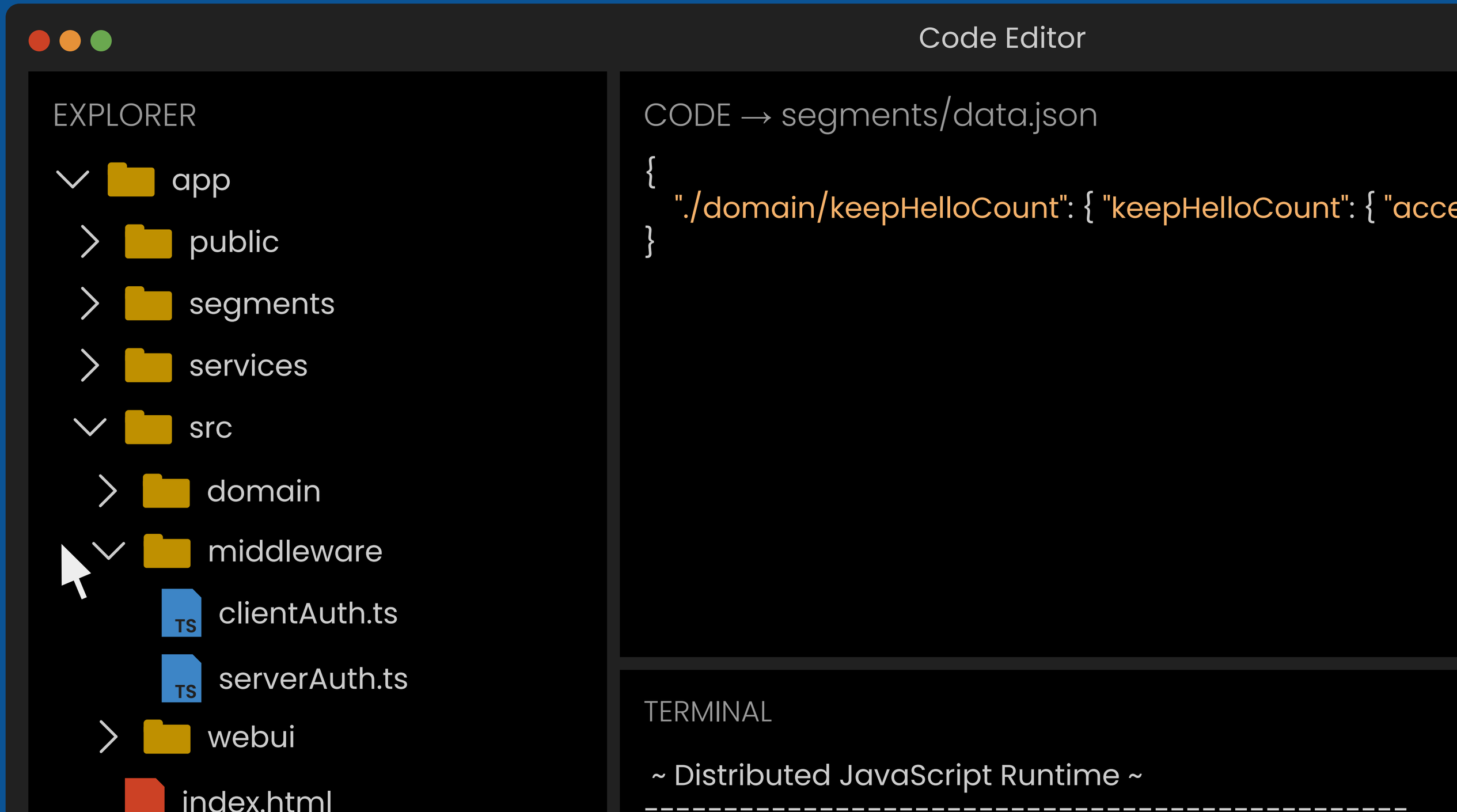
