Code Editor

CODE → src/middleware/serverAuth.ts

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

class ServerAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        const name = request.getHeader('Authorization');
        request.setArgument('name', name);

        return next();
    }
}

export default new ServerAuth();
```

TERMINAL

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

- gateway
- default
- data

Code Editor

CODE → src/middleware/serverAuth.ts

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

class ServerAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        const name = request.getHeader('Authorization');
        request.setArgument('name', name);

        return next();
    }
}

export default new ServerAuth();
```

TERMINAL

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

gateway

default

data

Code Editor

CODE → src/middleware/serverAuth.ts

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

class ServerAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        const name = request.getHeader('Authorization');
        request.setArgument('name', name);

        return next();
    }
}

export default new ServerAuth();
```

TERMINAL

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

- gateway
- default
- data

Code Editor

CODE → src/middleware/serverAuth.ts

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

class ServerAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        const name = request.getHeader('Authorization');
        request.setArgument('name', name);

        return next();
    }
}

export default new ServerAuth();
```

TERMINAL

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

- gateway
- default
- data

Code Editor

CODE → src/middleware/serverAuth.ts

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

class ServerAuth implements Middleware
{
    async handle(request: Request, next: NextHandler): Promise<Response>
    {
        const name = request.getHeader('Authorization');
        request.setArgument('name', name);

        return next();
    }
}

export default new ServerAuth();
```

TERMINAL

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

- gateway
- default
- data

Code Editor

CODE → vite.config.ts

```
import { defineConfig } from 'vite';
import react from '@vitejs/plugin-react';
import jitar from '@jitar/plugin-vite';

export default defineConfig({
  build: {
    emptyOutDir: false
  },
  plugins: [
    react(),
    jitar({ sourceDir: 'src', targetDir: 'dist', jitarDir: 'domain', jitarUrl: 'http://localhost:3000',
segments: [] })
  ]
});
```

TERMINAL

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

gateway

default

data

Code Editor

CODE → vite.config.ts

```
import { defineConfig } from 'vite';
import react from '@vitejs/plugin-react';
import jitar from '@jitar/plugin-vite';

export default defineConfig({
  build: {
    emptyOutDir: false
  },
  plugins: [
    react(),
    jitar({
      sourceDir: 'src',
      targetDir: 'dist',
      jitarDir: 'domain',
```

TERMINAL

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

gateway
default
data

Code Editor

CODE → vite.config.ts

```
import react from '@vitejs/plugin-react';
import jitar from '@jitar/plugin-vite';

export default defineConfig({
  build: {
    emptyOutDir: false
  },
  plugins: [
    react(),
    jitar({
      sourceDir: 'src',
      targetDir: 'dist',
      jitarDir: 'domain',
      jitarUrl: 'http://localhost:3000',
```

TERMINAL

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

gateway

default

data

Code Editor

CODE → vite.config.ts

```ts
import jitar from '@jitar/plugin-vite';

export default defineConfig({
  build: {
    emptyOutDir: false
  },
  plugins: [
    react(),
    jitar({
      sourceDir: 'src',
      targetDir: 'dist',
      jitarDir: 'domain',
      jitarUrl: 'http://localhost:3000',
      segments: []
```

TERMINAL

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

gateway

default

data

Code Editor

CODE → vite.config.ts

```ts
export default defineConfig({
  build: {
    emptyOutDir: false
  },
  plugins: [
    react(),
    jitar({
      sourceDir: 'src',
      targetDir: 'dist',
      jitarDir: 'domain',
      jitarUrl: 'http://localhost:3000',
      segments: []
    })
```

TERMINAL

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

gateway
default
data

Code Editor

CODE → vite.config.ts

```
export default defineConfig({
  build: {
    emptyOutDir: false
  },
  plugins: [
    react(),
    jitar({
      sourceDir: 'src',
      targetDir: 'dist',
      jitarDir: 'domain',
      jitarUrl: 'http://localhost:3000',
      segments: []
    })
  ]
```

TERMINAL

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

gateway

default

data

Code Editor

CODE → vite.config.ts

```
export default defineConfig({
  build: {
    emptyOutDir: false
  },
  plugins: [
    react(),
    jitar({
      sourceDir: 'src',
      targetDir: 'dist',
      jitarDir: 'domain',
      jitarUrl: 'http://localhost:3000',

      segments: []
    })
```

TERMINAL

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

gateway
default
data

Code Editor

CODE → vite.config.ts

```
export default defineConfig({
  build: {
    emptyOutDir: false
  },
  plugins: [
    react(),
    jitar({
      sourceDir: 'src',
      targetDir: 'dist',
      jitarDir: 'domain',
      jitarUrl: 'http://localhost:3000',
      middleware: [ './middleware/clientAuth' ],
      segments: []
    })
```

TERMINAL

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

gateway
default
data

Code Editor

CODE → services/proxy.json

```
{
    "url": "http://127.0.0.1:3000",
    "proxy":
    {
        "gateway": "http://127.0.0.1:3001",
        "repository": "http://127.0.0.1:3002",
    }
}
```

TERMINAL

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

gateway
default
data

Code Editor

CODE → services/proxy.json

```
{
   "url": "http://127.0.0.1:3000",

   "proxy":
   {
      "gateway": "http://127.0.0.1:3001",
      "repository": "http://127.0.0.1:3002",
   }
}
```

TERMINAL

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

gateway
default
data

Code Editor

CODE → services/proxy.json

```
{
   "url": "http://127.0.0.1:3000",
   "middleware": [ "./middleware/serverAuth" ],
   "proxy":
   {
      "gateway": "http://127.0.0.1:3001",
      "repository": "http://127.0.0.1:3002",
   }
}
```

TERMINAL

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

gateway

default

data