Creating a Modern App to Interact with IBM z/OS

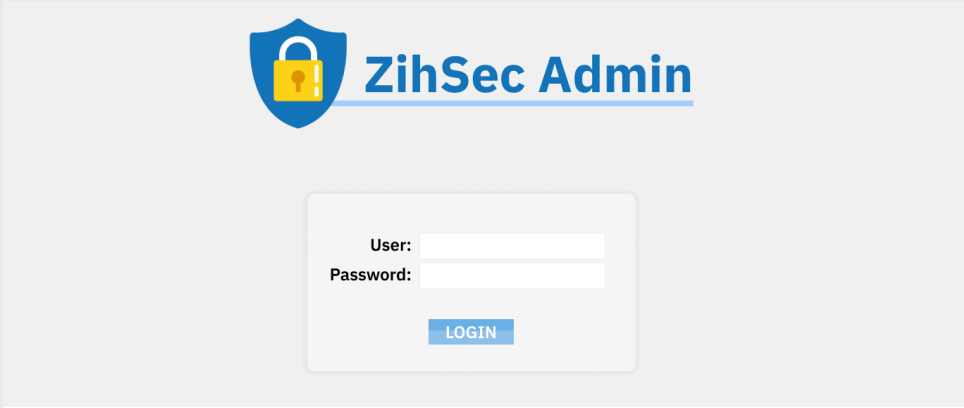
—

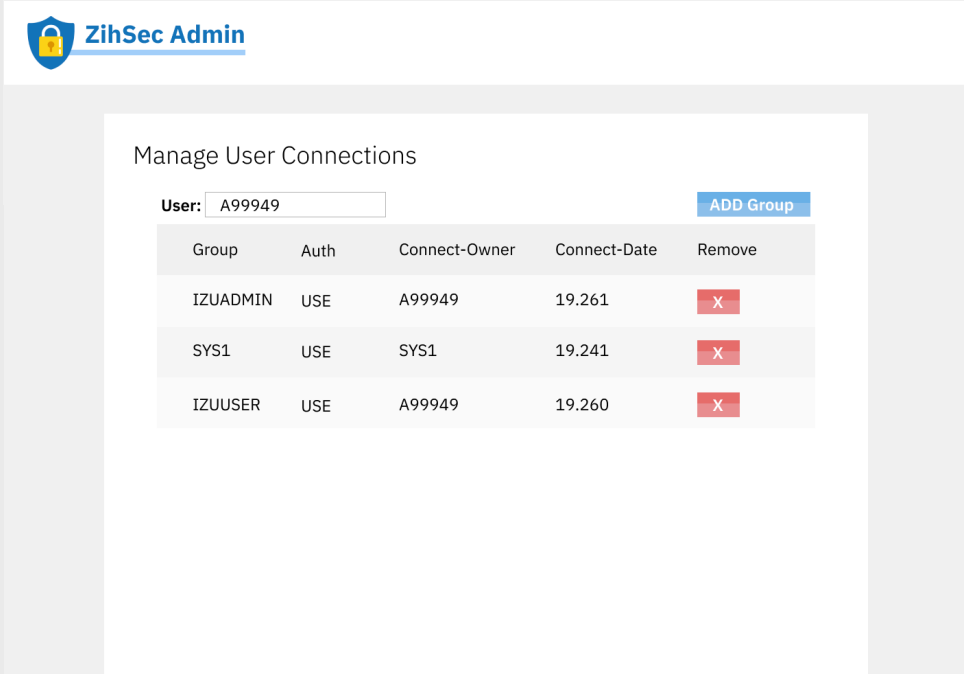
Bill Pereira
IBM Z, Developer Advocate

think

IBM

Contents





Our architecture

LOGIN

Back-end

- Core business
- Database connection
- E-mail services
- Webservices
- User authentication
- Crypto and Security

JSON

Front-end web

JSON

External services

How we came to this point?

API

Application Programming Interface

From modules to remote calls

60’s – Remote Procedure Call

90’s- Remote Method Invocation

98 -XML-RPC (Dave Winer)

SOAP – Simple Object Access Protocol

Created by Tim Berners-Lee to share documents

2005 Flash

AJAX - Tim Berners-Lee

Asynchronous Requests

JSON - Douglas Crockford

Documented

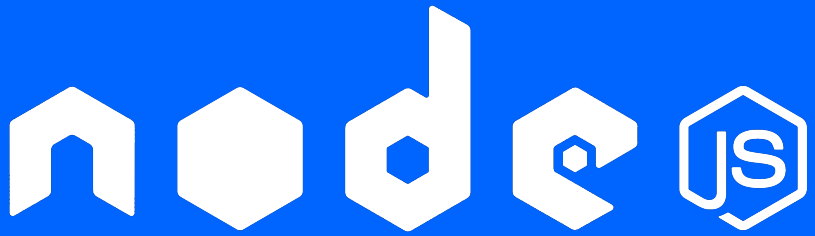
REST (Representational State Transfer) - **Roy** Thomas Fielding

Reusable

Initializing our repository

<coding
awesome
apps/>

Our Backend



Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.

