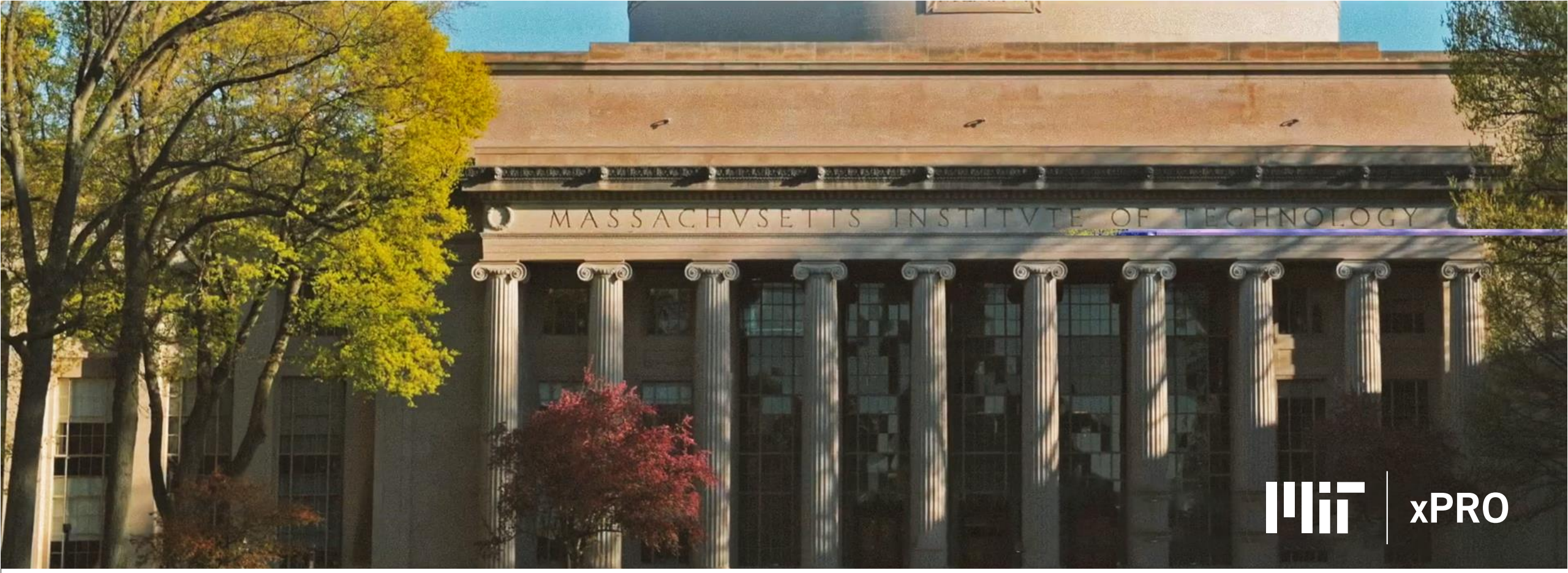
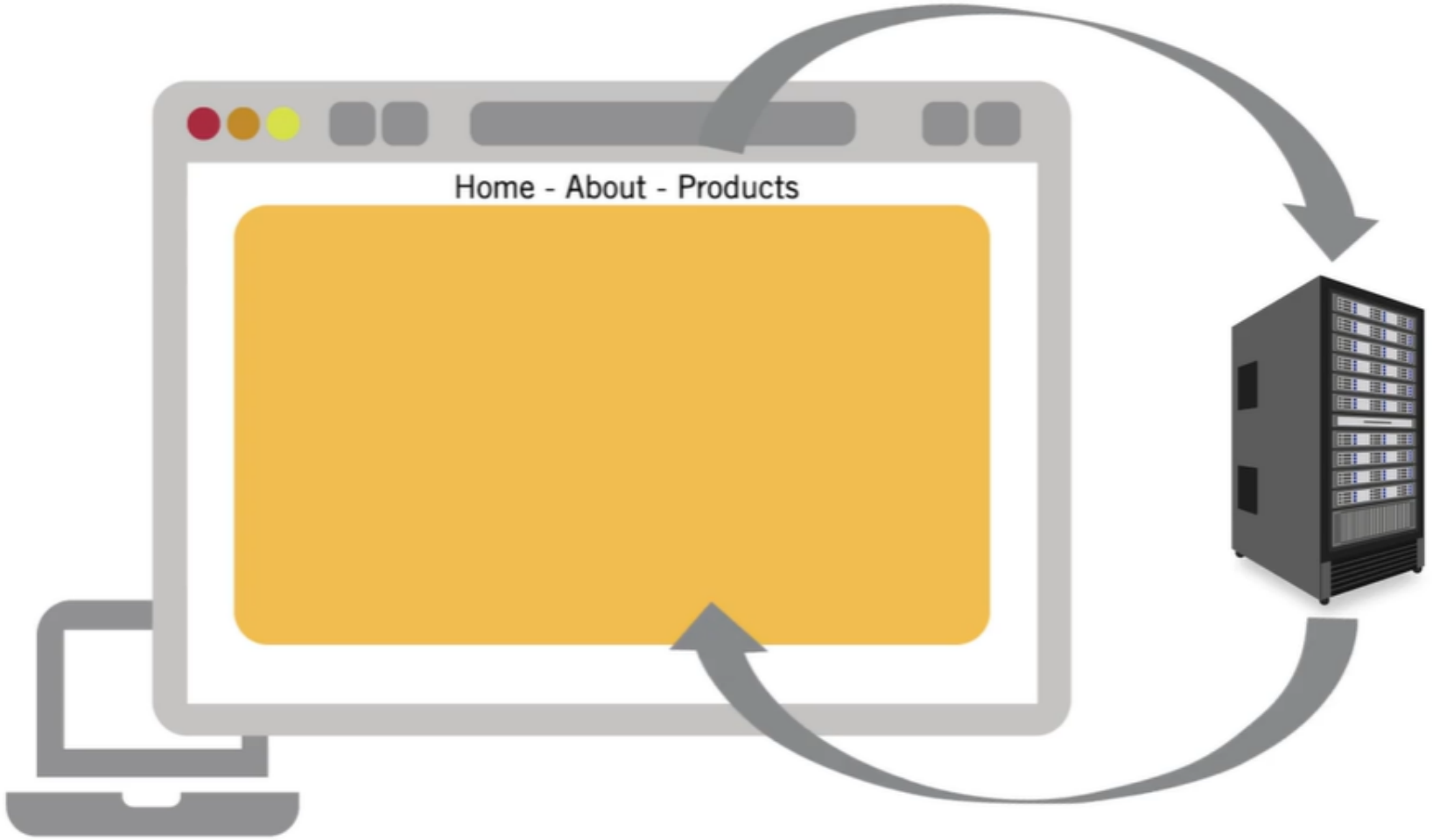


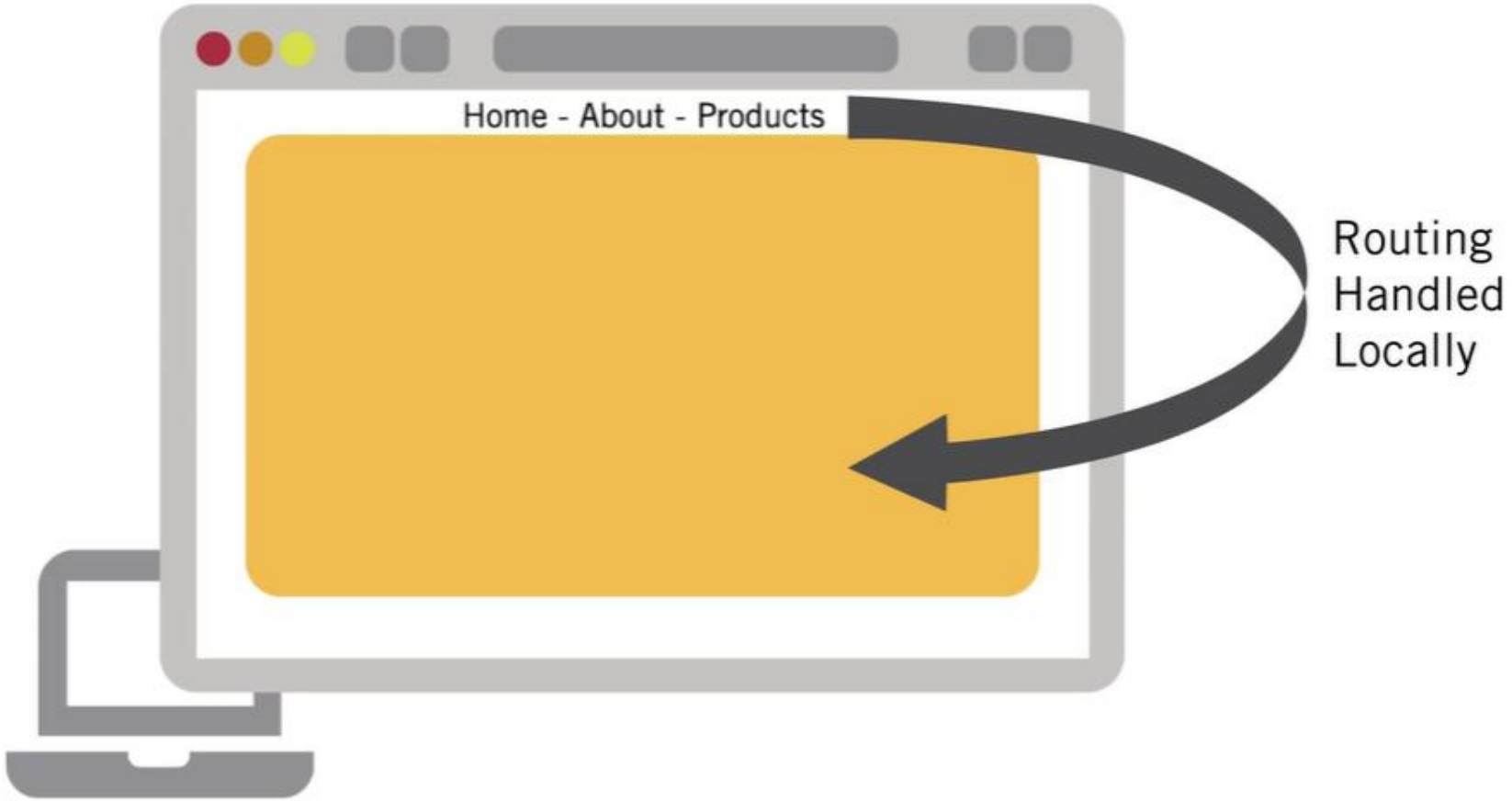
Professional Certificate in Coding: Full Stack Development with MERN: Week 18

Bad Bank Exercise

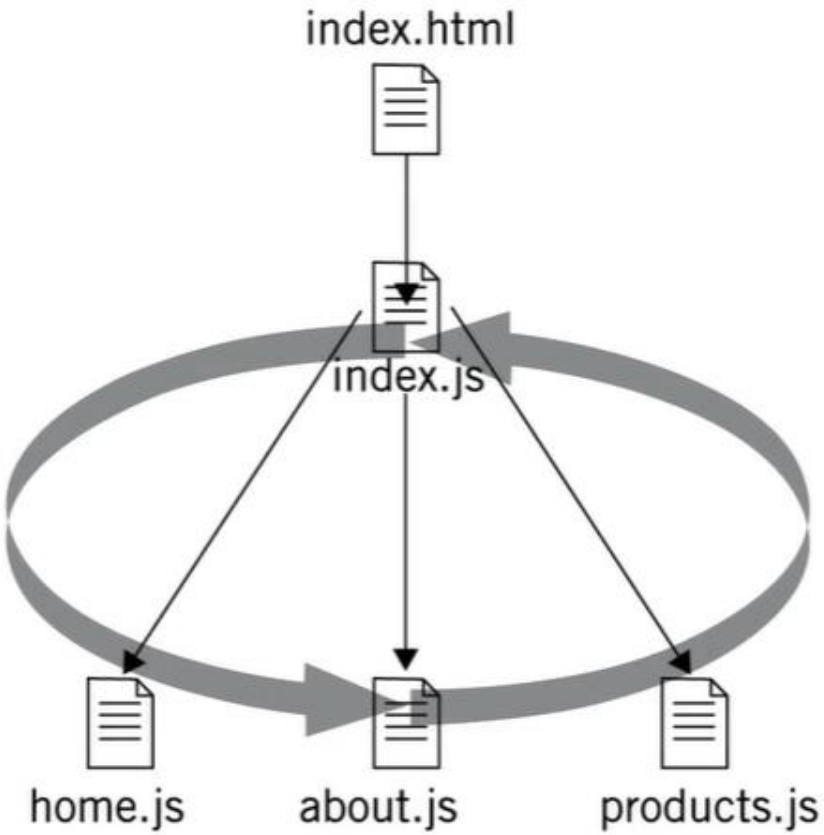
History – Server Request Per Page



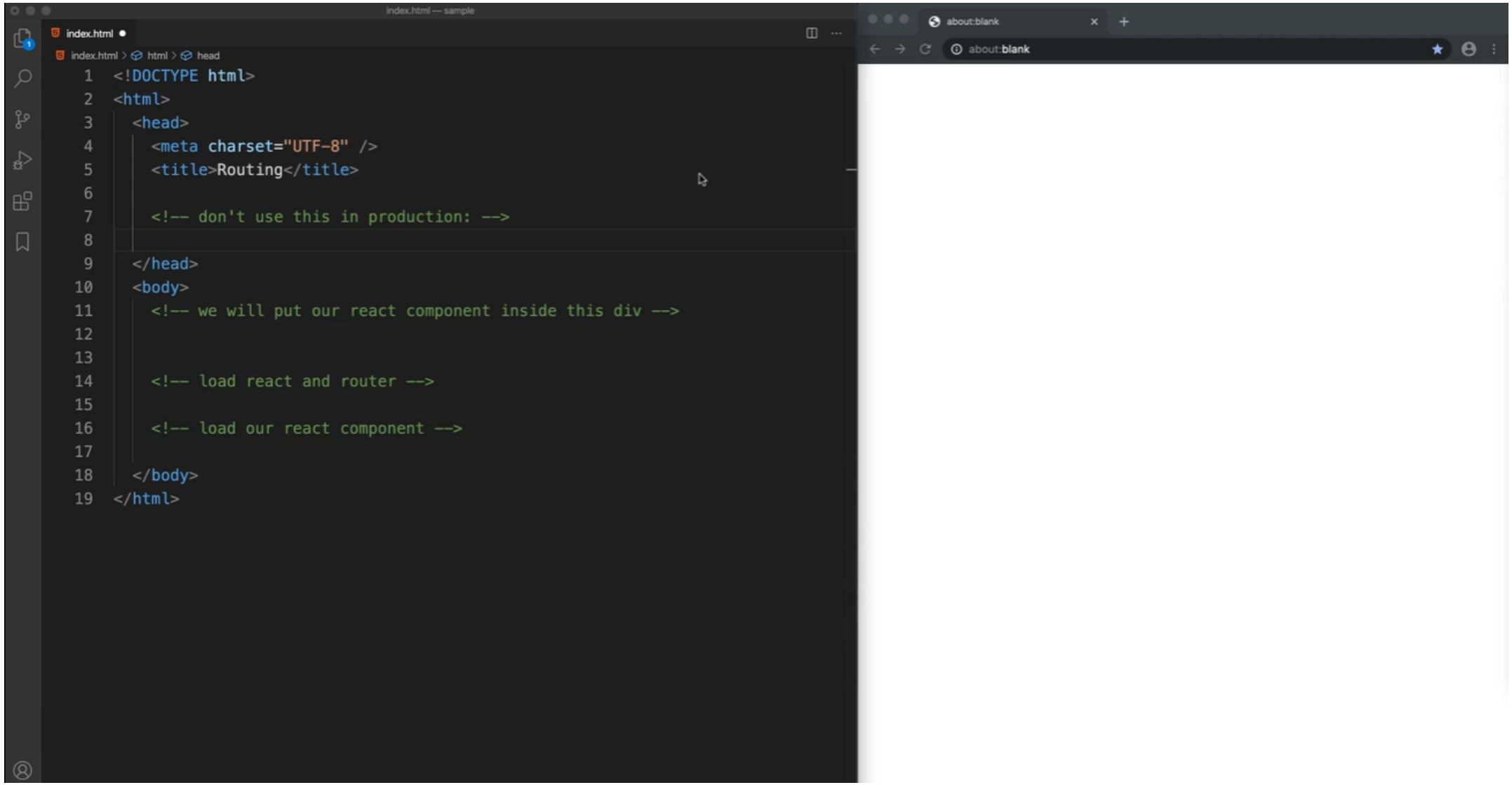
SPA – Single Page Application – Local Routing



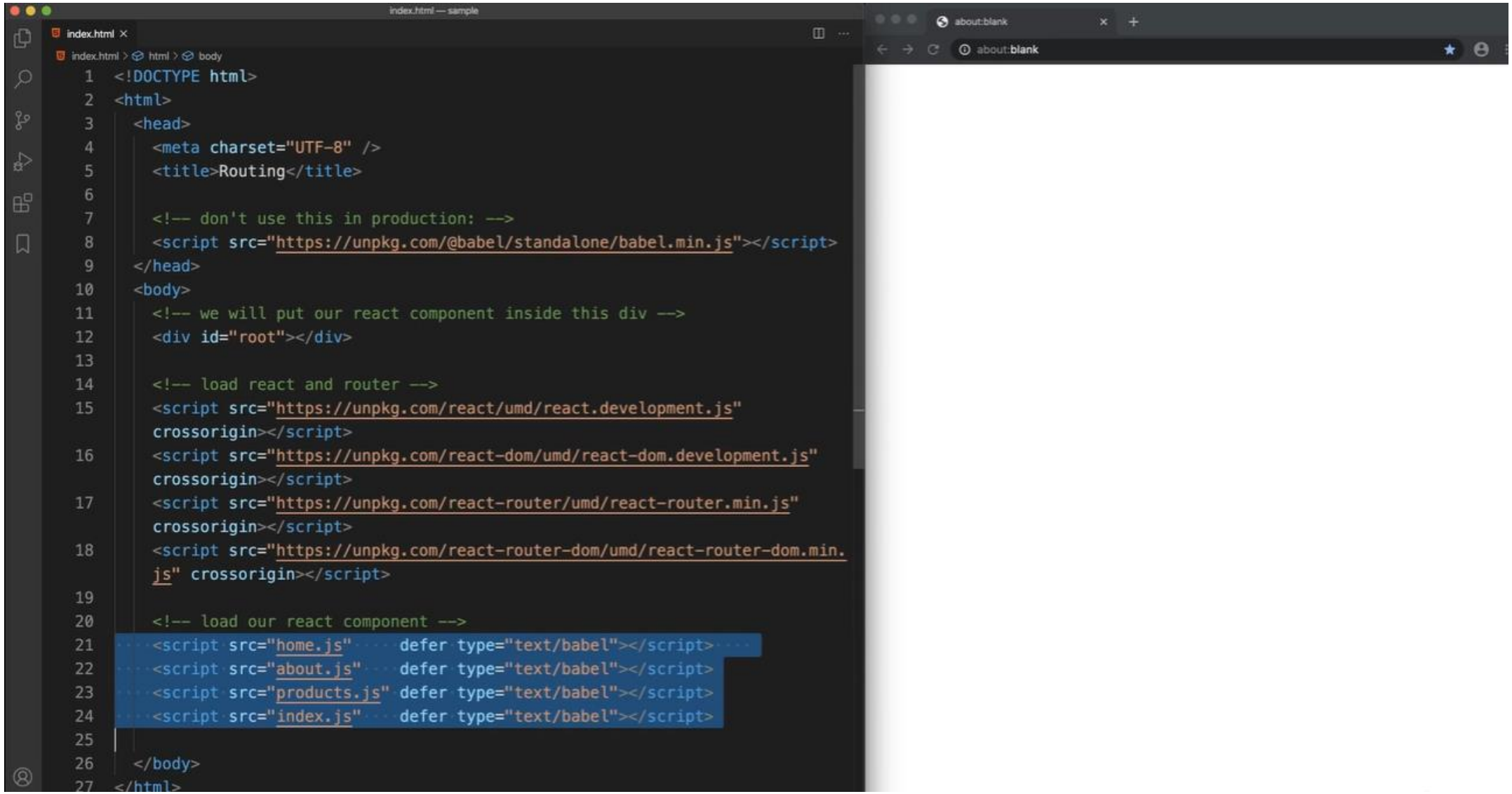
Routing



Routing: Demonstration (1/19)

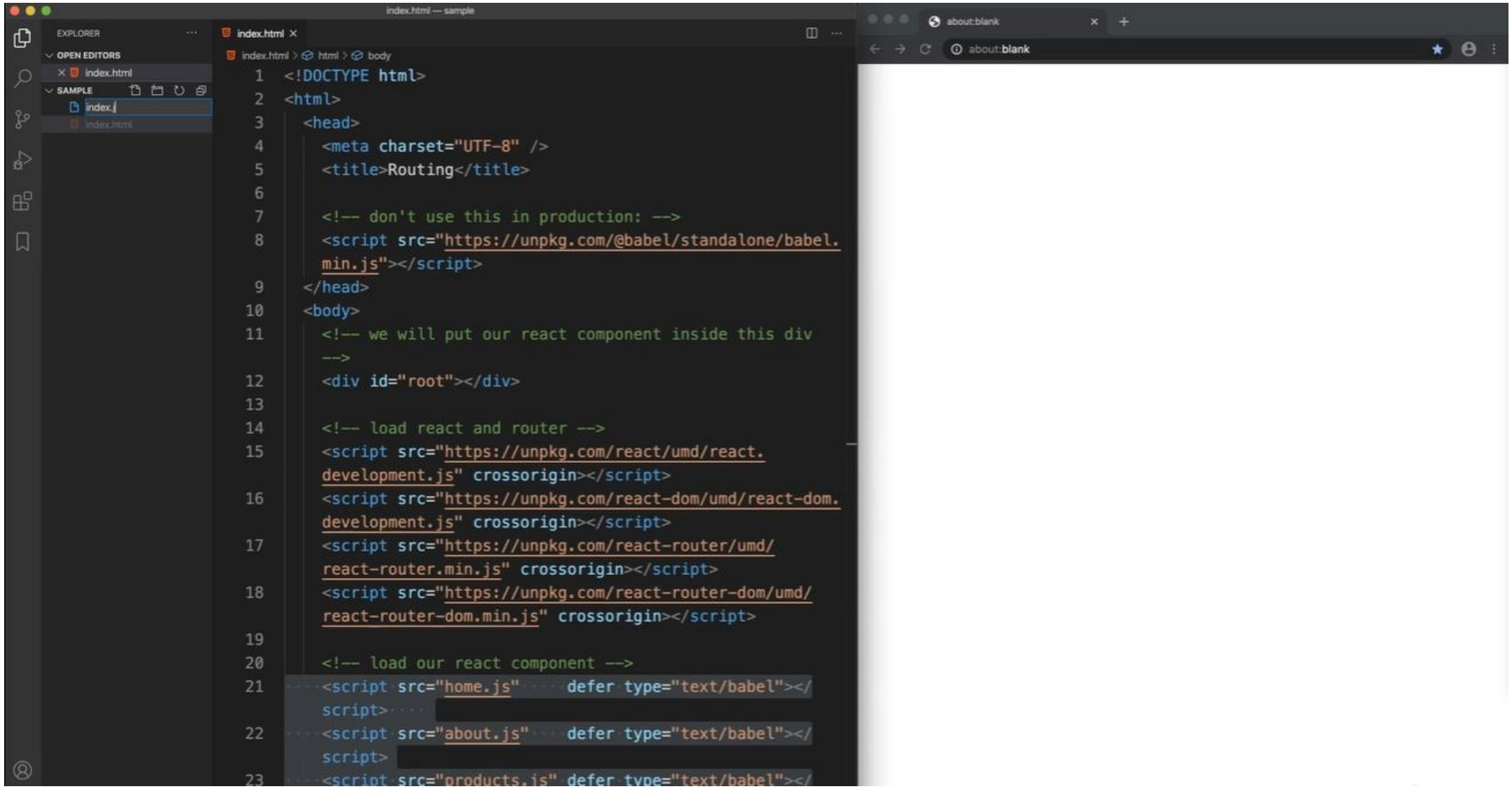


Routing: Demonstration (2/19)

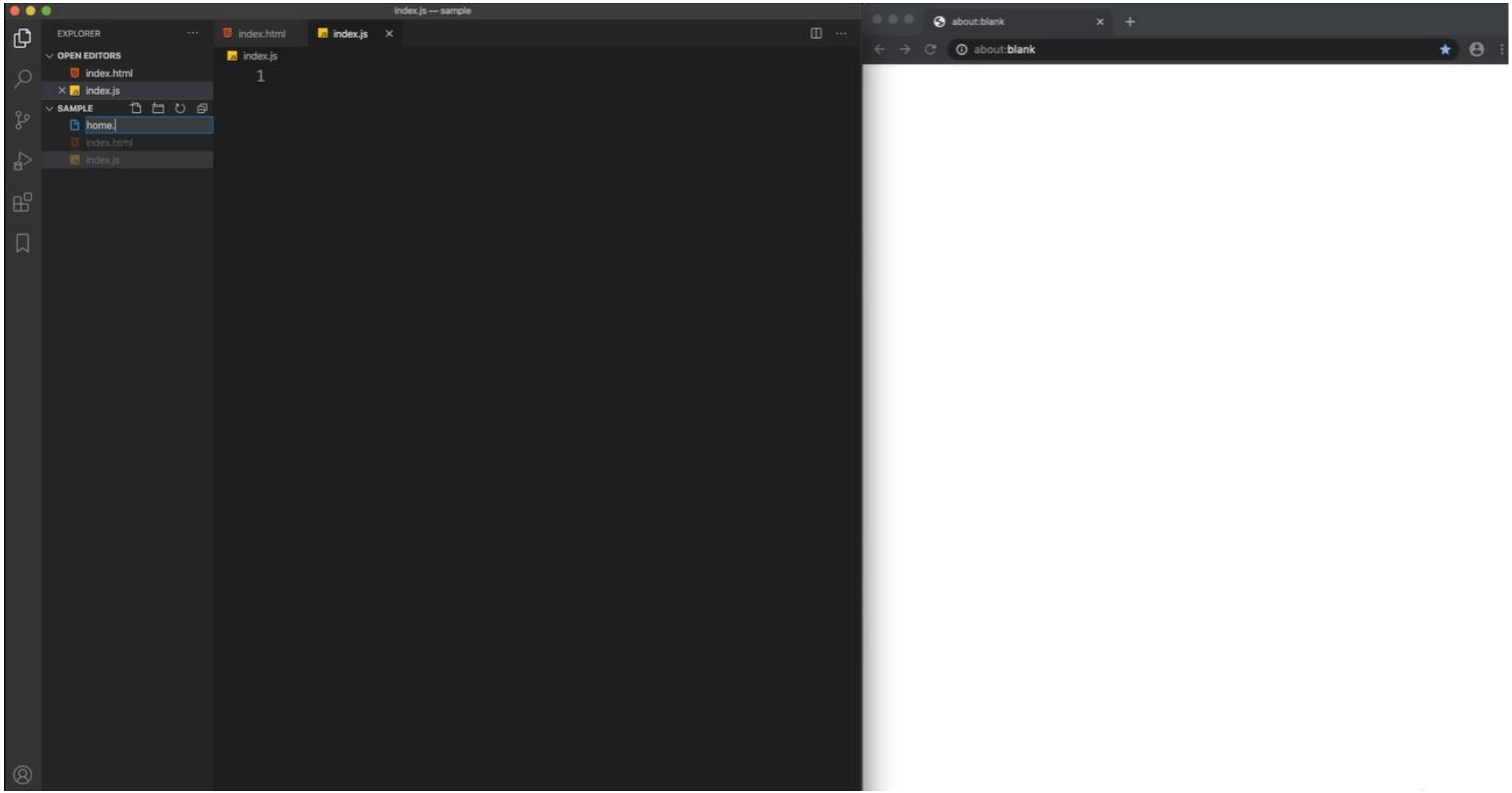



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <title>Routing</title>

    <!-- don't use this in production: -->
    <script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
  </head>
  <body>
    <!-- we will put our react component inside this div -->
    <div id="root"></div>

    <!-- load react and router -->
    <script src="https://unpkg.com/react/umd/react.development.js"
    crossorigin></script>
    <script src="https://unpkg.com/react-dom/umd/react-dom.development.js"
    crossorigin></script>
    <script src="https://unpkg.com/react-router/umd/react-router.min.js"
    crossorigin></script>
    <script src="https://unpkg.com/react-router-dom/umd/react-router-dom.min.
    js" crossorigin></script>

    <!-- load our react component -->
    <script src="home.js"     defer type="text/babel"></script>
    <script src="about.js"    defer type="text/babel"></script>
    <script src="products.js" defer type="text/babel"></script>
    <script src="index.js"    defer type="text/babel"></script>

