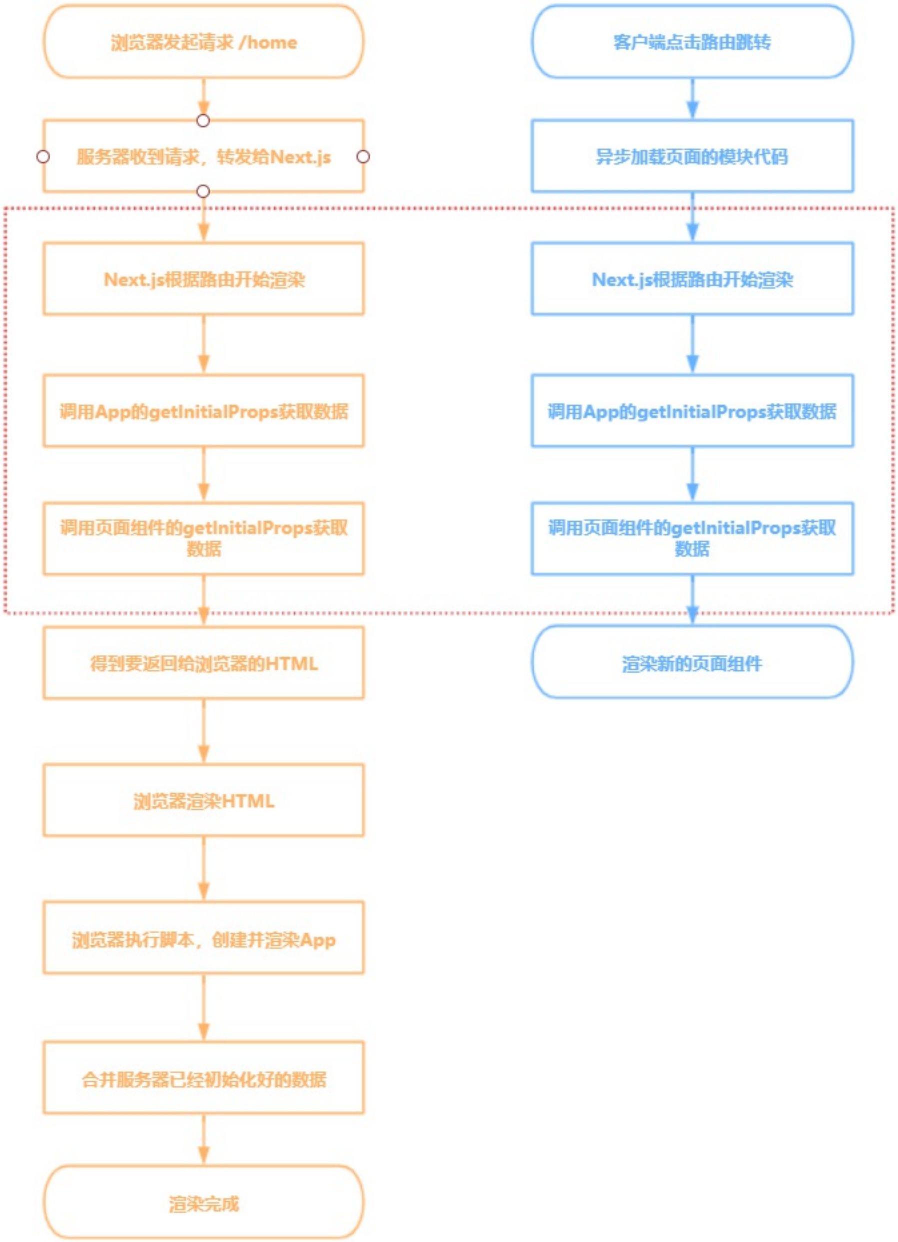
link: null
title: 珠峰架构师成长计划
description: null
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=194 sentences=532, words=3451

1. 同构

- 同构的项目支持客户端渲染和服务器端渲染
- 客户端渲染缺点
  - 首屏速度加载慢
  - 不支持SEO和搜索引擎优化
  - 首页需要通过请求初始化数据


浏览器发起请求 /home
服务器收到请求，转发给Next.js
Next.js根据路由开始渲染
调用App的getInitialProps获取数据
调用页面组件的getInitialProps获取数据
得到要返回给浏览器的HTML
浏览器渲染HTML
浏览器执行脚本，创建并渲染App
合并服务器已经初始化好的数据
渲染完成
客户端点击路由跳转
异步加载页面的模块代码
Next.js根据路由开始渲染
调用App的getInitialProps获取数据
调用页面组件的getInitialProps获取数据
渲染新的页面组件