express

@zowe/cli

axios

dotenv

z/OSMF AUTHENTICATION

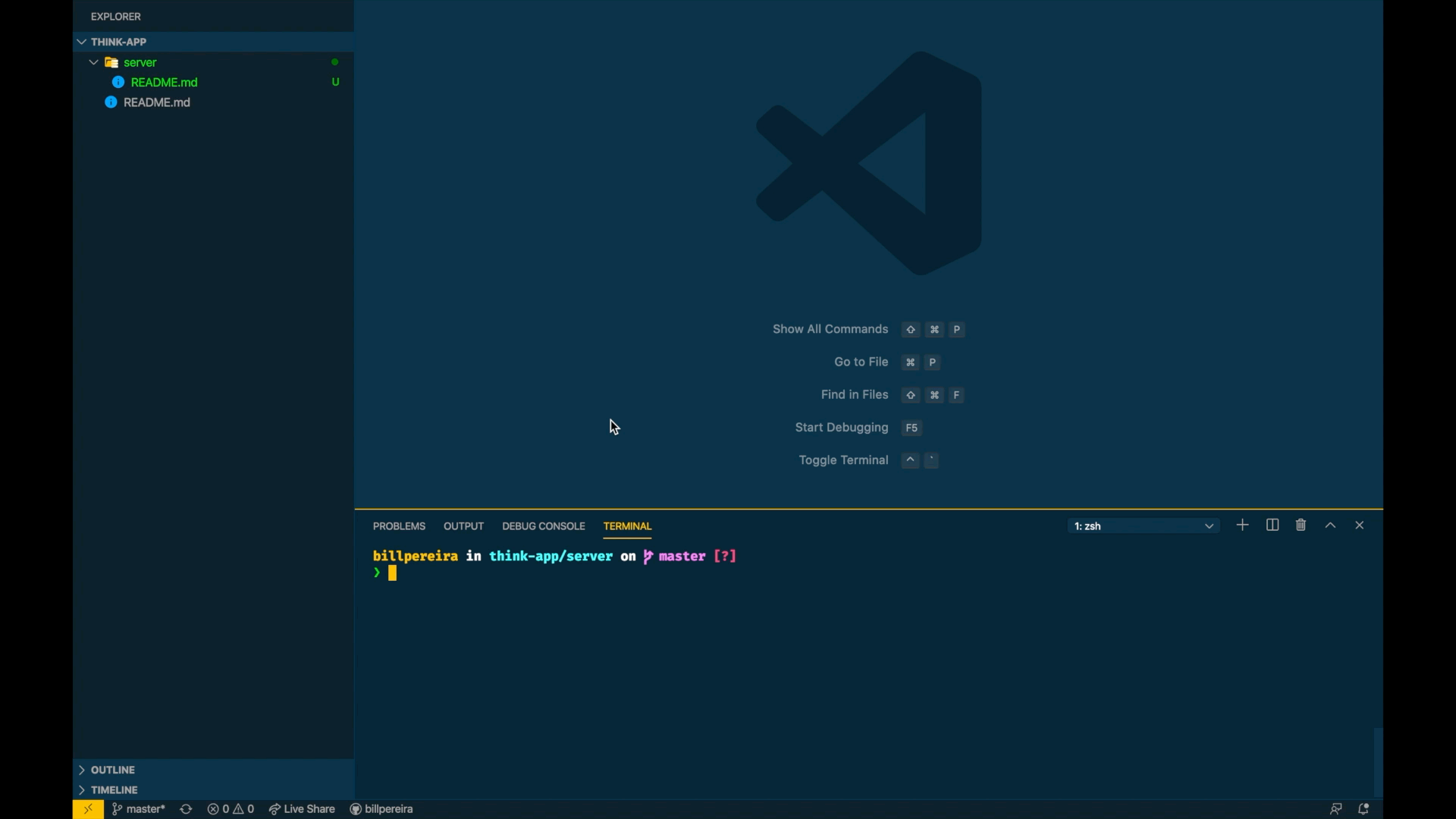
POST /services/authenticate

RACF

LU *USERID*

RE *USERID* GROUP(*GROUP*)

CO *USERID* GROUP(*GROUP*)


EXPLORER
THINK-APP
server
README.md
U
README.md
Show All Commands
Go to File
Find in Files
Start Debugging
F5
Toggle Terminal
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: zsh
billpereira in think-app/server on master [?]
OUTLINE
TIMELINE
master*
0 0
Live Share
billpereira

Hello World

```
require('dotenv/config')
const express = require('express');

const PORT = process.env.PORT || 3333

const app = express()

app.get('/api',(req,res)=>{
  return res.json({message:'Hello World'})
})

app.listen(PORT,()=>{
  console.log(`App listening on port ${PORT}`)
})
```

