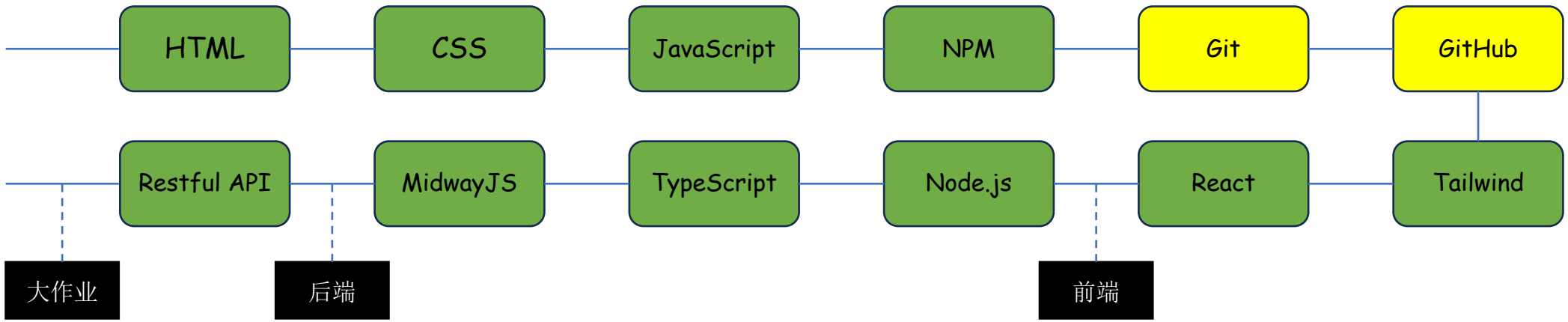
Web 持续集成、部署与性能优化

章许帆


HTML
CSS
JavaScript
NPM
Git
GitHub
Restful API
MidwayJS
TypeScript
Node.js
React
Tailwind
大作业
后端
前端

代码是如何部署成一个服务的?