2.Next.js

- Next.js英文文档 (https://nextjs.org/docs) Next.js中文文档 (https://nextjs.frontendx.cn/docs/) 是一个轻量级的 React 服务端渲染应用框架
- 默认支持服务端渲染
- 自动根据页面进行代码分割
- 基于页面的客户端路由方案
- 基于 Webpack 的开发环境，支持热模块替换
- 可以跟 `Koa`或者其它 `Node.js`服务器进行集成
- 支持 Babel 和 Webpack 的配置项定制
- 静态文件服务 public

3.初始化

3.1 创建项目

```
mkdir zhufengnextjs
cd zhufengnextjs
npm init -y
yarn add --dev typescript react @types/react react-dom @types/node next axios
mkdir pages
```

3.2 创建脚本

package.json

```
{
  "scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start"
  }
}
```

https://github.com/zeit/next.js/blob/canary/.gitignore (https://github.com/zeit/next.js/blob/canary/.gitignore)

3.3 访问

- 以 ./pages作为服务端的渲染和索引的根目录
- public静态文件服务映射
- pages组件代码自动分割

index.tsx

```
export default function () {
    return (
        <div>Homediv>
    )
}
```

```
curl http:
```

4.跑通路由

4.1 知识点

- 绑定 styled-jsx 来生成独立作用域的 CSS
- 如何支持 css
- 如何支持 antd并实现按需加载
- 路由的使用和两种跳转路径的方法

4.2 安装依赖包

```
yarn add @zeit/next-css antd babel-plugin-import
```

4.3 .babelrc

.babelrc

```
{
    "presets": [
        "next/babel"
    ],
    "plugins": [
        [
            "import",
            {
                "libraryName": "antd"
            }
        ]
    ]
}
```

4.4 next.config.js

- next-css (https://github.com/zeit/next-plugins/tree/master/packages/next-css)可以让你在项目中引用CSS

next.config.js

```
const withCSS = require('@zeit/next-css');

if (typeof require !== 'undefined') {
    require.extensions['.css'] = file => { }
}

module.exports = withCSS({})
```

4.5 pages_app.tsx

- App组件来初始化页面,你可以重写它来控制页面初始化
    - 当页面变化时保持页面布局
    - 当路由变化时保持页面状态
    - 注入额外数据到页面里

pages_app.tsx

```
import App, { Container } from 'next/app';
import Link from 'next/link';
import { Layout, Menu, Icon } from 'antd';
import 'antd/dist/antd.css';
import { withRouter } from 'next/router';
const { Header, Footer } = Layout;
class LayoutApp extends App<any> {
    render() {
        let { Component } = this.props as any;
        let pathname = this.props.router.pathname;
        pathname = '/' + (pathname.split('/')[1]);
        return (
            <Container>
                <style jsx>
                     {`a{display:inline-block!important;}`}
                style>
                <Layout>
                    <Header className="header">
                       <img src="/images/jglogo.png" style={{ width: 120, height: 31, margin: '16px 24px 16px 0px', float: 'left' }} />
                        <Menu theme="dark" mode="horizontal" style={{ lineHeight: '64px', display: 'inline-block' }}
                            selectedKeys={[pathname]}
                            defaultSelectedKeys={[pathname]}>
                            <Menu.Item key="/">
                                <Icon type="home" /><Link href="/"><a>首页a>Link>
                            Menu.Item>
                            <Menu.Item key="/user">
                                <Icon type="/user" /> <Link href="/user" ><a>用户管理a>Link>
                            Menu.Item>
                            <Menu.Item key="/profile">
                                <Icon type="profile" /><Link href="/profile"><a>个人中心a>Link>
                            Menu.Item>
                            <Menu.Item key="/login">
                                <Icon type="login" /><Link href="/login"><a>登录a>Link>
                            Menu.Item>
                        Menu>
                    Header>
                    <Component />
                    <Footer style={{ textAlign: 'center' }} >@copyright 珠峰架构Footer>
                Layout>
            Container>
        )
    }
}
export default withRouter(LayoutApp);
```