  </body>
</html>
```

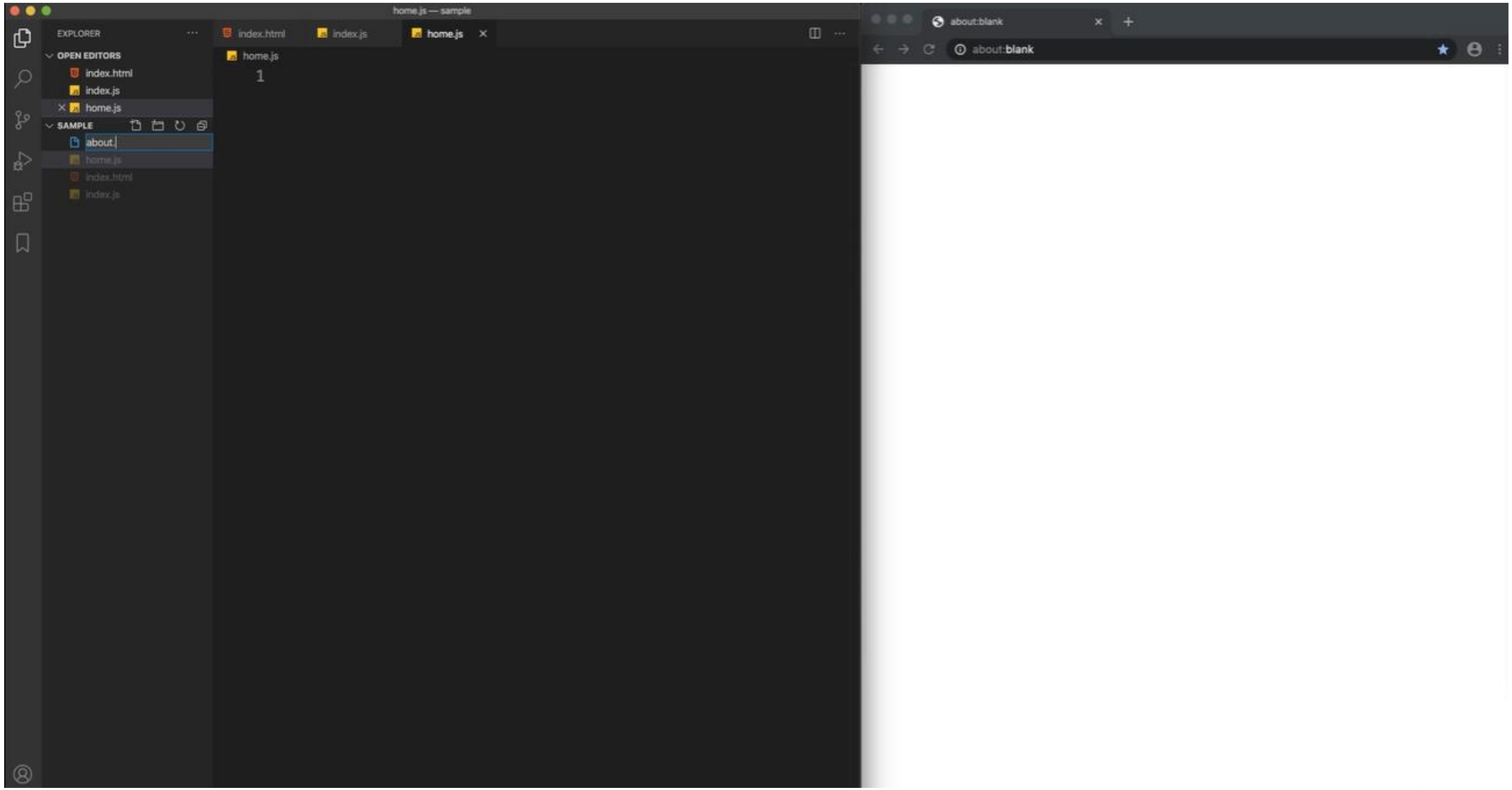
Routing: Demonstration (3/19)



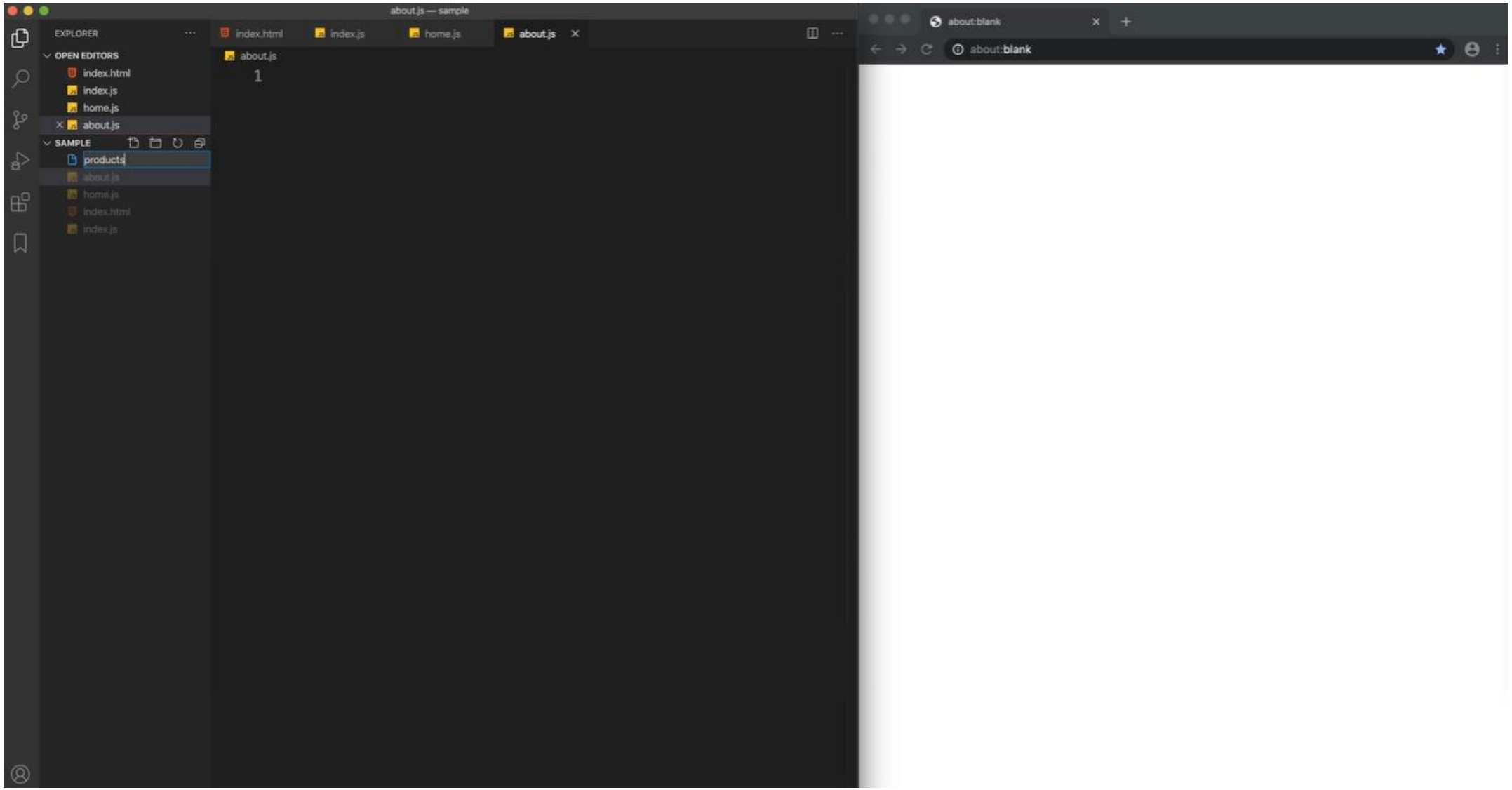
Routing: Demonstration (4/19)



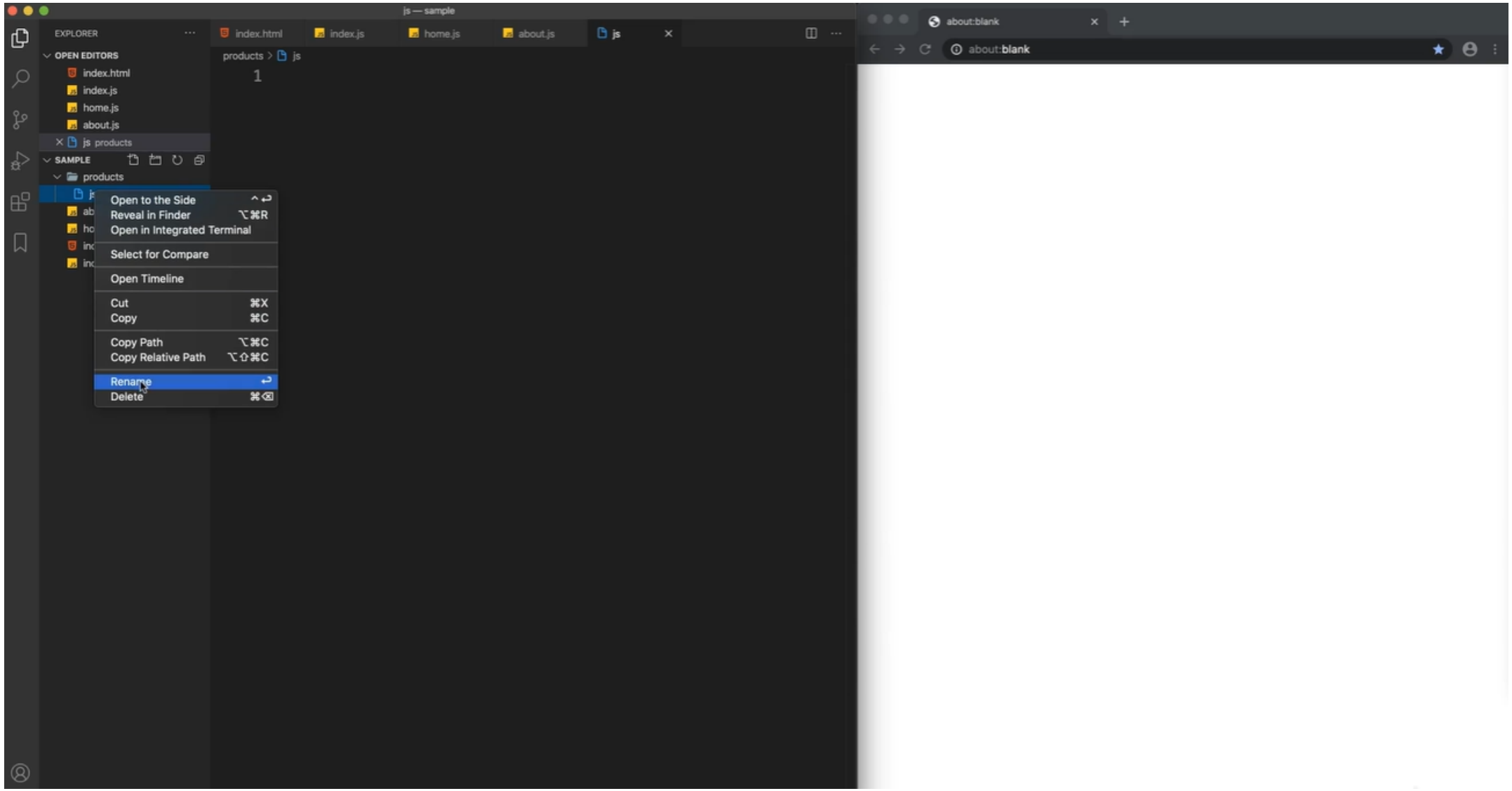
Routing: Demonstration (5/19)



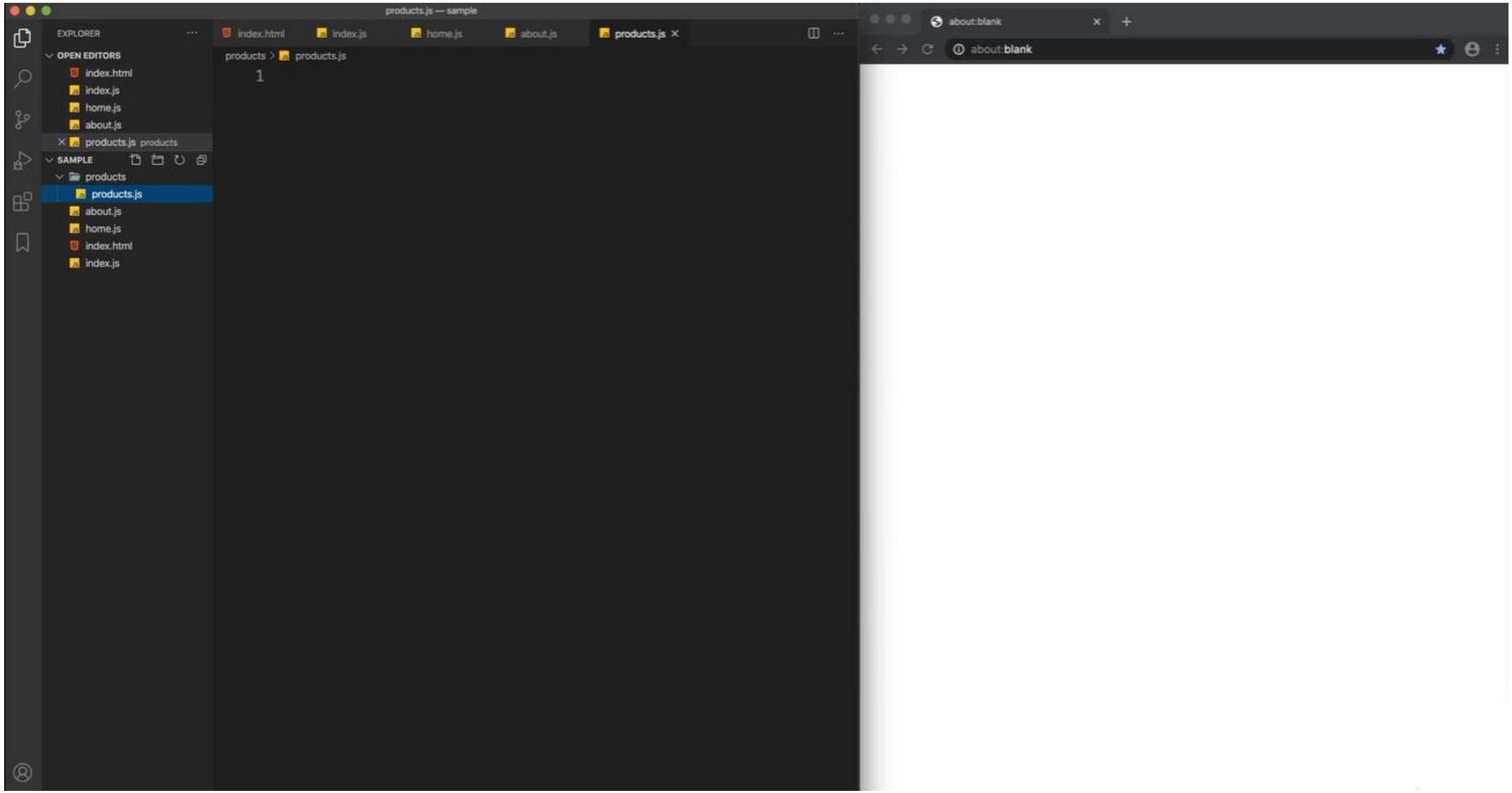
Routing: Demonstration (6/19)



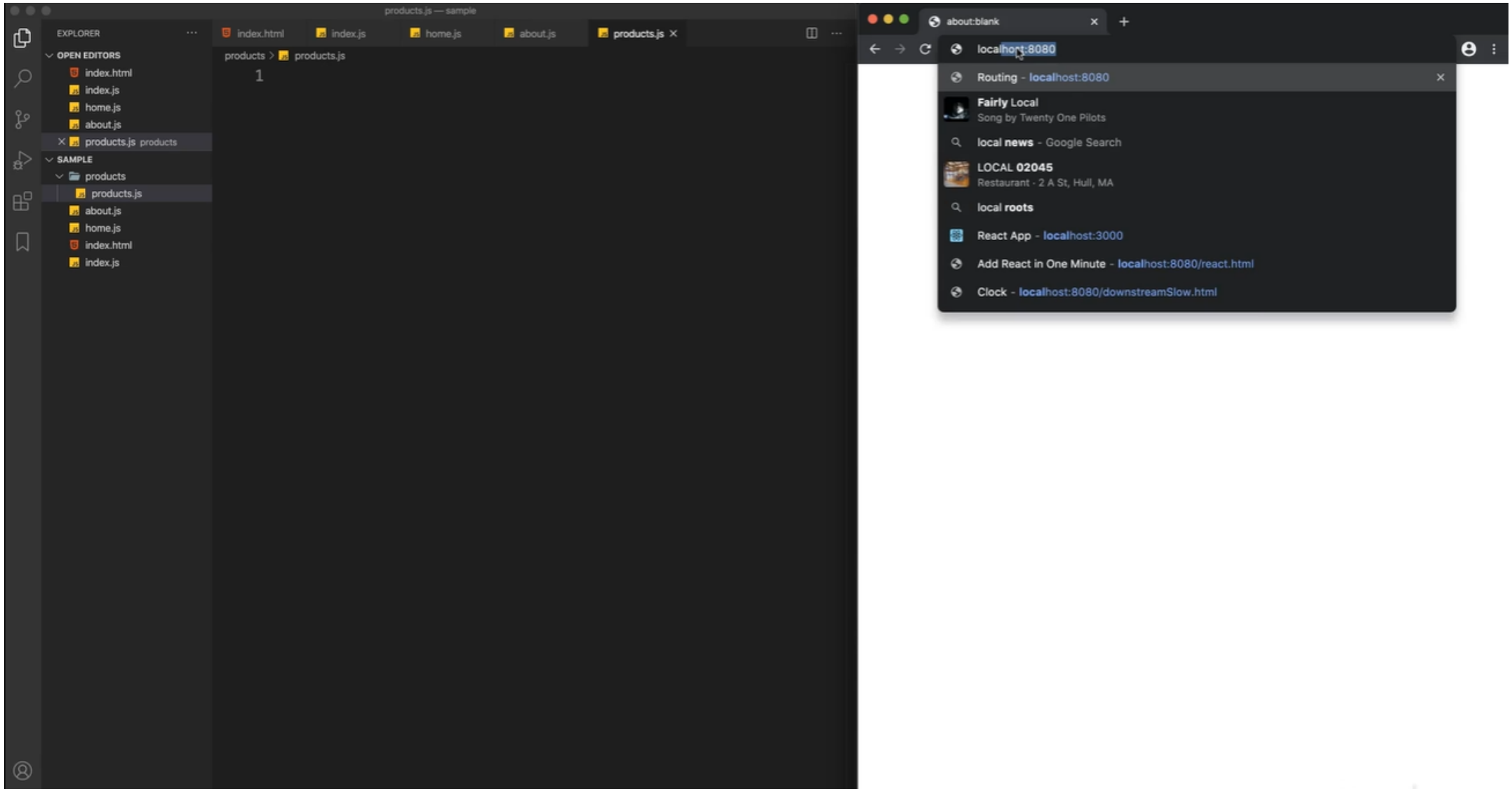
Routing: Demonstration (7/19)



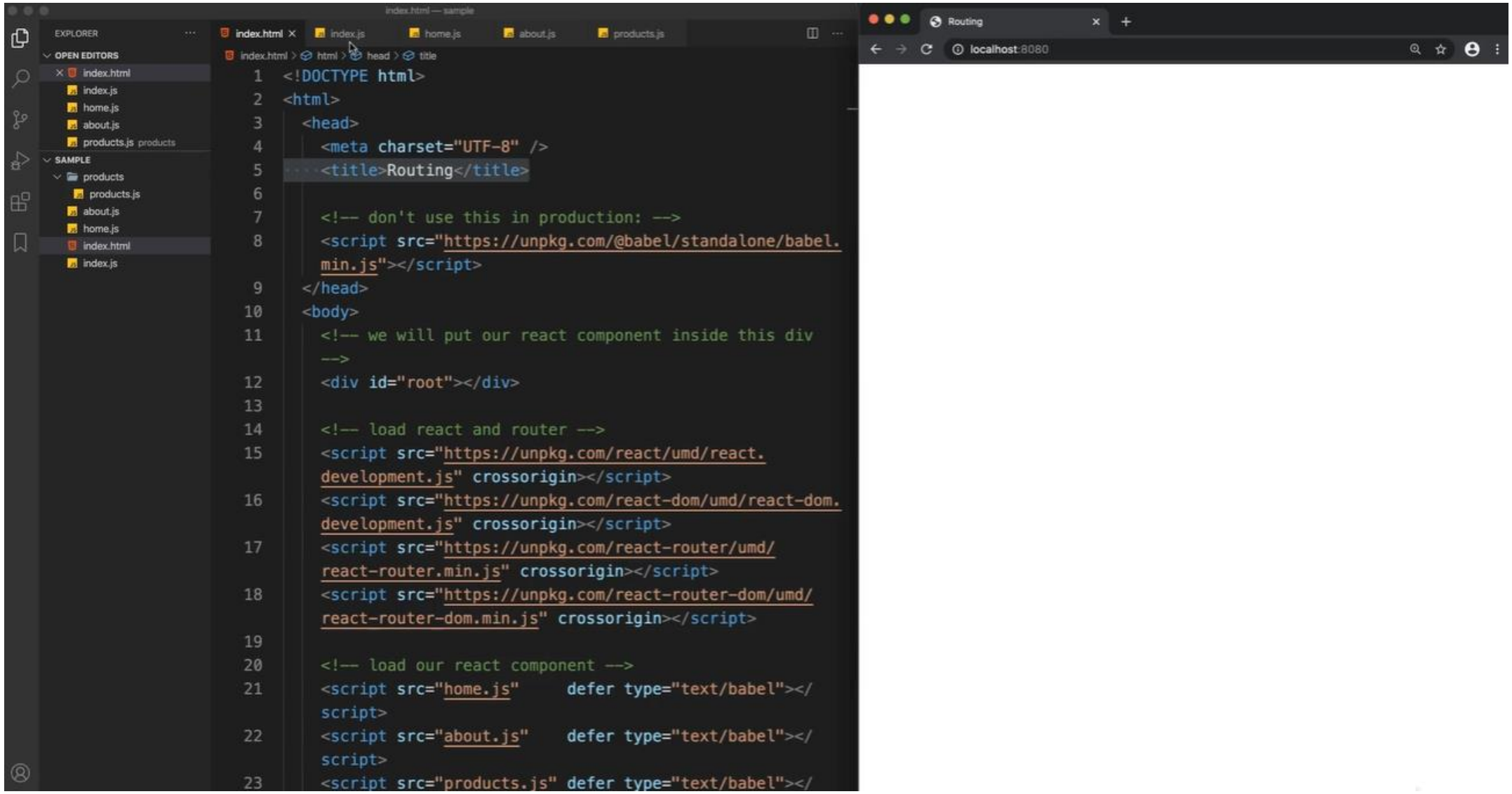
Routing: Demonstration (8/19)



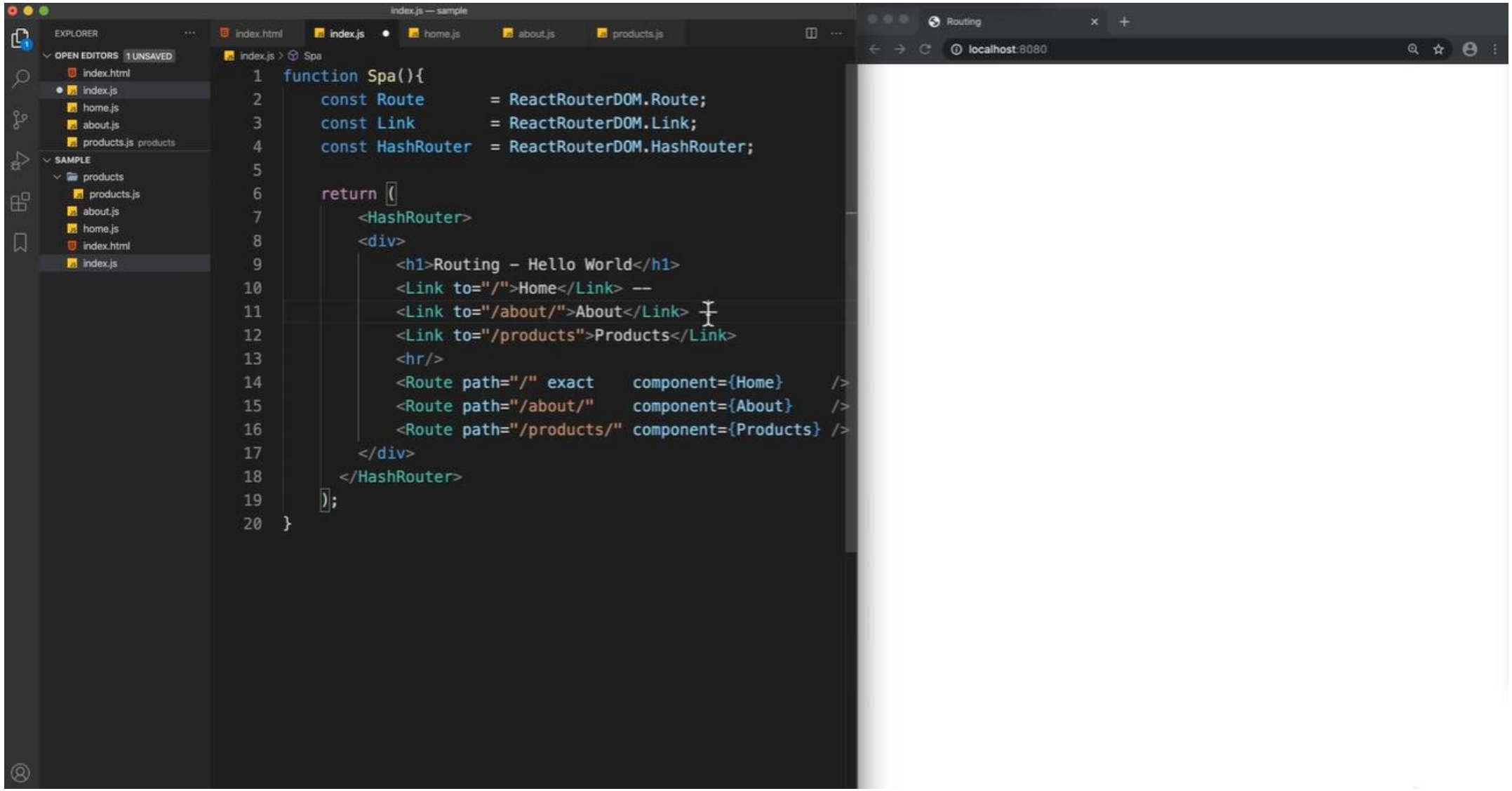
Routing: Demonstration (9/19)



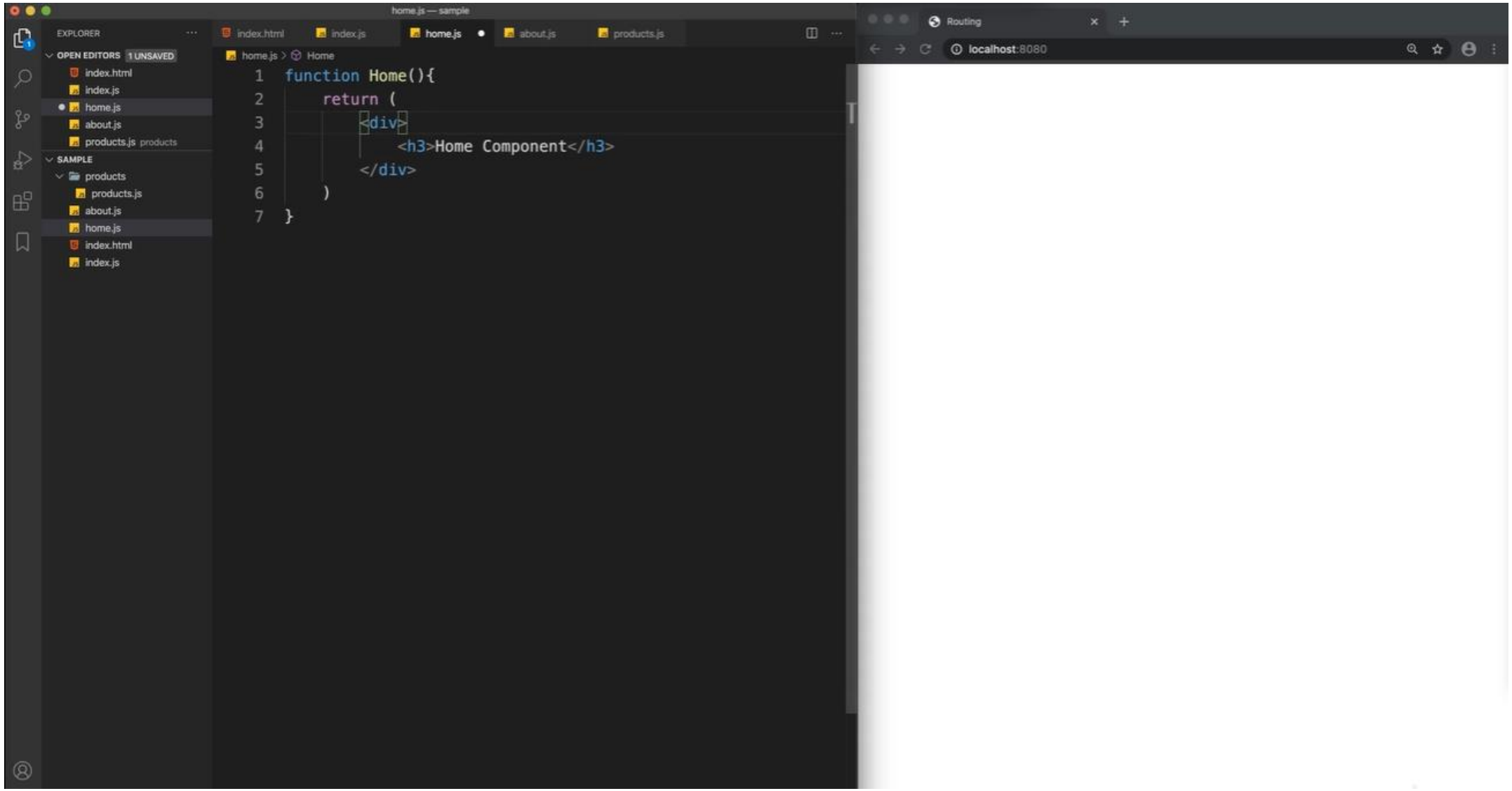
Routing: Demonstration (10/19)



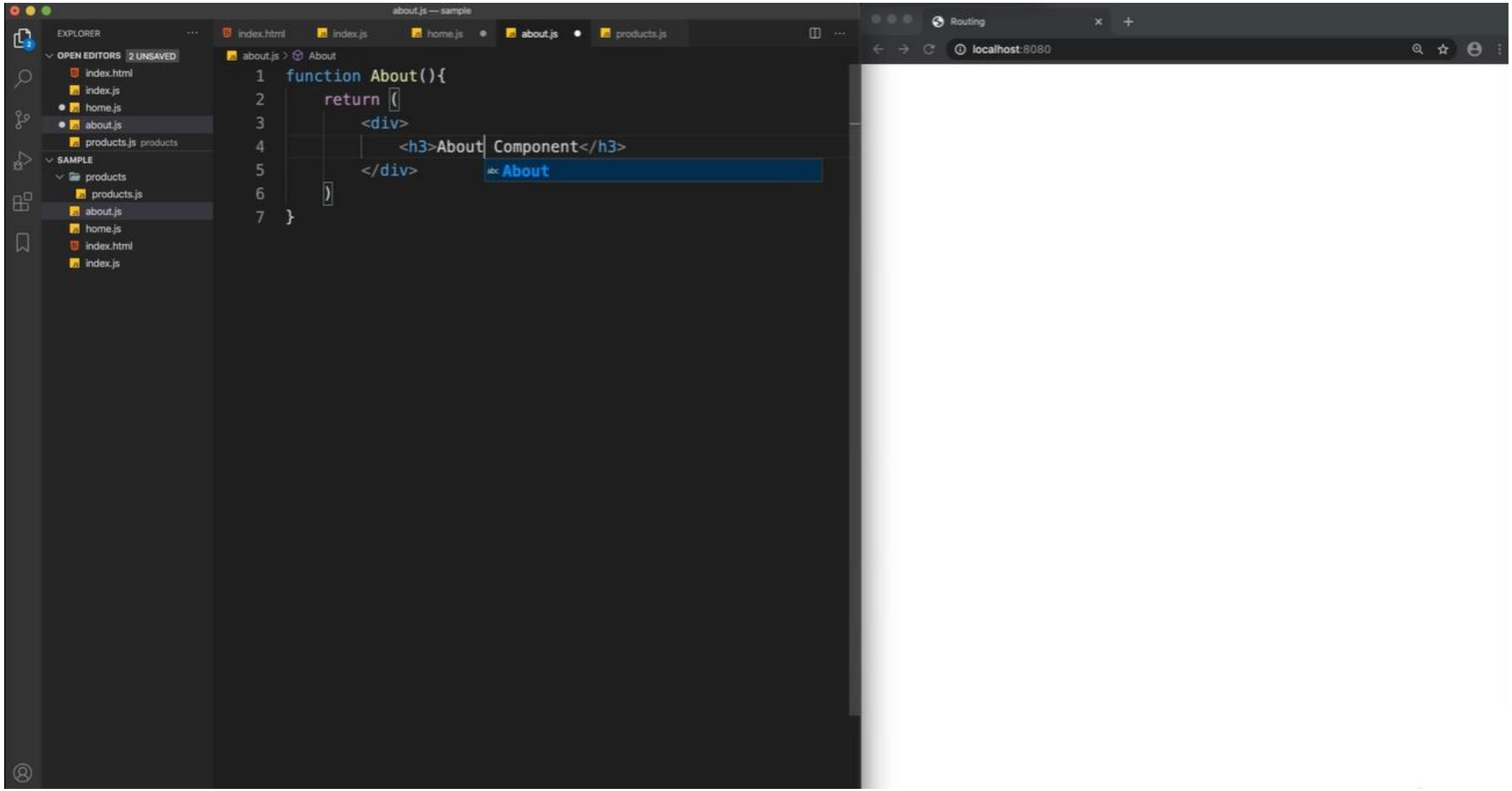
Routing: Demonstration (11/19)



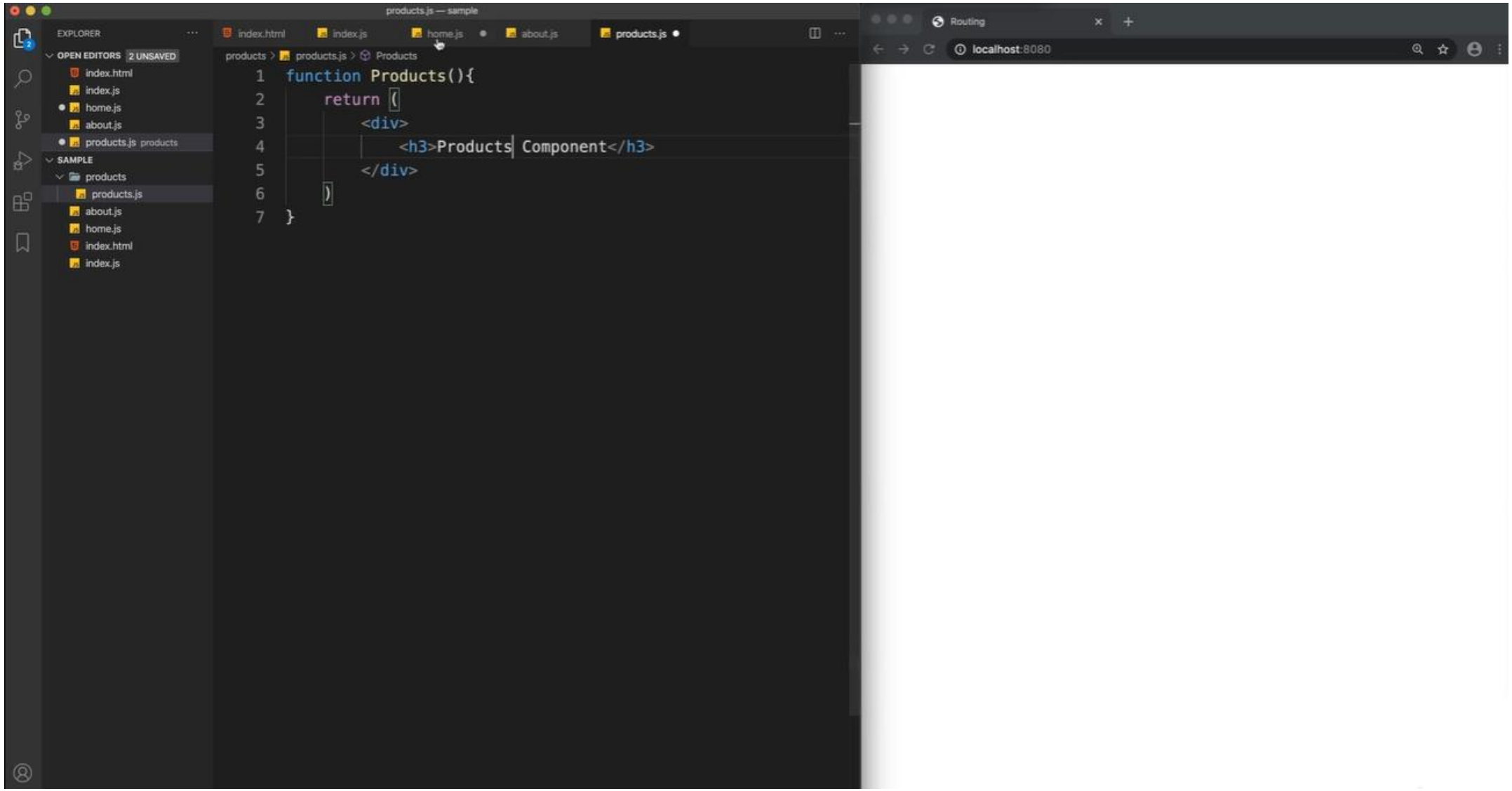
Routing: Demonstration (12/19)



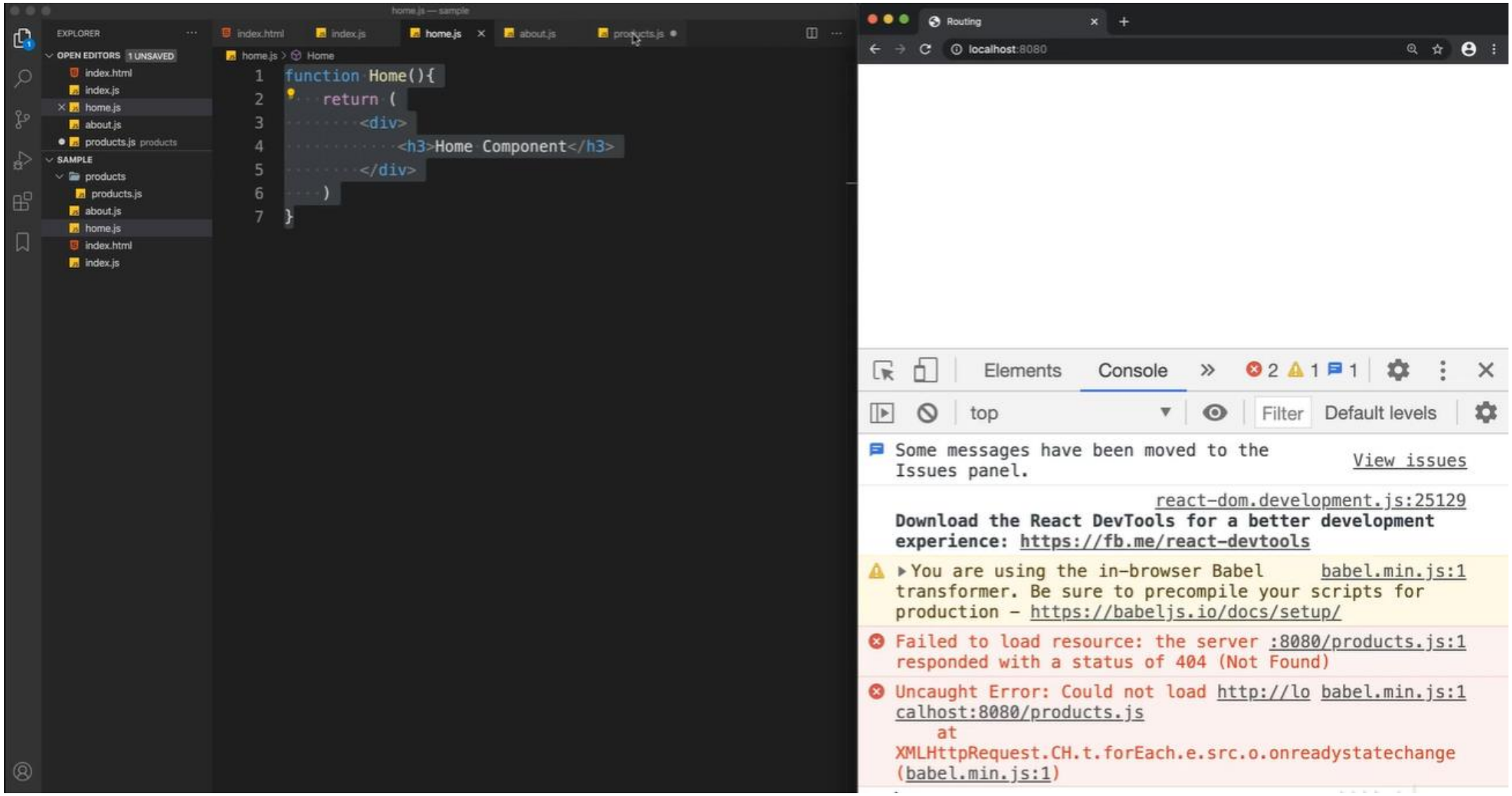
Routing: Demonstration (13/19)



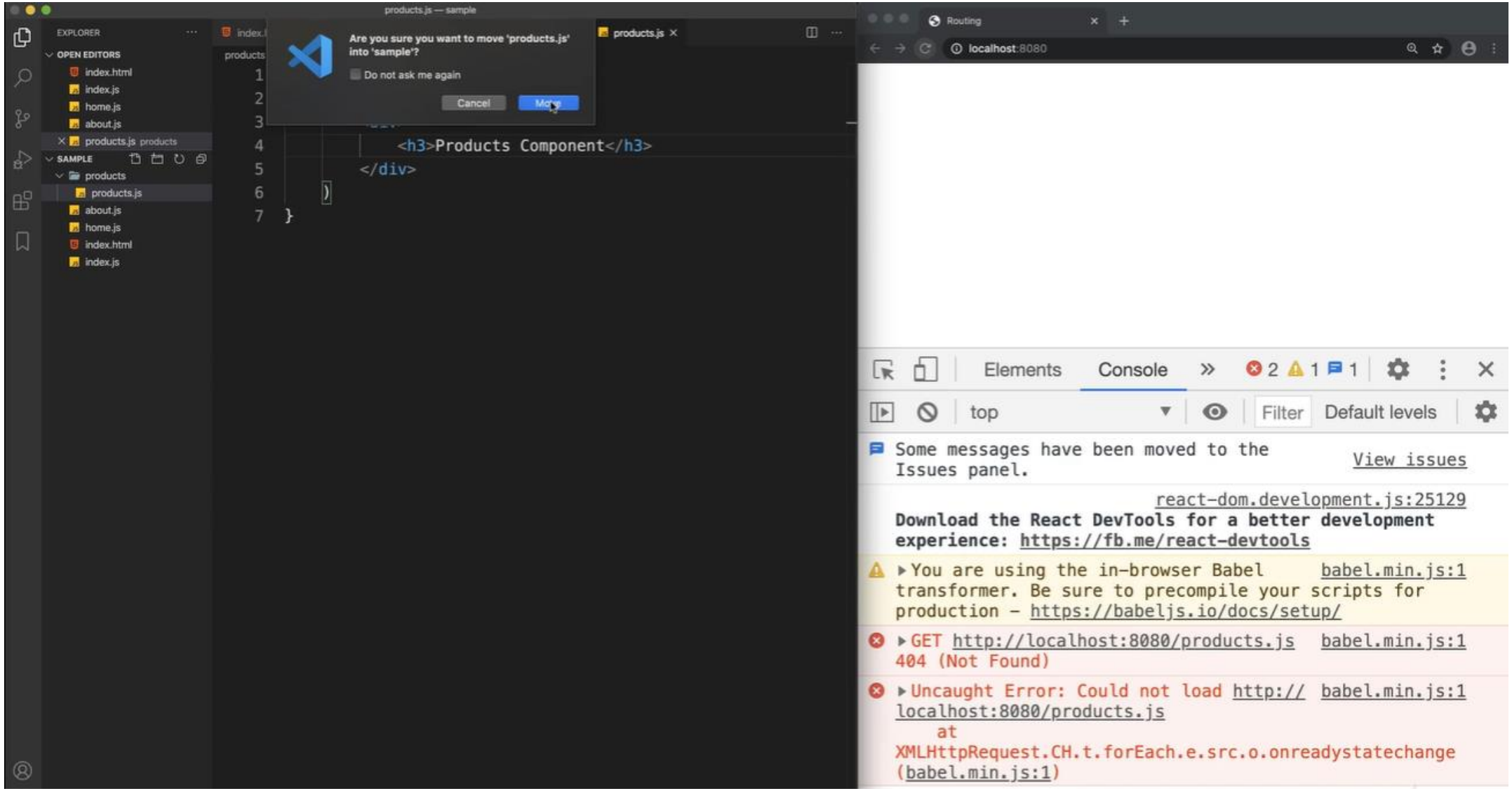
Routing: Demonstration (14/19)



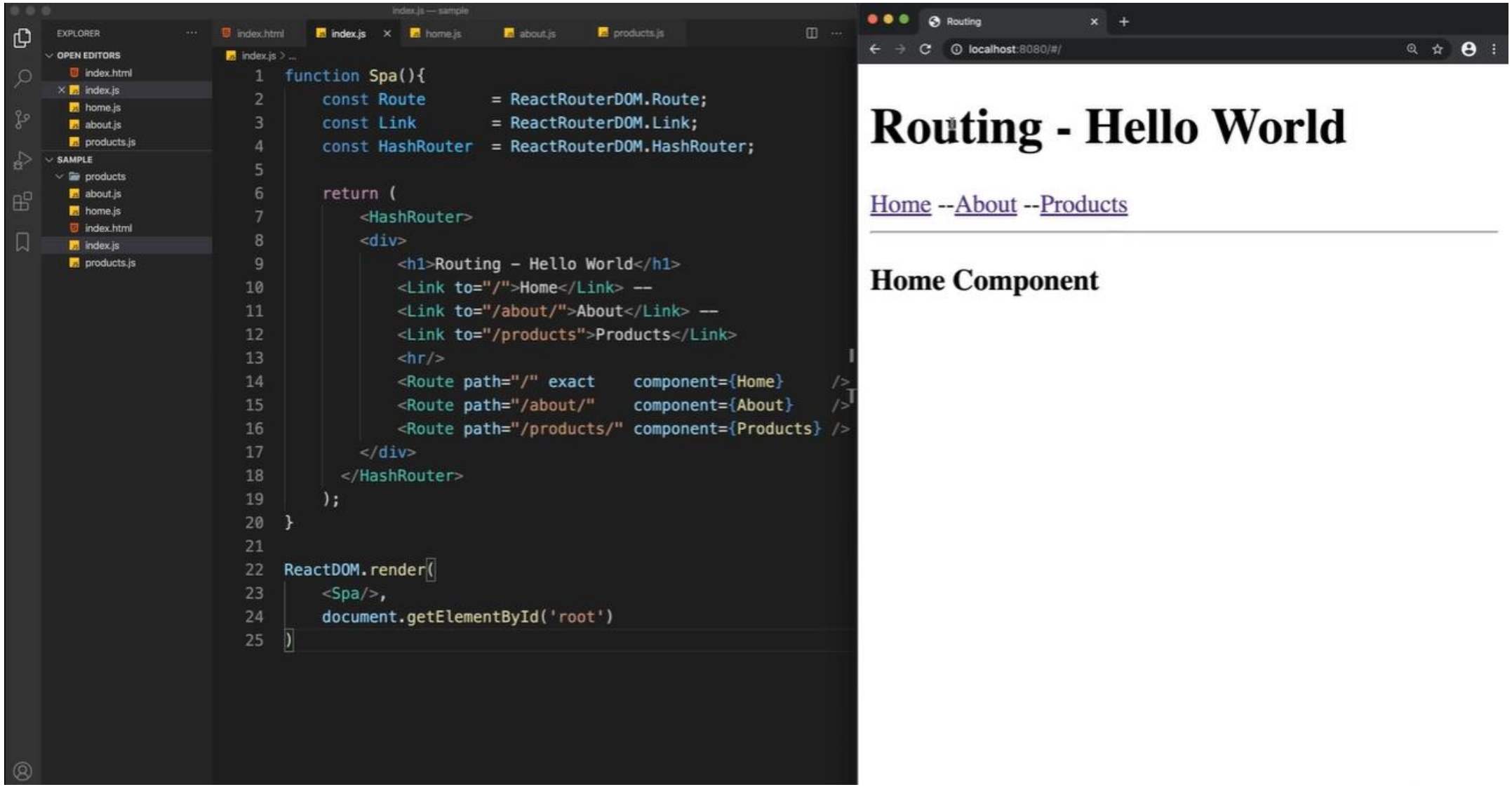
Routing: Demonstration (15/19)



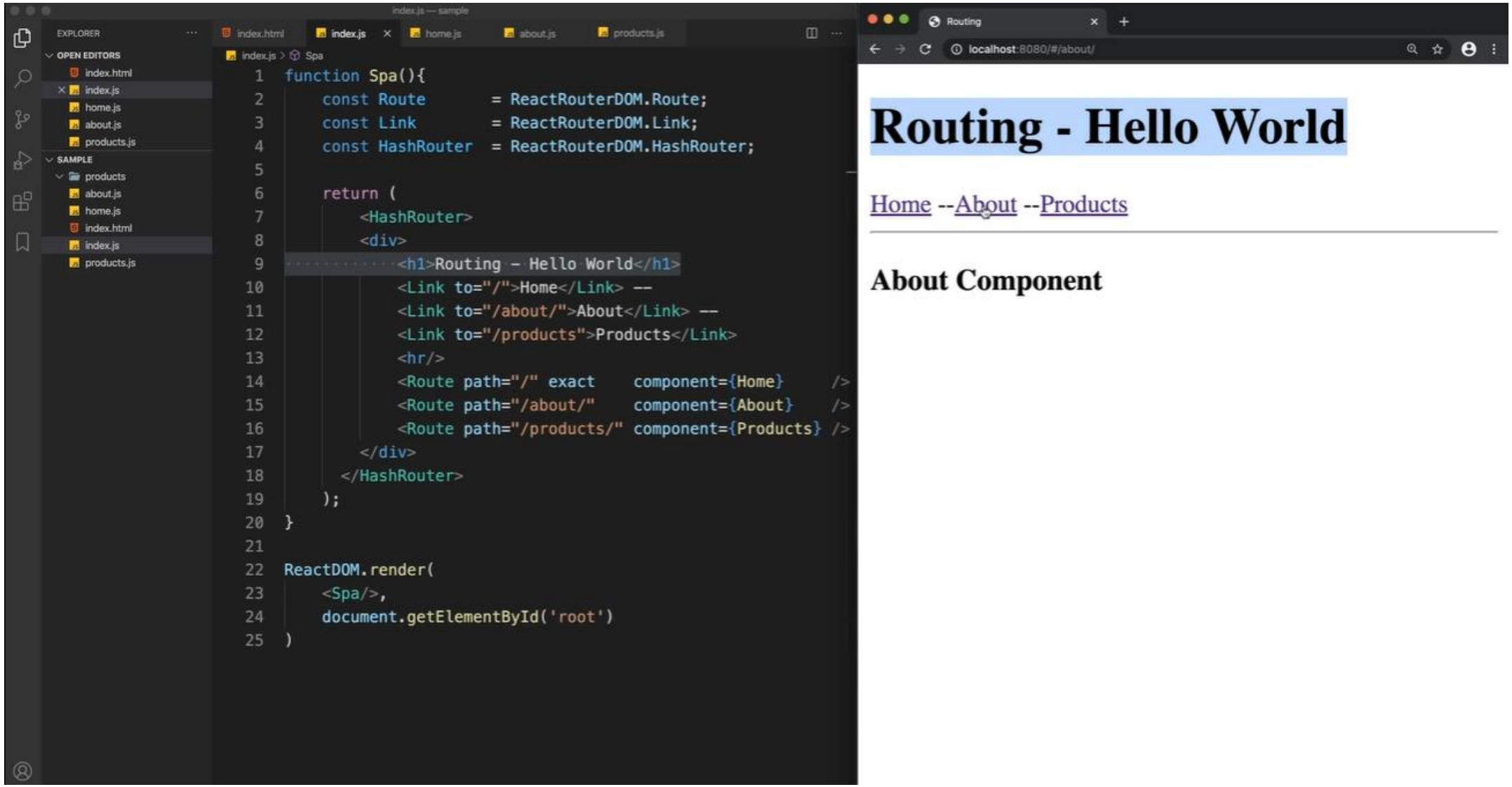
Routing: Demonstration (16/19)



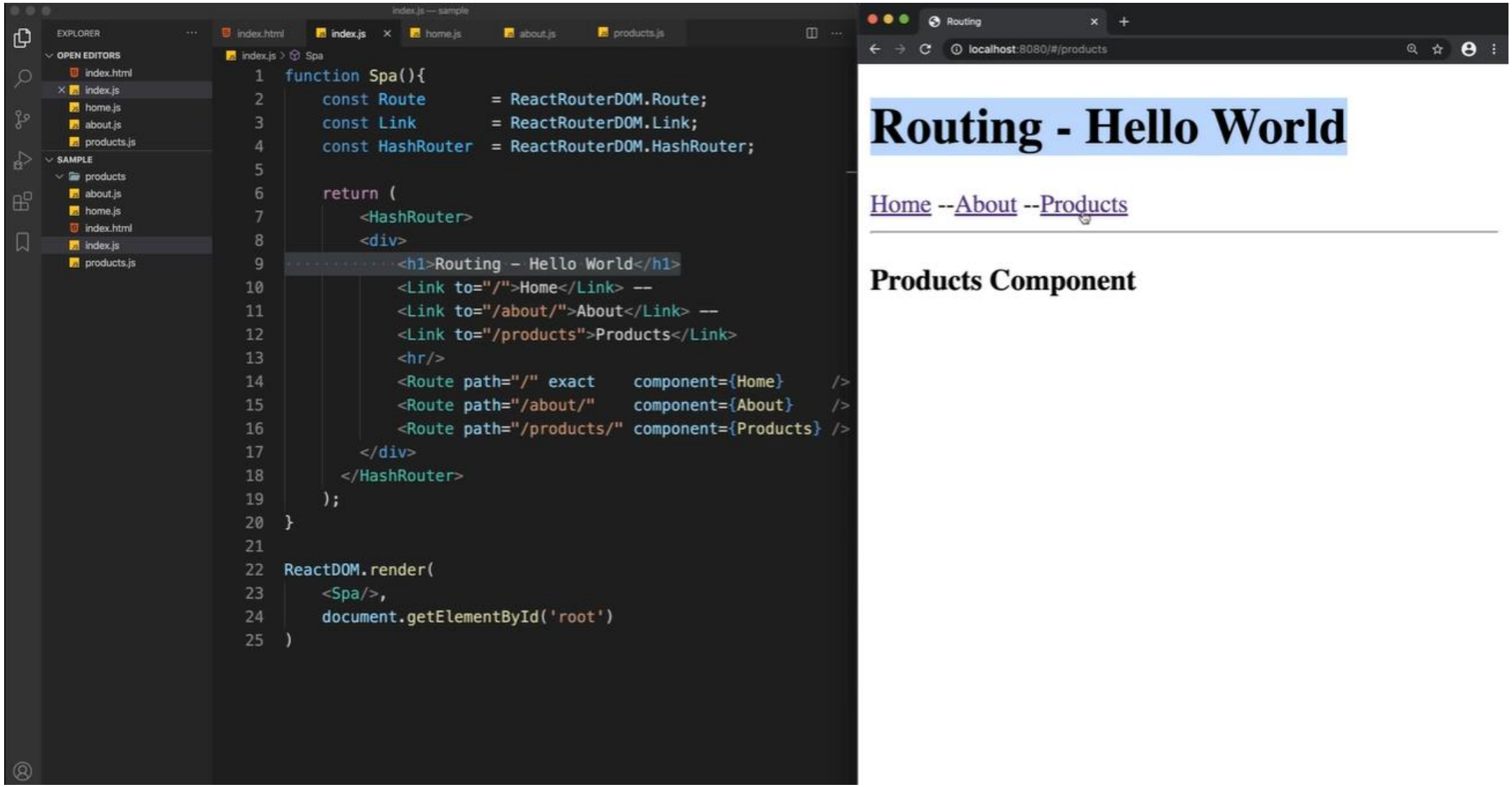
Routing: Demonstration (17/19)



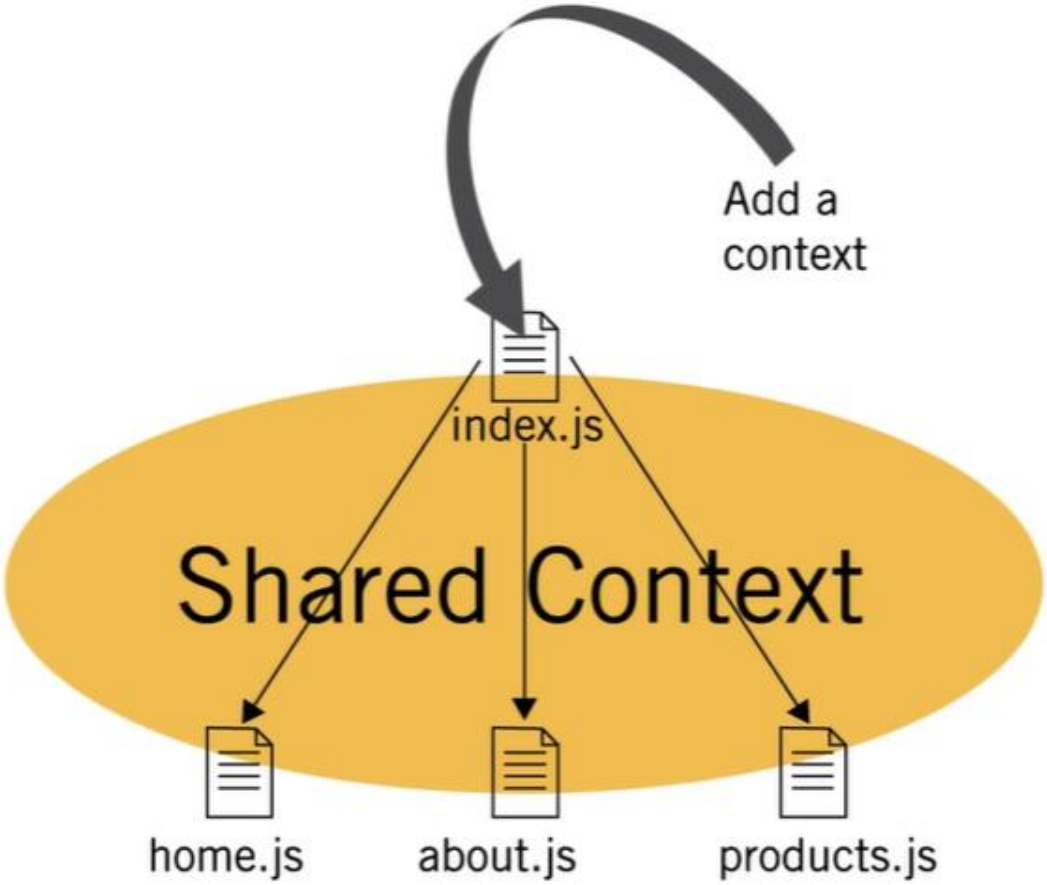
Routing: Demonstration (18/19)



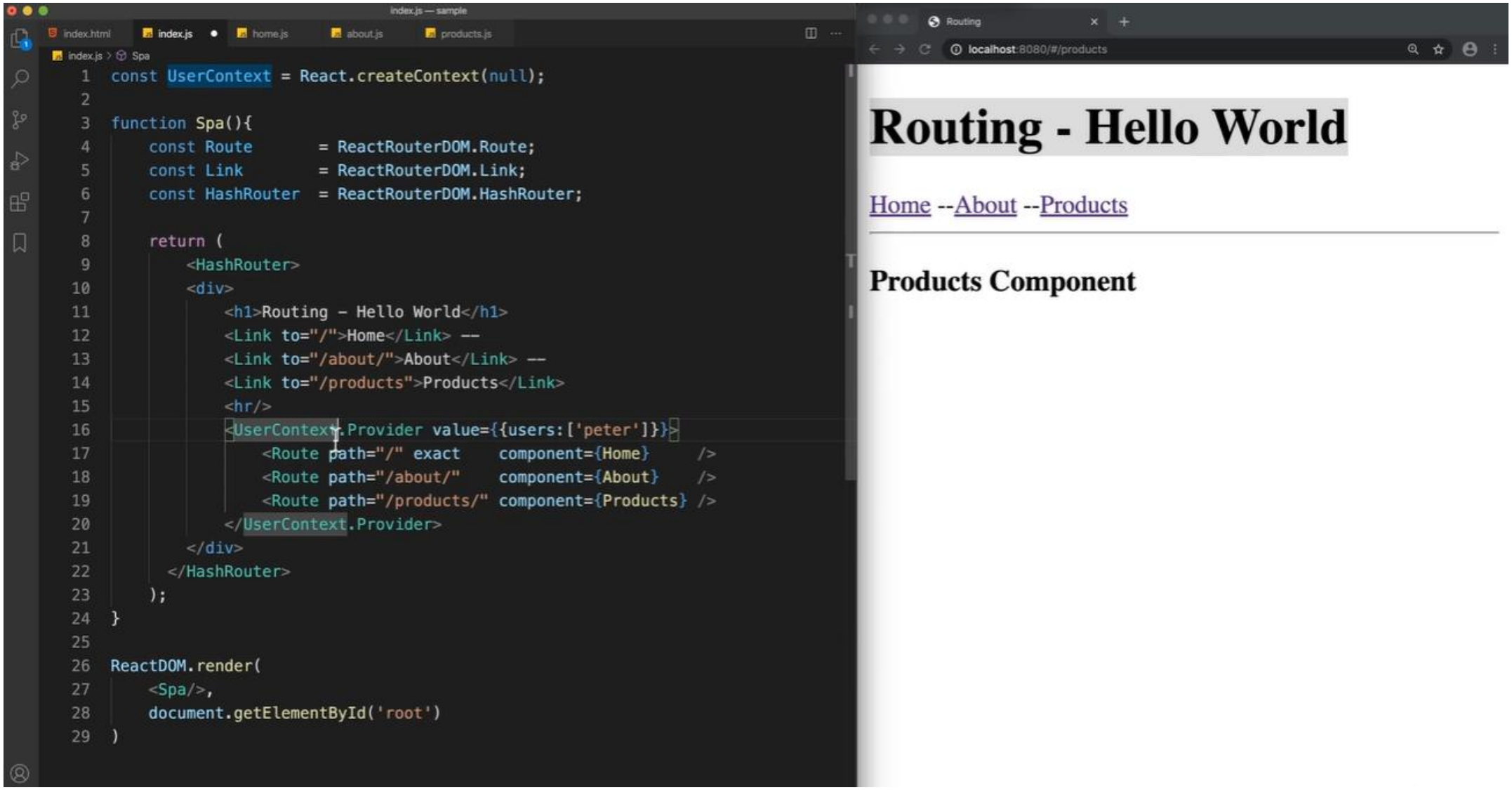
Routing: Demonstration (19/19)



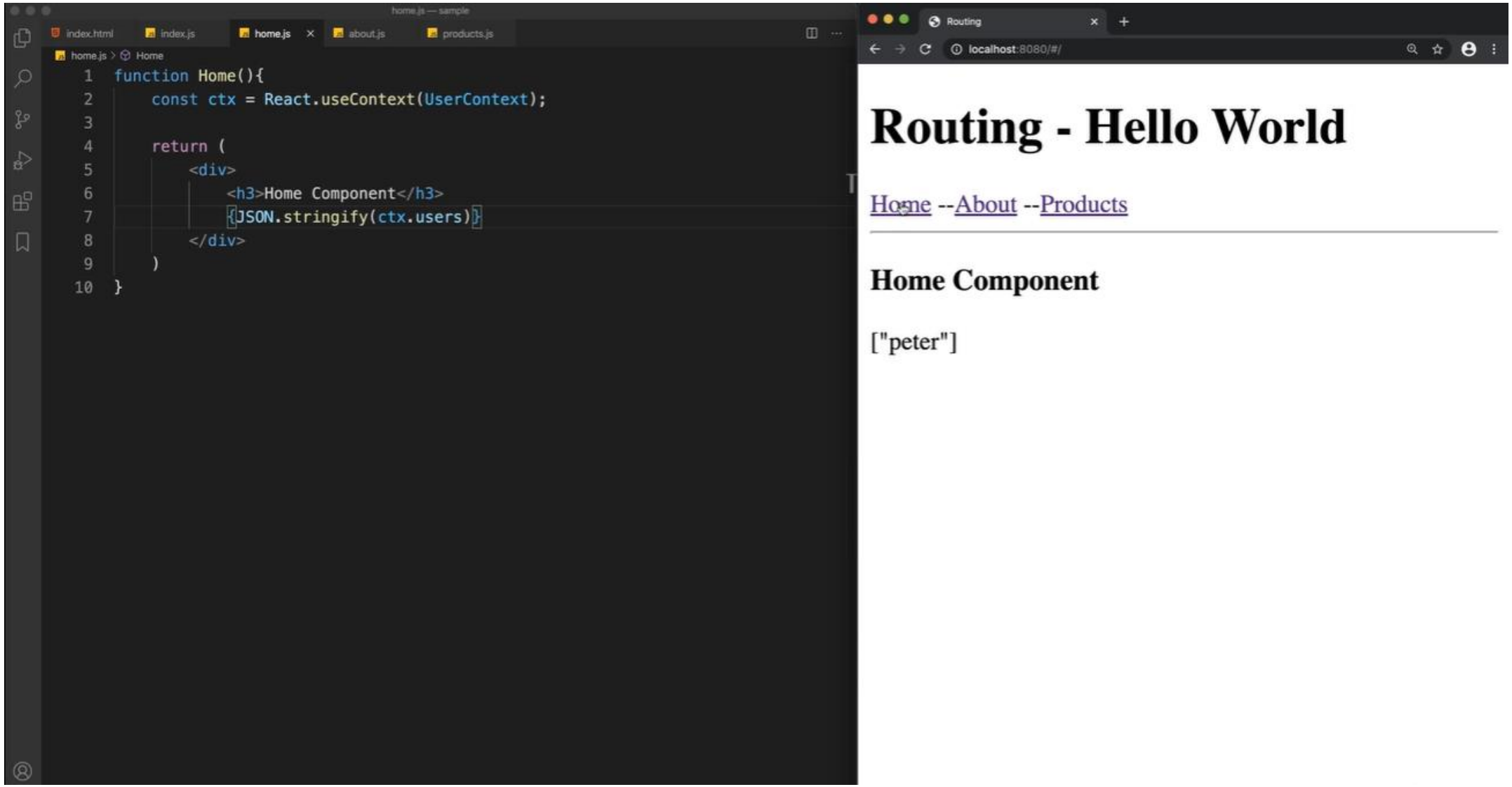
Shared Context



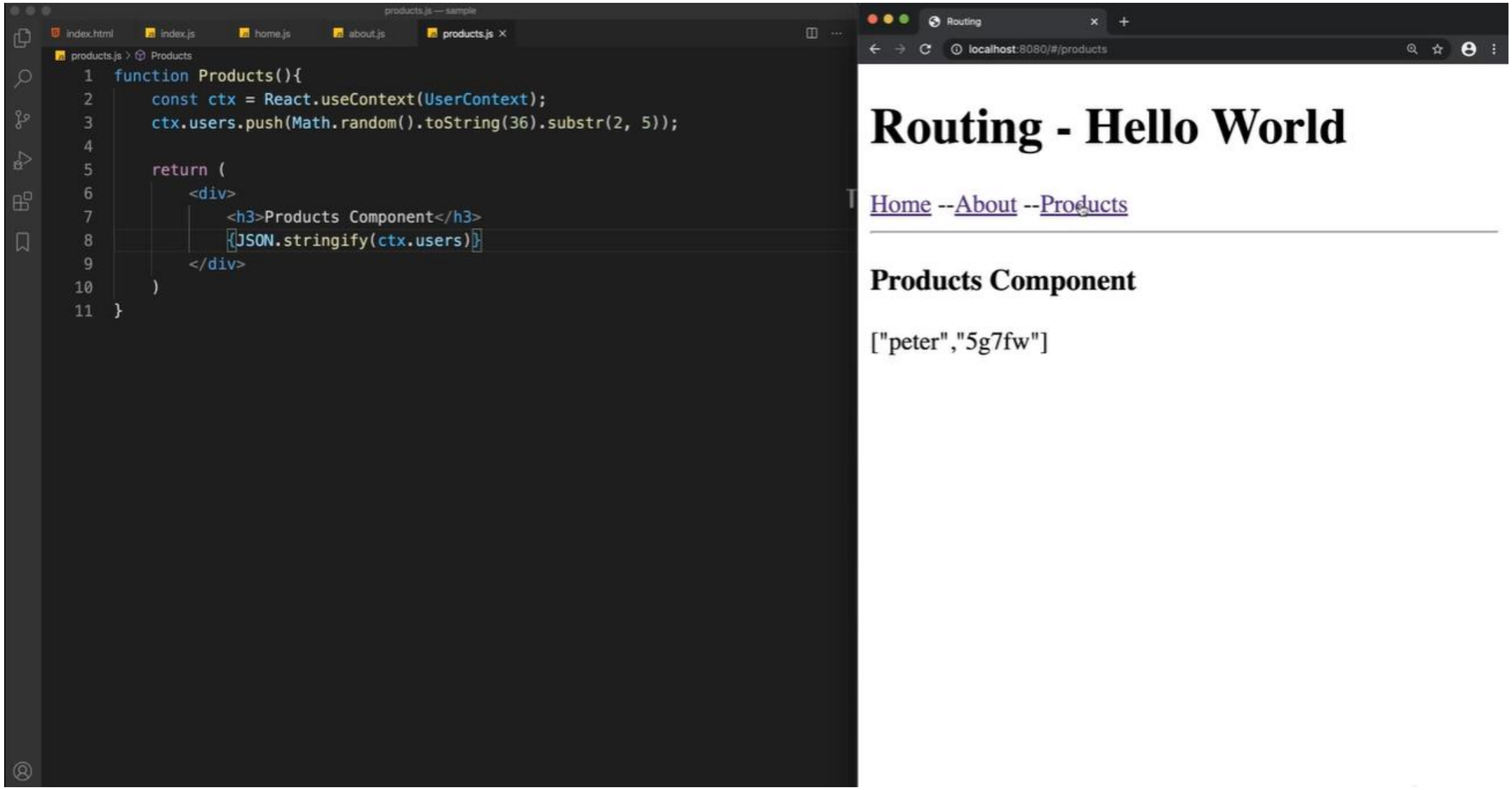
Shared Context: Demonstration (1/6)



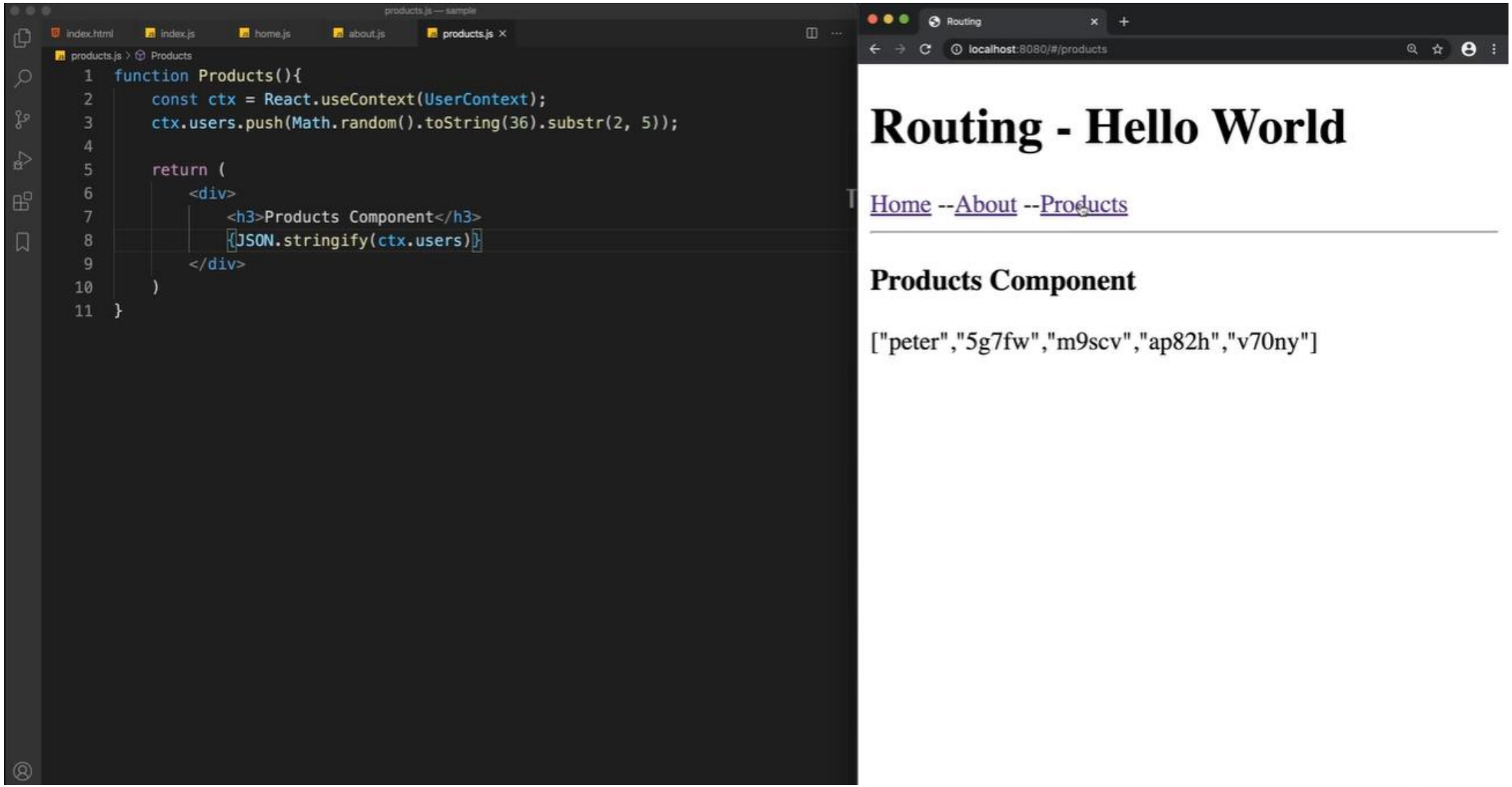
Shared Context: Demonstration (2/6)



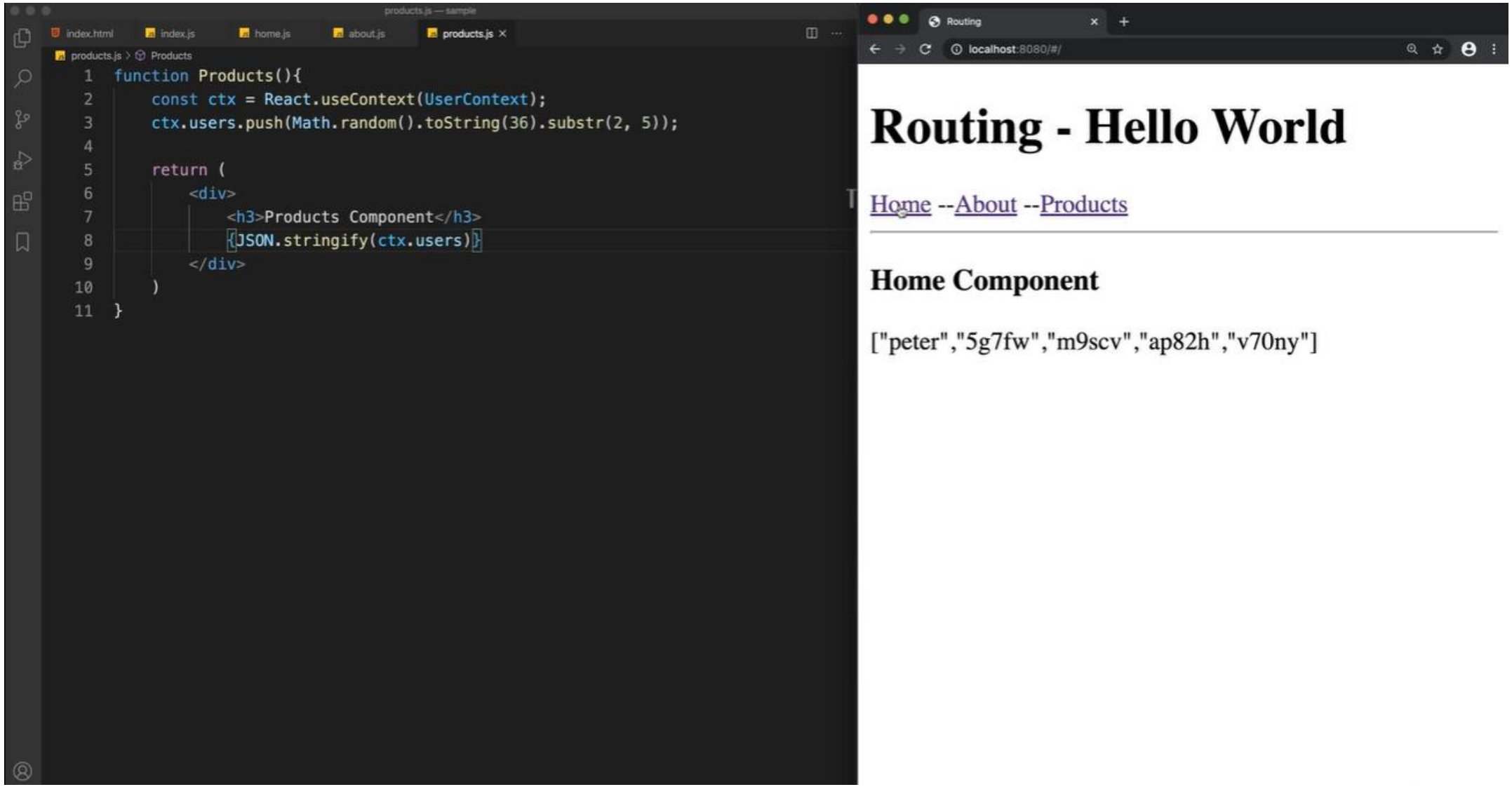
Shared Context: Demonstration (3/6)



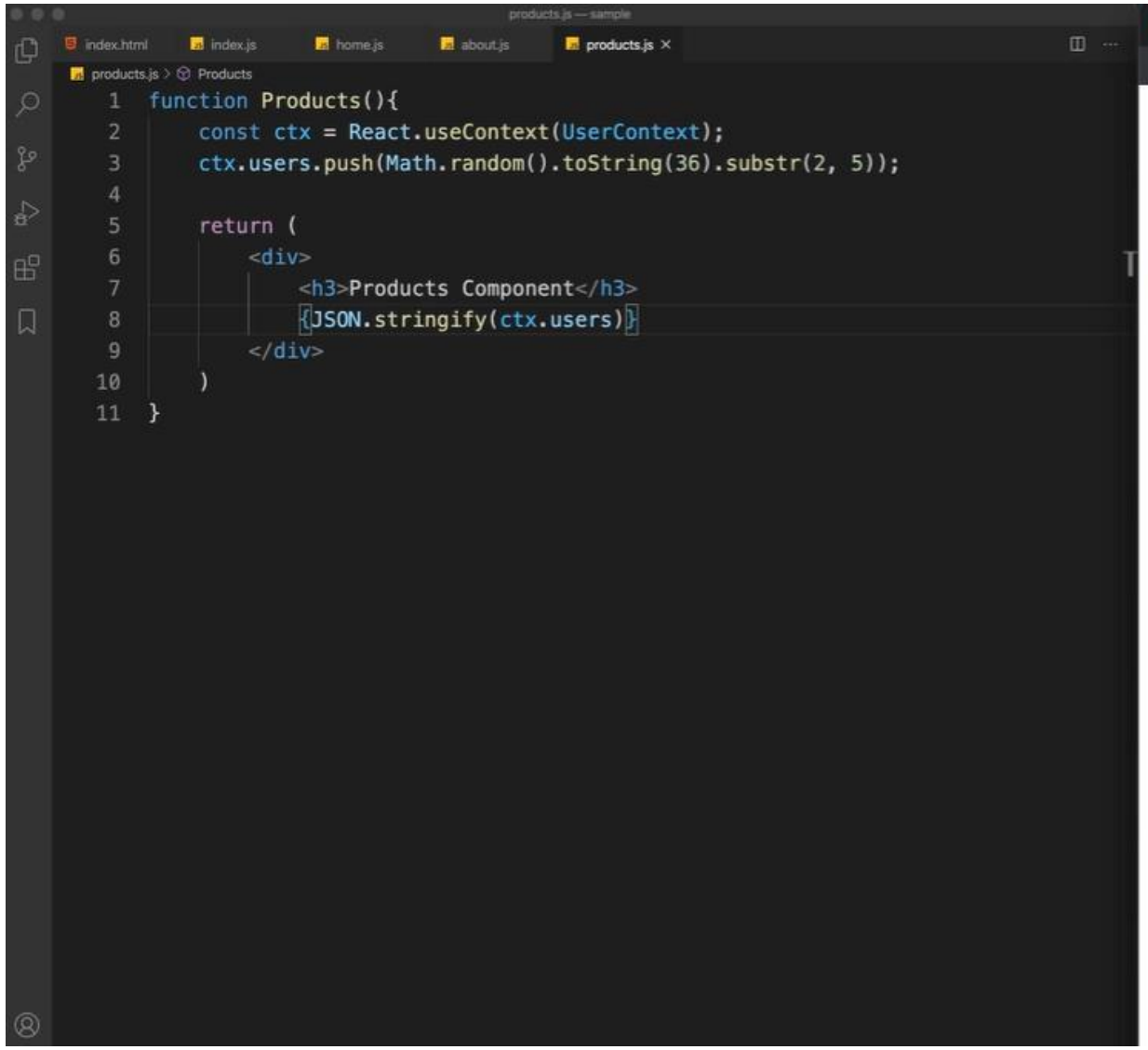
Shared Context: Demonstration (4/6)



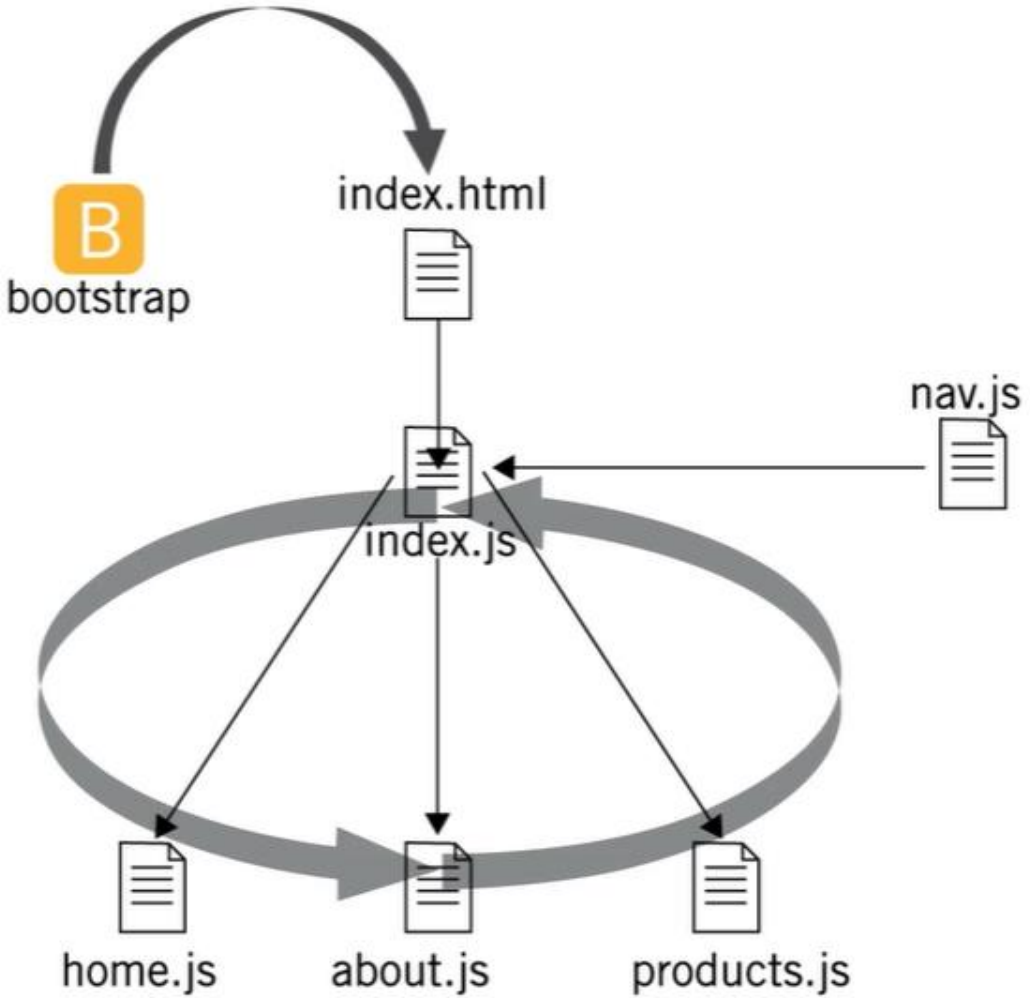
Shared Context: Demonstration (5/6)



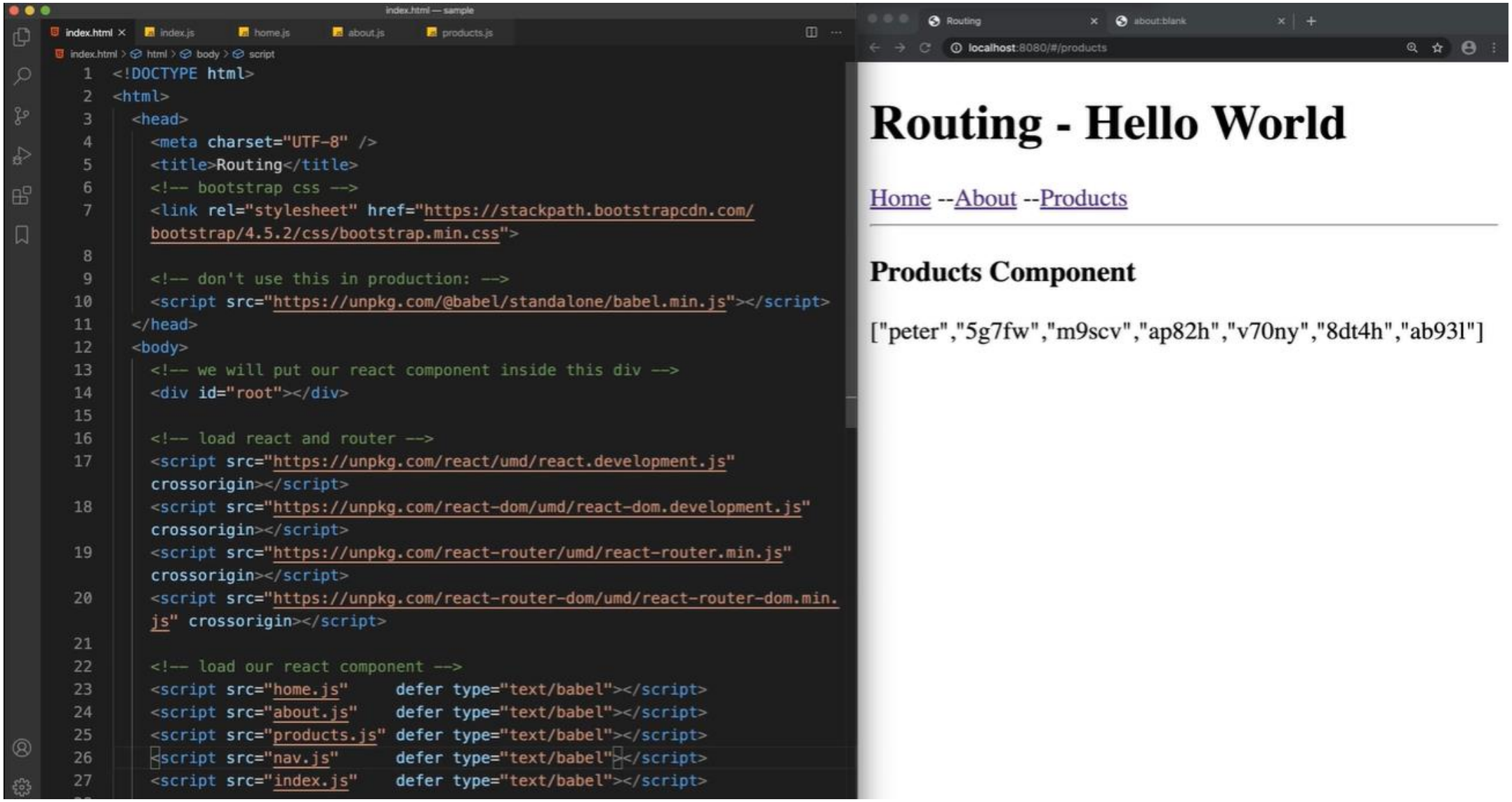
Shared Context: Demonstration (6/6)



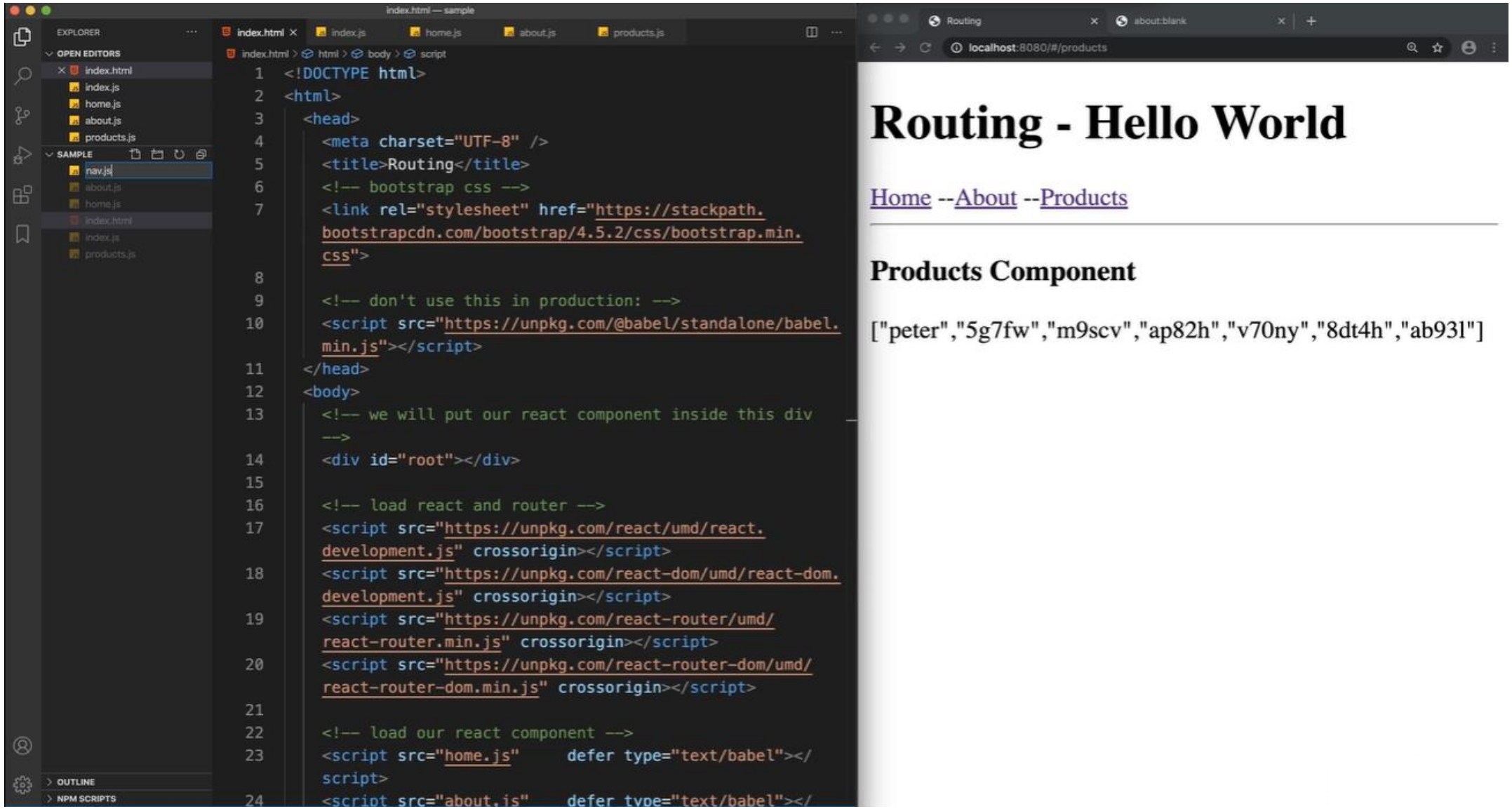
Styles


B
bootstrap
index.html
nav.js
index.js
home.js
about.js
products.js

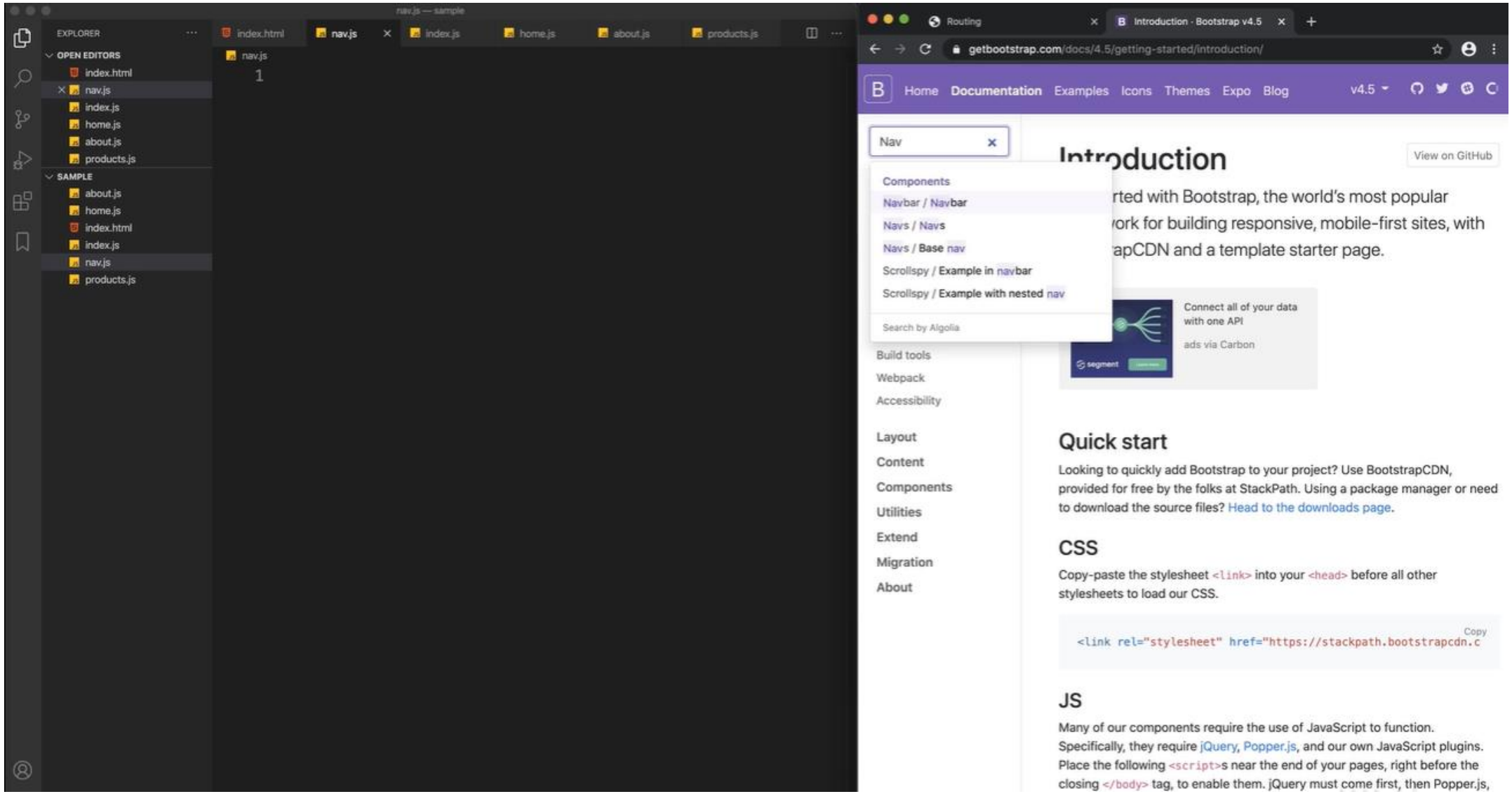
Styles: Demonstration (1/9)



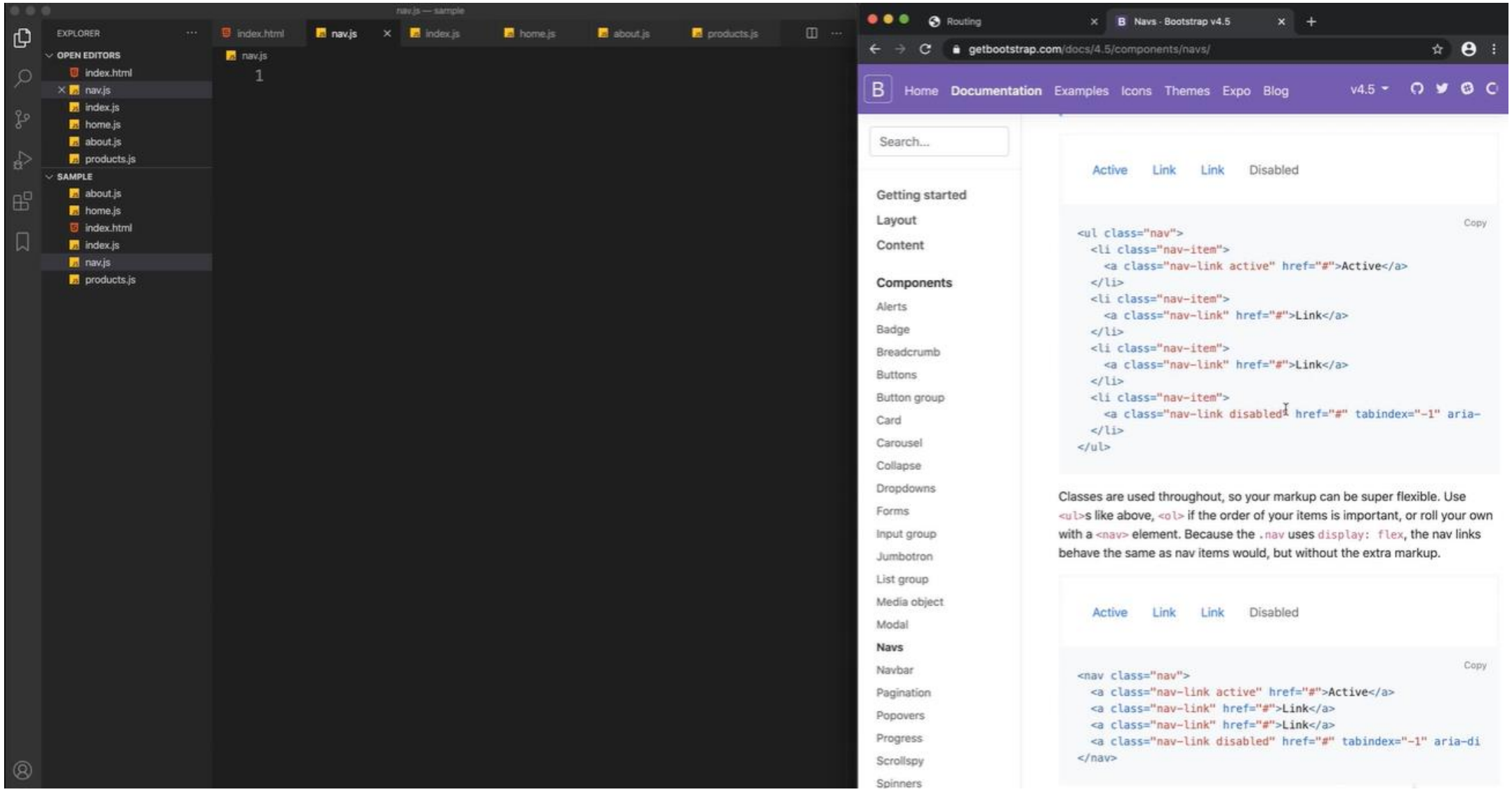
Styles: Demonstration (2/9)



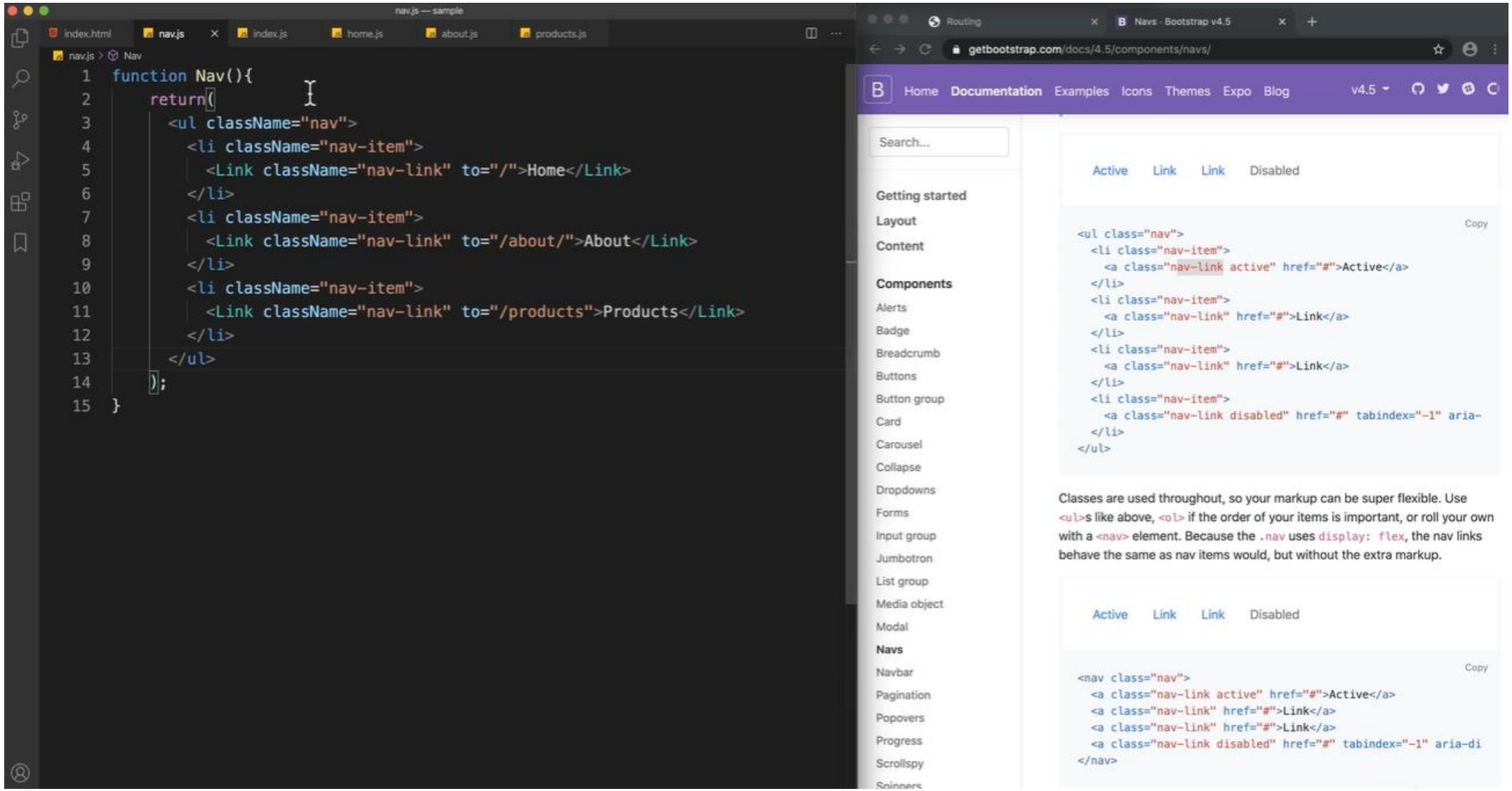
Styles: Demonstration (3/9)



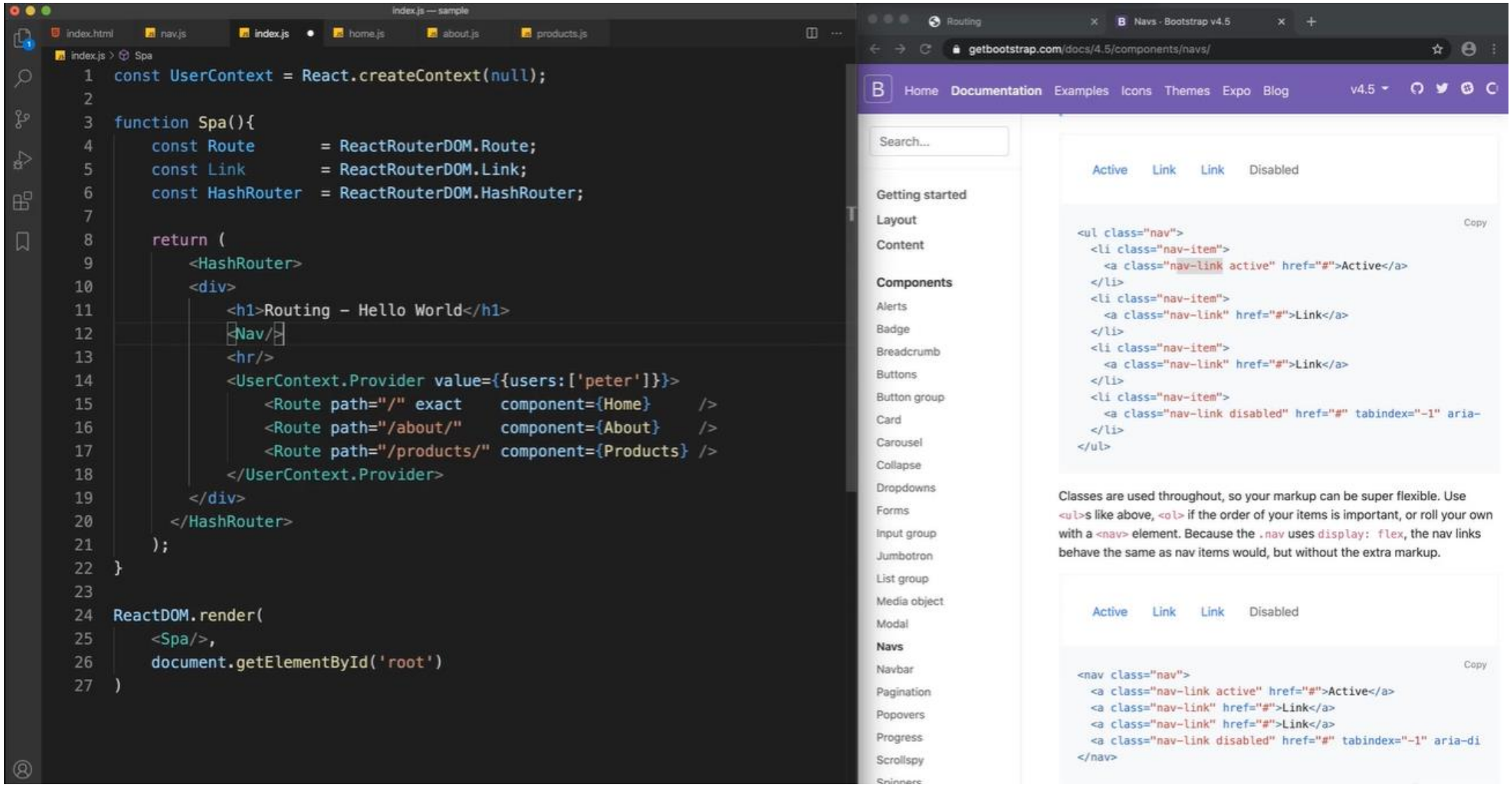
Styles: Demonstration (4/9)



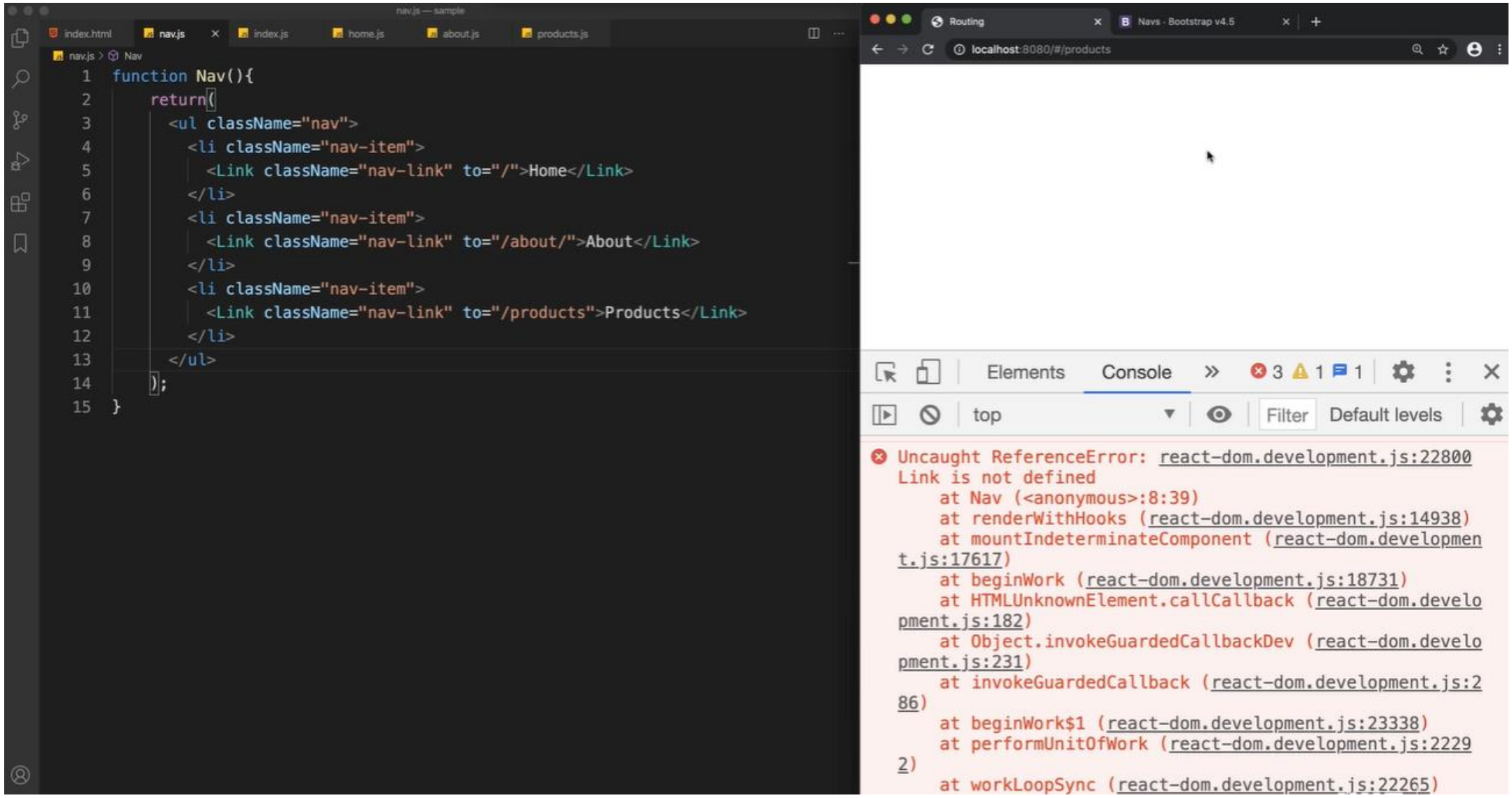
Styles: Demonstration (5/9)



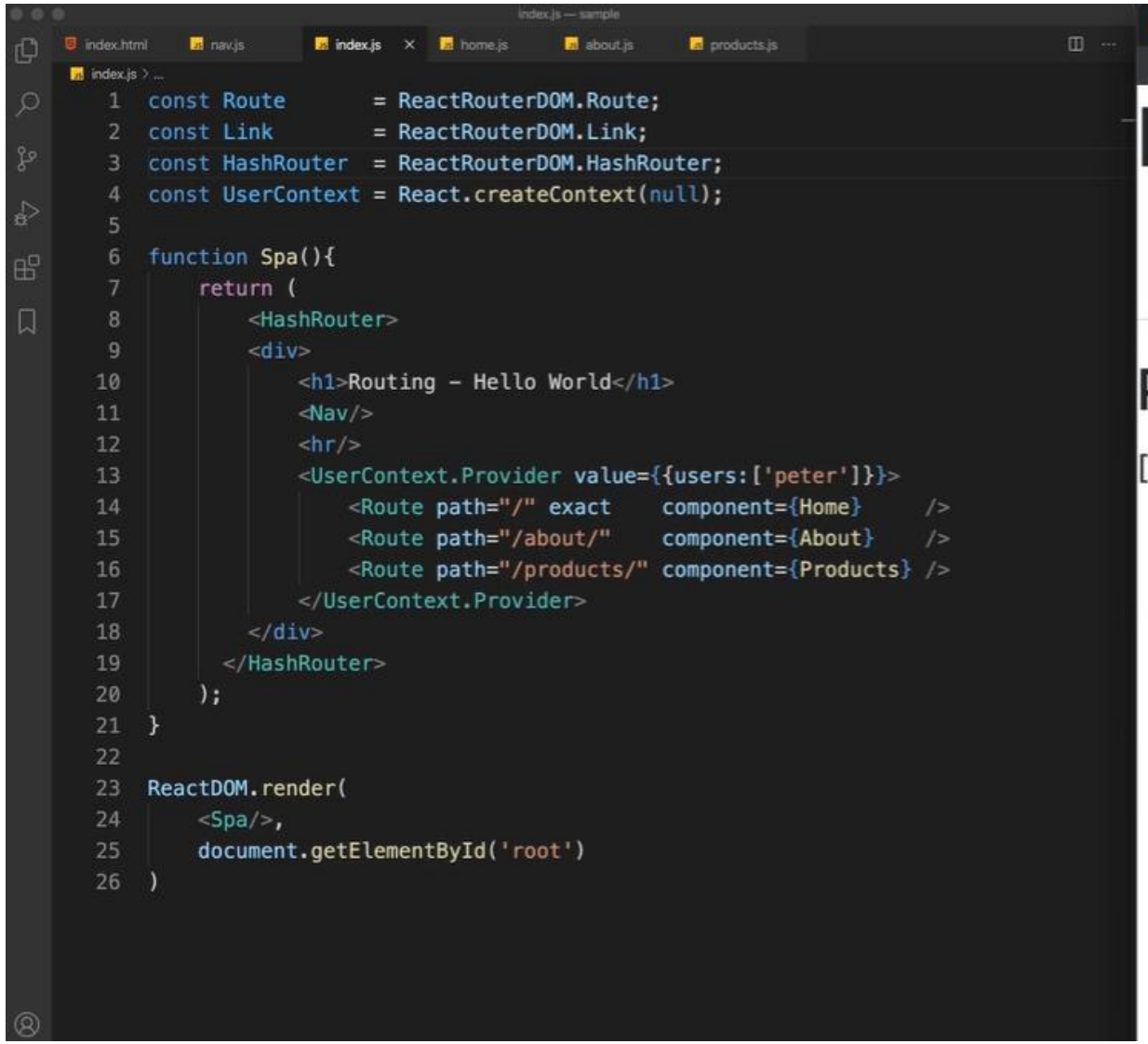
Styles: Demonstration (6/9)

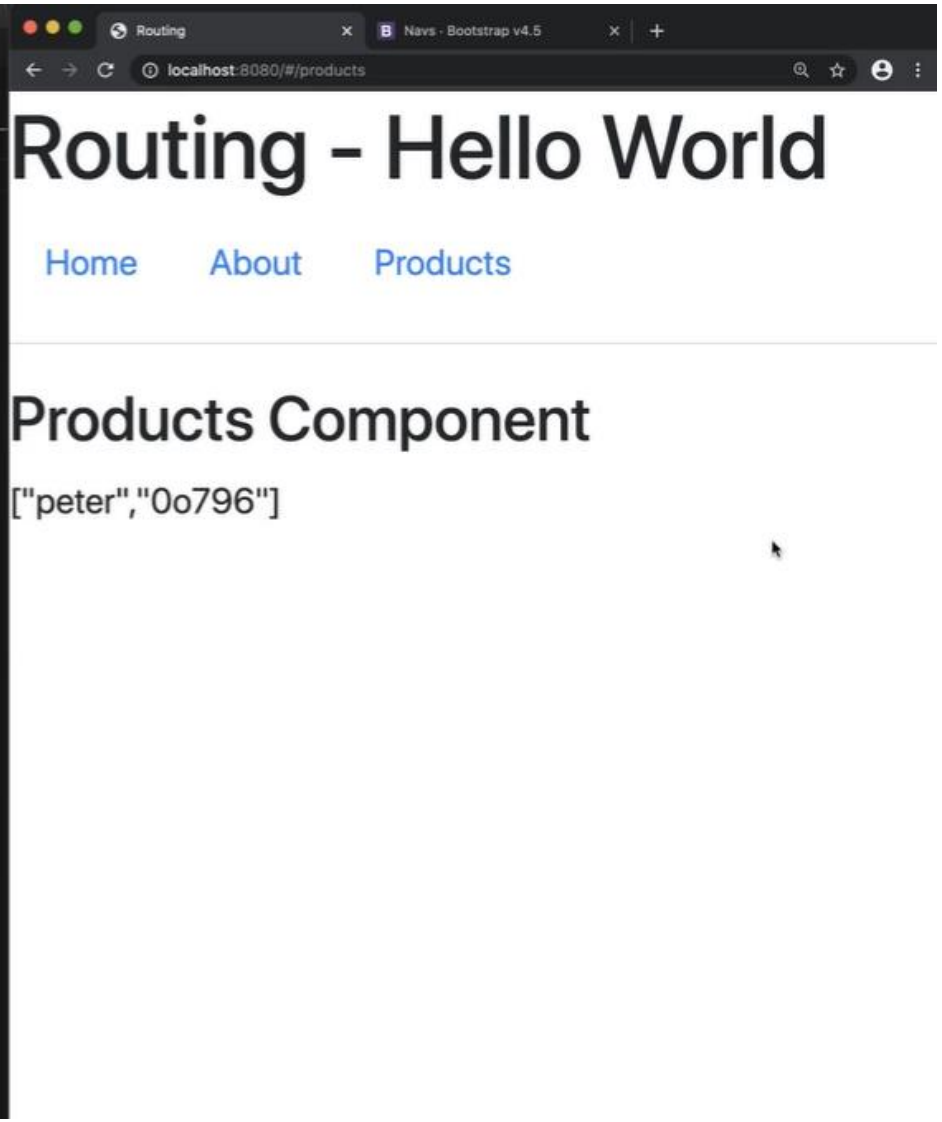


Styles: Demonstration (7/9)

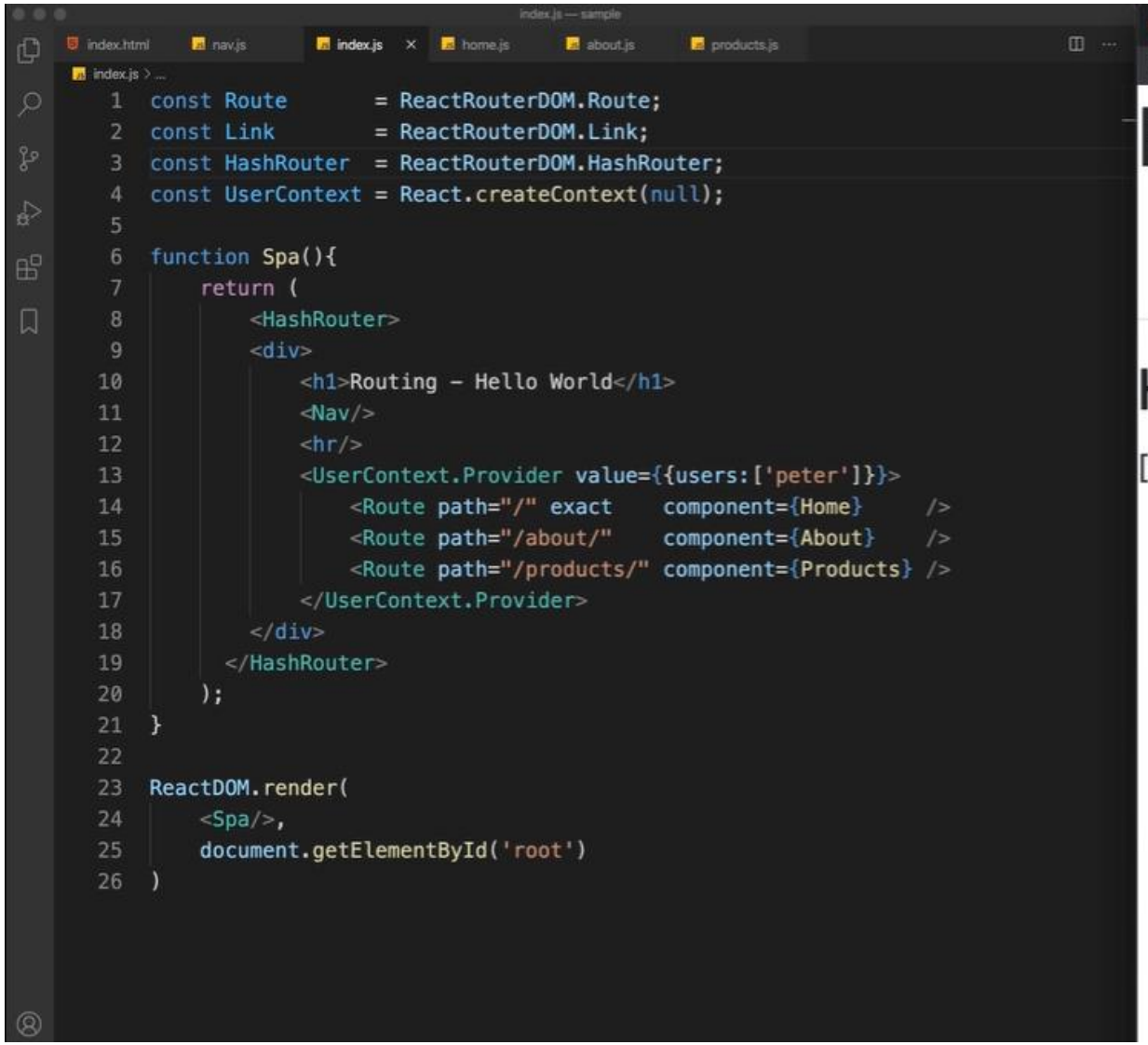


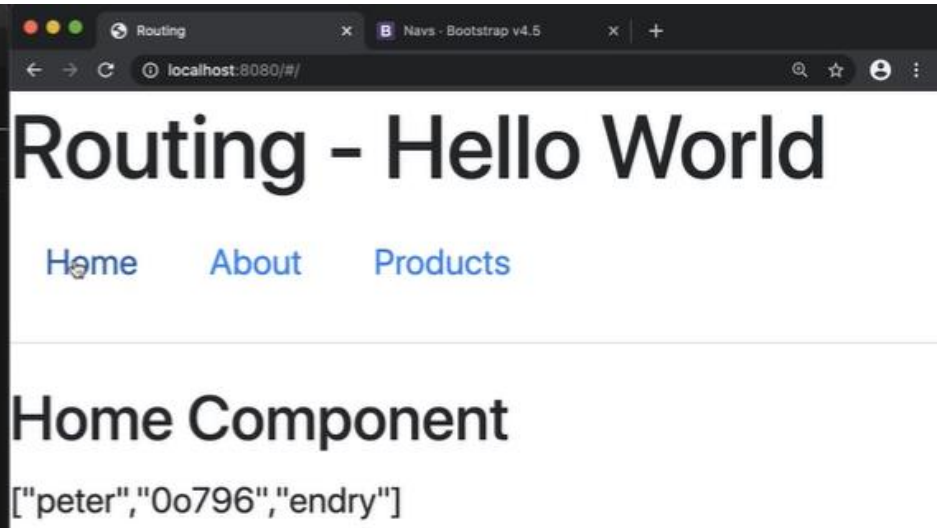
Styles: Demonstration (8/9)





Styles: Demonstration (9/9)

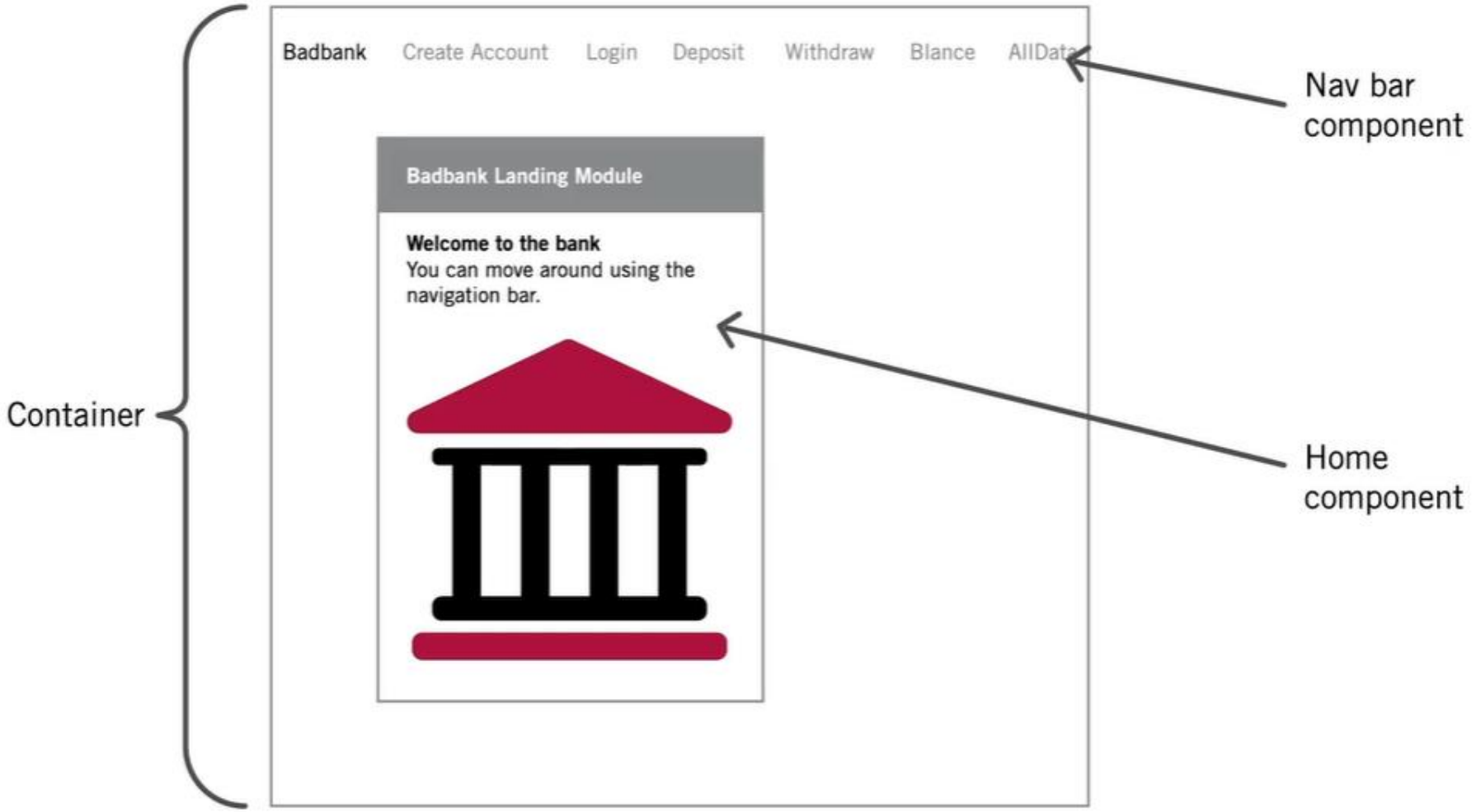




Bad Bank – No Security

Bad Bank

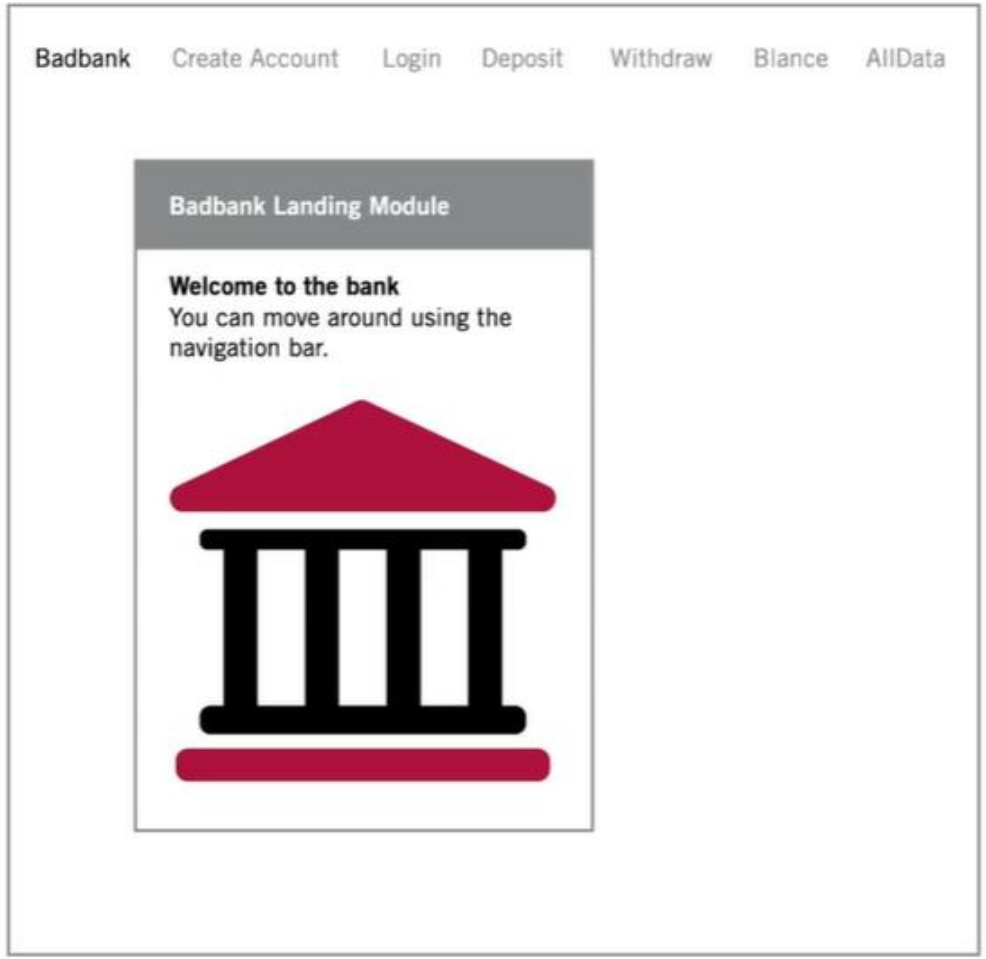
Design


Badbank
Create Account
Login
Deposit
Withdraw
Blance
AllData
Badbank Landing Module
Welcome to the bank
You can move around using the navigation bar.
Container
Nav bar component
Home component

Practice

- Routing
- Context
- Styles
- Components
- Parent-Child components
- Decisions on where to keep state
- Forms
- Handling events

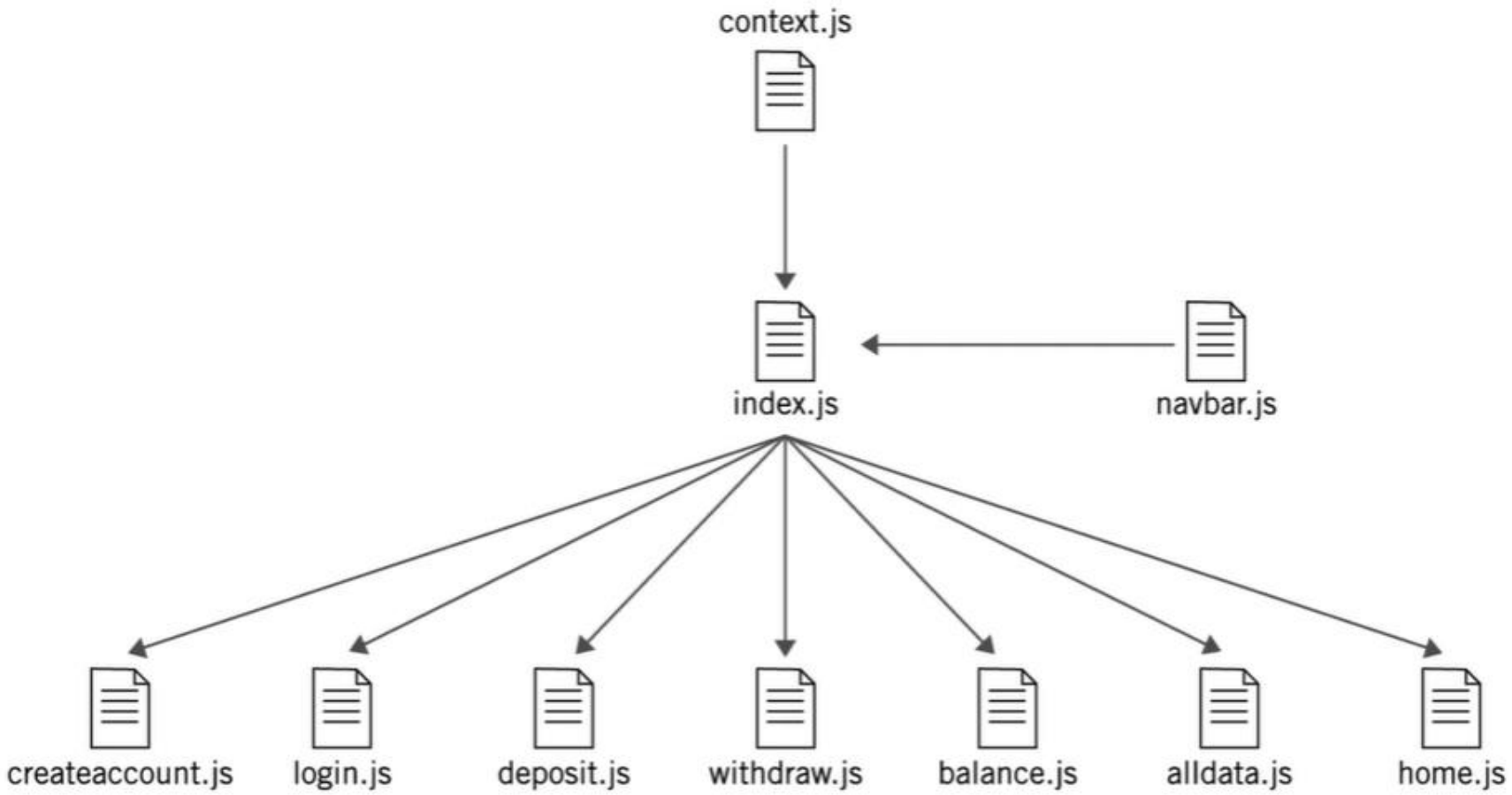
Design – Components Loaded Based On Path



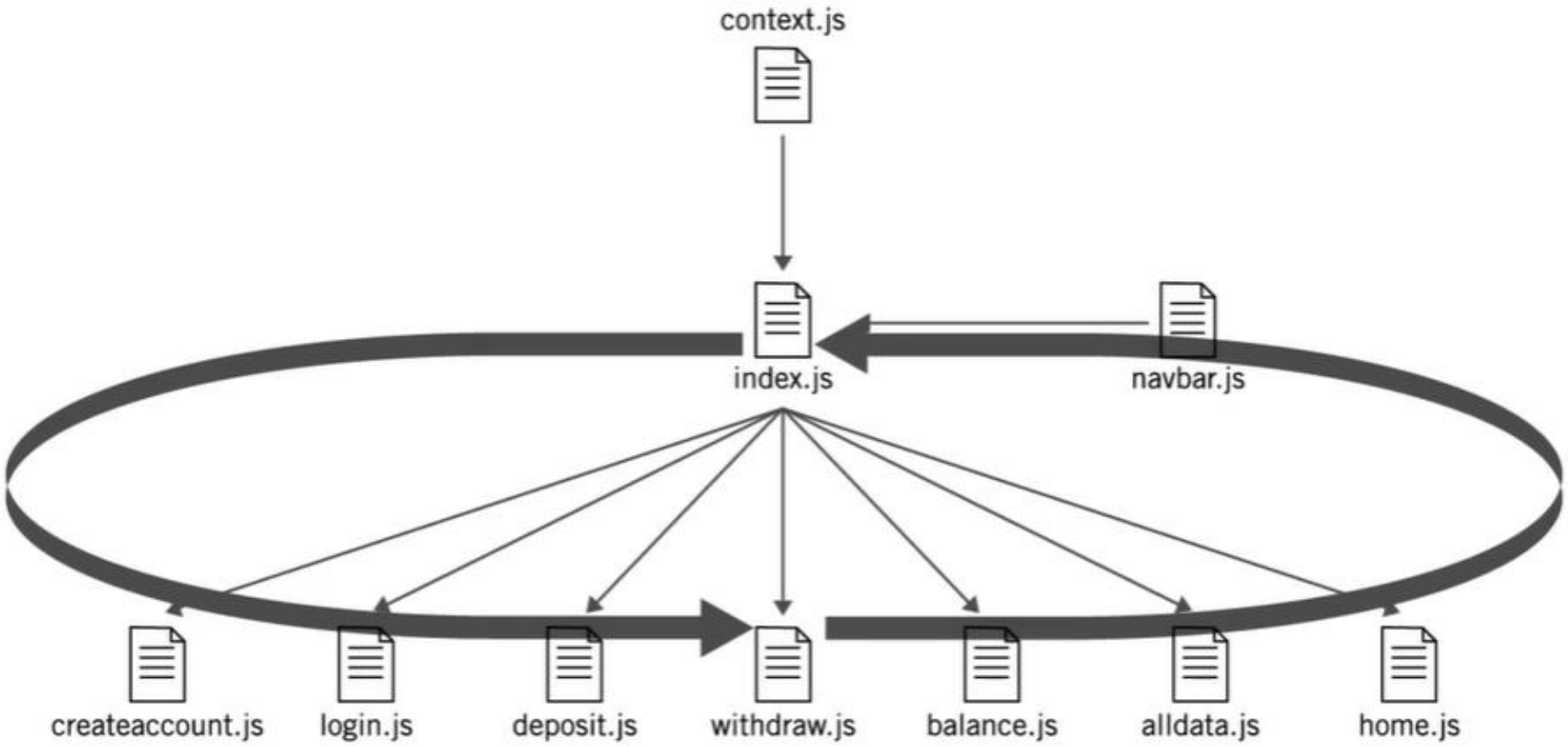
Components:

- Home
- Create account
- Login
- Deposit
- Withdraw
- Balance
- All data

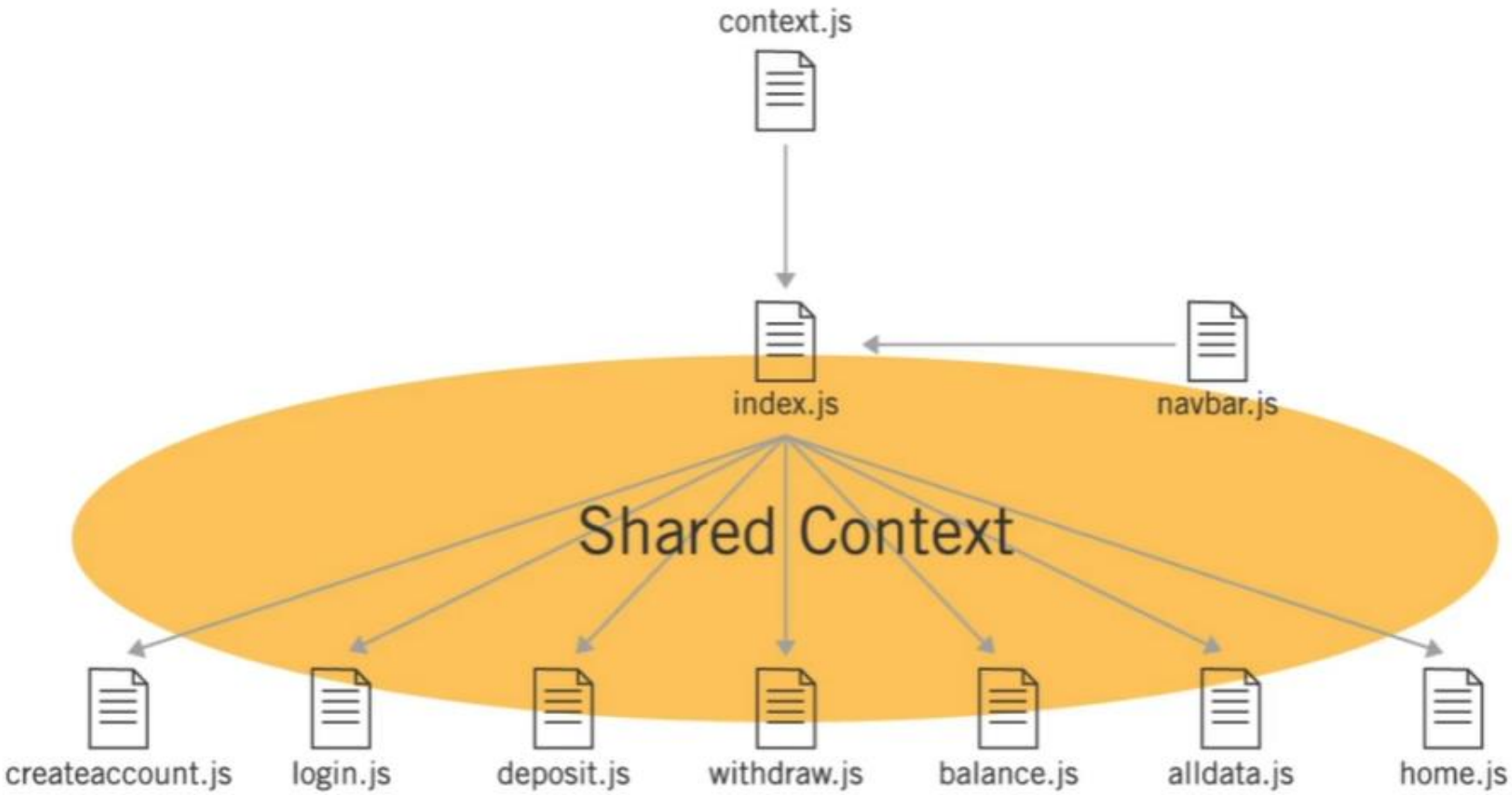
Architecture



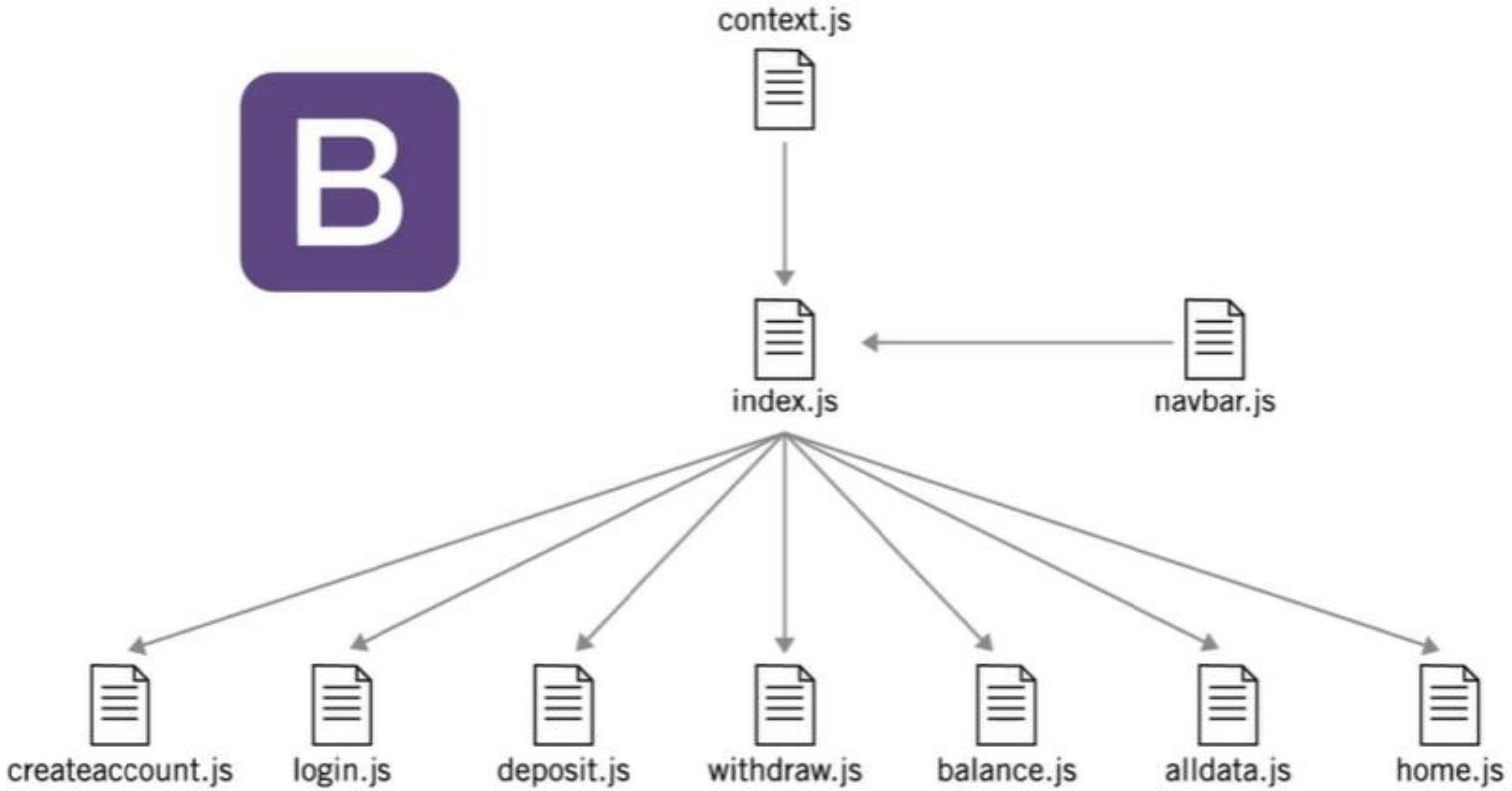
Routing


context.js
index.js
navbar.js
createaccount.js
login.js
deposit.js
withdraw.js
balance.js
alldata.js
home.js

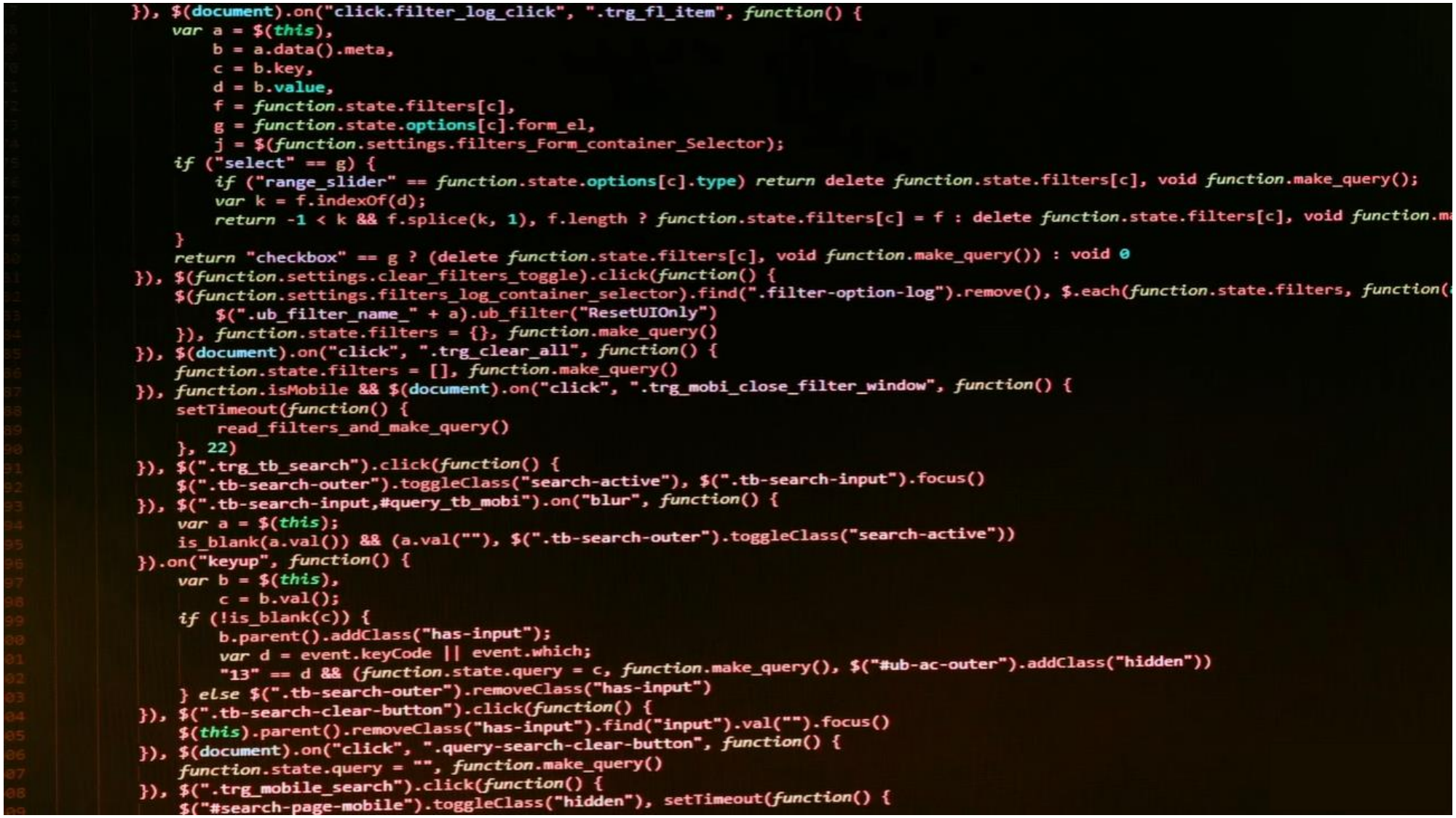
Shared Context



Styles – Bootstrap

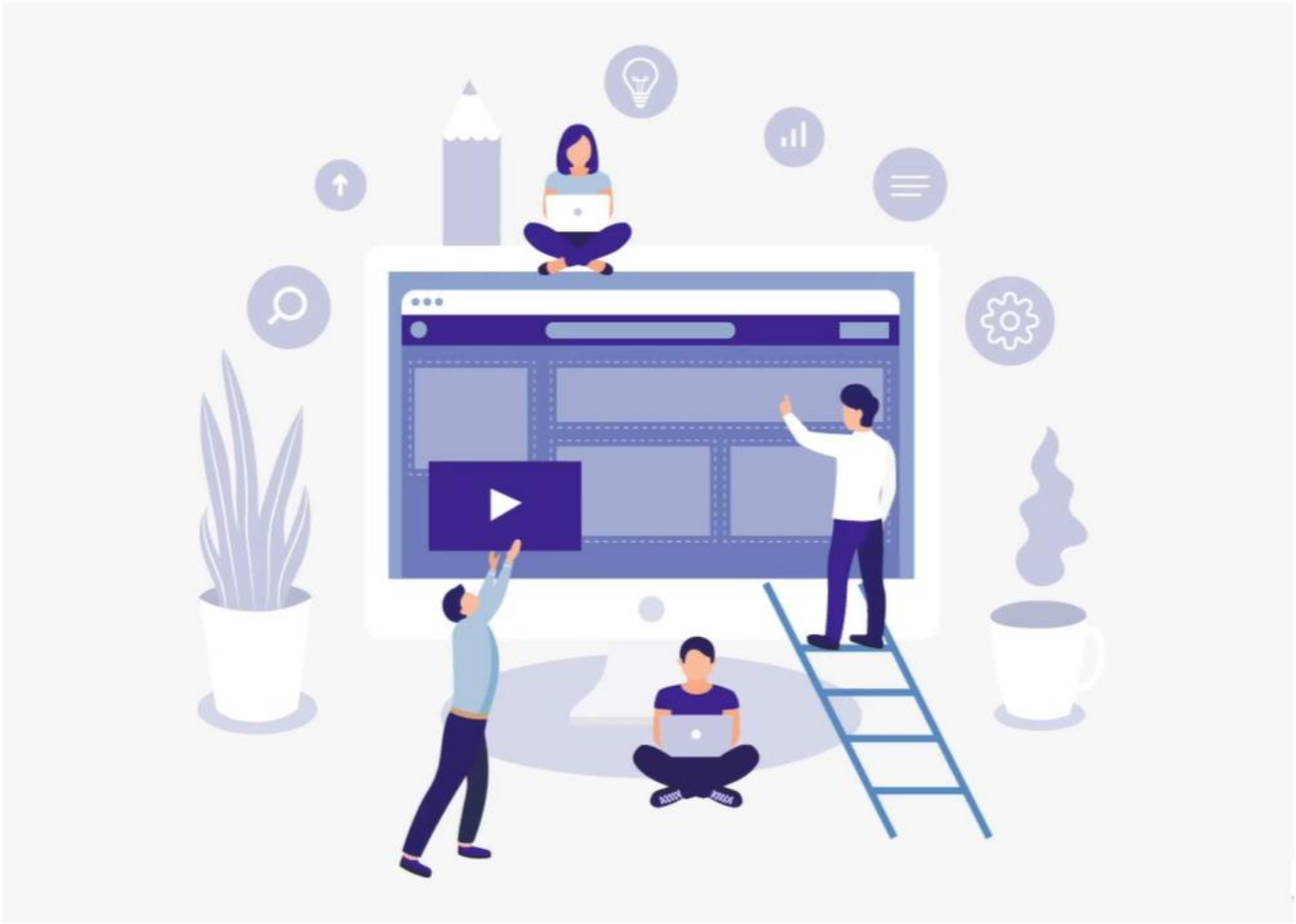


Create Bad Bank Application Files (1/2)

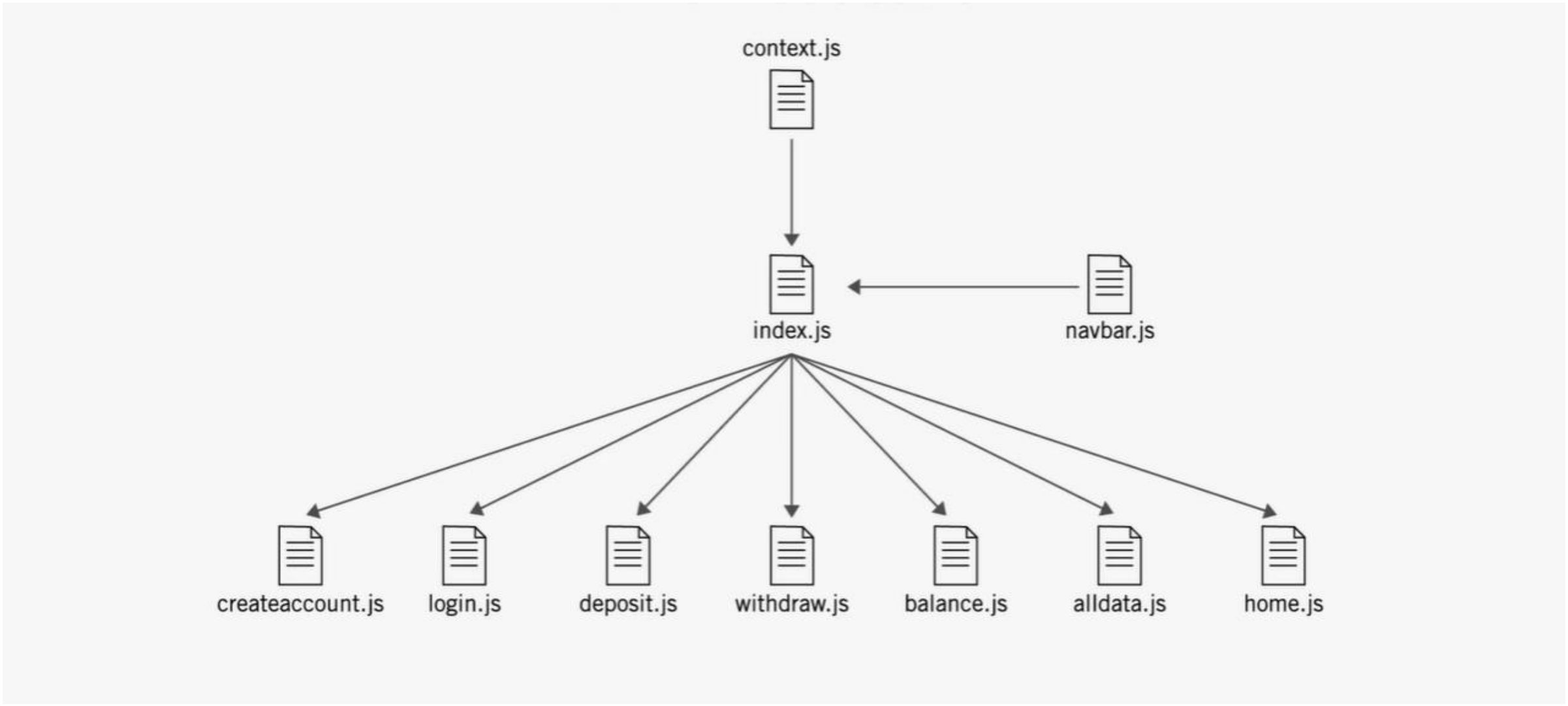