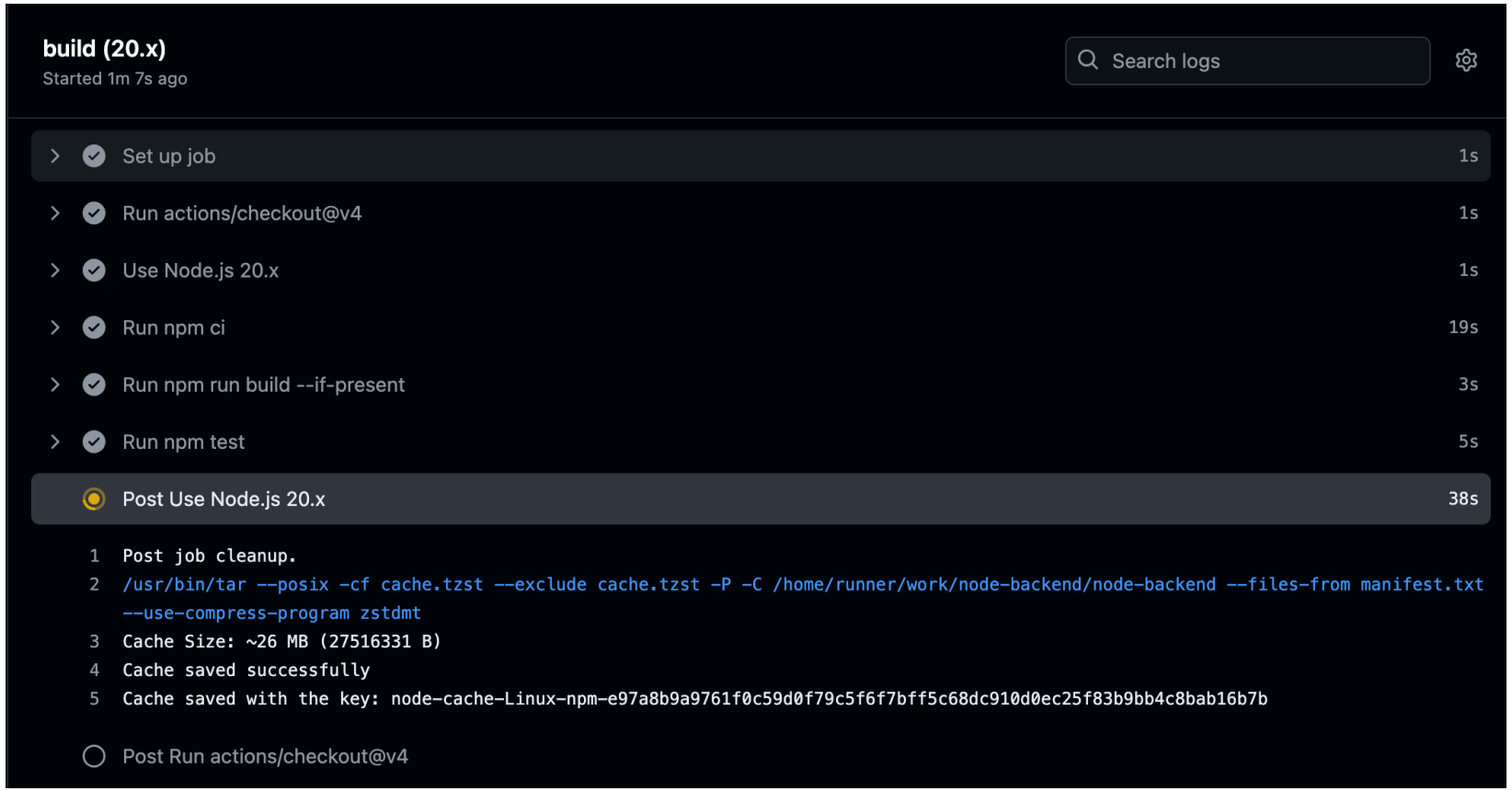
持续集成 GitHub Actions

https://docs.github.com/zh/actions/automating-builds-and-tests/building-and-testing-nodejs

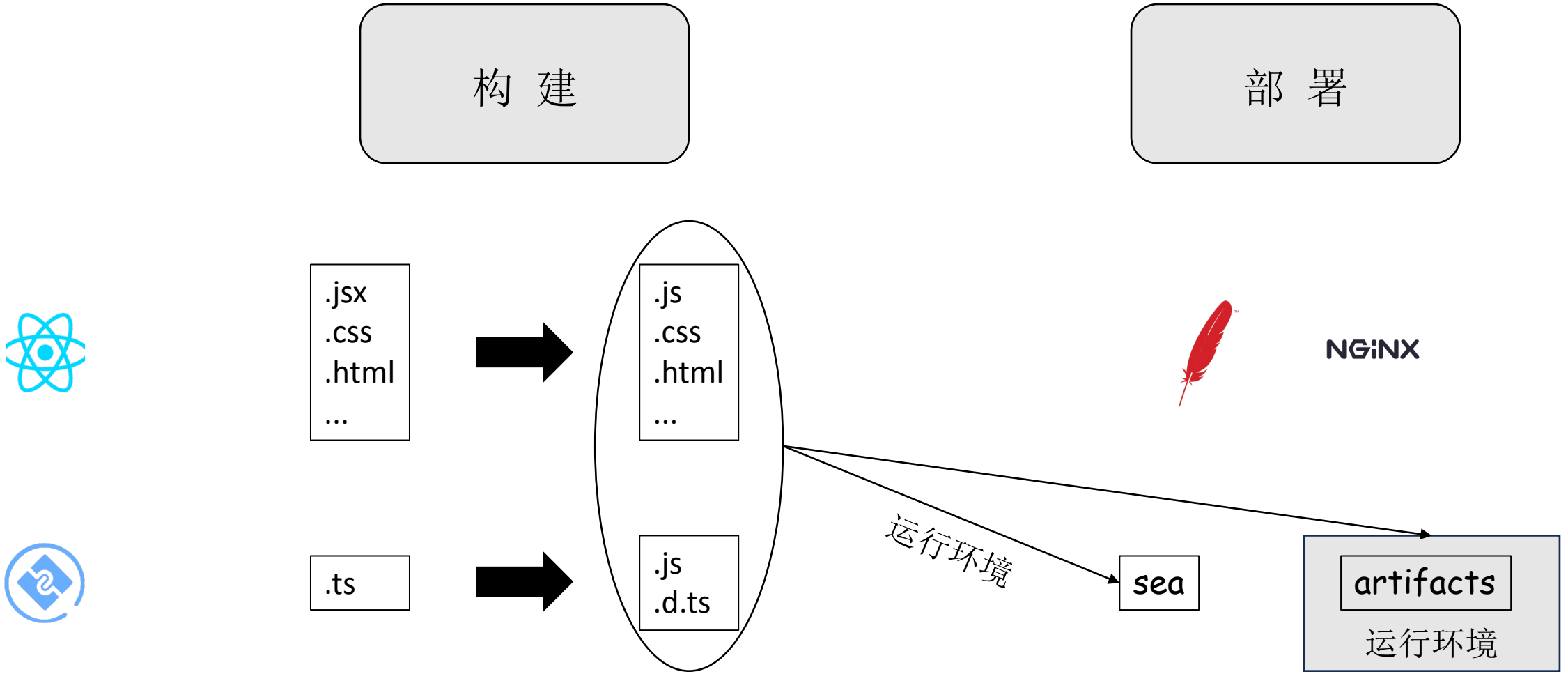
.github > workflows > ! node.js.yml

```yaml
name: Node.js CI
on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]
jobs:
  build:
    runs-on: ubuntu-latest
    strategy:
      matrix:
        node-version: [20.x]
    steps:
    - uses: actions/checkout@v4
    - name: Use Node.js ${{ matrix.node-version }}
      uses: actions/setup-node@v3
      with:
        node-version: ${{ matrix.node-version }}
        cache: 'npm'
    - run: npm ci
    - run: npm run build --if-present
    - run: npm test
```