4.6 pages\index.tsx

pages\index.tsx

```
import router from 'next/router'
import { Button, Layout } from 'antd';
const { Content } = Layout;
export default function () {
    return (
        <Content  style={{ margin: '20px auto' }}>
            <div>Homediv>
            <Button onClick={() => router.push('/user')}>/userButton>
        Content>

    )
}
```

4.7 pages\user.tsx

pages\user.tsx

```
import router from 'next/router';
import { Button, Layout } from 'antd';
const { Content } = Layout;
export default function () {
    return (
        <Content  style={{ margin: '20px auto' }}>
            <div>Userdiv>
            <Button onClick={() => router.push('/profile')}>/profileButton>
        Content>
    )
}
```

4.8 pages\profile.tsx

pages\profile.tsx

```
import router from 'next/router';
import { Button, Layout } from 'antd';
const { Content } = Layout;
export default function () {
    return (
        <Content  style={{ margin: '20px auto' }}>
            <div>Profilediv>
            <Button onClick={() => router.back()}>返回Button>
        Content>
    )
}
```

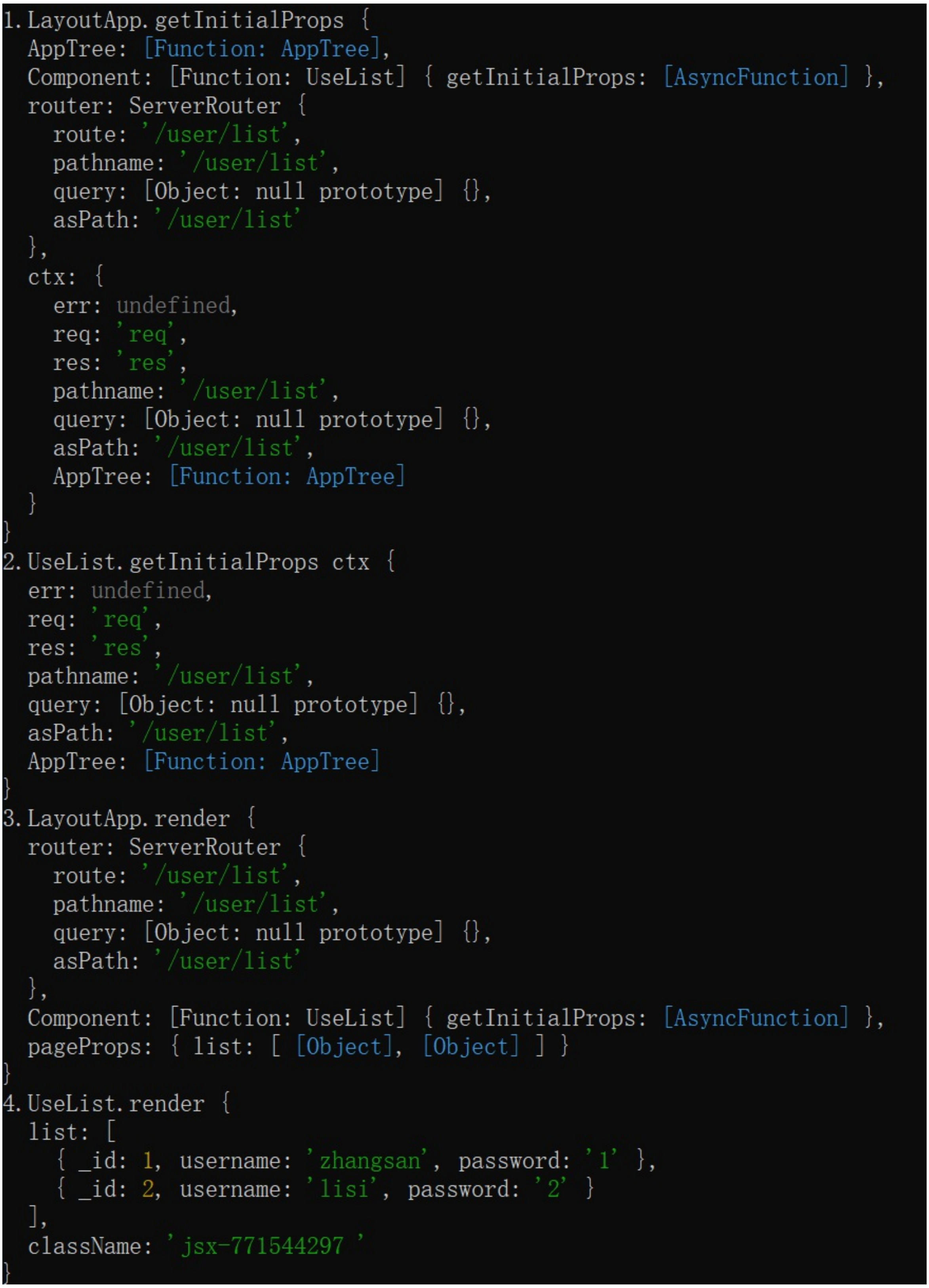
5.二级路由

5.1 知识点

- 支持二级路由
- 实现二级布局组件
- 路由跳转传递参数
- 页面组件通过 getInitialProps获取数据

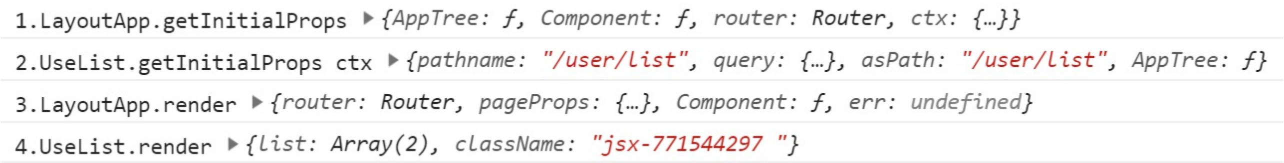
5.2 执行顺序

5.2.1 后台顺序



```
1.LayoutApp.getInitialProps {
  AppTree: [Function: AppTree],
  Component: [Function: UseList] { getInitialProps: [AsyncFunction] },
