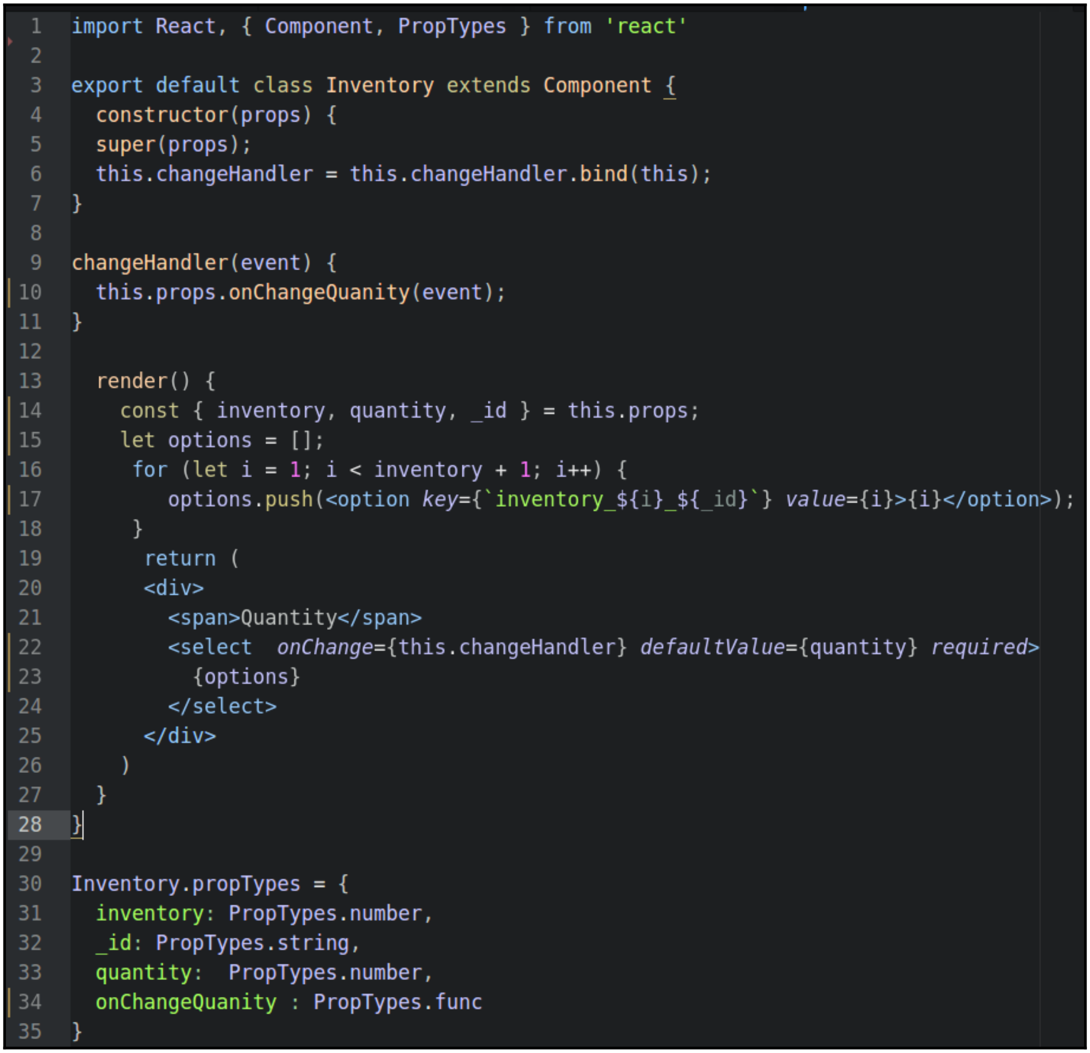


```
▼<body data-gr-c-s-loaded="true"> == $0
  ▶<script type="text/javascript">…</script>
   <script type="text/javascript" src="/packages/underscore.js?hash=8993f1a8af5a53c8887a593804b7594ee040ef91"></script>
   <script type="text/javascript" src="/packages/meteor.js?hash=f9ccb2ff6079255e5772603b7fc6c401e5161ae6"></script>
   <script type="text/javascript" src="/packages/meteor-base.js?hash=93d647d4f37895a036c061cc57cf6eba719db451"></script>
   <script type="text/javascript" src="/packages/mobile-experience.js?hash=8932e82d8a85c8f775a6518d83363d240724658b"></script>
   <script type="text/javascript" src="/packages/modules-runtime.js?hash=5b5615c907a5b9a4d19081582f04748e2ad64275"></script>
   <script type="text/javascript" src="/packages/modules.js?hash=3cbe46361dc0da9d985ebb325ce2a98593dd0f23"></script>
   <script type="text/javascript" src="/packages/es5-shim.js?hash=fabfa8e9e2d0f4157a9d1c29bbabdcf24e2d2e78"></script>
   <script type="text/javascript" src="/packages/promise.js?hash=65ca076c10732880985bc40f187f06960613ea13"></script>
   <script type="text/javascript" src="/packages/ecmascript-runtime.js?hash=881a1c82b485fa7ffcc244e030cb022089b6d1a4"></script>
   <script type="text/javascript" src="/packages/babel-compiler.js?hash=b00c58b51946517d0662ab00589293b2aa55d8cb"></script>
   <script type="text/javascript" src="/packages/ecmascript.js?hash=32ba8df5a5f394369ec54dd7fbb58279aed7afc1"></script>
   <script type="text/javascript" src="/packages/base64.js?hash=436cac2bc143a581f5edb93822b412ee15a641ab"></script>
   <script type="text/javascript" src="/packages/ejson.js?hash=0ac1c5b2e1a7164cf90e47298f5411133f6590b1"></script>
   <script type="text/javascript" src="/packages/id-map.js?hash=073be1ff6bd3a1dceeda0094b4761e7545568926"></script>
   <script type="text/javascript" src="/packages/ordered-dict.js?hash=4c13482516a43fc43f1708a2b854c547ae6c0322"></script>
   <script type="text/javascript" src="/packages/tracker.js?hash=e52e5febdef644d89e21db20456ef53daa80b912"></script>
   <script type="text/javascript" src="/packages/babel-runtime.js?hash=9cb9b7f41aeb5ebcb14ef293a2d1b4512bb2a694"></script>
   <script type="text/javascript" src="/packages/random.js?hash=f640404ff5ee70d6d48ddfb3846f43033005cfd6"></script>
   <script type="text/javascript" src="/packages/mongo-id.js?hash=cc69777cbac99617738749584c6868b4949d52e1"></script>
   <script type="text/javascript" src="/packages/diff-sequence.js?hash=6d33d619a37b4bbd3db717ee6c2a25c5d880b9bd"></script>
   <script type="text/javascript" src="/packages/geojson-utils.js?hash=49d3a92226e81d44af8ae4c2896ef2dda465b906"></script>
   <script type="text/javascript" src="/packages/minimongo.js?hash=8aad8cfd5810181a65321d853cab302b956e5322"></script>
   <script type="text/javascript" src="/packages/check.js?hash=d4de24f57efd5d716b6ce9a7a90bf6a91b307bf4"></script>
   <script type="text/javascript" src="/packages/retry.js?hash=5179d91da075444c09bc870af3e6fcbd8be695b6"></script>
   <script type="text/javascript" src="/packages/ddp-common.js?hash=7e3f0c9d877af74b7eb5219d6bfbe02794a818ca"></script>
   <script type="text/javascript" src="/packages/reload.js?hash=b2d1086fd7bbc28deb93350f437ce0eef242baa3"></script>
   <script type="text/javascript" src="/packages/ddp-client.js?hash=98f50c1583d868f0e596ecb98ed32c7eb2bddf6d"></script>
   <script type="text/javascript" src="/packages/ddp.js?hash=6dc7ae4d7e07b3c1c2140c88207b0d1e5e1a0d48"></script>
   <script type="text/javascript" src="/packages/ddp-server.js?hash=d113bccbffeb5a7e546434f1fac70878711c2a7e"></script>
   <script type="text/javascript" src="/packages/allow-deny.js?hash=3f84d98a06ac6f4ce21a196fb538a26f443f2844"></script>
   <script type="text/javascript" src="/packages/insecure.js?hash=4b295c398ffa84184b91d904043e9e8801ba69a8"></script>
   <script type="text/javascript" src="/packages/mongo.js?hash=3037285b99384cbead7ba1aaeb2db56a6649844e"></script>
```

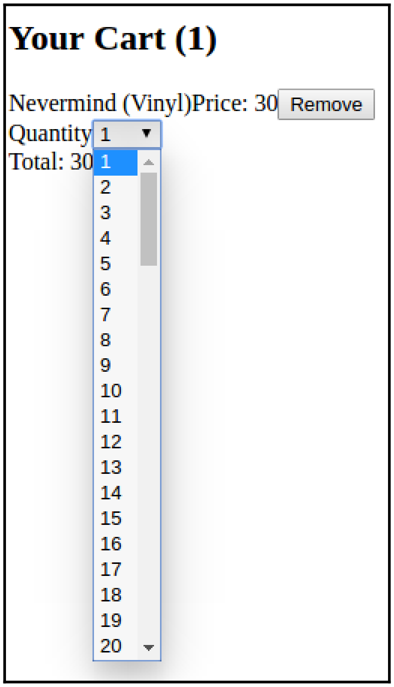

ProductsCollection
CartCollection
read
insert
ProductsContainer
props
List of Products
ProductComponent
ProductComponent
ProductComponent
....



```
import React, { Component, PropTypes } from 'react'

export default class CartItem extends Component {
  render() {
    const { title, price, onRemoveItem } = this.props
      return (
      <div>
        <span>{title}</span>
        <span>Price: {price}</span>
        <button onClick={onRemoveItem}>
          Remove
        </button>
      </div>
    )
  }
}

CartItem.propTypes = {
  title: PropTypes.string,
  price: PropTypes.number,
  onRemoveItem : PropTypes.func

}
```



```
import React, { Component, PropTypes } from 'react'

export default class Inventory extends Component {
  constructor(props) {
  super(props);
  this.changeHandler = this.changeHandler.bind(this);
}

changeHandler(event) {
  this.props.onChangeQuanity(event);
}

  render() {
    const { inventory, quantity, _id } = this.props;
    let options = [];
     for (let i = 1; i < inventory + 1; i++) {
        options.push(<option key={`inventory_${i}_${_id}`} value={i}>{i}</option>);
     }
      return (
      <div>
        <span>Quantity</span>
        <select  onChange={this.changeHandler} defaultValue={quantity} required>
          {options}
        </select>
      </div>
    )
  }
}

Inventory.propTypes = {
  inventory: PropTypes.number,
  _id: PropTypes.string,
  quantity:  PropTypes.number,
  onChangeQuanity : PropTypes.func
}
```

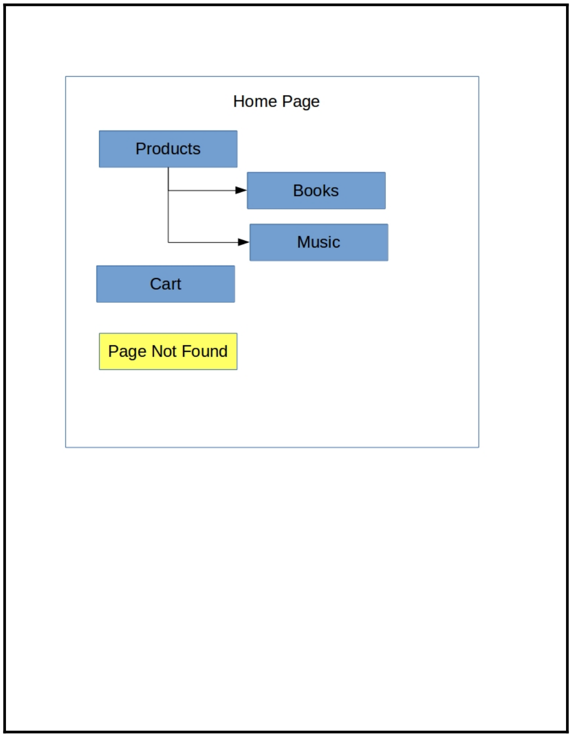


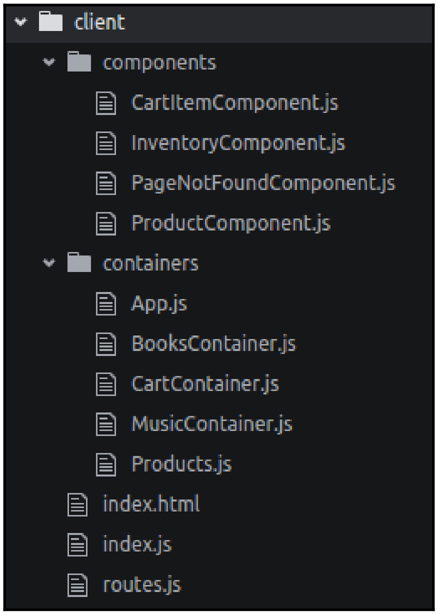
Your Cart (1)

Nevermind (Vinyl)Price: 30

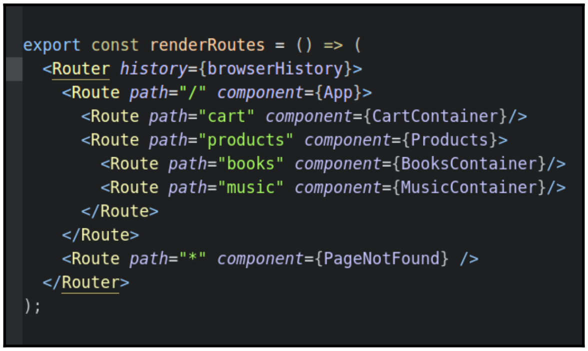
Quantity 1

Total: 30

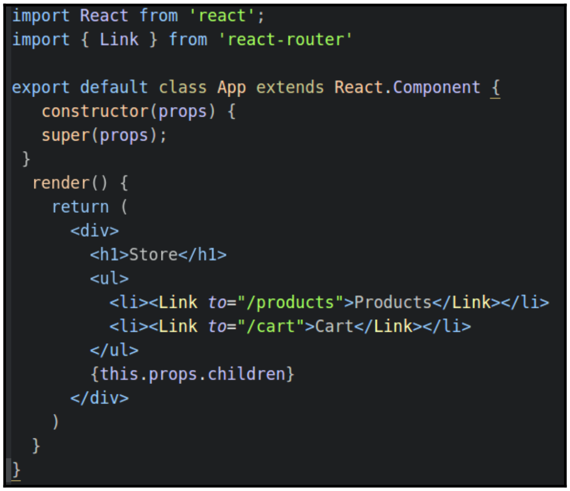




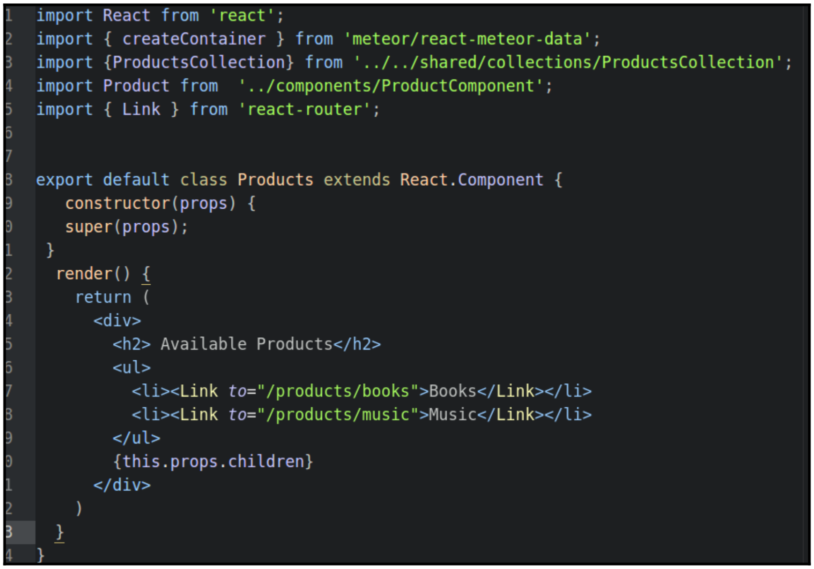
```
client
  components
    CartItemComponent.js
    InventoryComponent.js
    PageNotFoundComponent.js
    ProductComponent.js
  containers
    App.js
    BooksContainer.js
    CartContainer.js
    MusicContainer.js
    Products.js
  index.html
  index.js
  routes.js
```



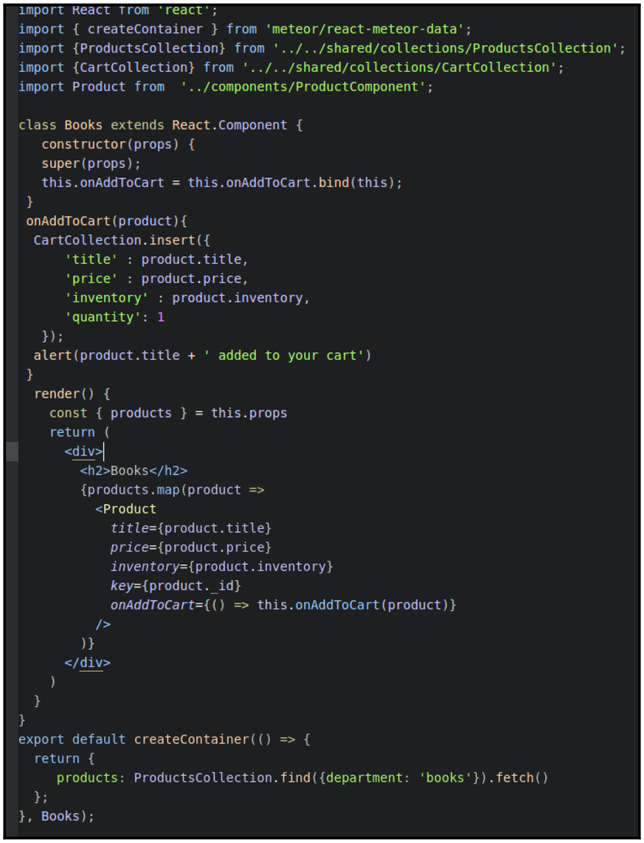
```
export const renderRoutes = () => (
  <Router history={browserHistory}>
    <Route path="/" component={App}>
      <Route path="cart" component={CartContainer}/>
      <Route path="products" component={Products}>
        <Route path="books" component={BooksContainer}/>
        <Route path="music" component={MusicContainer}/>
      </Route>
    </Route>
    <Route path="*" component={PageNotFound} />
  </Router>
);
```



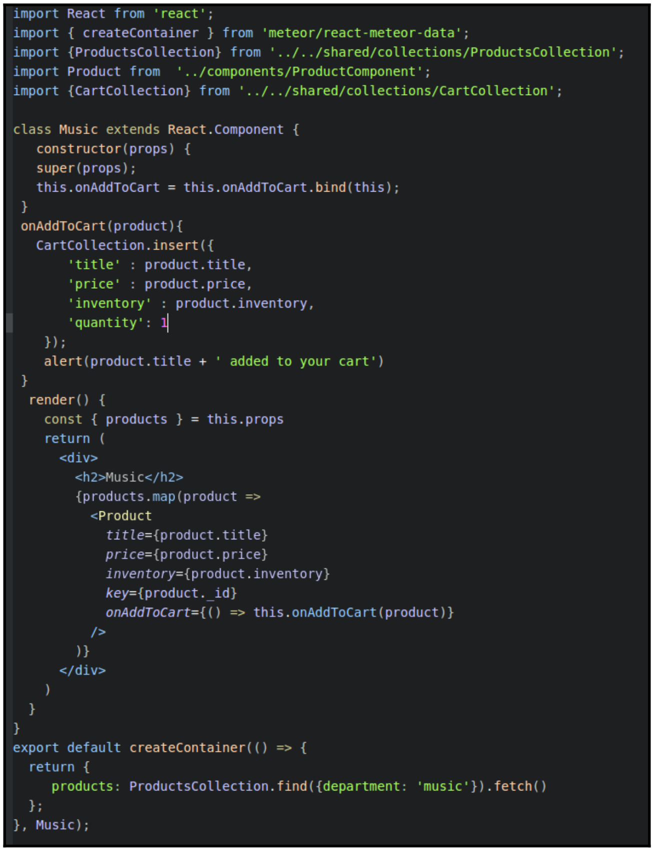
```
import React from 'react';
import { Link } from 'react-router'

export default class App extends React.Component {
   constructor(props) {
   super(props);
 }
  render() {
    return (
      <div>
        <h1>Store</h1>
        <ul>
          <li><Link to="/products">Products</Link></li>
          <li><Link to="/cart">Cart</Link></li>
        </ul>
        {this.props.children}
      </div>
    )
  }
}
```



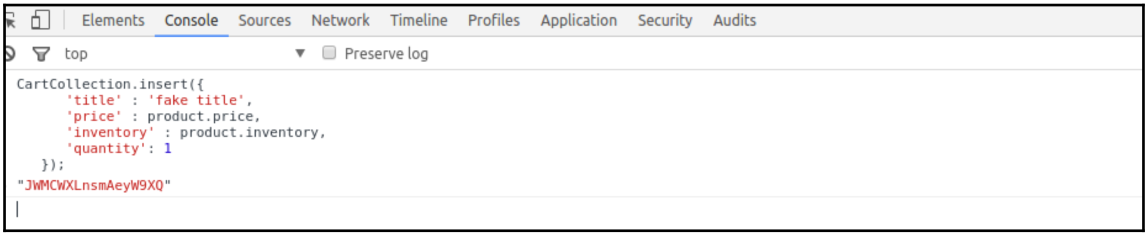
```
import React from 'react';
import { createContainer } from 'meteor/react-meteor-data';
import {ProductsCollection} from '../../shared/collections/ProductsCollection';
import Product from  '../components/ProductComponent';
import { Link } from 'react-router';


export default class Products extends React.Component {
   constructor(props) {
   super(props);
 }
  render() {
    return (
      <div>
        <h2> Available Products</h2>
        <ul>
          <li><Link to="/products/books">Books</Link></li>
          <li><Link to="/products/music">Music</Link></li>
        </ul>
        {this.props.children}
      </div>
    )
  }
}
```



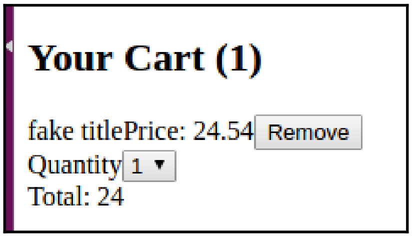
```
import React from 'react';
import { createContainer } from 'meteor/react-meteor-data';
import {ProductsCollection} from '../../shared/collections/ProductsCollection';
import {CartCollection} from '../../shared/collections/CartCollection';
import Product from  '../components/ProductComponent';

class Books extends React.Component {
   constructor(props) {
   super(props);
   this.onAddToCart = this.onAddToCart.bind(this);
 }
 onAddToCart(product){
  CartCollection.insert({
      'title' : product.title,
      'price' : product.price,
      'inventory' : product.inventory,
      'quantity': 1
   });
  alert(product.title + ' added to your cart')
 }
  render() {
    const { products } = this.props
    return (
      <div>
        <h2>Books</h2>
        {products.map(product =>
          <Product
            title={product.title}
            price={product.price}
            inventory={product.inventory}
            key={product._id}
            onAddToCart={() => this.onAddToCart(product)}
          />
        )}
      </div>
    )
  }
}
export default createContainer(() => {
  return {
     products: ProductsCollection.find({department: 'books'}).fetch()
  };
}, Books);
```



```
import React from 'react';
import { createContainer } from 'meteor/react-meteor-data';
import {ProductsCollection} from '../../shared/collections/ProductsCollection';
import Product from  '../components/ProductComponent';
import {CartCollection} from '../../shared/collections/CartCollection';

class Music extends React.Component {
   constructor(props) {
   super(props);
   this.onAddToCart = this.onAddToCart.bind(this);
 }
 onAddToCart(product){
   CartCollection.insert({
       'title' : product.title,
       'price' : product.price,
       'inventory' : product.inventory,
       'quantity': 1
    });
    alert(product.title + ' added to your cart')
 }
  render() {
    const { products } = this.props
    return (
      <div>
        <h2>Music</h2>
        {products.map(product =>
          <Product
            title={product.title}
            price={product.price}
            inventory={product.inventory}
            key={product._id}
            onAddToCart={() => this.onAddToCart(product)}
          />
        )}
      </div>
    )
  }
}
export default createContainer(() => {
  return {
     products: ProductsCollection.find({department: 'music'}).fetch()
  };
}, Music);
```



```
CartCollection.insert({
       'title' : 'fake title',
       'price' : product.price,
       'inventory' : product.inventory,
       'quantity': 1
    });
"JWMCWXLnsmAeyW9XQ"
```

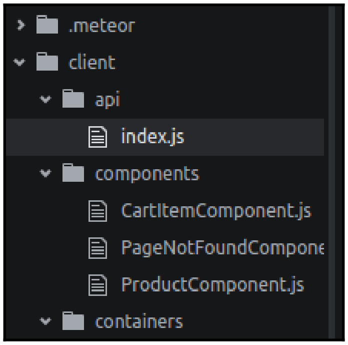


Your Cart (1)

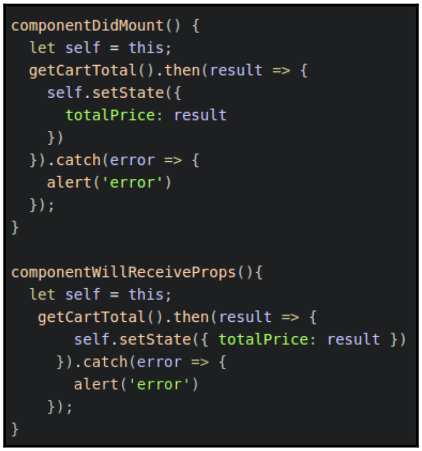
fake titlePrice: 24.54 Remove

Quantity 1 ▾

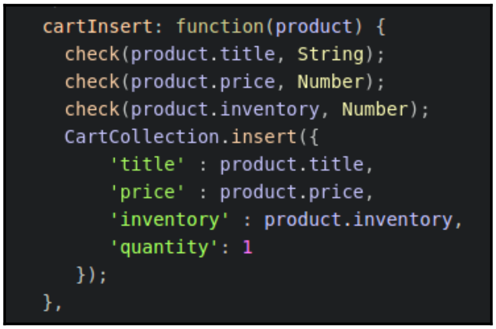
Total: 24



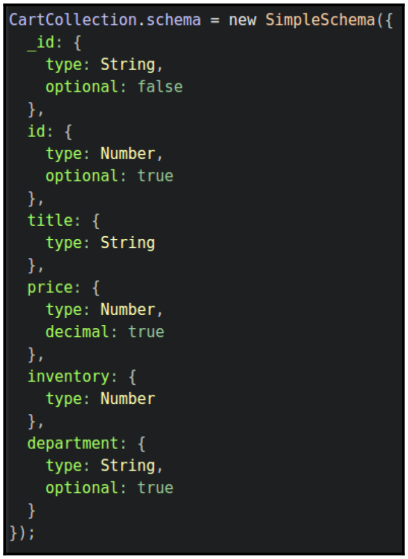
```
> .meteor
v client
  v api
      index.js
  v components
      CartItemComponent.js
      PageNotFoundCompone
      ProductComponent.js
  v containers
```



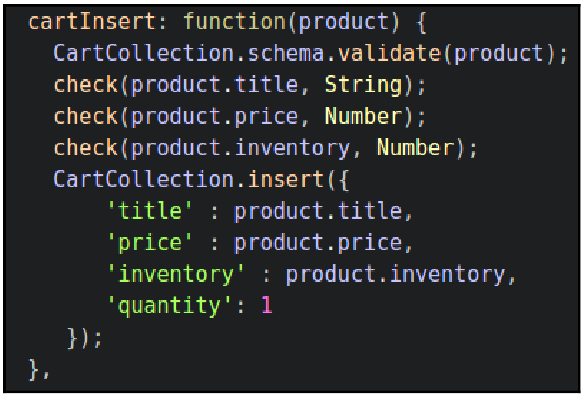
```
componentDidMount() {
  let self = this;
  getCartTotal().then(result => {
    self.setState({
      totalPrice: result
    })
  }).catch(error => {
    alert('error')
  });
}

componentWillReceiveProps(){
  let self = this;
   getCartTotal().then(result => {
       self.setState({ totalPrice: result })
     }).catch(error => {
       alert('error')
    });
}
```



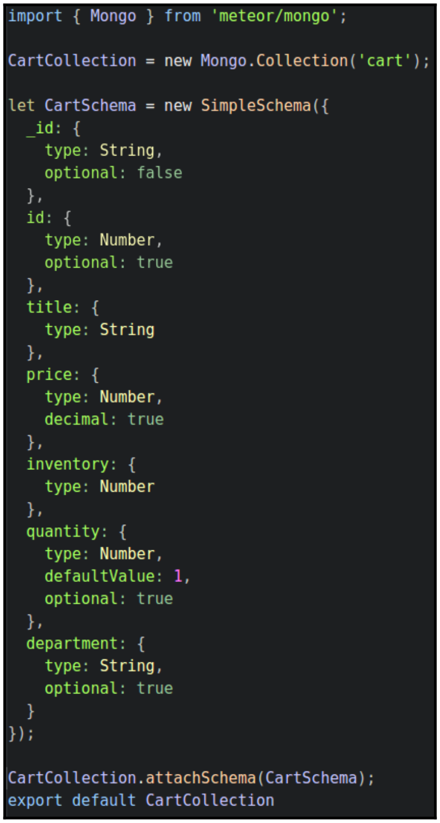
```
cartInsert: function(product) {
  check(product.title, String);
  check(product.price, Number);
  check(product.inventory, Number);
  CartCollection.insert({
      'title' : product.title,
      'price' : product.price,
      'inventory' : product.inventory,
      'quantity': 1
   });
},
```



```
CartCollection.schema = new SimpleSchema({
  _id: {
    type: String,
    optional: false
  },
  id: {
    type: Number,
    optional: true
  },
  title: {
    type: String
  },
  price: {
    type: Number,
    decimal: true
  },
  inventory: {
    type: Number
  },
  department: {
    type: String,
    optional: true
  }
});
```

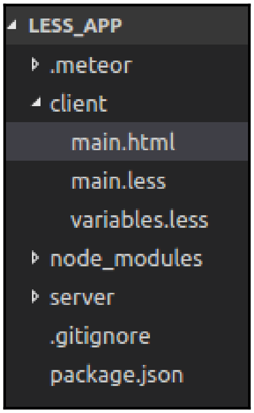


```
cartInsert: function(product) {
 CartCollection.schema.validate(product);
 check(product.title, String);
 check(product.price, Number);
 check(product.inventory, Number);
 CartCollection.insert({
     'title' : product.title,
     'price' : product.price,
     'inventory' : product.inventory,
     'quantity': 1
  });
},
```

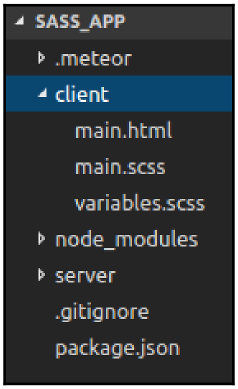


```
import { Mongo } from 'meteor/mongo';

CartCollection = new Mongo.Collection('cart');

let CartSchema = new SimpleSchema({
  _id: {
    type: String,
    optional: false
  },
  id: {
    type: Number,
    optional: true
  },
  title: {
    type: String
  },
  price: {
    type: Number,
    decimal: true
  },
  inventory: {
    type: Number
  },
  quantity: {
    type: Number,
    defaultValue: 1,
    optional: true
  },
  department: {
    type: String,
    optional: true
  }
});

CartCollection.attachSchema(CartSchema);
export default CartCollection
```

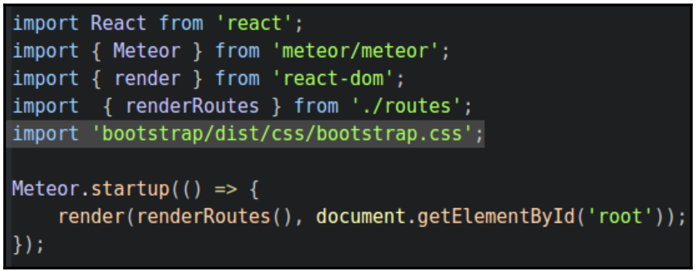
Chapter 3:
Style Your React Components with Bootstrap and Material Design



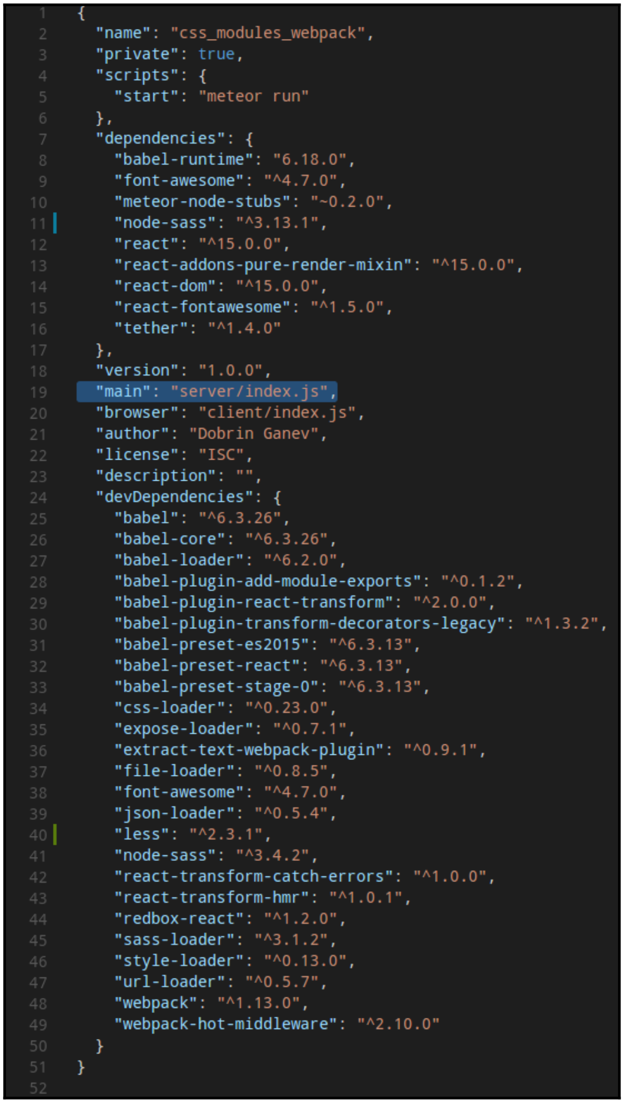
```
LESS_APP
  .meteor
  client
    main.html
    main.less
    variables.less
  node_modules
  server
  .gitignore
  package.json
```



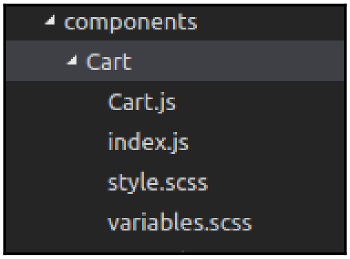
```
SASS_APP
  .meteor
  client
    main.html
    main.scss
    variables.scss
  node_modules
  server
  .gitignore
  package.json
```

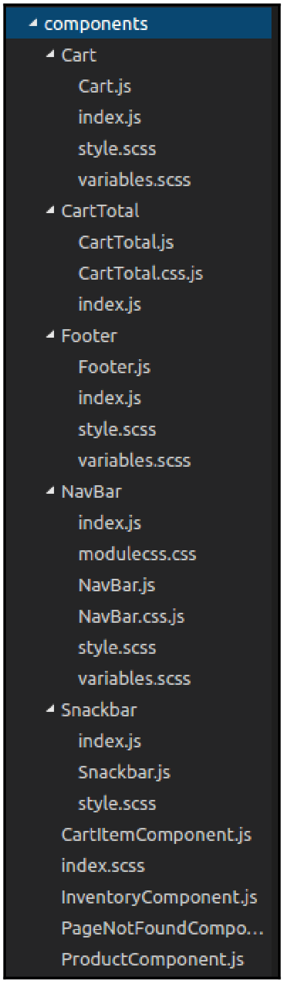


```
import React from 'react';
import { Meteor } from 'meteor/meteor';
import { render } from 'react-dom';
import  { renderRoutes } from './routes';
import 'bootstrap/dist/css/bootstrap.css';

Meteor.startup(() => {
    render(renderRoutes(), document.getElementById('root'));
});
```