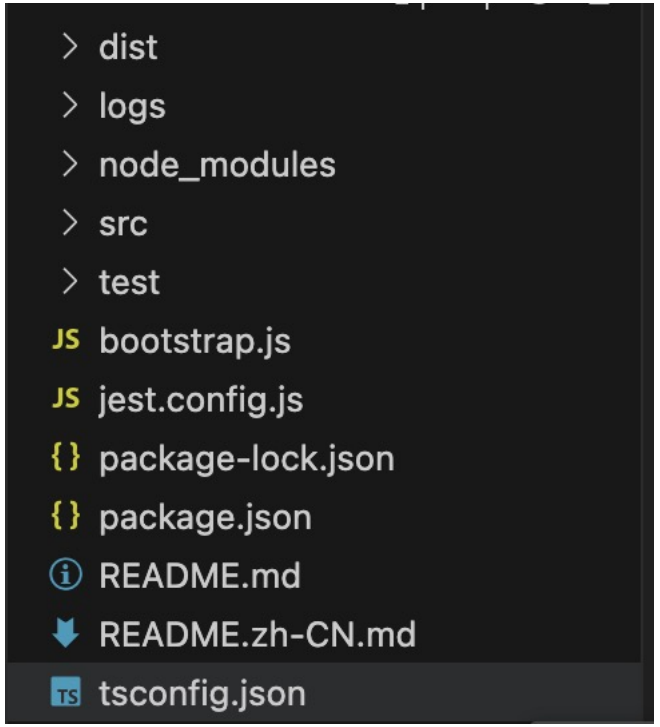
GitHub Actions



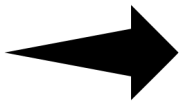
软件构建与部署

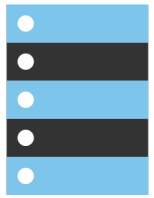

构 建
部 署
.jsx
.css
.html
...
.ts
.js
.css
.html
...
.js
.d.ts
运行环境
sea
NGINX
artifacts
运行环境

软件构建



- dist
- node_modules
- bootstrap.js
- package.json





Server

软件部署-pm2


```
$ pm2 start      # 启动一个服务
$ pm2 list       # 列出当前的服务
$ pm2 stop           # 停止某个服务
$ pm2 restart        # 重启某个服务
$ pm2 delete         # 删除某个服务
$ pm2 logs           # 查看服务的输出日志
```

pm2 start ./bootstrap.js --name [backend]

```
fan@Finleys-MacBook-Pro backend % pm2 list
```