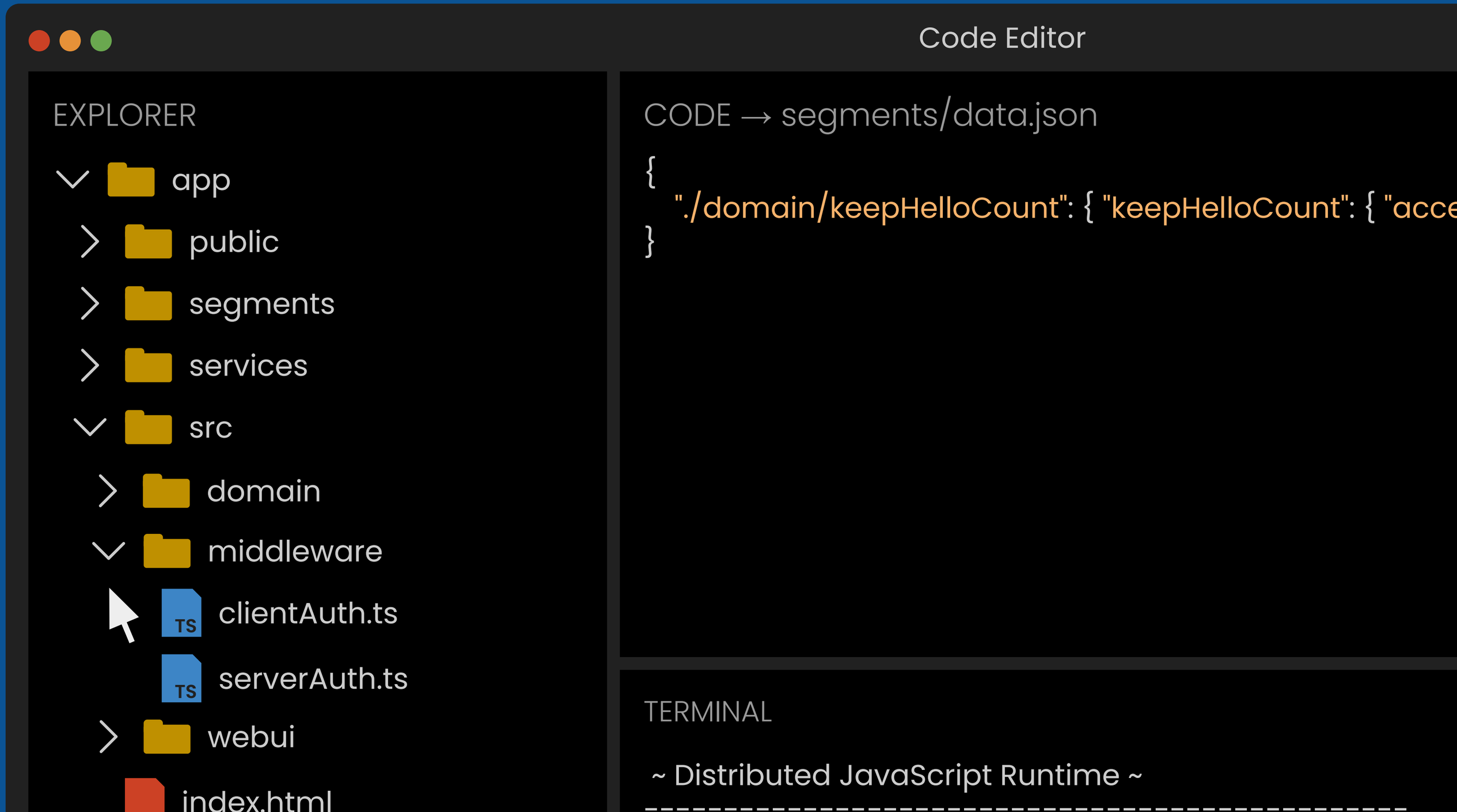