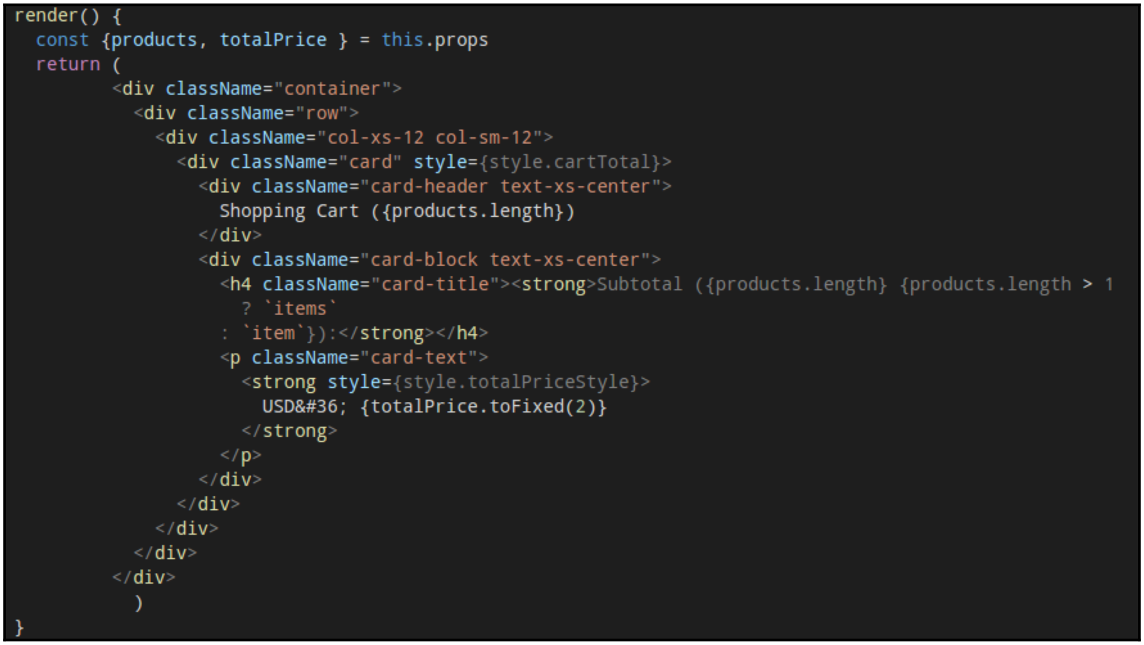


```
{
  "name": "css_modules_webpack",
  "private": true,
  "scripts": {
    "start": "meteor run"
  },
  "dependencies": {
    "babel-runtime": "6.18.0",
    "font-awesome": "^4.7.0",
    "meteor-node-stubs": "~0.2.0",
    "node-sass": "^3.13.1",
    "react": "^15.0.0",
    "react-addons-pure-render-mixin": "^15.0.0",
    "react-dom": "^15.0.0",
    "react-fontawesome": "^1.5.0",
    "tether": "^1.4.0"
  },
  "version": "1.0.0",
  "main": "server/index.js",
  "browser": "client/index.js",
  "author": "Dobrin Ganev",
  "license": "ISC",
  "description": "",
  "devDependencies": {
    "babel": "^6.3.26",
    "babel-core": "^6.3.26",
    "babel-loader": "^6.2.0",
    "babel-plugin-add-module-exports": "^0.1.2",
    "babel-plugin-react-transform": "^2.0.0",
    "babel-plugin-transform-decorators-legacy": "^1.3.2",
    "babel-preset-es2015": "^6.3.13",
    "babel-preset-react": "^6.3.13",
    "babel-preset-stage-0": "^6.3.13",
    "css-loader": "^0.23.0",
    "expose-loader": "^0.7.1",
    "extract-text-webpack-plugin": "^0.9.1",
    "file-loader": "^0.8.5",
    "font-awesome": "^4.7.0",
    "json-loader": "^0.5.4",
    "less": "^2.3.1",
    "node-sass": "^3.4.2",
    "react-transform-catch-errors": "^1.0.0",
    "react-transform-hmr": "^1.0.1",
    "redbox-react": "^1.2.0",
    "sass-loader": "^3.1.2",
    "style-loader": "^0.13.0",
    "url-loader": "^0.5.7",
    "webpack": "^1.13.0",
    "webpack-hot-middleware": "^2.10.0"
  }
}
```

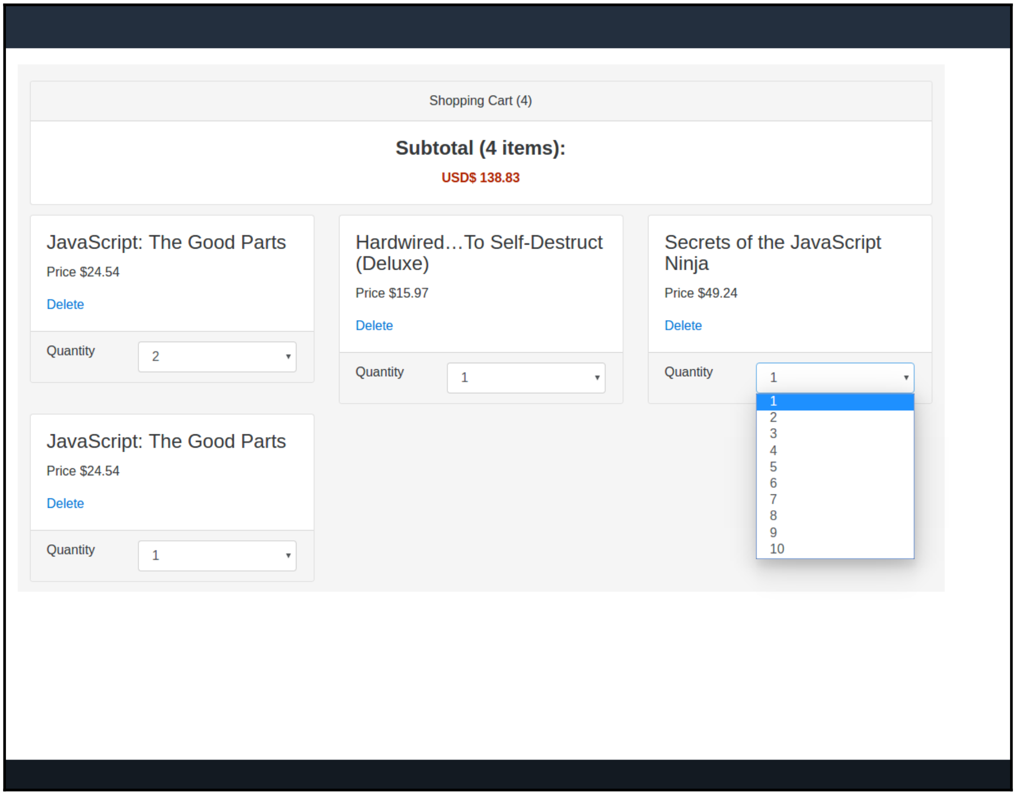




```
components
  Cart
    Cart.js
    index.js
    style.scss
    variables.scss
  CartTotal
    CartTotal.js
    CartTotal.css.js
    index.js
  Footer
    Footer.js
    index.js
    style.scss
    variables.scss
  NavBar
    index.js
    modulecss.css
    NavBar.js
    NavBar.css.js
    style.scss
    variables.scss
  Snackbar
    index.js
    Snackbar.js
    style.scss
  CartItemComponent.js
  index.scss
  InventoryComponent.js
  PageNotFoundCompo...
  ProductComponent.js
```



```
render() {
  const {products, totalPrice } = this.props
  return (
         <div className="container">
           <div className="row">
             <div className="col-xs-12 col-sm-12">
               <div className="card" style={style.cartTotal}>
                 <div className="card-header text-xs-center">
                   Shopping Cart ({products.length})
                 </div>
                 <div className="card-block text-xs-center">
                   <h4 className="card-title"><strong>Subtotal ({products.length} {products.length > 1
                     ? `items`
                   : `item`}):</strong></h4>
                   <p className="card-text">
                     <strong style={style.totalPriceStyle}>
                       USD&#36; {totalPrice.toFixed(2)}
                     </strong>
                   </p>
                 </div>
               </div>
             </div>
           </div>
         </div>
           )
}
```



Shopping Cart (4)

Subtotal (4 items):

USD$ 138.83

JavaScript: The Good Parts

Price $24.54

Delete

Quantity 2

Hardwired…To Self-Destruct (Deluxe)

Price $15.97

Delete

Quantity 1

Secrets of the JavaScript Ninja

Price $49.24

Delete

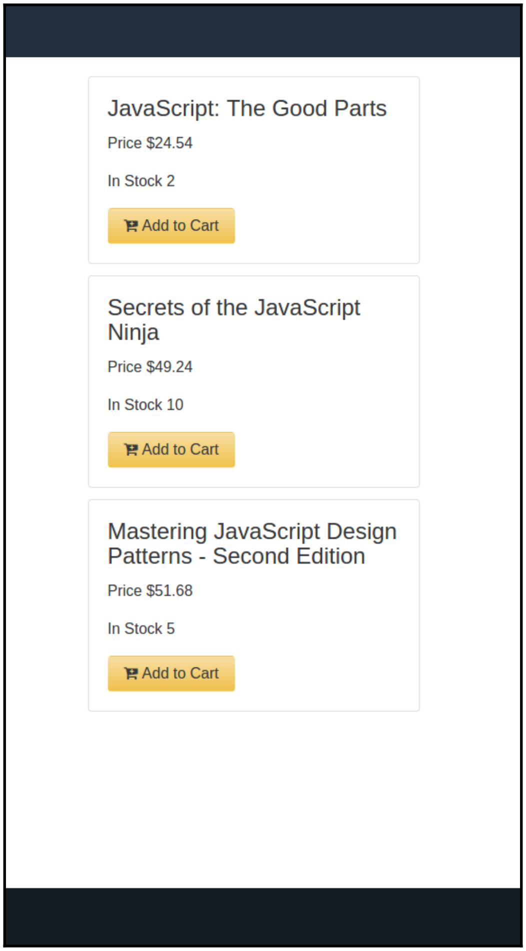
Quantity 1

JavaScript: The Good Parts

Price $24.54

Delete

Quantity 1



JavaScript: The Good Parts

Price $24.54

In Stock 2

Secrets of the JavaScript Ninja

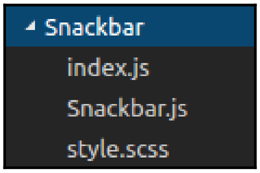
Price $49.24

In Stock 10

Mastering JavaScript Design Patterns - Second Edition

Price $51.68

In Stock 5

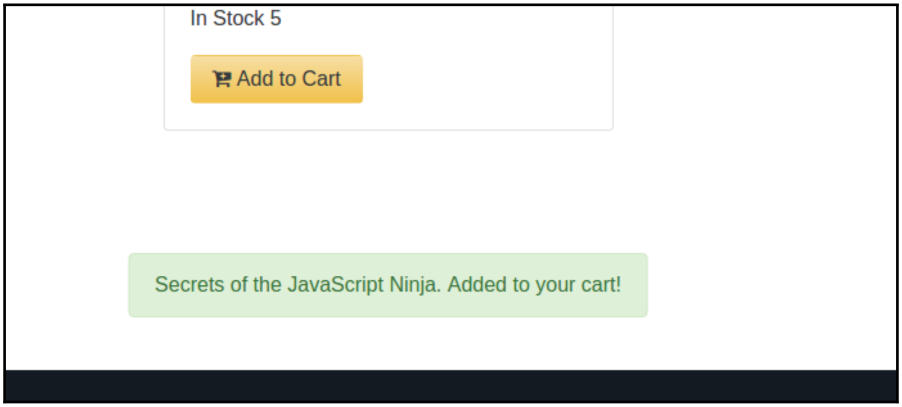


Snackbar

index.js

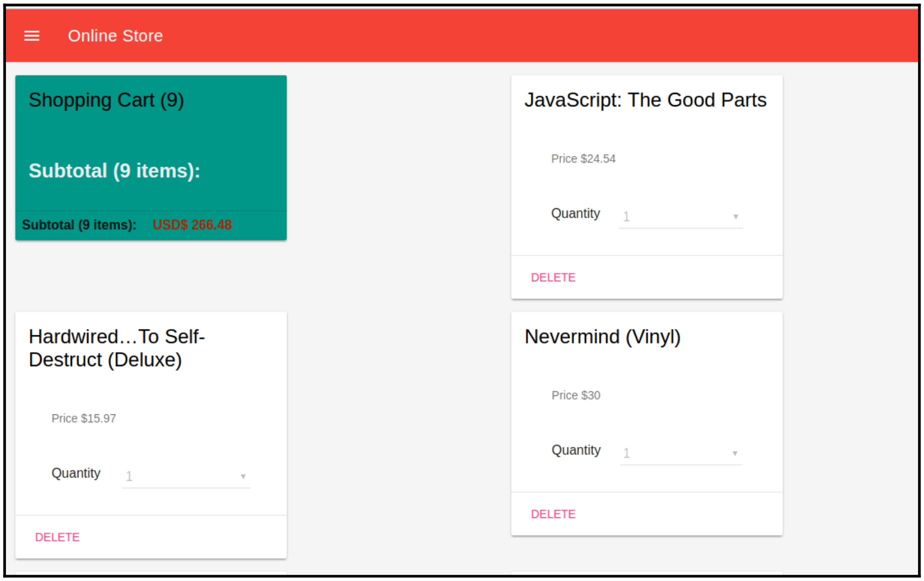
Snackbar.js

style.scss



In Stock 5

Secrets of the JavaScript Ninja. Added to your cart!



Shopping Cart (9)

Subtotal (9 items):

Subtotal (9 items): USD$ 266.48

JavaScript: The Good Parts

Price $24.54

Quantity 1

Hardwired…To Self-Destruct (Deluxe)

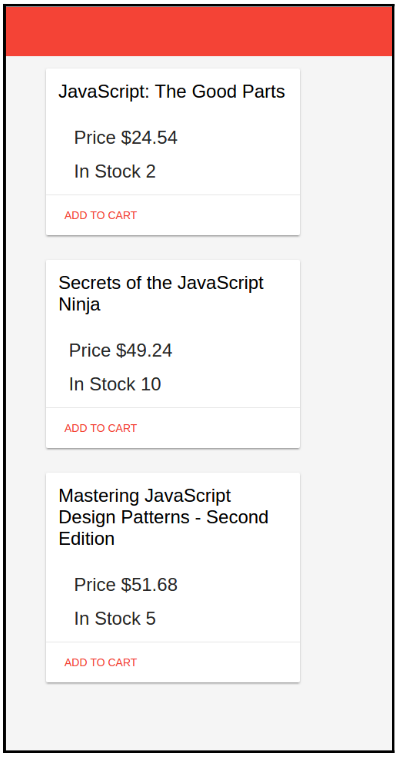
Price $15.97

Quantity 1

Nevermind (Vinyl)

Price $30

Quantity 1



JavaScript: The Good Parts

Price $24.54

In Stock 2

Secrets of the JavaScript Ninja

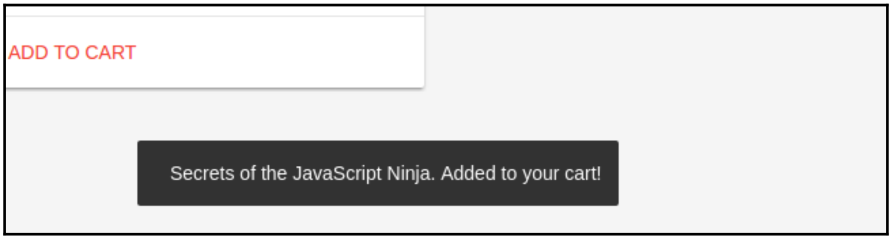
Price $49.24

In Stock 10

Mastering JavaScript Design Patterns - Second Edition

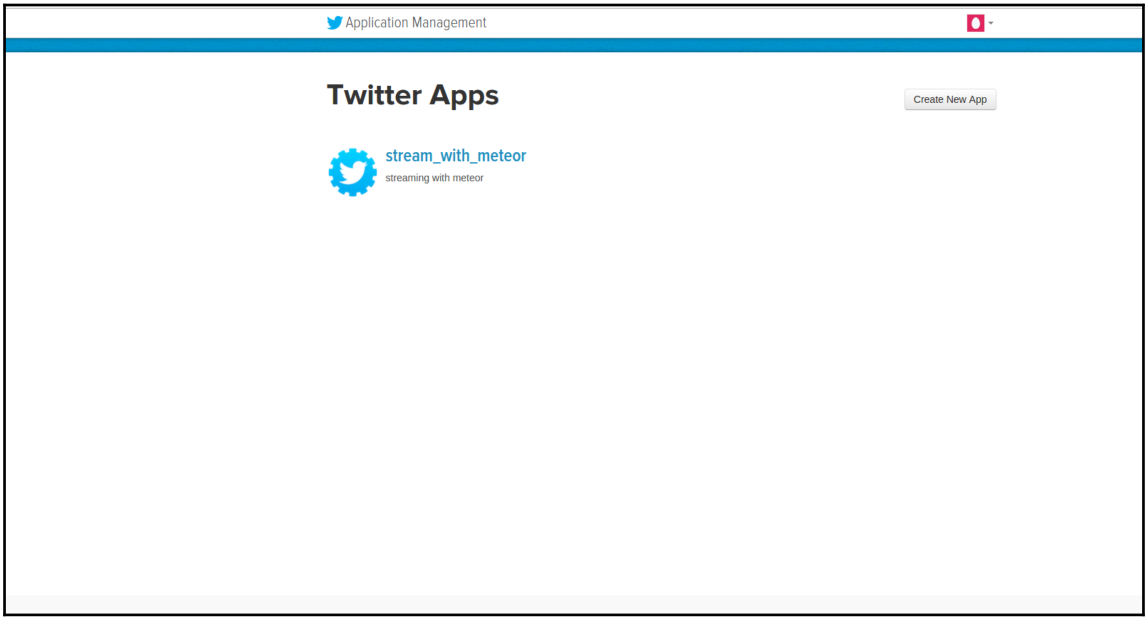
Price $51.68

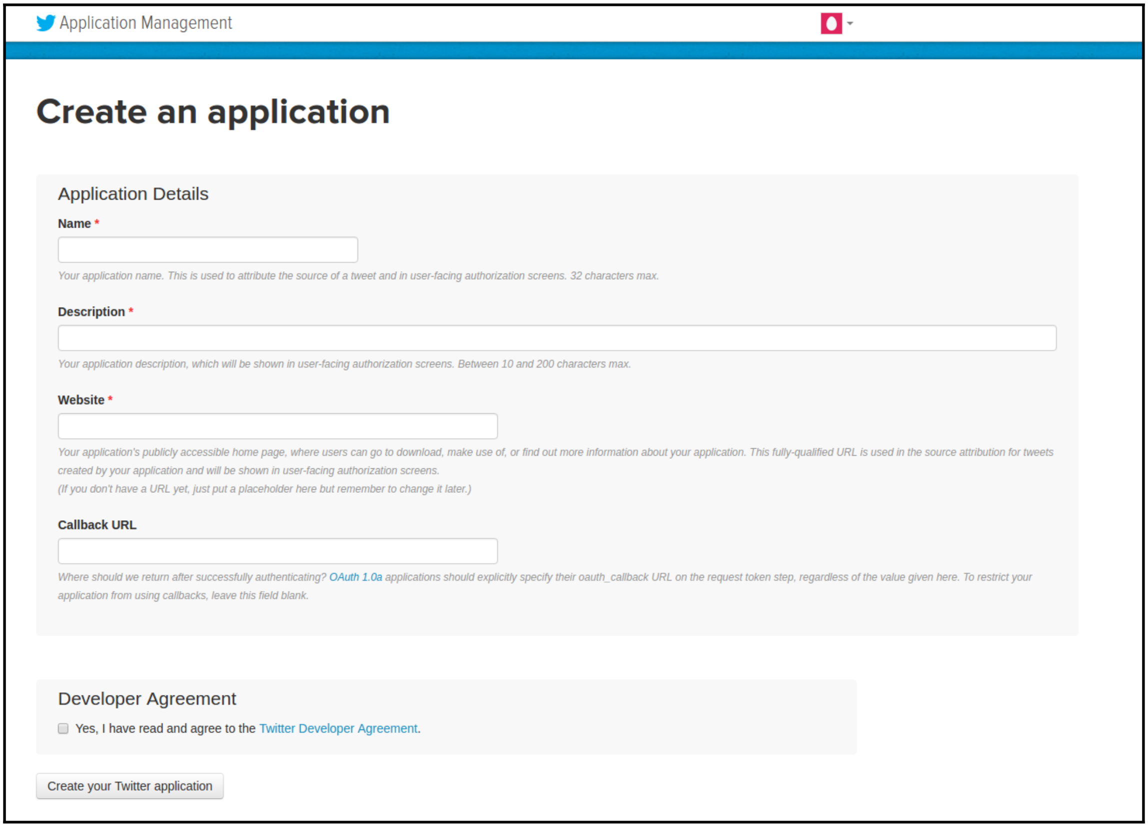
In Stock 5



Secrets of the JavaScript Ninja. Added to your cart!

Chapter 4: Real-Time Twitter Streaming





Create an application

Application Details

Name *

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

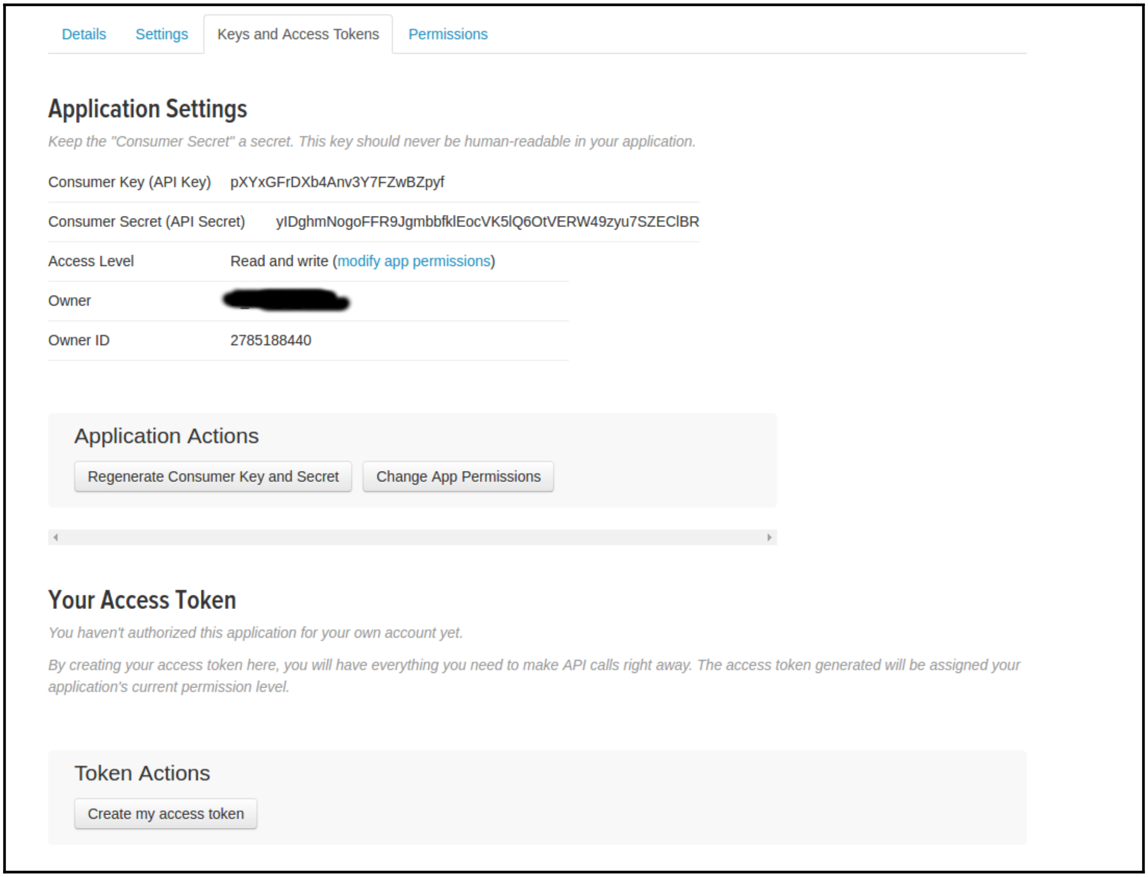
Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

Developer Agreement

□ Yes, I have read and agree to the Twitter Developer Agreement.



Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	pXYxGFrDXb4Anv3Y7FZwBZpyf
Consumer Secret (API Secret)	yIDghmNogoFFR9JgmbbfklEocVK5lQ6OtVERW49zyu7SZEClBR
Access Level	Read and write (modify app permissions)
Owner	[redacted]
Owner ID	2785188440

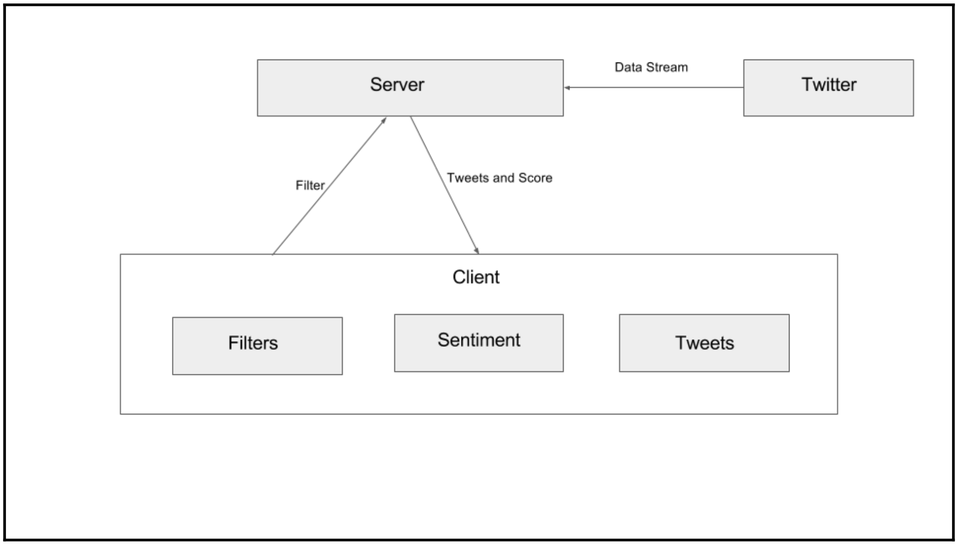
Application Actions

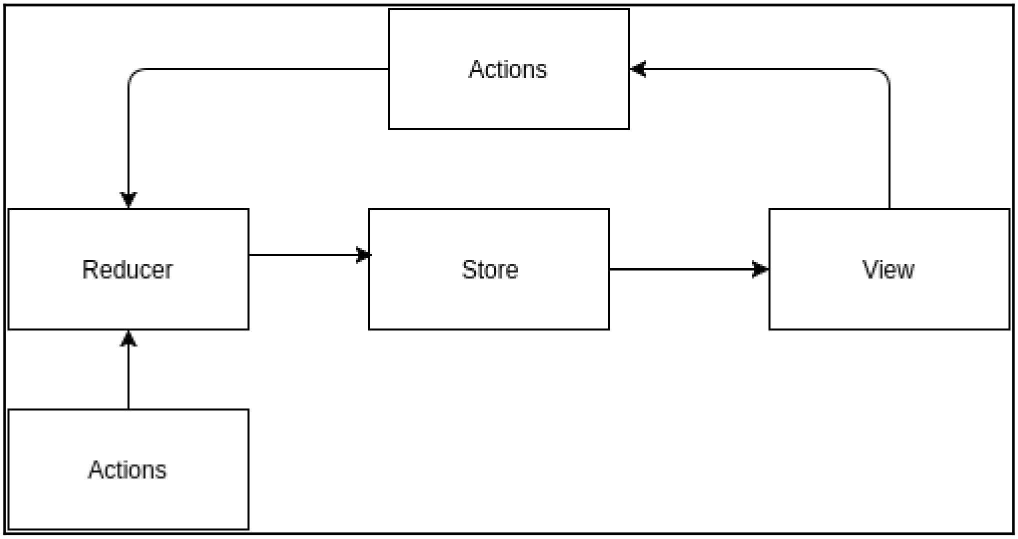
Your Access Token

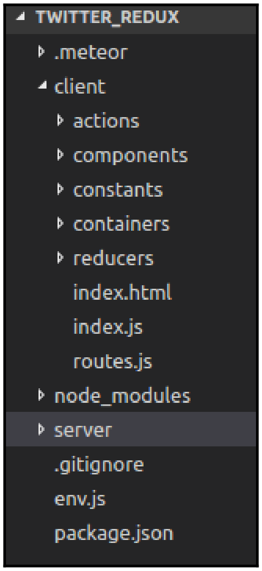
You haven't authorized this application for your own account yet.

By creating your access token here, you will have everything you need to make API calls right away. The access token generated will be assigned your application's current permission level.

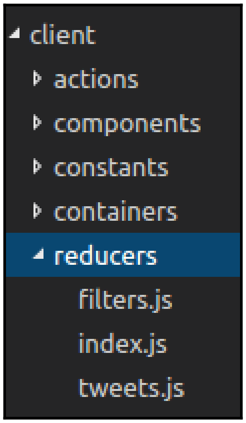
Token Actions


Server
Data Stream
Twitter
Filter
Tweets and Score
Client
Filters
Sentiment
Tweets

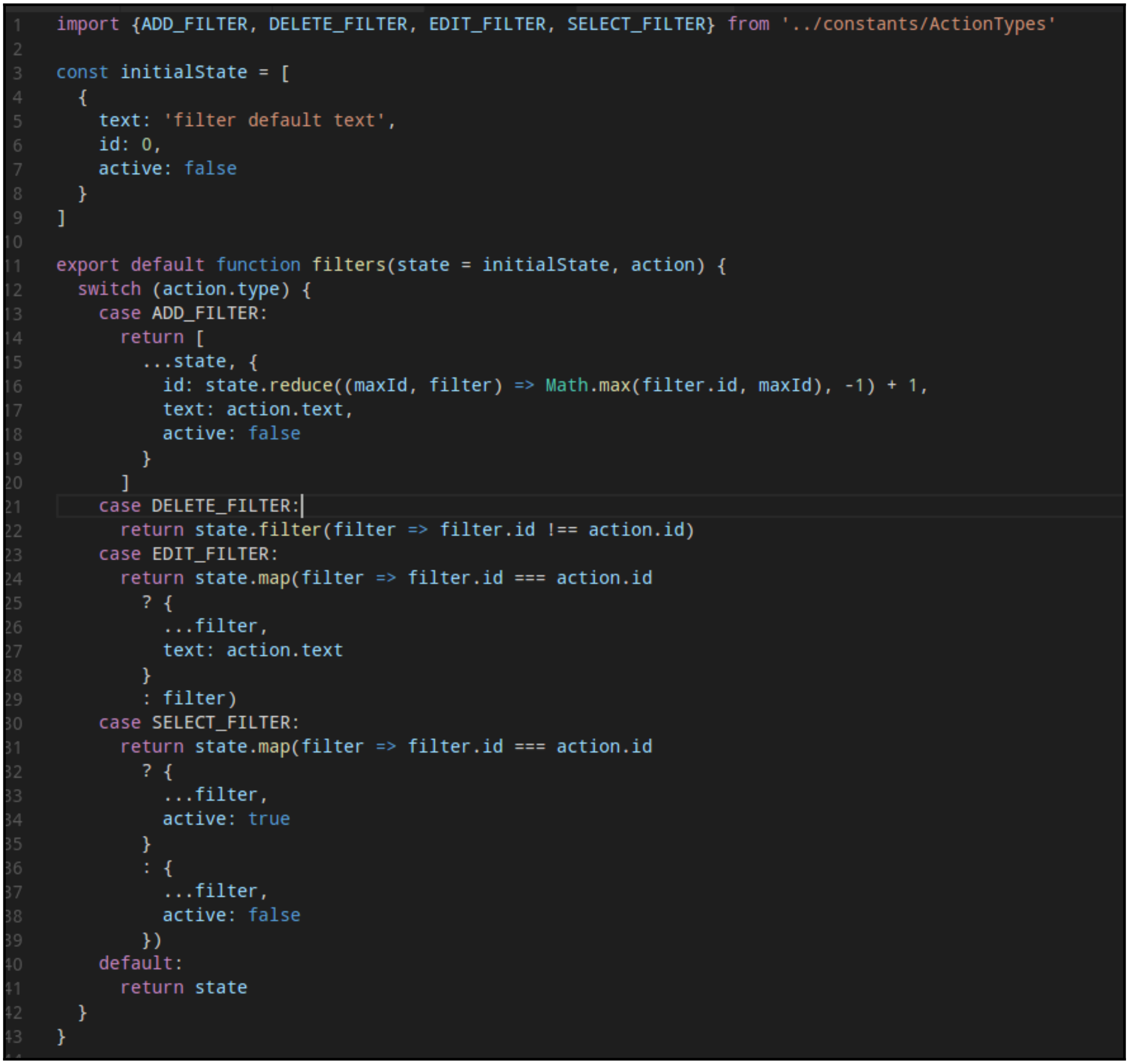

Actions
Reducer
Store
View
Actions



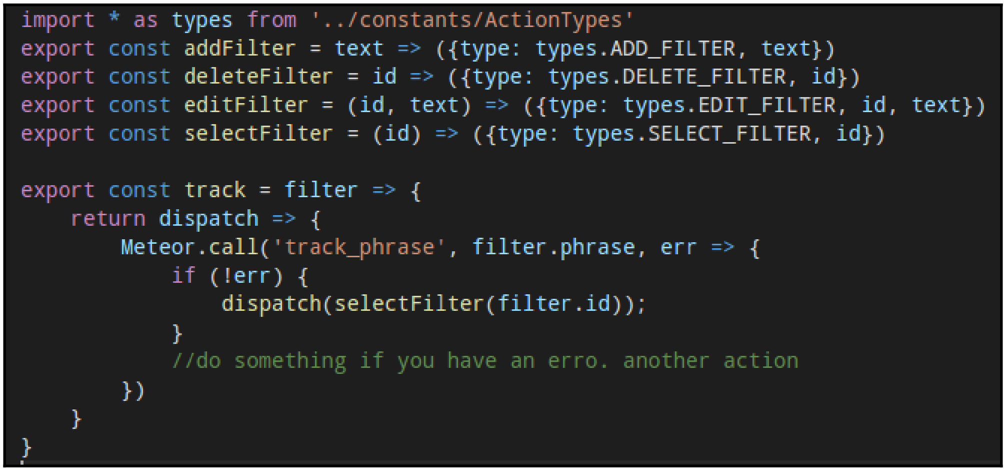
```
◢ TWITTER_REDUX
  ▷ .meteor
  ◢ client
    ▷ actions
    ▷ components
    ▷ constants
    ▷ containers
    ▷ reducers
      index.html
      index.js
      routes.js
  ▷ node_modules
  ▷ server
    .gitignore
    env.js
    package.json
```



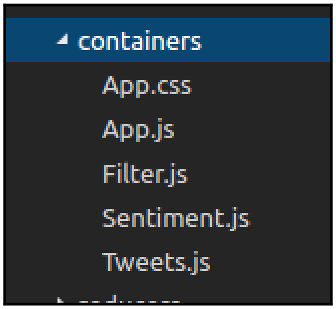
```
◢ client
  ▷ actions
  ▷ components
  ▷ constants
  ▷ containers
  ◢ reducers
      filters.js
      index.js
      tweets.js
```