id	name	mode	↺	status	cpu	memory
0	backend	fork	0	online	0%	57.6mb

软件部署-pm2

简单易用

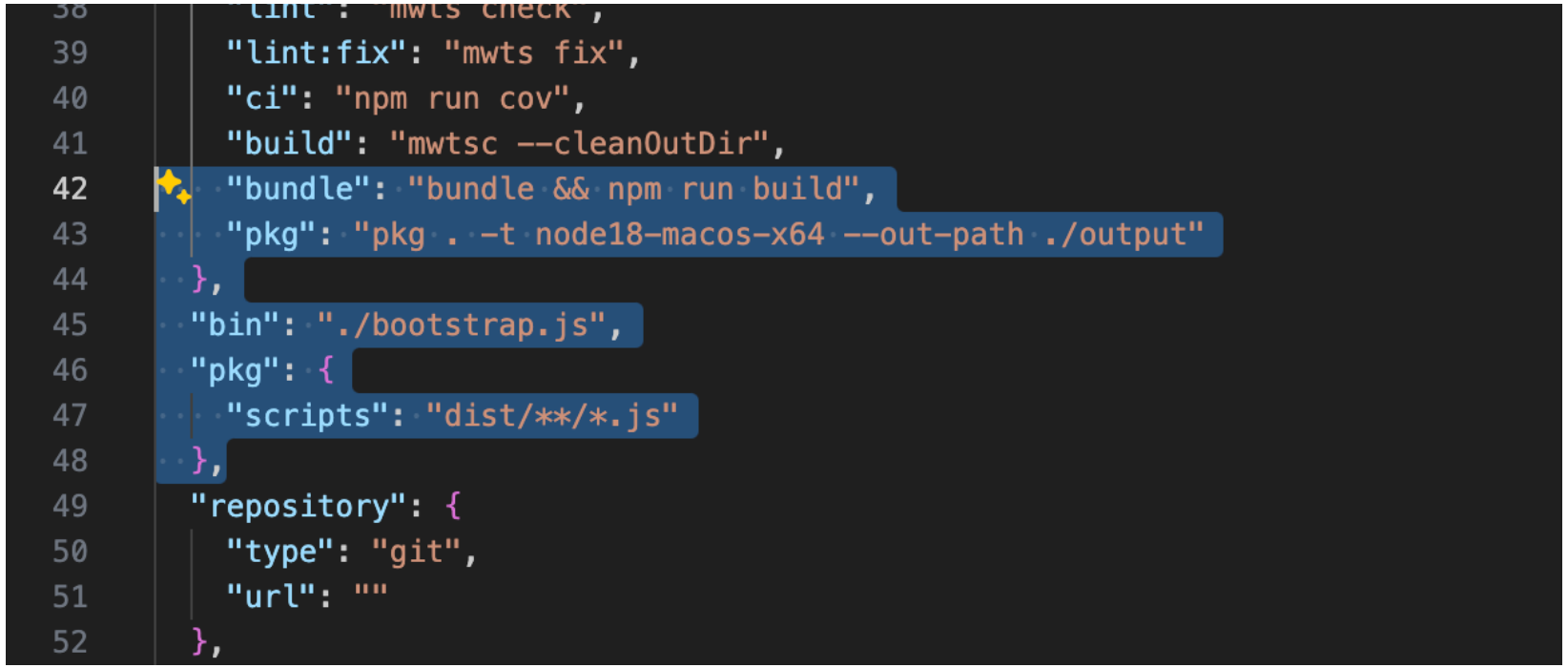
进程管理

自动重启

运行诊断

资源限制

软件部署-pkg


"lint": "mwts check",
"lint:fix": "mwts fix",
"ci": "npm run cov",
"build": "mwtsc --cleanOutDir",
"bundle": "bundle && npm run build",
"pkg": "pkg . -t node18-macos-x64 --out-path ./output"
},
"bin": "./bootstrap.js",
"pkg": {
"scripts": "dist/**/*.js"
},
"repository": {
"type": "git",
"url": ""
},