```
}), $(document).on("click.filter_log_click", ".trg_fl_item", function() {
    var a = $(this),
        b = a.data().meta,
        c = b.key,
        d = b.value,
        f = function.state.filters[c],
        g = function.state.options[c].form_el,
        j = $(function.settings.filters_form_container_Selector);
    if ("select" == g) {
        if ("range_slider" == function.state.options[c].type) return delete function.state.filters[c], void function.make_query();
        var k = f.indexOf(d);
        return -1 < k && f.splice(k, 1), f.length ? function.state.filters[c] = f : delete function.state.filters[c], void function.ma
    }
    return "checkbox" == g ? (delete function.state.filters[c], void function.make_query()) : void 0
}), $(function.settings.clear_filters_toggle).click(function() {
    $(function.settings.filters_log_container_selector).find(".filter-option-log").remove(), $.each(function.state.filters, function(
        $(".ub_filter_name_" + a).ub_filter("ResetUIOnly")
    }), function.state.filters = {}, function.make_query()
}), $(document).on("click", ".trg_clear_all", function() {
    function.state.filters = [], function.make_query()
}), function.isMobile && $(document).on("click", ".trg_mobi_close_filter_window", function() {
    setTimeout(function() {
        read_filters_and_make_query()
    }, 22)
}), $(".trg_tb_search").click(function() {
    $(".tb-search-outer").toggleClass("search-active"), $(".tb-search-input").focus()
}), $(".tb-search-input,#query_tb_mobi").on("blur", function() {
    var a = $(this);
    is_blank(a.val()) && (a.val(""), $(".tb-search-outer").toggleClass("search-active"))
}).on("keyup", function() {
    var b = $(this),
        c = b.val();
    if (!is_blank(c)) {
        b.parent().addClass("has-input");
        var d = event.keyCode || event.which;
        "13" == d && (function.state.query = c, function.make_query(), $("#ub-ac-outer").addClass("hidden"))
    } else $(".tb-search-outer").removeClass("has-input")
}), $(".tb-search-clear-button").click(function() {
    $(this).parent().removeClass("has-input").find("input").val("").focus()
}), $(document).on("click", ".query-search-clear-button", function() {
    function.state.query = "", function.make_query()
}), $(".trg_mobile_search").click(function() {
    $("#search-page-mobile").toggleClass("hidden"), setTimeout(function() {
```

Create Bad Bank Application Files (2/2)



Architecture

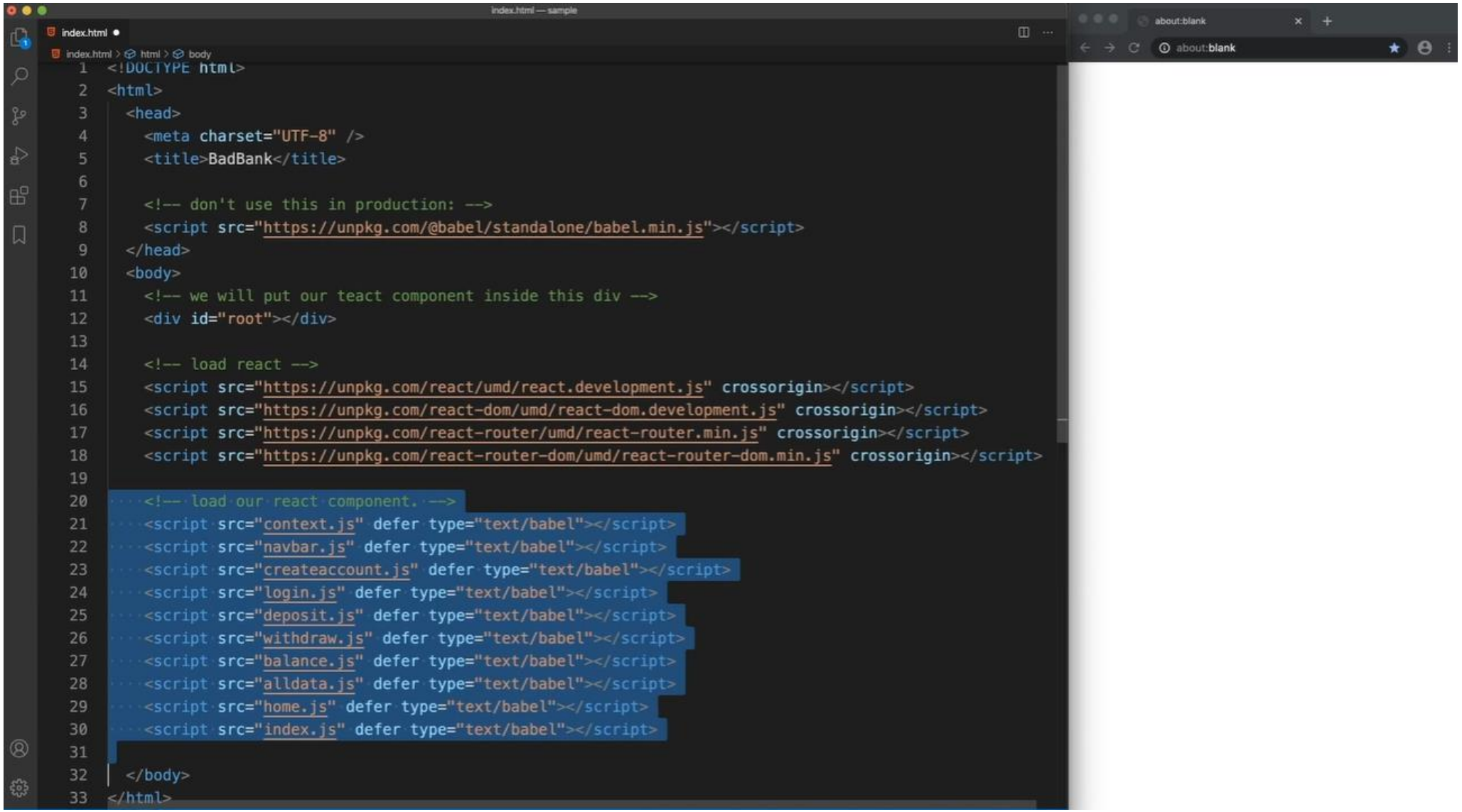


File Loading – Done In index.html

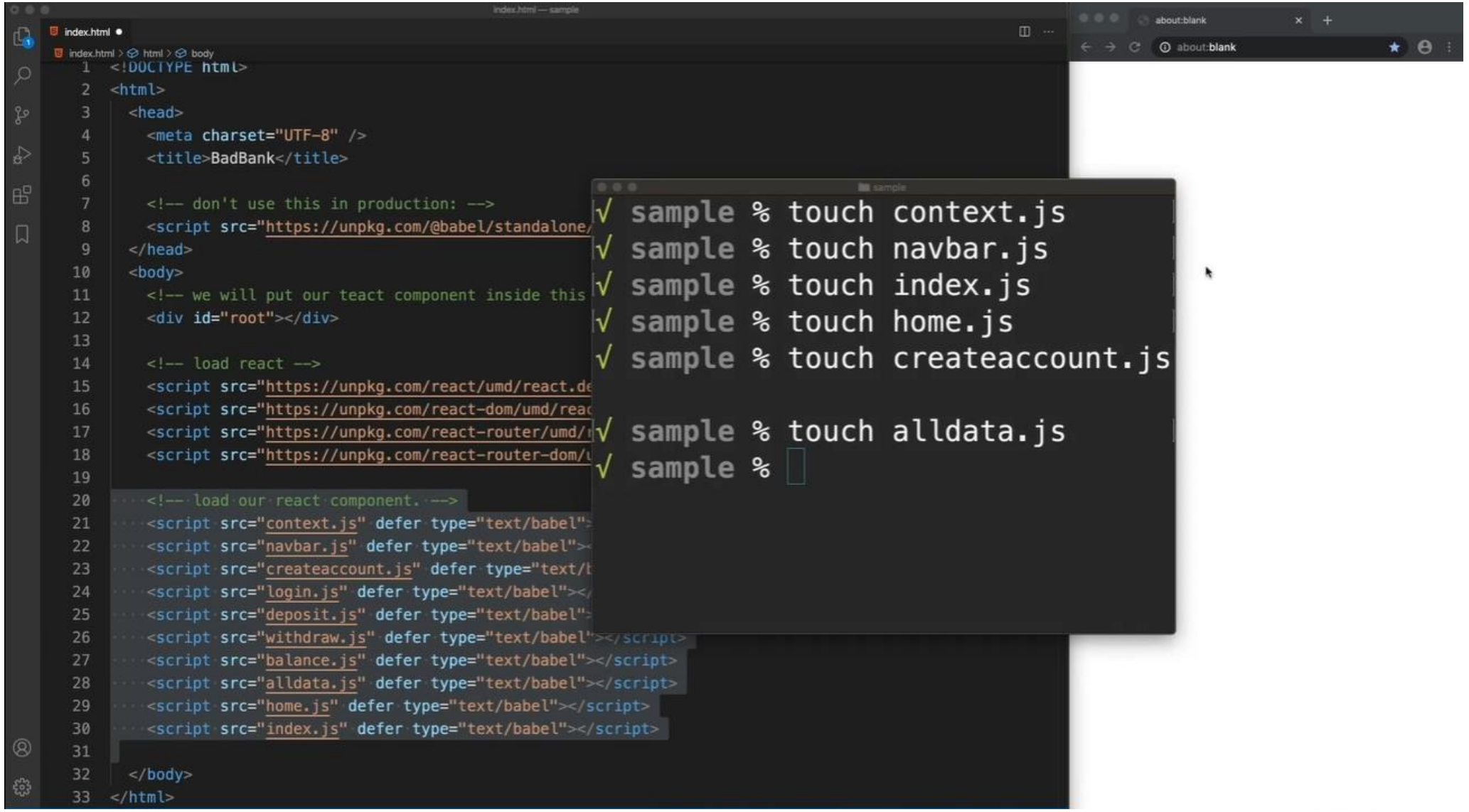
components loaded in index.html