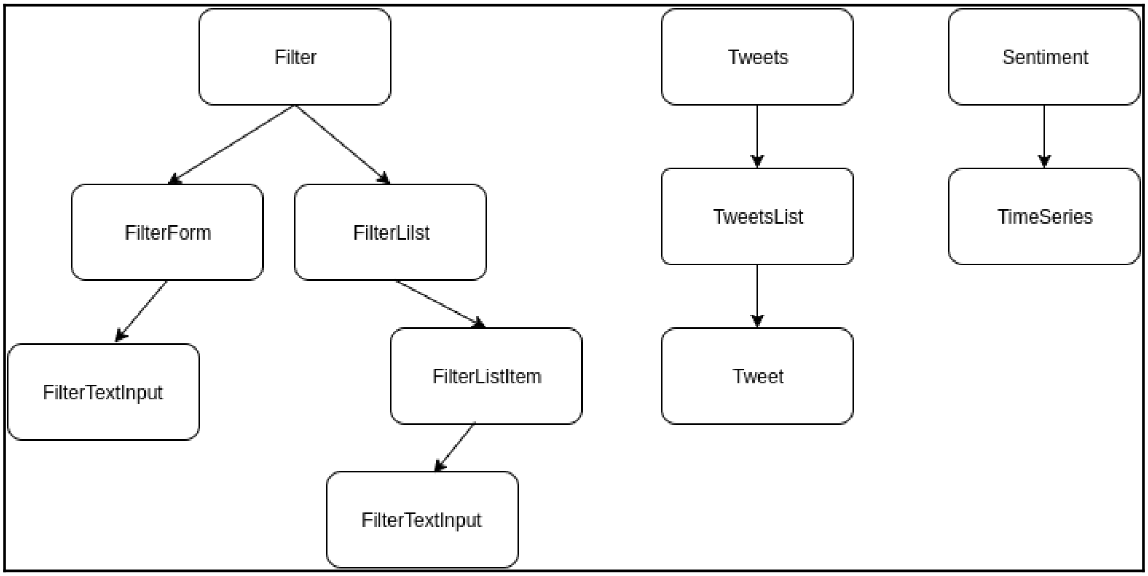


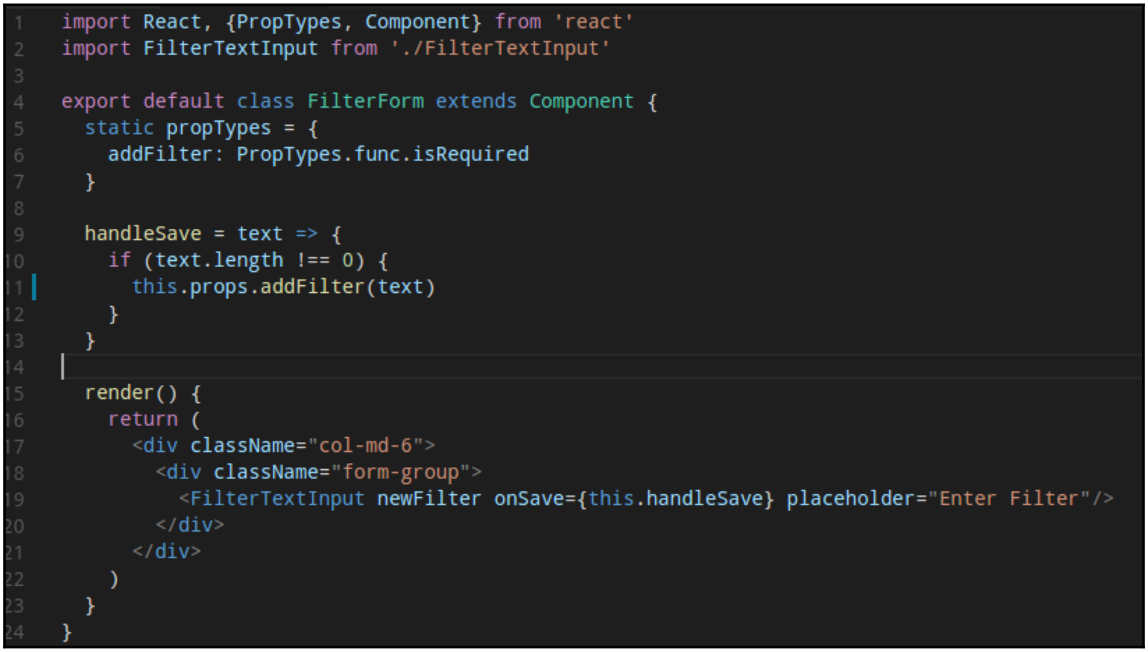
```
import {ADD_FILTER, DELETE_FILTER, EDIT_FILTER, SELECT_FILTER} from '../constants/ActionTypes'

const initialState = [
  {
    text: 'filter default text',
    id: 0,
    active: false
  }
]

export default function filters(state = initialState, action) {
  switch (action.type) {
    case ADD_FILTER:
      return [
        ...state, {
          id: state.reduce((maxId, filter) => Math.max(filter.id, maxId), -1) + 1,
          text: action.text,
          active: false
        }
      ]
    case DELETE_FILTER:
      return state.filter(filter => filter.id !== action.id)
    case EDIT_FILTER:
      return state.map(filter => filter.id === action.id
        ? {
          ...filter,
          text: action.text
        }
        : filter)
    case SELECT_FILTER:
      return state.map(filter => filter.id === action.id
        ? {
          ...filter,
          active: true
        }
        : {
          ...filter,
          active: false
        })
    default:
      return state
  }
}
```



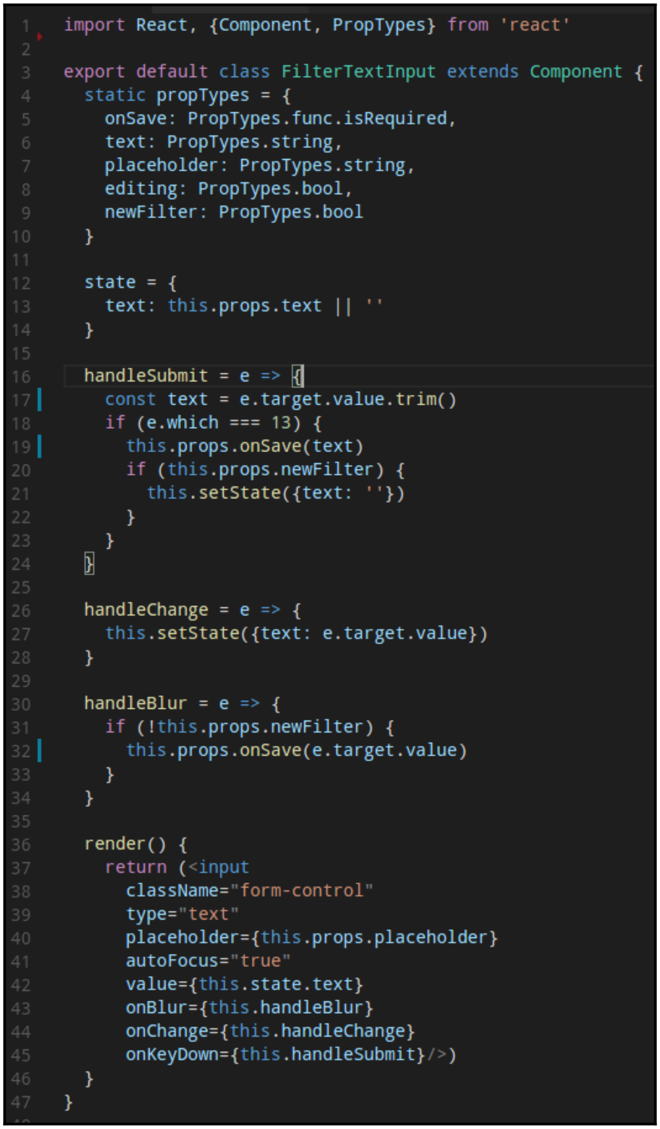
```
import * as types from '../constants/ActionTypes'
export const addFilter = text => ({type: types.ADD_FILTER, text})
export const deleteFilter = id => ({type: types.DELETE_FILTER, id})
export const editFilter = (id, text) => ({type: types.EDIT_FILTER, id, text})
export const selectFilter = (id) => ({type: types.SELECT_FILTER, id})

export const track = filter => {
    return dispatch => {
        Meteor.call('track_phrase', filter.phrase, err => {
            if (!err) {
                dispatch(selectFilter(filter.id));
            }
            //do something if you have an erro. another action
        })
    }
}
```



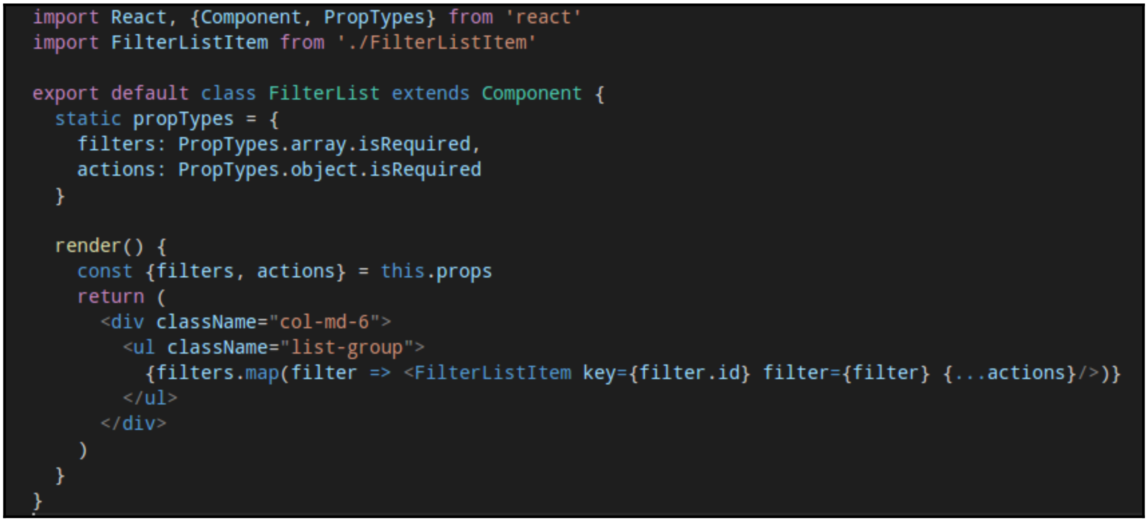




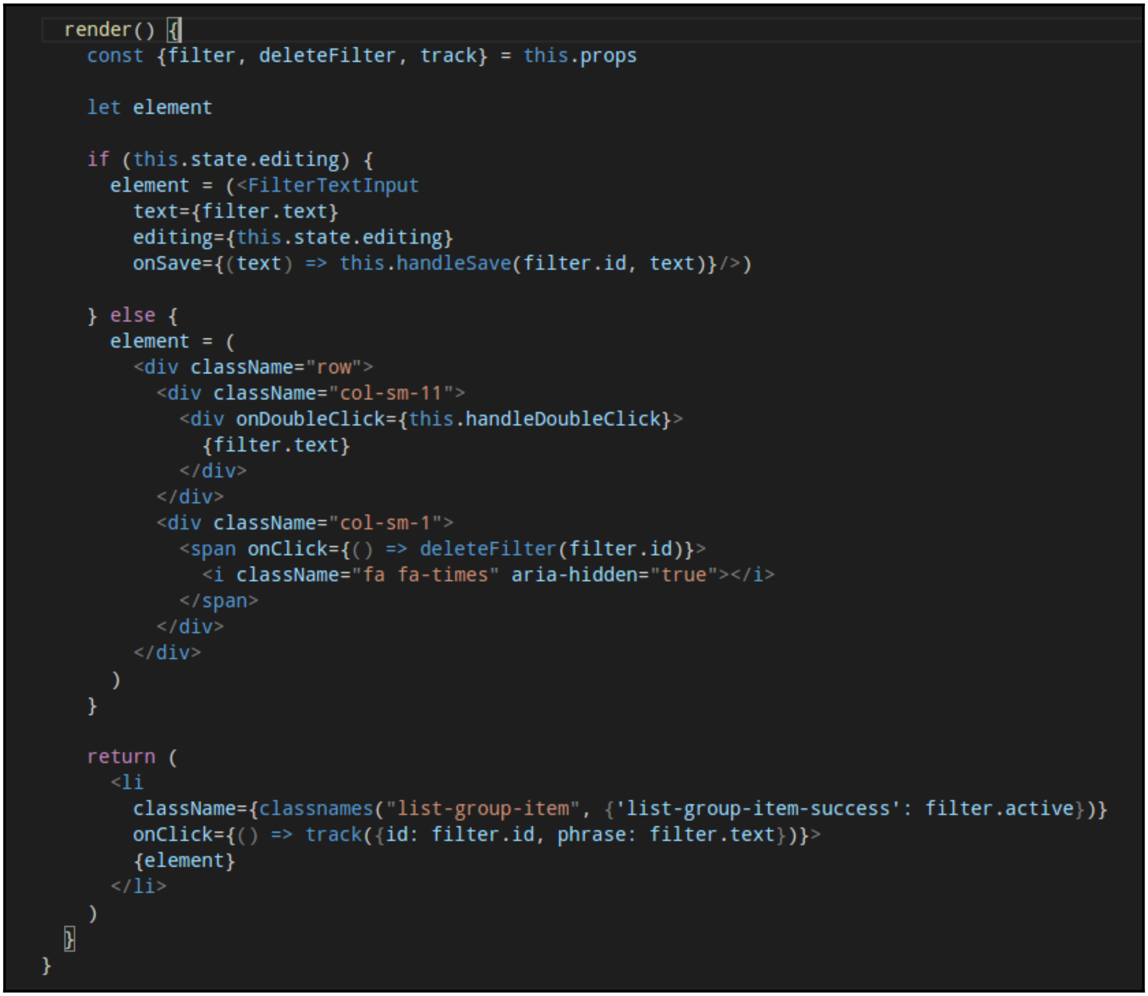
```
import React, {PropTypes, Component} from 'react'
import FilterTextInput from './FilterTextInput'

export default class FilterForm extends Component {
  static propTypes = {
    addFilter: PropTypes.func.isRequired
  }

  handleSave = text => {
    if (text.length !== 0) {
      this.props.addFilter(text)
    }
  }

  render() {
    return (
      <div className="col-md-6">
        <div className="form-group">
          <FilterTextInput newFilter onSave={this.handleSave} placeholder="Enter Filter"/>
        </div>
      </div>
    )
  }
}
```



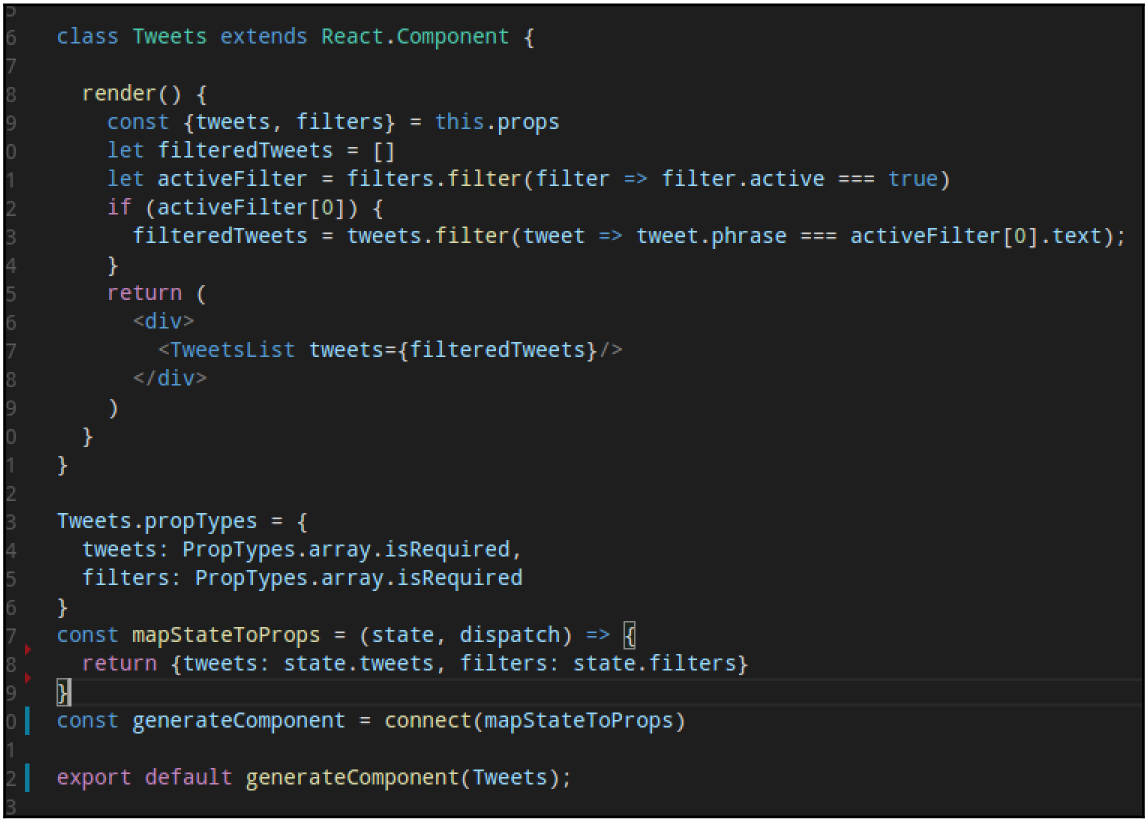
```
import React, {Component, PropTypes} from 'react'

export default class FilterTextInput extends Component {
  static propTypes = {
    onSave: PropTypes.func.isRequired,
    text: PropTypes.string,
    placeholder: PropTypes.string,
    editing: PropTypes.bool,
    newFilter: PropTypes.bool
  }

  state = {
    text: this.props.text || ''
  }

  handleSubmit = e => {
    const text = e.target.value.trim()
    if (e.which === 13) {
      this.props.onSave(text)
      if (this.props.newFilter) {
        this.setState({text: ''})
      }
    }
  }

  handleChange = e => {
    this.setState({text: e.target.value})
  }

  handleBlur = e => {
    if (!this.props.newFilter) {
      this.props.onSave(e.target.value)
    }
  }

  render() {
    return (<input
      className="form-control"
      type="text"
      placeholder={this.props.placeholder}
      autoFocus="true"
      value={this.state.text}
      onBlur={this.handleBlur}
      onChange={this.handleChange}
      onKeyDown={this.handleSubmit}/>)
  }
}
```



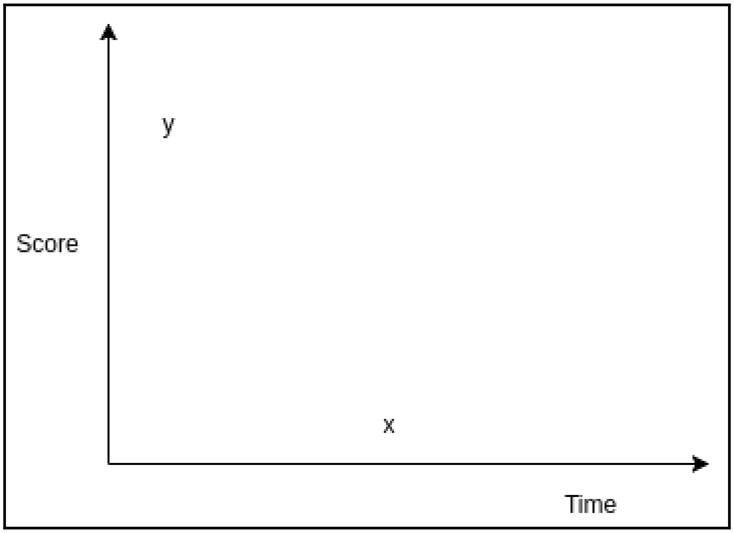
```
import React, {Component, PropTypes} from 'react'
import FilterListItem from './FilterListItem'

export default class FilterList extends Component {
  static propTypes = {
    filters: PropTypes.array.isRequired,
    actions: PropTypes.object.isRequired
  }

  render() {
    const {filters, actions} = this.props
    return (
      <div className="col-md-6">
        <ul className="list-group">
          {filters.map(filter => <FilterListItem key={filter.id} filter={filter} {...actions}/>)}
        </ul>
      </div>
    )
  }
}
```

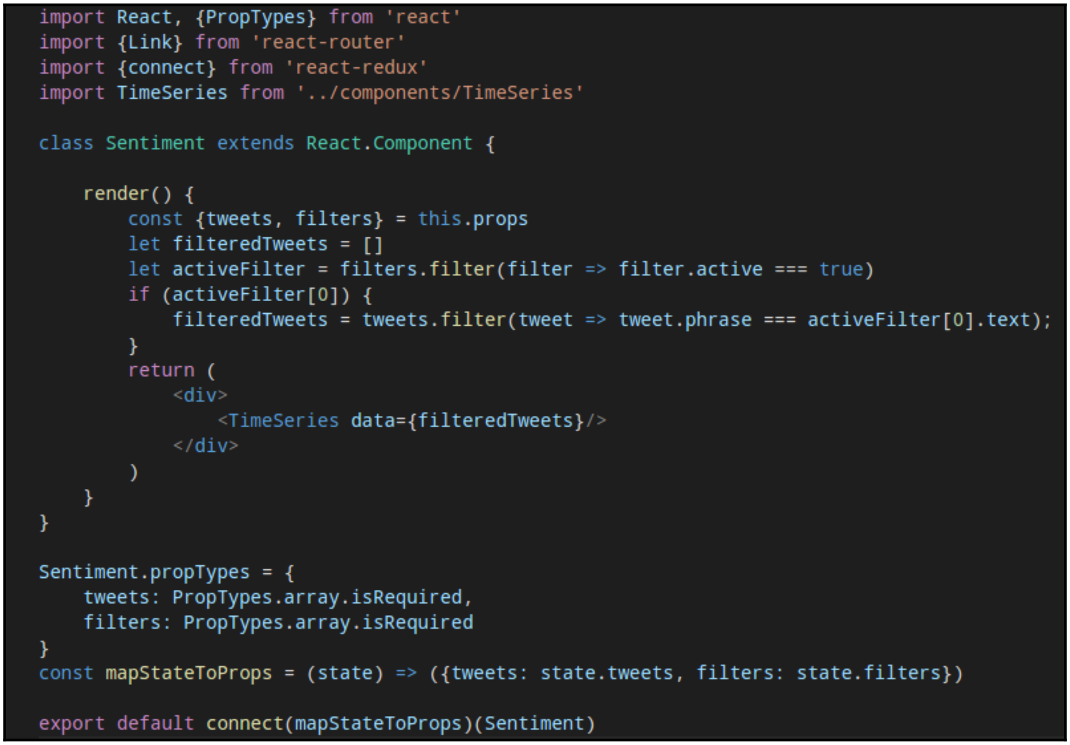


```
  render() {
    const {filter, deleteFilter, track} = this.props

    let element

    if (this.state.editing) {
      element = (<FilterTextInput
        text={filter.text}
        editing={this.state.editing}
        onSave={(text) => this.handleSave(filter.id, text)}/>)

    } else {
      element = (
        <div className="row">
          <div className="col-sm-11">
            <div onDoubleClick={this.handleDoubleClick}>
              {filter.text}
            </div>
          </div>
          <div className="col-sm-1">
            <span onClick={() => deleteFilter(filter.id)}>
              <i className="fa fa-times" aria-hidden="true"></i>
            </span>
          </div>
        </div>
      )
    }

    return (
      <li
        className={classnames("list-group-item", {'list-group-item-success': filter.active})}
        onClick={() => track({id: filter.id, phrase: filter.text})}>
        {element}
      </li>
    )
  }
}
```

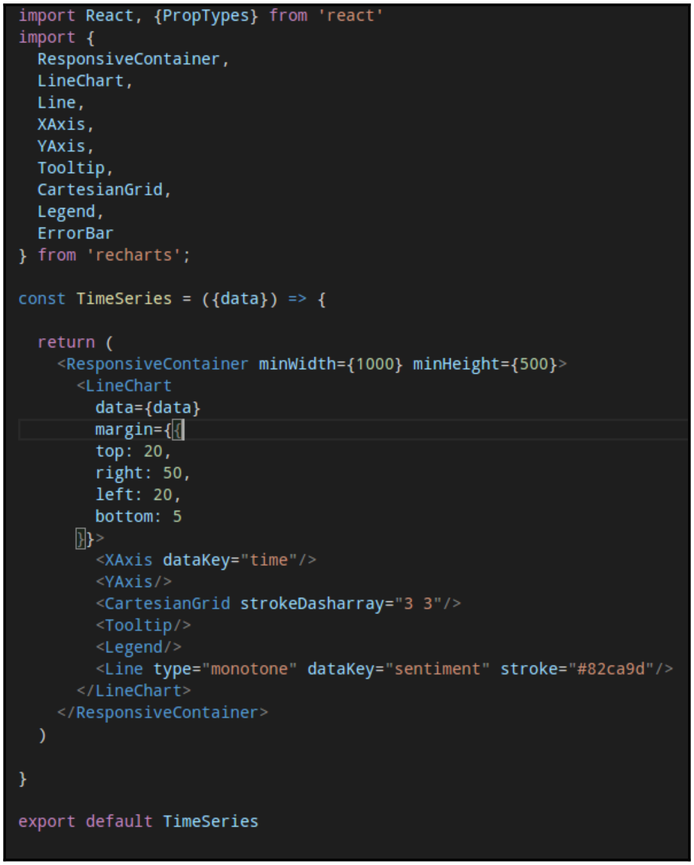


```
class Tweets extends React.Component {

  render() {
    const {tweets, filters} = this.props
    let filteredTweets = []
    let activeFilter = filters.filter(filter => filter.active === true)
    if (activeFilter[0]) {
      filteredTweets = tweets.filter(tweet => tweet.phrase === activeFilter[0].text);
    }
    return (
      <div>
        <TweetsList tweets={filteredTweets}/>
      </div>
    )
  }
}

Tweets.propTypes = {
  tweets: PropTypes.array.isRequired,
  filters: PropTypes.array.isRequired
}
const mapStateToProps = (state, dispatch) => {
  return {tweets: state.tweets, filters: state.filters}
}
const generateComponent = connect(mapStateToProps)

export default generateComponent(Tweets);
```

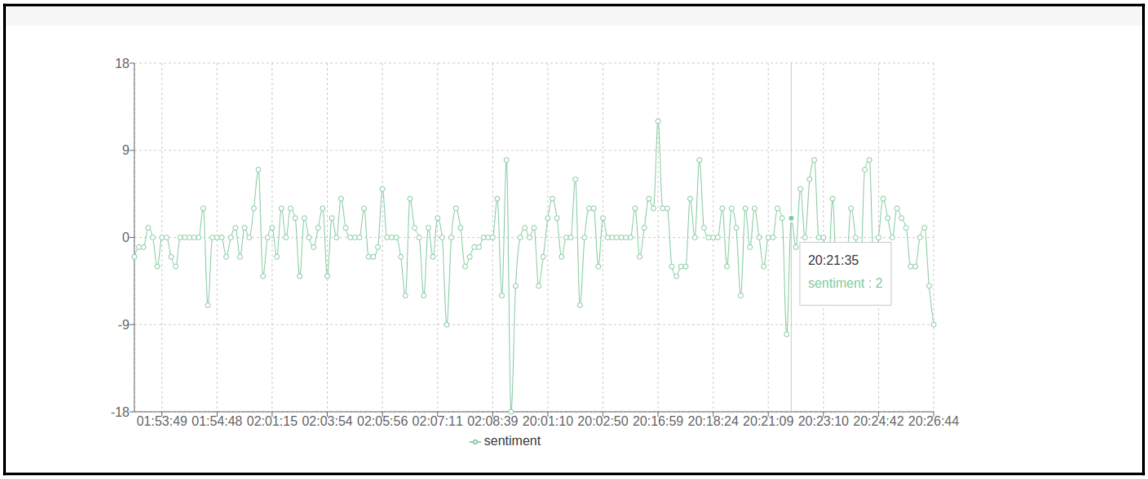




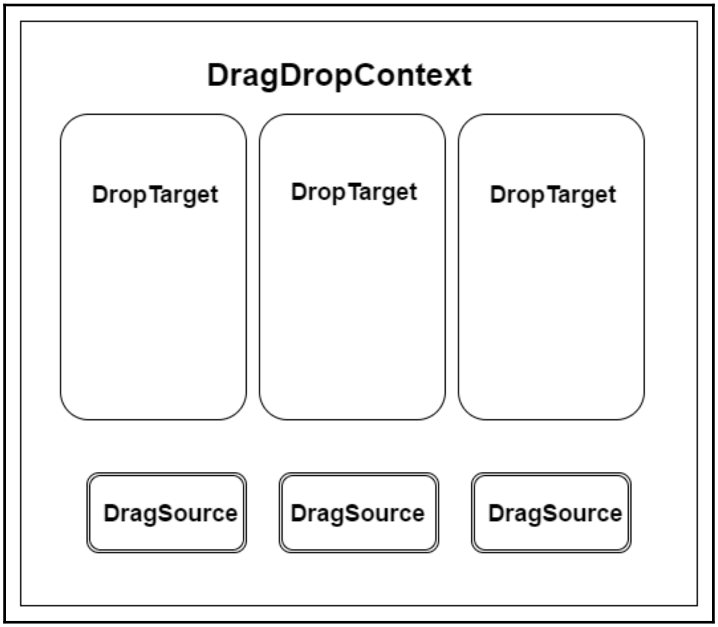
```
import React, {PropTypes} from 'react'
import {Link} from 'react-router'
import {connect} from 'react-redux'
import TimeSeries from '../components/TimeSeries'

class Sentiment extends React.Component {

    render() {
        const {tweets, filters} = this.props
        let filteredTweets = []
        let activeFilter = filters.filter(filter => filter.active === true)
        if (activeFilter[0]) {
            filteredTweets = tweets.filter(tweet => tweet.phrase === activeFilter[0].text);
        }
        return (
            <div>
                <TimeSeries data={filteredTweets}/>
            </div>
        )
    }
}

Sentiment.propTypes = {
    tweets: PropTypes.array.isRequired,
    filters: PropTypes.array.isRequired
}
const mapStateToProps = (state) => ({tweets: state.tweets, filters: state.filters})

export default connect(mapStateToProps)(Sentiment)
```

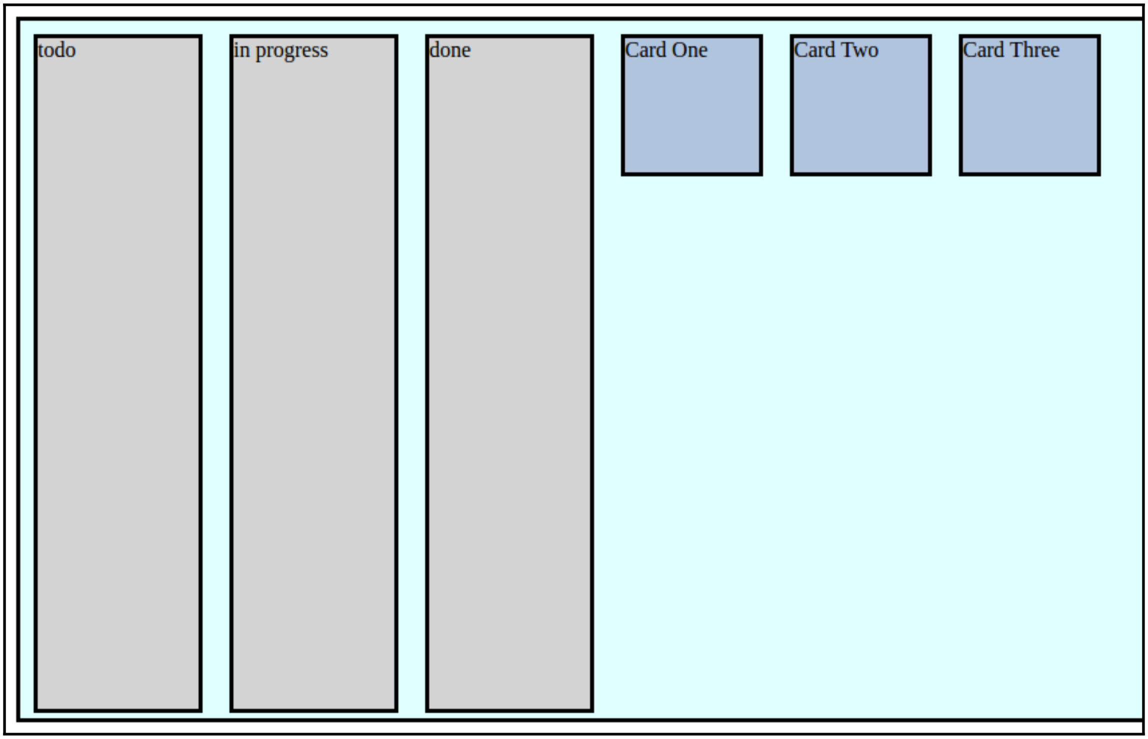


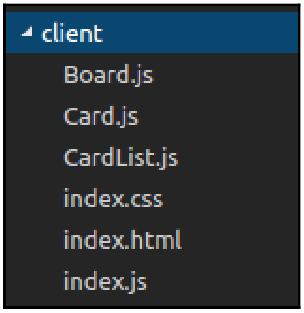
```
import React, {PropTypes} from 'react'
import {
  ResponsiveContainer,
  LineChart,
  Line,
  XAxis,
  YAxis,
  Tooltip,
  CartesianGrid,
  Legend,
  ErrorBar
} from 'recharts';

const TimeSeries = ({data}) => {

  return (
    <ResponsiveContainer minWidth={1000} minHeight={500}>
      <LineChart
        data={data}
        margin={{
        top: 20,
        right: 50,
        left: 20,
        bottom: 5
      }}>
        <XAxis dataKey="time"/>
        <YAxis/>
        <CartesianGrid strokeDasharray="3 3"/>
        <Tooltip/>
        <Legend/>
        <Line type="monotone" dataKey="sentiment" stroke="#82ca9d"/>
      </LineChart>
    </ResponsiveContainer>
  )

}

export default TimeSeries
```

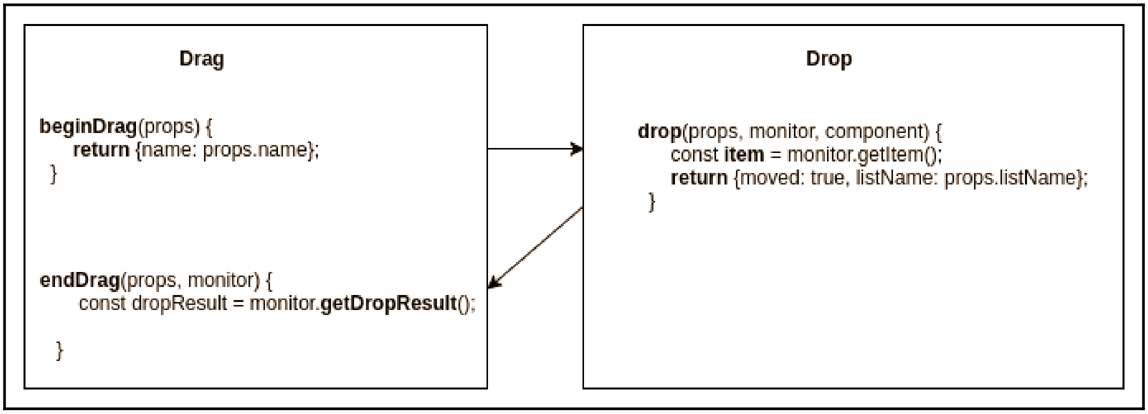


Chapter 5:
Developing Kanban Project Management To ol



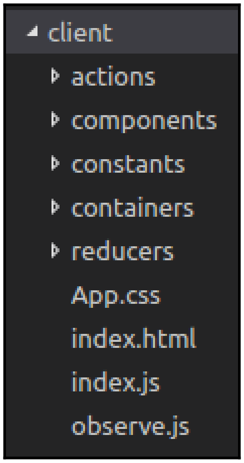

todo
in progress
done
Card One
Card Two
Card Three


client
Board.js
Card.js
CardList.js
index.css
index.html
index.js

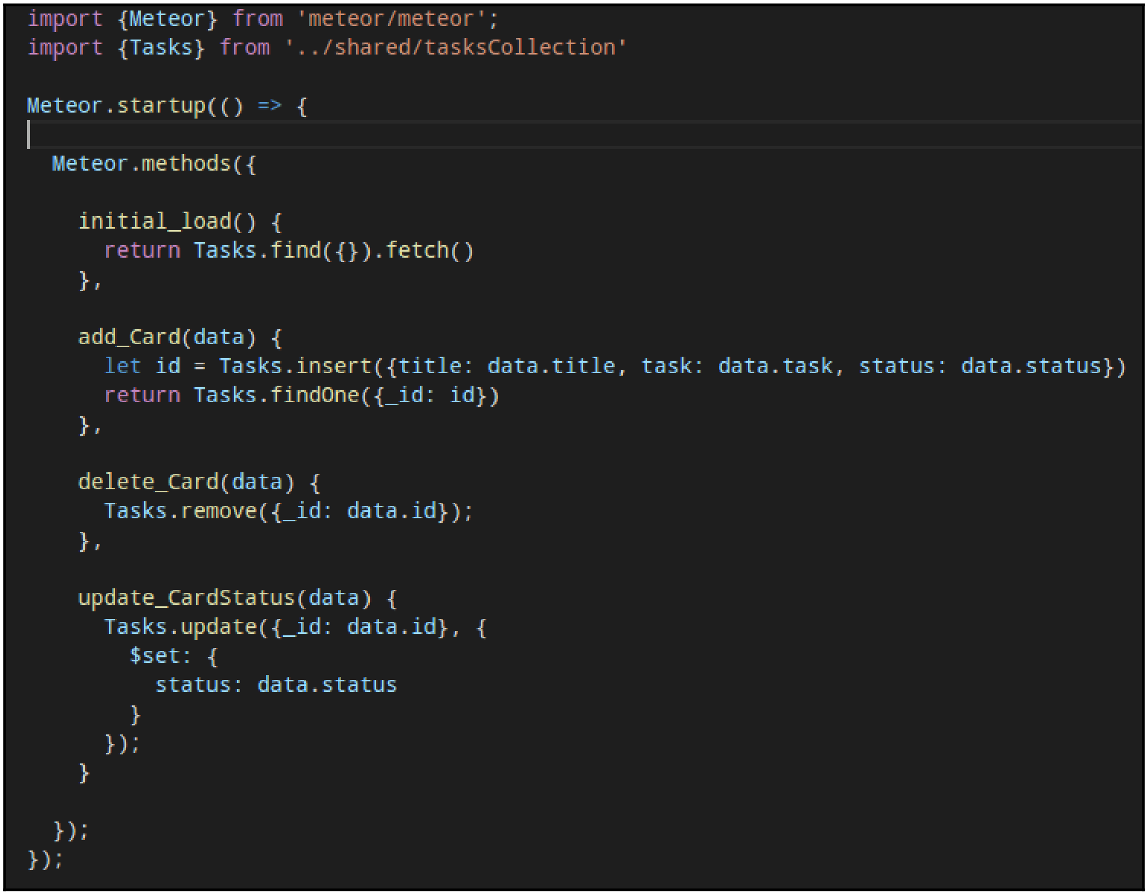


```
beginDrag(props) {
     return {name: props.name};
 }

endDrag(props, monitor) {
      const dropResult = monitor.getDropResult();

  }
```