  router: ServerRouter {
    route: '/user/list',
    pathname: '/user/list',
    query: [Object: null prototype] {},
    asPath: '/user/list'
  },
  ctx: {
    err: undefined,
    req: 'req',
    res: 'res',
    pathname: '/user/list',
    query: [Object: null prototype] {},
    asPath: '/user/list',
    AppTree: [Function: AppTree]
  }
}
2.UseList.getInitialProps ctx {
  err: undefined,
  req: 'req',
  res: 'res',
  pathname: '/user/list',
  query: [Object: null prototype] {},
  asPath: '/user/list',
  AppTree: [Function: AppTree]
}
3.LayoutApp.render {
  router: ServerRouter {
    route: '/user/list',
    pathname: '/user/list',
    query: [Object: null prototype] {},
    asPath: '/user/list'
  },
  Component: [Function: UseList] { getInitialProps: [AsyncFunction] },
  pageProps: { list: [ [Object], [Object] ] }
}
4.UseList.render {
  list: [
    { _id: 1, username: 'zhangsan', password: '1' },
    { _id: 2, username: 'lisi', password: '2' }
  ],
  className: 'jsx-771544297 '
}
```

5.2.2 前台顺序



```
1.LayoutApp.getInitialProps ▶{AppTree: ƒ, Component: ƒ, router: Router, ctx: {…}}
2.UseList.getInitialProps ctx ▶{pathname: "/user/list", query: {…}, asPath: "/user/list", AppTree: ƒ}
3.LayoutApp.render ▶{router: Router, pageProps: {…}, Component: ƒ, err: undefined}
4.UseList.render ▶{list: Array(2), className: "jsx-771544297 "}
```