```
<script src = "context.js" defer type = "text/label" ></script>
<script src = "navbar.js" defer type = "text/label" ></script>
<script src = "createaccount.js" defer type = "text/label" ></script>
<script src = "login.js" defer type = "text/label" ></script>
<script src = "deposit.js" defer type = "text/label" ></script>
<script src = "withdraw.js" defer type = "text/label" ></script>
<script src = "balance.js" defer type = "text/label" ></script>
<script src = "alldata.js" defer type = "text/label" ></script>
<script src = "home.js" defer type = "text/label" ></script>
<script src = "index.js" defer type = "text/label" ></script>
```

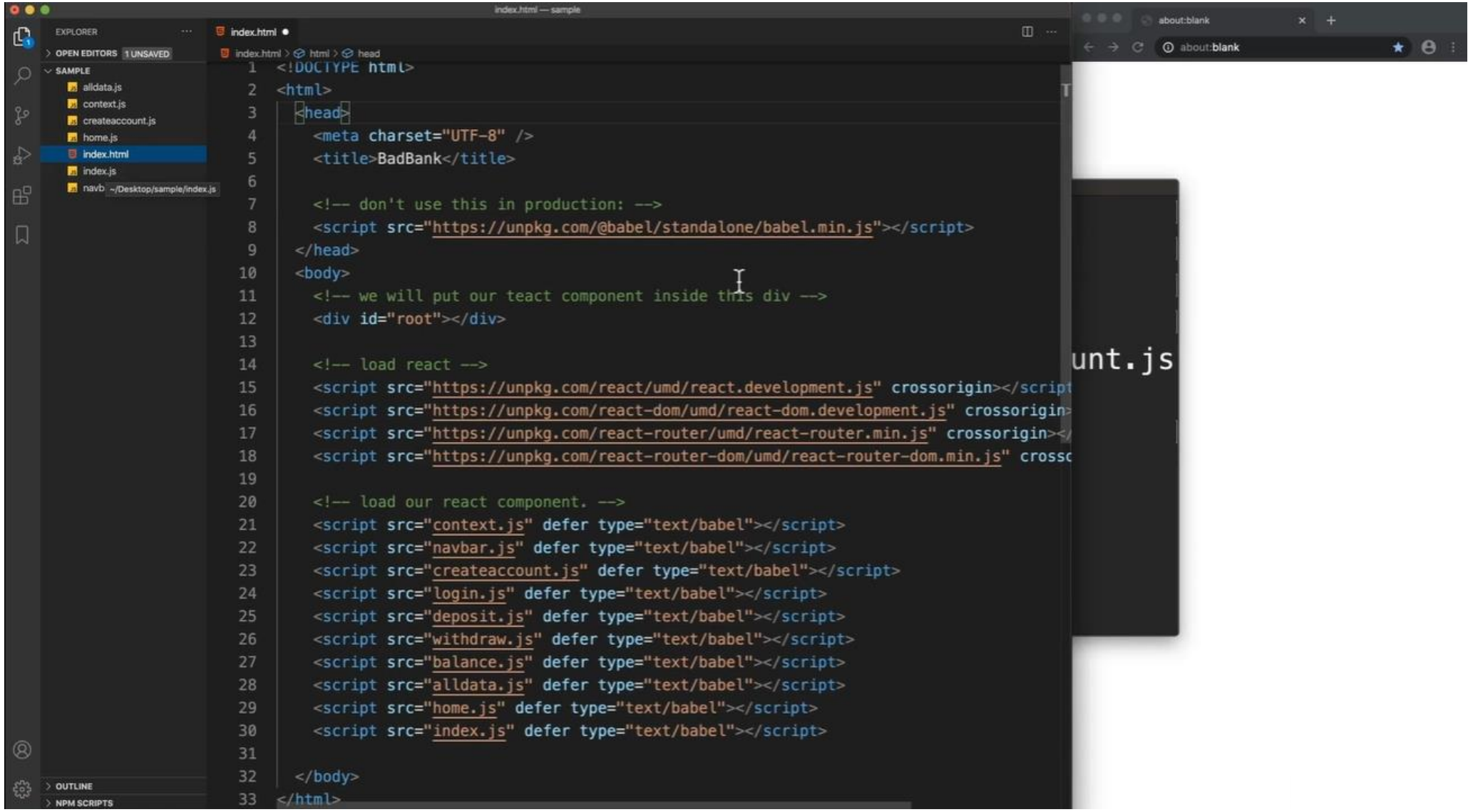
Components Loaded: Demonstration (1/6)



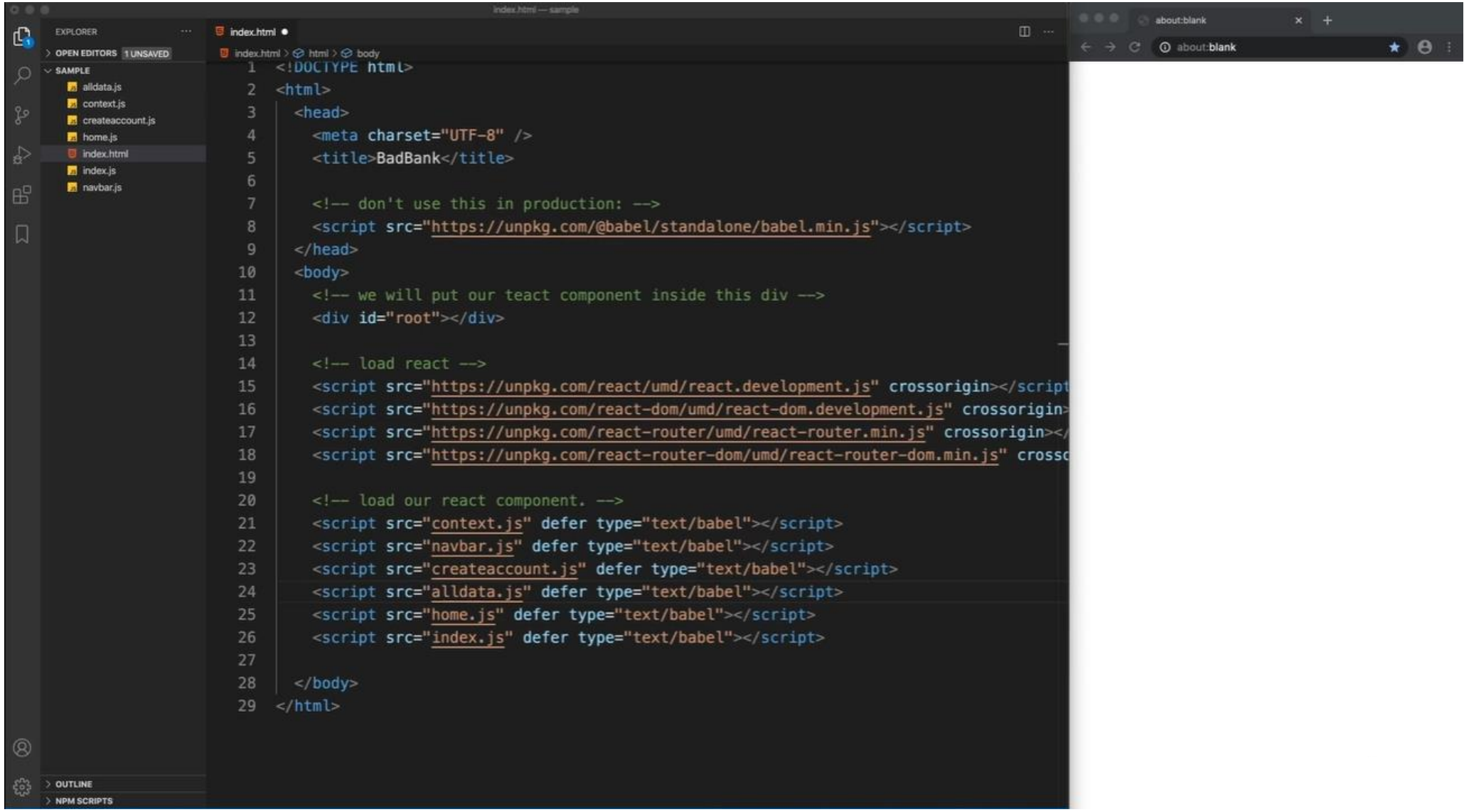
Components Loaded: Demonstration (2/6)



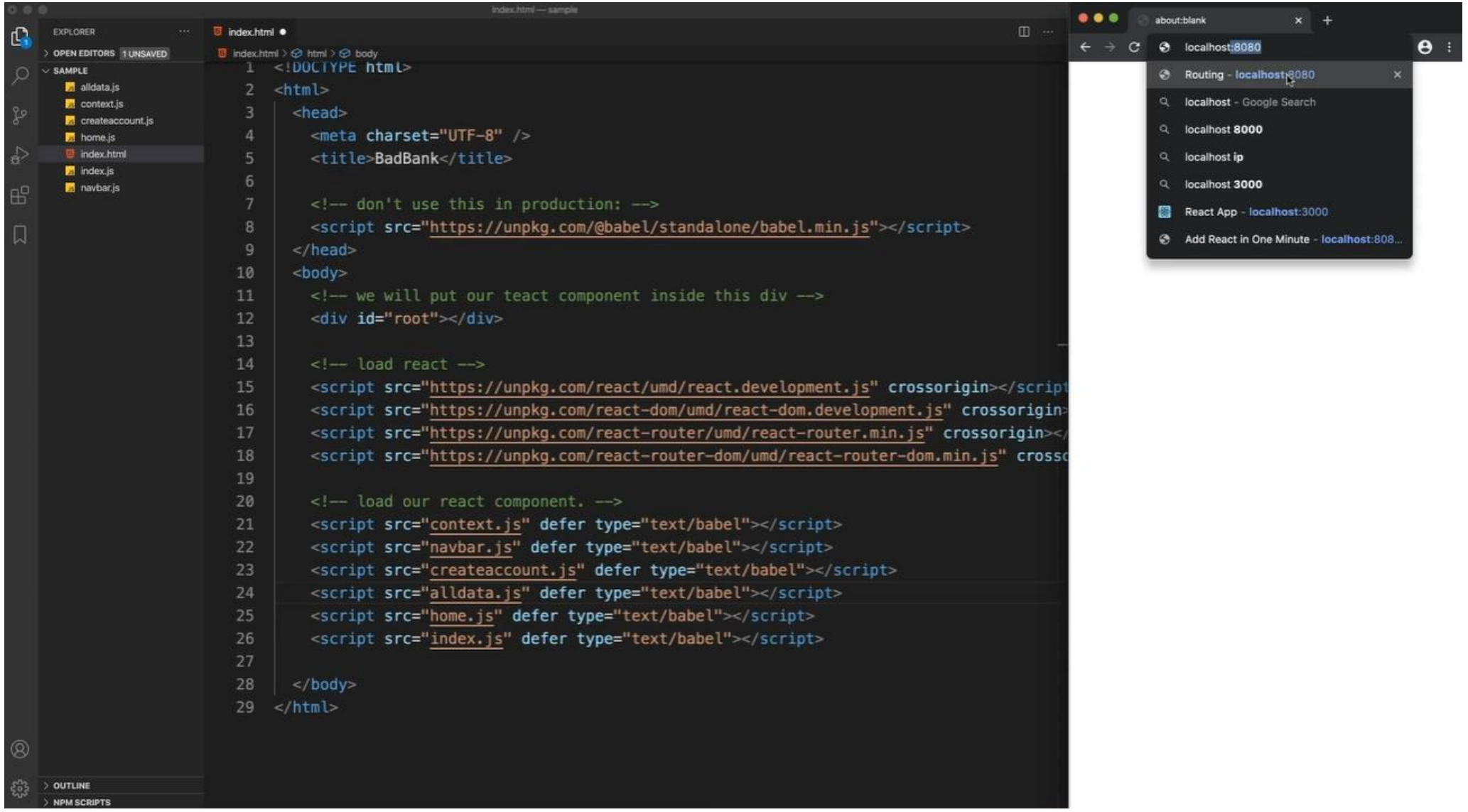
Components Loaded: Demonstration (3/6)



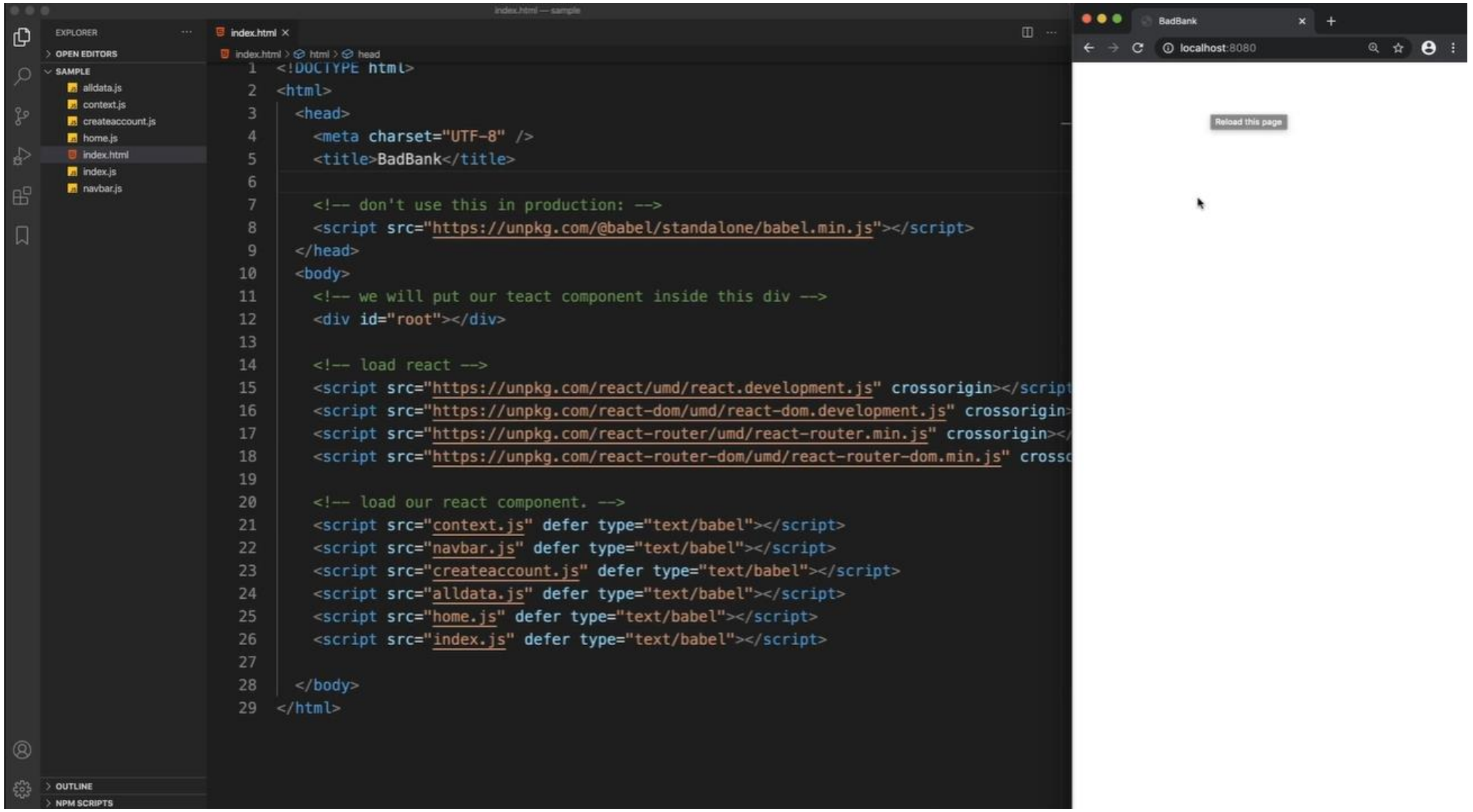
Components Loaded: Demonstration (4/6)



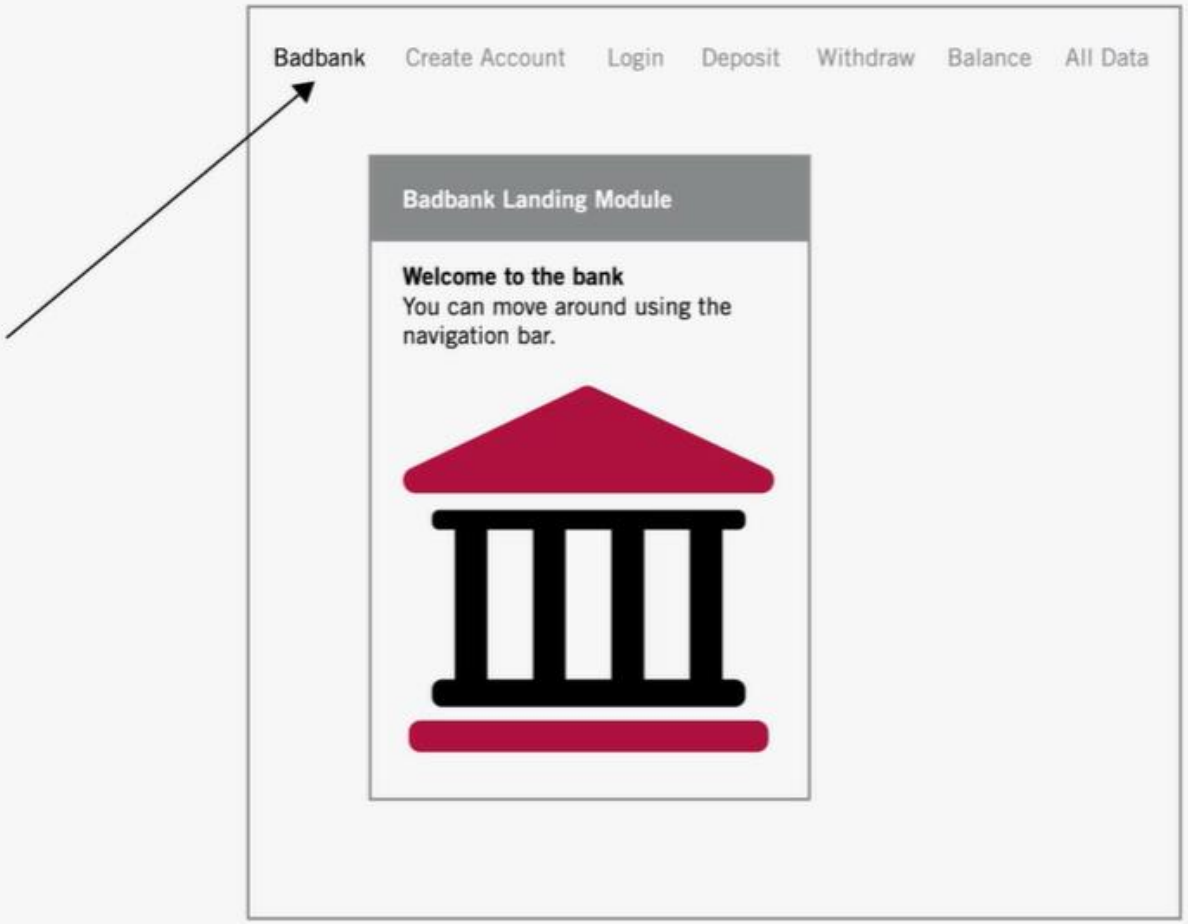
Components Loaded: Demonstration (5/6)



Components Loaded: Demonstration (6/6)



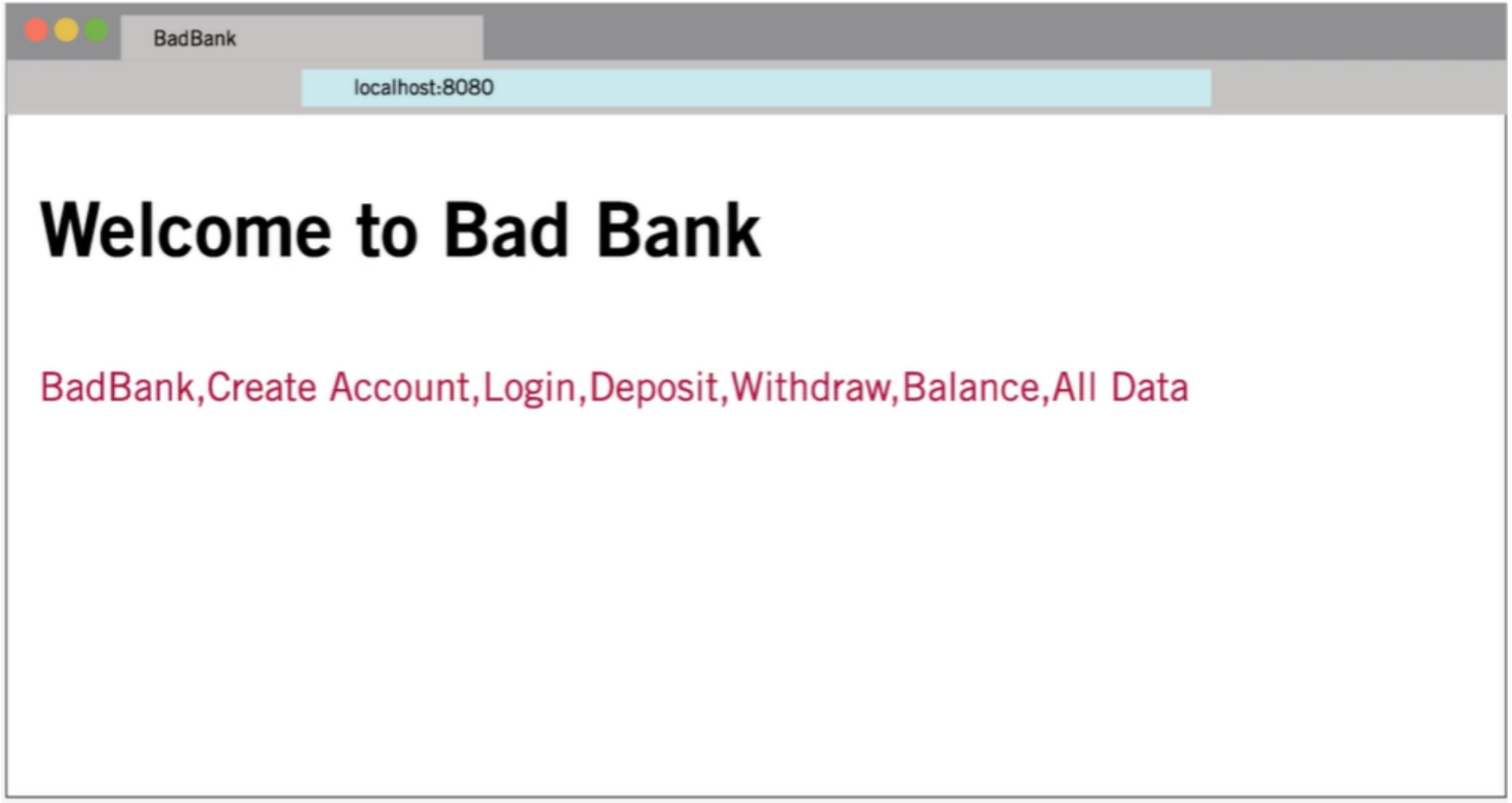
Navigation Bar



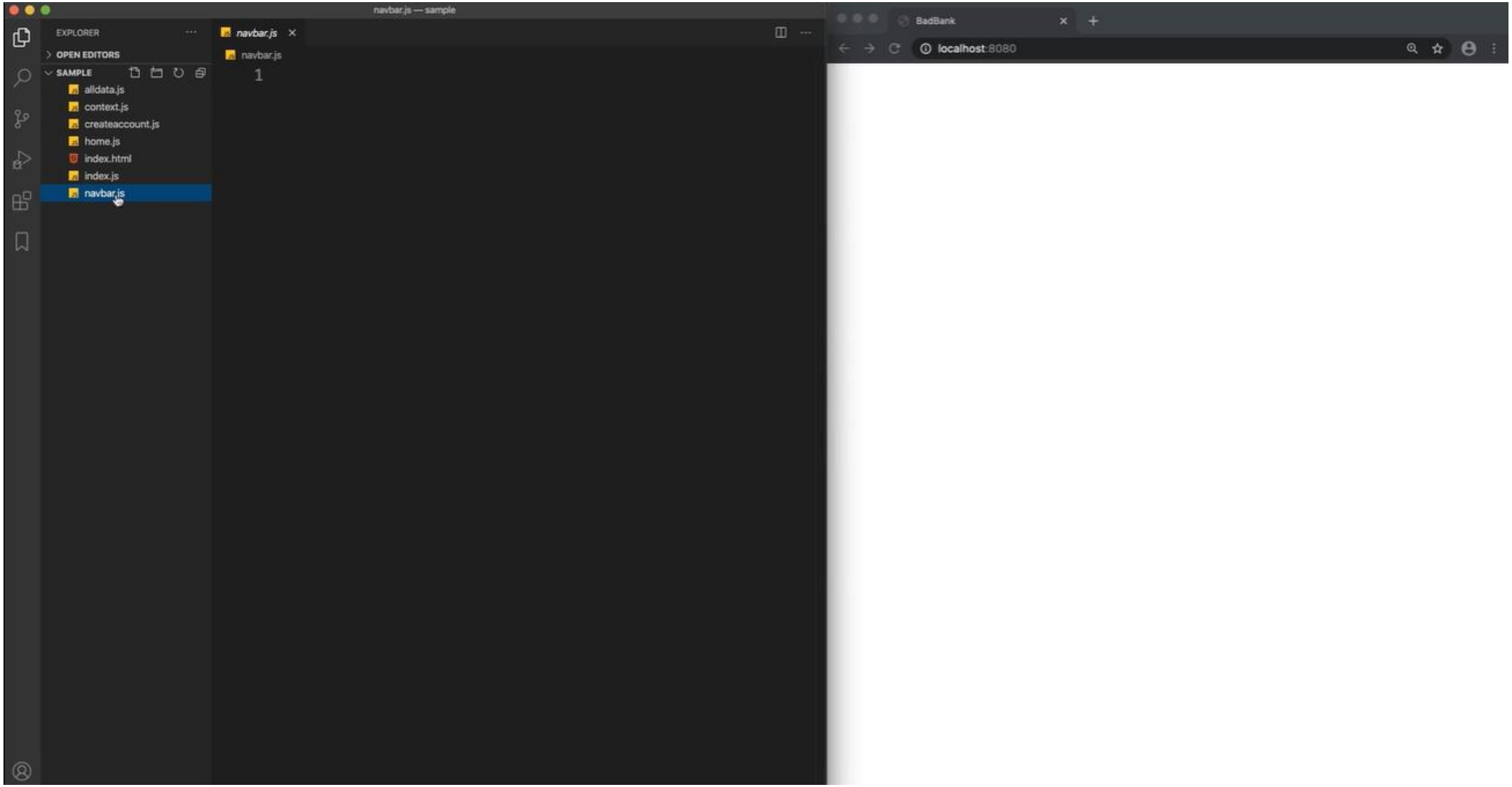
Components:

- Home
- Create account
- Login
- Deposit
- Withdraw
- Balance
- All data

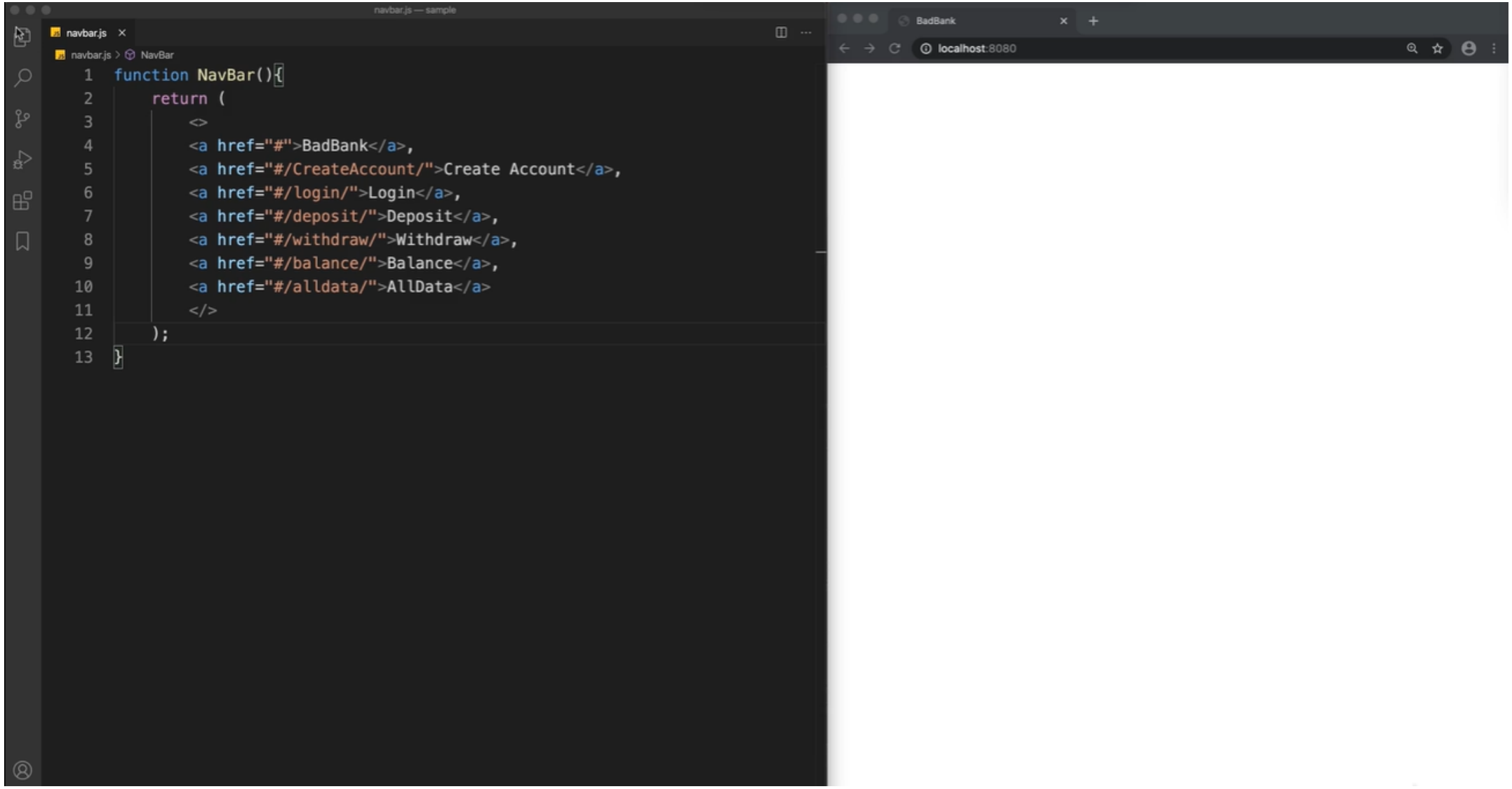
Output Preview



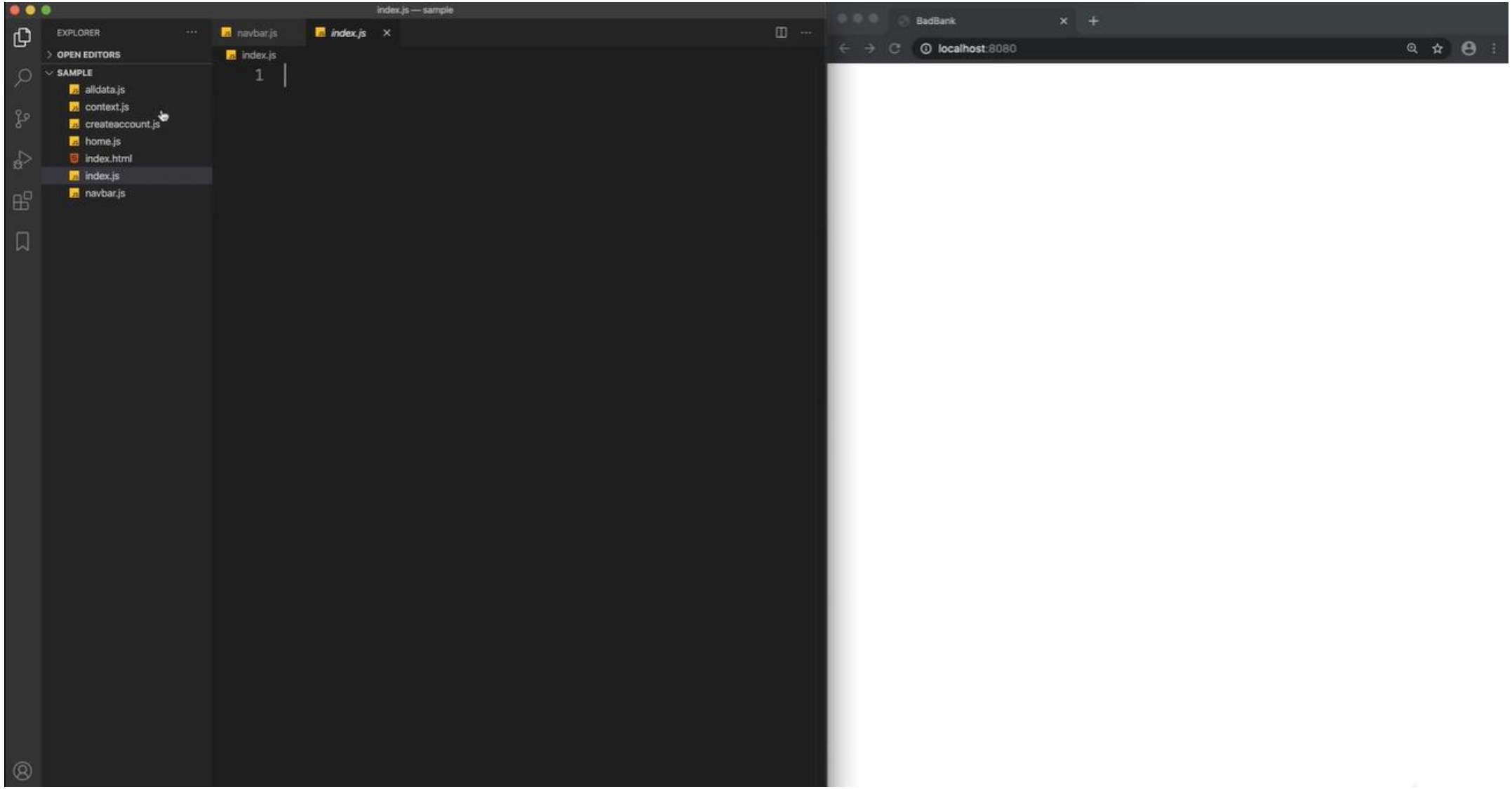
Navigation bar: Demonstration (1/5)



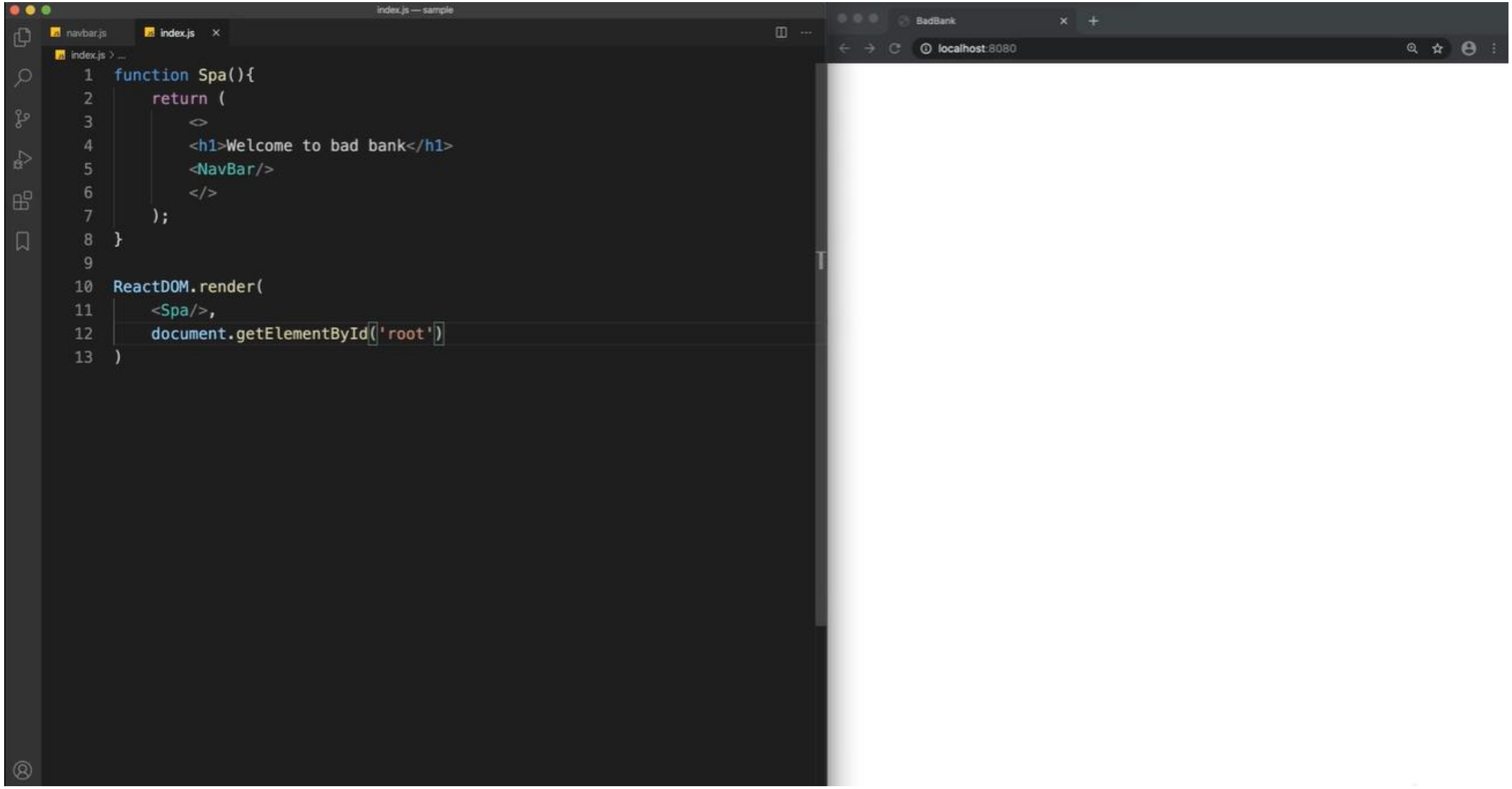
Navigation bar: Demonstration (2/5)



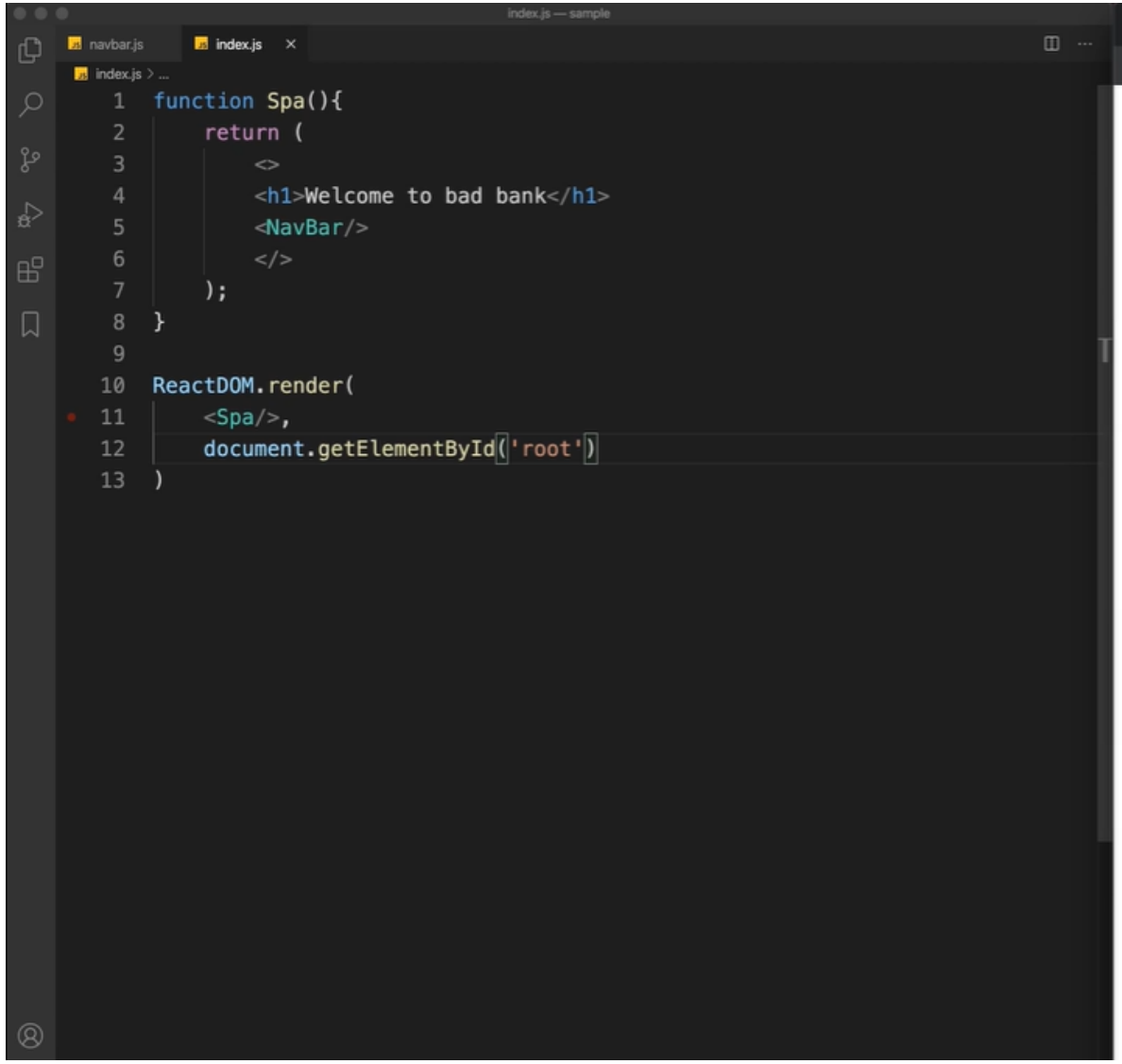
Navigation bar: Demonstration (3/5)

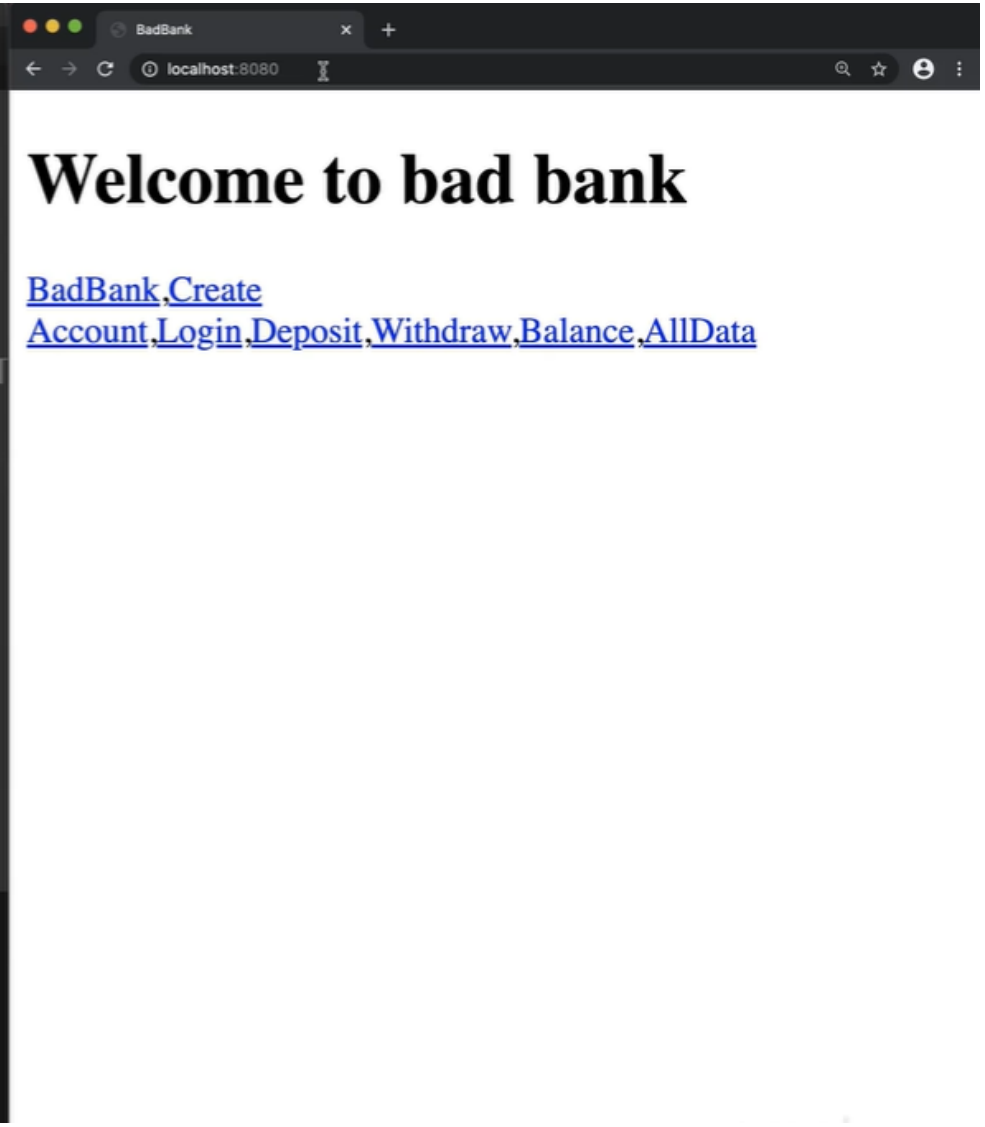


Navigation bar: Demonstration (4/5)

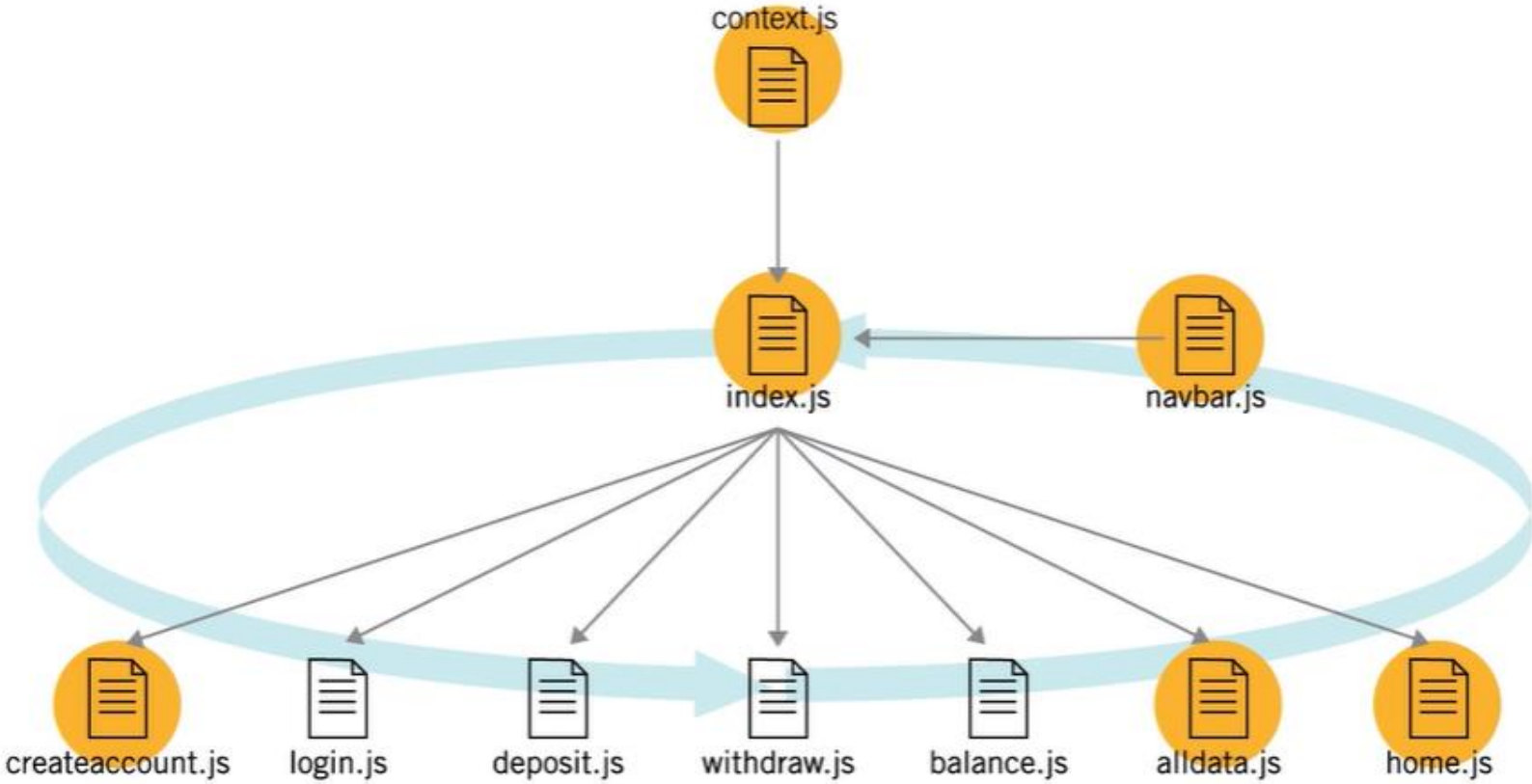


Navigation bar: Demonstration (5/5)





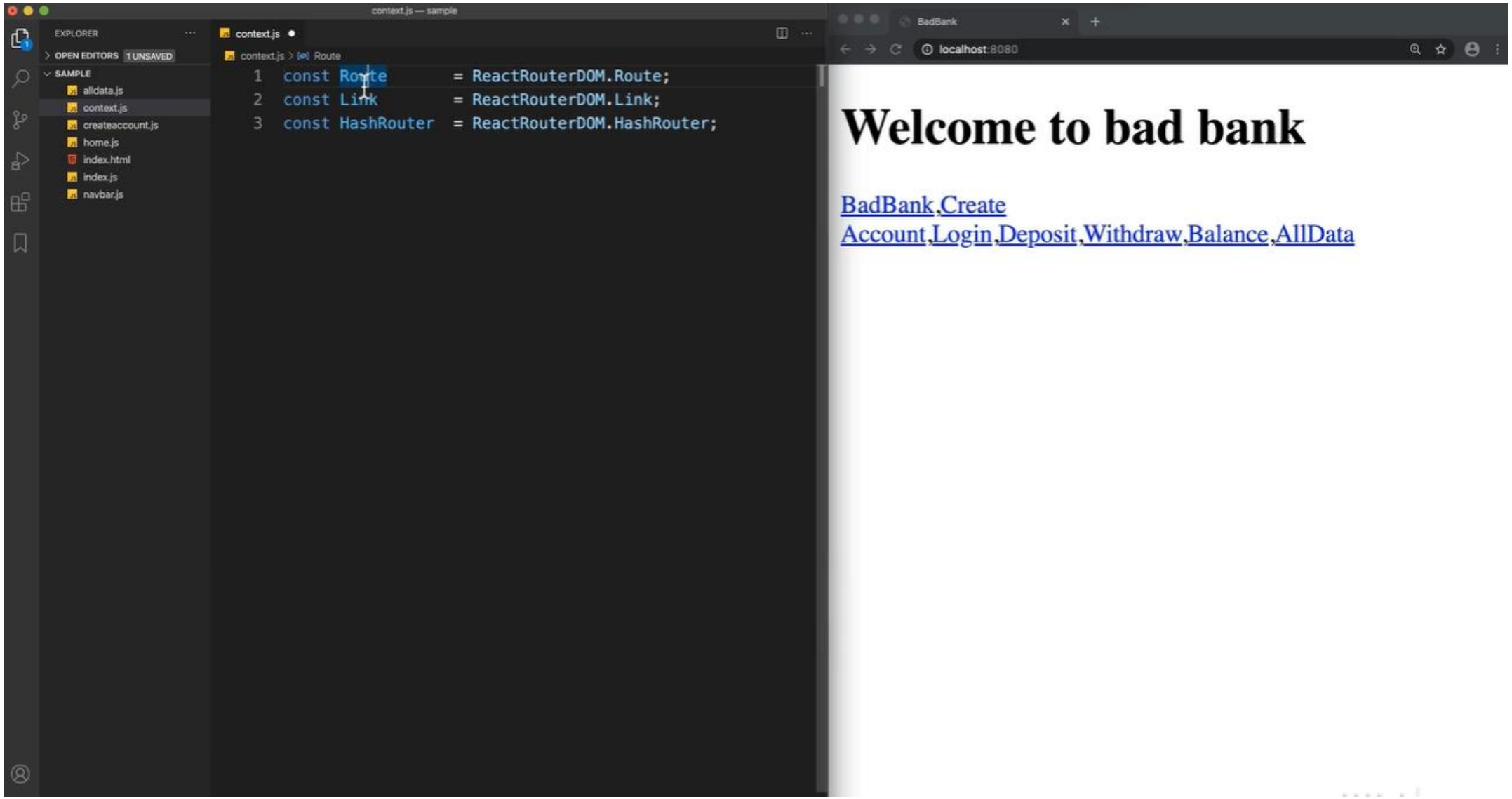
Routing


context.js
index.js
navbar.js
createaccount.js
login.js
deposit.js
withdraw.js
balance.js
alldata.js
home.js

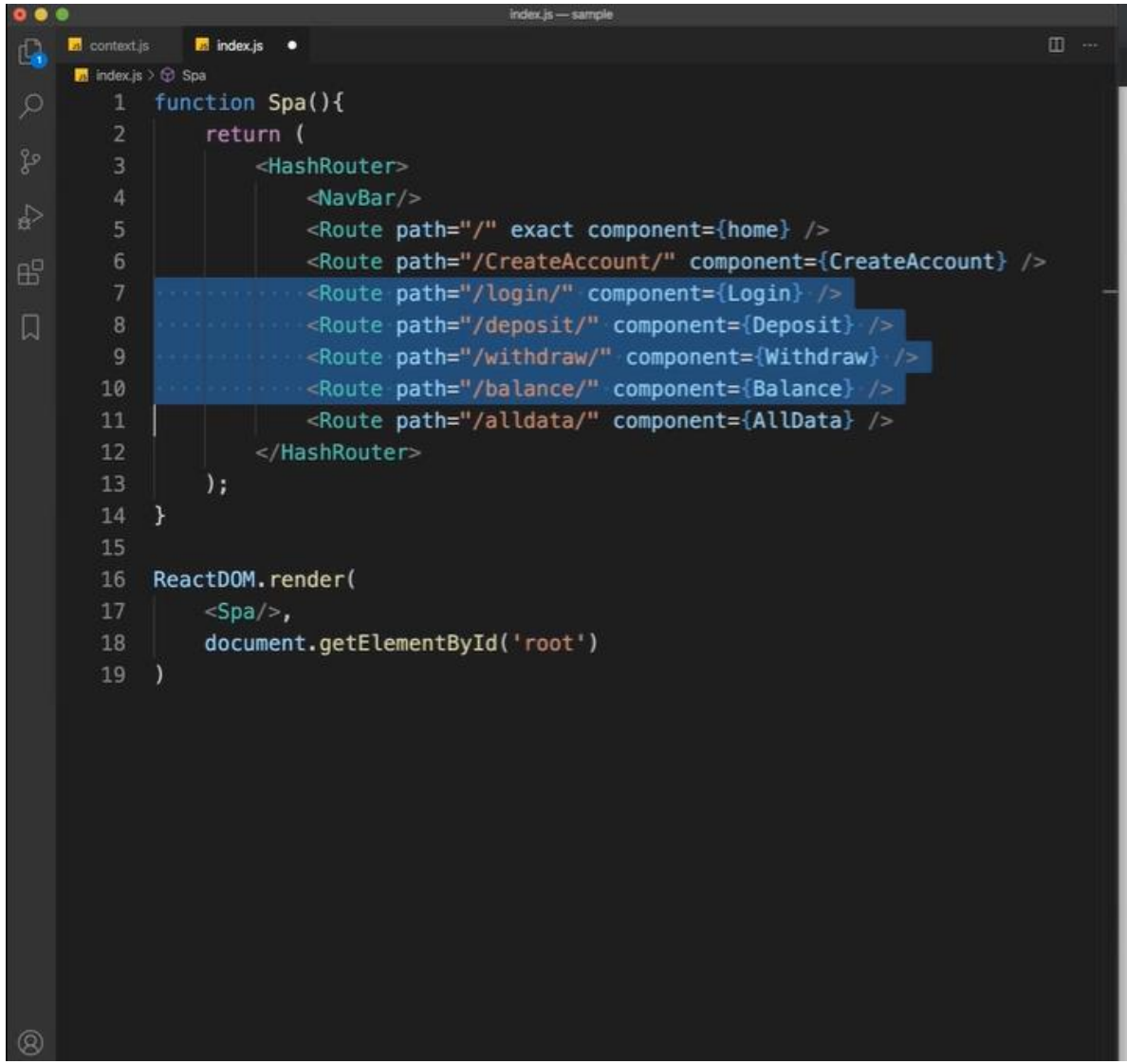
Routing – Route, Link, HashRouter

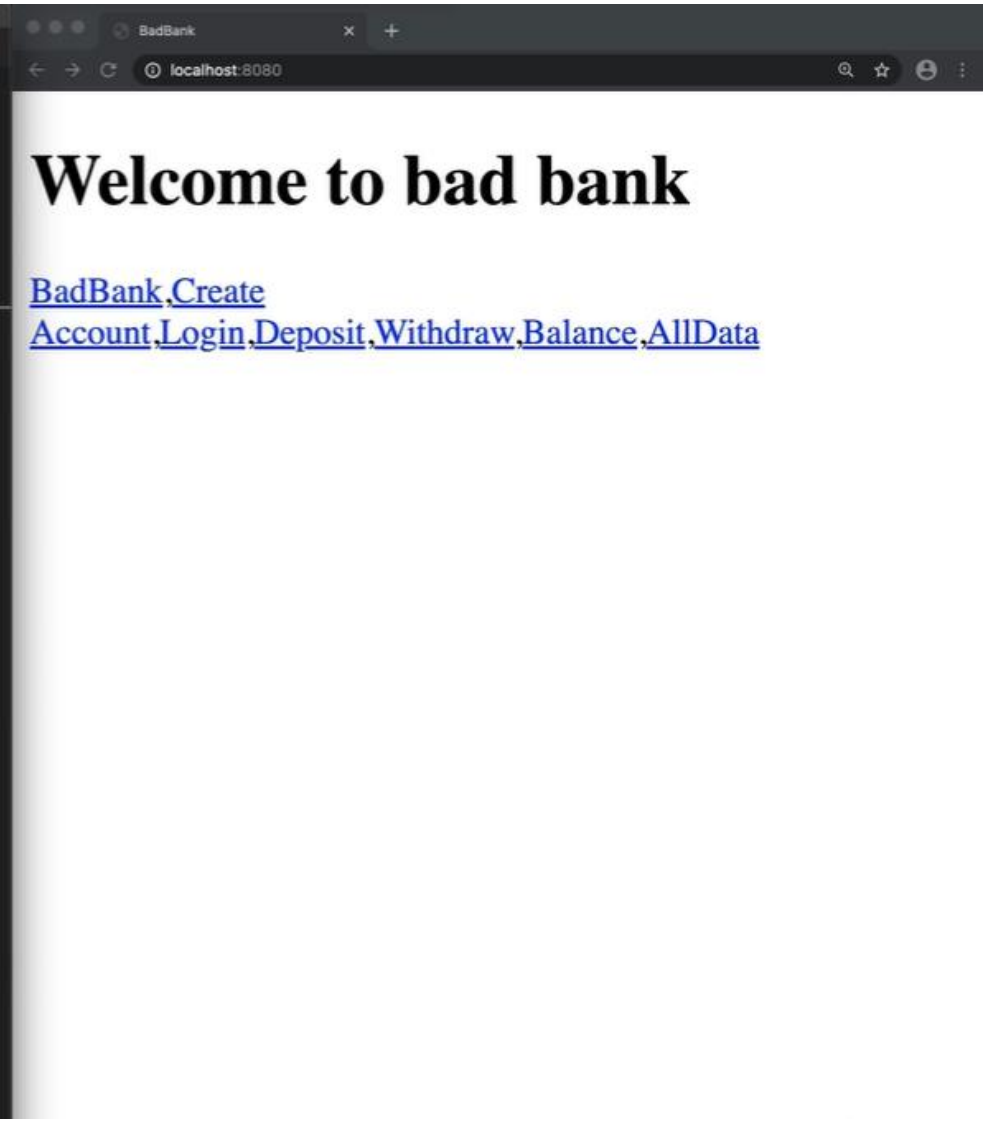
```
const Route = ReactRouterDOM.Route;
const Link = ReactRouterDOM.Link;
const HashRouter = ReactRouterDOM.HashRouter;
```

Navbar Routing: Demonstration (1/9)

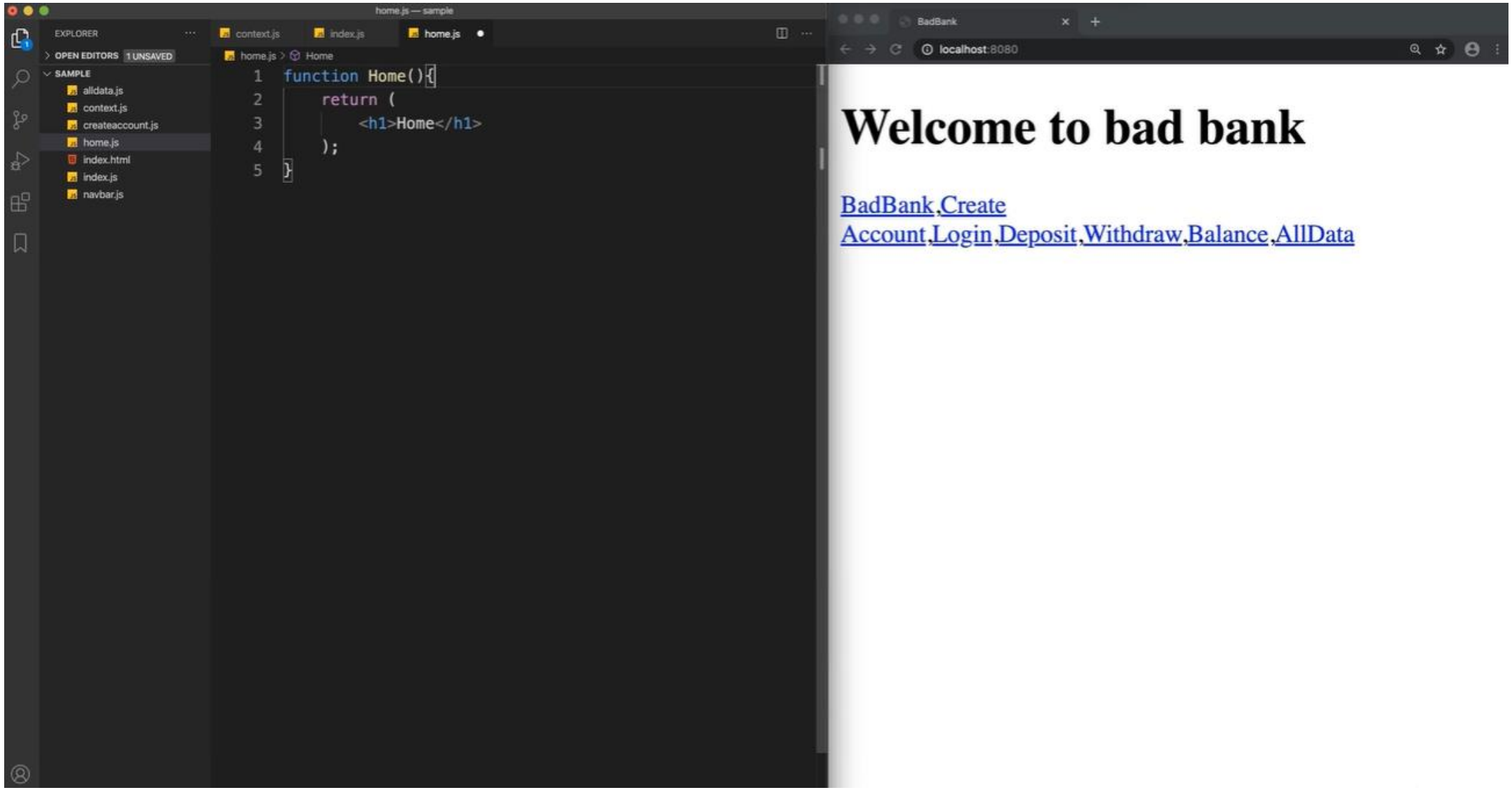


Navbar Routing: Demonstration (2/9)

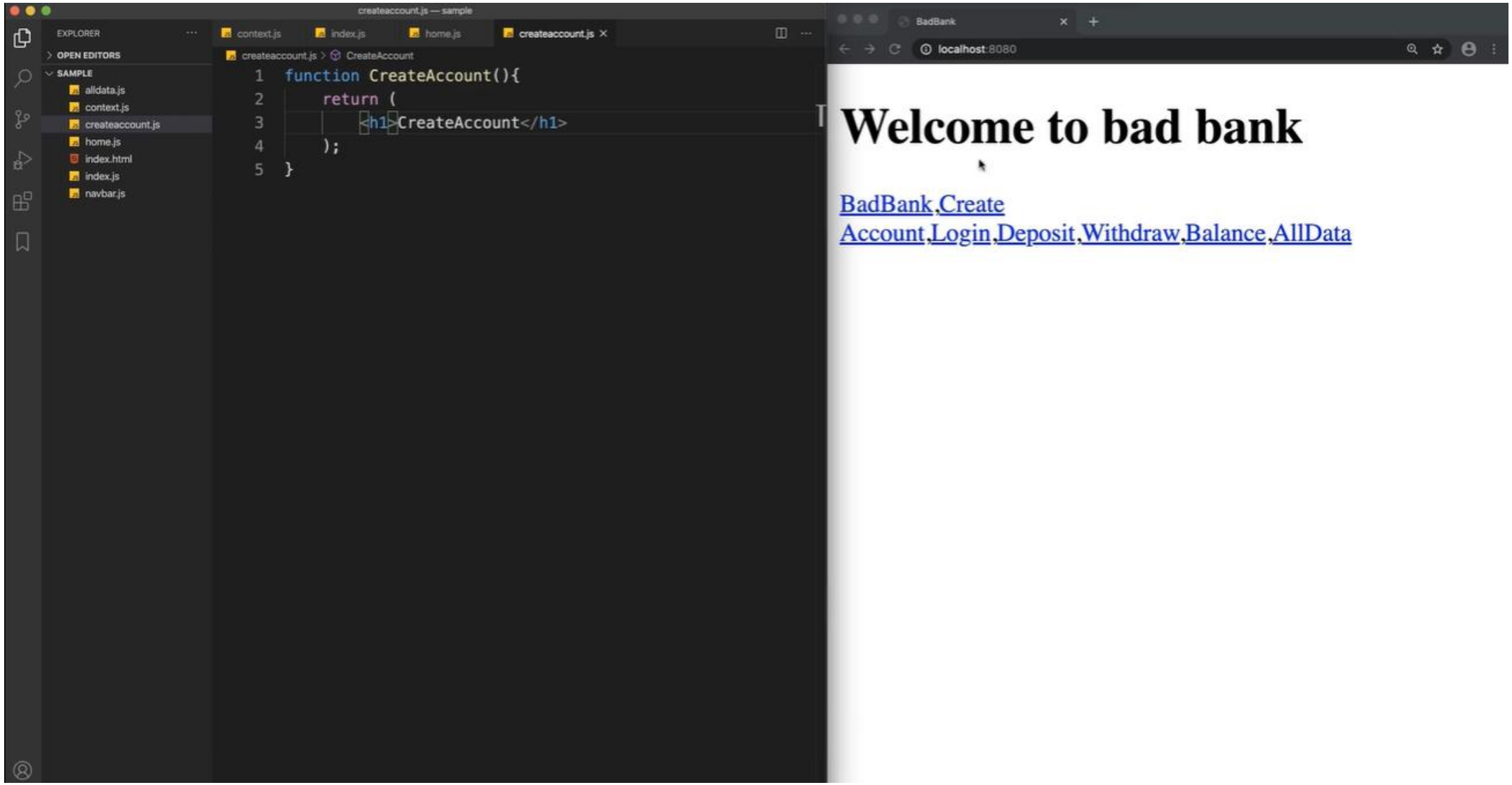




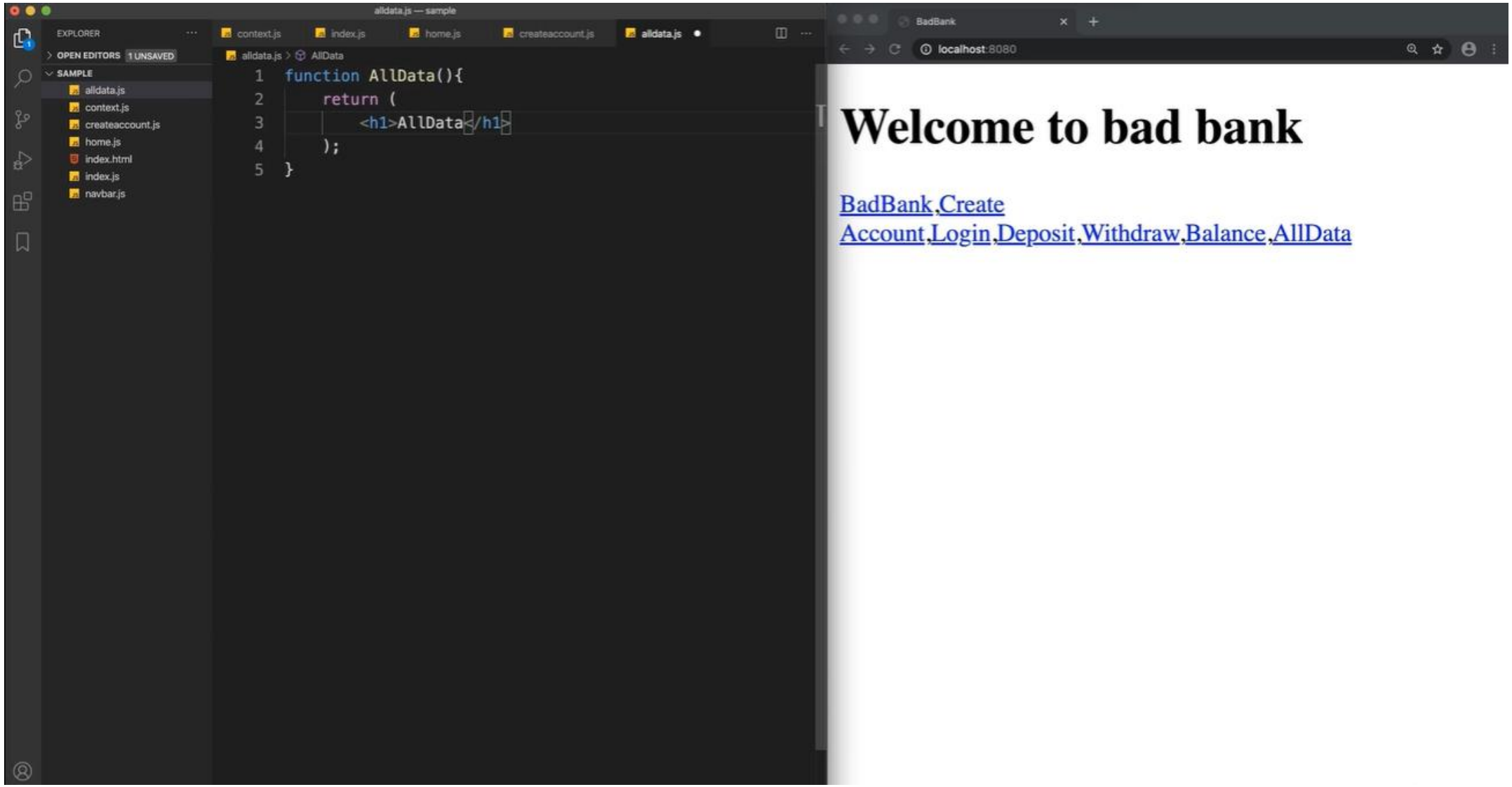
Navbar Routing: Demonstration (3/9)



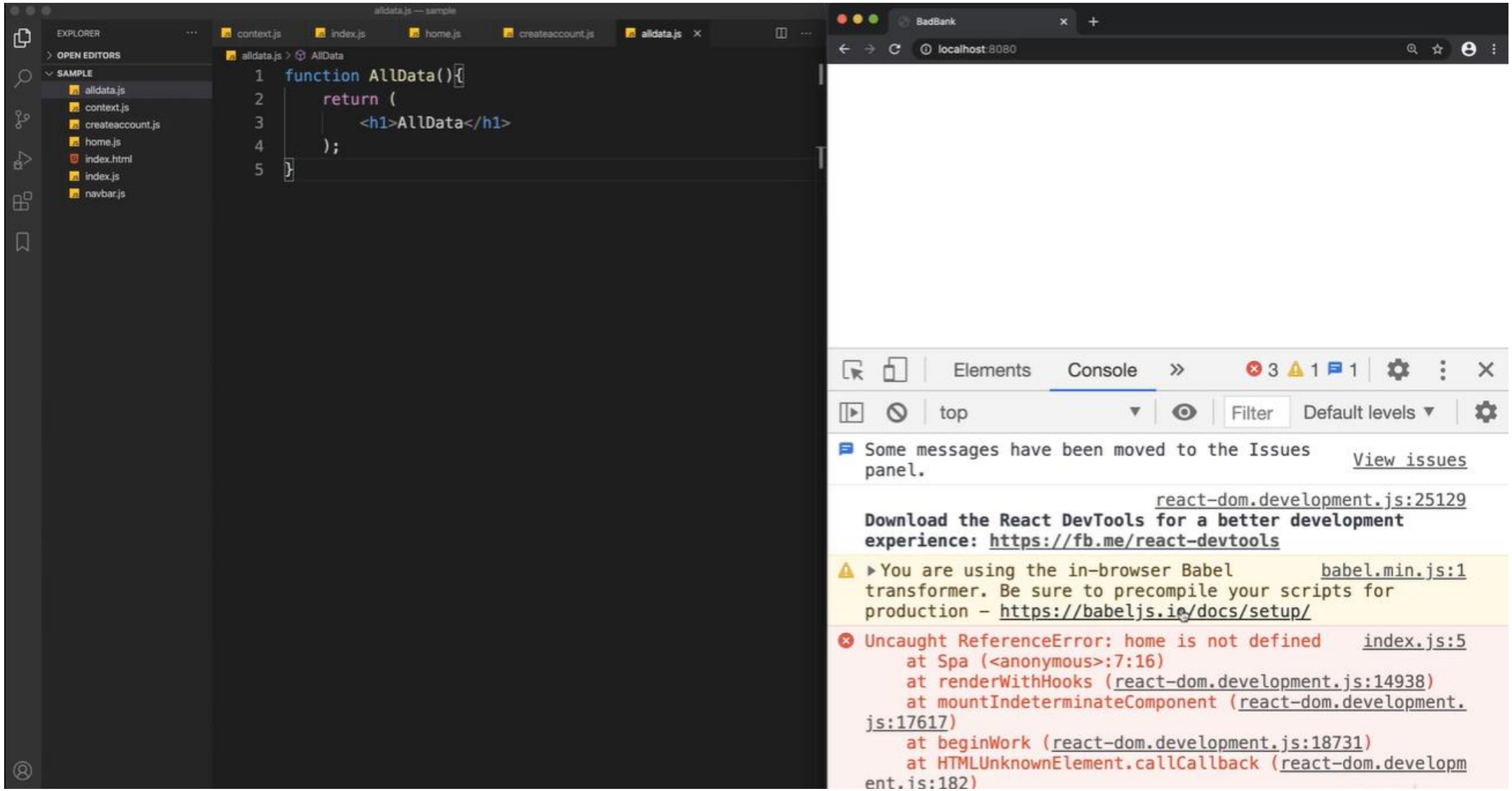
Navbar Routing: Demonstration (4/9)



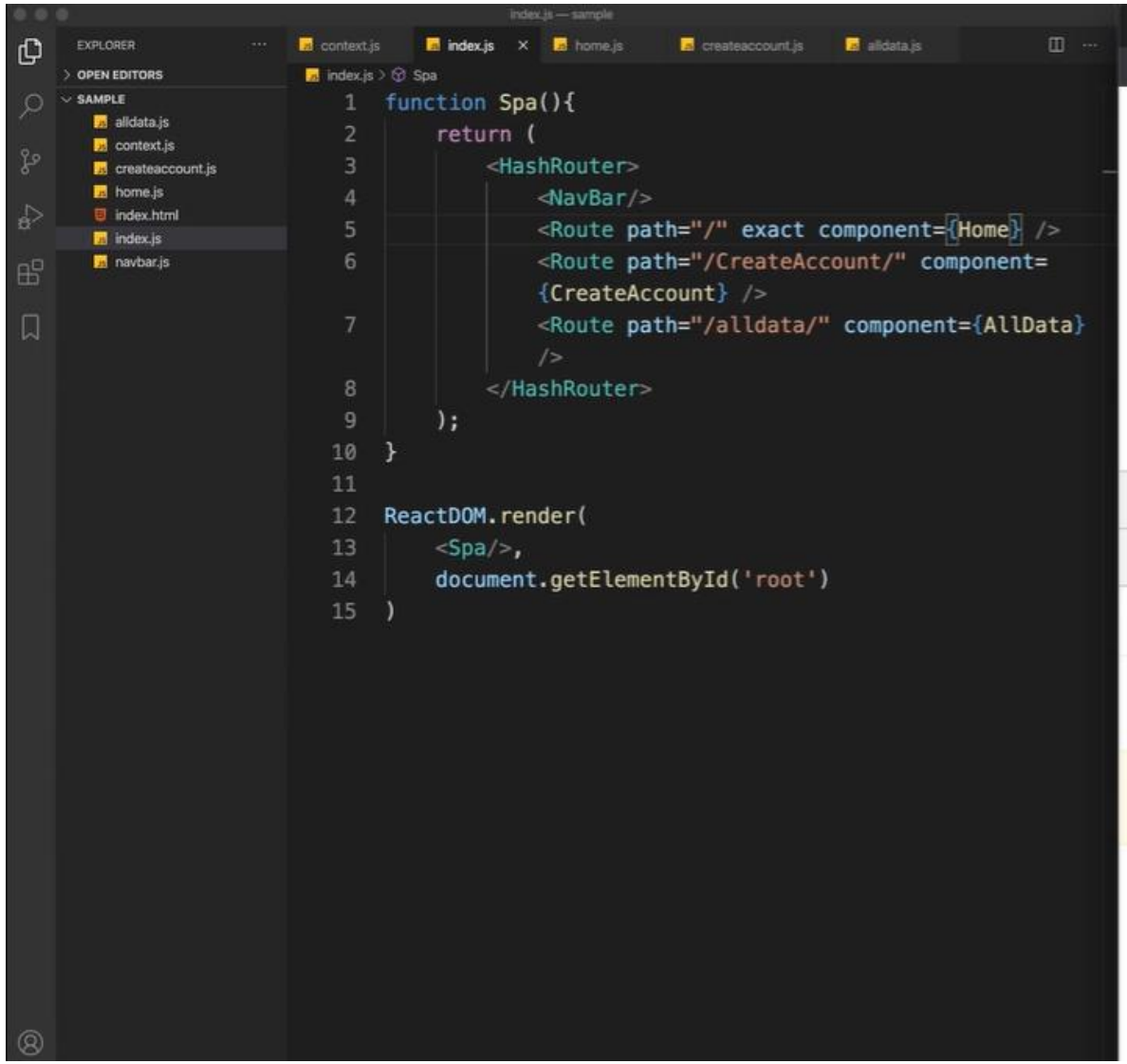