Authentication

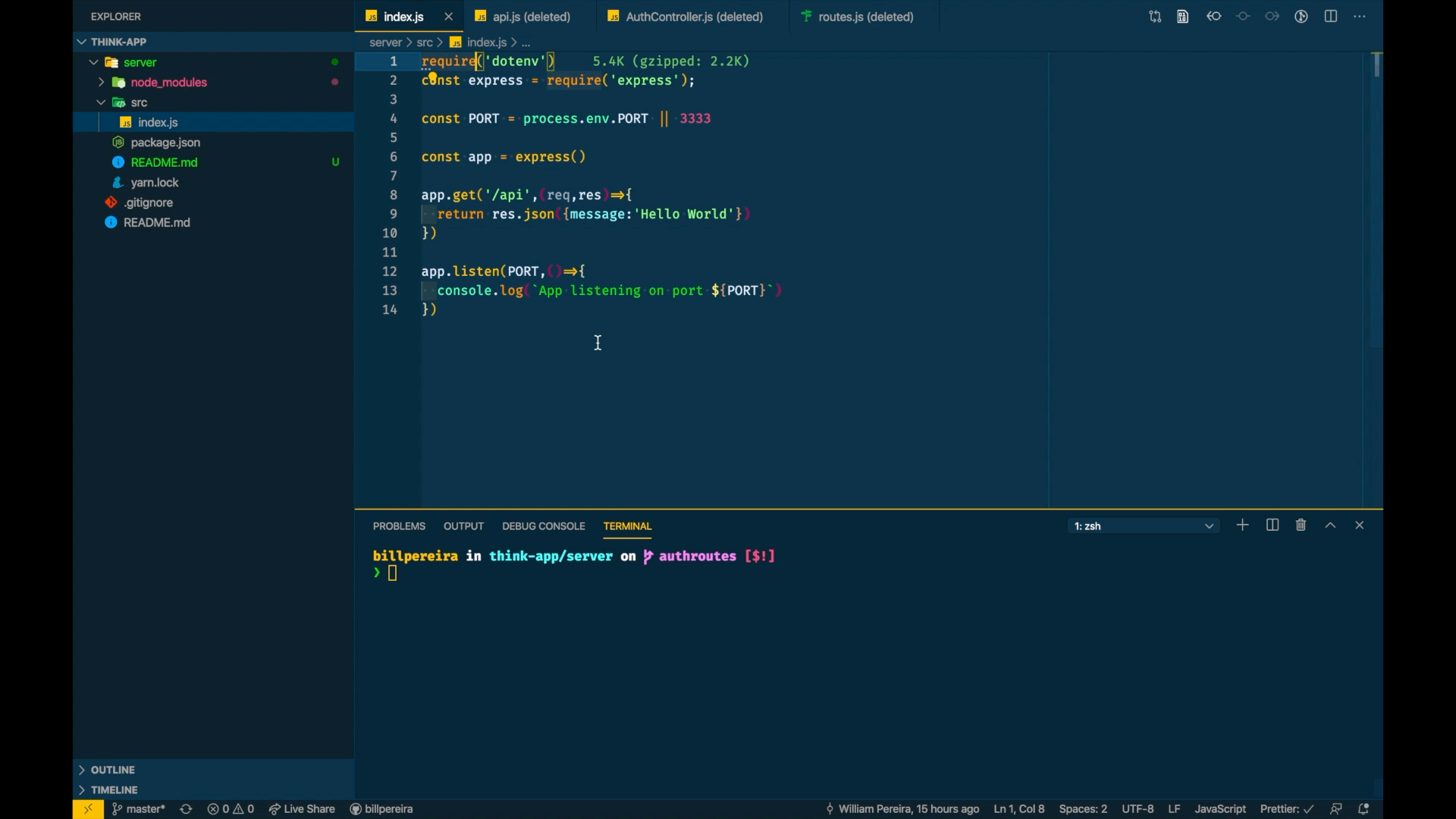
Quick Update

.env

```
ZOSMF_HOST=YOUR.HOST.COM
ZOSMF_PORT=443
ZOSMF_USER=YOURUSER
ZOSMF_PWD=YOURPSWD
```

.gitignore

```
**/.env
**/node_modules
```


EXPLORER
index.js
api.js (deleted)
AuthController.js (deleted)
routes.js (deleted)
THINK-APP
server
node_modules
src
index.js
package.json
README.md
yarn.lock
.gitignore
README.md
server > src > index.js > ...
require('dotenv') 5.4K (gzipped: 2.2K)
const express = require('express');
const PORT = process.env.PORT || 3333
const app = express()
app.get('/api',(req,res)=>{
return res.json({message:'Hello World'})
})
app.listen(PORT,()=>{
console.log(`App listening on port ${PORT}`)
})
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: zsh
billpereira in think-app/server on authroutes [$!]
OUTLINE
TIMELINE
master*
Live Share
billpereira
William Pereira, 15 hours ago
Ln 1, Col 8
Spaces: 2
UTF-8
LF
JavaScript
Prettier:

Authentication

zOSMF Services for authentication

- **POST**
- **/services/authentication**
- **Basic Auth**

src/controllers/services/api.js

```
const axios = require("axios");
const https = require("https");

const api = axios.create({
baseURL: process.env.ZOSMF_HOST + ":" +
process.env.ZOSMF_PORT,
headers: { "X-CSRF-ZOSMF-HEADER": "" },
httpsAgent: new https.Agent({
rejectUnauthorized: false,
}),
});

module.exports = api;
```

Authentication

zOSMF Services for authentication

- **POST**
- **/services/authentication**
- **Basic Auth**

src/controllers/AuthController.js

```
const api = require("./services/api");

const login = async (req, res) => {
api.defaults.auth = req.body;
const response = await
api.post("/zosmf/services/authenticate");
return res.json({
    auth: !!response.headers["set-cookie"]
  });
};

module.exports = {
login,
};
```

Authentication

zOSMF Services for authentication

- **POST**
- **/services/authentication**
- **Basic Auth**

src/routes.js

```
const { Router } = require("express");
const AuthController =
       require("./controllers/AuthController");

const routes = Router();
routes.post("/api/login", AuthController.login);

module.exports = routes;
```

Authentication

zOSMF Services for authentication

- **POST**
- **/services/authentication**
- **Basic Auth**

src/index.js

```
require("dotenv/config");
const express = require("express");
const routes = require("./routes");

const PORT = process.env.PORT || 3333;

const app = express();
app.use(express.json());
app.use(routes);

app.listen(PORT, () => {
console.log(`App listening on port ${PORT}`);
});
```

Zowe and RACF

RACF

Two of the fundamental elements of RACF® are users and groups. Users, of course, are the many people who log on to a system, each with a unique user ID. Administration of a small number of users is not too difficult. However, when there are thousands of users, administration becomes a very large task. To make this task more manageable, the concept of groups was developed.

A group is a RACF entity with which any number of users are associated. Usually, the users in a group have some logical relationship to one another. The relationship used most frequently is members of a department. Many installations pattern their group-user structure after their organization charts.

At the top of the RACF group-user structure is a group called SYS1. When you install RACF, it defines this group for you. The SYS1 group is the highest group in the total RACF group-user structure. You can define your system administrator and system auditor as members of this group. The system administrator has the SPECIAL attribute and the system auditor has the AUDITOR attribute.

List a User: LISTUSER (USERID)

Connect User/Group: CONNECT (USERID...) GROUP(GROUP-NAME)

Remove connection User/Group: REMOVE (USERID...) GROUP(GROUP-NAME)

@zowe/cli

ZosmfSession.createBasicZosmfSession

```
static createBasicZosmfSession(profile: IProfile): Session;


options: "zos124 --host zos124 --user ibmuser --password myp4ss --reject-
unauthorized false",
description: "Create a zosmf profile called 'zos124' to connect to z/OSMF
at the host zos124 (default port - 443) " +
"and allow self-signed certificates"
```