软件部署-pkg

部署简洁

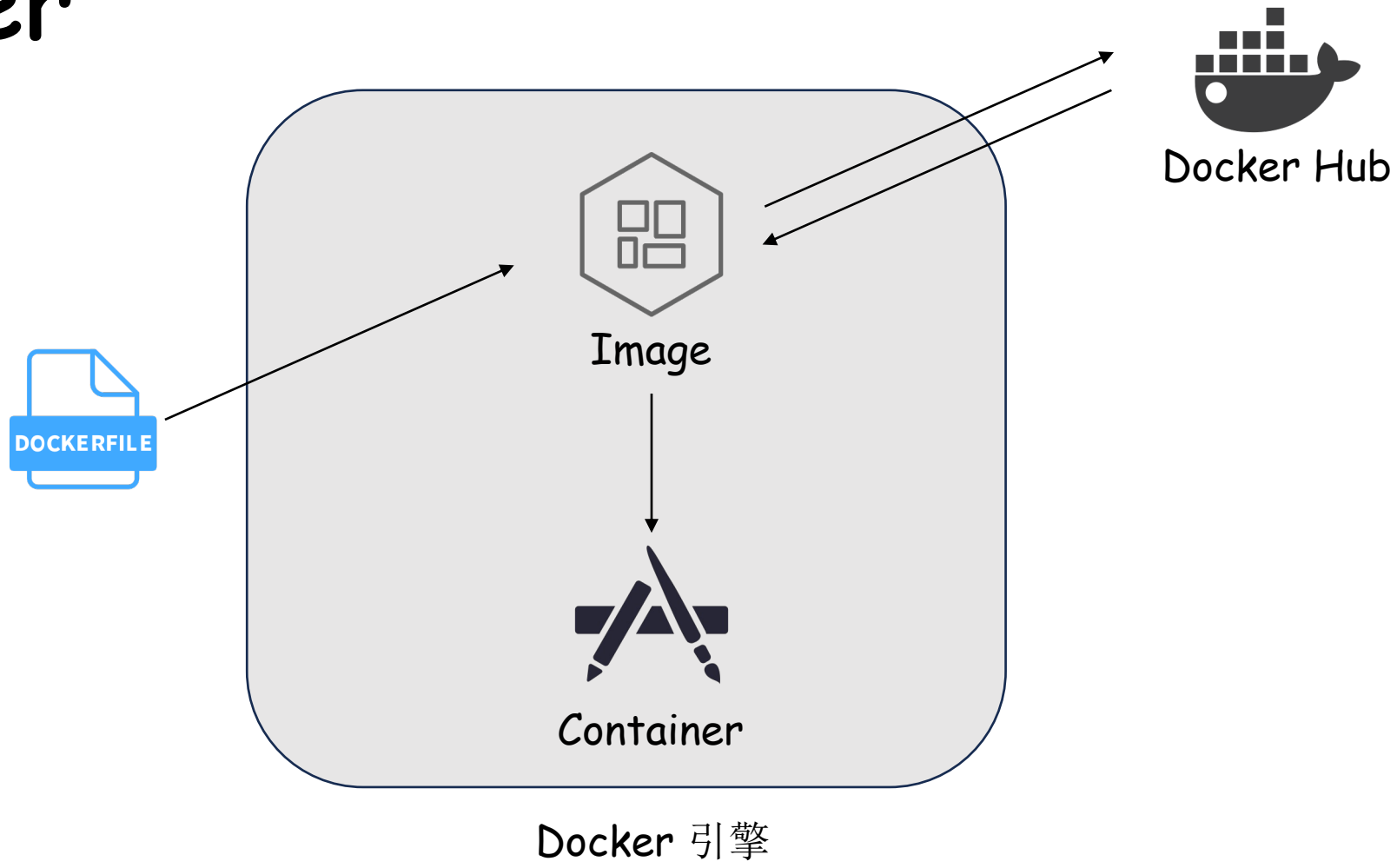
无需环境

易于分发

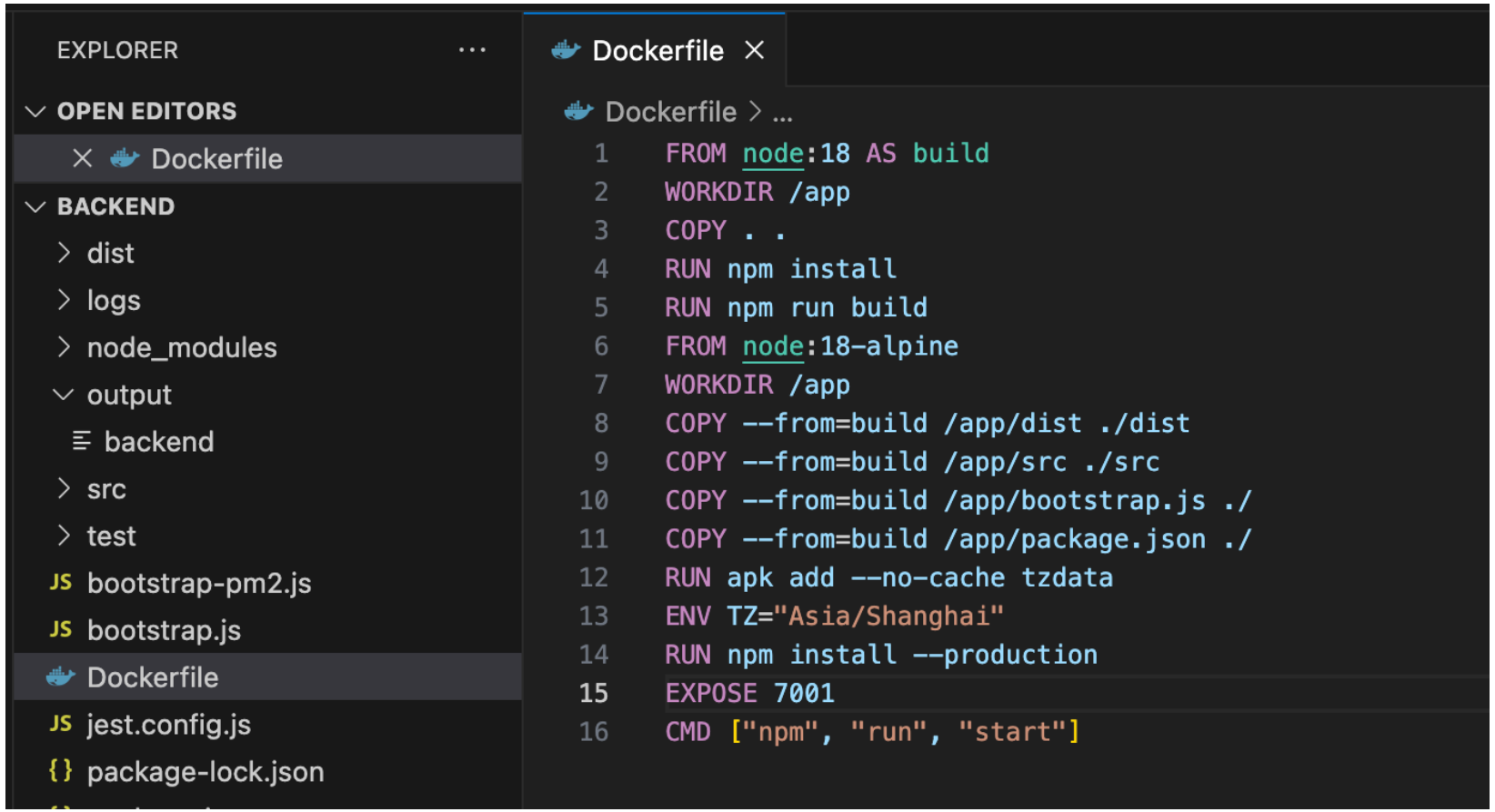
平台依赖

全量更新

docker



软件部署-docker


EXPLORER
OPEN EDITORS
Dockerfile
BACKEND
dist
logs
node_modules
output
backend
src
test
bootstrap-pm2.js
bootstrap.js
Dockerfile
jest.config.js
package-lock.json
Dockerfile
Dockerfile > ...
1 FROM node:18 AS build
2 WORKDIR /app
3 COPY . .
4 RUN npm install
5 RUN npm run build
6 FROM node:18-alpine
7 WORKDIR /app
8 COPY --from=build /app/dist ./dist
9 COPY --from=build /app/src ./src
10 COPY --from=build /app/bootstrap.js ./
11 COPY --from=build /app/package.json ./
12 RUN apk add --no-cache tzdata
13 ENV TZ="Asia/Shanghai"
14 RUN npm install --production
15 EXPOSE 7001
16 CMD ["npm", "run", "start"]