Navbar Routing: Demonstration (5/9)

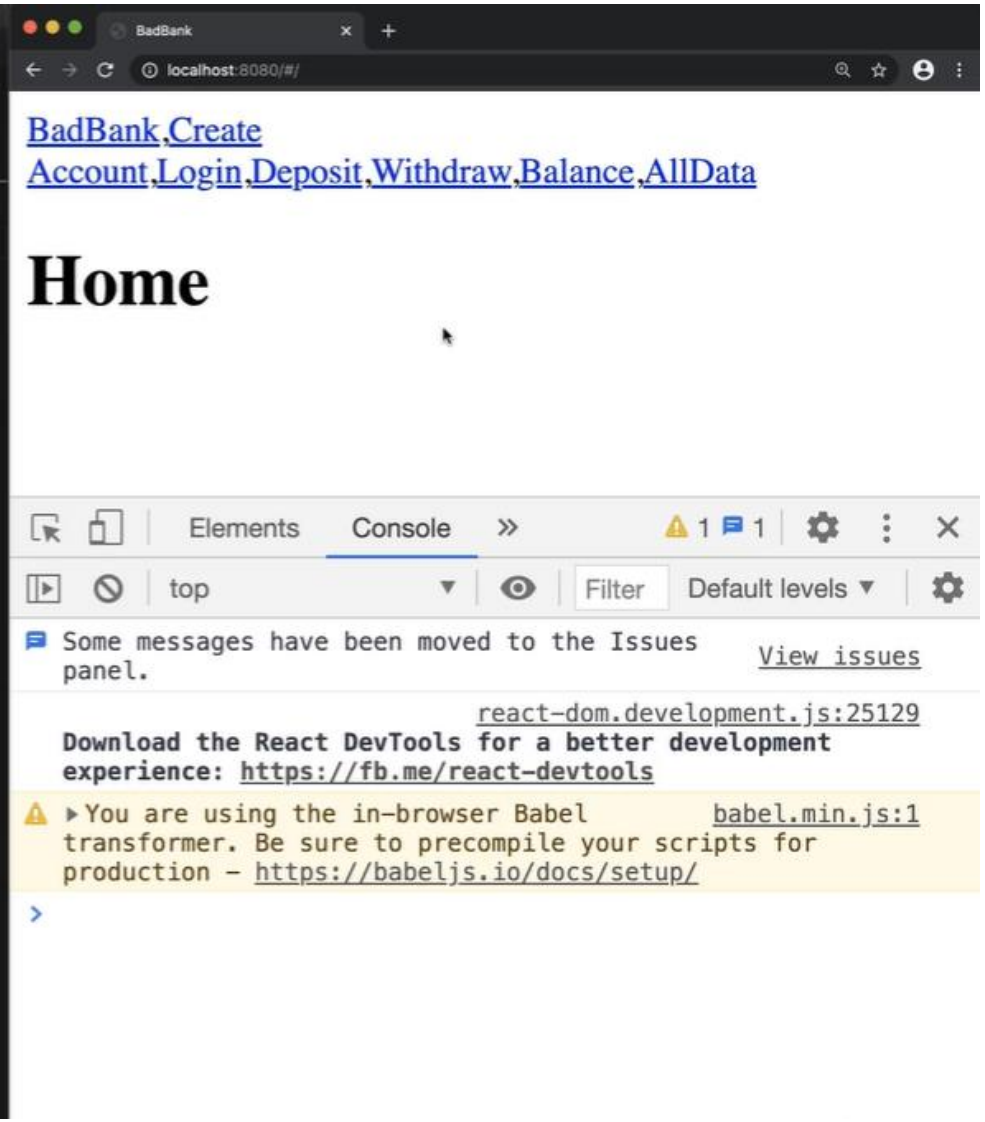


Navbar Routing: Demonstration (6/9)

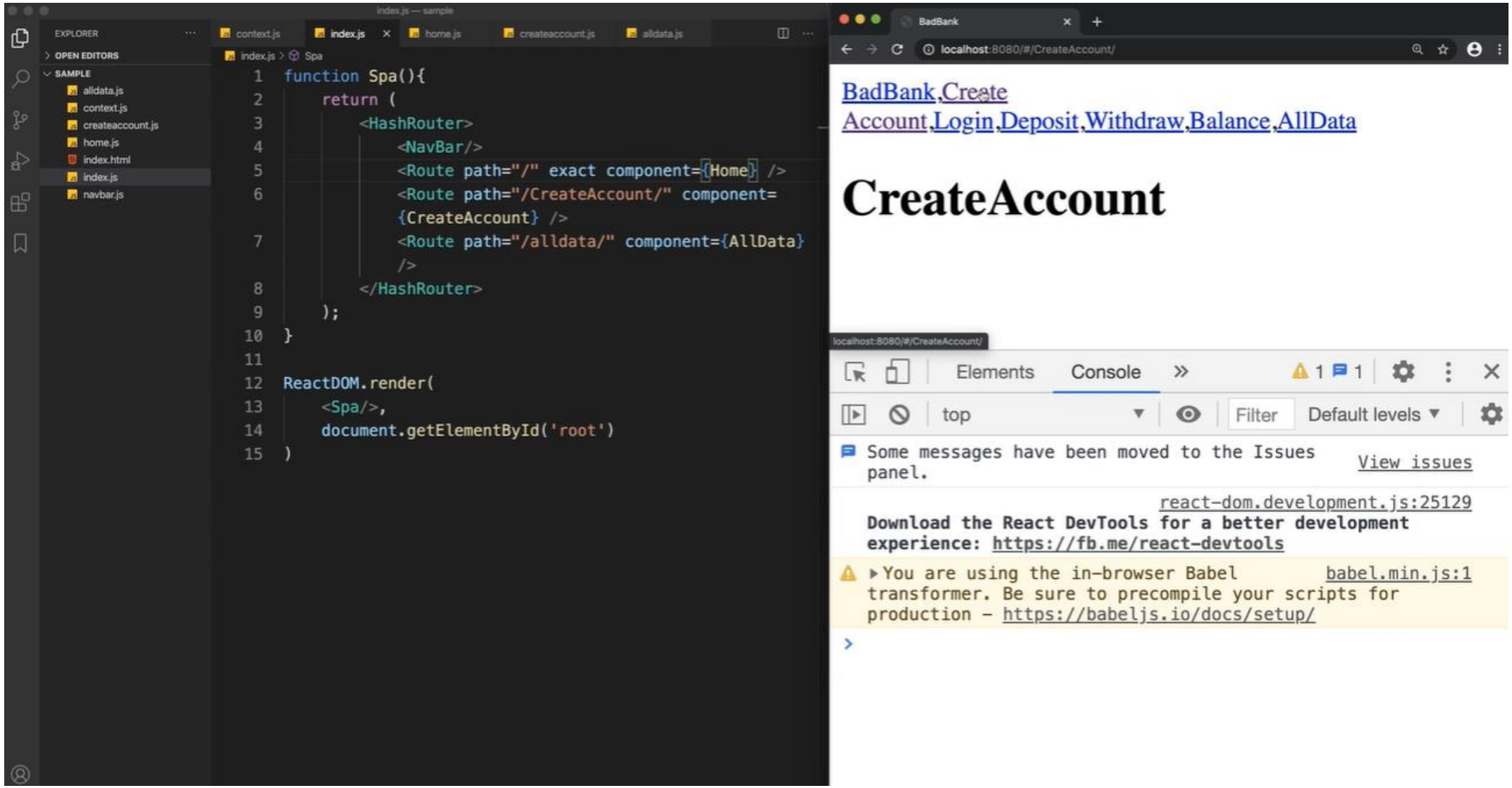


Navbar Routing: Demonstration (7/9)

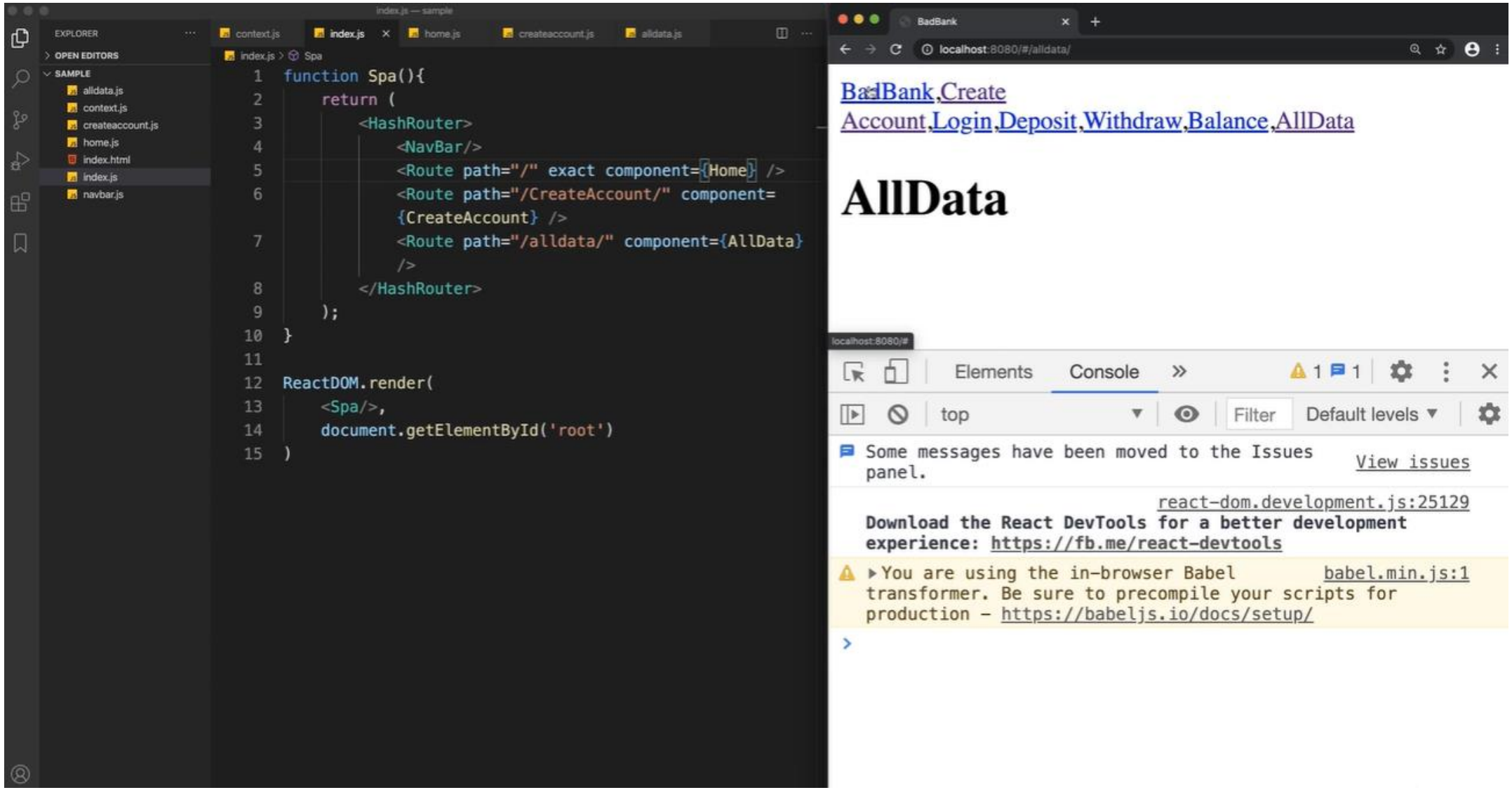




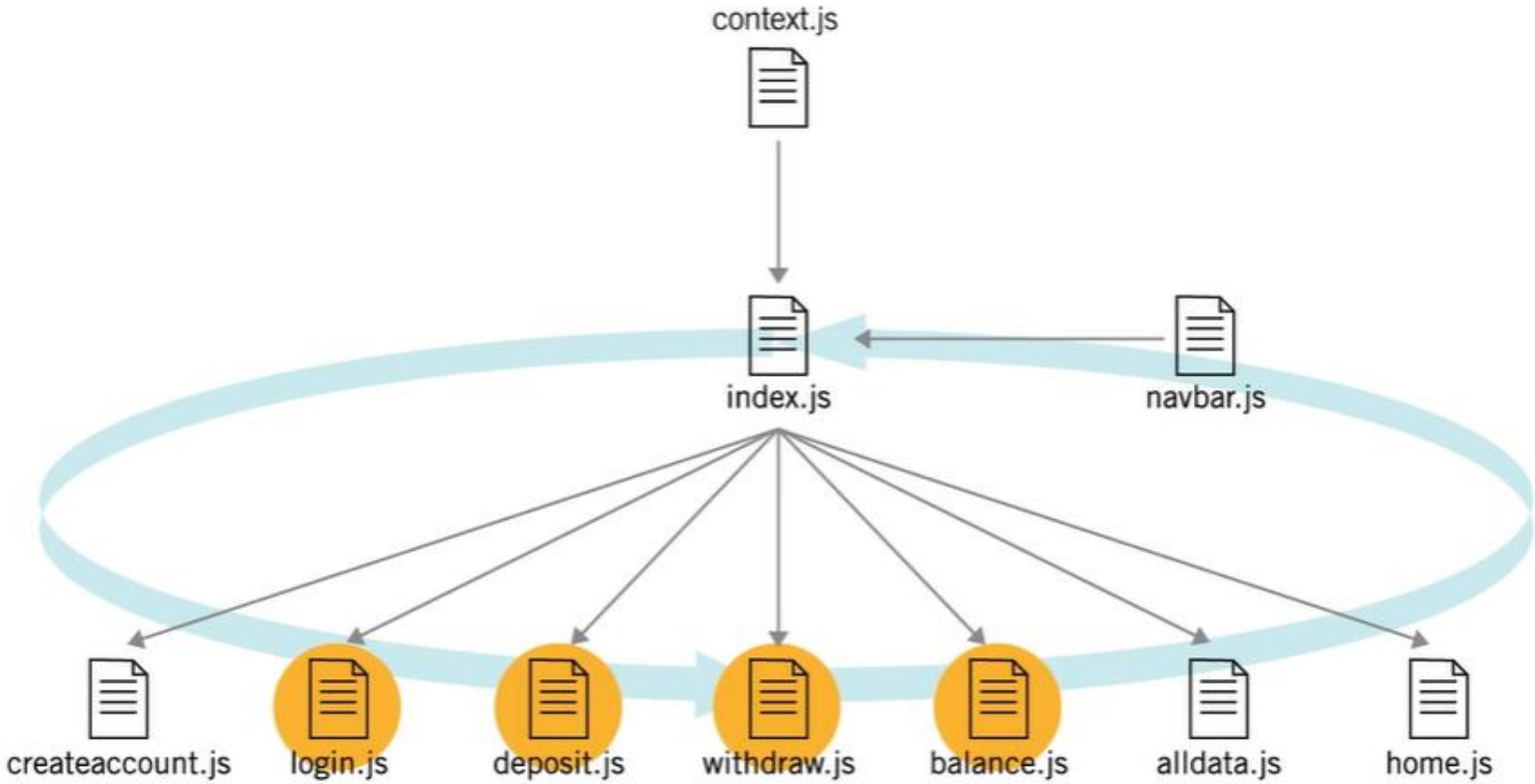
Navbar Routing: Demonstration (8/9)



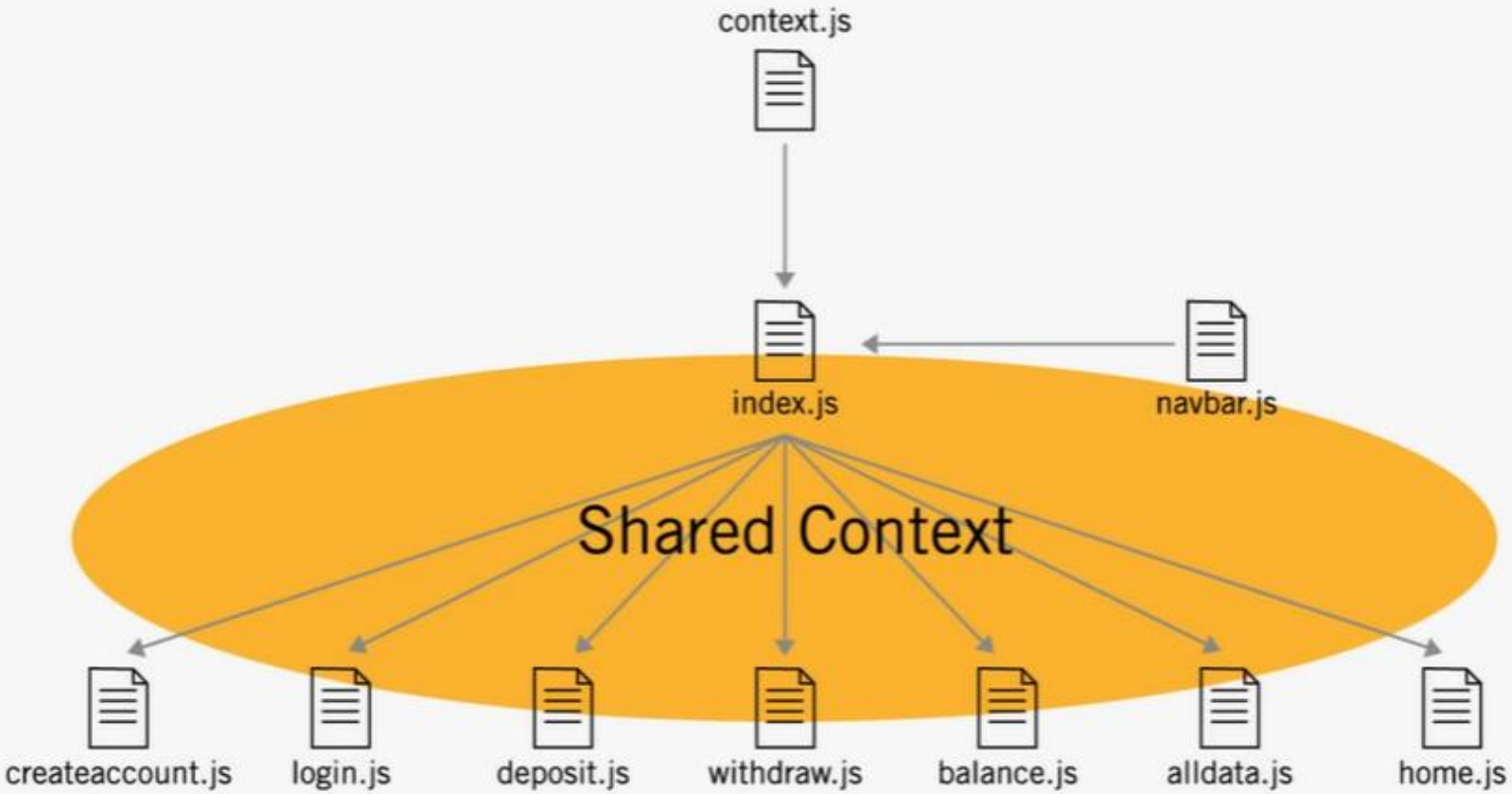
Navbar Routing: Demonstration (9/9)



Navbar Routing: Your Turn



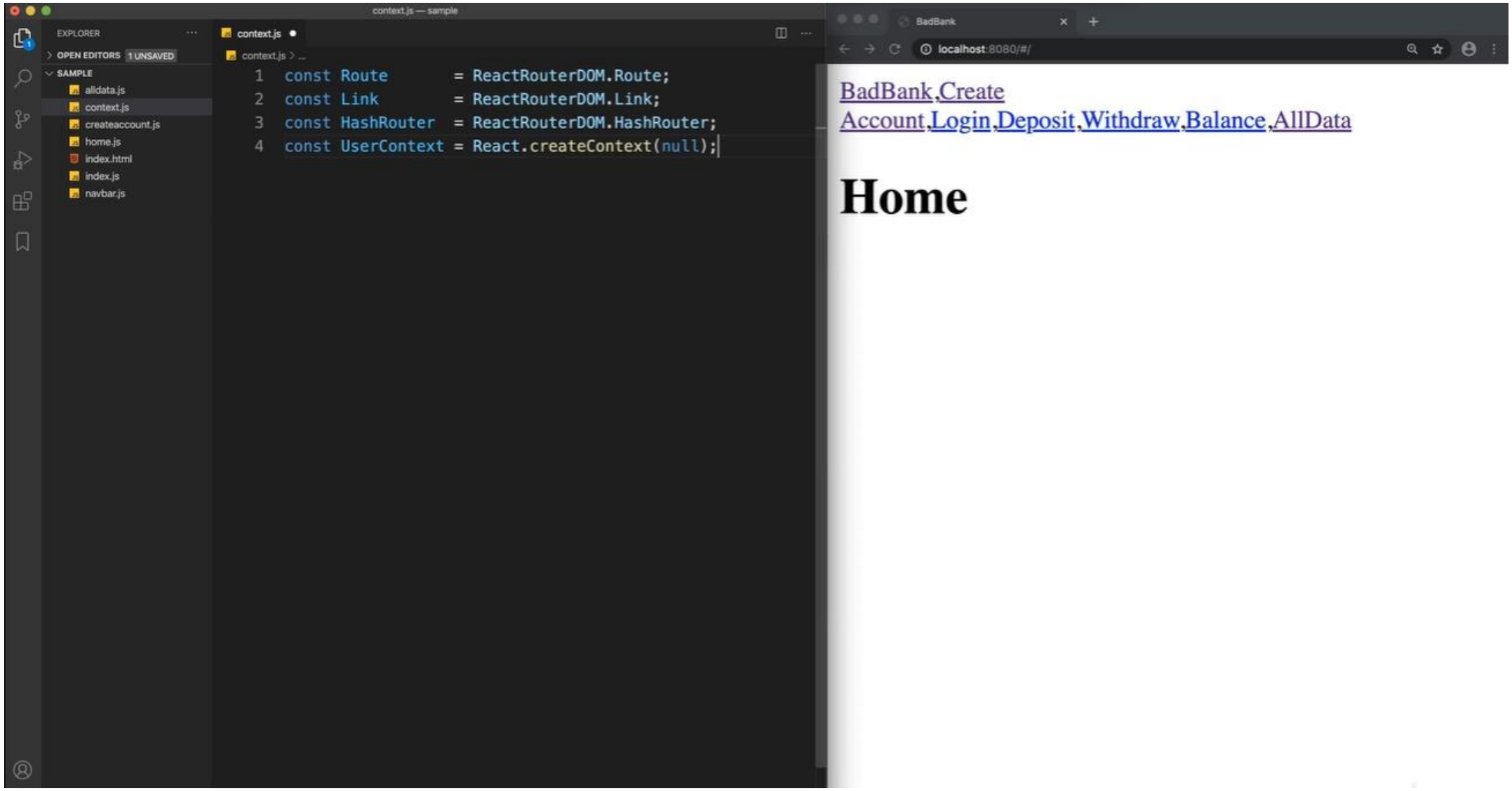
Shared Context



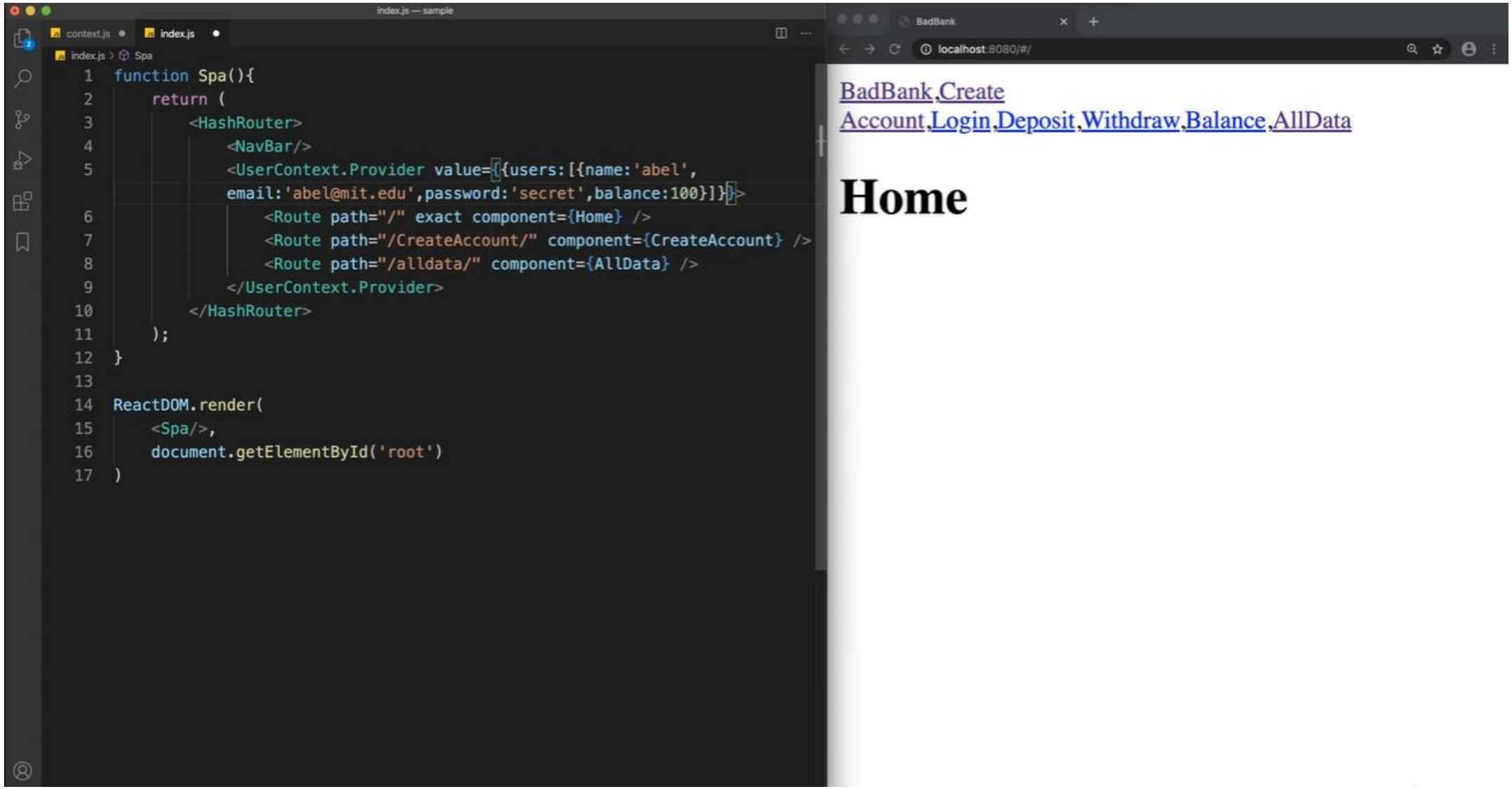
Create Context

- Context object comes with a provider that takes a value.
- The value is passed to consuming components.

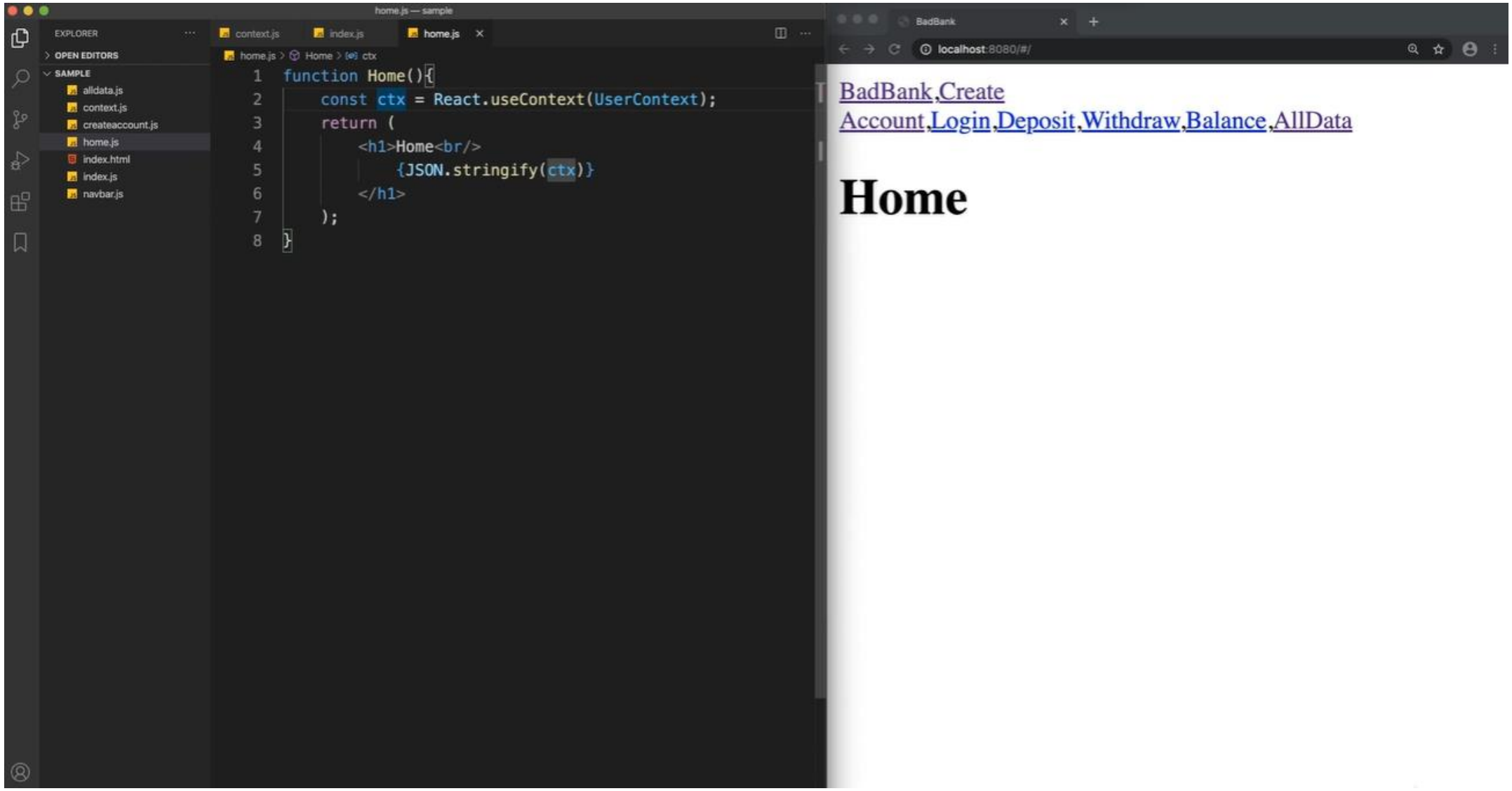
Shared Context: Demonstration (1/8)



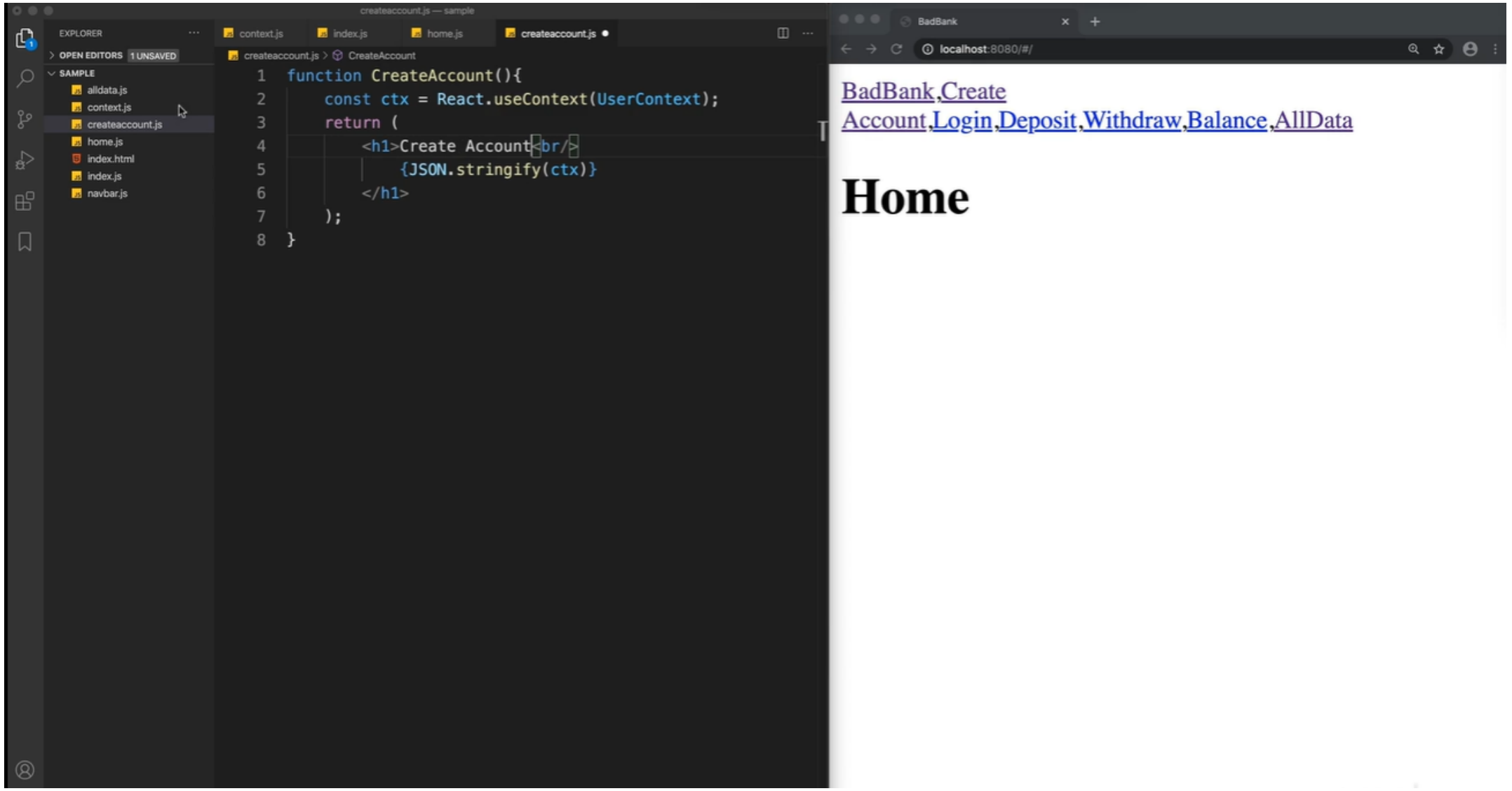
Shared Context: Demonstration (2/8)



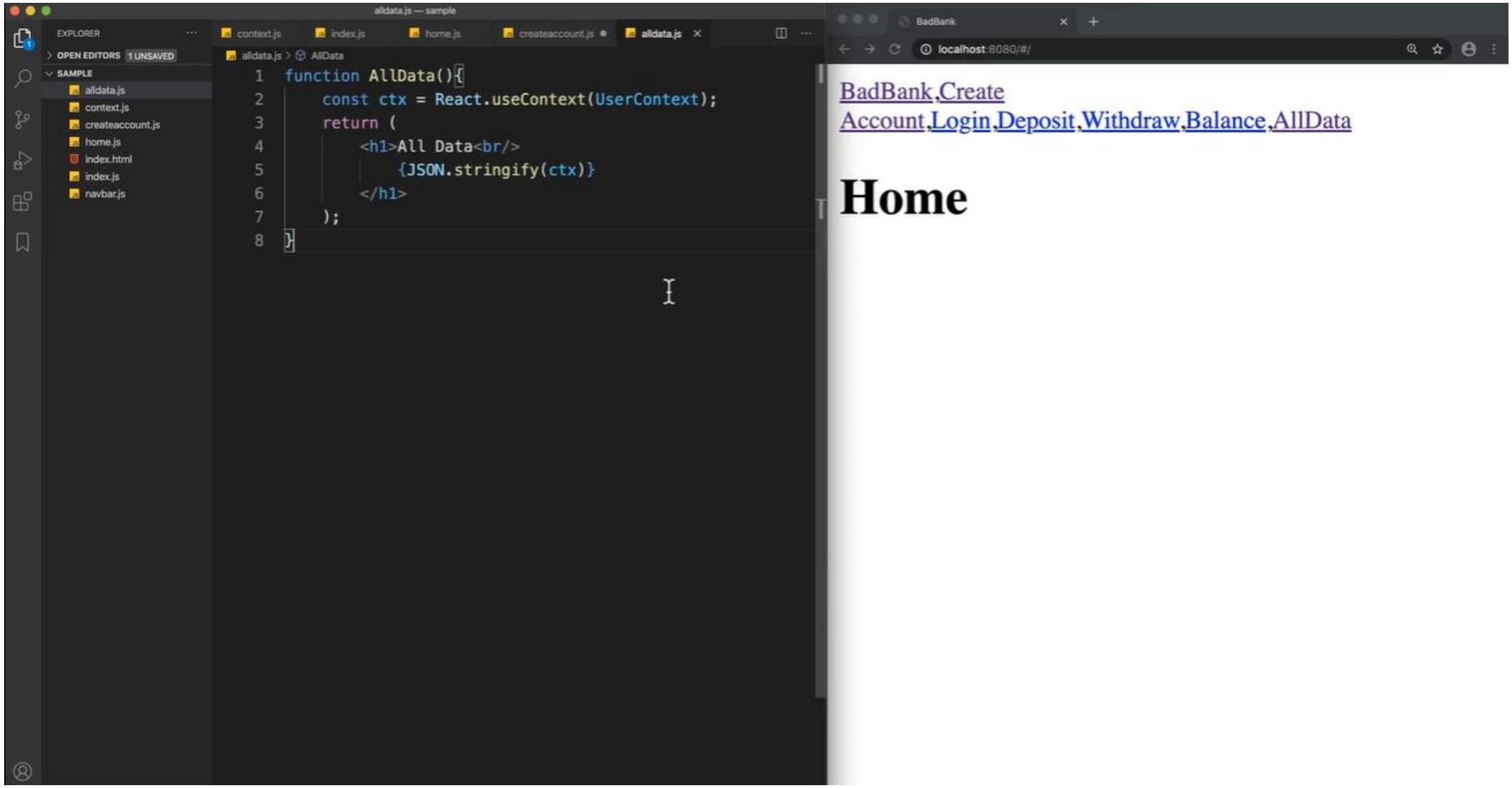
Shared Context: Demonstration (3/8)



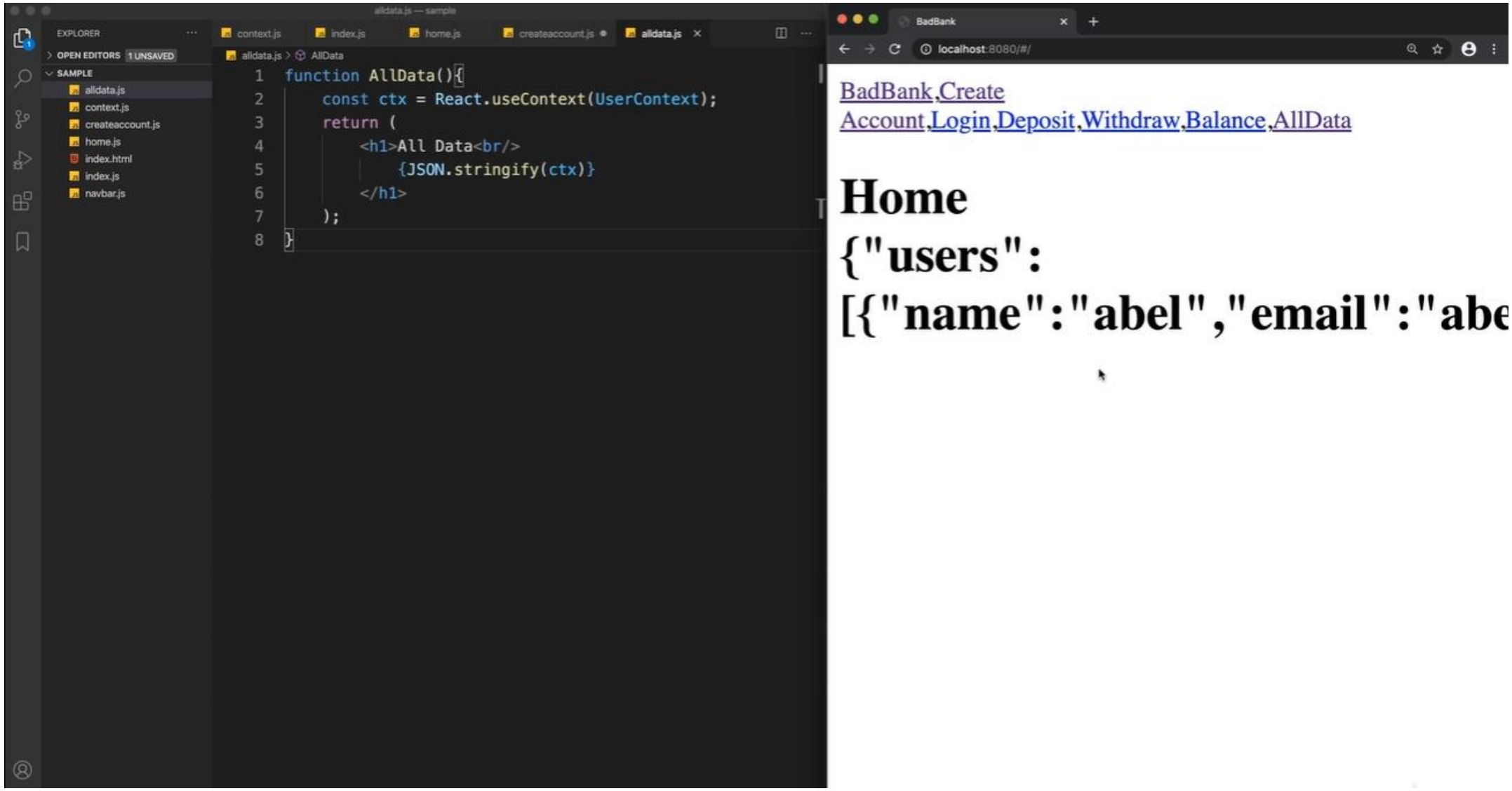
Shared Context: Demonstration (4/8)



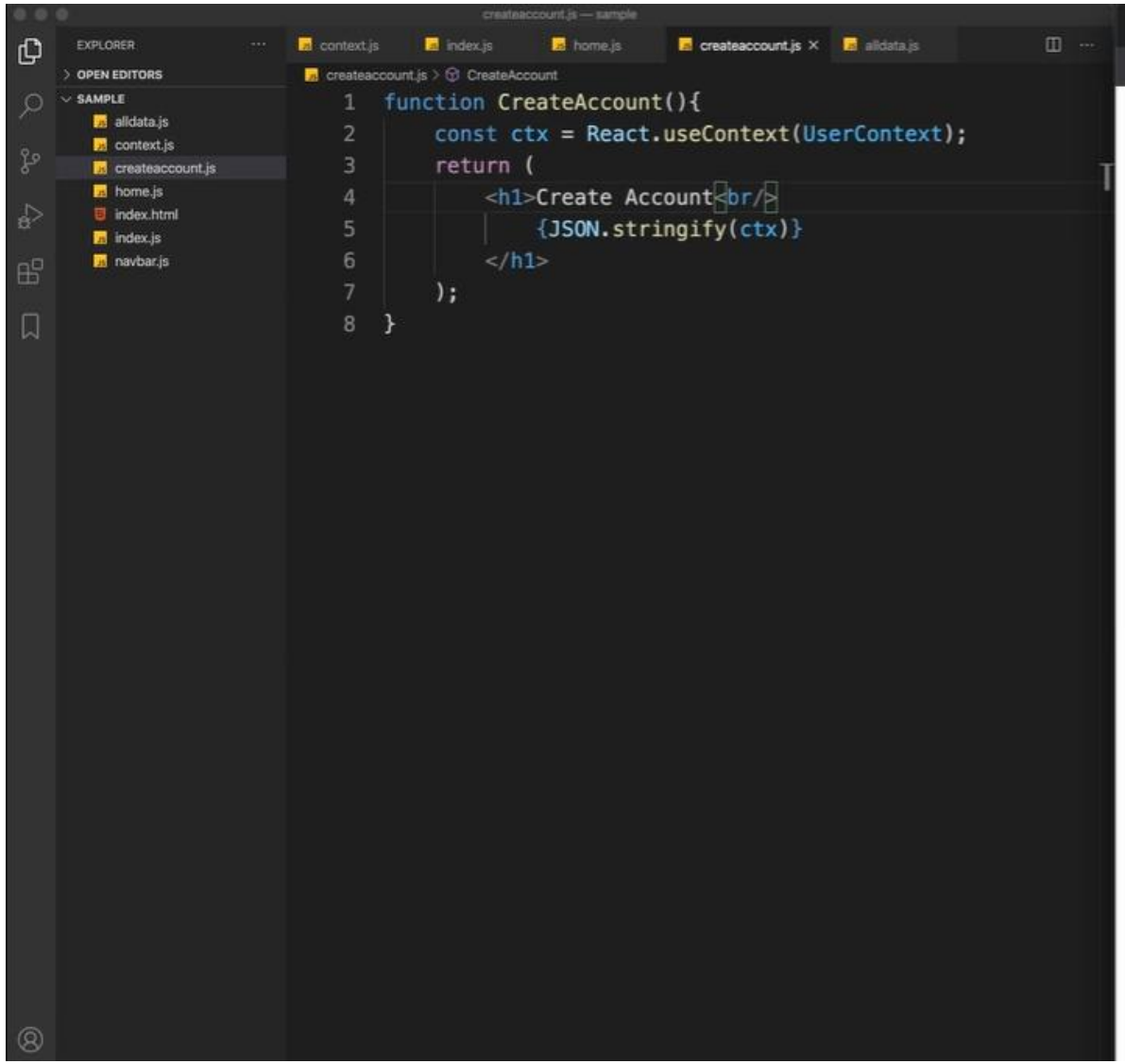
Shared Context: Demonstration (5/8)

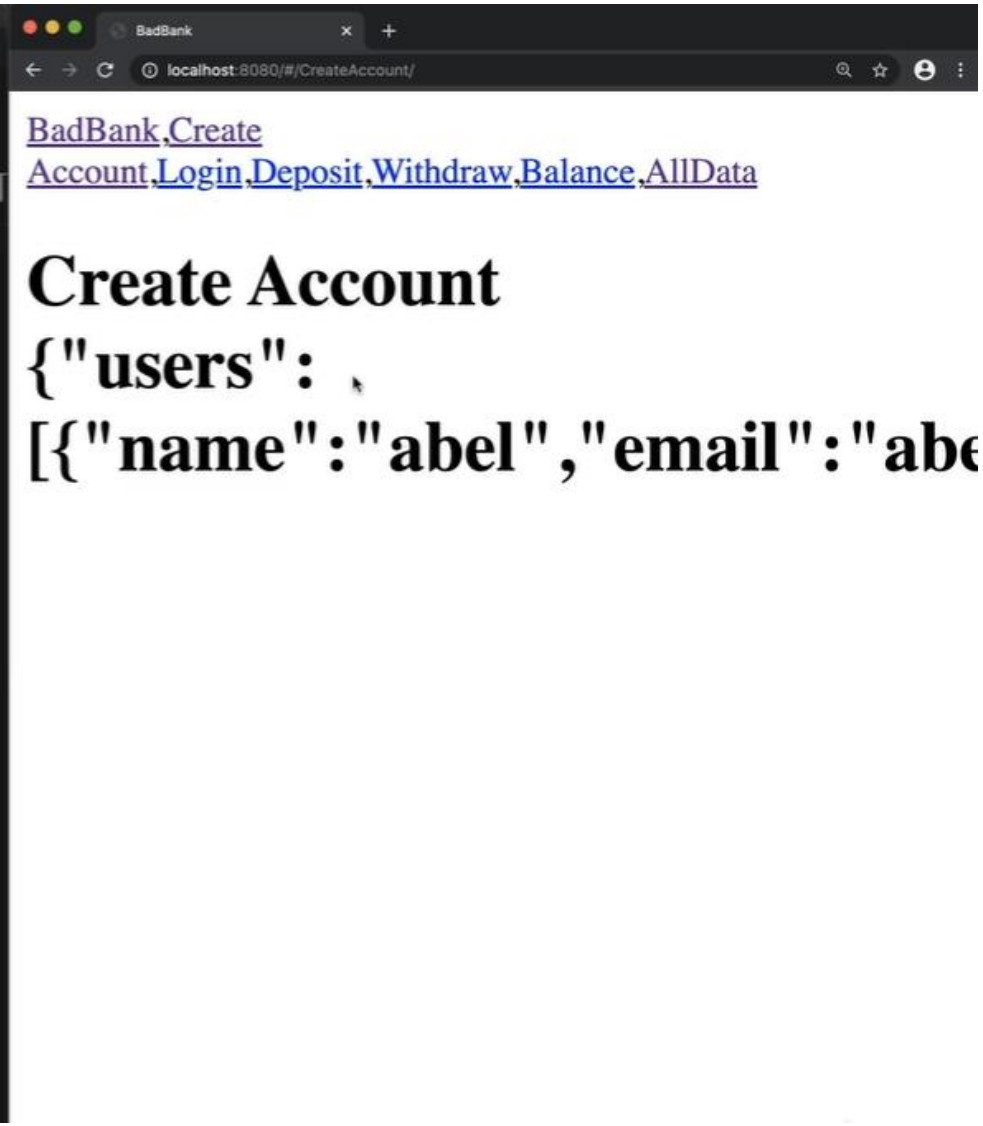


Shared Context: Demonstration (6/8)

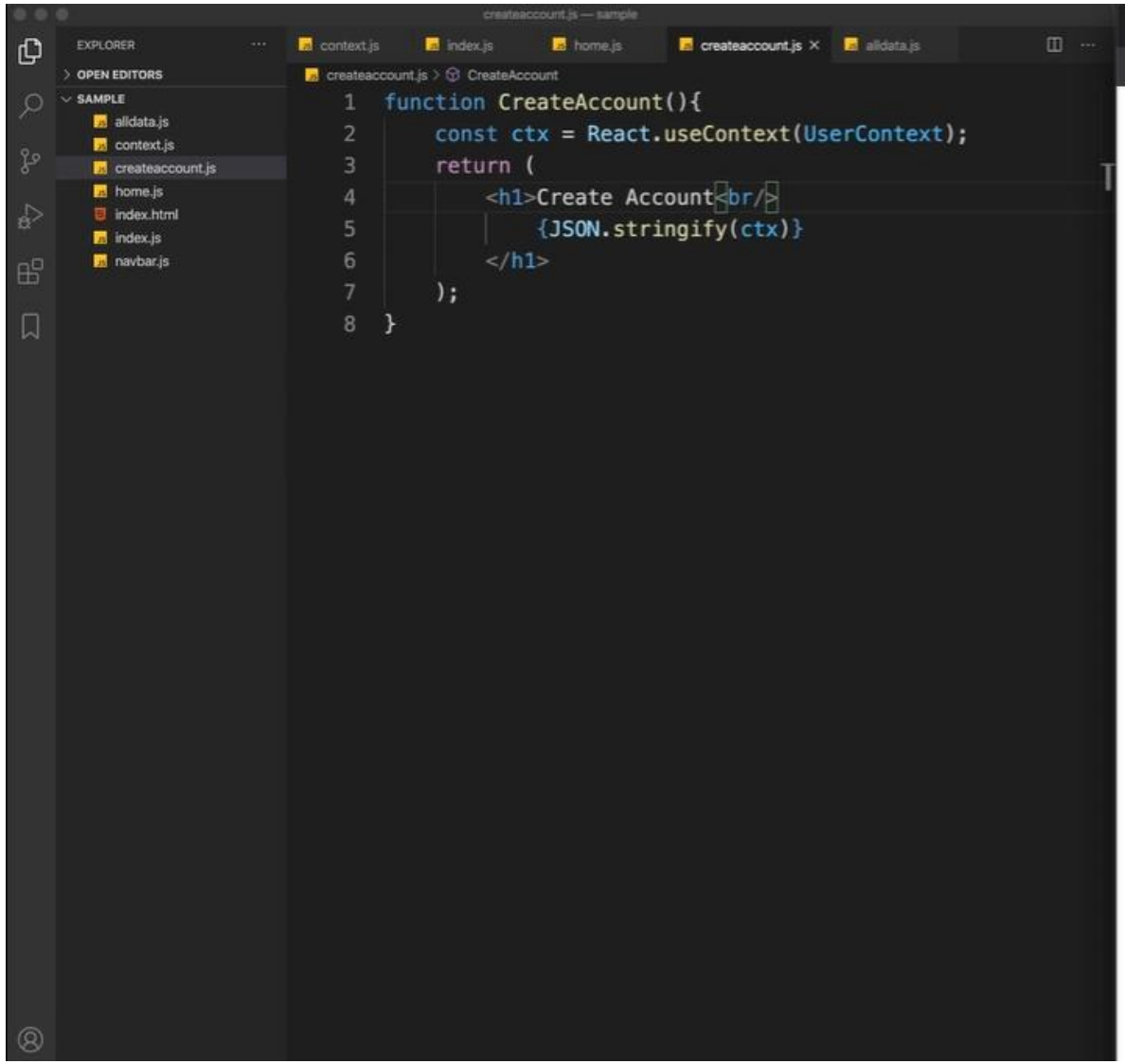


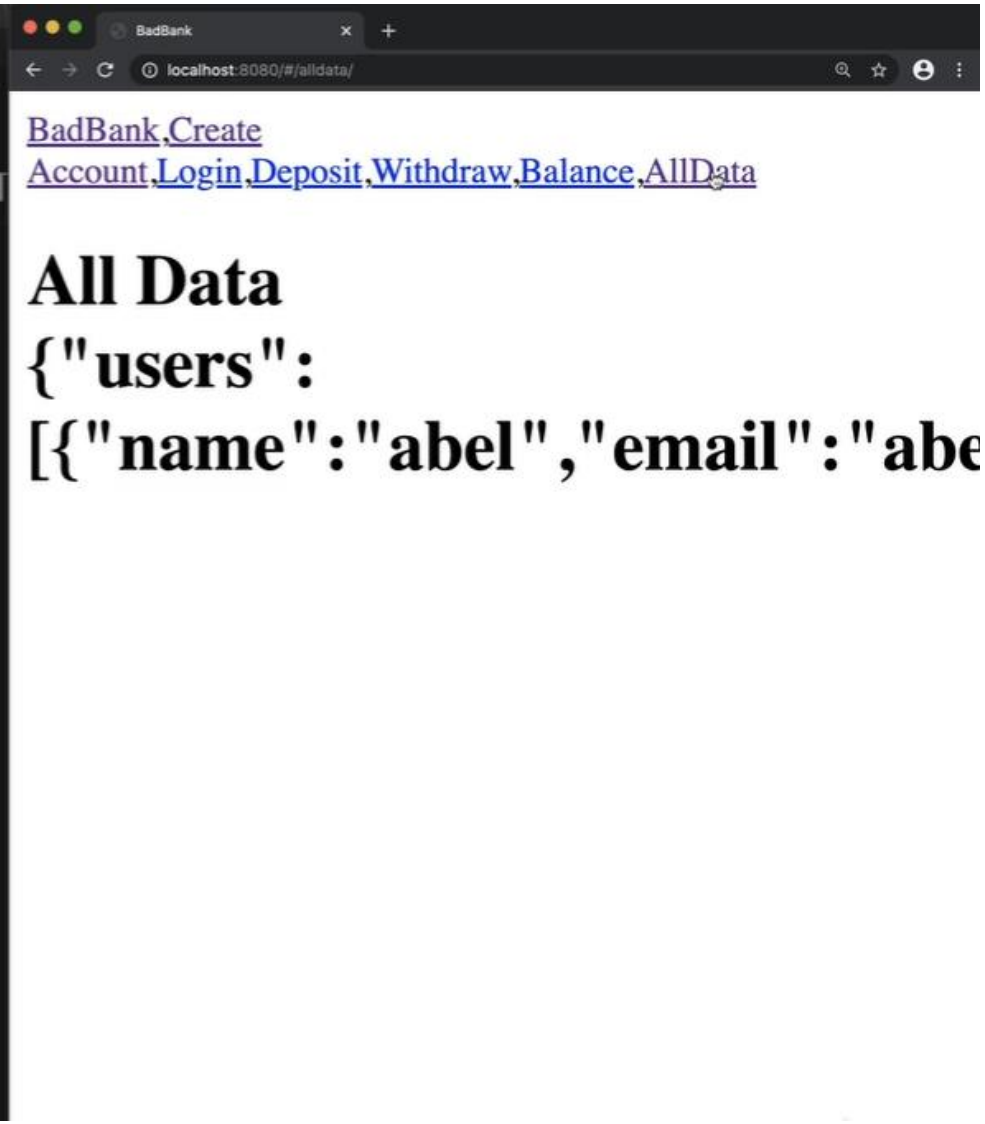
Shared Context: Demonstration (7/8)





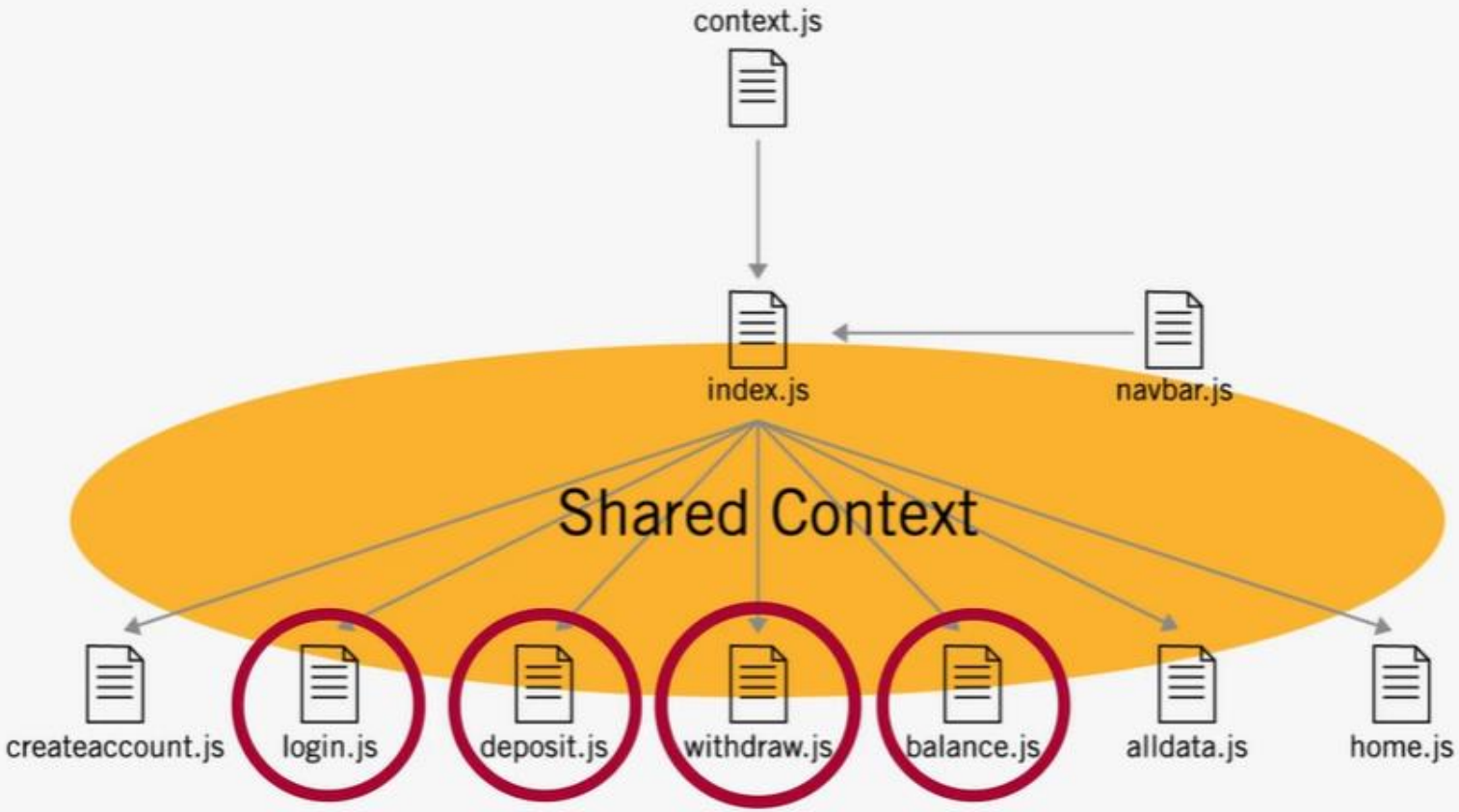
Shared Context: Demonstration (8/8)



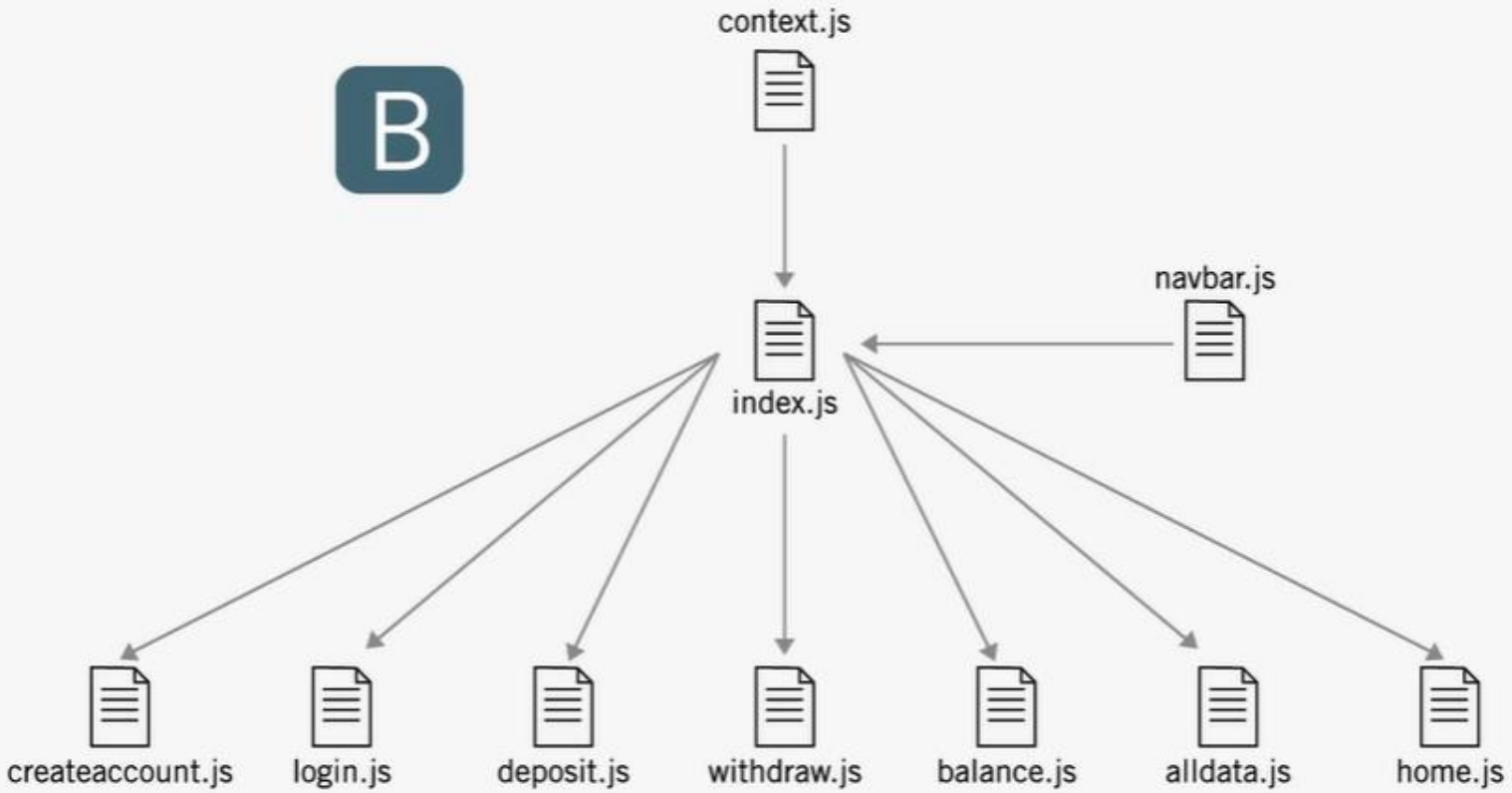


MIT xPRO

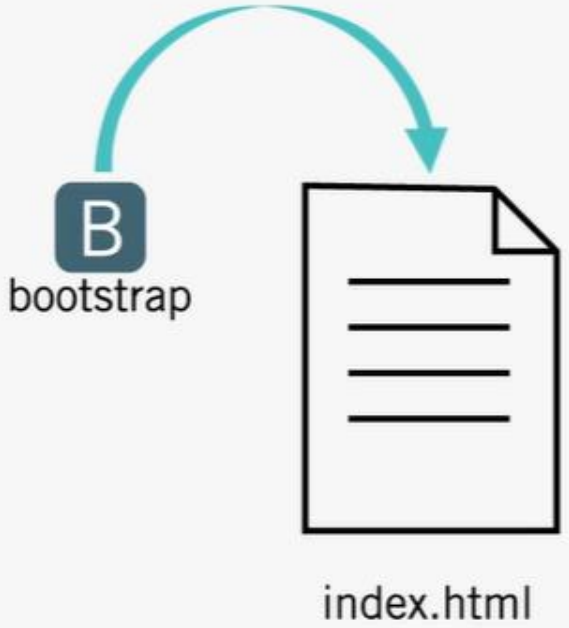
Shared Context: Your Turn



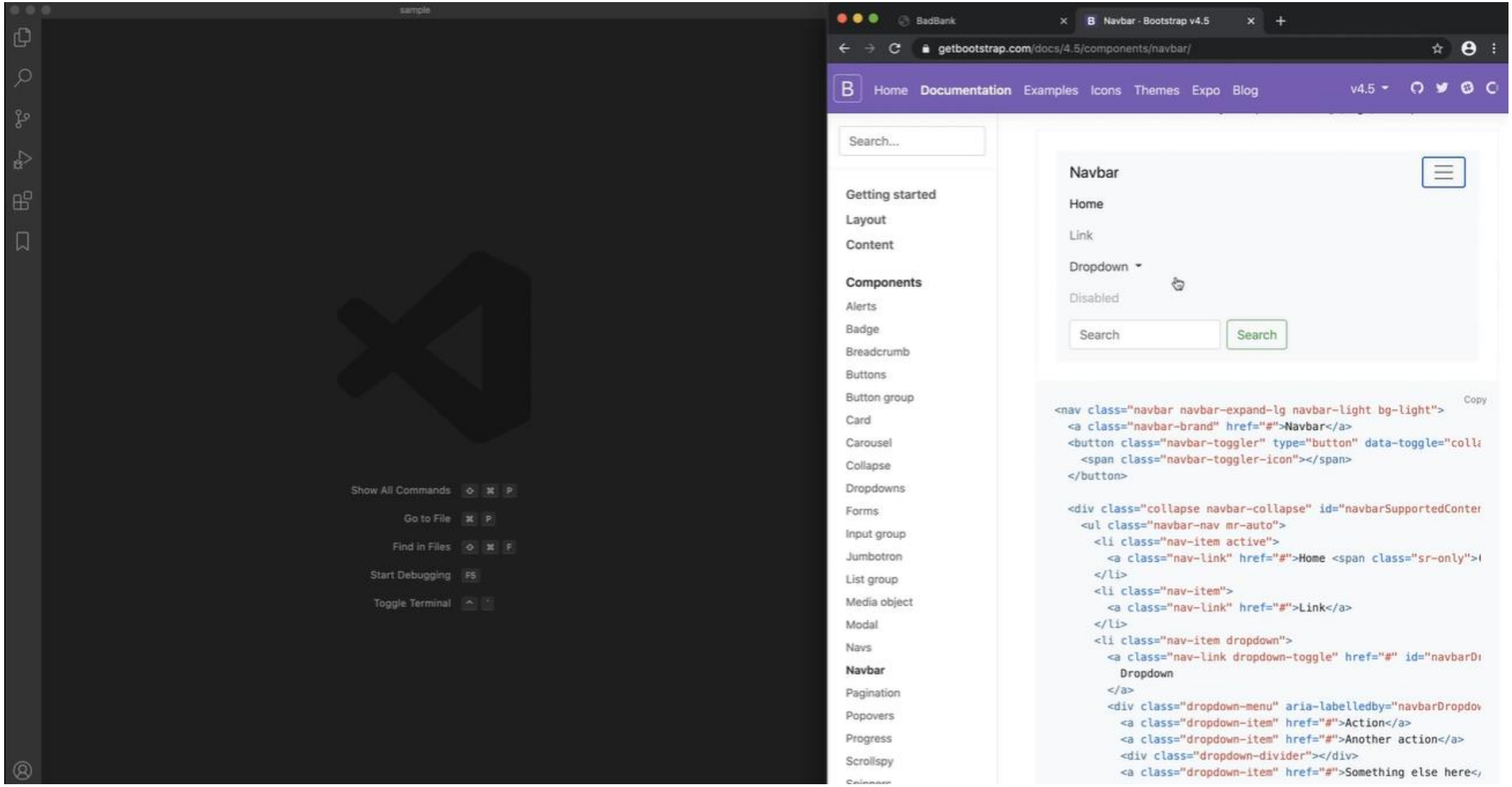
Styles – Bootstrap



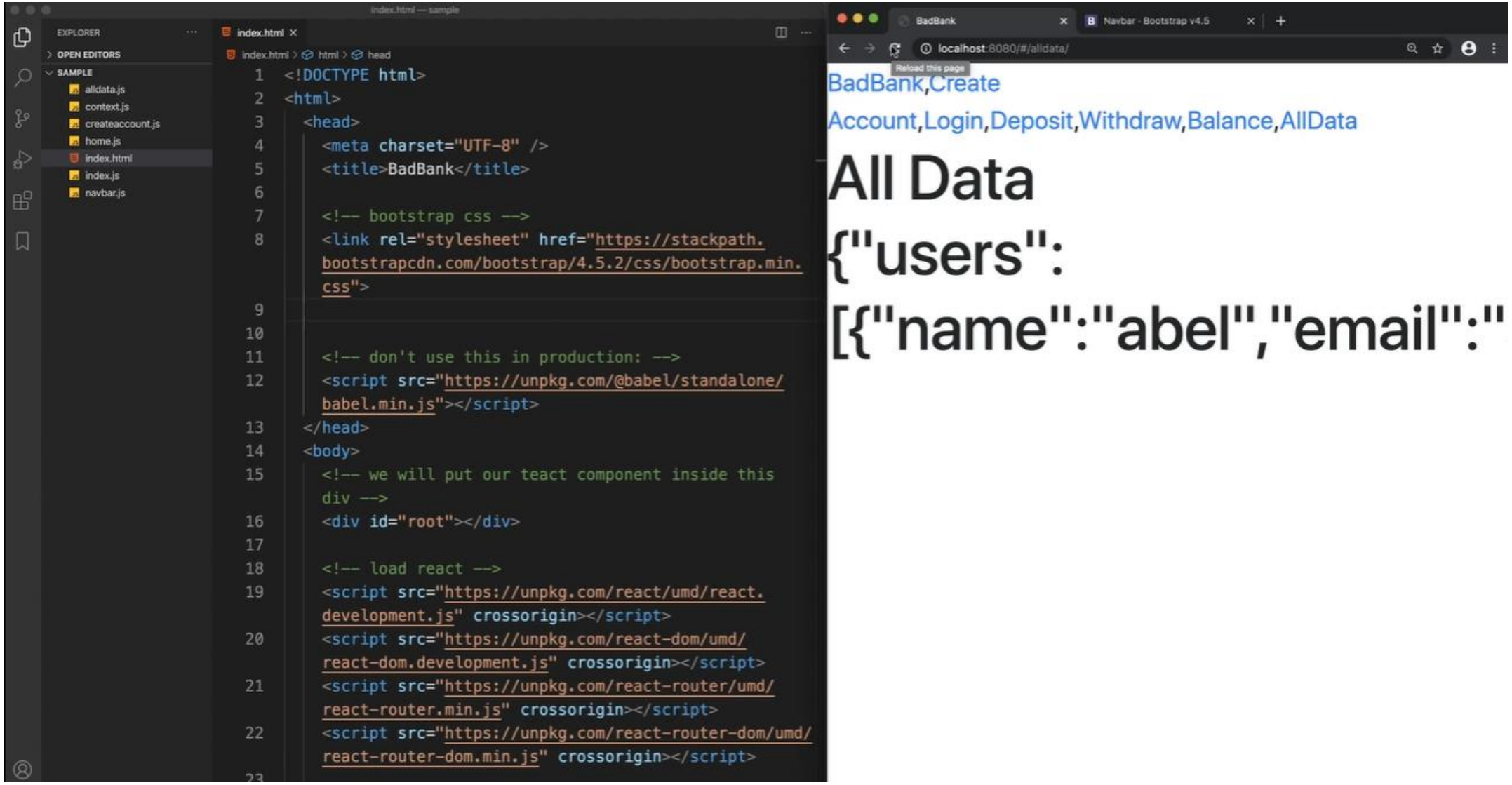
Add Bootstrap Styles



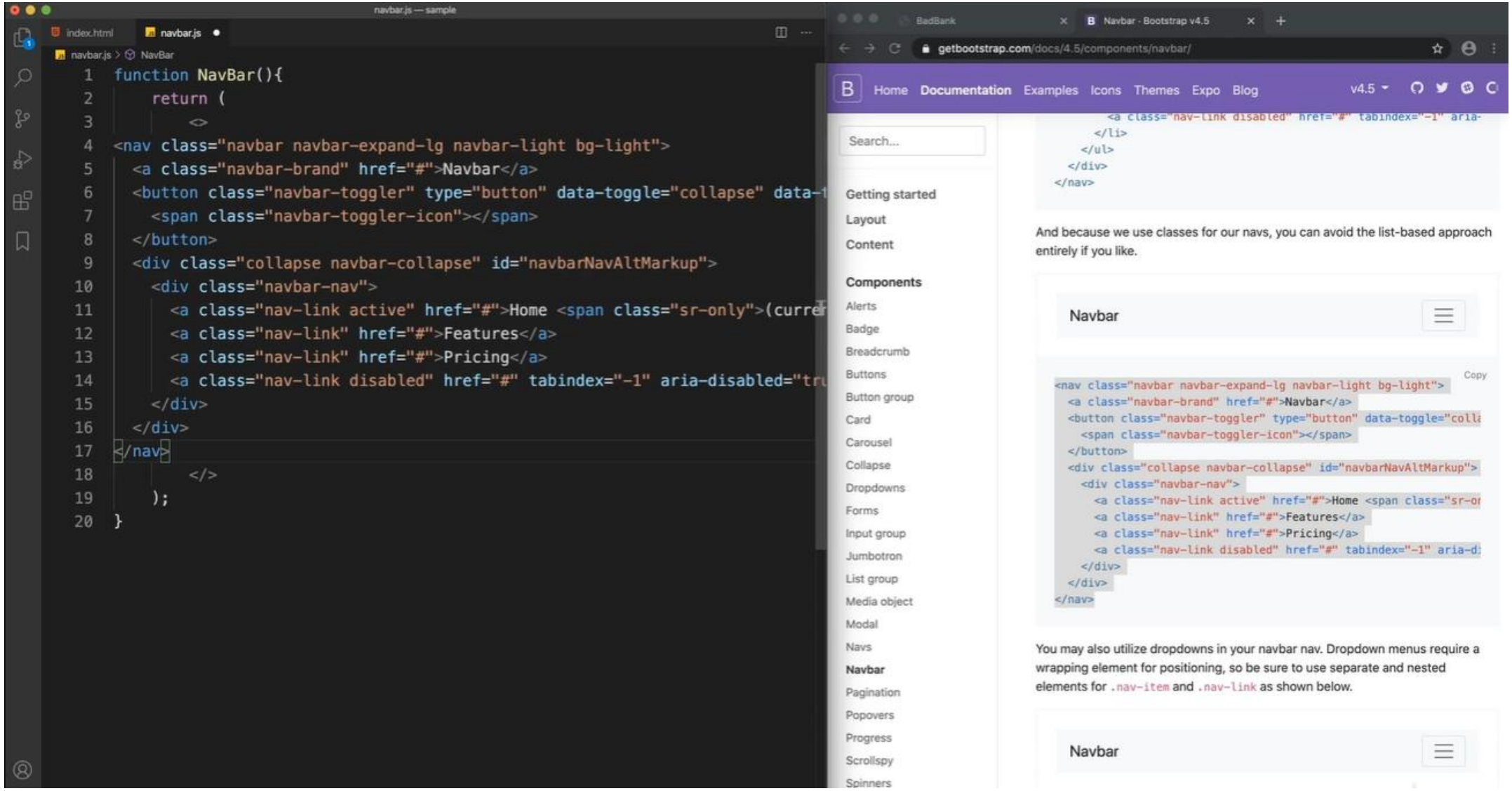
Add Bootstrap Styles: Demonstration (1/7)



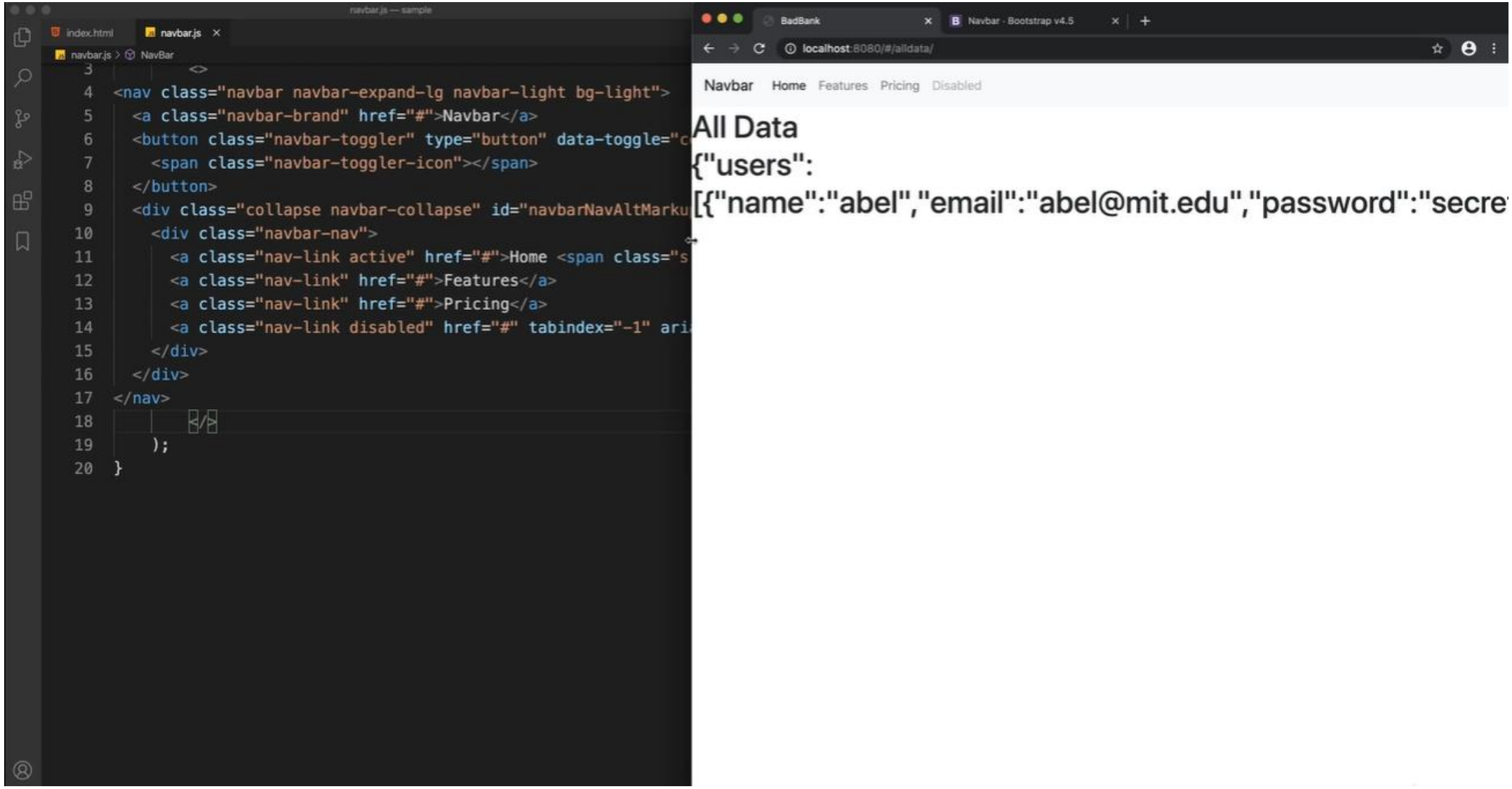
Add Bootstrap Styles: Demonstration (2/7)



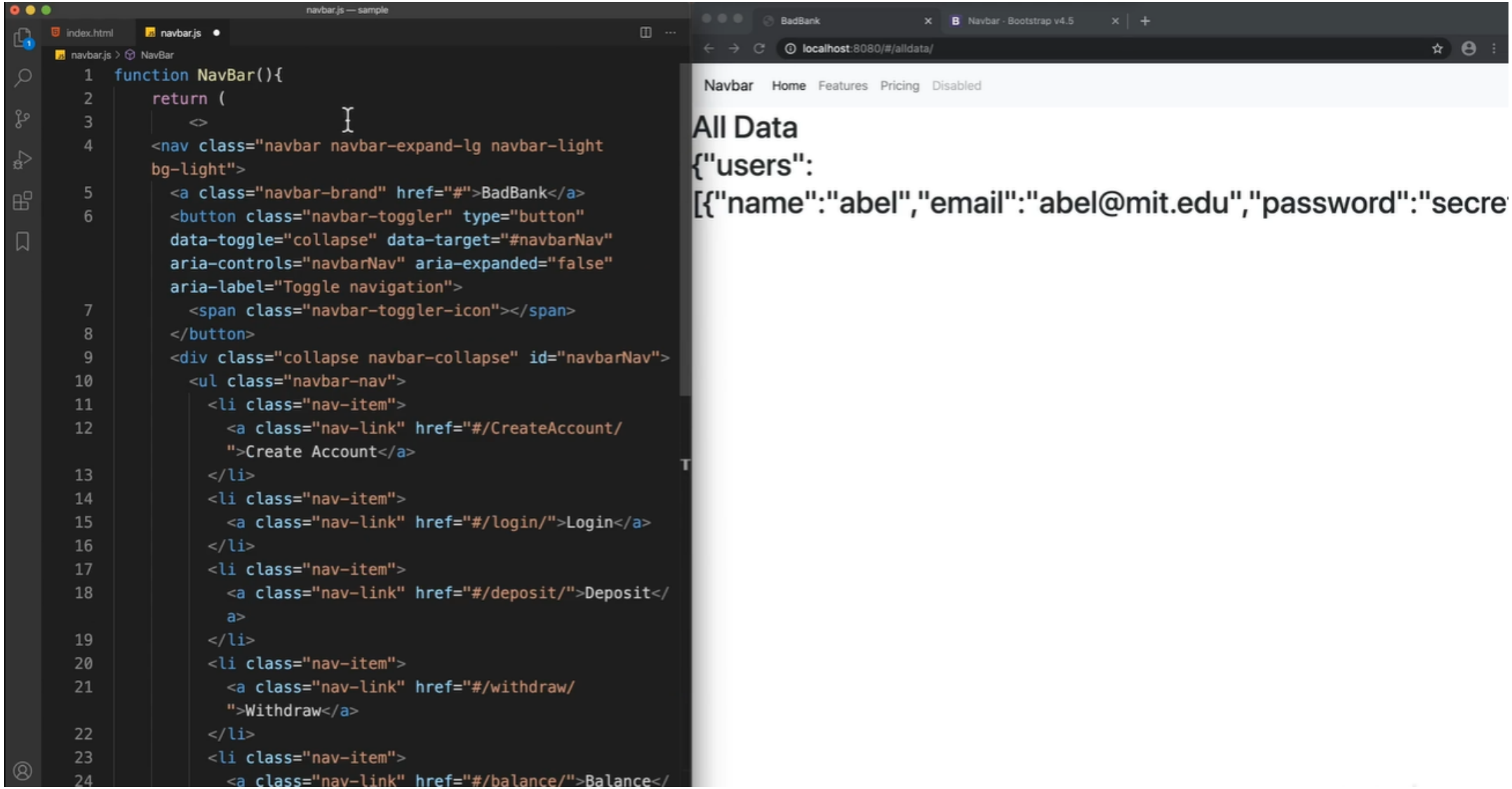
Add Bootstrap Styles: Demonstration (3/7)



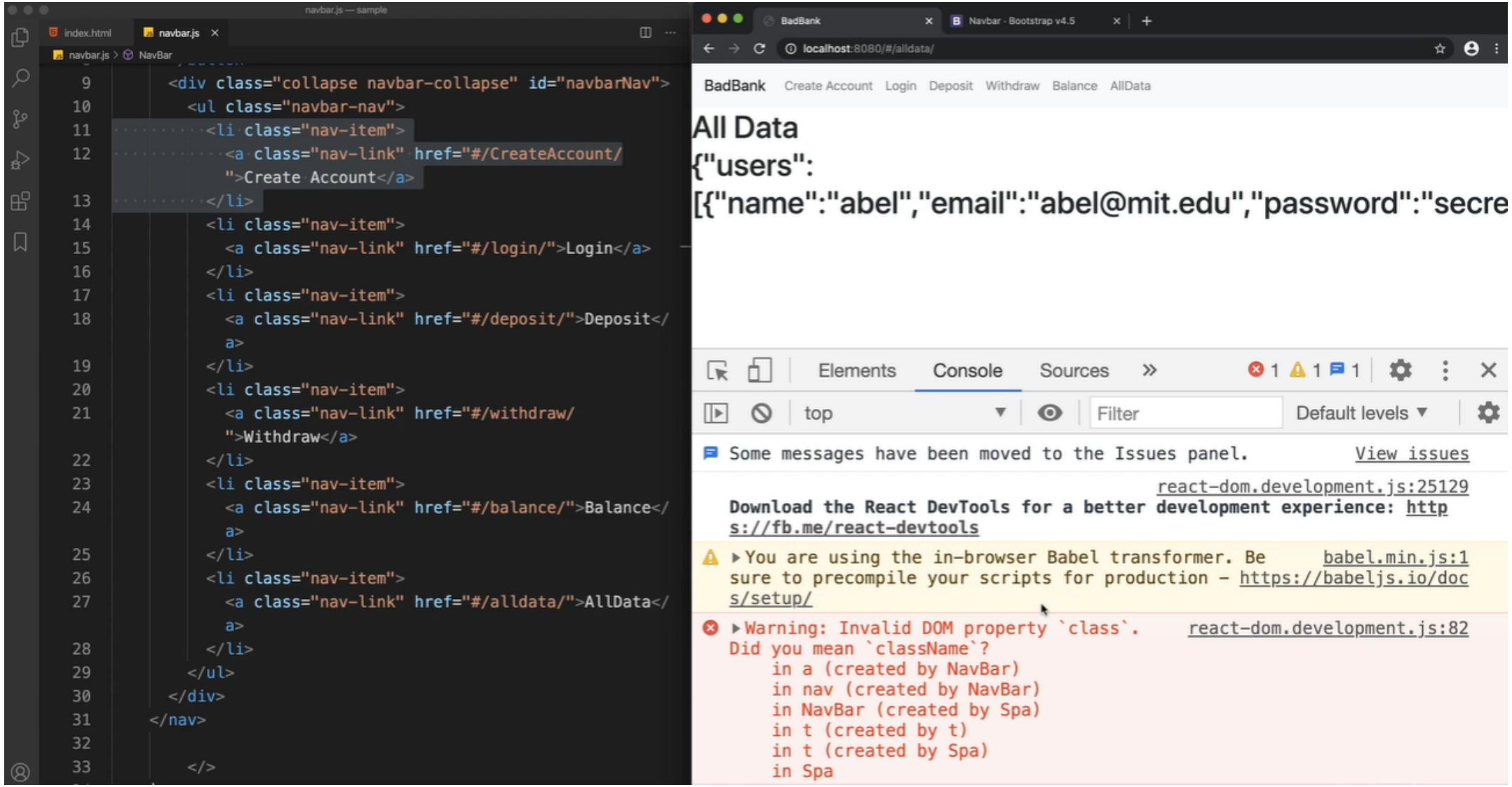
Add Bootstrap Styles: Demonstration (4/7)



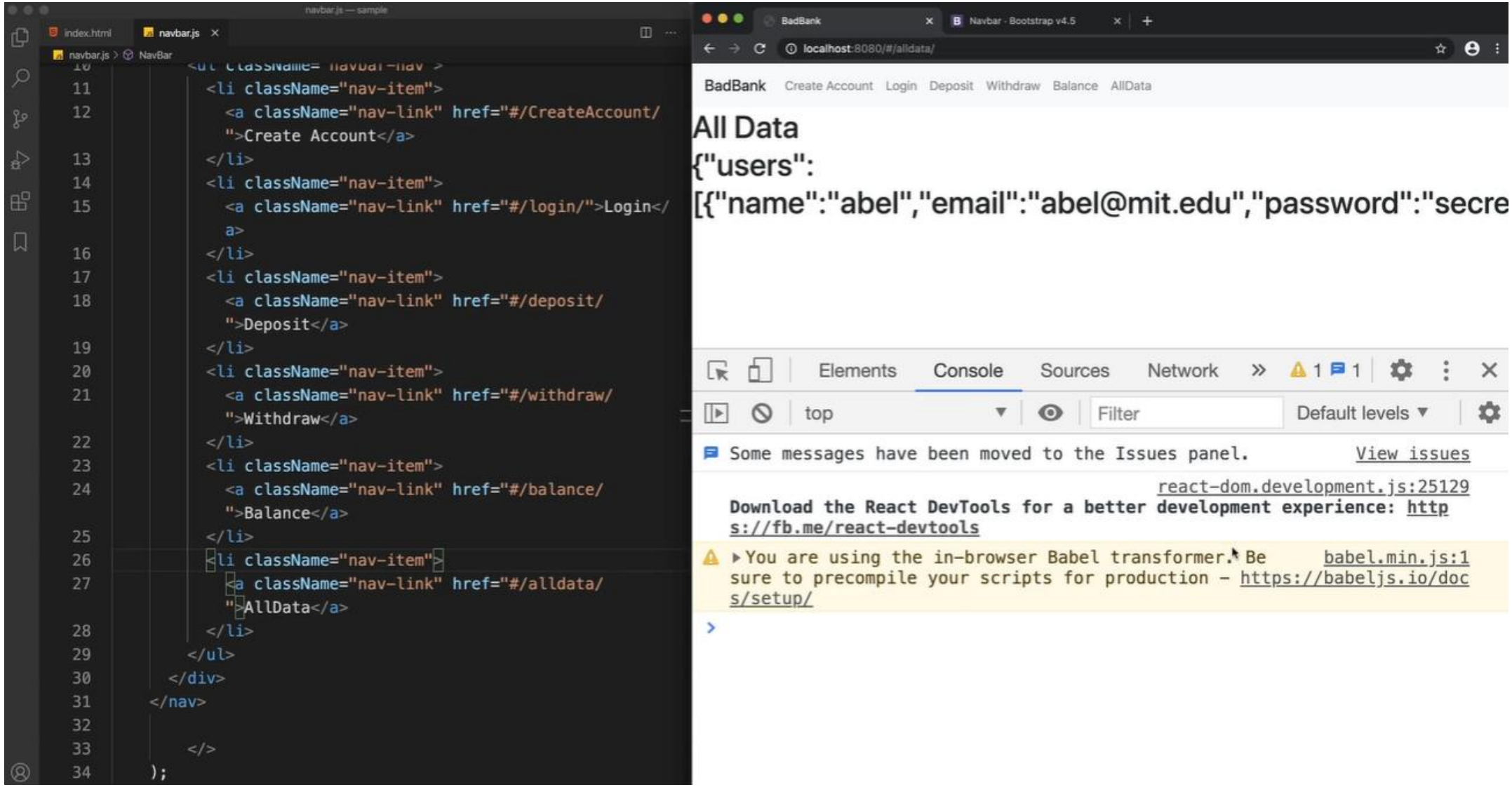
Add Bootstrap Styles: Demonstration (5/7)



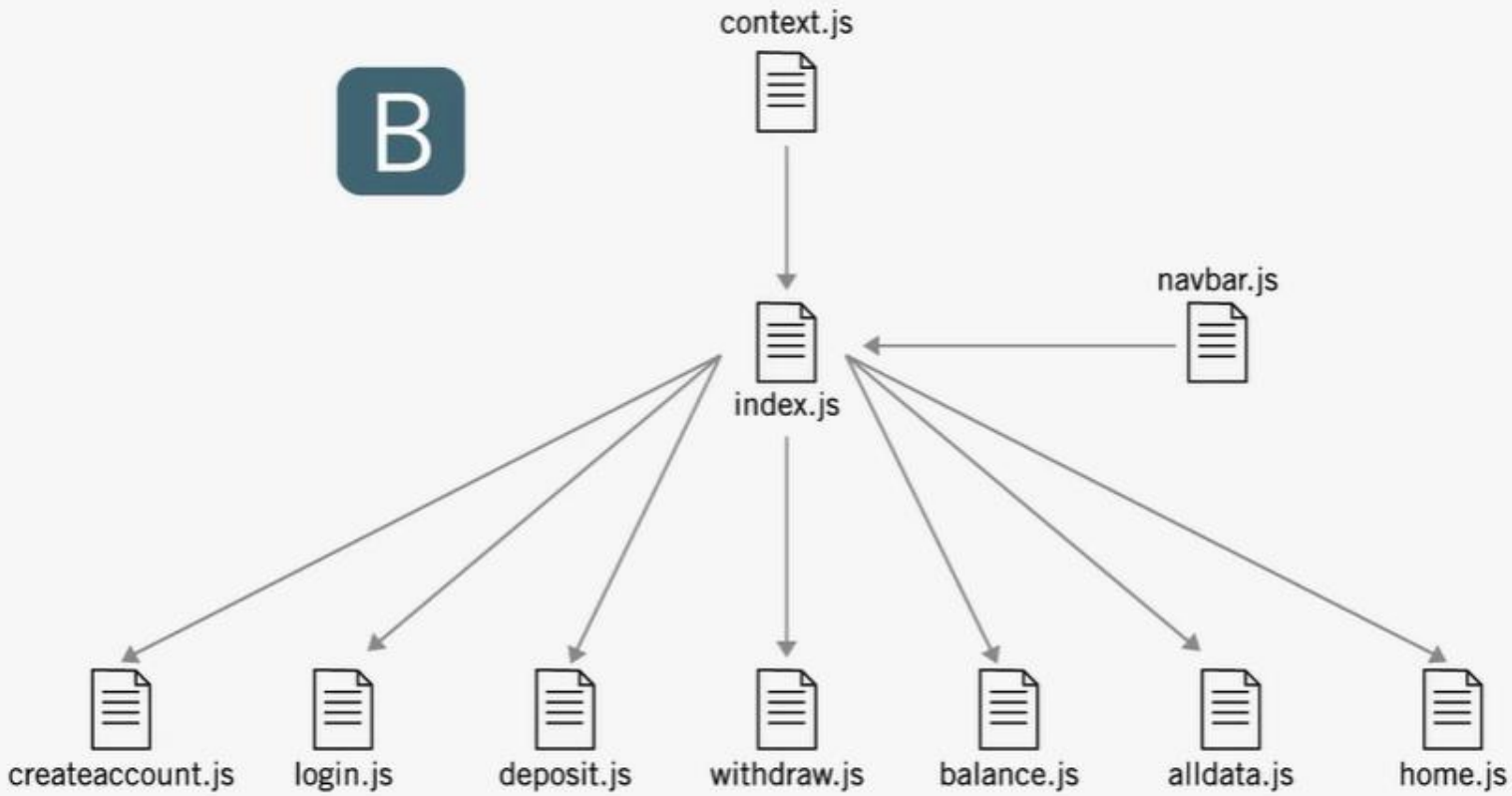
Add Bootstrap Styles: Demonstration (6/7)



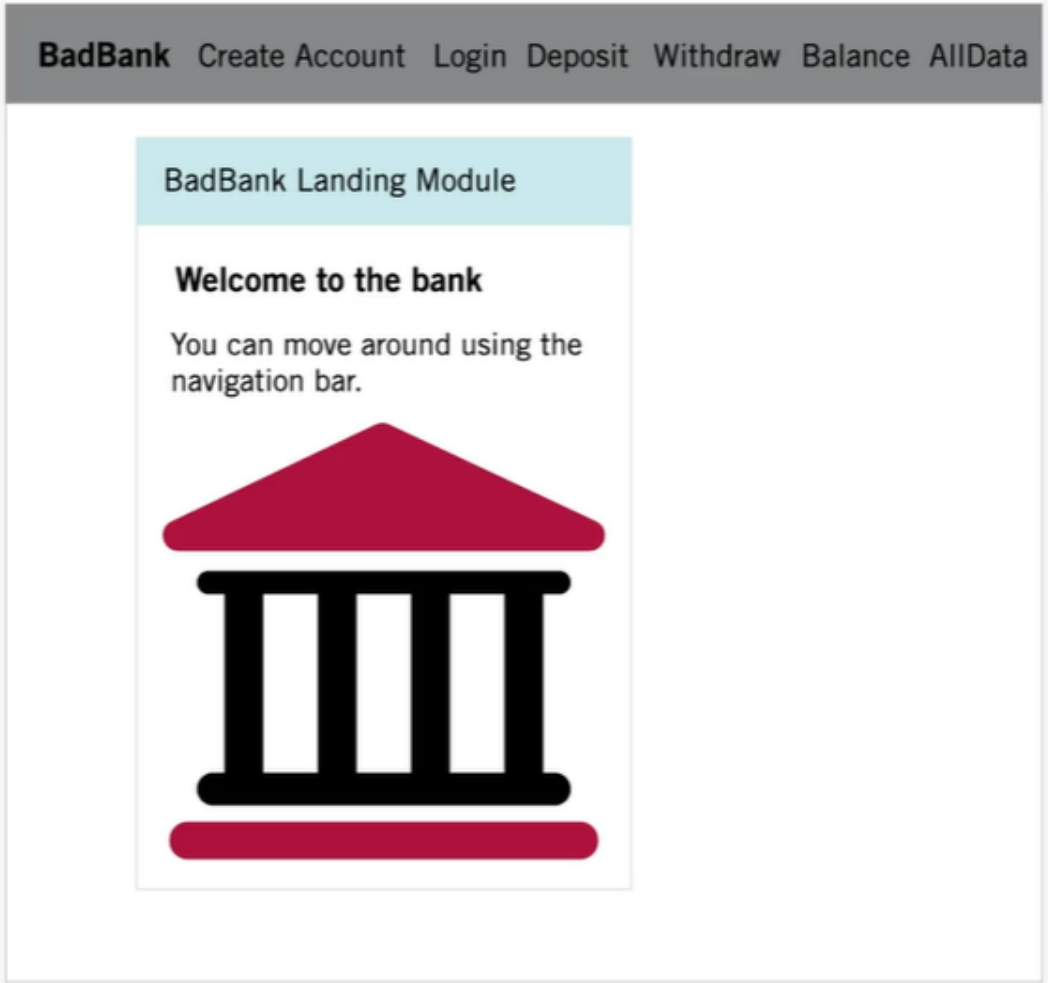
Add Bootstrap Styles: Demonstration (7/7)



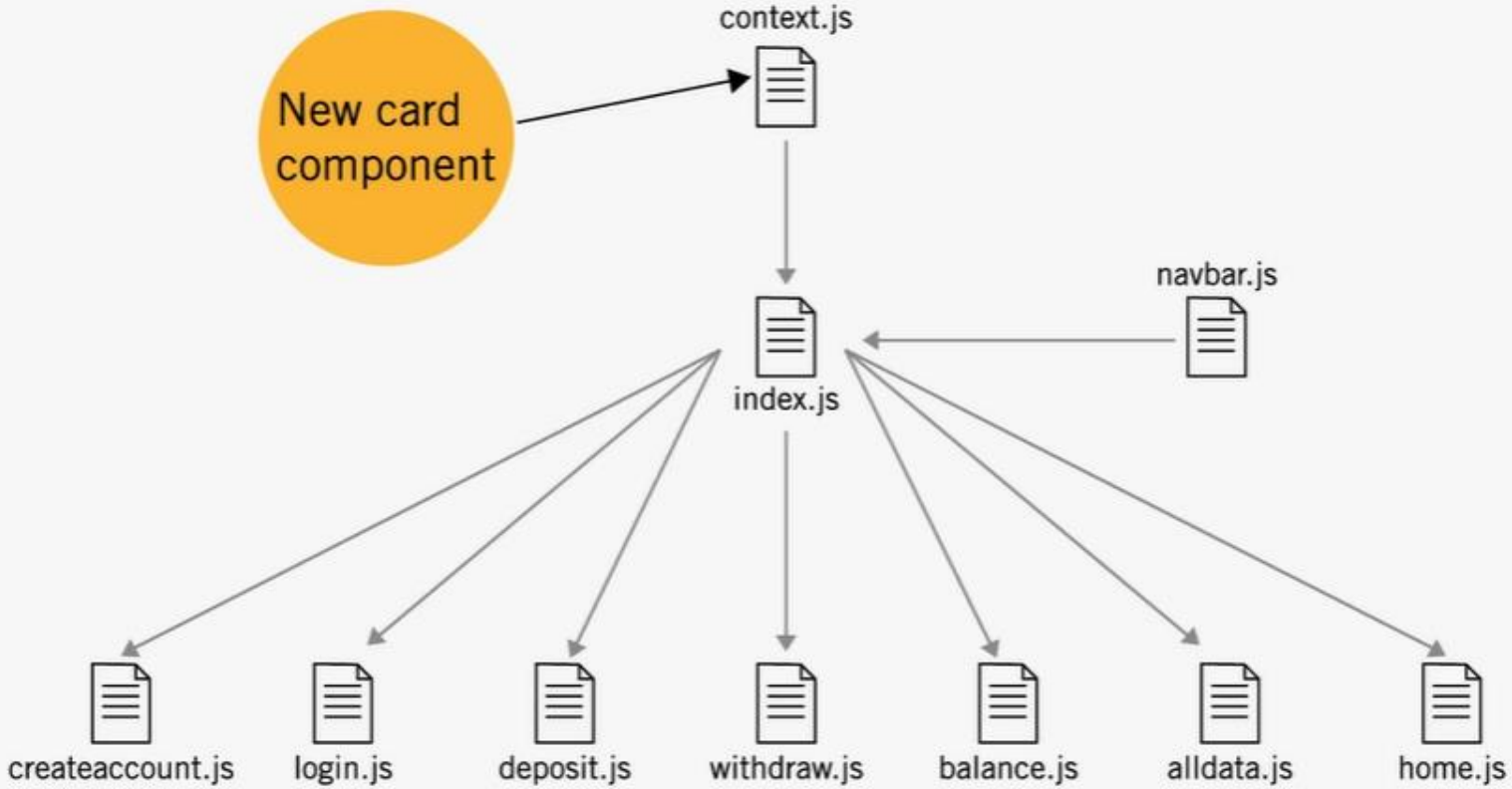
Add Bootstrap Styles: Your Turn



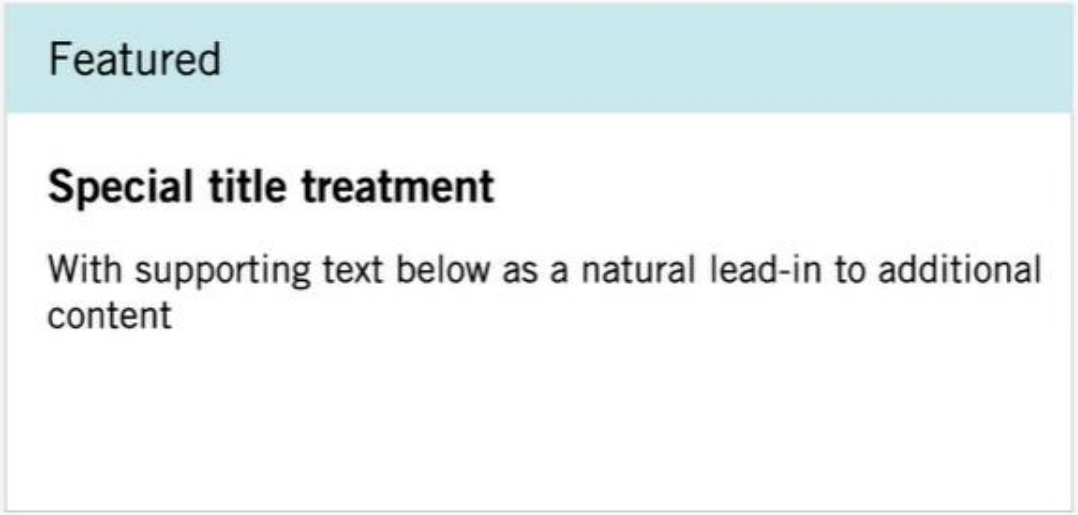
Bootstrap Style – Card



Styles – Bootstrap



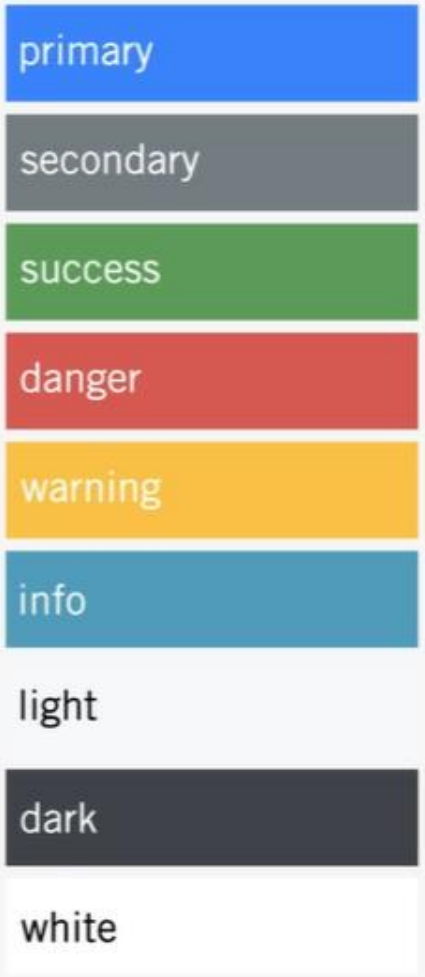
Card – For All Components



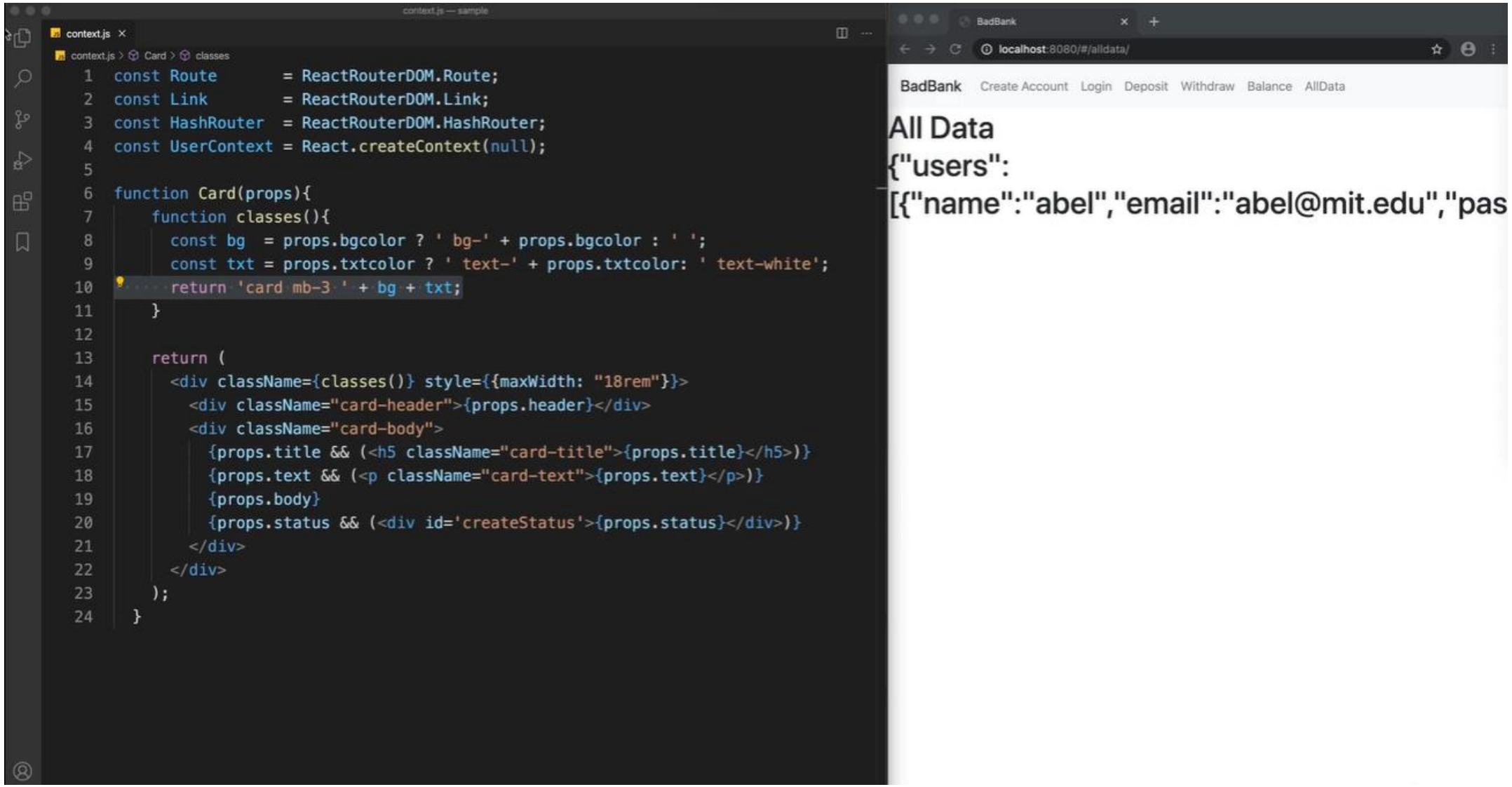
Syntax Components