IssueTso.issueTsoCommand

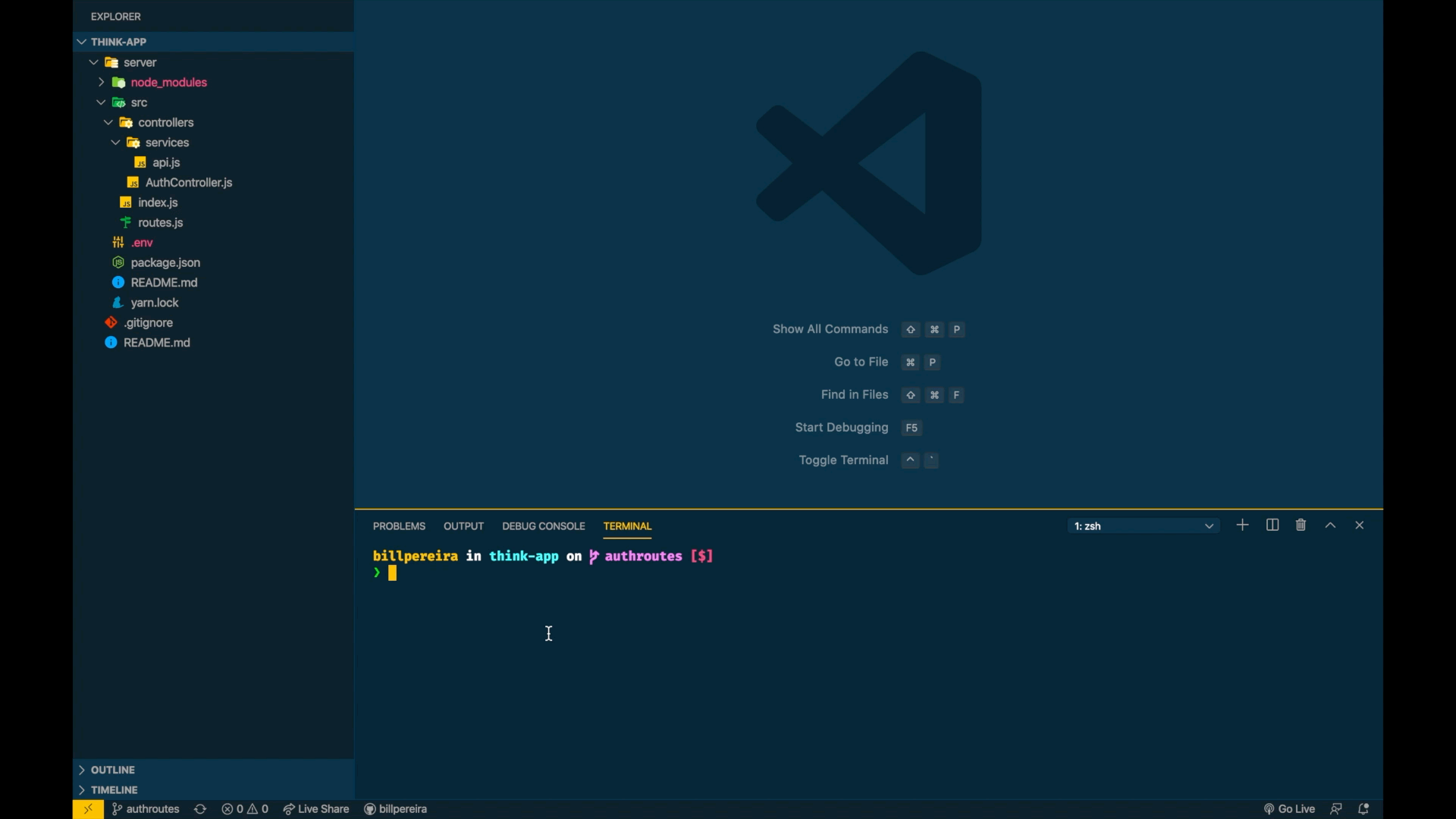
```
static issueTsoCommand(session: AbstractSession, accountNumber: string, command:
string, startParams?: IStartTsoParms): Promise<IIssueResponse>;
```

RACF Controller

Handle our RACF Interaction through Zowe with updated user object

- **GET /api/user/list?user=XXXXXX**
- **POST /api/user/remove {user,group}**
- **POST /api/user/connect {user,group}**


```
{
  "userData": {
    "user": "z99997",
    "connections": [
      {
        "group": "STUDENT1",
        "auth": "USE",
        "connectOwner": "SYS1",
        "connectDate": "20.049"
      },
      {
        "group": "IZUADMIN",
        "auth": "USE",
        "connectOwner": "A99997",
        "connectDate": "20.126"
      }
    ]
  }
}
```


EXPLORER
THINK-APP
server
node_modules
src
controllers
services
api.js
AuthController.js
index.js
routes.js
.env
package.json
README.md
yarn.lock
.gitignore
README.md
Show All Commands
Go to File
Find in Files
Start Debugging
Toggle Terminal
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: zsh
billpereira in think-app on authroutes [$]
OUTLINE
TIMELINE
authroutes
Live Share
billpereira
Go Live

src/controllers/services/zowe.js

```
const zowe = require("@zowe/cli");

const profile = {
user: process.env.ZOSMF_USER,
password: process.env.ZOSMF_PWD,
host: process.env.ZOSMF_IP,
port: process.env.ZOSMF_PORT,
rejectUnauthorized: false,
};

const sendCommand = async (command) =>{
const session = await
zowe.ZosmfSession.createBasicZosmfSession(profile);
const { commandResponse } = await
zowe.IssueTso.issueTsoCommand(
session,
"FB3",
command
);
return commandResponse
}
```

```
const getUserInfo = async (user) => {
const command = `LU ${user}`;
const commandResponse = await sendCommand(command)
const connections = commandResponse
.split("GROUP")
.splice(2)
.map((group) => {
const connection = {
group: group.substr(1, 8).trim(),
auth: group.substr(16, 8).trim(),
connectOwner: group.substr(39, 8).trim(),
connectDate: group.substr(62, 6).trim(),
};
return connection;
});
return { user, connections };
};

module.exports = {
sendCommand,
getUserInfo
}
```

src/controllers/RacfController.js

```
const {getUserInfo,sendCommand} =
require('./services/zowe');

const listUser = async (req, res) => {
const { user } = req.query;
const userData = await getUserInfo(user);
return res.json({ userData });
};

const connectUser = async(req,res)=>{
const { user,group } = req.body;
const command = `CO ${user} GROUP(${group})`
await sendCommand(command)
const userData = await getUserInfo(user);
return res.json({ userData });
}
const removeConnection = async(req,res)=>{
const { user,group } = req.body;
const command = `RE ${user} GROUP(${group})`
await sendCommand(command)
const userData = await getUserInfo(user);
return res.json({ userData });
}

module.exports = {
listUser,
connectUser,
removeConnection
};
```

The Frontend

React

Declarative

React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes.

Declarative views make your code more predictable and easier to debug.