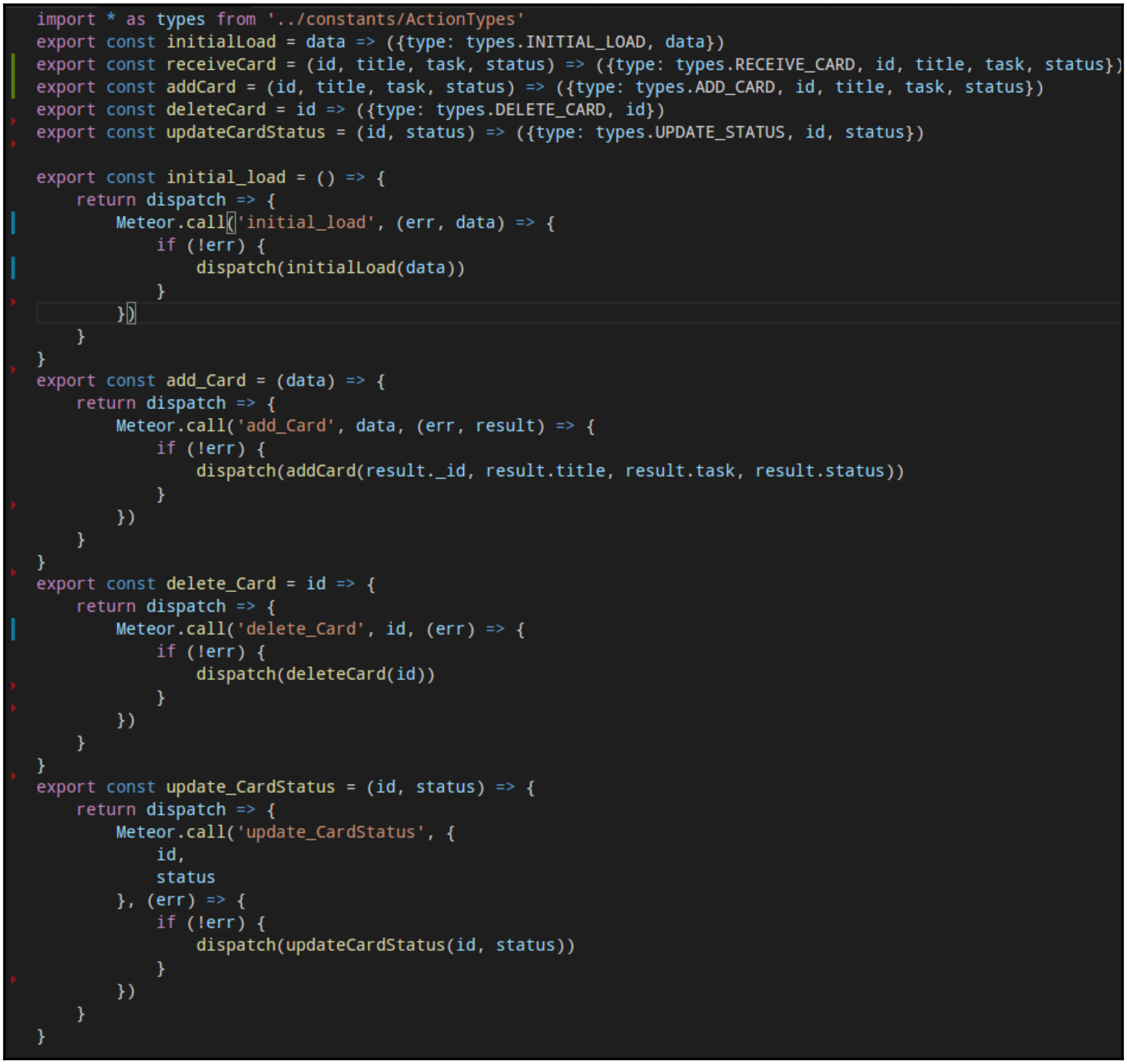
```
drop(props, monitor, component) {
     const item = monitor.getItem();
     return {moved: true, listName: props.listName};
  }
```



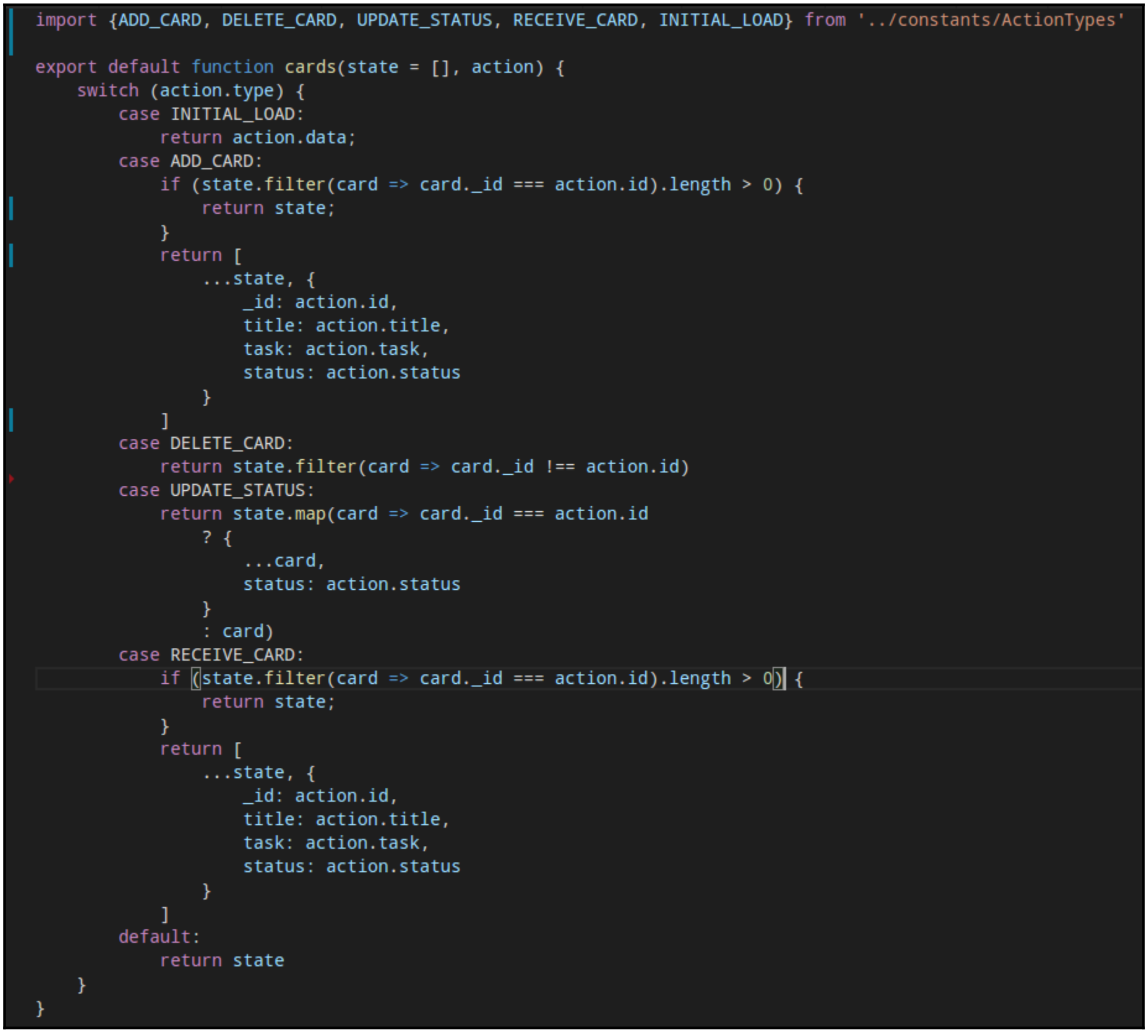
```
◢ client
   ▷ actions
   ▷ components
   ▷ constants
   ▷ containers
   ▷ reducers
     App.css
     index.html
     index.js
     observe.js
```



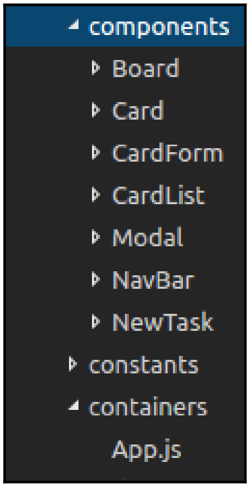
```
import {Meteor} from 'meteor/meteor';
import {Tasks} from '../shared/tasksCollection'

Meteor.startup(() => {

  Meteor.methods({

    initial_load() {
      return Tasks.find({}).fetch()
    },

    add_Card(data) {
      let id = Tasks.insert({title: data.title, task: data.task, status: data.status})
      return Tasks.findOne({_id: id})
    },

    delete_Card(data) {
      Tasks.remove({_id: data.id});
    },

    update_CardStatus(data) {
      Tasks.update({_id: data.id}, {
        $set: {
          status: data.status
        }
      });
    }

  });
});
```



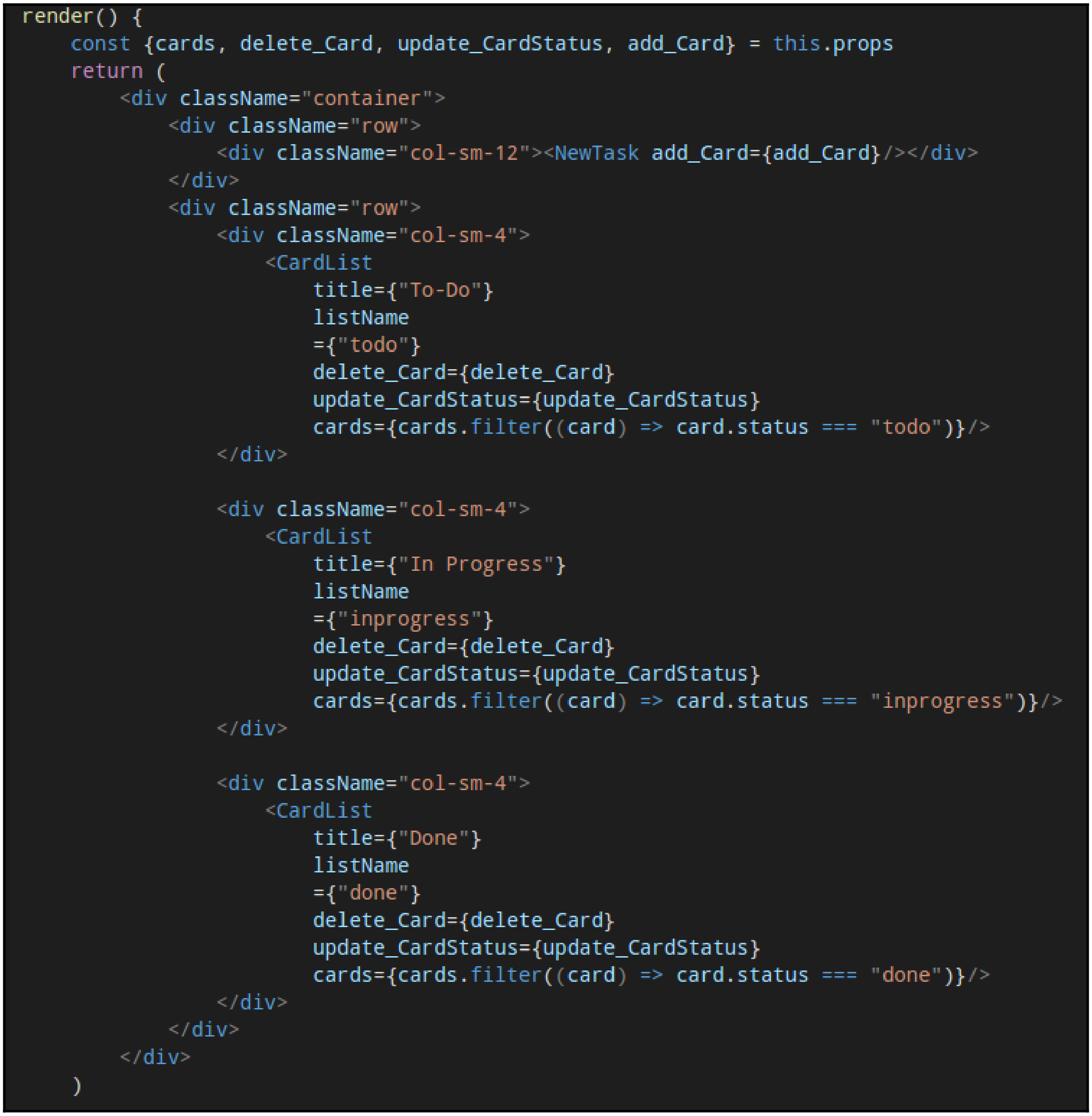
```
import * as types from '../constants/ActionTypes'
export const initialLoad = data => ({type: types.INITIAL_LOAD, data})
export const receiveCard = (id, title, task, status) => ({type: types.RECEIVE_CARD, id, title, task, status})
export const addCard = (id, title, task, status) => ({type: types.ADD_CARD, id, title, task, status})
export const deleteCard = id => ({type: types.DELETE_CARD, id})
export const updateCardStatus = (id, status) => ({type: types.UPDATE_STATUS, id, status})

export const initial_load = () => {
    return dispatch => {
        Meteor.call('initial_load', (err, data) => {
            if (!err) {
                dispatch(initialLoad(data))
            }
        })
    }
}
export const add_Card = (data) => {
    return dispatch => {
        Meteor.call('add_Card', data, (err, result) => {
            if (!err) {
                dispatch(addCard(result._id, result.title, result.task, result.status))
            }
        })
    }
}
export const delete_Card = id => {
    return dispatch => {
        Meteor.call('delete_Card', id, (err) => {
            if (!err) {
                dispatch(deleteCard(id))
            }
        })
    }
}
export const update_CardStatus = (id, status) => {
    return dispatch => {
        Meteor.call('update_CardStatus', {
            id,
            status
        }, (err) => {
            if (!err) {
                dispatch(updateCardStatus(id, status))
            }
        })
    }
}
```



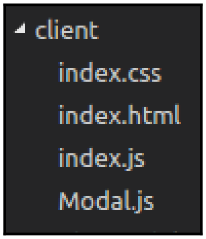
```
import {ADD_CARD, DELETE_CARD, UPDATE_STATUS, RECEIVE_CARD, INITIAL_LOAD} from '../constants/ActionTypes'

export default function cards(state = [], action) {
    switch (action.type) {
        case INITIAL_LOAD:
            return action.data;
        case ADD_CARD:
            if (state.filter(card => card._id === action.id).length > 0) {
                return state;
            }
            return [
                ...state, {
                    _id: action.id,
                    title: action.title,
                    task: action.task,
                    status: action.status
                }
            ]
        case DELETE_CARD:
            return state.filter(card => card._id !== action.id)
        case UPDATE_STATUS:
            return state.map(card => card._id === action.id
                ? {
                    ...card,
                    status: action.status
                }
                : card)
        case RECEIVE_CARD:
            if (state.filter(card => card._id === action.id).length > 0) {
                return state;
            }
            return [
                ...state, {
                    _id: action.id,
                    title: action.title,
                    task: action.task,
                    status: action.status
                }
            ]
        default:
            return state
    }
}
```



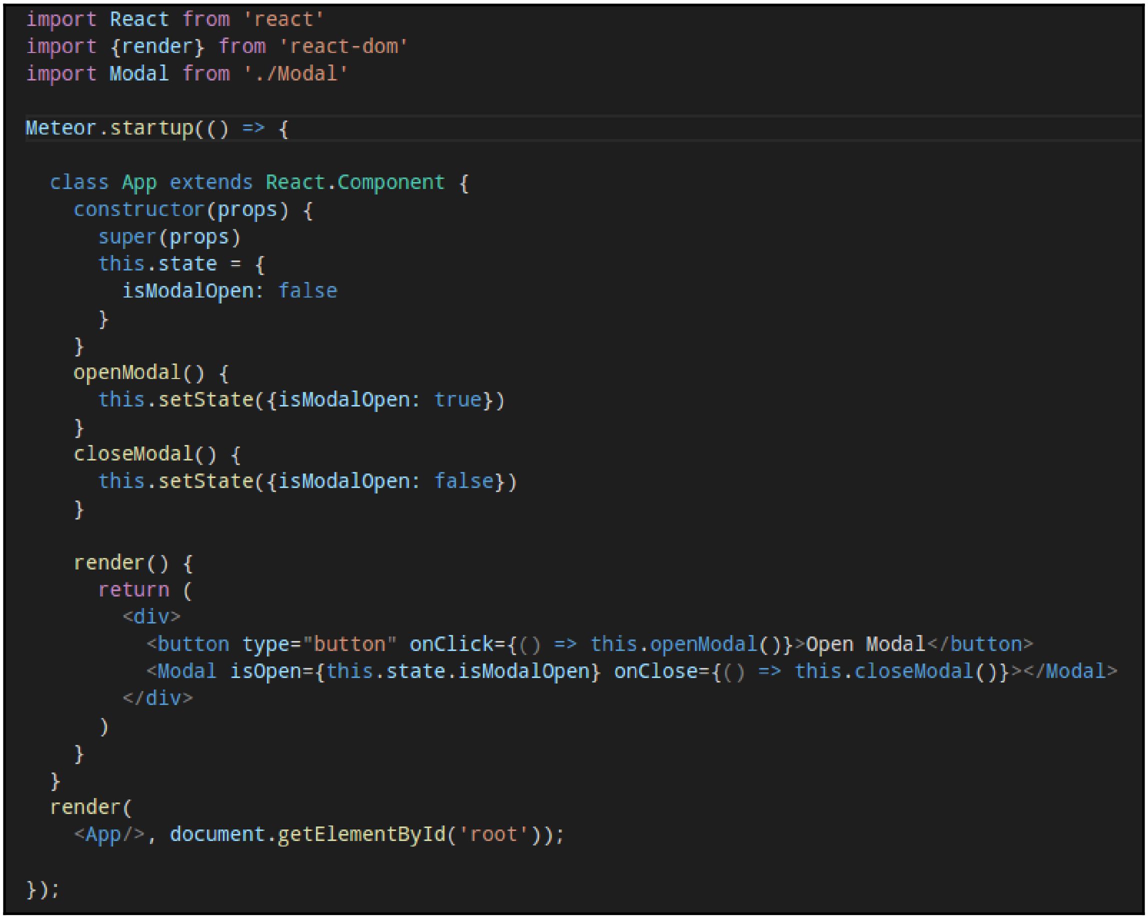
```
◢ components
    ▷ Board
    ▷ Card
    ▷ CardForm
    ▷ CardList
    ▷ Modal
    ▷ NavBar
    ▷ NewTask
▷ constants
◢ containers
    App.js
```



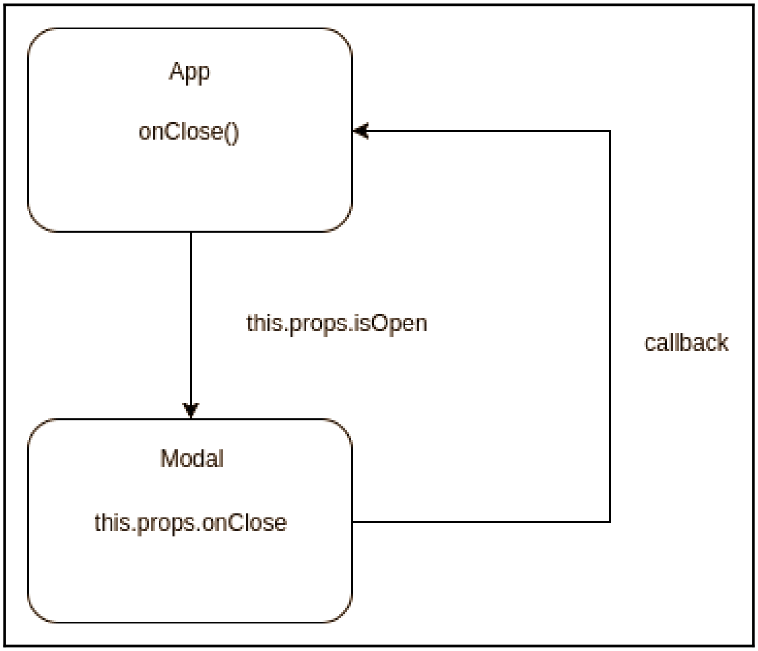
```
render() {
    const {cards, delete_Card, update_CardStatus, add_Card} = this.props
    return (
        <div className="container">
            <div className="row">
                <div className="col-sm-12"><NewTask add_Card={add_Card}/></div>
            </div>
            <div className="row">
                <div className="col-sm-4">
                    <CardList
                        title={"To-Do"}
                        listName
                        ={"todo"}
                        delete_Card={delete_Card}
                        update_CardStatus={update_CardStatus}
                        cards={cards.filter((card) => card.status === "todo")}/>
                </div>

                <div className="col-sm-4">
                    <CardList
                        title={"In Progress"}
                        listName
                        ={"inprogress"}
                        delete_Card={delete_Card}
                        update_CardStatus={update_CardStatus}
                        cards={cards.filter((card) => card.status === "inprogress")}/>
                </div>

                <div className="col-sm-4">
                    <CardList
                        title={"Done"}
                        listName
                        ={"done"}
                        delete_Card={delete_Card}
                        update_CardStatus={update_CardStatus}
                        cards={cards.filter((card) => card.status === "done")}/>
                </div>
            </div>
        </div>
    )
```

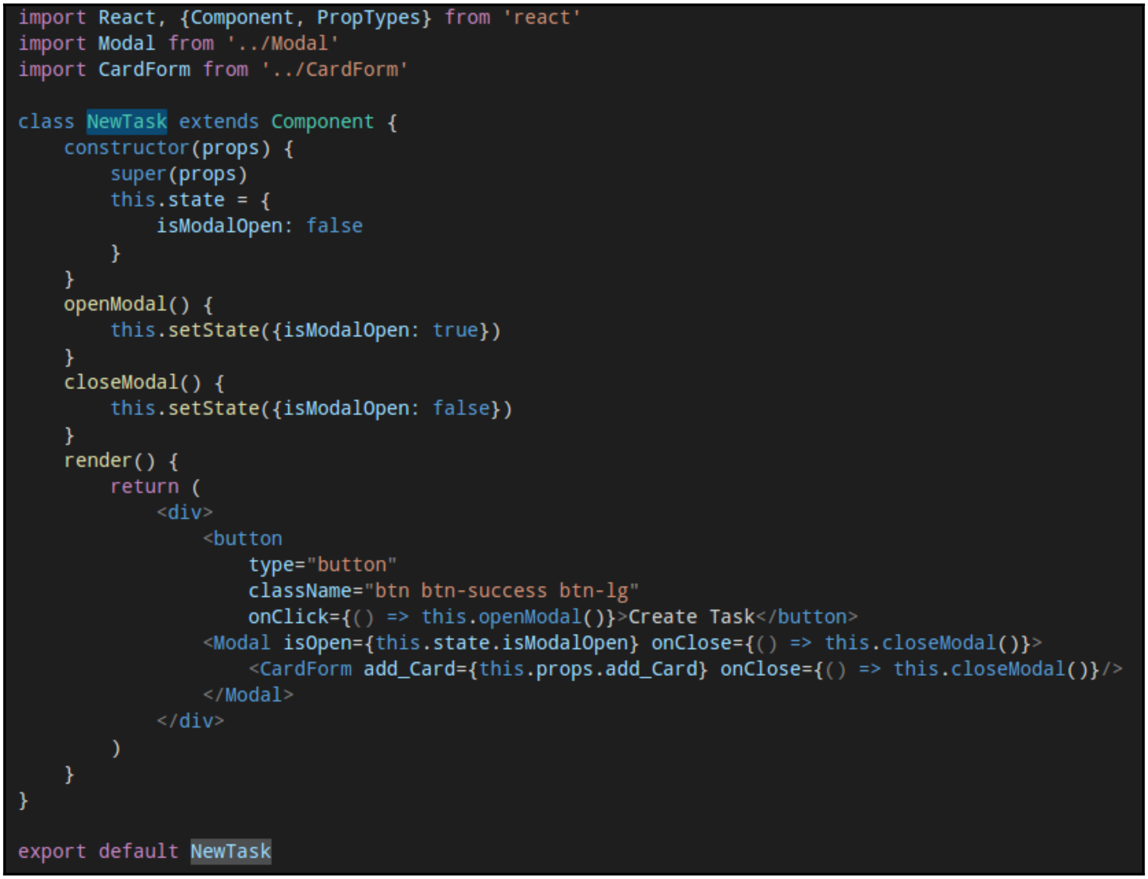


```
◢ client
    index.css
    index.html
    index.js
    Modal.js
```

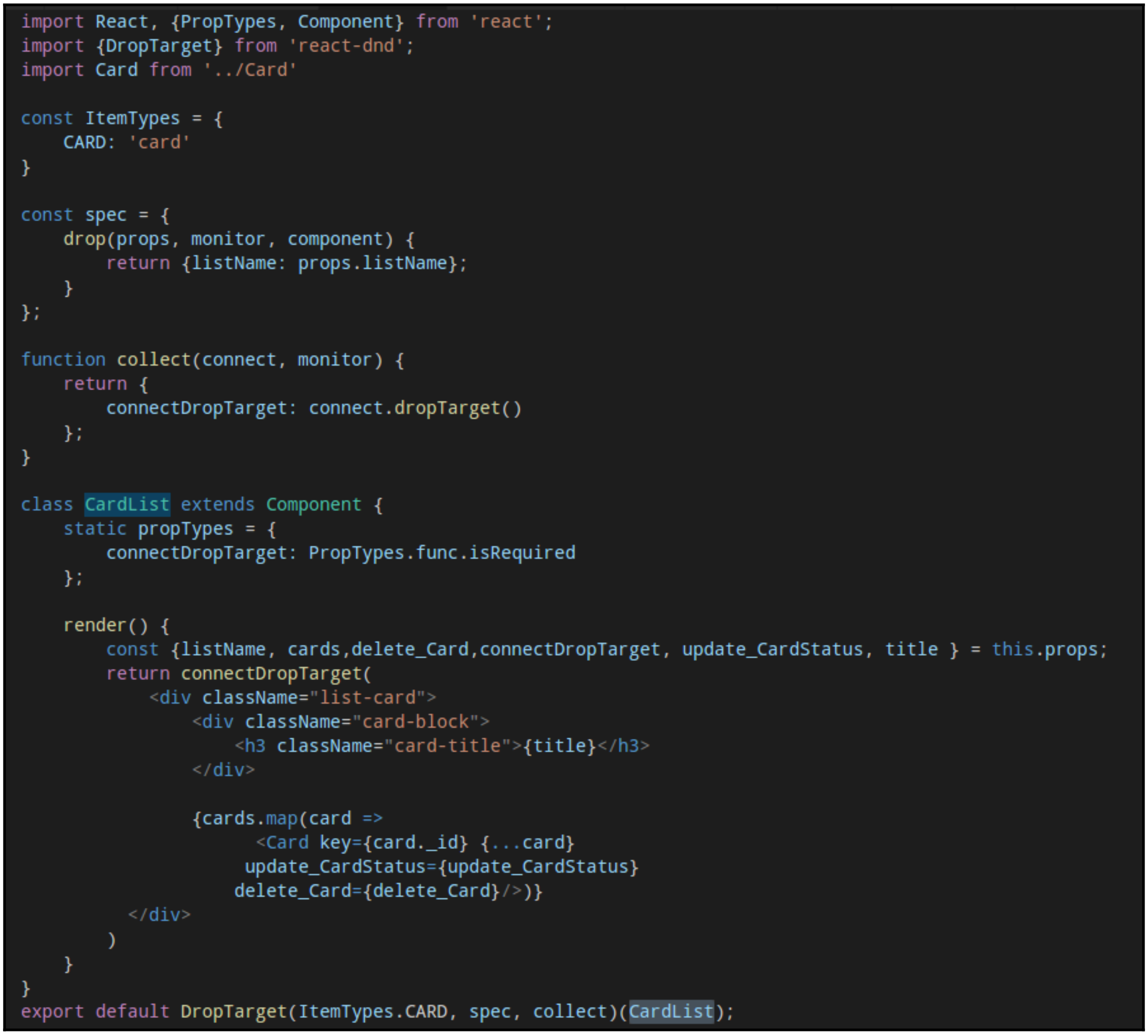


```
import React from 'react'
import {render} from 'react-dom'
import Modal from './Modal'

Meteor.startup(() => {

  class App extends React.Component {
    constructor(props) {
      super(props)
      this.state = {
        isModalOpen: false
      }
    }
    openModal() {
      this.setState({isModalOpen: true})
    }
    closeModal() {
      this.setState({isModalOpen: false})
    }

    render() {
      return (
        <div>
          <button type="button" onClick={() => this.openModal()}>Open Modal</button>
          <Modal isOpen={this.state.isModalOpen} onClose={() => this.closeModal()}></Modal>
        </div>
      )
    }
  }
  render(
    <App/>, document.getElementById('root'));

});
```

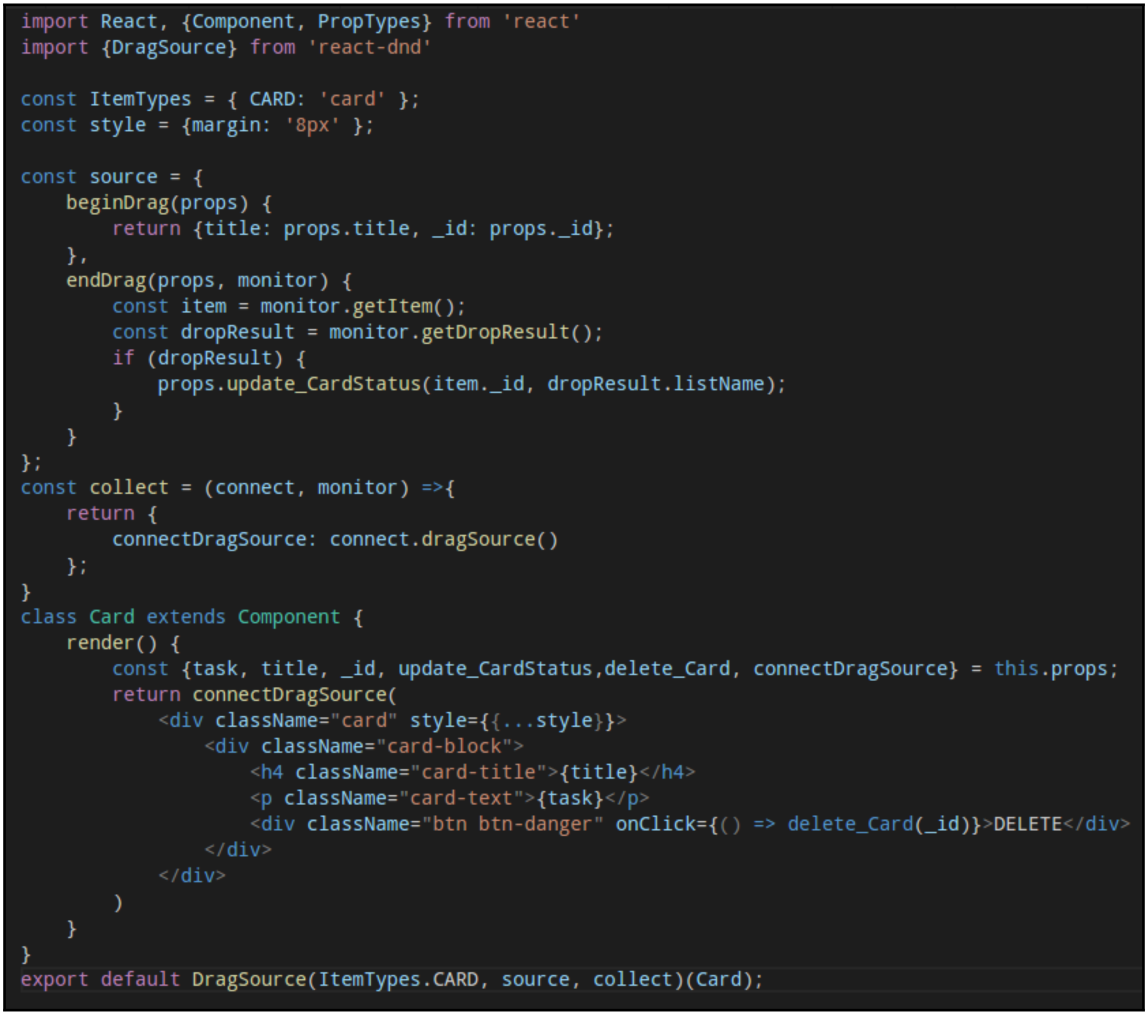

App
onClose()
this.props.isOpen
callback
Modal
this.props.onClose



```
import React, {Component, PropTypes} from 'react'
import Modal from '../Modal'
import CardForm from '../CardForm'

class NewTask extends Component {
    constructor(props) {
        super(props)
        this.state = {
            isModalOpen: false
        }
    }
    openModal() {
        this.setState({isModalOpen: true})
    }
    closeModal() {
        this.setState({isModalOpen: false})
    }
    render() {
        return (
            <div>
                <button
                    type="button"
                    className="btn btn-success btn-lg"
                    onClick={() => this.openModal()}>Create Task</button>
                <Modal isOpen={this.state.isModalOpen} onClose={() => this.closeModal()}>
                    <CardForm add_Card={this.props.add_Card} onClose={() => this.closeModal()}/>
                </Modal>
            </div>
        )
    }
}

export default NewTask
```