```
<div class="card">
    <div class="card-header">
        Featured
    </div>
    <div class="card-body">
        <h5 class="card-title">Special title treatment</h5>
        <p class="card-title">With supporting text below as
        a natural lead-in to additional content.</p>
</div>
</div>
```

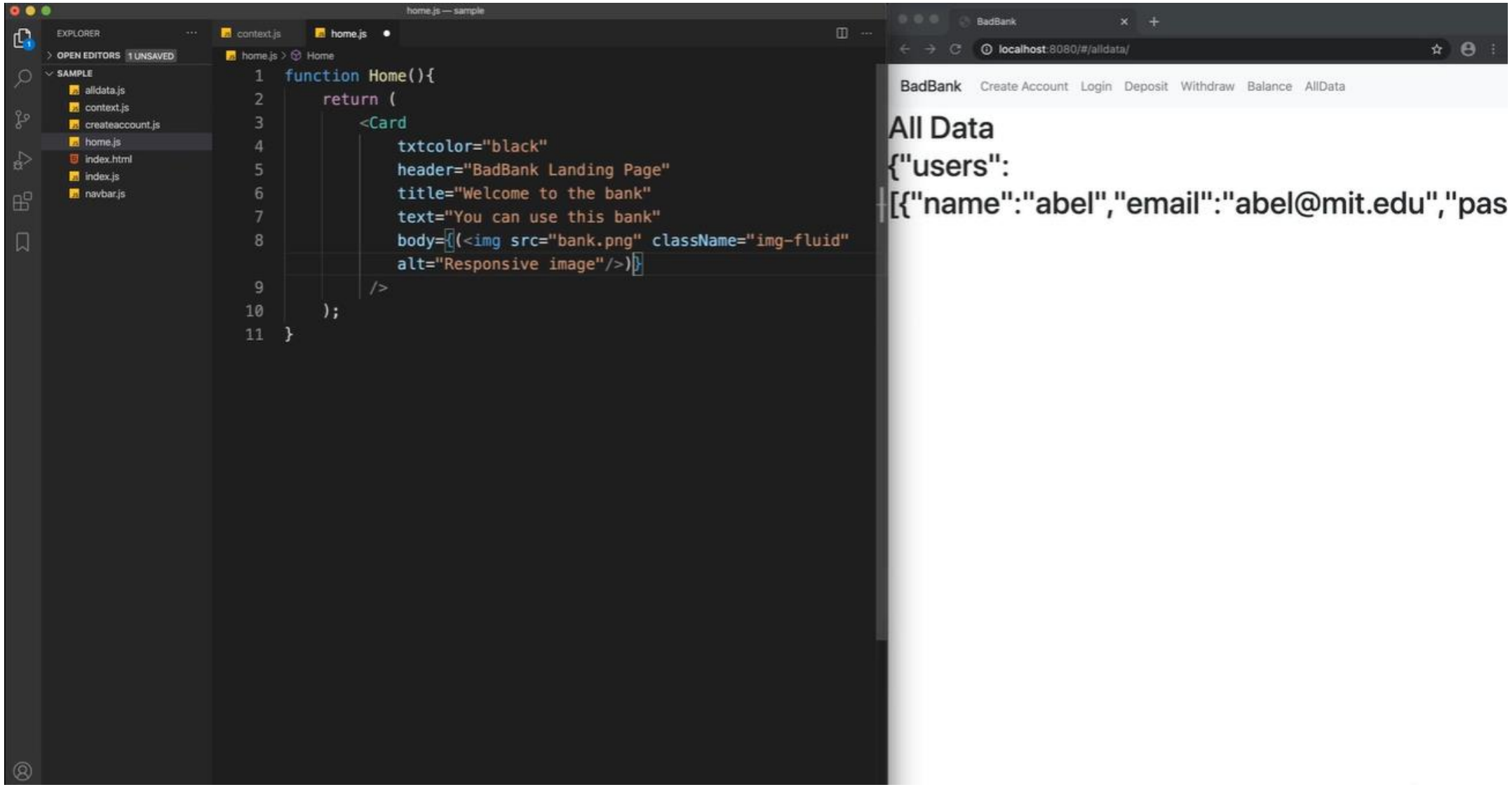
Colors


primary
secondary
success
danger
warning
info
light
dark
white

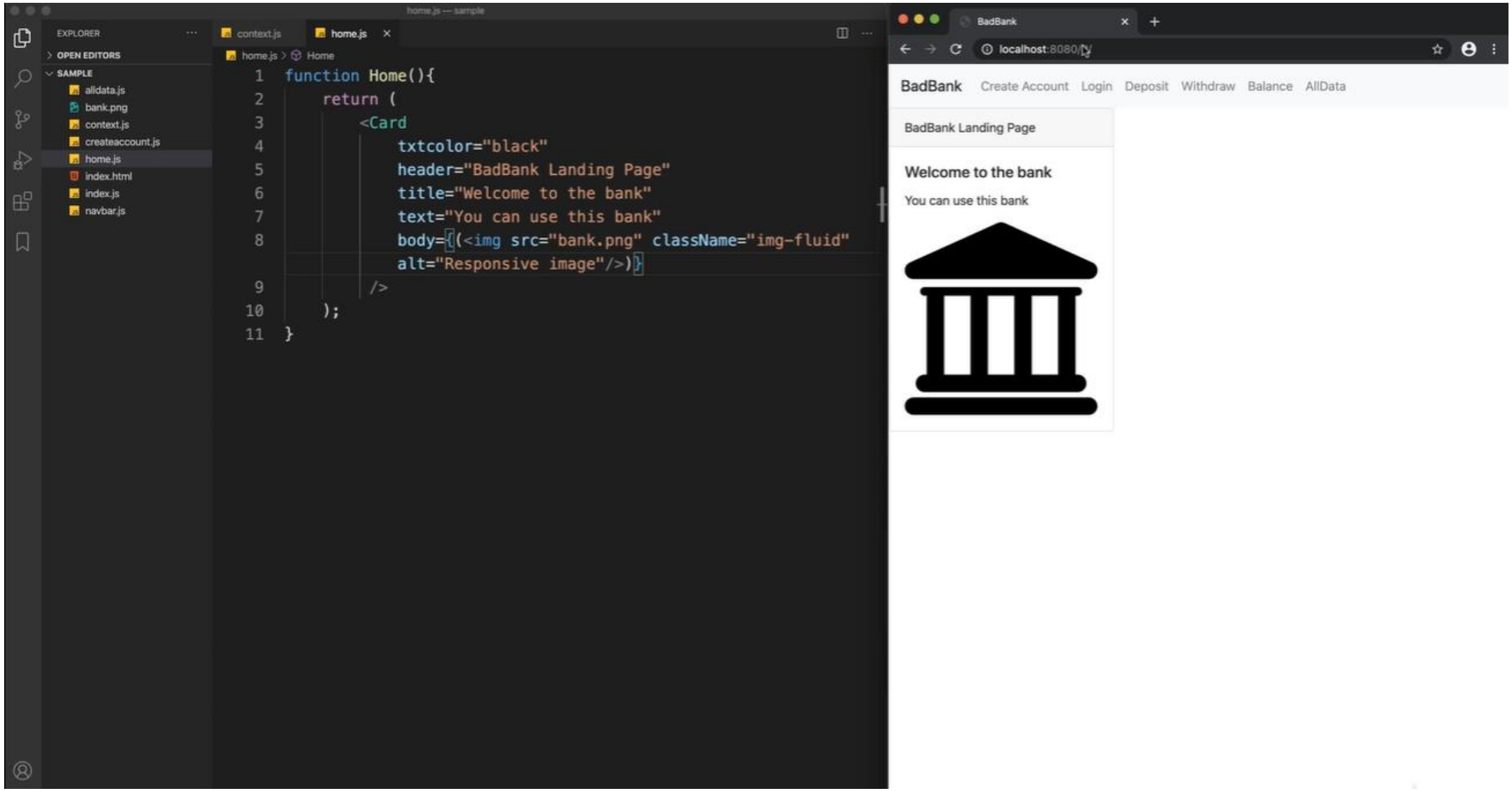
Bootstrap Style – Card: Demonstration (1/5)



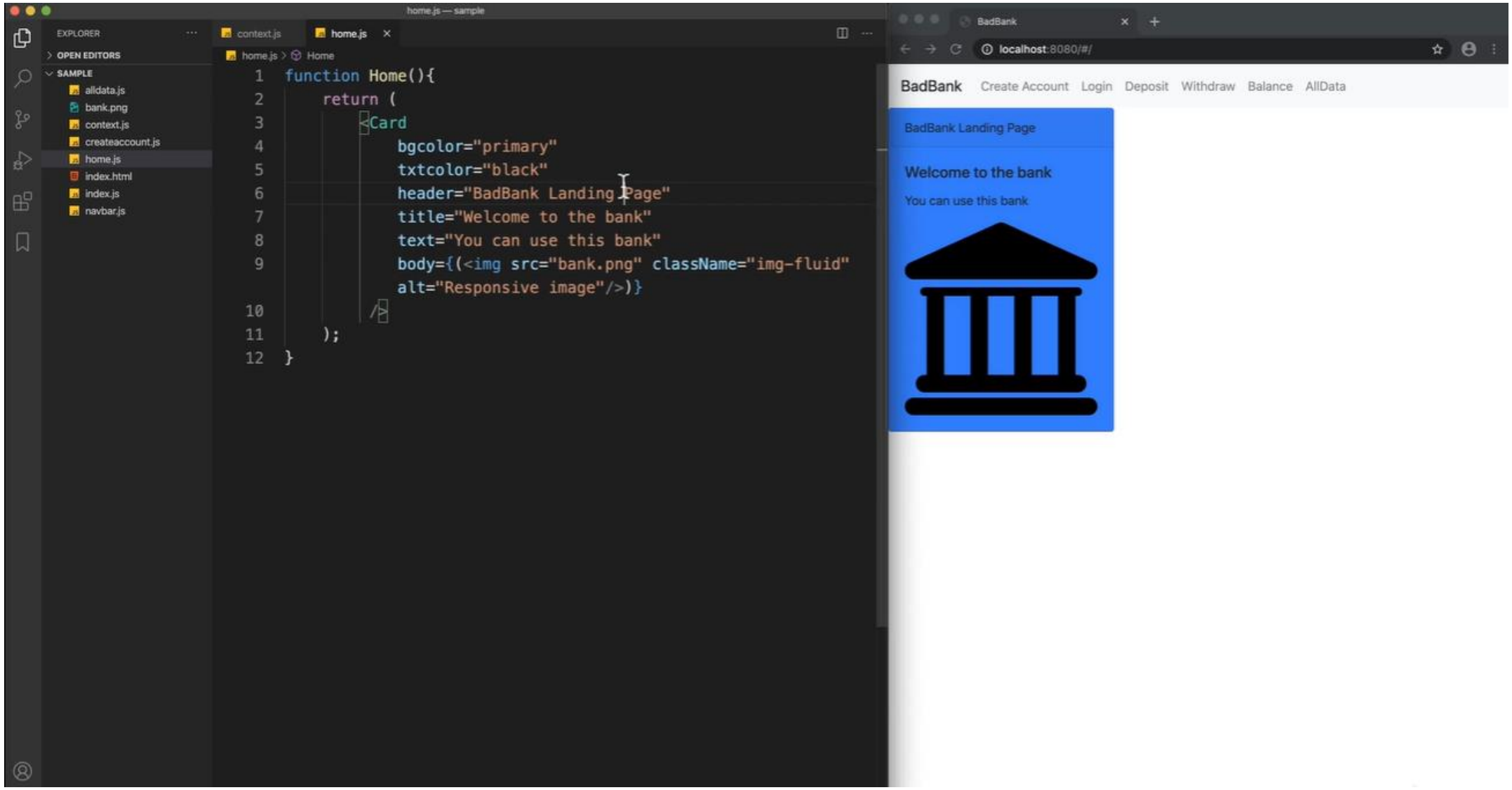
Bootstrap Style – Card: Demonstration (2/5)



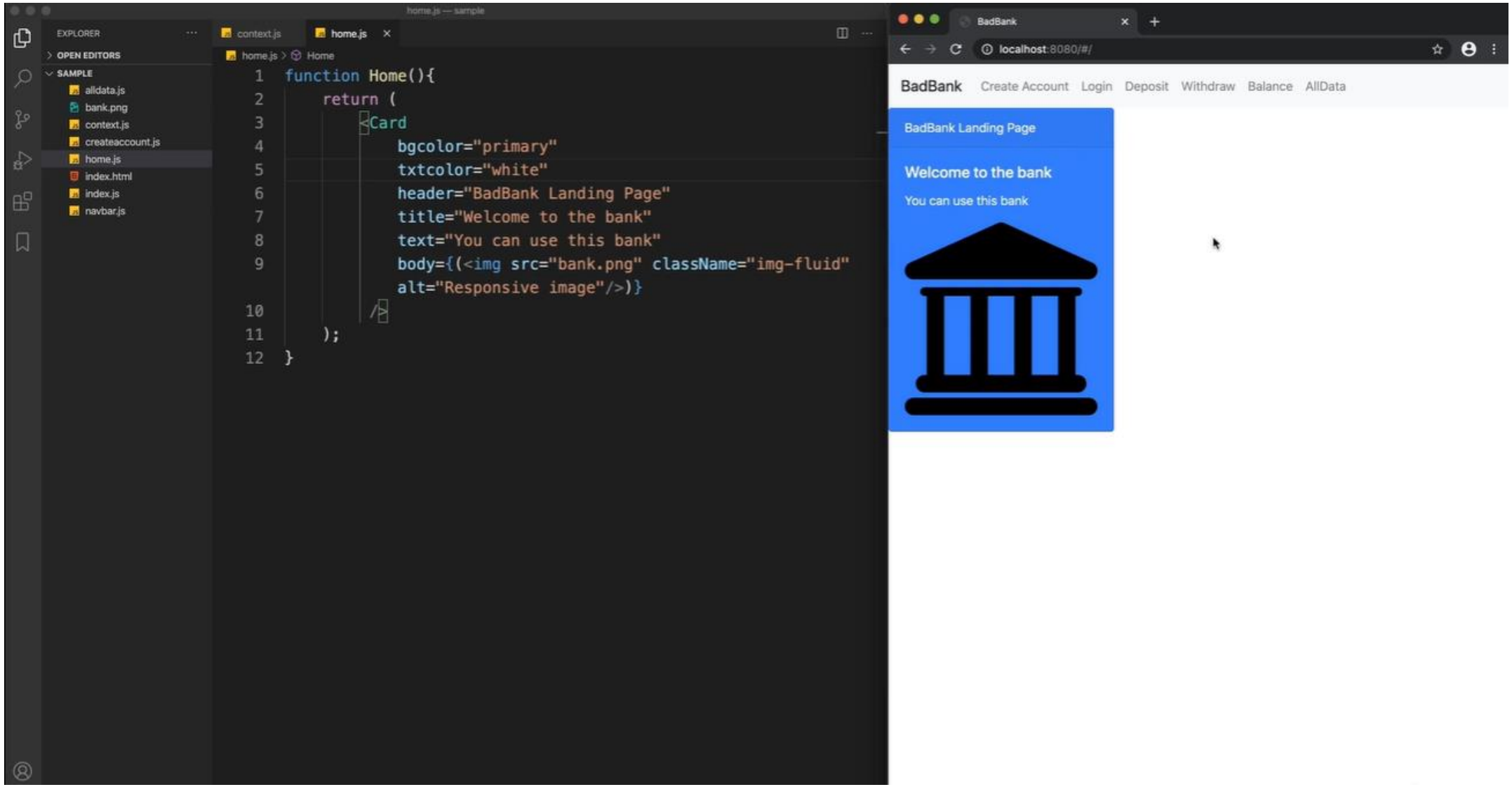
Bootstrap Style – Card: Demonstration (3/5)



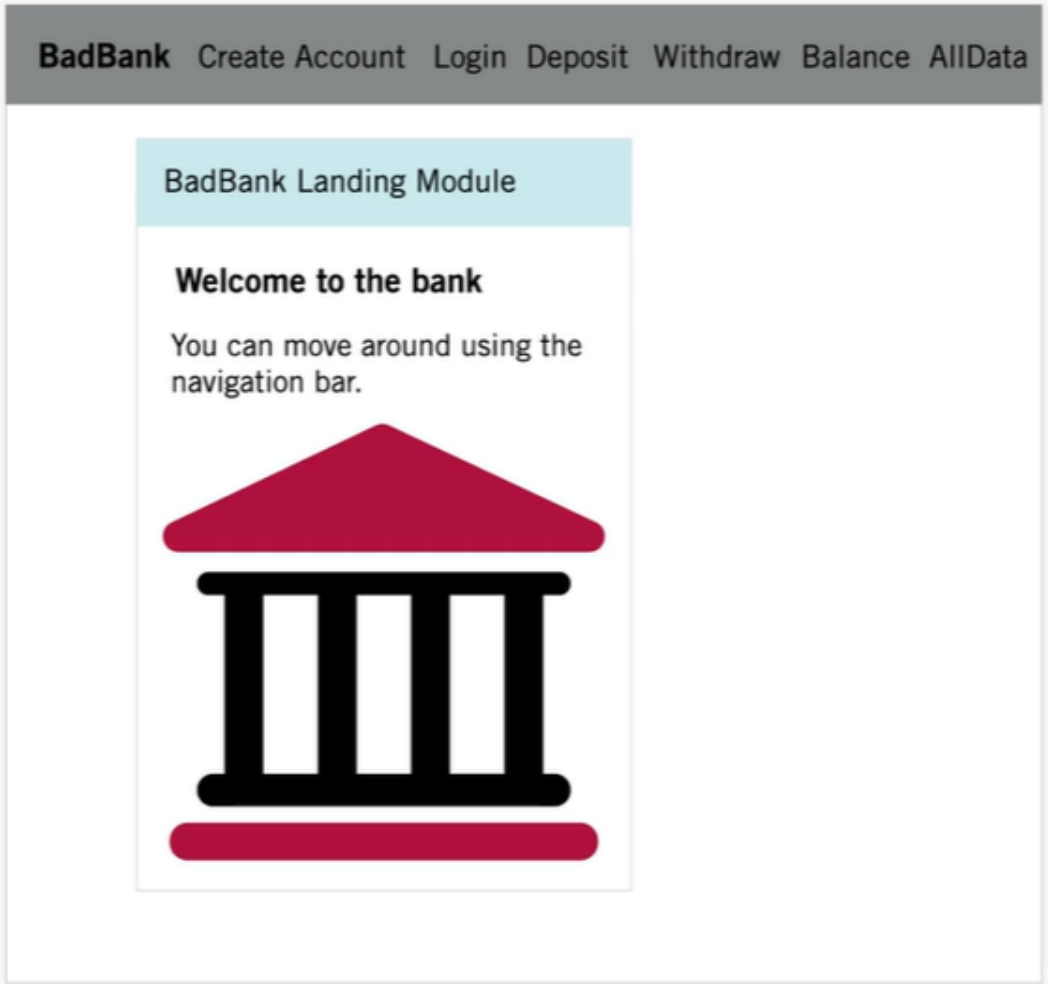
Bootstrap Style – Card: Demonstration (4/5)



Bootstrap Style – Card: Demonstration (5/5)



Bootstrap Style – Card: Your Turn



Code Blocks


State: status, name, email, password, UserContext
Event: handleCreate, validate, clearform
show
if
Fields: styling, value, onChange event
Button: handle event
else
Fields: styling, value, onChange event
Button: handle event

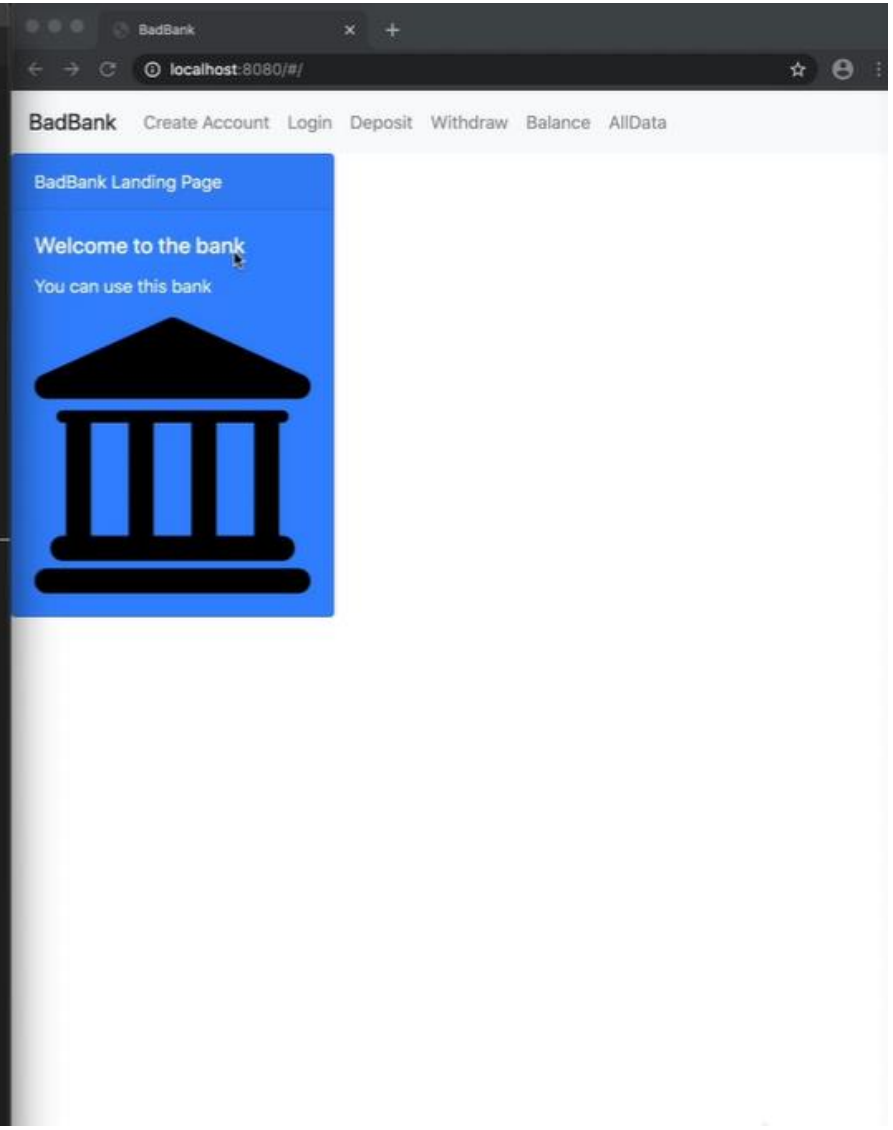
Creating Interactive Form (1/8)

createaccount.js — sample

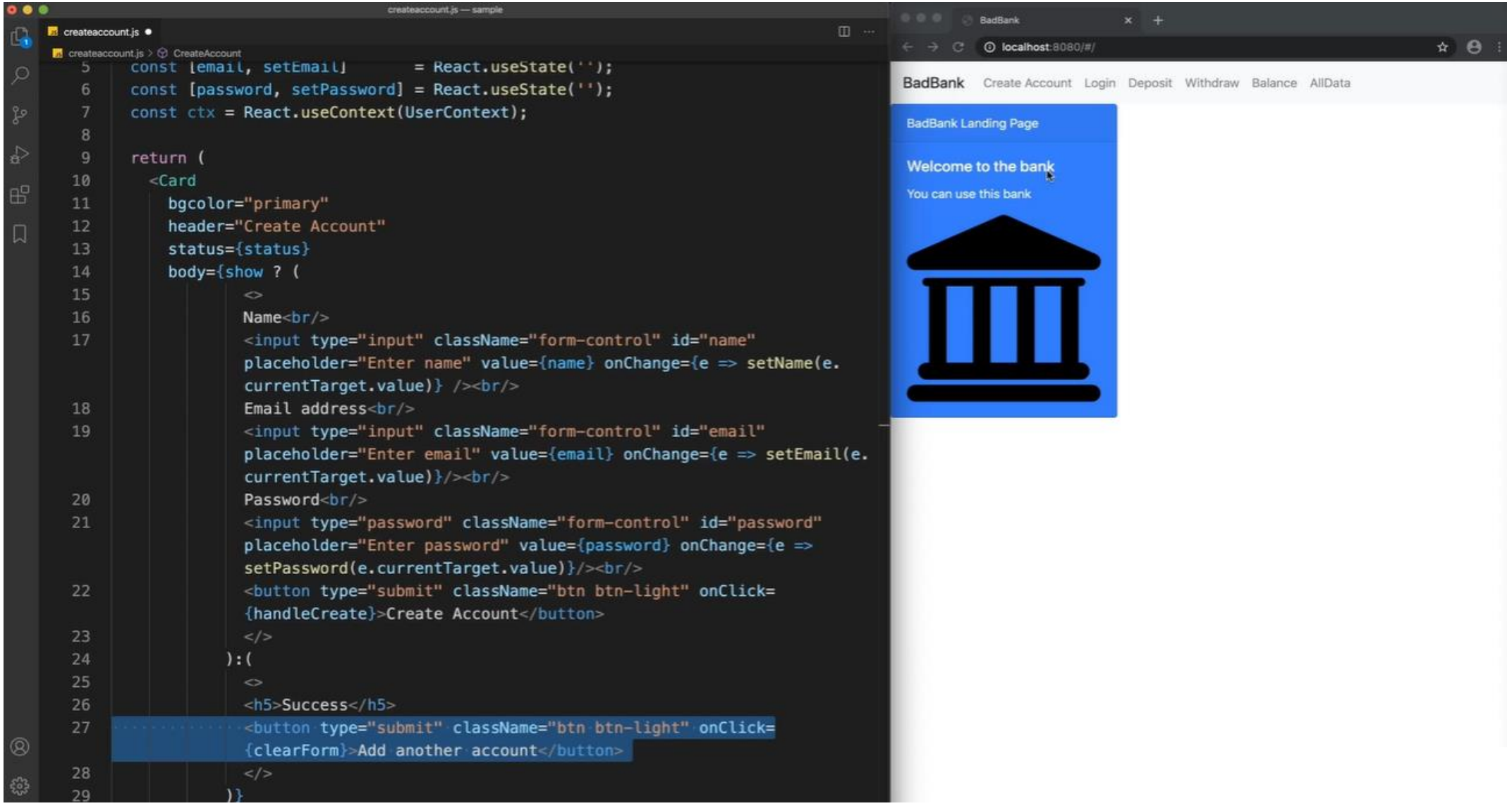
createaccount.js ●

createaccount.js > CreateAccount