Component-Based

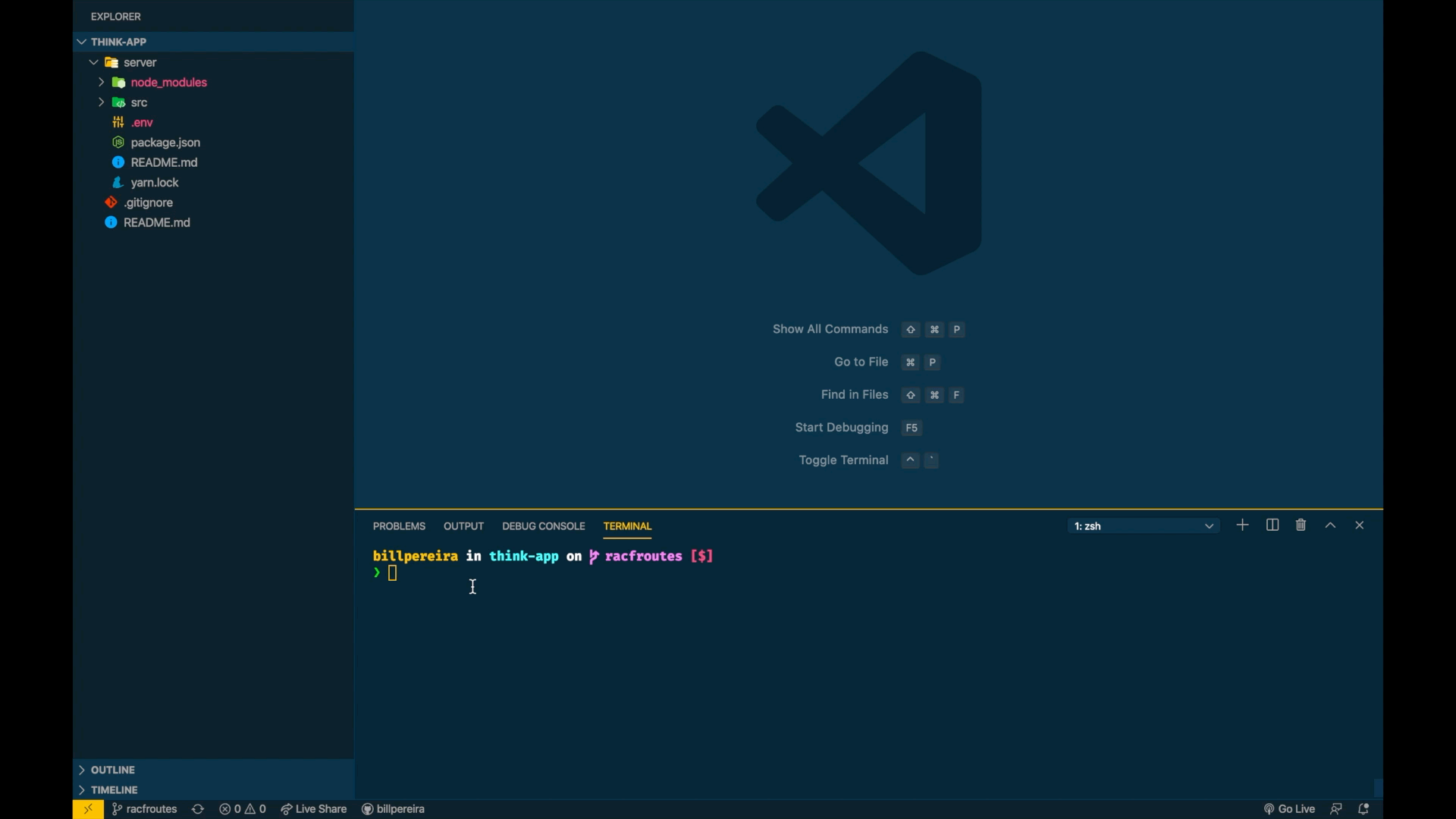
Build encapsulated components that manage their own state, then compose them to make complex UIs.

Since component logic is written in JavaScript instead of templates, you can easily pass rich data through your app and keep state out of the DOM.

Learn Once, Write Anywhere

Independent of the rest of your technology stack, so you can develop new features in React without rewriting existing code.

React can also render on the server using Node and power mobile apps using React Native.


EXPLORER
THINK-APP
server
node_modules
src
.env
package.json
README.md
yarn.lock
.gitignore
README.md
Show All Commands
Go to File
Find in Files
Start Debugging
F5
Toggle Terminal
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1: zsh
billpereira in think-app on racfroutes [$]
OUTLINE
TIMELINE
racfroutes
0 0
Live Share
billpereira
Go Live

Hello World

Create react-app

React Router Dom

src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

ReactDOM.render(
<React.StrictMode>
<App />
</React.StrictMode>,
document.getElementById('root')
);
```

Hello World

Create react-app

React Router Dom

src/app.js

```
import React from 'react';
import {BrowserRouter as Router, Route} from 'react-
router-dom'
import LoginPage from './components/LoginPage'
import GroupsPage from './components/GroupsPage'


function App() {
return (
<Router>
<Route exact path='/' component={LoginPage}/>
<Route exact path='/groups' component={GroupsPage}/>
</Router>
);
}