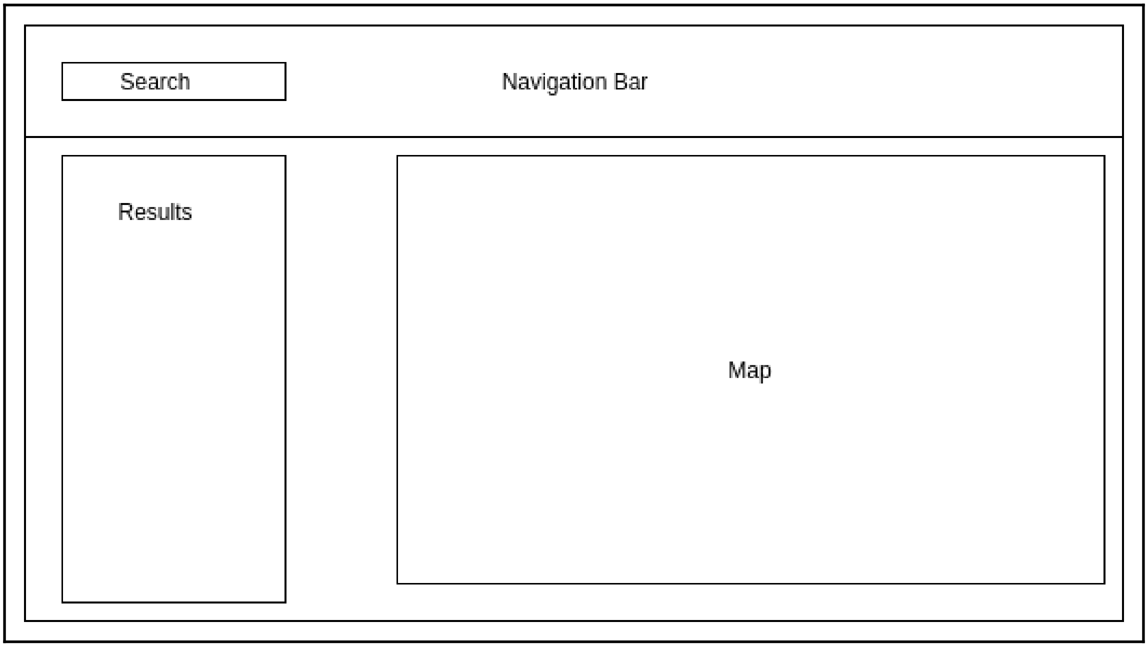


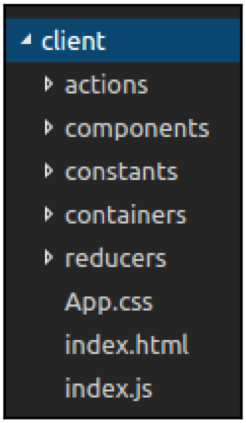
```
import React, {PropTypes, Component} from 'react';
import {DropTarget} from 'react-dnd';
import Card from '../Card'

const ItemTypes = {
    CARD: 'card'
}

const spec = {
    drop(props, monitor, component) {
        return {listName: props.listName};
    }
};

function collect(connect, monitor) {
    return {
        connectDropTarget: connect.dropTarget()
    };
}

class CardList extends Component {
    static propTypes = {
        connectDropTarget: PropTypes.func.isRequired
    };

    render() {
        const {listName, cards,delete_Card,connectDropTarget, update_CardStatus, title } = this.props;
        return connectDropTarget(
            <div className="list-card">
                <div className="card-block">
                    <h3 className="card-title">{title}</h3>
                </div>

                {cards.map(card =>
                      <Card key={card._id} {...card}
                     update_CardStatus={update_CardStatus}
                    delete_Card={delete_Card}/>)}
          </div>
        )
    }
}
export default DropTarget(ItemTypes.CARD, spec, collect)(CardList);
```



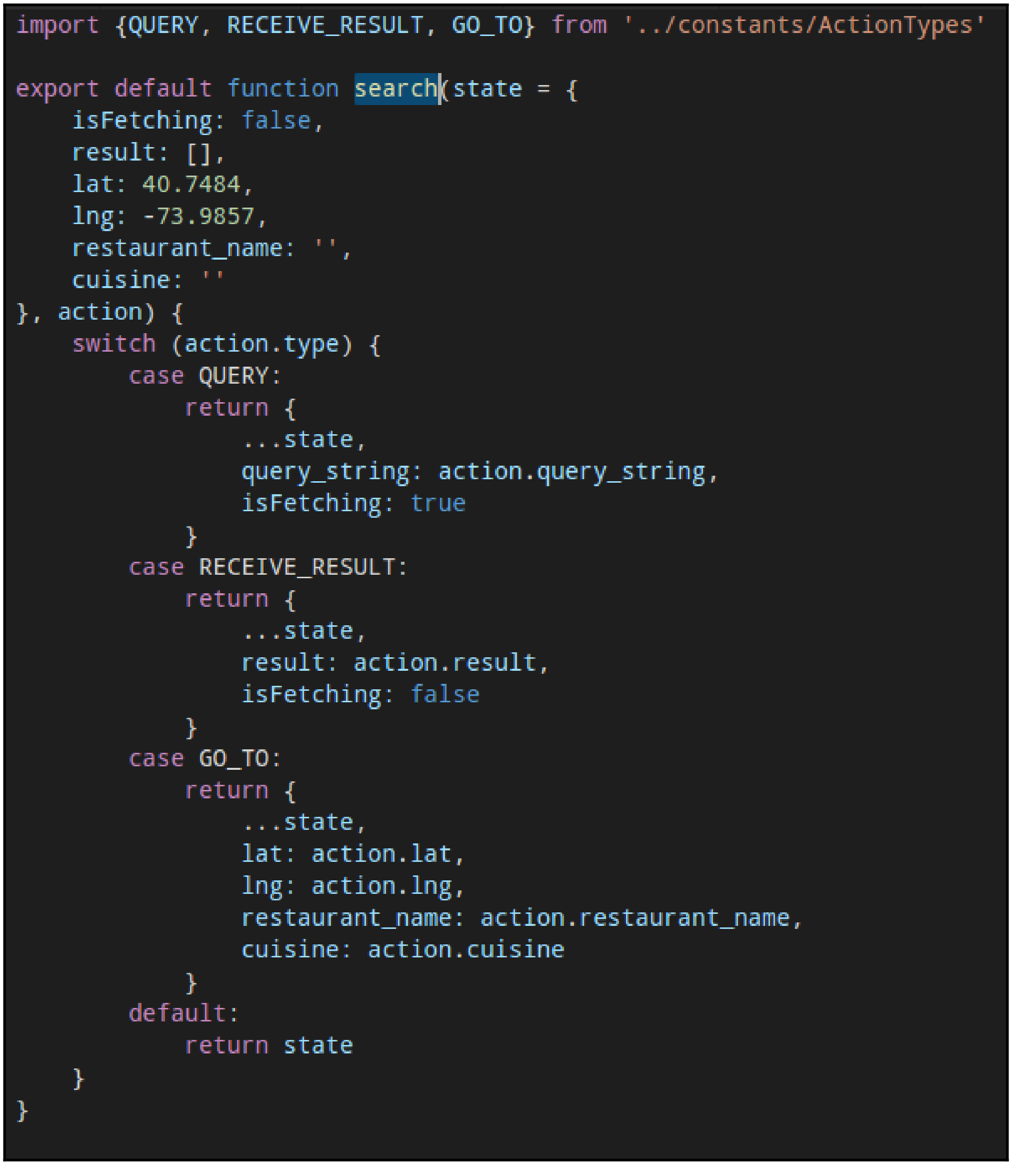
```
import React, {Component, PropTypes} from 'react'
import {DragSource} from 'react-dnd'

const ItemTypes = { CARD: 'card' };
const style = {margin: '8px' };

const source = {
    beginDrag(props) {
        return {title: props.title, _id: props._id};
    },
    endDrag(props, monitor) {
        const item = monitor.getItem();
        const dropResult = monitor.getDropResult();
        if (dropResult) {
            props.update_CardStatus(item._id, dropResult.listName);
        }
    }
};
const collect = (connect, monitor) =>{
    return {
        connectDragSource: connect.dragSource()
    };
}
class Card extends Component {
    render() {
        const {task, title, _id, update_CardStatus,delete_Card, connectDragSource} = this.props;
        return connectDragSource(
            <div className="card" style={{...style}}>
                <div className="card-block">
                    <h4 className="card-title">{title}</h4>
                    <p className="card-text">{task}</p>
                    <div className="btn btn-danger" onClick={() => delete_Card(_id)}>DELETE</div>
                </div>
            </div>
        )
    }
}
export default DragSource(ItemTypes.CARD, source, collect)(Card);
```

Chapter 6: Building a Real-Time Search Application

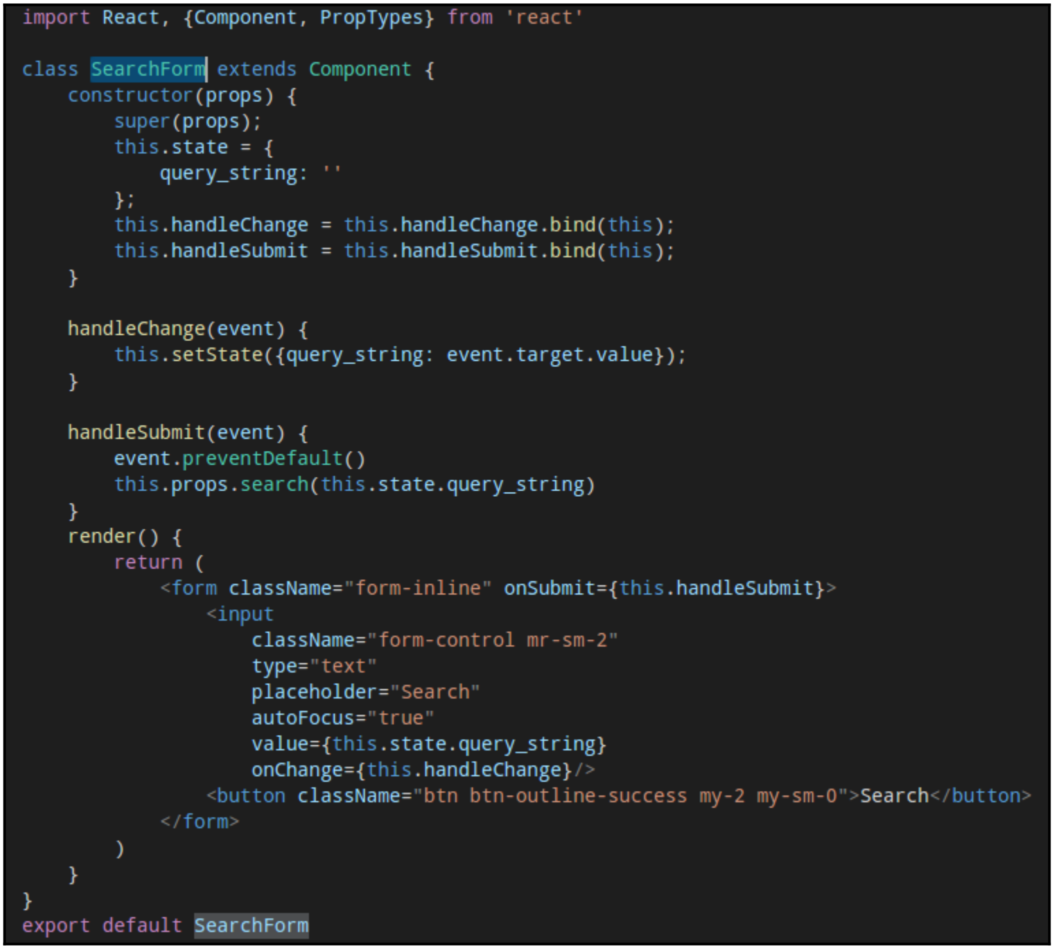




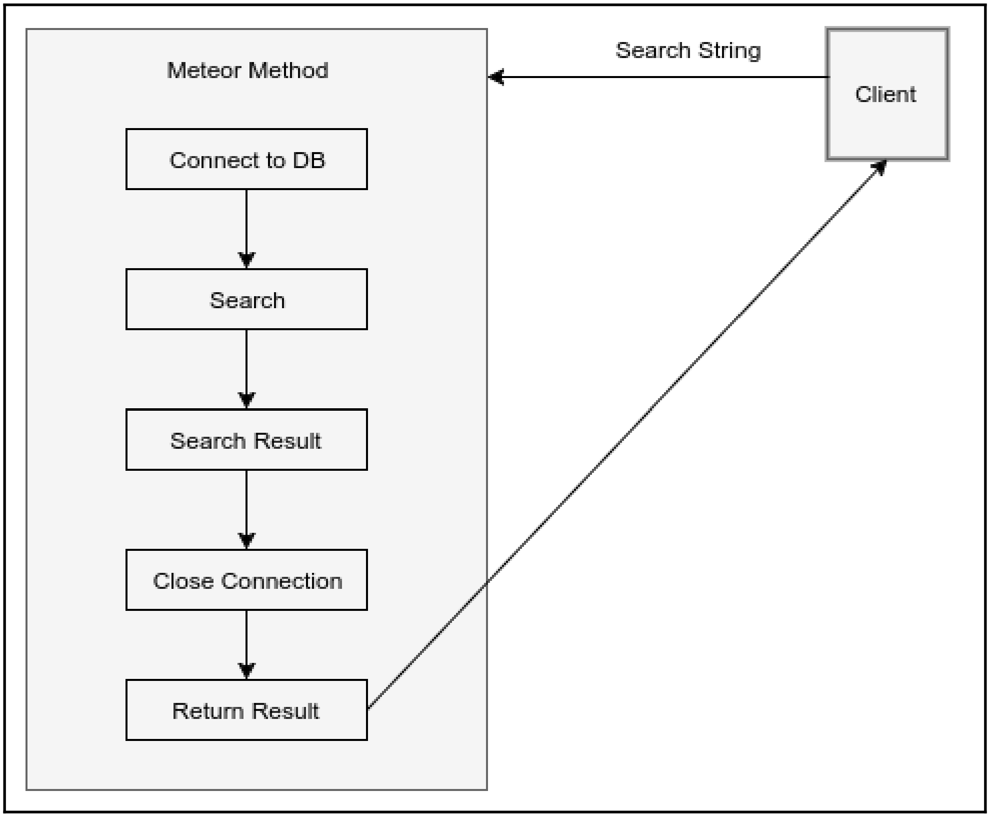
```
◢ client
   ▸ actions
   ▸ components
   ▸ constants
   ▸ containers
   ▸ reducers
     App.css
     index.html
     index.js
```

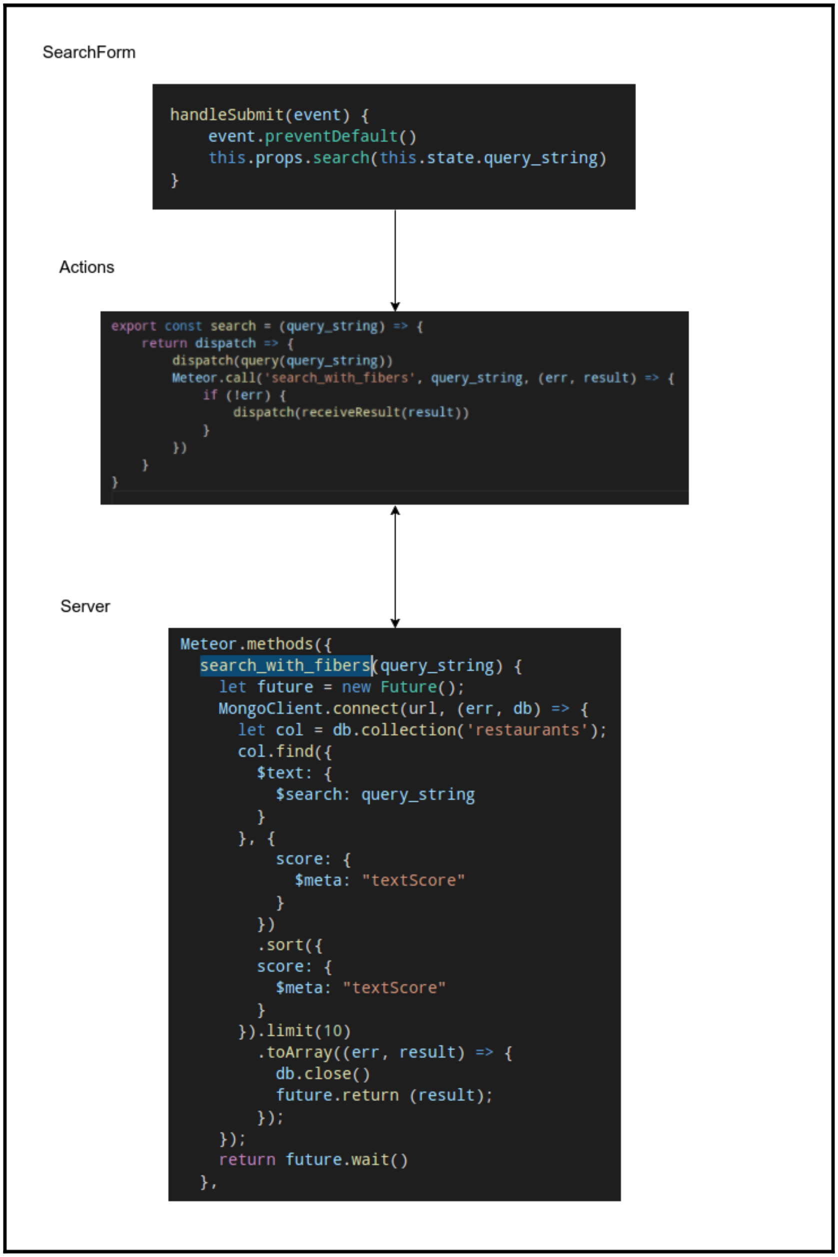


```
import {QUERY, RECEIVE_RESULT, GO_TO} from '../constants/ActionTypes'

export default function search(state = {
    isFetching: false,
    result: [],
    lat: 40.7484,
    lng: -73.9857,
    restaurant_name: '',
    cuisine: ''
}, action) {
    switch (action.type) {
        case QUERY:
            return {
                ...state,
                query_string: action.query_string,
                isFetching: true
            }
        case RECEIVE_RESULT:
            return {
                ...state,
                result: action.result,
                isFetching: false
            }
        case GO_TO:
            return {
                ...state,
                lat: action.lat,
                lng: action.lng,
                restaurant_name: action.restaurant_name,
                cuisine: action.cuisine
            }
        default:
            return state
    }
}
```



```
import React, {Component, PropTypes} from 'react'

class SearchForm extends Component {
    constructor(props) {
        super(props);
        this.state = {
            query_string: ''
        };
        this.handleChange = this.handleChange.bind(this);
        this.handleSubmit = this.handleSubmit.bind(this);
    }

    handleChange(event) {
        this.setState({query_string: event.target.value});
    }

    handleSubmit(event) {
        event.preventDefault()
        this.props.search(this.state.query_string)
    }
    render() {
        return (
            <form className="form-inline" onSubmit={this.handleSubmit}>
                <input
                    className="form-control mr-sm-2"
                    type="text"
                    placeholder="Search"
                    autoFocus="true"
                    value={this.state.query_string}
                    onChange={this.handleChange}/>
                <button className="btn btn-outline-success my-2 my-sm-0">Search</button>
            </form>
        )
    }
}
export default SearchForm
```

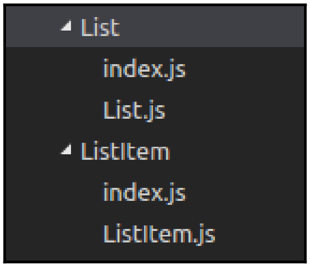

Meteor Method
Connect to DB
Search
Search Result
Close Connection
Return Result
Search String
Client



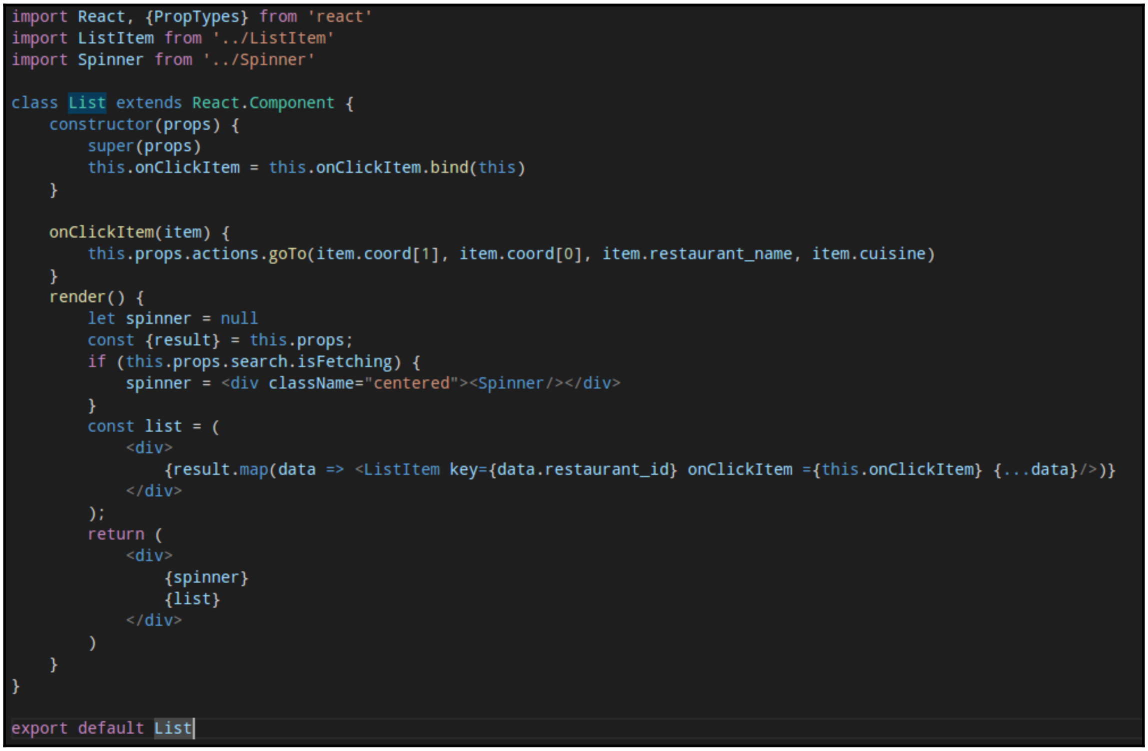
```
handleSubmit(event) {
    event.preventDefault()
    this.props.search(this.state.query_string)
}
```

```
export const search = (query_string) => {
    return dispatch => {
        dispatch(query(query_string))
        Meteor.call('search_with_fibers', query_string, (err, result) => {
            if (!err) {
                dispatch(receiveResult(result))
            }
        })
    }
}
```

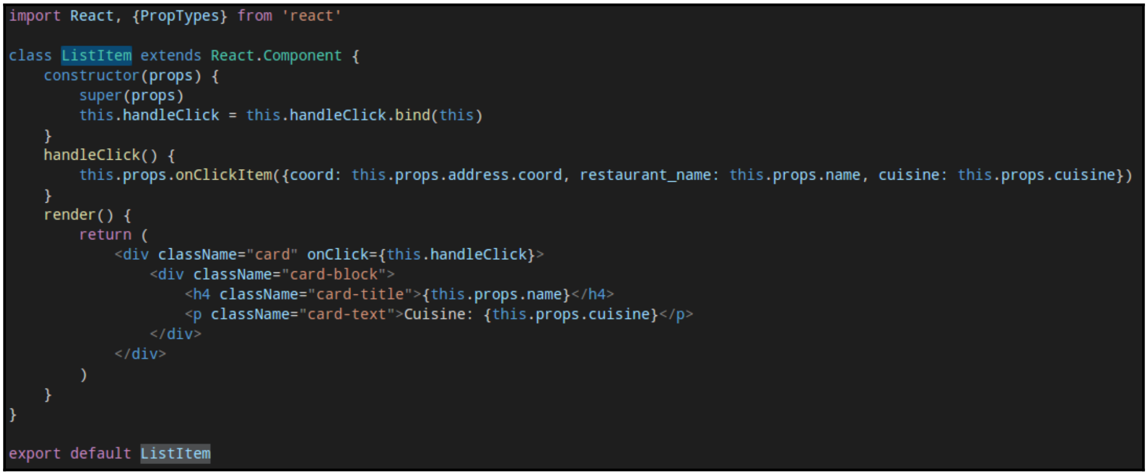
```
Meteor.methods({
  search_with_fibers(query_string) {
    let future = new Future();
    MongoClient.connect(url, (err, db) => {
      let col = db.collection('restaurants');
      col.find({
        $text: {
          $search: query_string
        }
      }, {
          score: {
            $meta: "textScore"
          }
        })
        .sort({
        score: {
          $meta: "textScore"
        }
      }).limit(10)
        .toArray((err, result) => {
          db.close()
          future.return (result);
        });
    });
    return future.wait()
  },
```



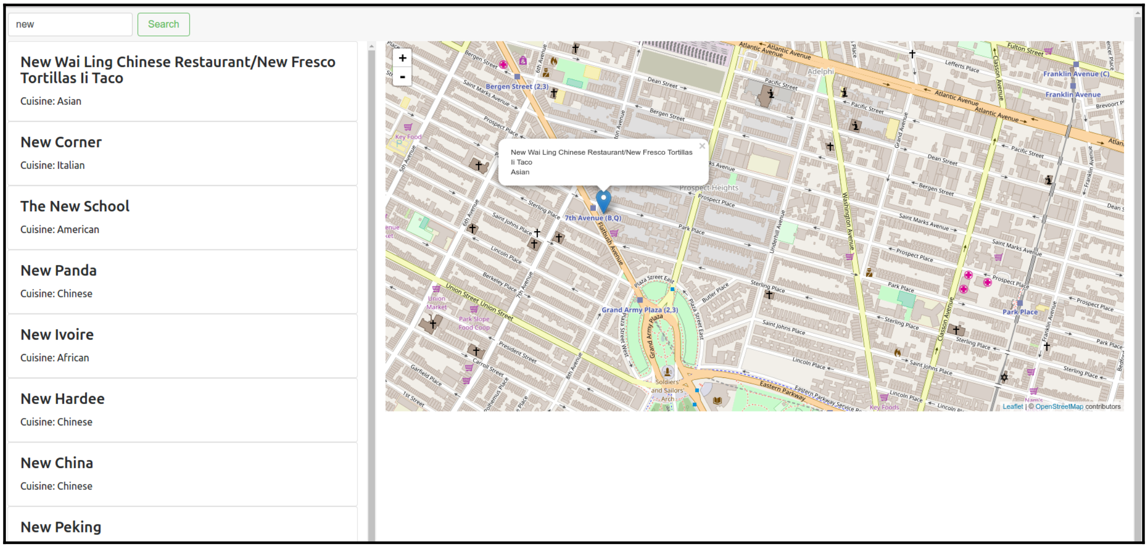
```
◢ List
    index.js
    List.js
◢ ListItem
    index.js
    ListItem.js
```



```
import React, {PropTypes} from 'react'
import ListItem from '../ListItem'
import Spinner from '../Spinner'

class List extends React.Component {
    constructor(props) {
        super(props)
        this.onClickItem = this.onClickItem.bind(this)
    }

    onClickItem(item) {
        this.props.actions.goTo(item.coord[1], item.coord[0], item.restaurant_name, item.cuisine)
    }
    render() {
        let spinner = null
        const {result} = this.props;
        if (this.props.search.isFetching) {
            spinner = <div className="centered"><Spinner/></div>
        }
        const list = (
            <div>
                {result.map(data => <ListItem key={data.restaurant_id} onClickItem ={this.onClickItem} {...data}/>)}
            </div>
        );
        return (
            <div>
                {spinner}
                {list}
            </div>
        )
    }
}

export default List
```



```
import React, {PropTypes} from 'react'

class ListItem extends React.Component {
    constructor(props) {
        super(props)
        this.handleClick = this.handleClick.bind(this)
    }
    handleClick() {
        this.props.onClickItem({coord: this.props.address.coord, restaurant_name: this.props.name, cuisine: this.props.cuisine})
    }
    render() {
        return (
            <div className="card" onClick={this.handleClick}>
                <div className="card-block">
                    <h4 className="card-title">{this.props.name}</h4>
                    <p className="card-text">Cuisine: {this.props.cuisine}</p>
                </div>
            </div>
        )
    }
}

export default ListItem
```



New Wai Ling Chinese Restaurant/New Fresco Tortillas Ii Taco

Cuisine: Asian

New Corner

Cuisine: Italian

The New School

Cuisine: American

New Panda

Cuisine: Chinese

New Ivoire

Cuisine: African

New Hardee

Cuisine: Chinese

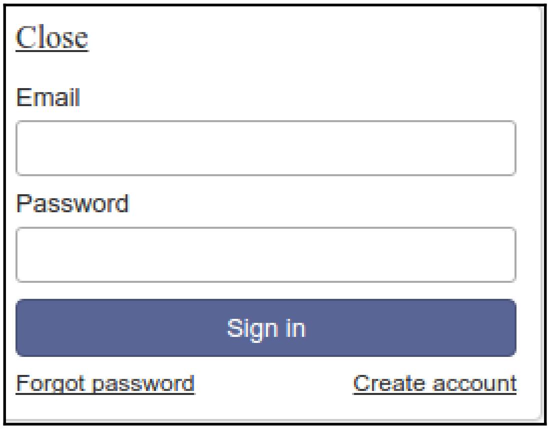
New China

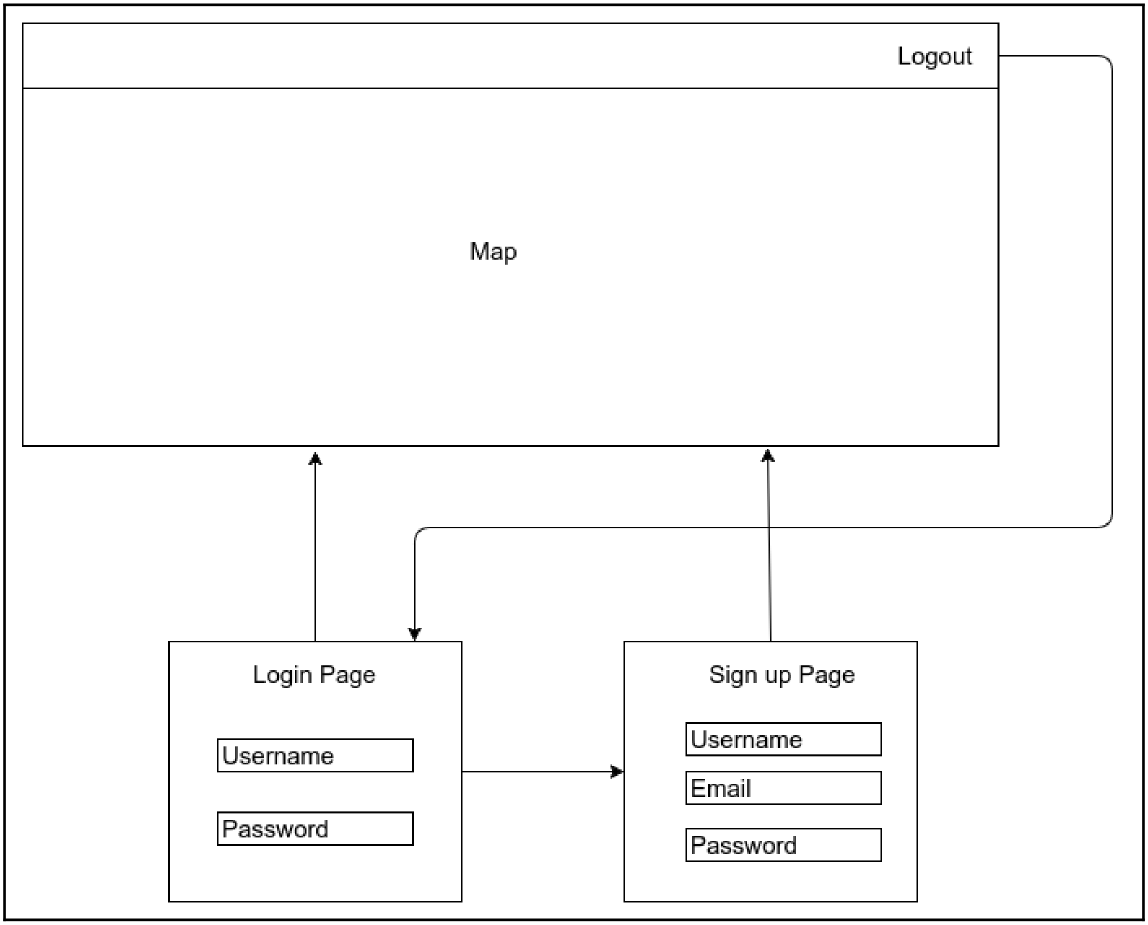
Cuisine: Chinese

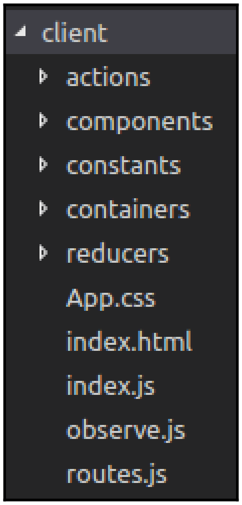
New Peking

New Wai Ling Chinese Restaurant/New Fresco Tortillas Ii Taco
Asian

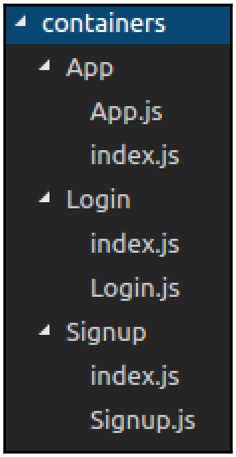
Chapter 7: Real-Time Maps



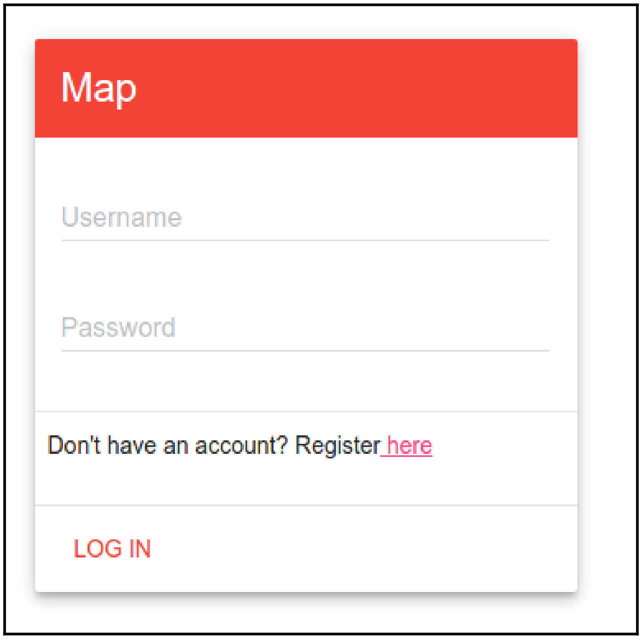

Logout
Map
Login Page
Username
Password
Sign up Page
Username
Email
Password



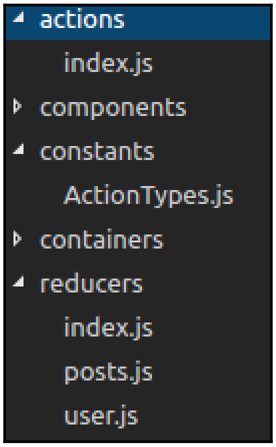
```
client
  actions
  components
  constants
  containers
  reducers
  App.css
  index.html
  index.js
  observe.js
  routes.js
```

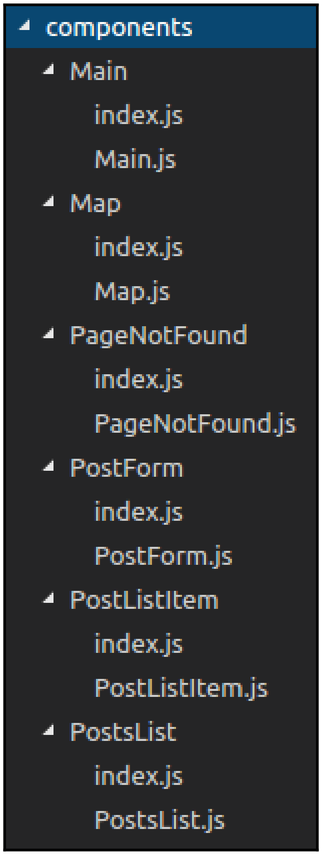


```
containers
  App
    App.js
    index.js
  Login
    index.js
    Login.js
  Signup
    index.js
    Signup.js
```