```
function CreateAccount(){
  const [show, setShow]         = React.useState(true);
  const [status, setStatus]     = React.useState('');
  const [name, setName]         = React.useState('');
  const [email, setEmail]       = React.useState('');
  const [password, setPassword] = React.useState('');
  const ctx = React.useContext(UserContext);

  return (
    <Card
      bgcolor="primary"
      header="Create Account"
      status={status}
      body={show ? (
              <>
              Name<br/>
              <input type="input" className="form-control" id="name"
              placeholder="Enter name" value={name} onChange={e => setName(e.
              currentTarget.value)} /><br/>
              Email address<br/>
              <input type="input" className="form-control" id="email"
              placeholder="Enter email" value={email} onChange={e => setEmail(e.
              currentTarget.value)}/><br/>
              Password<br/>
              <input type="password" className="form-control" id="password"
              placeholder="Enter password" value={password} onChange={e =>
              setPassword(e.currentTarget.value)}/><br/>
              <button type="submit" className="btn btn-light" onClick=
              {handleCreate}>Create Account</button>
              </>
            ):(
```



Creating Interactive Form (2/8)



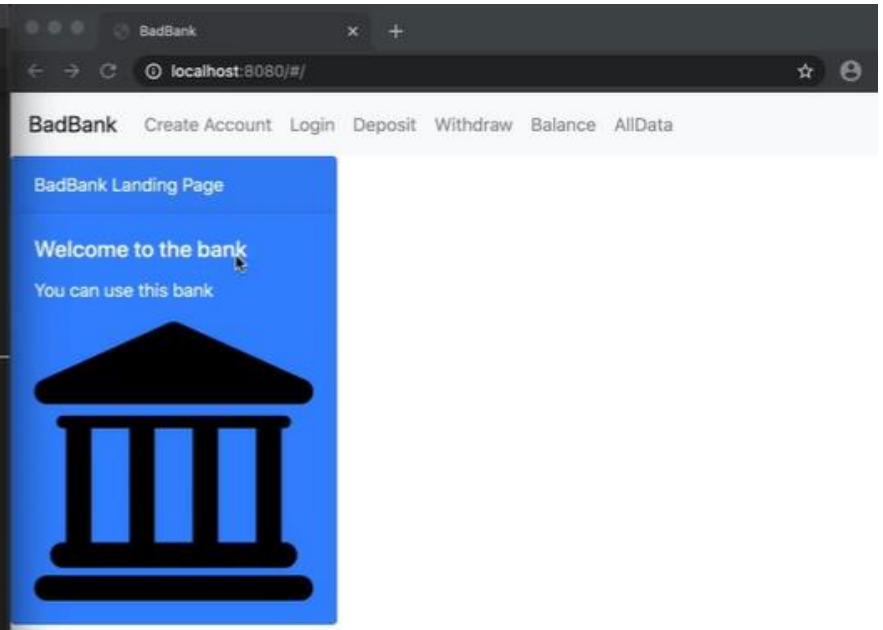
Creating Interactive Form (3/8)

createaccount.js — sample

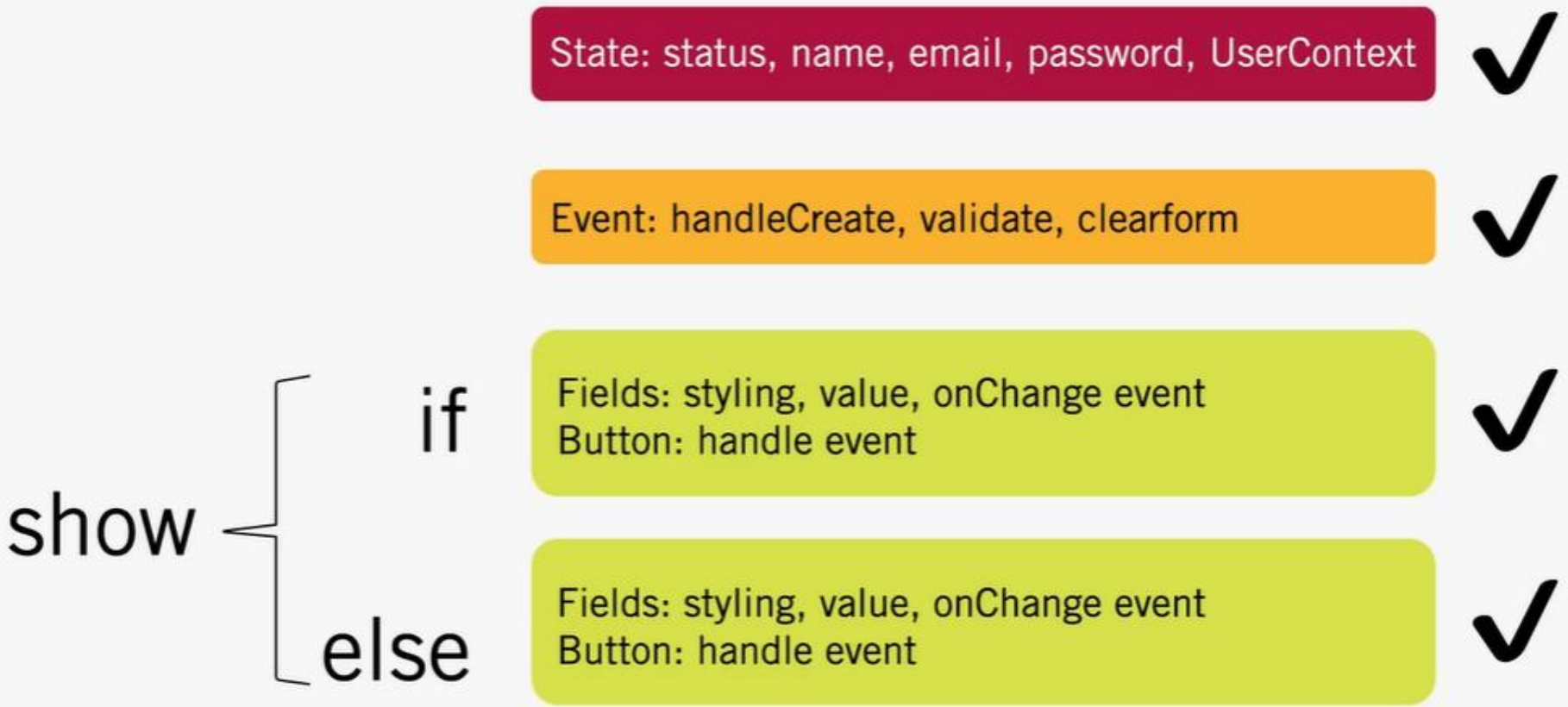
createaccount.js ●

createaccount.js > CreateAccount > handleCreate