export default App;
```

Hello World

Create react-app

React Router Dom

src/app.js

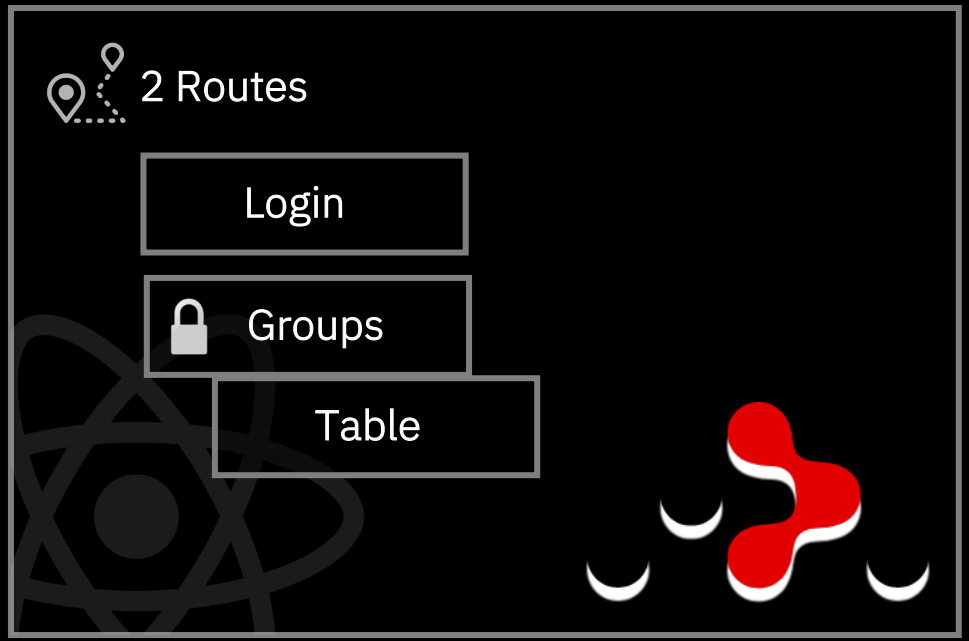
```
import React from 'react'

const LoginPage = () => {
return (
<div>
Hello from Login
</div>
)
}

export default LoginPage
```

What are we building

The App...



JSON

Creating the components

2 Routes

- Login
- Groups

src/App.js

```
import React from 'react';
import {BrowserRouter as Router, Route} from 'react-
router-dom'
import LoginPage from './components/LoginPage'
import GroupsPage from './components/GroupsPage'
import './App.css'


function App() {
return (
<Router>
<Route exact path='/' component={LoginPage}/>
<Route exact path='/groups' component={GroupsPage}/>
</Router>
);
}

export default App;
```

Creating the components

2 Routes

- Login
- Groups

src/components/Login.js

```
…
const LoginPage = () => {
return (
<div>
<div className="row">
<img className="login-logo" src={logo} alt="logo" />
</div>
<div className="row">
<div className="box-login">
<div className="box-row">
User:<input type="text"></input>
</div>
<div className="box-row">
Password:<input type="password"></input>
</div>
<div className="box-row">
<button className="login-button">LOGIN</button>
</div>
</div>
</div>
</div>
);
};
…
```

Creating the components

2 Routes

- Login
- Groups

src/components/Groups.js

```
import logo from "../logo-pages.png";
import Table from "./Table";

…

const userData = {};

const GroupsPage = () => {
return (
…
<div className="row">
<Table userData={userData}></Table>
…
);
};

export default GroupsPage;
```

Creating the components

2 Routes

- Login
- Groups

src/components/Table.js

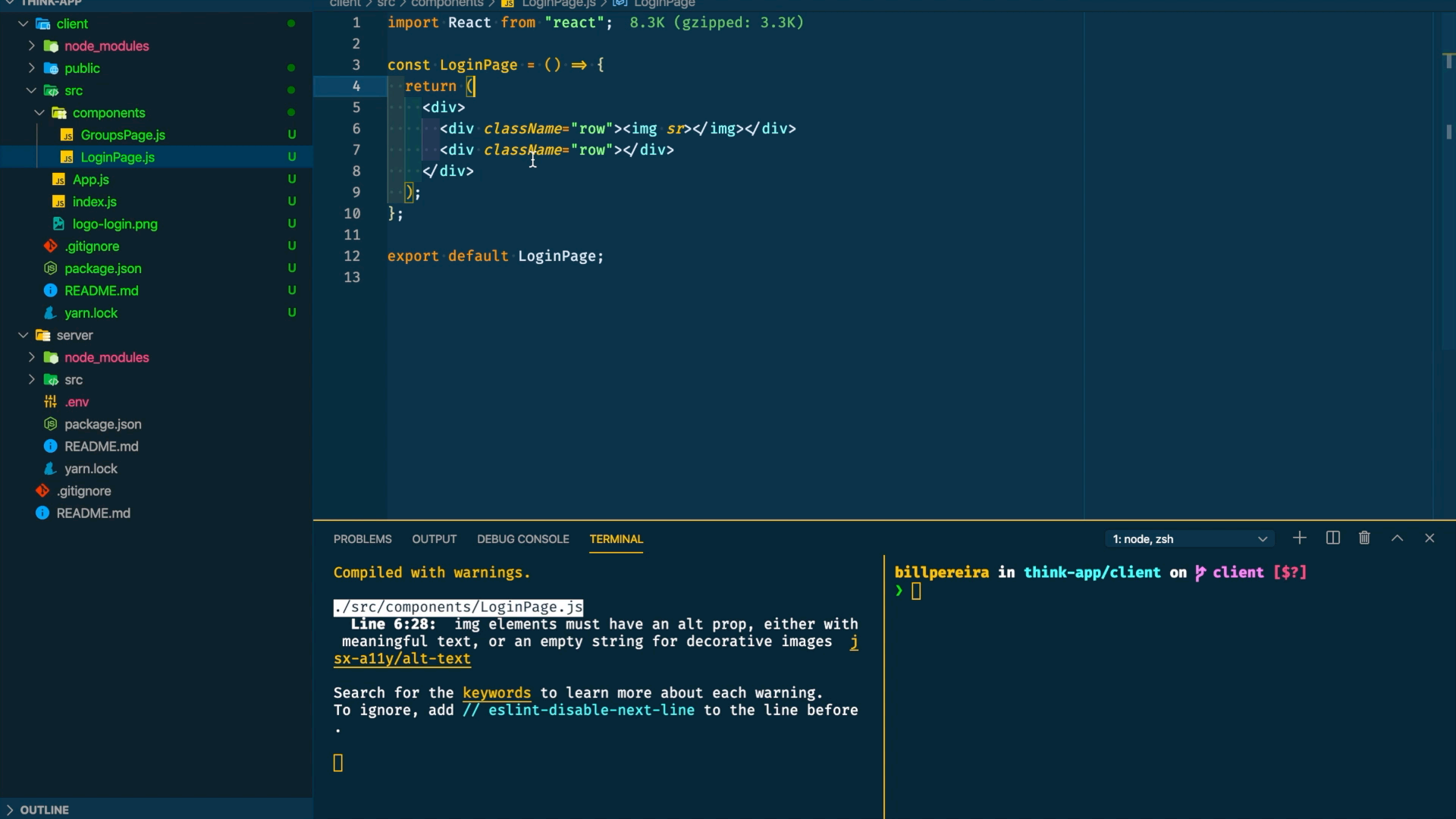
```
…

const Table = (userData) => {
const { connections } = userData;
…
{connections &&
connections.map((connection, i) => {
return (
<div className={i % 2 === 0 ? "line-light" : "line-
heavy"}>
<div className="field">{connection.group}</div>
<div className="field">{connection.auth}</div>
<div className="field">{connection.connectOwner}</div>
<div className="field">{connection.connectDate}</div>
<div className="field">
<button className="delete-btn">X</button>
</div>
</div>
);
})}