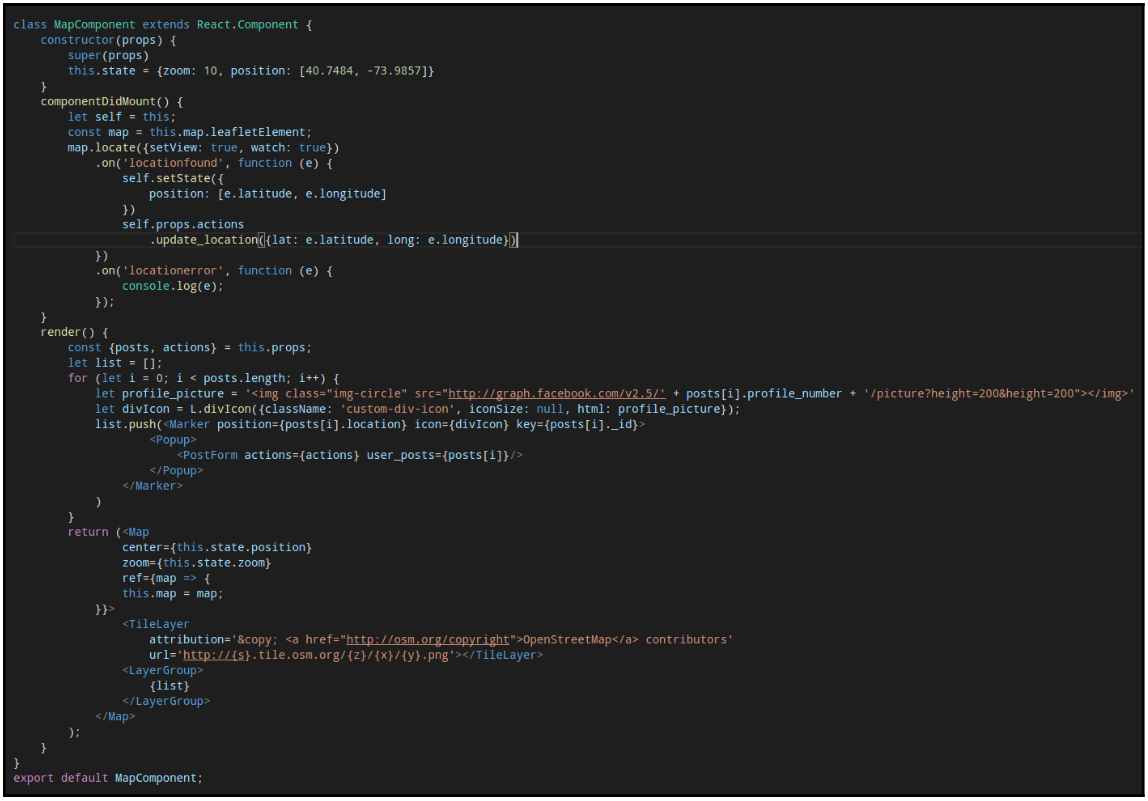


Don't have an account? Register here

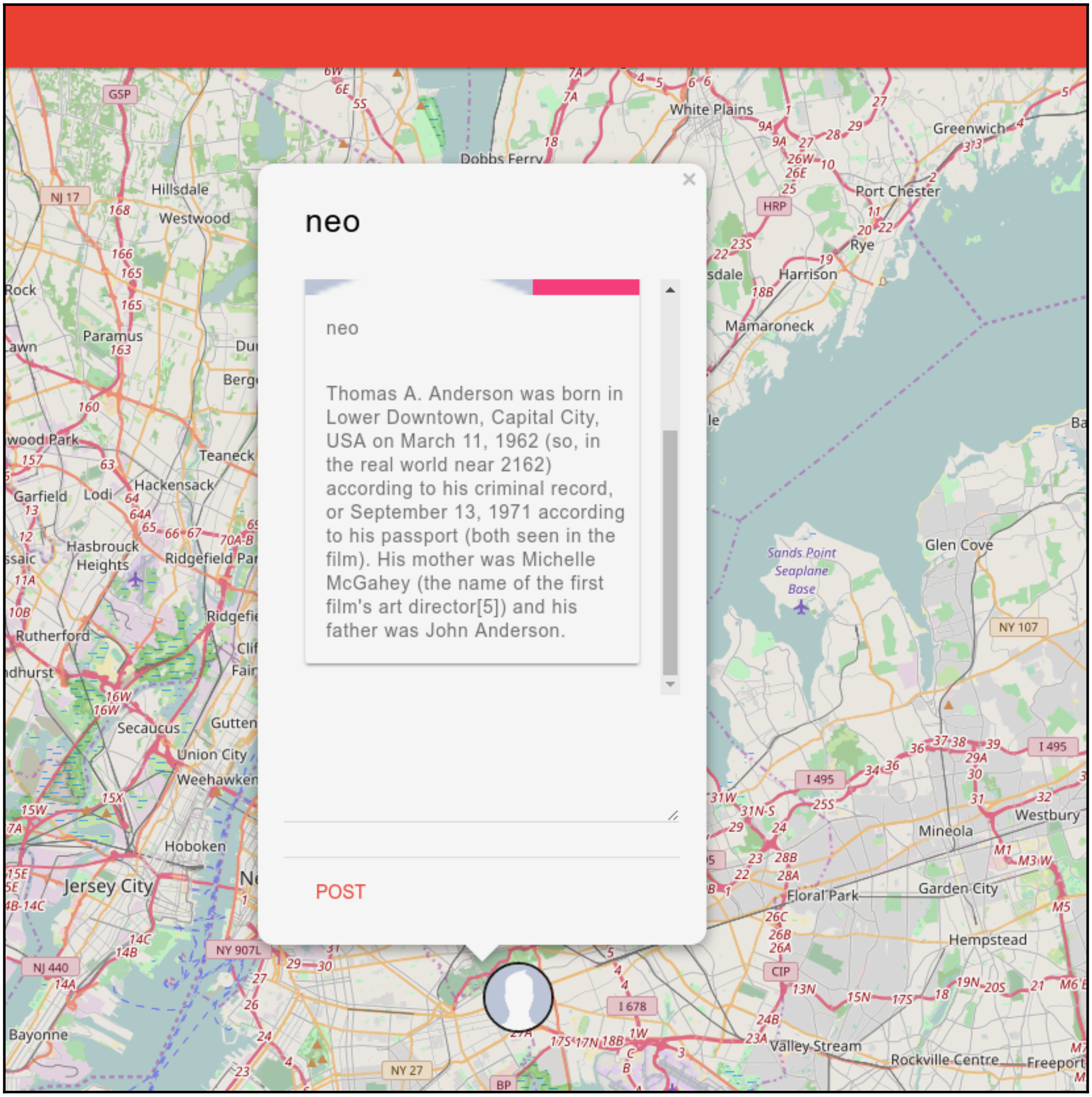




```
◢ components
  ◢ Main
      index.js
      Main.js
  ◢ Map
      index.js
      Map.js
  ◢ PageNotFound
      index.js
      PageNotFound.js
  ◢ PostForm
      index.js
      PostForm.js
  ◢ PostListItem
      index.js
      PostListItem.js
  ◢ PostsList
      index.js
      PostsList.js
```



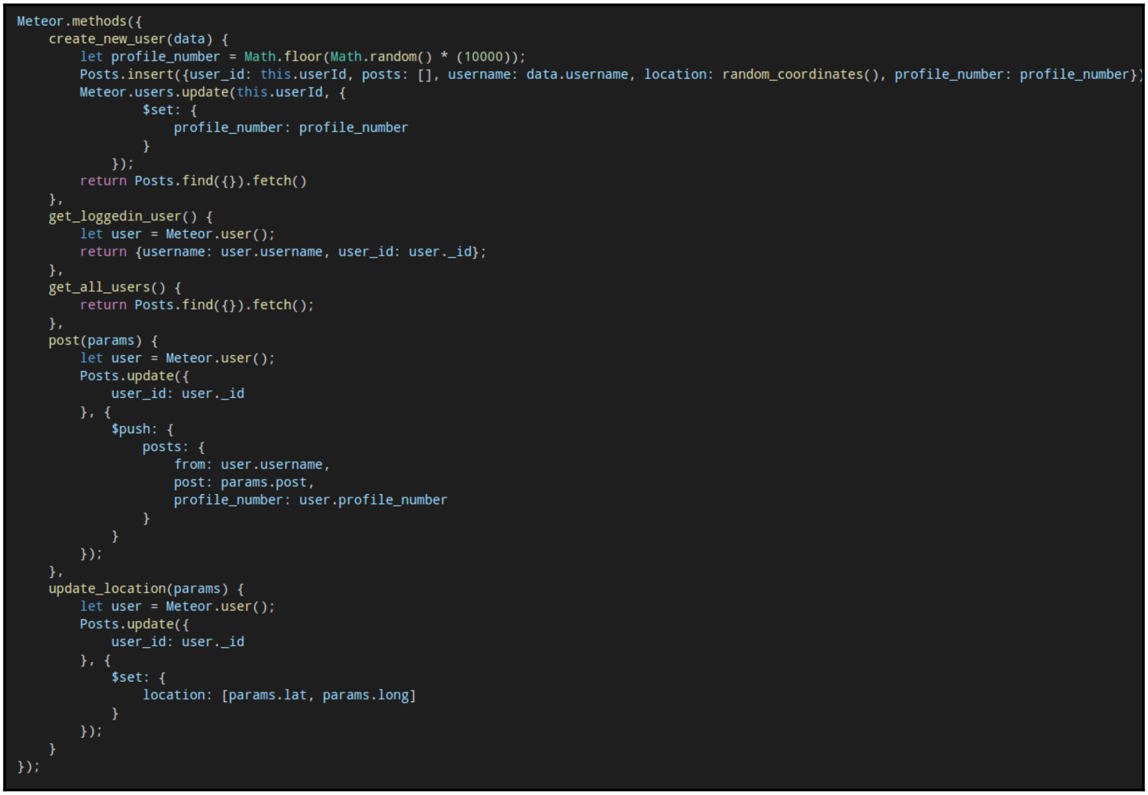
```
class MapComponent extends React.Component {
    constructor(props) {
        super(props)
        this.state = {zoom: 10, position: [40.7484, -73.9857]}
    }
    componentDidMount() {
        let self = this;
        const map = this.map.leafletElement;
        map.locate({setView: true, watch: true})
            .on('locationfound', function (e) {
                self.setState({
                    position: [e.latitude, e.longitude]
                })
                self.props.actions
                    .update_location({lat: e.latitude, long: e.longitude})
            })
            .on('locationerror', function (e) {
                console.log(e);
            });
    }
    render() {
        const {posts, actions} = this.props;
        let list = [];
        for (let i = 0; i < posts.length; i++) {
            let profile_picture = '<img class="img-circle" src="http://graph.facebook.com/v2.5/' + posts[i].profile_number + '/picture?height=200&height=200"></img>'
            let divIcon = L.divIcon({className: 'custom-div-icon', iconSize: null, html: profile_picture});
            list.push(<Marker position={posts[i].location} icon={divIcon} key={posts[i]._id}>
                    <Popup>
                        <PostForm actions={actions} user_posts={posts[i]}/>
                    </Popup>
                </Marker>
            )
        }
        return (<Map
                center={this.state.position}
                zoom={this.state.zoom}
                ref={map => {
                this.map = map;
            }}>
                <TileLayer
                    attribution='&copy; <a href="http://osm.org/copyright">OpenStreetMap</a> contributors'
                    url='http://{s}.tile.osm.org/{z}/{x}/{y}.png'></TileLayer>
                <LayerGroup>
                    {list}
                </LayerGroup>
            </Map>
        );
    }
}
export default MapComponent;
```



neo

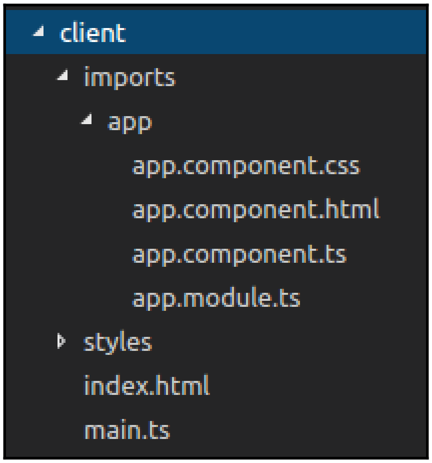
neo

Thomas A. Anderson was born in Lower Downtown, Capital City, USA on March 11, 1962 (so, in the real world near 2162) according to his criminal record, or September 13, 1971 according to his passport (both seen in the film). His mother was Michelle McGahey (the name of the first film's art director[5]) and his father was John Anderson.

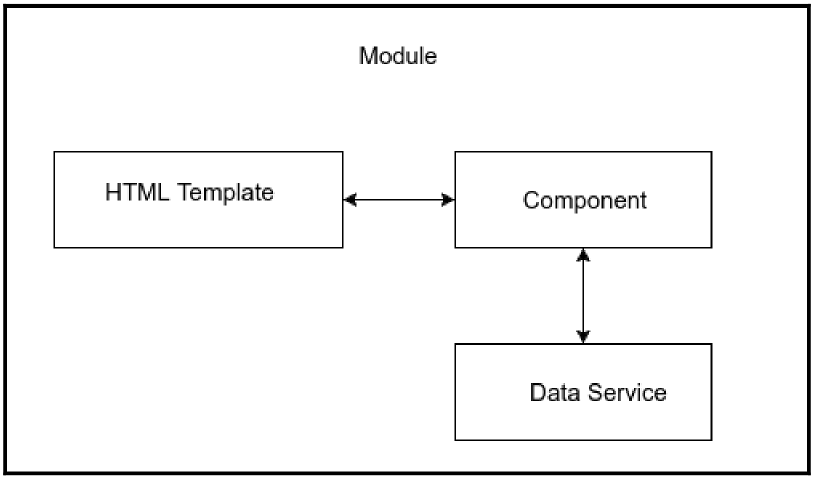


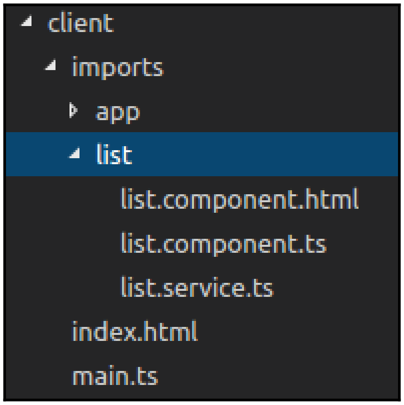
```
Meteor.methods({
    create_new_user(data) {
        let profile_number = Math.floor(Math.random() * (10000));
        Posts.insert({user_id: this.userId, posts: [], username: data.username, location: random_coordinates(), profile_number: profile_number})
        Meteor.users.update(this.userId, {
                $set: {
                    profile_number: profile_number
                }
            });
        return Posts.find({}).fetch()
    },
    get_loggedin_user() {
        let user = Meteor.user();
        return {username: user.username, user_id: user._id};
    },
    get_all_users() {
        return Posts.find({}).fetch();
    },
    post(params) {
        let user = Meteor.user();
        Posts.update({
            user_id: user._id
        }, {
            $push: {
                posts: {
                    from: user.username,
                    post: params.post,
                    profile_number: user.profile_number
                }
            }
        });
    },
    update_location(params) {
        let user = Meteor.user();
        Posts.update({
            user_id: user._id
        }, {
            $set: {
                location: [params.lat, params.long]
            }
        });
    }
});
```

Chapter 8:
Build a Chatbot with Facebook's Messenger Platform

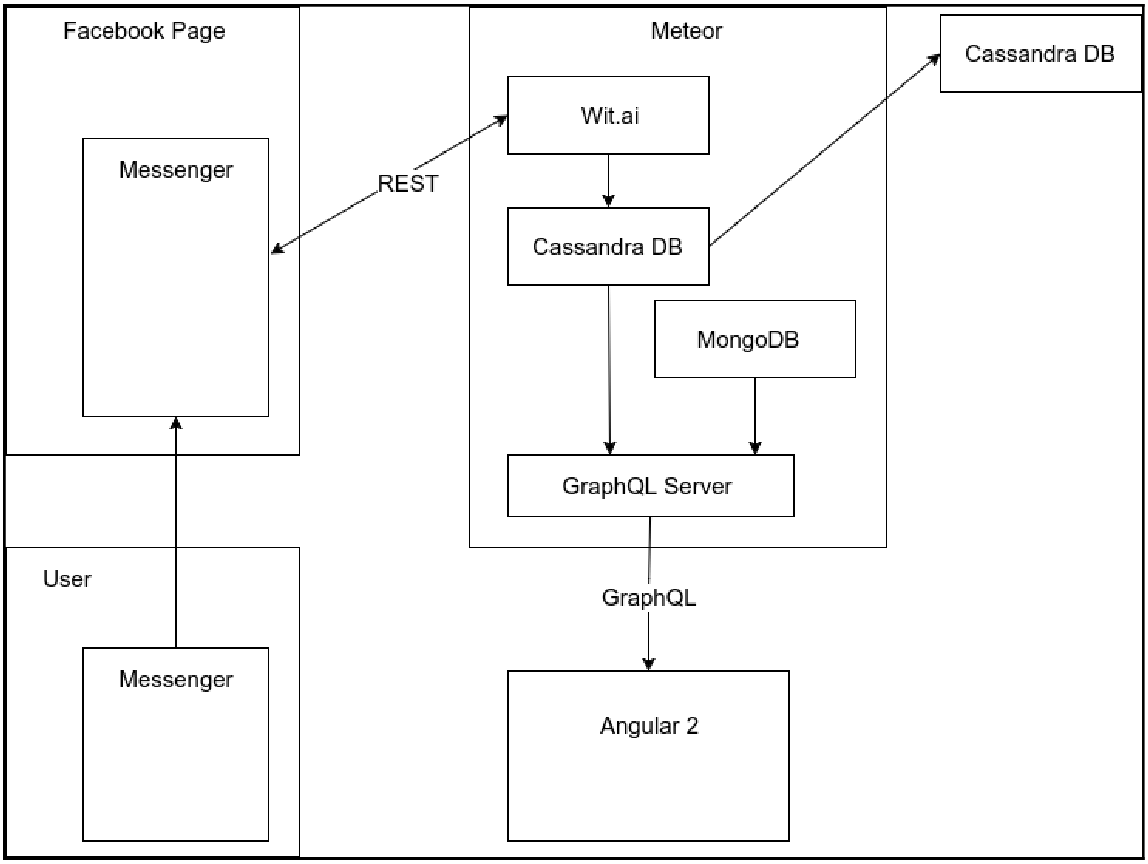


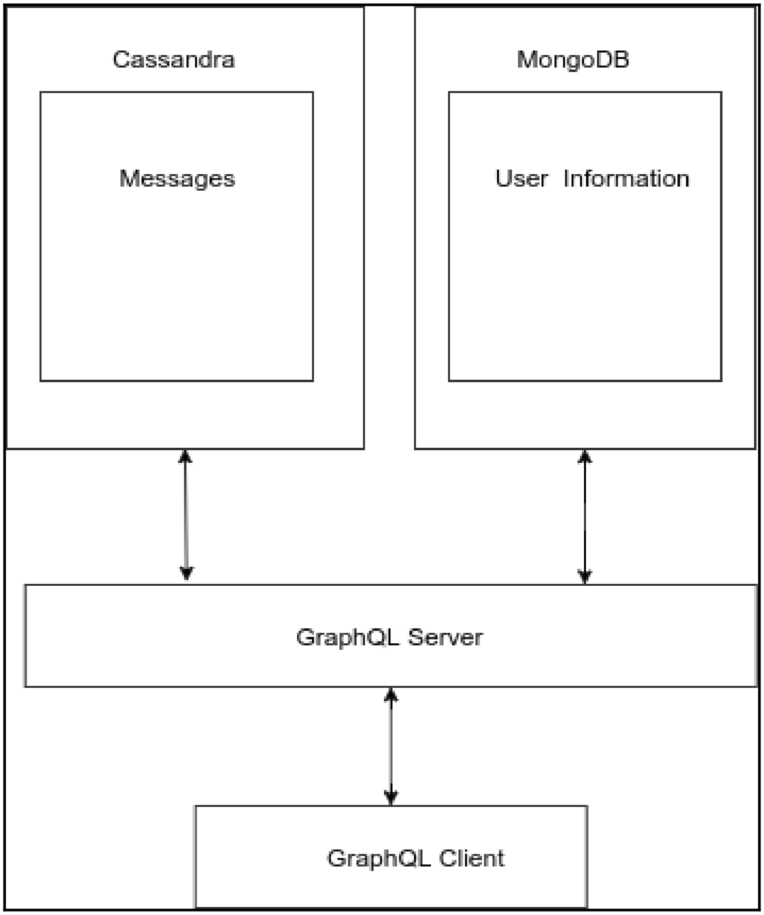
```
client
  imports
    app
      app.component.css
      app.component.html
      app.component.ts
      app.module.ts
  styles
  index.html
  main.ts
```

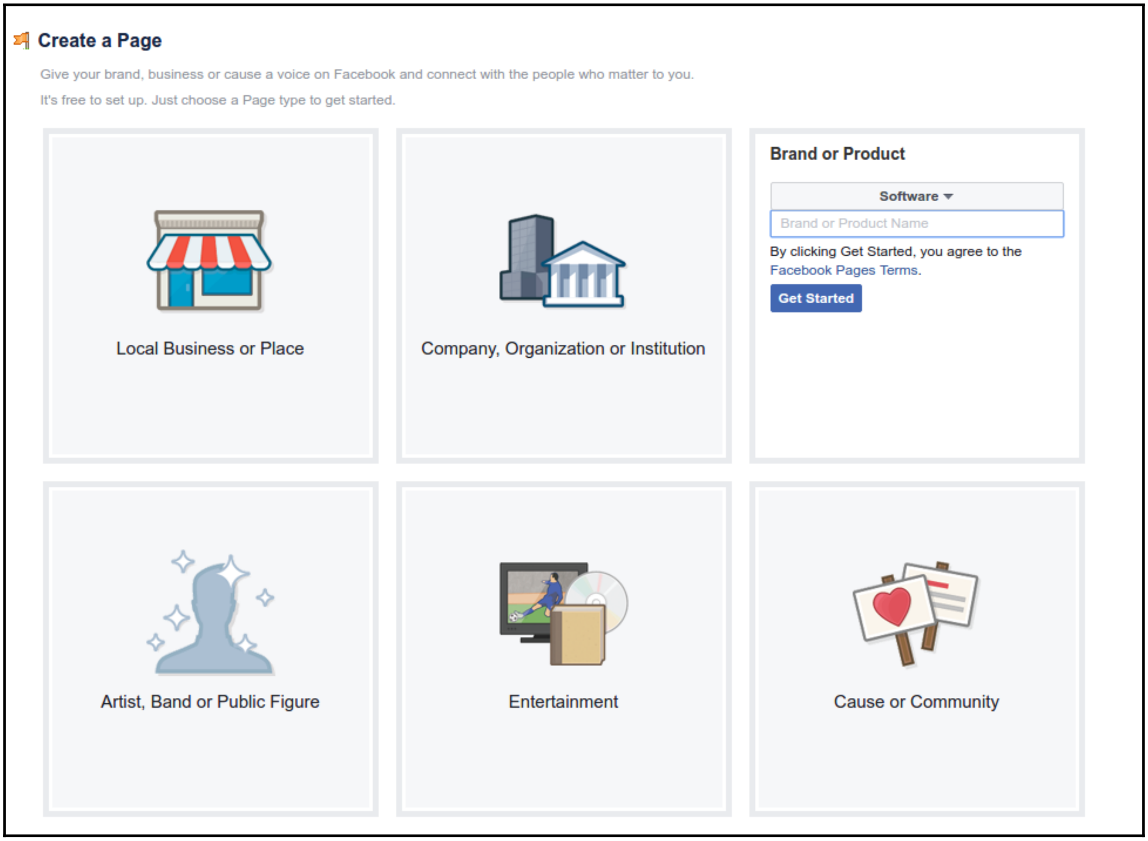




```
◢ client
  ◢ imports
    ▷ app
    ◢ list
        list.component.html
        list.component.ts
        list.service.ts
  index.html
  main.ts
```




Cassandra
Messages
MongoDB
User Information
GraphQL Server
GraphQL Client



Create a Page

Give your brand, business or cause a voice on Facebook and connect with the people who matter to you.

It's free to set up. Just choose a Page type to get started.

Local Business or Place

Company, Organization or Institution

Brand or Product

Software ▾

Brand or Product Name

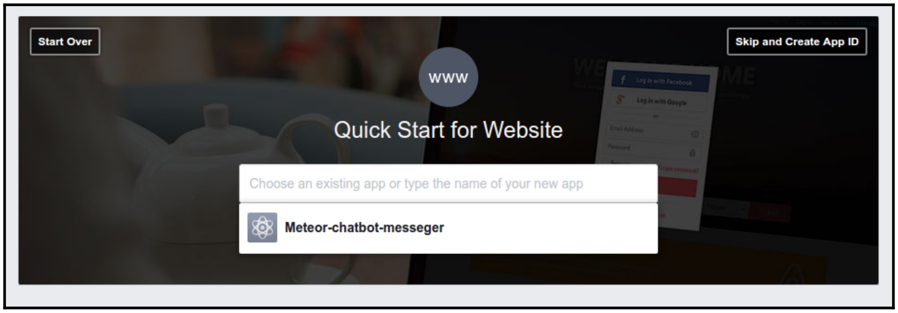
By clicking Get Started, you agree to the Facebook Pages Terms.

Get Started

Artist, Band or Public Figure

Entertainment

Cause or Community



Start Over

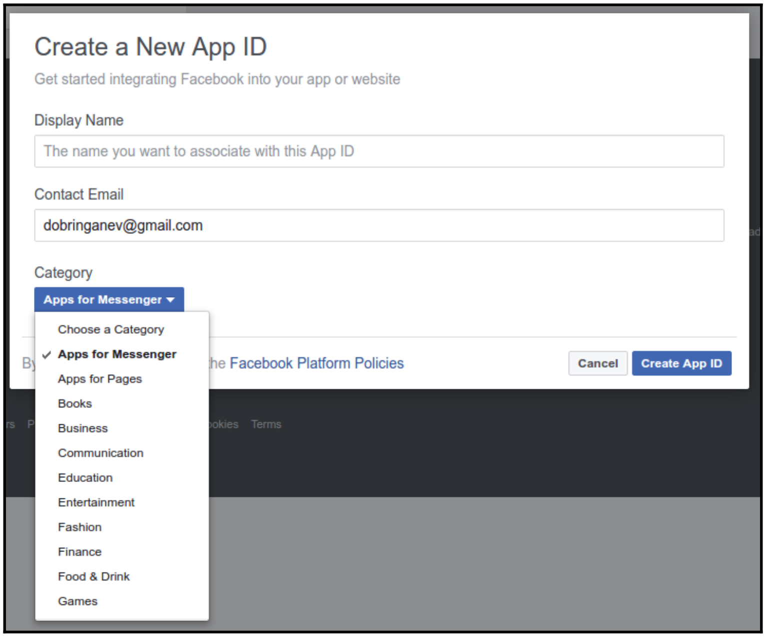
Skip and Create App ID

www

Quick Start for Website

Choose an existing app or type the name of your new app

Meteor-chatbot-messeger



Create a New App ID

Get started integrating Facebook into your app or website

Display Name

The name you want to associate with this App ID

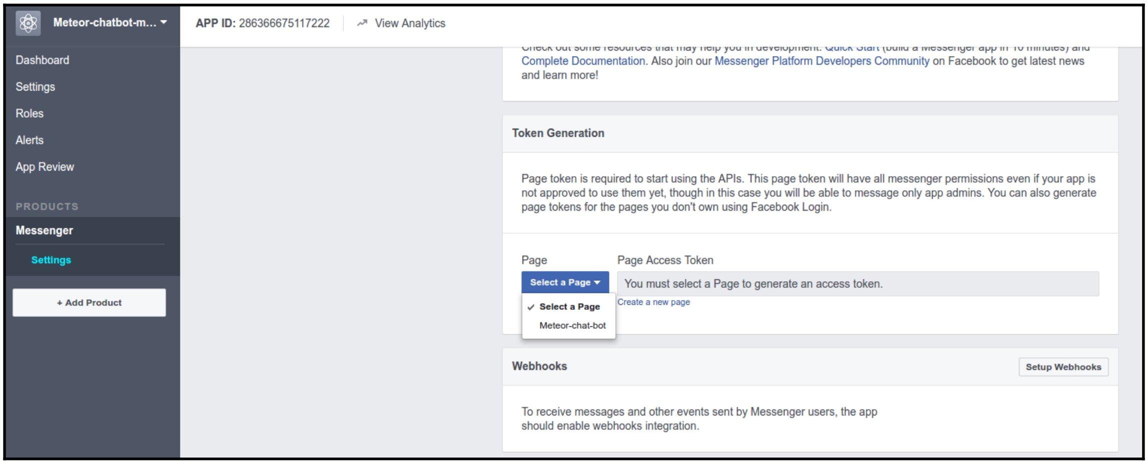
Contact Email

dobringanev@gmail.com

Category

Apps for Messenger

By ... the Facebook Platform Policies



APP ID: 286366675117222

Check out some resources that may help you in development. Quick Start (build a Messenger app in 10 minutes) and Complete Documentation. Also join our Messenger Platform Developers Community on Facebook to get latest news and learn more!

Token Generation

Page token is required to start using the APIs. This page token will have all messenger permissions even if your app is not approved to use them yet, though in this case you will be able to message only app admins. You can also generate page tokens for the pages you don't own using Facebook Login.

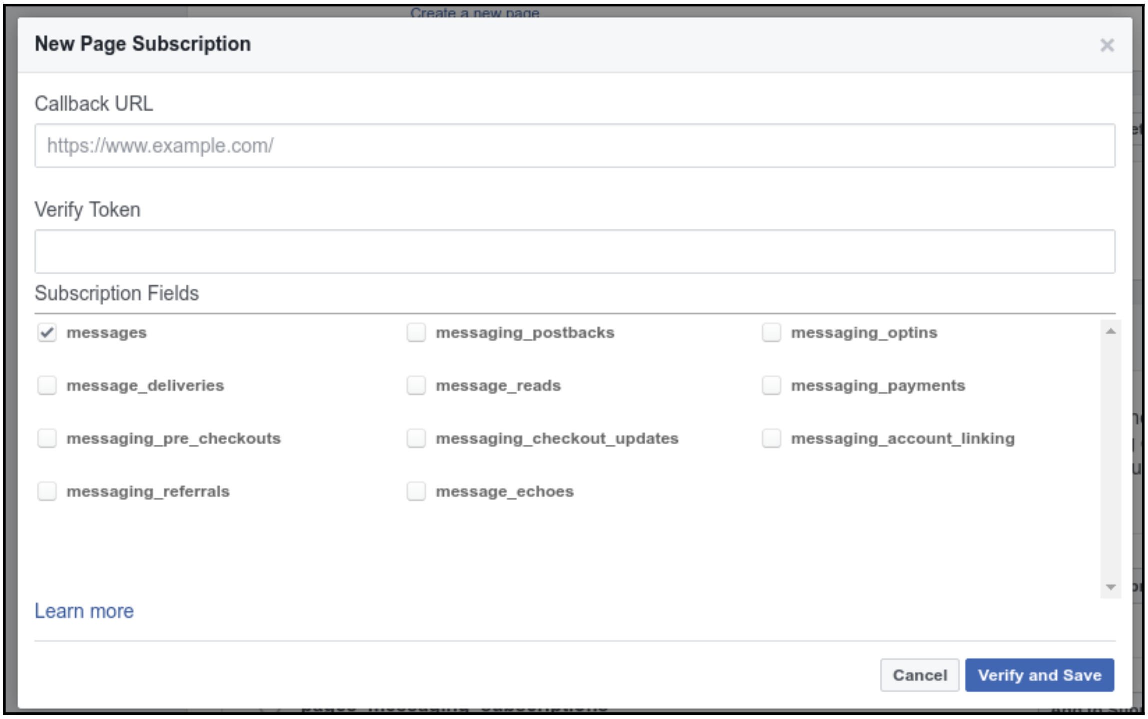
Page

Page Access Token

You must select a Page to generate an access token.

Webhooks

To receive messages and other events sent by Messenger users, the app should enable webhooks integration.