软件部署-docker

环境隔离

轻量

可移植

版本控制

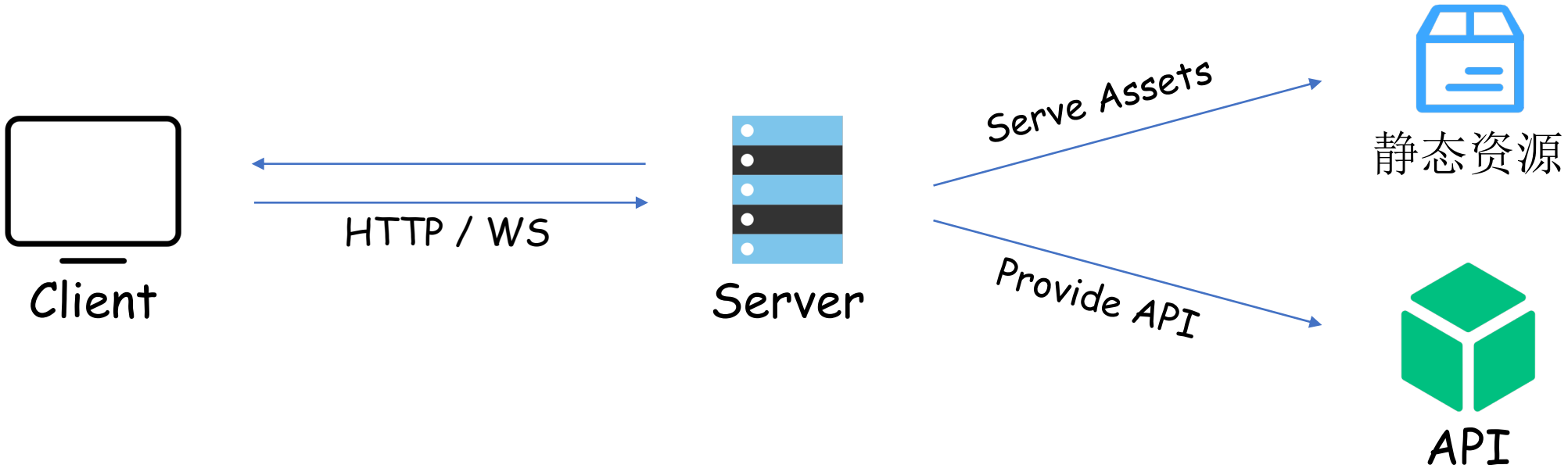
可复现

便于扩展

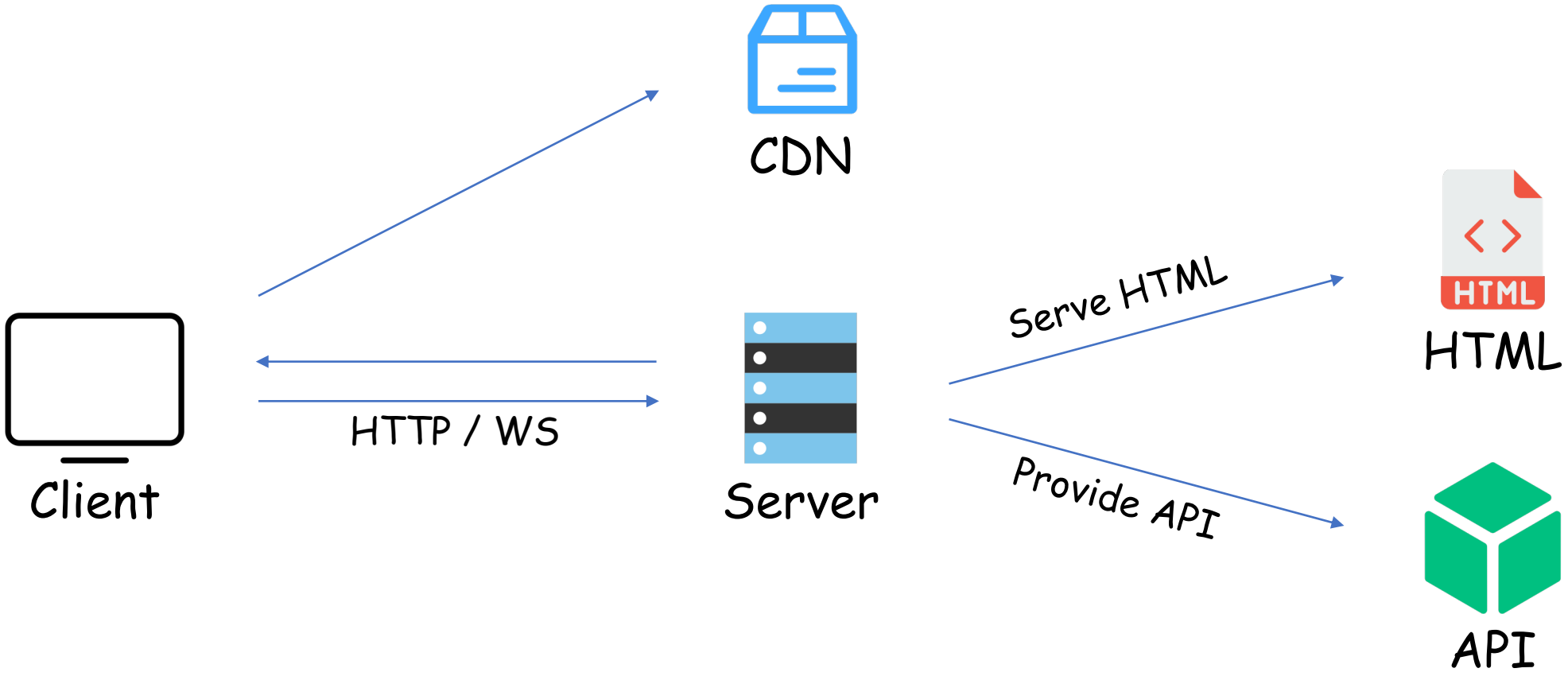
资源消耗

维护复杂

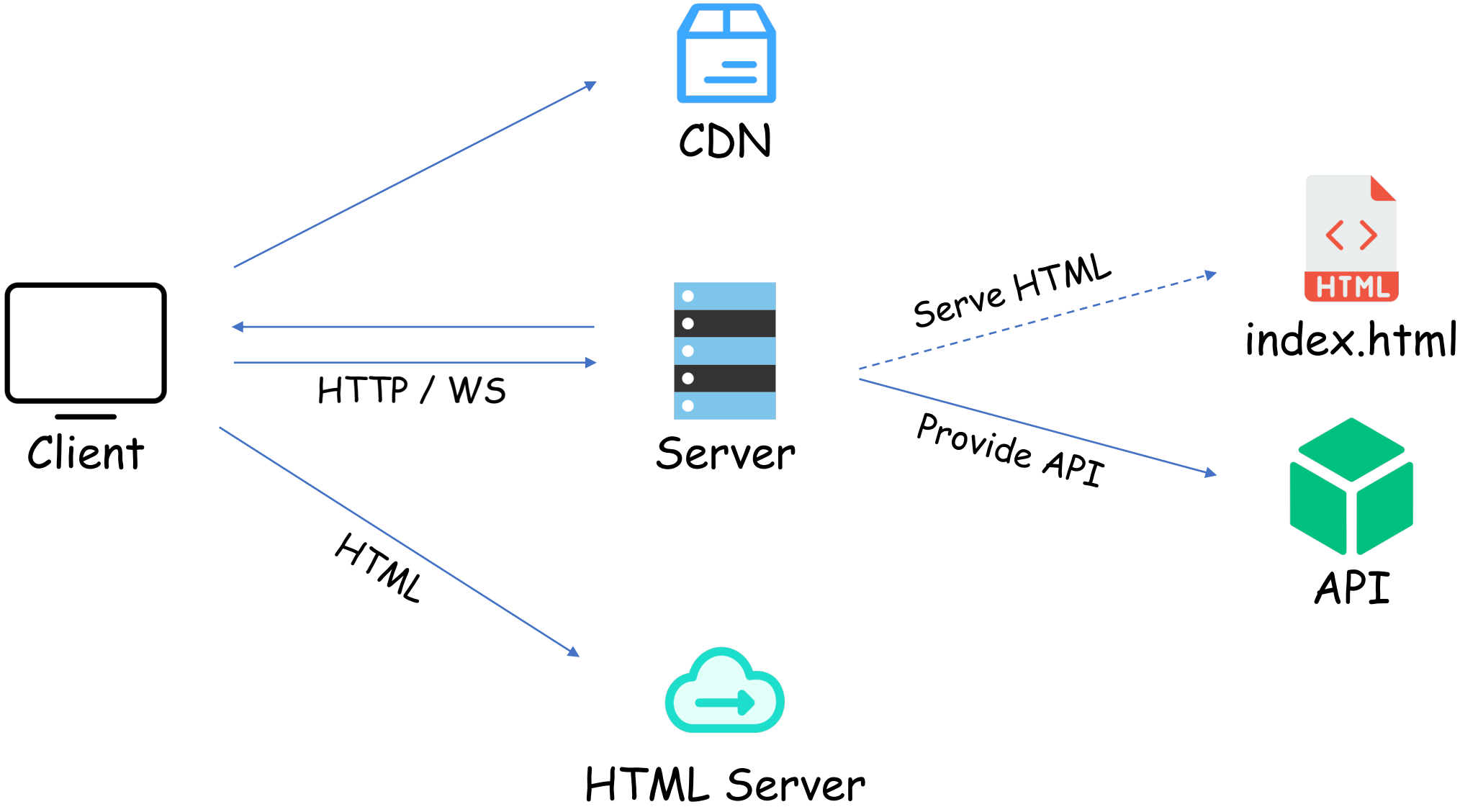
一体化部署



CDN + Server



CDN + HTML Server + Web Server



案例分析

一个有时间显示、有数据展示的 Web 界面，如果它响应用户的界面点击操作存在较大的延迟，结合本门课程中提到的Web应用相关的技术基础，分析一下，可能是由哪些原因造成的？

相关技术栈

https://nodejs.org/

https://vitejs.dev/guide/

https://react.dev/

https://tailwindcss.com/

https://midwayjs.org/

https://vitest.dev/guide/

https://jestjs.io/

https://developer.mozilla.org/en-US/docs/Web/API/WebSocket

https://pm2.io/

谢　谢