…

export default Table;
```

Styling the pages


client > src > components > LoginPage.js > LoginPage
client
node_modules
public
src
components
GroupsPage.js U
LoginPage.js U
App.js U
index.js U
logo-login.png U
.gitignore U
package.json U
README.md U
yarn.lock U
server
node_modules
src
.env
package.json
README.md
yarn.lock
.gitignore
README.md
OUTLINE
1 import React from "react"; 8.3K (gzipped: 3.3K)
2
3 const LoginPage = () => {
4 return (
5
6 <img sr></img>
7
8
9);
10 };
11
12 export default LoginPage;
13
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: node, zsh
Compiled with warnings.
./src/components/LoginPage.js
Line 6:28: img elements must have an alt prop, either with meaningful text, or an empty string for decorative images jsx-a11y/alt-text
Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before .
billpereira in think-app/client on client [$?]

Calling the backend

Axios

```
import axios from "axios";

const api = axios.create({
baseURL: "http://localhost:3333",
});

export default api;
```

State variables

```
import React, { useState } from "react";

…

const LoginPage = () => {
const [username, setUsername] = useState("");
…
…
<input
type="text"
placeholder="your id"
value={username}
onChange={(e) => setUsername(e.target.value)}
></input>
…
```

Authentication

```
…

const handleLogin = async () => {
const data = { username, password };

try {
const response = await api.post("api/login", data);
localStorage.setItem("auth", response.data.auth);
response.data.auth
? history.push("/groups")
: alert("User or Password Invalid");
} catch (error) {
alert("Login Failed");
}
};
…
<form onSubmit={(e) => e.preventDefault()}>
…
<button className="login-button"
onClick={() => handleLogin()}
…
```

Protecting the route

```
const isAuthenticated = () =>
localStorage.getItem("auth");

const PrivateRoute = ({ component: Component, ...rest })
=> (
<Route
{...rest}
render={(props) =>
isAuthenticated() ? (
<Component {...props} />
) : (
<Redirect to={{ pathname: "/", state: { from:
props.location } }} />
)
}
/>
);

function App() {
return (
<Router>
<Route exact path="/" component={LoginPage} />
<PrivateRoute exact path="/groups"
component={GroupsPage} />
</Router>
);
}
```

Rendering the data