New Page Subscription

Callback URL

https://www.example.com/

Verify Token

Subscription Fields

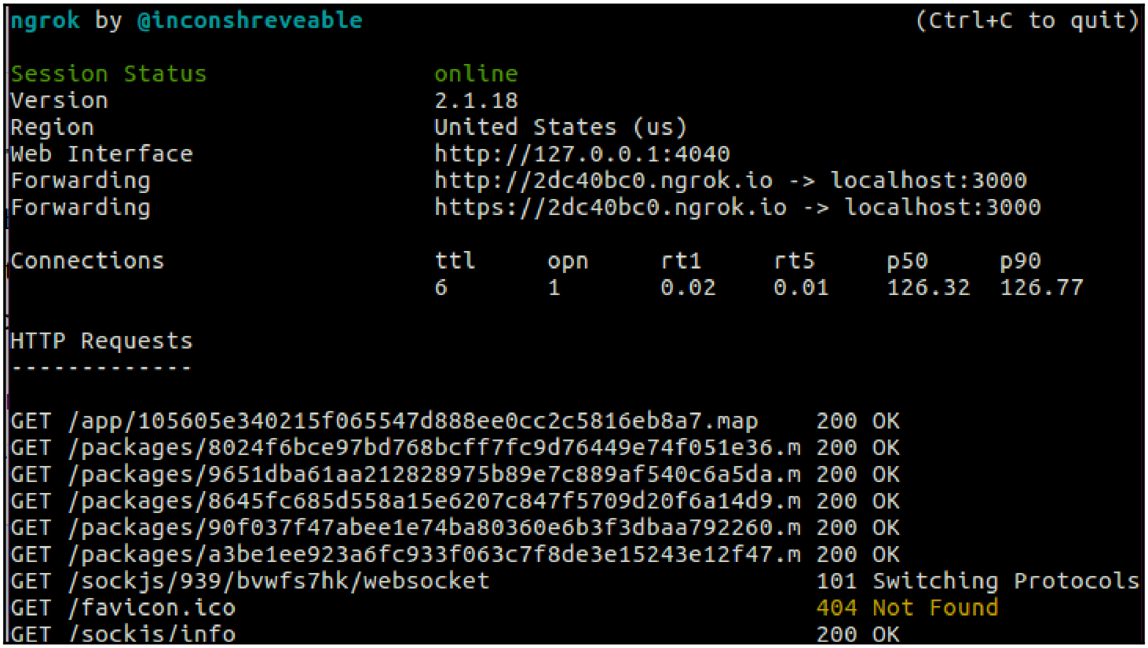
☑ messages ☐ messaging_postbacks ☐ messaging_optins

☐ message_deliveries ☐ message_reads ☐ messaging_payments

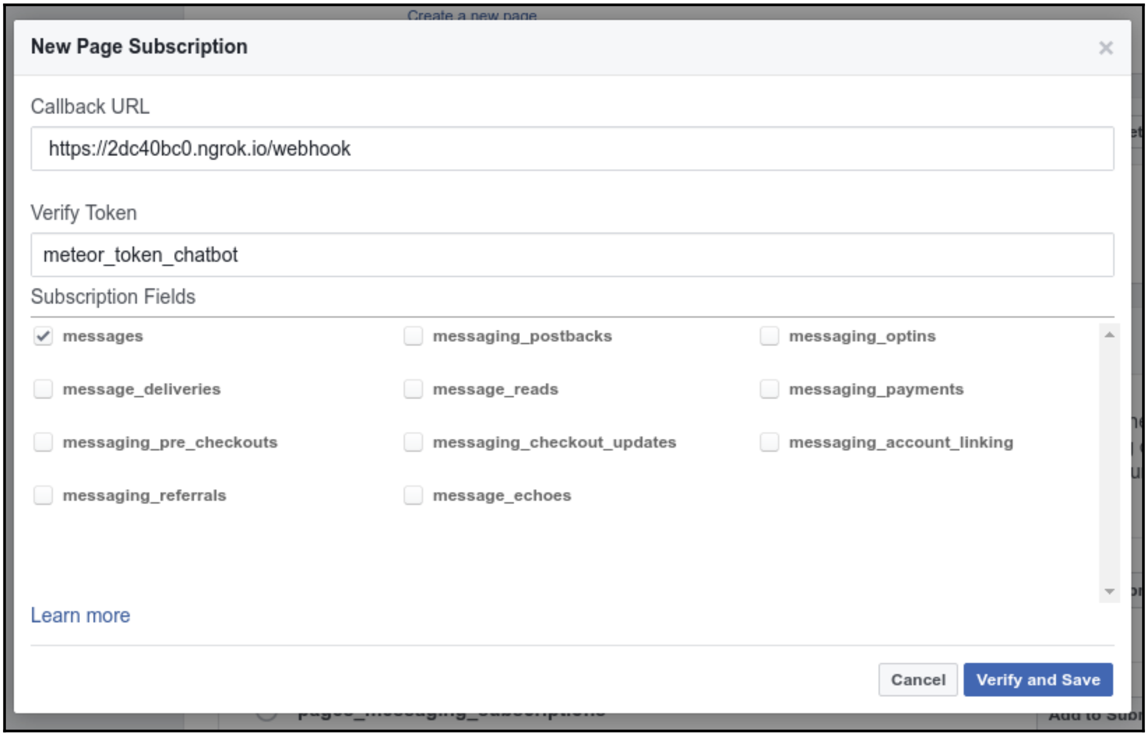
☐ messaging_pre_checkouts ☐ messaging_checkout_updates ☐ messaging_account_linking

☐ messaging_referrals ☐ message_echoes

Learn more



```
ngrok by @inconshreveable                                   (Ctrl+C to quit)

Session Status                online
Version                       2.1.18
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://2dc40bc0.ngrok.io -> localhost:3000
Forwarding                    https://2dc40bc0.ngrok.io -> localhost:3000

Connections                   ttl     opn     rt1     rt5     p50     p90
                              6       1       0.02    0.01    126.32  126.77

HTTP Requests
-------------

GET /app/105605e340215f065547d888ee0cc2c5816eb8a7.map    200 OK
GET /packages/8024f6bce97bd768bcff7fc9d76449e74f051e36.m 200 OK
GET /packages/9651dba61aa212828975b89e7c889af540c6a5da.m 200 OK
GET /packages/8645fc685d558a15e6207c847f5709d20f6a14d9.m 200 OK
GET /packages/90f037f47abee1e74ba80360e6b3f3dbaa792260.m 200 OK
GET /packages/a3be1ee923a6fc933f063c7f8de3e15243e12f47.m 200 OK
GET /sockjs/939/bvwfs7hk/websocket                       101 Switching Protocols
GET /favicon.ico                                         404 Not Found
GET /sockjs/info                                         200 OK
```



New Page Subscription

Callback URL

https://2dc40bc0.ngrok.io/webhook

Verify Token

meteor_token_chatbot

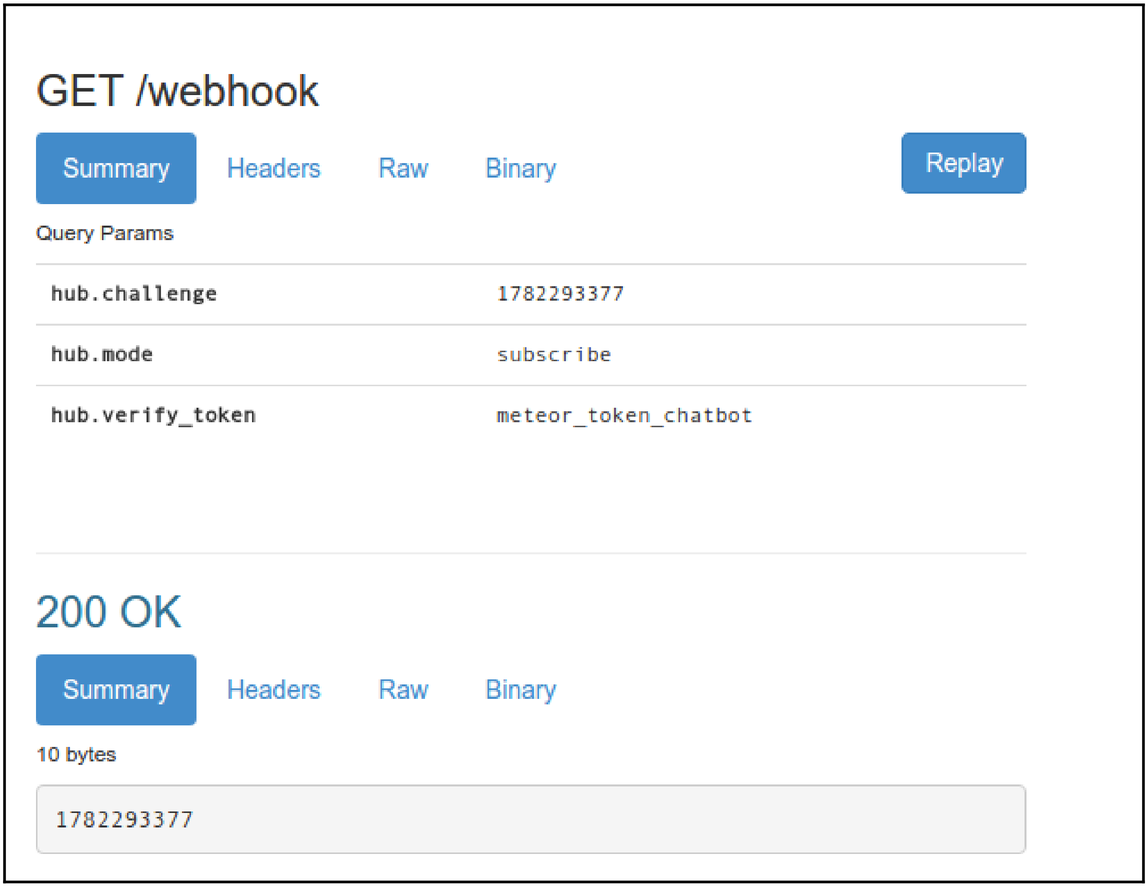
Subscription Fields

☑ messages ☐ messaging_postbacks ☐ messaging_optins

☐ message_deliveries ☐ message_reads ☐ messaging_payments

☐ messaging_pre_checkouts ☐ messaging_checkout_updates ☐ messaging_account_linking

☐ messaging_referrals ☐ message_echoes



GET /webhook

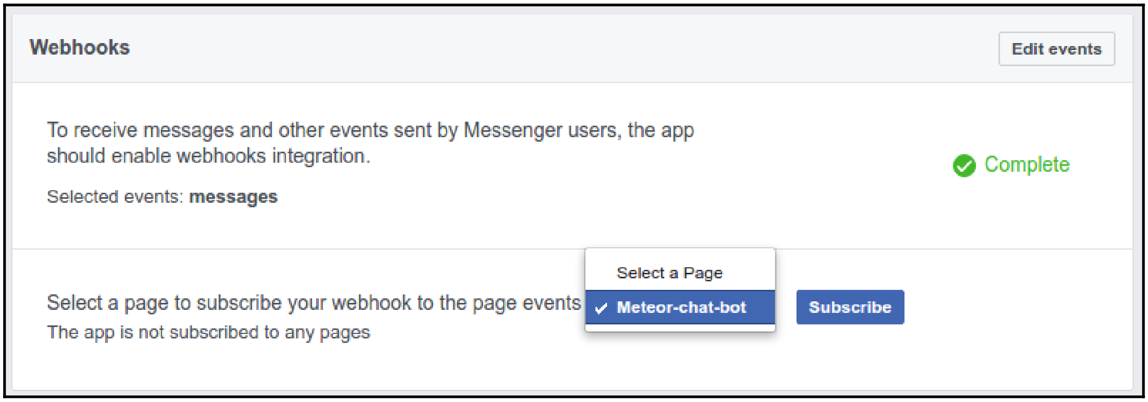
Query Params

hub.challenge	1782293377
hub.mode	subscribe
hub.verify_token	meteor_token_chatbot

200 OK

10 bytes

```
1782293377
```



Webhooks

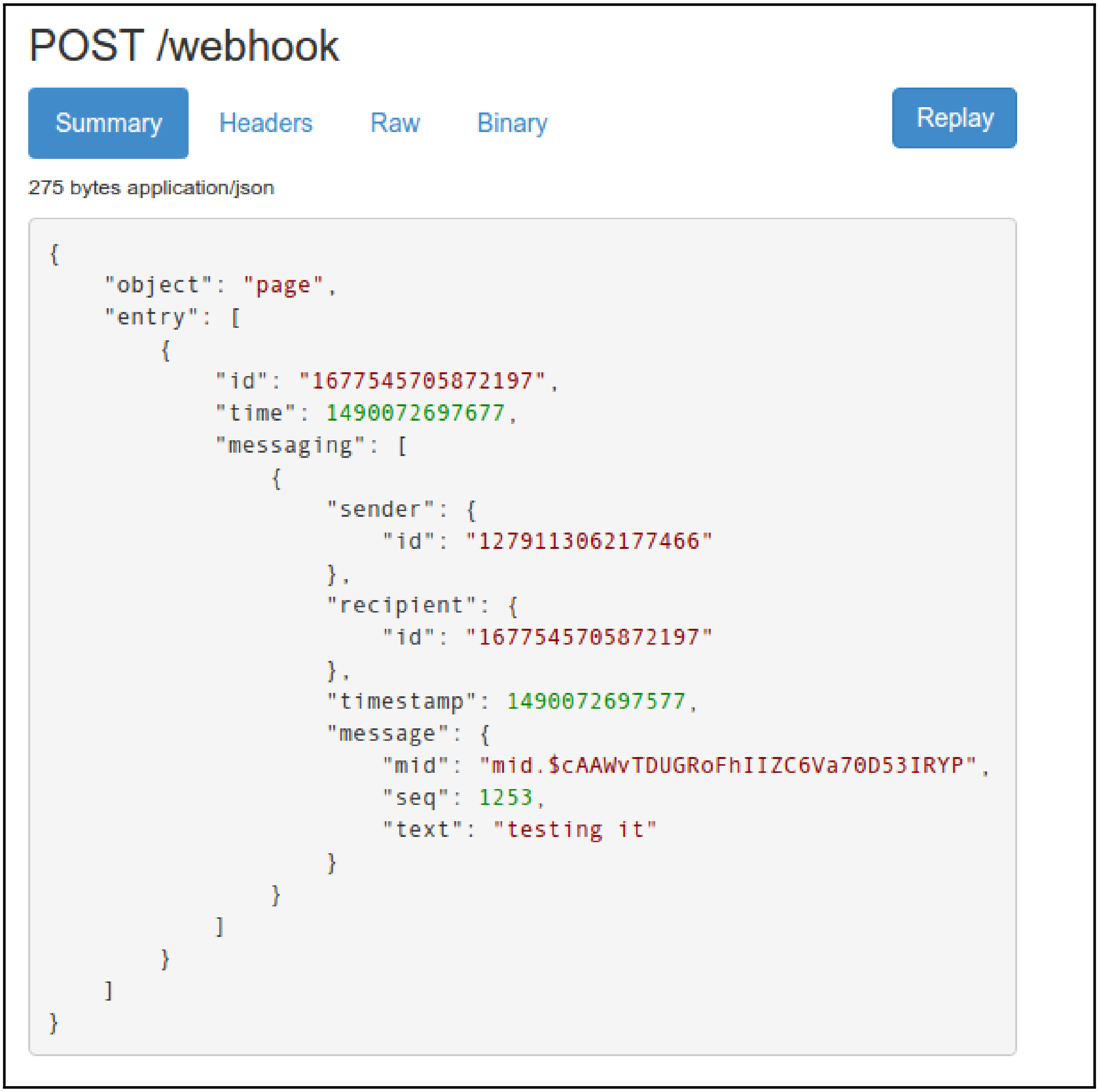
To receive messages and other events sent by Messenger users, the app should enable webhooks integration.

Selected events: **messages**

✓ Complete

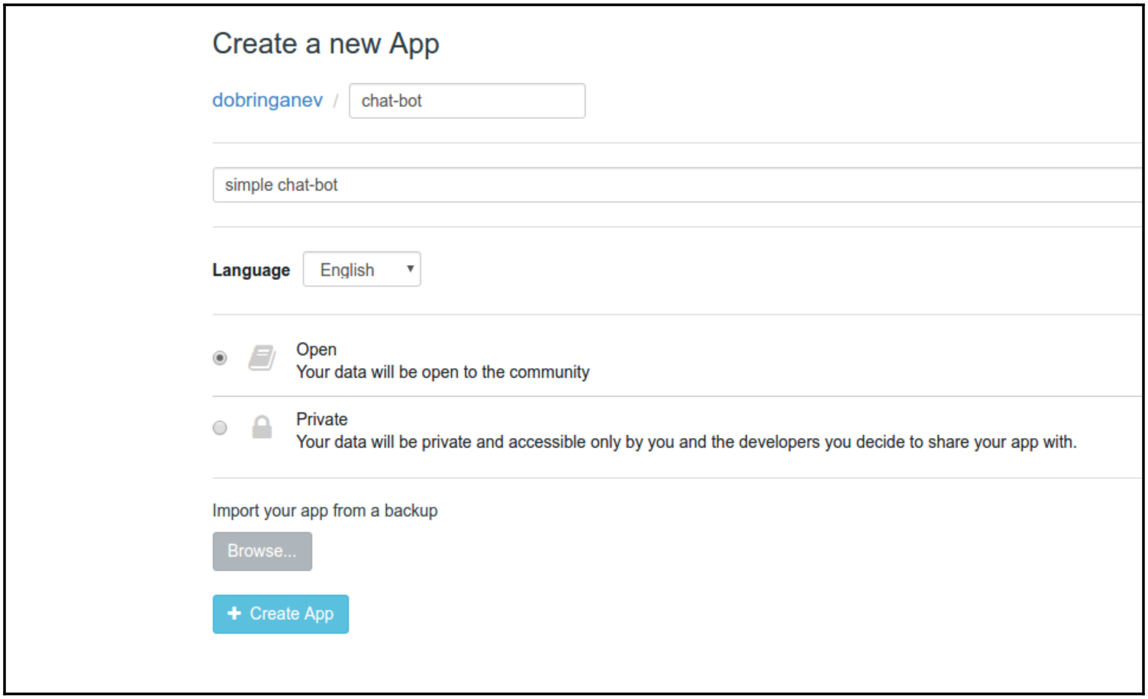
Select a page to subscribe your webhook to the page events

The app is not subscribed to any pages



275 bytes application/json

```
{
    "object": "page",
    "entry": [
        {
            "id": "1677545705872197",
            "time": 1490072697677,
            "messaging": [
                {
                    "sender": {
                        "id": "1279113062177466"
                    },
                    "recipient": {
                        "id": "1677545705872197"
                    },
                    "timestamp": 1490072697577,
                    "message": {
                        "mid": "mid.$cAAWvTDUGRoFhIIZC6Va70D53IRYP",
                        "seq": 1253,
                        "text": "testing it"
                    }
                }
            ]
        }
    ]
}
```



Create a new App

dobringanev / chat-bot

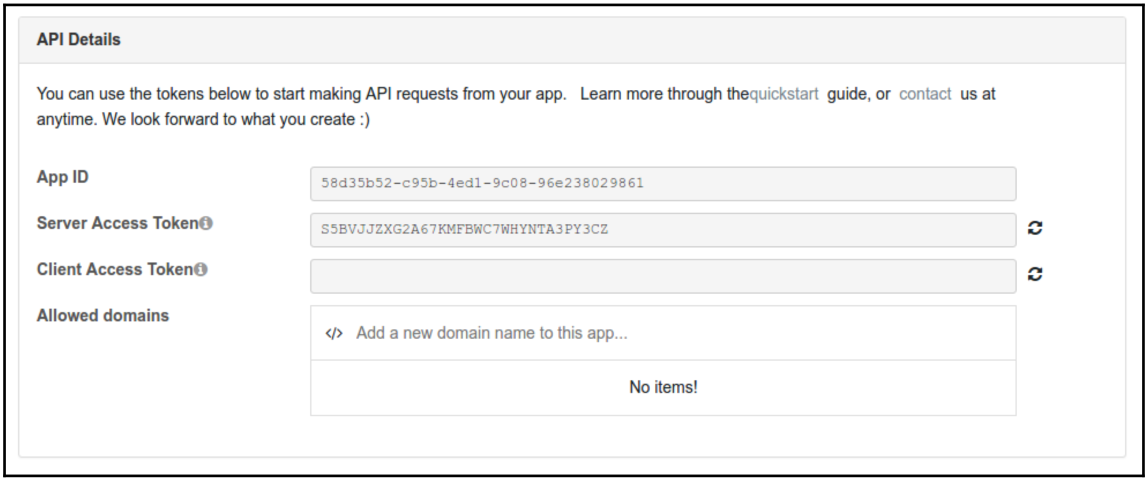
simple chat-bot

Language English ▾

● Open
Your data will be open to the community

○ Private
Your data will be private and accessible only by you and the developers you decide to share your app with.

Import your app from a backup



API Details

You can use the tokens below to start making API requests from your app.   Learn more through thequickstart guide, or contact us at anytime. We look forward to what you create :)

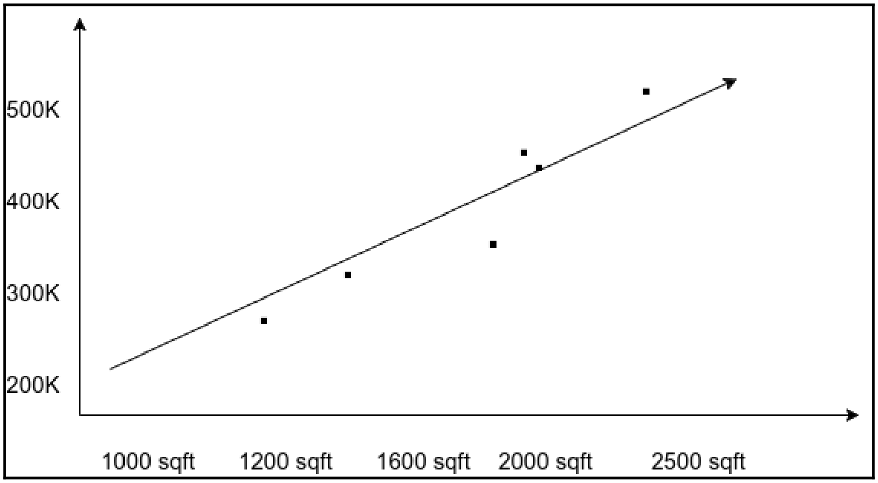
App ID 58d35b52-c95b-4ed1-9c08-96e238029861

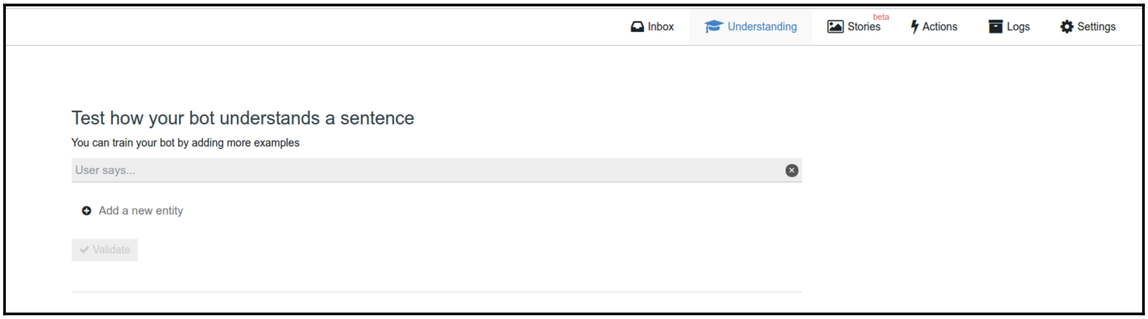
Server Access Token S5BVJJZXG2A67KMFBWC7WHYNTA3PY3CZ

Client Access Token

Allowed domains Add a new domain name to this app...

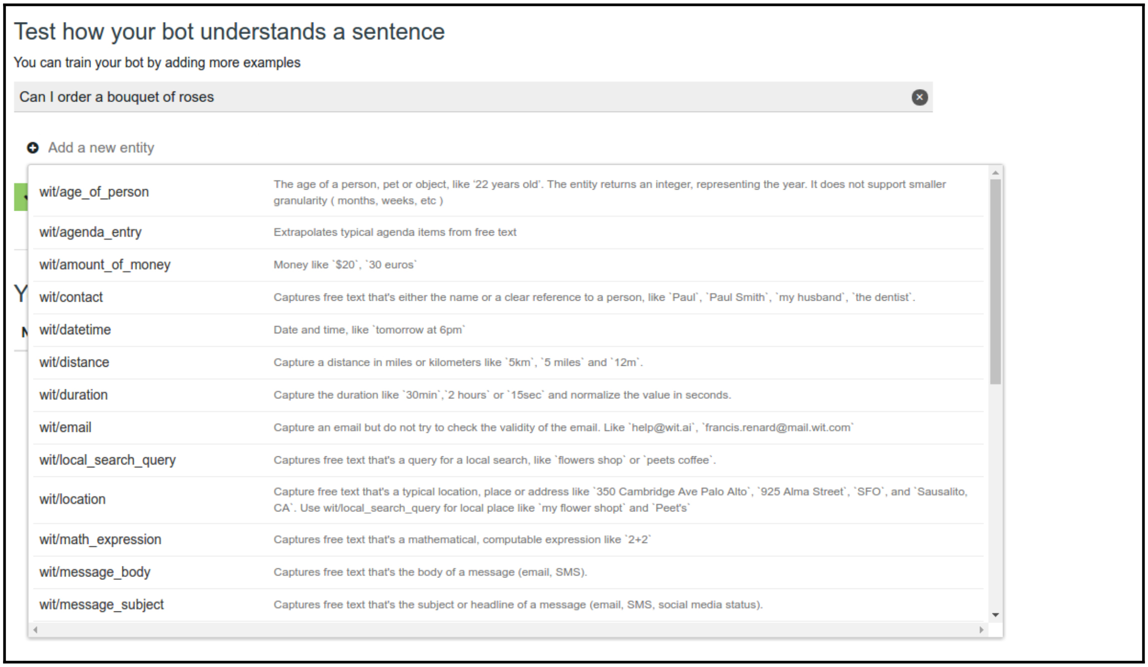
No items!





Test how your bot understands a sentence

You can train your bot by adding more examples



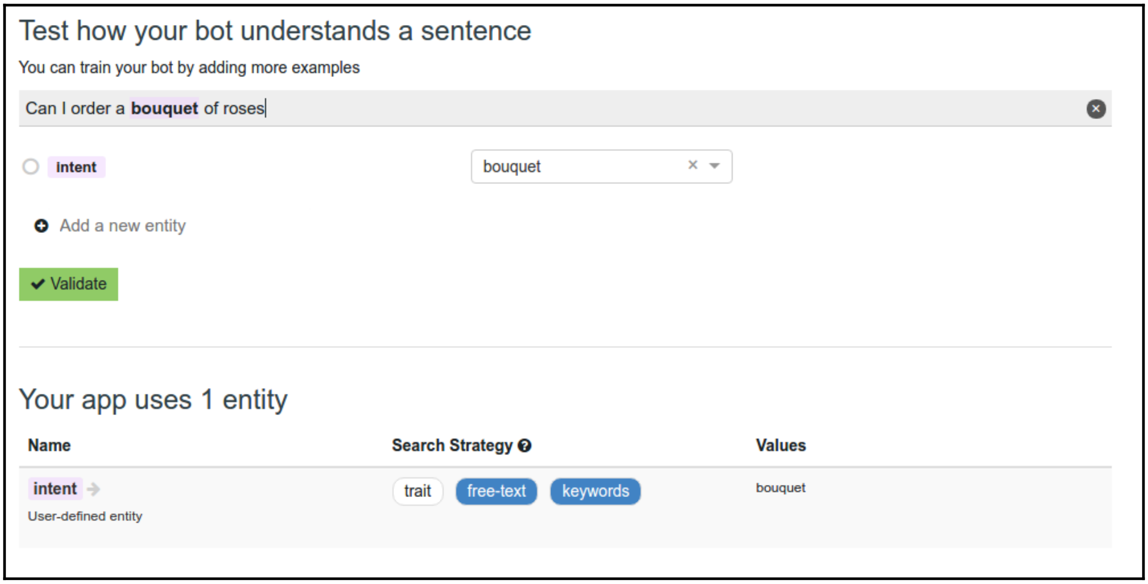
Test how your bot understands a sentence

You can train your bot by adding more examples

Can I order a bouquet of roses

Add a new entity

wit/age_of_person	The age of a person, pet or object, like ‘22 years old’. The entity returns an integer, representing the year. It does not support smaller granularity (months, weeks, etc)
wit/agenda_entry	Extrapolates typical agenda items from free text
wit/amount_of_money	Money like `$20`, `30 euros`
wit/contact	Captures free text that's either the name or a clear reference to a person, like `Paul`, `Paul Smith`, `my husband`, `the dentist`.
wit/datetime	Date and time, like `tomorrow at 6pm`
wit/distance	Capture a distance in miles or kilometers like `5km`, `5 miles` and `12m`.
wit/duration	Capture the duration like `30min`,`2 hours` or `15sec` and normalize the value in seconds.
wit/email	Capture an email but do not try to check the validity of the email. Like `help@wit.ai`, `francis.renard@mail.wit.com`
wit/local_search_query	Captures free text that's a query for a local search, like `flowers shop` or `peets coffee`.
wit/location	Capture free text that's a typical location, place or address like `350 Cambridge Ave Palo Alto`, `925 Alma Street`, `SFO`, and `Sausalito, CA`. Use wit/local_search_query for local place like `my flower shopt` and `Peet's`
wit/math_expression	Captures free text that's a mathematical, computable expression like `2+2`
wit/message_body	Captures free text that's the body of a message (email, SMS).
wit/message_subject	Captures free text that's the subject or headline of a message (email, SMS, social media status).



Test how your bot understands a sentence

You can train your bot by adding more examples

Can I order a **bouquet** of roses

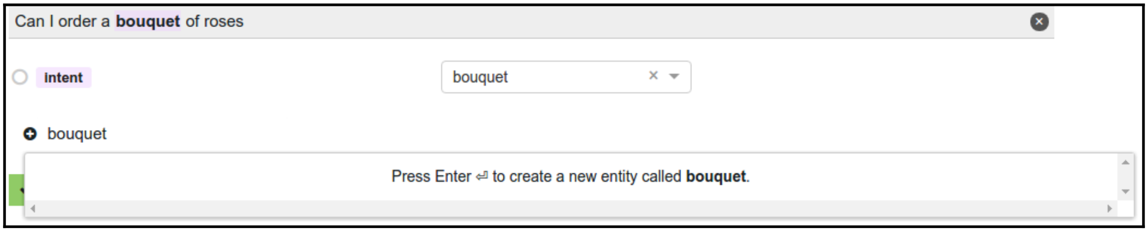
○ **intent** — bouquet

Add a new entity

✓ Validate

Your app uses 1 entity

Name	Search Strategy	Values
intent User-defined entity	trait free-text keywords	bouquet



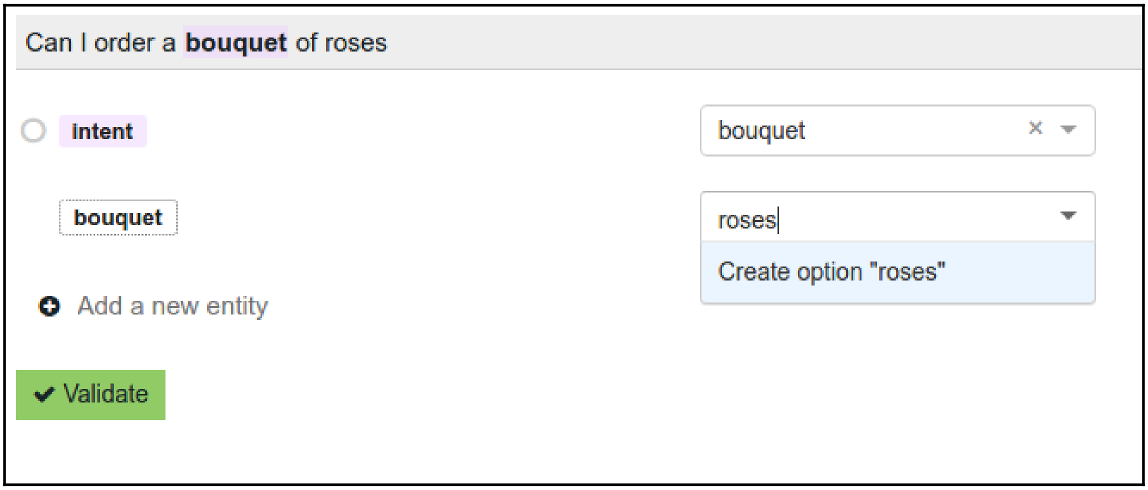
Can I order a **bouquet** of roses

○ intent

bouquet × ▾

⊕ bouquet

Press Enter ⏎ to create a new entity called **bouquet**.



Can I order a **bouquet** of roses

○ **intent**

bouquet × ▾

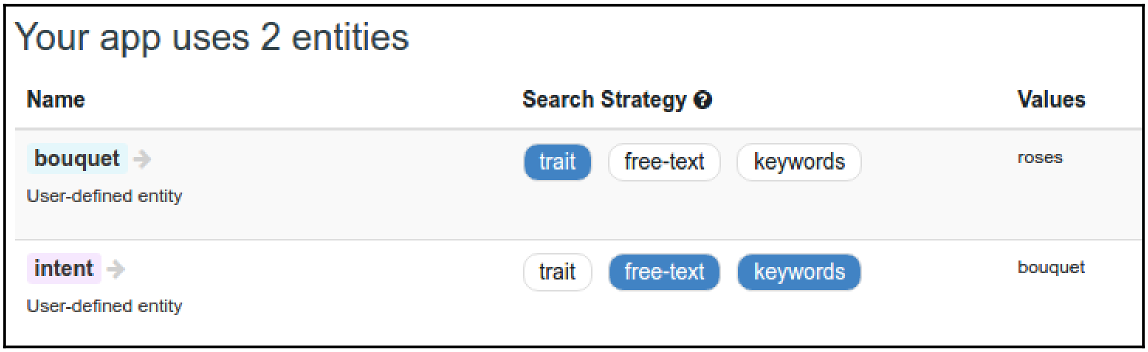
bouquet

roses ▾

Create option "roses"

⊕ Add a new entity

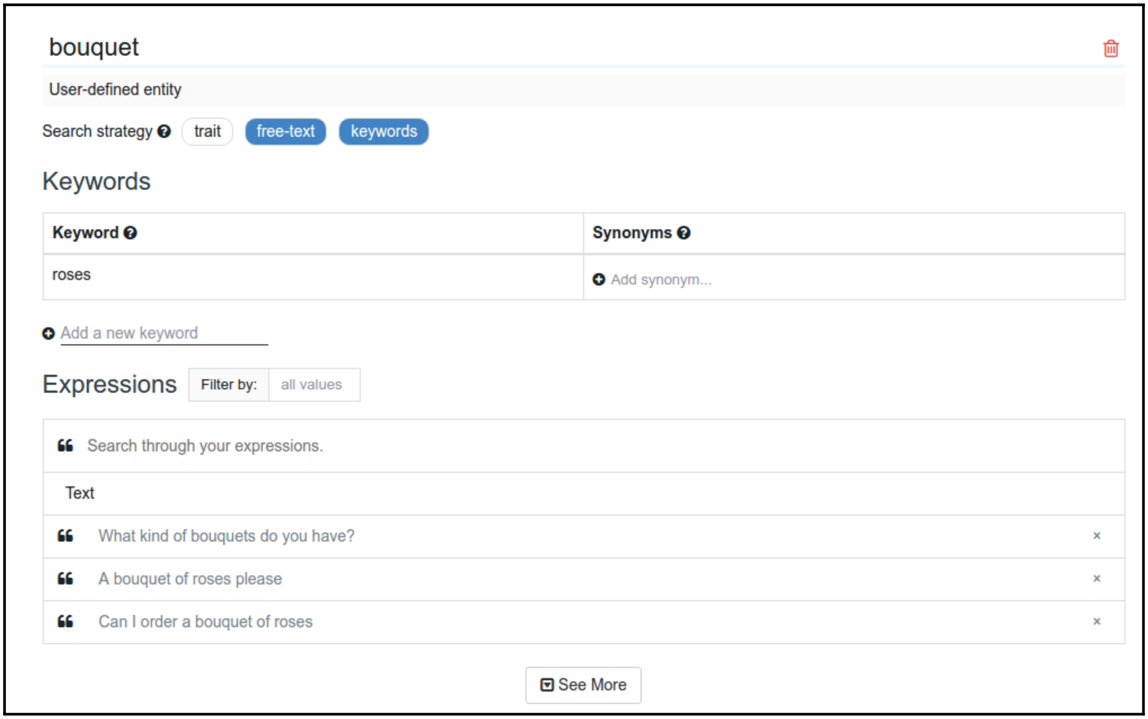
✓ Validate



Your app uses 2 entities

Name	Search Strategy ?	Values
bouquet → User-defined entity	trait free-text keywords	roses
intent → User-defined entity	trait free-text keywords	bouquet

bouquet →



bouquet

User-defined entity

Search strategy: trait, free-text, keywords

Keywords

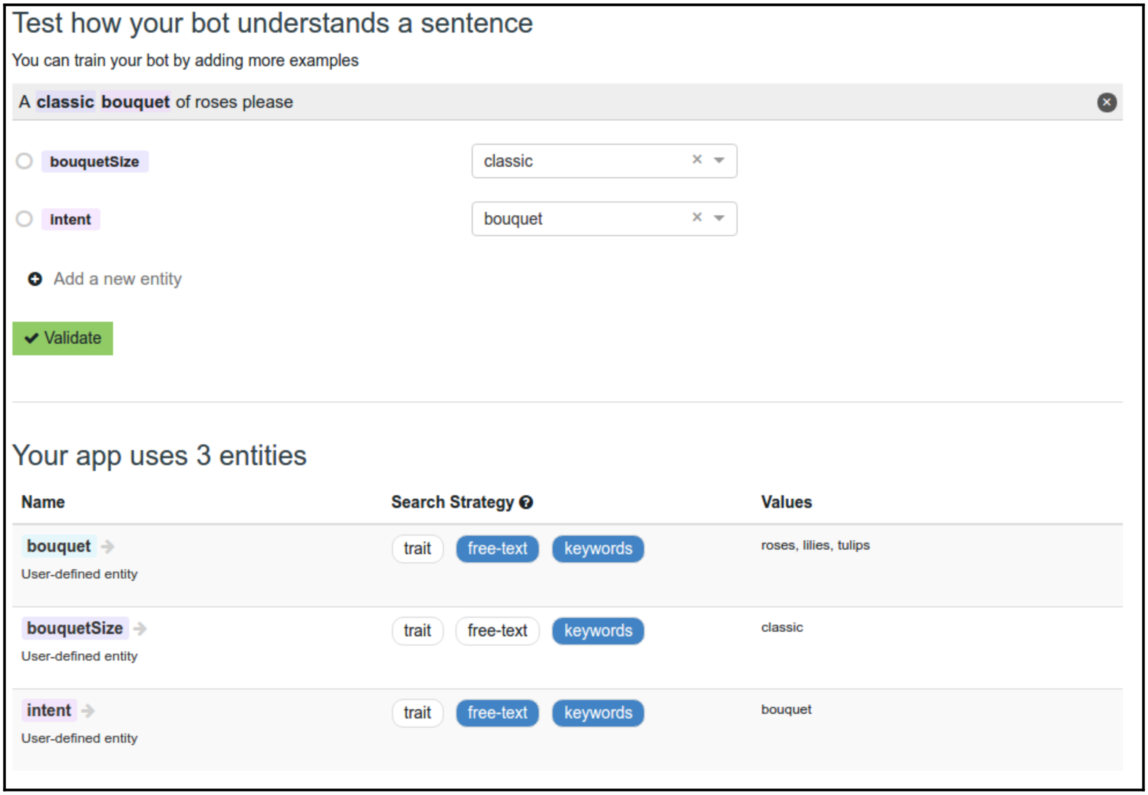
Keyword	Synonyms
roses	Add synonym...

Add a new keyword

Expressions — Filter by: all values

Search through your expressions.