Code Editor
EXPLORER
app
public
segments
services
src
domain
middleware
clientAuth.ts
serverAuth.ts
webui
index.html
CODE → segments/data.json
{
"./domain/keepHelloCount": { "keepHelloCount": { "acce
}
TERMINAL
~ Distributed JavaScript Runtime ~


Code Editor
EXPLORER
app
public
segments
services
src
domain
middleware
clientAuth.ts
serverAuth.ts
webui
index.html
CODE → segments/data.json
{
"./domain/keepHelloCount": { "keepHelloCount": { "acce
}
TERMINAL
~ Distributed JavaScript Runtime ~

Code Editor

EXPLORER

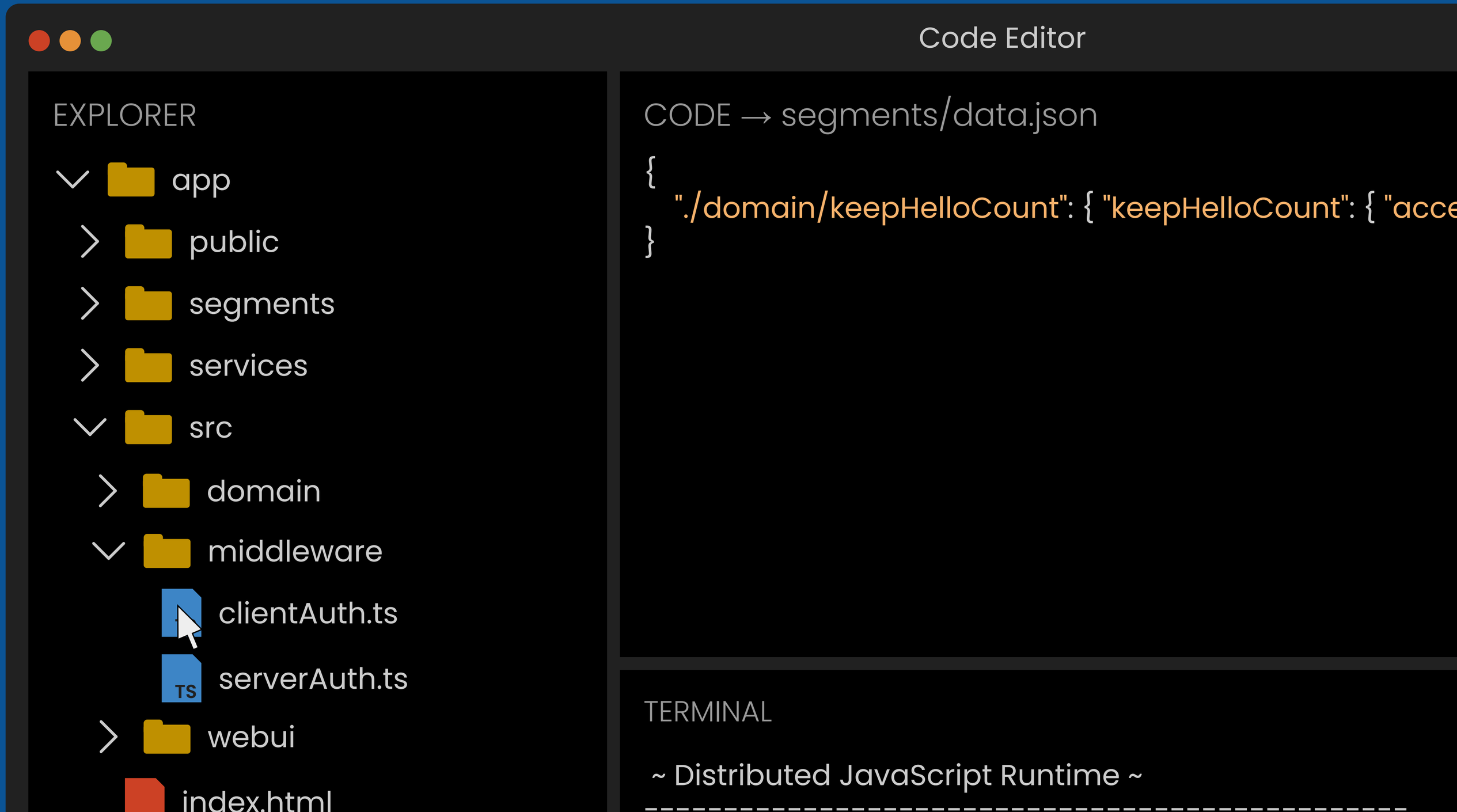
- app
  - public
  - segments
  - services
  - src
    - domain
    - middleware
      - clientAuth.ts
      - serverAuth.ts
    - webui
  - index.html

CODE → segments/data.json

```
{
  "./domain/keepHelloCount": { "keepHelloCount": { "acce
}
```

TERMINAL

~ Distributed JavaScript Runtime ~


Code Editor
EXPLORER
app
public
segments
services
src
domain
middleware
clientAuth.ts
serverAuth.ts
webui
index.html
CODE → segments/data.json
{
"./domain/keepHelloCount": { "keepHelloCount": { "acce
}
TERMINAL
~ Distributed JavaScript Runtime ~