```
5   const [email, setEmail]       = React.useState('');
6   const [password, setPassword] = React.useState('');
7   const ctx = React.useContext(UserContext);
8
9   function validate(field, label){
10      if (!field) {
11        setStatus('Error: ' + label);
12        setTimeout(() => setStatus(''),3000);
13        return false;
14      }
15      return true;
16  }
17
18  function handleCreate(){
19    console.log(name,email,password);
20    if (!validate(name,     'name'))     return;
21    if (!validate(email,    'email'))    return;
22    if (!validate(password, 'password')) return;
23    ctx.users.push({name,email,password,balance:100});
24    setShow(false);
25  }
26
27  function clearForm(){
28    setName('');
29    setEmail('');
30    setPassword('');
31    setShow(true);
32  }
33
34  return (
35    <Card
```



Creating Interactive Form (4/8)



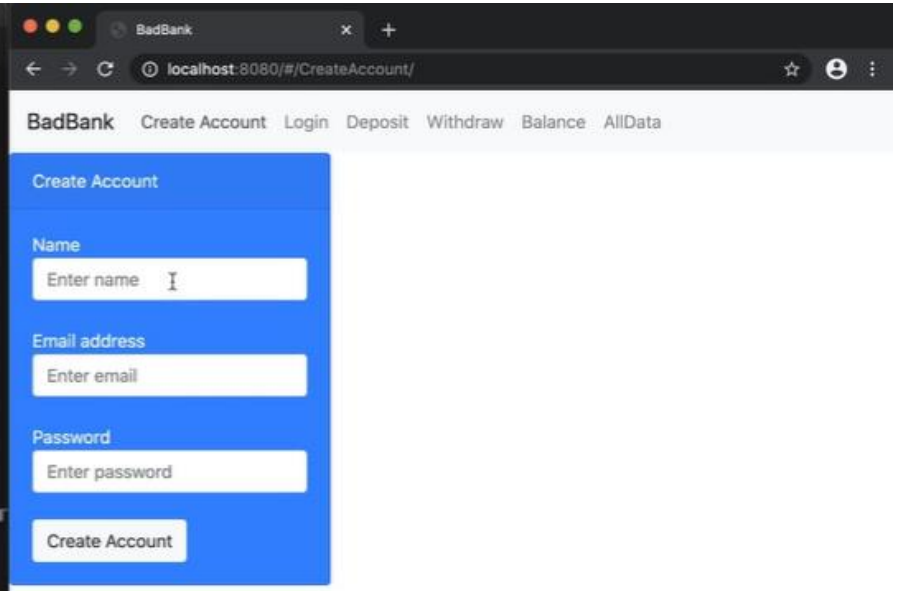
Creating Interactive Form (5/8)

createaccount.js — sample

createaccount.js ×

createaccount.js > CreateAccount

```
1  function CreateAccount(){
2    const [show, setShow]         = React.useState(true);
3    const [status, setStatus]     = React.useState('');
4    const [name, setName]         = React.useState('');
5    const [email, setEmail]       = React.useState('');
6    const [password, setPassword] = React.useState('');
7    const ctx = React.useContext(UserContext);
8
9    function validate(field, label){
10       if (!field) {
11         setStatus('Error: ' + label);
12         setTimeout(() => setStatus(''),3000);
13         return false;
14       }
15       return true;
16   }
17
18   function handleCreate(){
19     console.log(name,email,password);
20     if (!validate(name,     'name'))     return;
21     if (!validate(email,    'email'))    return;
22     if (!validate(password, 'password')) return;
23     ctx.users.push({name,email,password,balance:100});
24     setShow(false);
25   }
26
27   function clearForm(){
28     setName('');
29     setEmail('');
30     setPassword('');
31     setShow(true);
```

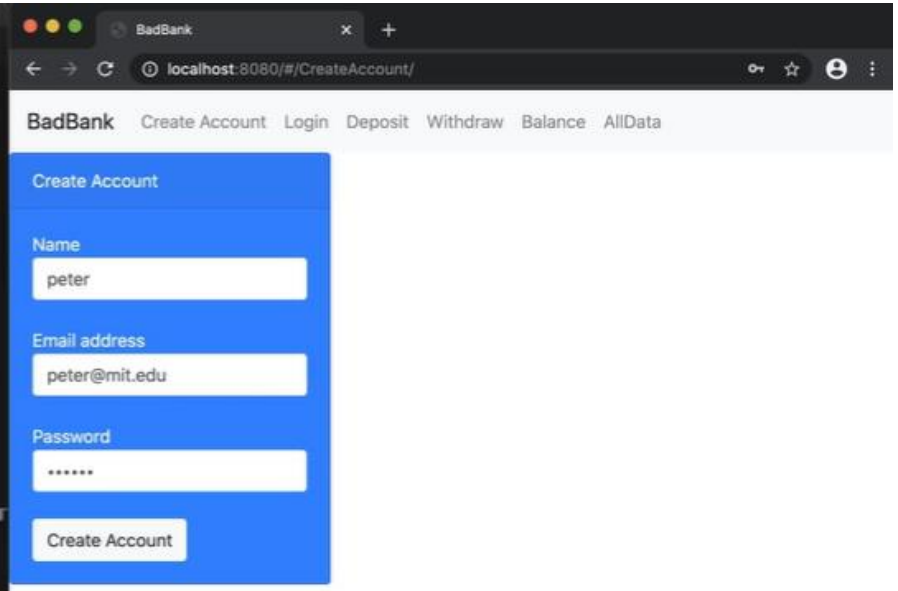


Creating Interactive Form (6/8)

createaccount.js — sample

createaccount.js × createaccount.js > CreateAccount

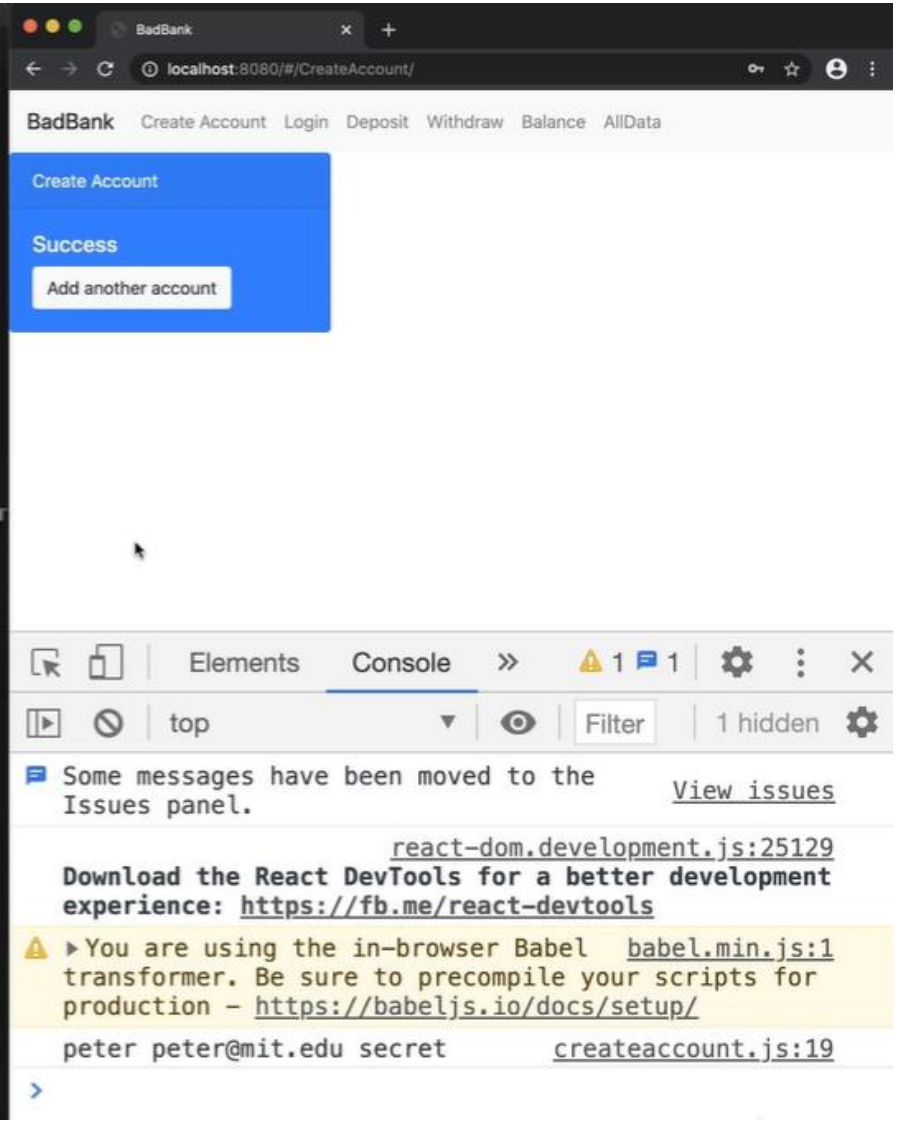
```
1  function CreateAccount(){
2    const [show, setShow]         = React.useState(true);
3    const [status, setStatus]     = React.useState('');
4    const [name, setName]         = React.useState('');
5    const [email, setEmail]       = React.useState('');
6    const [password, setPassword] = React.useState('');
7    const ctx = React.useContext(UserContext);
8
9    function validate(field, label){
10       if (!field) {
11         setStatus('Error: ' + label);
12         setTimeout(() => setStatus(''),3000);
13         return false;
14       }
15       return true;
16   }
17
18   function handleCreate(){
19     console.log(name,email,password);
20     if (!validate(name,     'name'))     return;
21     if (!validate(email,    'email'))    return;
22     if (!validate(password, 'password')) return;
23     ctx.users.push({name,email,password,balance:100});
24     setShow(false);
25   }
26
27   function clearForm(){
28     setName('');
29     setEmail('');
30     setPassword('');
31     setShow(true);
```



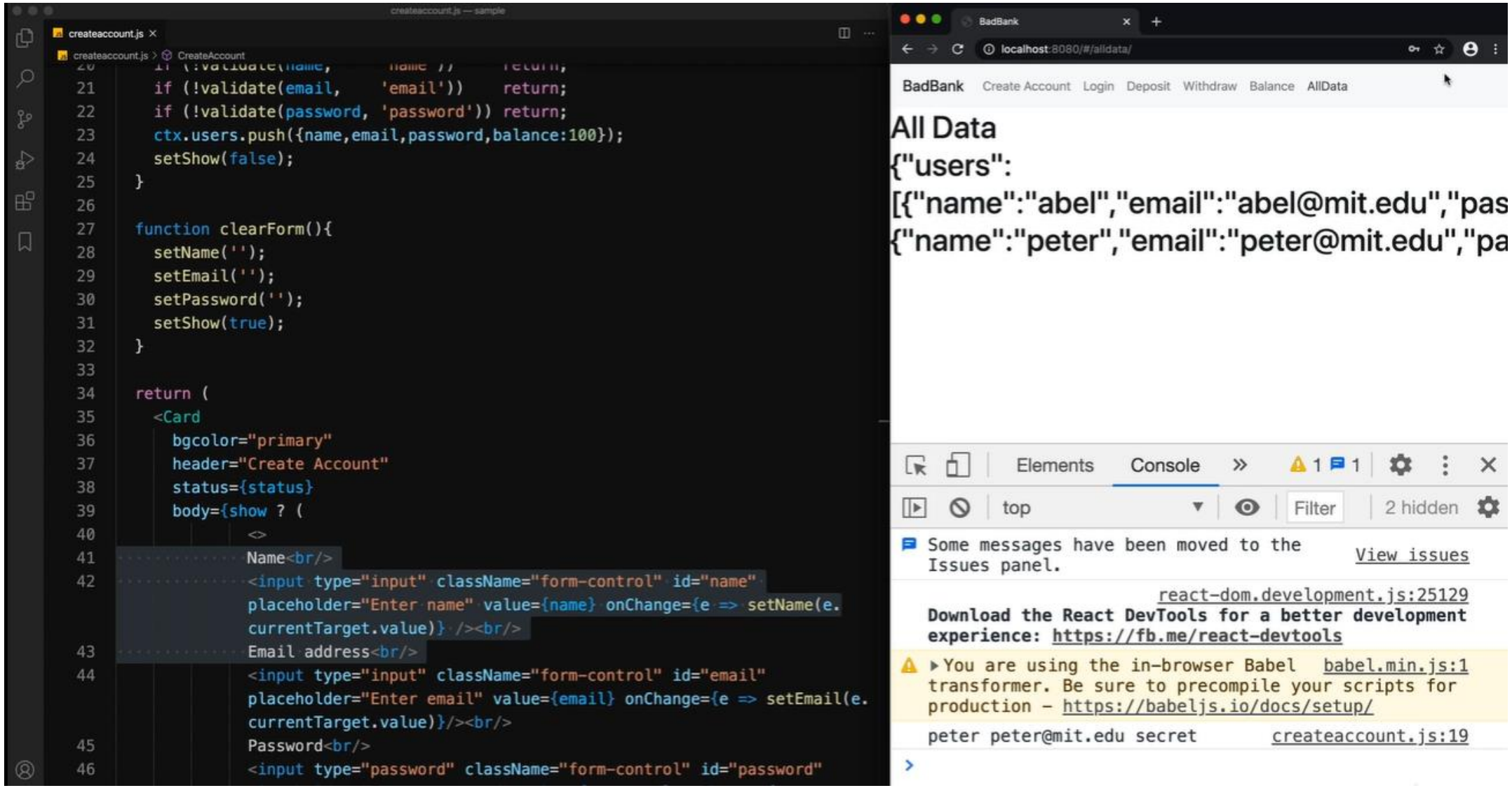
Creating Interactive Form (7/8)

createaccount.js

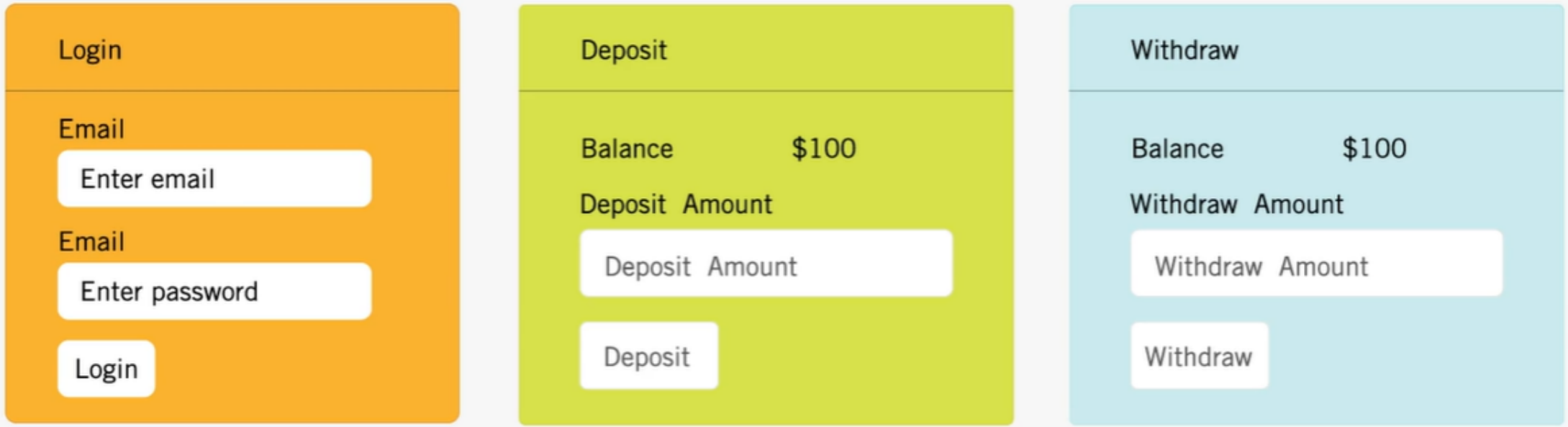
```
function CreateAccount(){
  const [show, setShow]         = React.useState(true);
  const [status, setStatus]     = React.useState('');
  const [name, setName]         = React.useState('');
  const [email, setEmail]       = React.useState('');
  const [password, setPassword] = React.useState('');
  const ctx = React.useContext(UserContext);

  function validate(field, label){
      if (!field) {
        setStatus('Error: ' + label);
        setTimeout(() => setStatus(''),3000);
        return false;
      }
      return true;
  }

  function handleCreate(){
    console.log(name,email,password);
    if (!validate(name,     'name'))     return;
    if (!validate(email,    'email'))    return;
    if (!validate(password, 'password')) return;
    ctx.users.push({name,email,password,balance:100});
    setShow(false);
  }

  function clearForm(){
    setName('');
    setEmail('');
    setPassword('');
    setShow(true);
```



Creating Interactive Form (8/8)

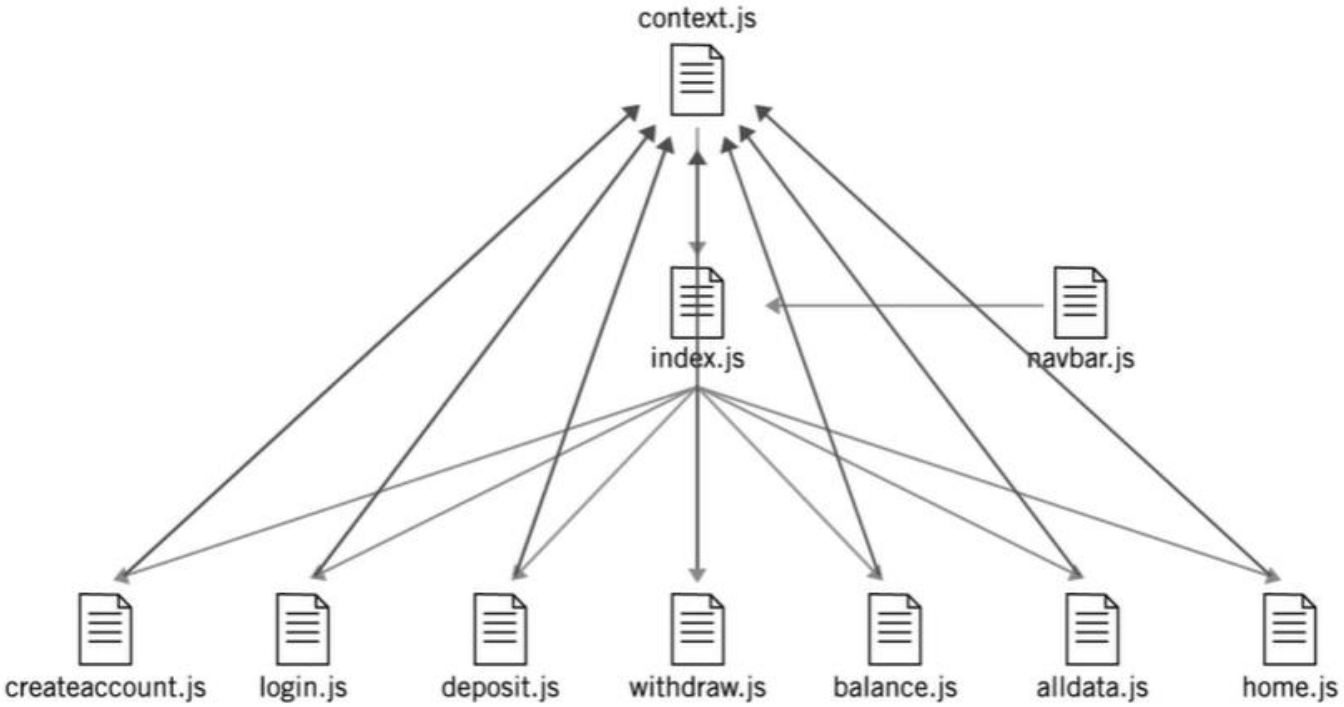


Creating Interactive Form: Your Turn



How To Re-Design Your Application?

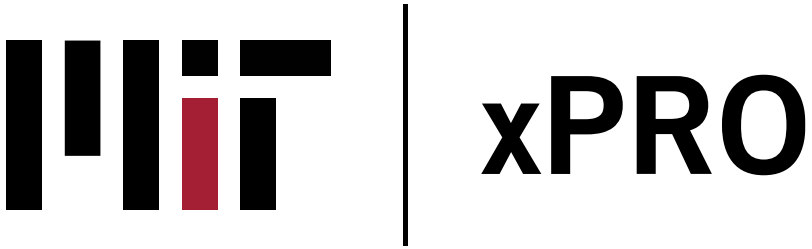
Bad Bank – How Can You Improvise Work Done ?



Code Snippet For Account Creation

createaccount.js

```
function CreateAccount(){
  const ctx = React.useContext(UserContext);

  function handle(data){
    ctx.users.push({name:data.name,email:data.
      email,password:data.password,balance:100});
    return true;
  }

  return (
    <BankForm
      bgcolor="primary"
      label="Create Account"
      handle={handle}
      hideAmount={true}
      successButton="Add another account"
    />
  )
}
```



© MIT xPRO 2021. All rights reserved.