Text
What kind of bouquets do you have?
A bouquet of roses please
Can I order a bouquet of roses



Test how your bot understands a sentence

You can train your bot by adding more examples

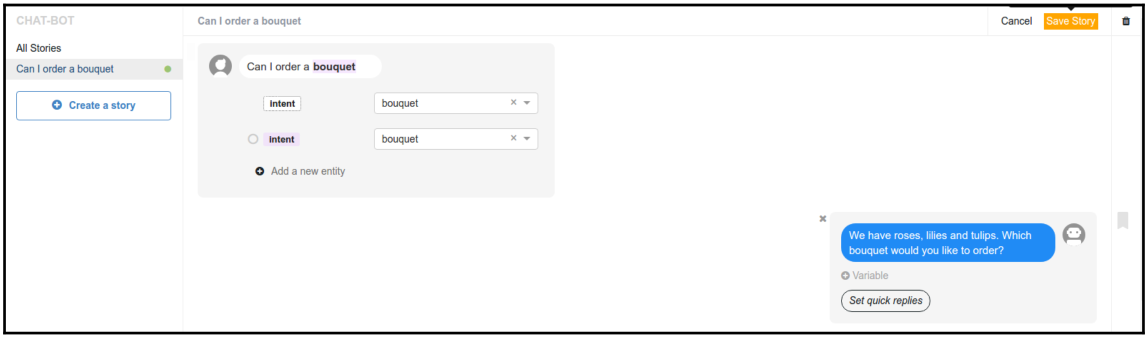
A **classic** **bouquet** of roses please

○ **bouquetSize** — classic

○ **intent** — bouquet

Your app uses 3 entities

Name	Search Strategy	Values
bouquet User-defined entity	trait, free-text, keywords	roses, lilies, tulips
bouquetSize User-defined entity	trait, free-text, keywords	classic
intent User-defined entity	trait, free-text, keywords	bouquet

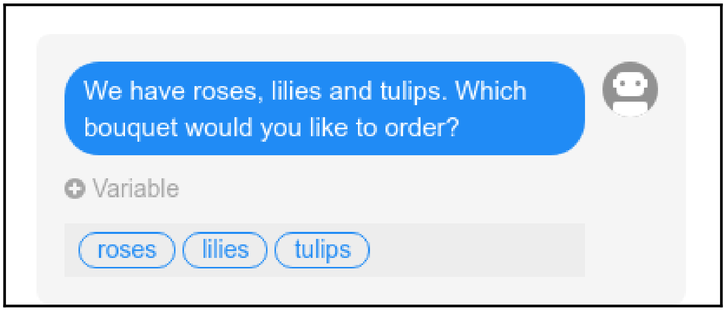


Can I order a **bouquet**

intent — bouquet

○ intent — bouquet

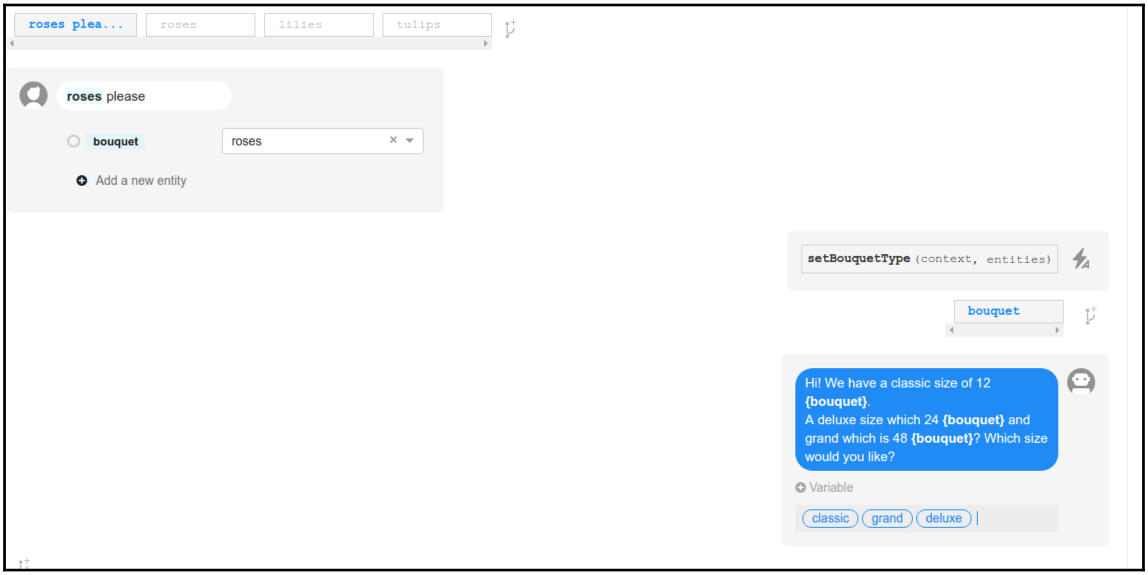
We have roses, lilies and tulips. Which bouquet would you like to order?



We have roses, lilies and tulips. Which bouquet would you like to order?

Variable

roses lilies tulips



roses plea... roses lilies tulips

roses please

bouquet roses

Add a new entity

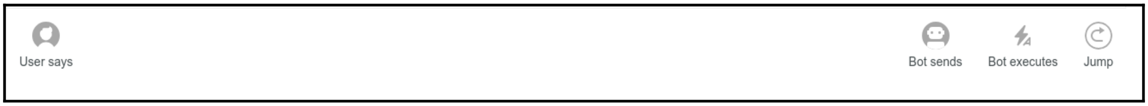
setBouquetType (context, entities)

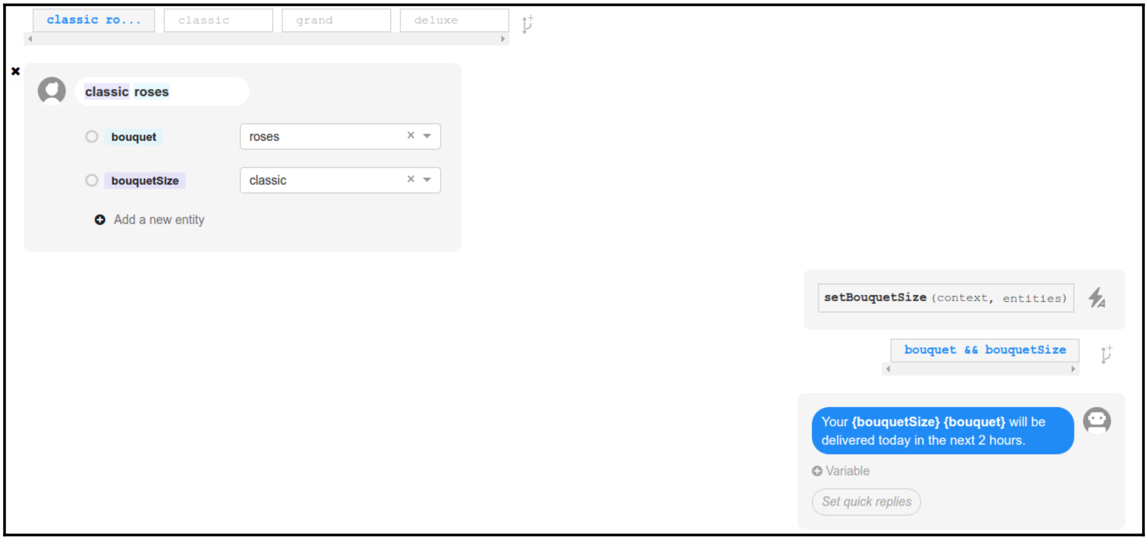
bouquet

Hi! We have a classic size of 12 **{bouquet}**.
A deluxe size which 24 **{bouquet}** and grand which is 48 **{bouquet}**? Which size would you like?

Variable

classic grand deluxe





classic roses

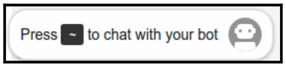
bouquet: roses

bouquetSize: classic

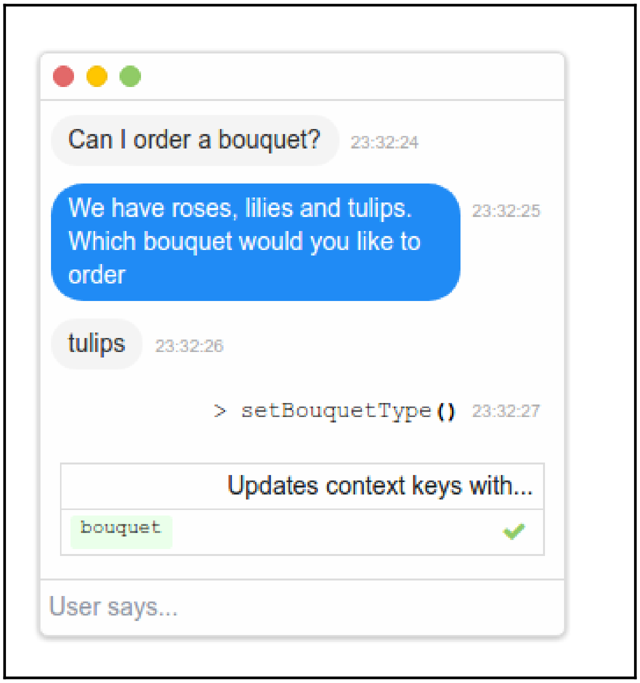
`setBouquetSize (context, entities)`

`bouquet && bouquetSize`

Your {bouquetSize} {bouquet} will be delivered today in the next 2 hours.



Press ~ to chat with your bot



Can I order a bouquet? 23:32:24

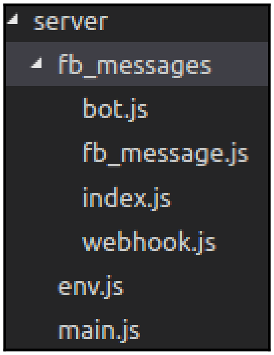
We have roses, lilies and tulips. Which bouquet would you like to order 23:32:25

tulips 23:32:26

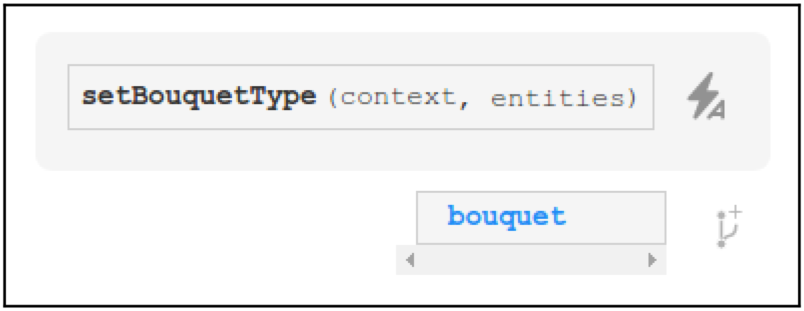
> setBouquetType() 23:32:27

Updates context keys with...

bouquet ✓

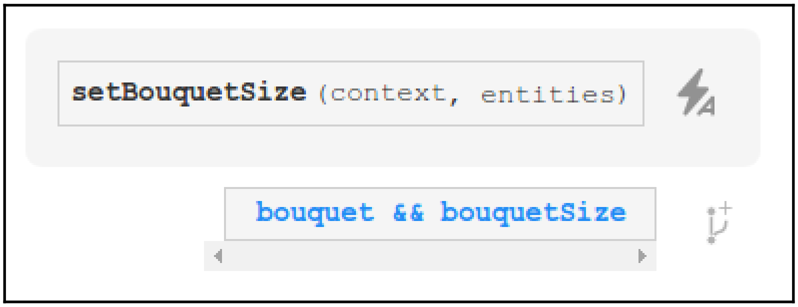


```
server
  fb_messages
    bot.js
    fb_message.js
    index.js
    webhook.js
  env.js
  main.js
```



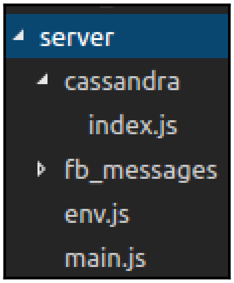
setBouquetType (context, entities)

bouquet

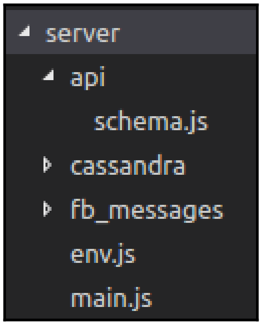


setBouquetSize (context, entities)

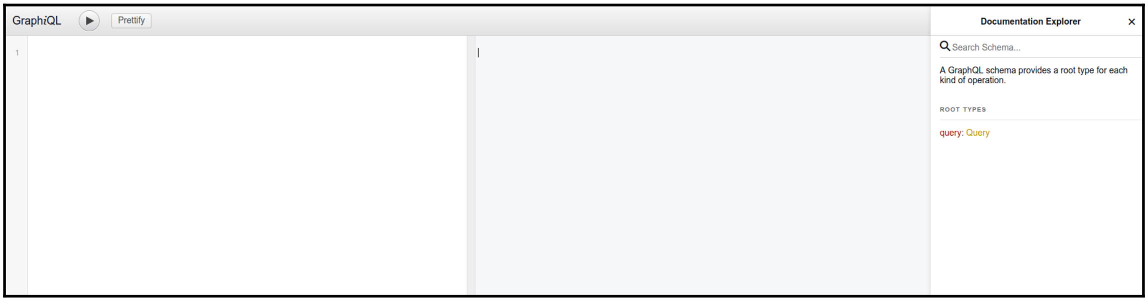
bouquet && bouquetSize



```
server
  cassandra
    index.js
  fb_messages
  env.js
  main.js
```



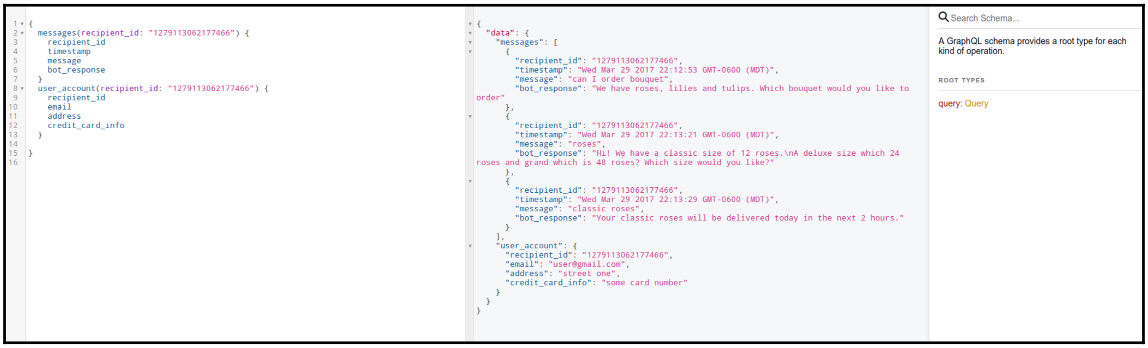
```
◢ server
  ◢ api
      schema.js
  ▷ cassandra
  ▷ fb_messages
    env.js
    main.js
```



A GraphQL schema provides a root type for each kind of operation.

ROOT TYPES

query: Query



```
{
  messages(recipient_id: "1279113062177466") {
    recipient_id
    timestamp
    message
    bot_response
  }
  user_account(recipient_id: "1279113062177466") {
    recipient_id
    email
    address
    credit_card_info
  }

}
```

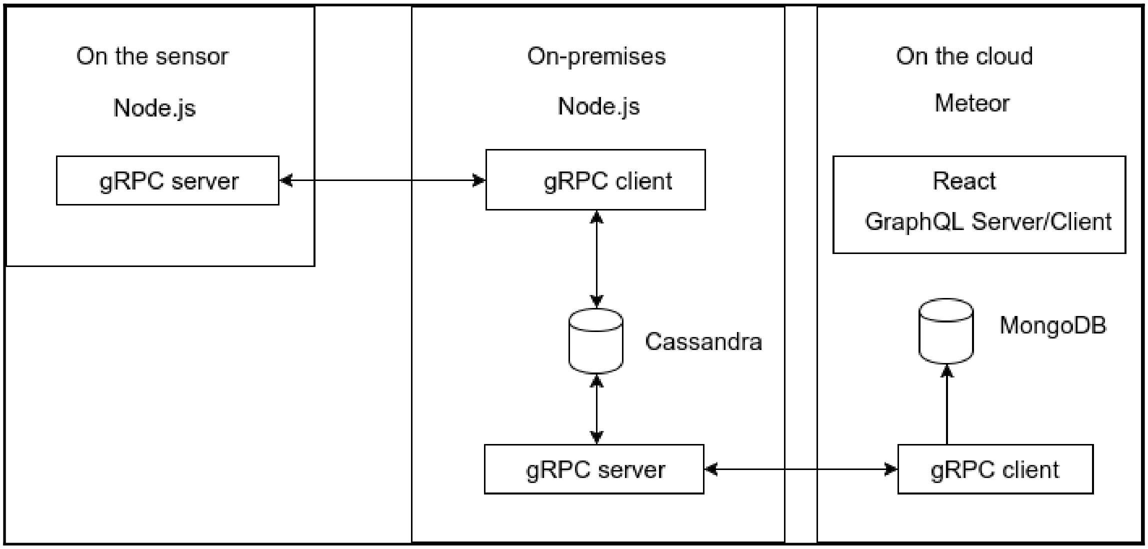
```
{
  "data": {
    "messages": [
      {
        "recipient_id": "1279113062177466",
        "timestamp": "Wed Mar 29 2017 22:12:53 GMT-0600 (MDT)",
        "message": "can I order bouquet",
        "bot_response": "We have roses, lilies and tulips. Which bouquet would you like to
order"
      },
      {
        "recipient_id": "1279113062177466",
        "timestamp": "Wed Mar 29 2017 22:13:21 GMT-0600 (MDT)",
        "message": "roses",
        "bot_response": "Hi! We have a classic size of 12 roses.\nA deluxe size which 24
roses and grand which is 48 roses? Which size would you like?"
      },
      {
        "recipient_id": "1279113062177466",
        "timestamp": "Wed Mar 29 2017 22:13:29 GMT-0600 (MDT)",
        "message": "classic roses",
        "bot_response": "Your classic roses will be delivered today in the next 2 hours."
      }
    ],
    "user_account": {
      "recipient_id": "1279113062177466",
      "email": "user@gmail.com",
      "address": "street one",
      "credit_card_info": "some card number"
    }
  }
}
```

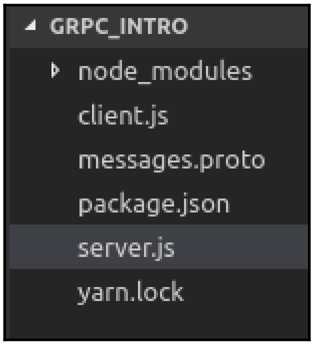
A GraphQL schema provides a root type for each kind of operation.

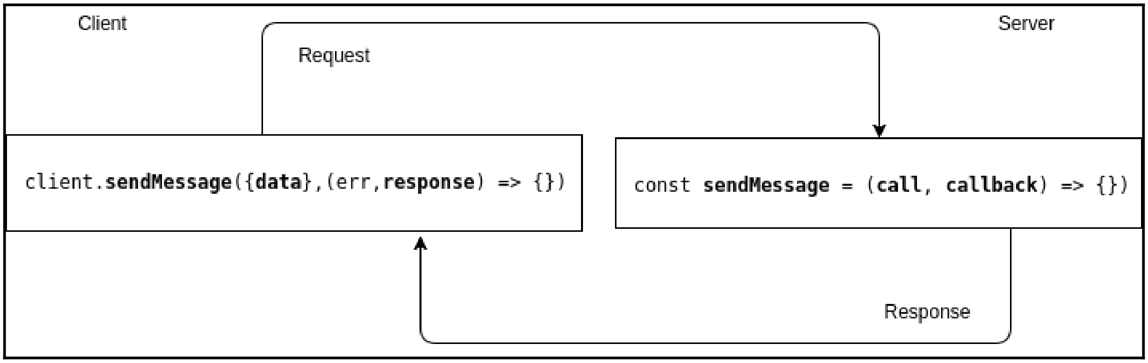
ROOT TYPES

query: Query

Chapter 9: Build Internet of Things Platform

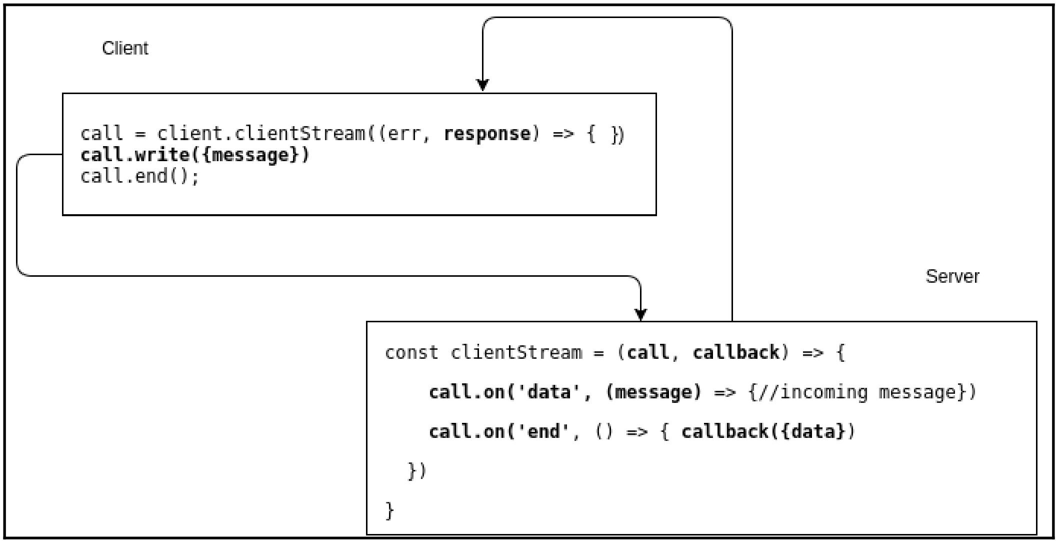



GRPC_INTRO
node_modules
client.js
messages.proto
package.json
server.js
yarn.lock



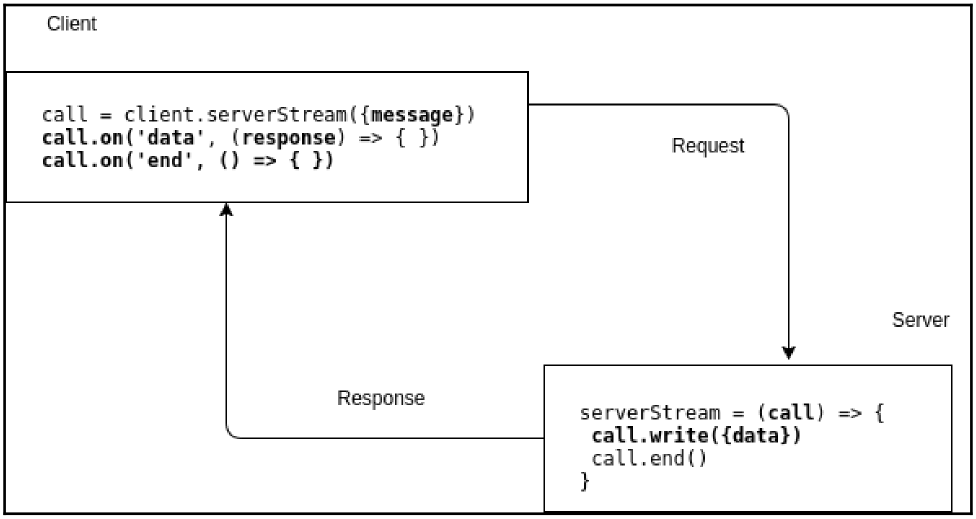
```
client.sendMessage({data},(err,response) => {})
```

```
const sendMessage = (call, callback) => {})
```



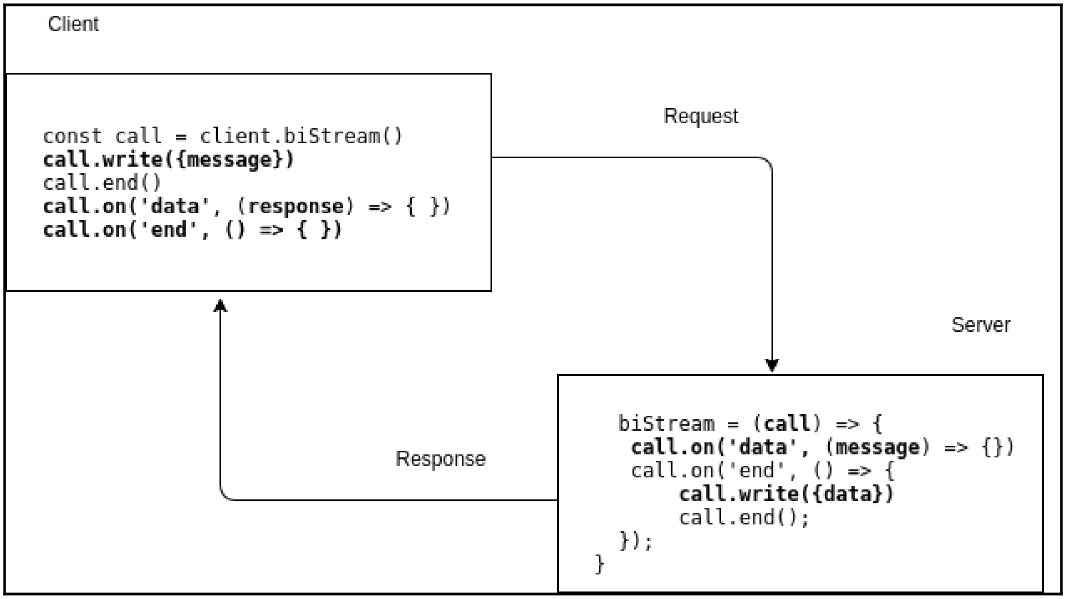
```
call = client.clientStream((err, response) => { })
call.write({message})
call.end();
```

```
const clientStream = (call, callback) => {
    call.on('data', (message) => {//incoming message})
    call.on('end', () => { callback({data})
  })
}
```



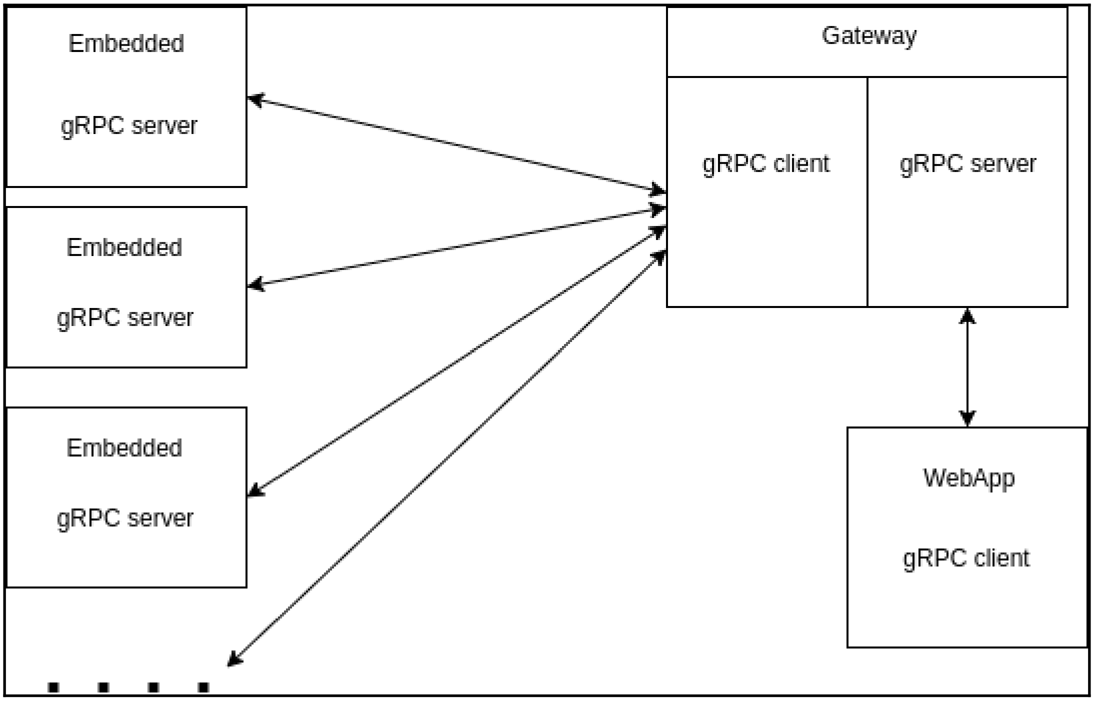
```
call = client.serverStream({message})
call.on('data', (response) => { })
call.on('end', () => { })
```

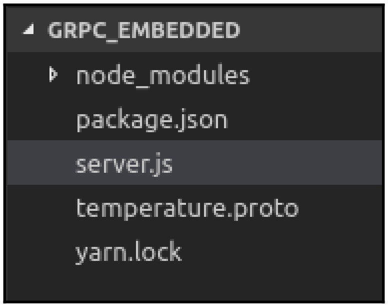
```
serverStream = (call) => {
 call.write({data})
 call.end()
}
```



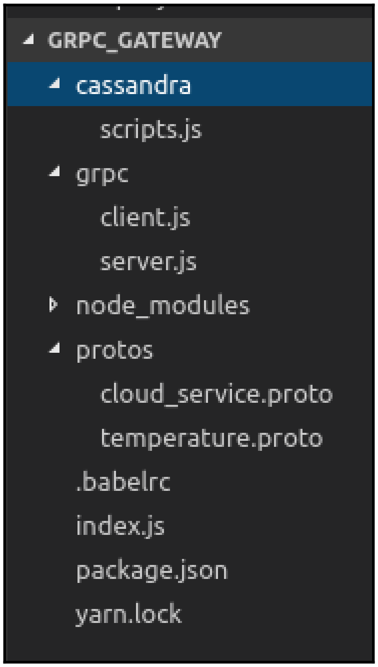
```
const call = client.biStream()
call.write({message})
call.end()
call.on('data', (response) => { })
call.on('end', () => { })
```

```
  biStream = (call) => {
   call.on('data', (message) => {})
   call.on('end', () => {
       call.write({data})
       call.end();
   });
}
```

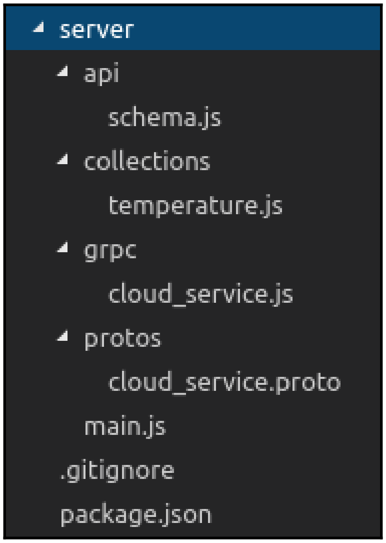




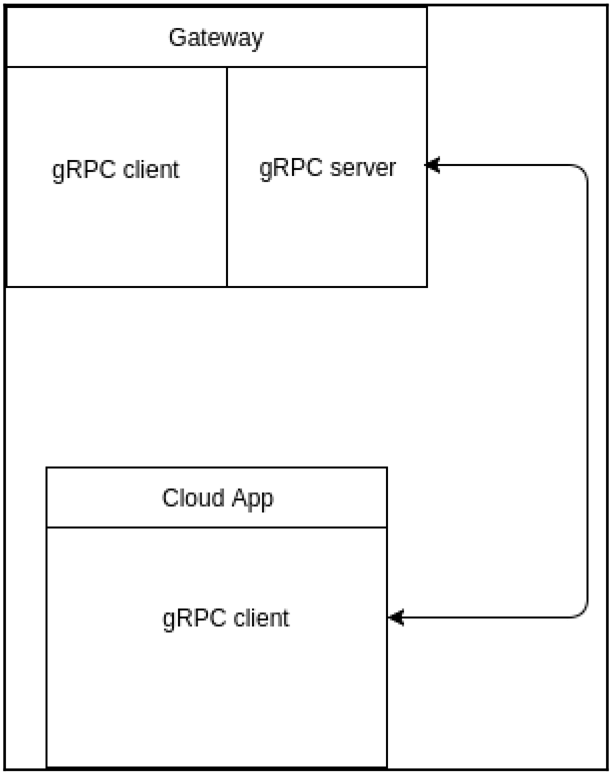
```
◢ GRPC_EMBEDDED
  ▷ node_modules
    package.json
    server.js
    temperature.proto
    yarn.lock
```

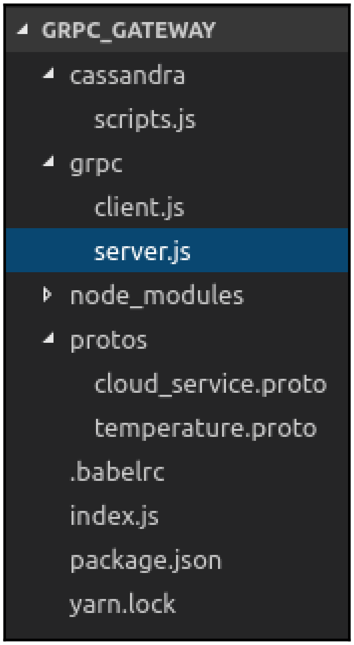


```
◢ GRPC_GATEWAY
  ◢ cassandra
      scripts.js
  ◢ grpc
      client.js
      server.js
  ▷ node_modules
  ◢ protos
      cloud_service.proto
      temperature.proto
    .babelrc
    index.js
    package.json
    yarn.lock
```

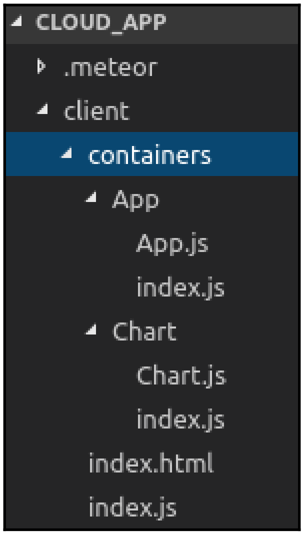


```
server
  api
    schema.js
  collections
    temperature.js
  grpc
    cloud_service.js
  protos
    cloud_service.proto
  main.js
.gitignore
package.json
```

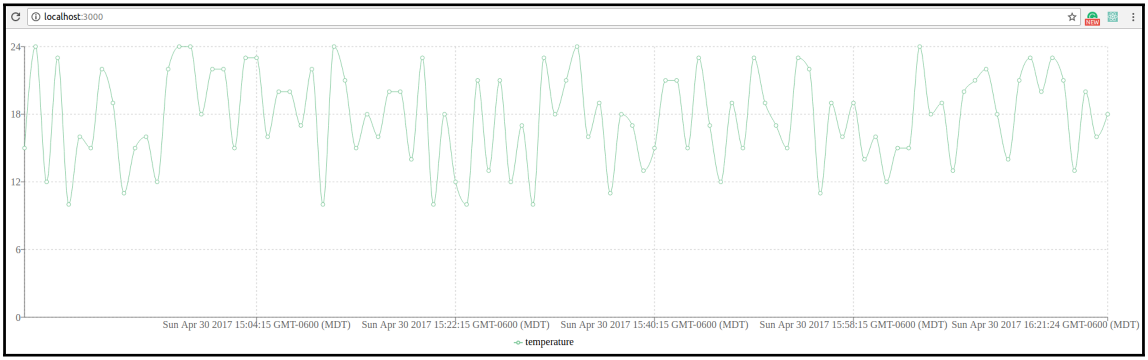




```
◢ GRPC_GATEWAY
  ◢ cassandra
      scripts.js
  ◢ grpc
      client.js
      server.js
  ▷ node_modules
  ◢ protos
      cloud_service.proto
      temperature.proto
    .babelrc
    index.js
    package.json
    yarn.lock
```



```
◢ CLOUD_APP
  ▷ .meteor
  ◢ client
    ◢ containers
      ◢ App
          App.js
          index.js
      ◢ Chart
          Chart.js
          index.js
      index.html
      index.js
```


localhost:3000
24
18
12
6
0
Sun Apr 30 2017 15:04:15 GMT-0600 (MDT)
Sun Apr 30 2017 15:22:15 GMT-0600 (MDT)
Sun Apr 30 2017 15:40:15 GMT-0600 (MDT)
Sun Apr 30 2017 15:58:15 GMT-0600 (MDT)
Sun Apr 30 2017 16:21:24 GMT-0600 (MDT)
temperature