Code Editor

EXPLORER

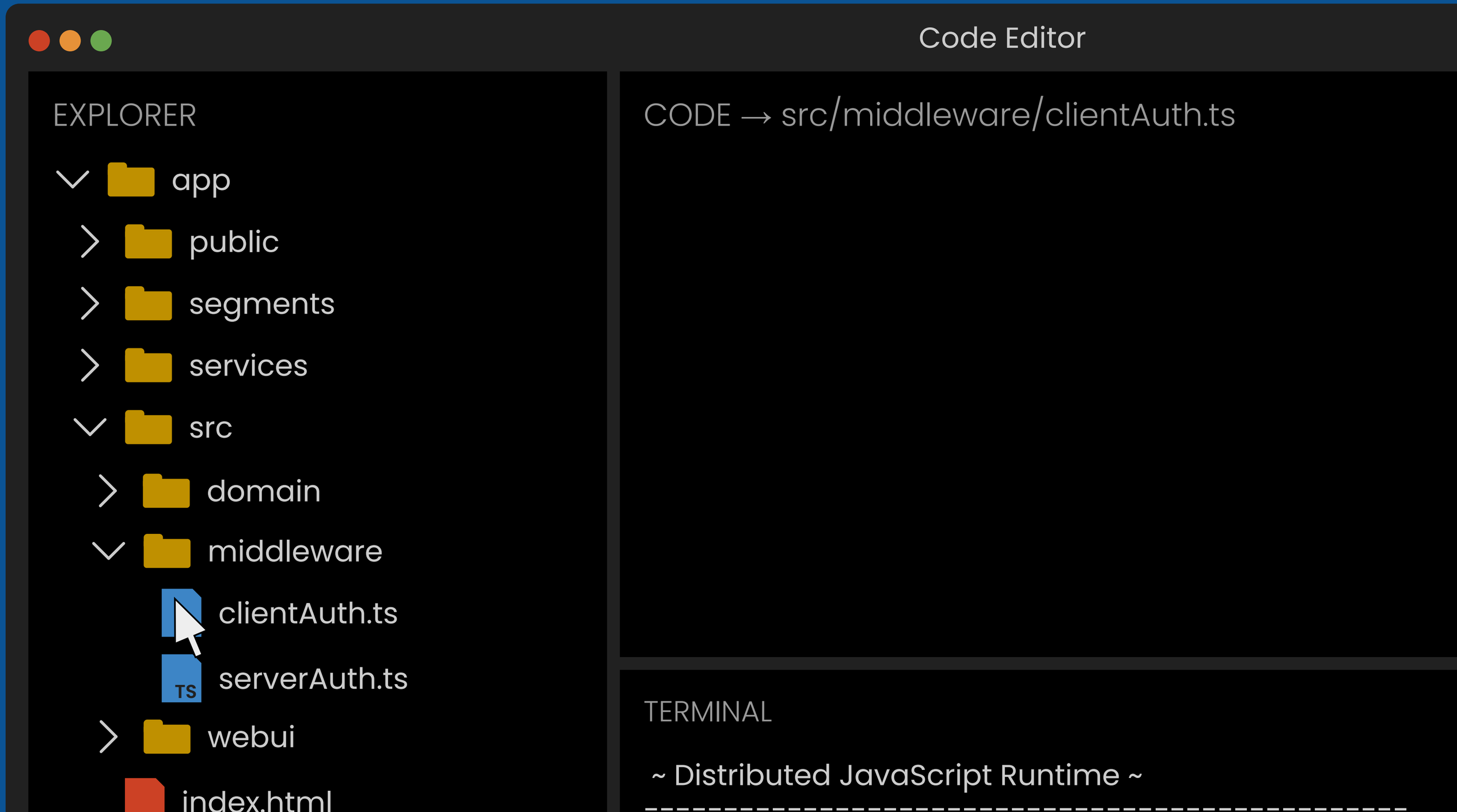
- app
  - public
  - segments
  - services
  - src
    - domain
    - middleware
      - clientAuth.ts
      - serverAuth.ts
    - webui
  - index.html