```
{connections &&
connections.map((connection, i) => {
return (
<div
className={i % 2 === 0 ? "line-light" : "line-heavy"}
key={i}
onMouseOver={() => {
setGroupToDelete(connection.group);
}}
>
<div className="field">{connection.group}</div>
<div className="field">{connection.auth}</div>
<div className="field">{connection.connectOwner}</div>
<div className="field">{connection.connectDate}</div>
<div className="field">
<button
className="delete-btn"
onClick={async () => {
deleteGroup();
}}
>
X
</button>
</div>
</div>
);
})}
```

Take a deep look on it...

All code is available on github

Hello-world

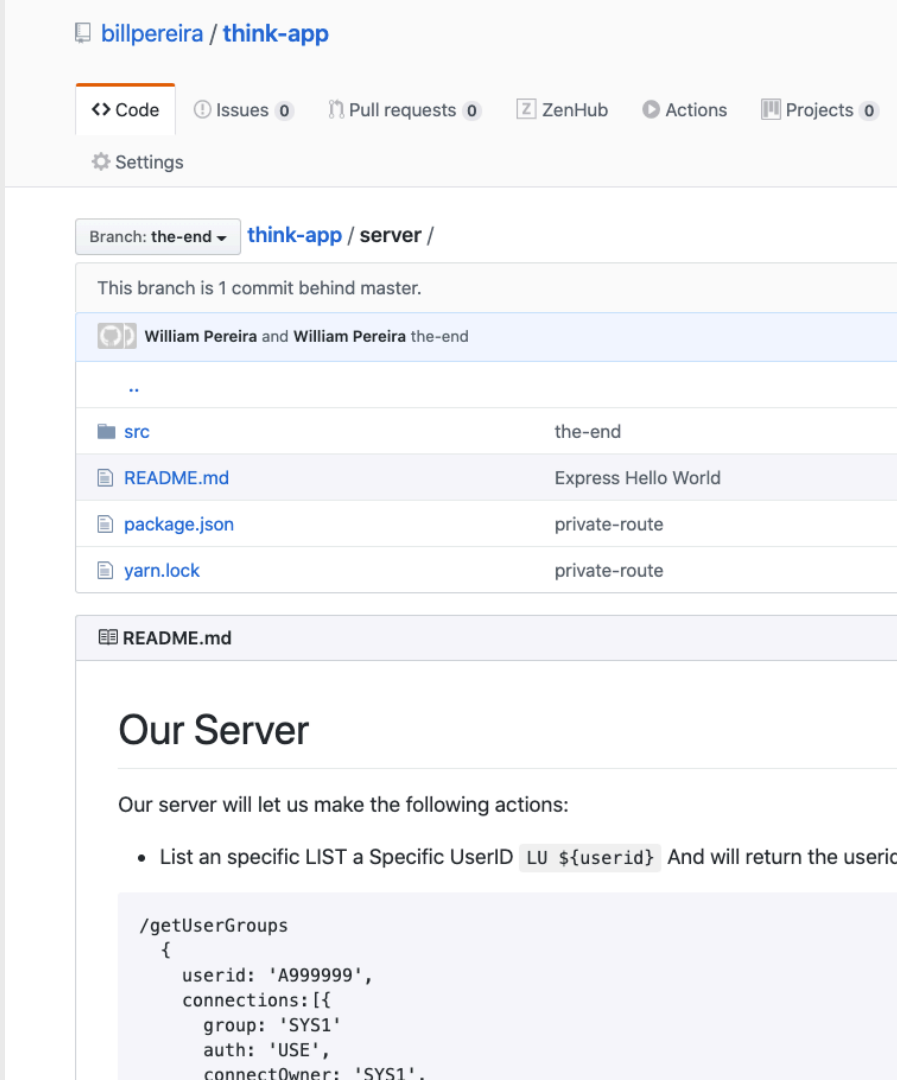
Authroutes

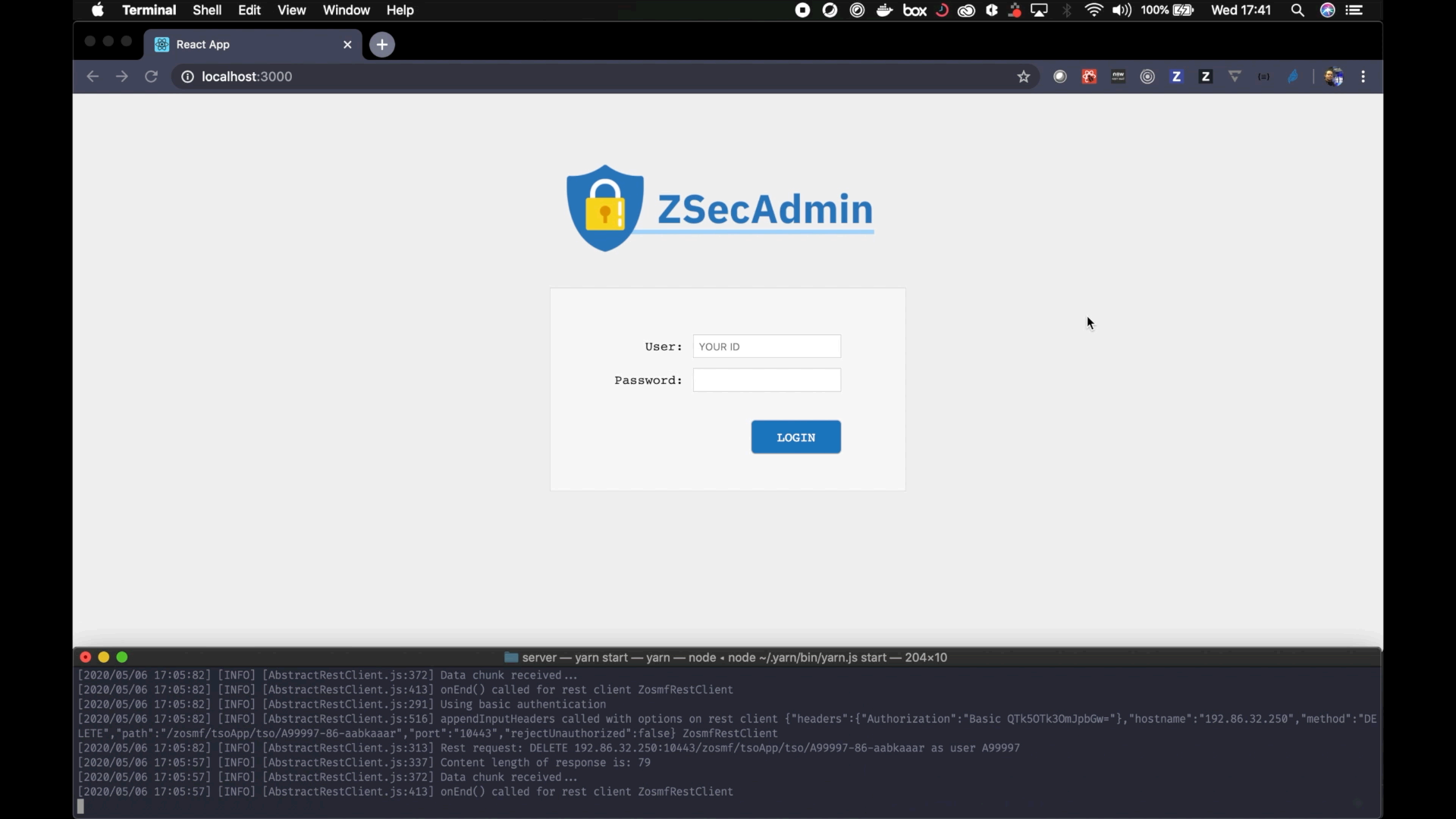
Racfroutes

Client

Private-route

The end




Terminal Shell Edit View Window Help
100% Wed 17:41
React App
localhost:3000
ZSecAdmin
User:
YOUR ID
Password:
LOGIN
server — yarn start — yarn — node ◂ node ~/.yarn/bin/yarn.js start — 204×10
[2020/05/06 17:05:82] [INFO] [AbstractRestClient.js:372] Data chunk received...
[2020/05/06 17:05:82] [INFO] [AbstractRestClient.js:413] onEnd() called for rest client ZosmfRestClient
[2020/05/06 17:05:82] [INFO] [AbstractRestClient.js:291] Using basic authentication
[2020/05/06 17:05:82] [INFO] [AbstractRestClient.js:516] appendInputHeaders called with options on rest client {"headers":{"Authorization":"Basic QTk5OTk3OmJpbGw="},"hostname":"192.86.32.250","method":"DELETE","path":"/zosmf/tsoApp/tso/A99997-86-aabkaaar","port":"10443","rejectUnauthorized":false} ZosmfRestClient
[2020/05/06 17:05:82] [INFO] [AbstractRestClient.js:313] Rest request: DELETE 192.86.32.250:10443/zosmf/tsoApp/tso/A99997-86-aabkaaar as user A99997
[2020/05/06 17:05:57] [INFO] [AbstractRestClient.js:337] Content length of response is: 79
[2020/05/06 17:05:57] [INFO] [AbstractRestClient.js:372] Data chunk received...
[2020/05/06 17:05:57] [INFO] [AbstractRestClient.js:413] onEnd() called for rest client ZosmfRestClient

Thank you

Bill Pereira
Developer Advocate
—
billpereira@br.ibm.com


IBM