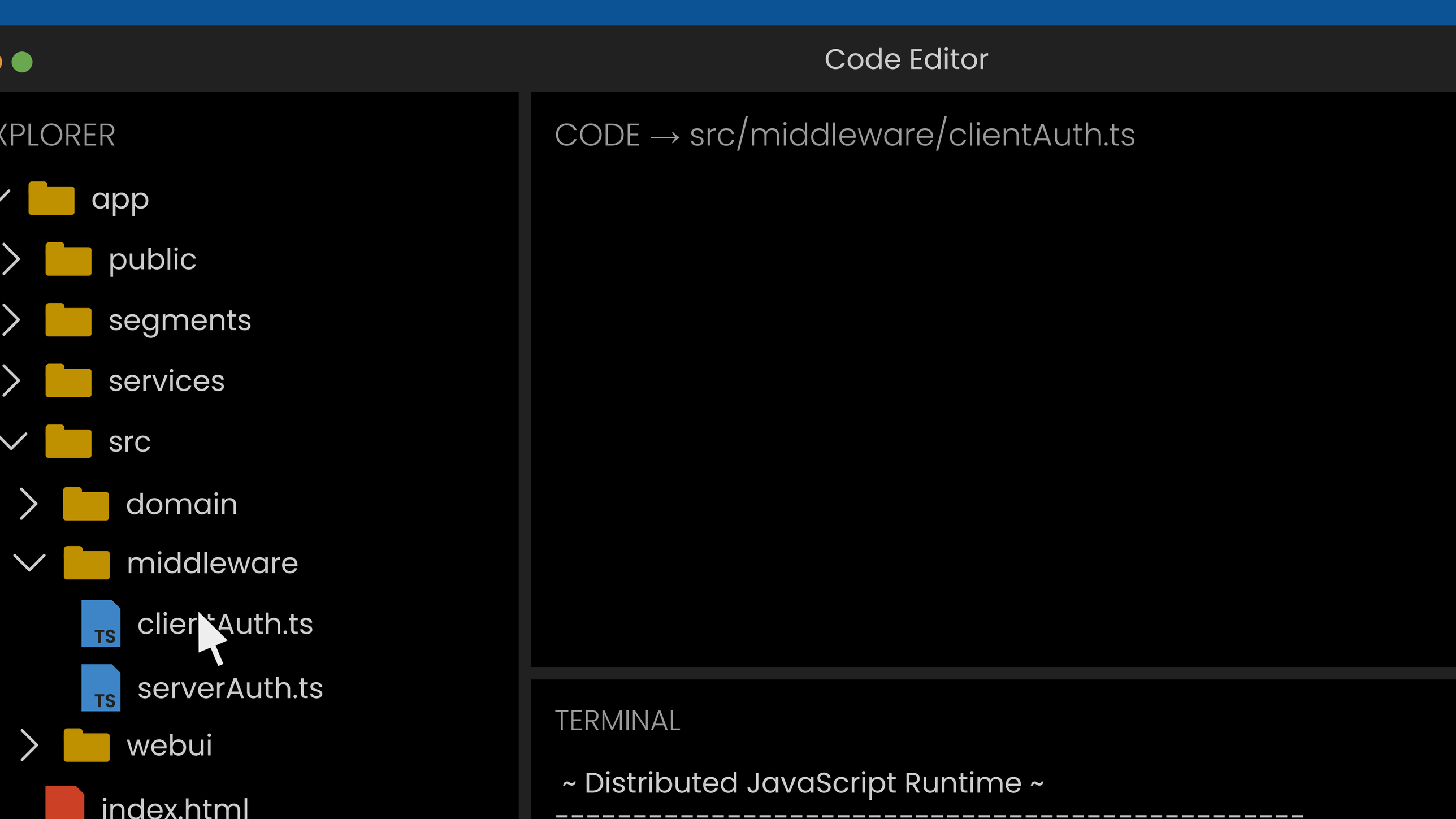
CODE → segments/data.json

```
{
  "./domain/keepHelloCount": { "keepHelloCount": { "acce
}
```

TERMINAL

~ Distributed JavaScript Runtime ~


Code Editor
EXPLORER
app
public
segments
services
src
domain
middleware
clientAuth.ts
serverAuth.ts
webui
index.html
CODE → src/middleware/clientAuth.ts
TERMINAL
~ Distributed JavaScript Runtime ~


Code Editor
XPLORER
app
public
segments
services
src
domain
middleware
clientAuth.ts
serverAuth.ts
webui
index.html
CODE → src/middleware/clientAuth.ts
TERMINAL
~ Distributed JavaScript Runtime ~

Code Editor

R

pp

public

segments

services

src

domain

middleware

clientAuth.ts

serverAuth.ts

webui

ndex.html

CODE → src/middleware/clientAuth.ts

TERMINAL

~ Distributed JavaScript Runtime ~

Code Editor

nts

s

n

eware

Auth.ts

rAuth.ts

ml

CODE → src/middleware/clientAuth.ts

TERMINAL

~ Distributed JavaScript Runtime ~

Code Editor

CODE → src/middleware/clientAuth.ts

TERMINAL

~ Distributed JavaScript Runtime ~

---

gatewa

default

data

Code Editor

CODE → src/middleware/clientAuth.ts

TERMINAL

~ Distributed JavaScript Runtime ~

---

gateway

default

data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway

default

data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

gateway

default

data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway

default

data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway

default

data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
-----------------------------------
```

gateway

default

data

Code Editor

CODE → src/middleware/clientAuth.ts

```
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

- gateway
- default
- data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------------
```

gateway
default
data

Code Editor

CODE → src/middleware/clientAuth.ts

```ts
import type { Middleware, NextHandler, Request, Response } from 'jitar';

class ClientAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        request.setHeader('Authorization', 'Richard');

        return next();
    }
}

export default new ClientAuth();
```

TERMINAL

```
 ~ Distributed JavaScript Runtime ~
--------------------------------------
```

gateway

default

data