5.3 pages_app.tsx

pages_app.tsx

```
import App from 'next/app';
import Link from 'next/link';
import { Layout, Menu, Icon } from 'antd';
import 'antd/dist/antd.css';
import { withRouter } from 'next/router';
const { Header, Footer } = Layout;
class LayoutApp extends App {
+    static async getInitialProps(params) {
+        if (params.ctx.req) params.ctx.req = 'req';
+        if (params.ctx.res) params.ctx.res = 'res';
+        console.log('1.LayoutApp.getInitialProps', params);
+        let { Component, ctx } = params;
+        let pageProps = {};
+        if (Component.getInitialProps)
+            pageProps = await Component.getInitialProps(ctx);
+        return { pageProps };
+    }
    render() {
+        console.log('3.LayoutApp.render', this.props);
        let { Component } = this.props as any;
        let pathname = this.props.router.pathname;
        pathname = '/' + (pathname.split('/')[1]);
        return (
            <>

<span class="hljs-addition">+                    {`a{display:inline-block!important;}`}</span>

+
+
                            selectedKeys={[pathname]}
                            defaultSelectedKeys={[pathname]}>

                                首页

                                 用户管理

                                个人中心

+
                    @copyright 珠峰架构

            </>
        )
    }
}
export default withRouter(LayoutApp);
```

5.4 user\index.tsx

pages\user\index.tsx

```
import { withRouter } from 'next/router';
import Link from 'next/link';
import { Layout, Menu, Icon } from 'antd';
const { Sider, Content } = Layout;
function UserLayout(props) {
    return (
        <>
            <style jsx>
                 {`a{display:inline-block!important;}`}
            style>
            <Layout>
                <Sider>
                    <Menu
                        theme="dark"
                        mode="inline"
                        defaultSelectedKeys={[props.router.pathname]}
                    >
                        <Menu.Item key="/user/list">
                            <Icon type="user" /><Link href="/user/list"><a>用户列表a>Link>
                        Menu.Item>
                        <Menu.Item key="/user/add">
                            <Icon type="plus" /><Link href="/user/add"><a>添加用户a>Link>
                        Menu.Item>
                    Menu>
                Sider>
                <Content style={{ padding: 8 }}>
                    {props.children}
                Content>
            Layout>
        </>
    )
}
export default withRouter(UserLayout);
```

5.5 user\list.tsx

pages\user\list.tsx

```
import UserLayout from './index';
import { List } from 'antd';
import Link from 'next/link';
function UseList(props) {
    console.log('4.UseList.render', props);
    return (
        <UserLayout>
            <List
                header={<div>用户列表div>}
                footer={<div>共计多少{props.list.length}个用户div>}
                bordered
                dataSource={props.list}
                renderItem={(item: any) => (
                    <List.Item key={item._id}>
                        <Link as={`/user/detail/${item._id}`} href={{ pathname: `/user/detail`, query: { id: item._id } }}><a>{item.username}a>Link>
                    List.Item>
                )}
            />
        UserLayout>
    )
}

UseList.getInitialProps = async (ctx) => {
    if (ctx.req) { ctx.req = 'req'; }
    if (ctx.res) { ctx.res = 'res'; }
    console.log('2.UseList.getInitialProps ctx', ctx);
    let list = [{ _id: 1, username: 'zhangsan', password: '1' }, { _id: 2, username: 'lisi', password: '2' }];
    return { list };
}

export default UseList;
```

5.6 user\add.tsx

pages\user\add.tsx

```
import UserLayout from './index';
import router from 'next/router';
import { Form, Input, Button, Icon } from 'antd';
function UserAdd(props) {
    const { getFieldDecorator } = props.form;
    return (
        <UserLayout>
            <Form className="login-form" style={{ maxWidth: '300px' }}>
                <Form.Item>
                    {getFieldDecorator('username', {
                        rules: [{ required: true, message: 'Please input your username!' }],
                    })(
                        <Input
                            prefix={<Icon type="user" style={{ color: 'rgba(0,0,0,.25)' }} />}
                            placeholder="Username"
                        />,
                    )}
                Form.Item>
                <Form.Item>
                    {getFieldDecorator('password', {
                        rules: [{ required: true, message: 'Please input your Password!' }],
                    })(
                        <Input
                            prefix={<Icon type="lock" style={{ color: 'rgba(0,0,0,.25)' }} />}
                            type="password"
                            placeholder="Password"
                        />,
                    )}
                Form.Item>
                <Form.Item>
                    <Button type="primary" htmlType="submit" className="login-form-button">
                        添加用户
                    Button>
                Form.Item>
            Form>
        UserLayout>
    )
}
export default Form.create({ name: 'UserAdd' })(UserAdd);
```

6.集成koa

- Next自身的服务器只能处理SSR请求
- 无法处理API接口的请求

6.1 知识点

- 如何集成 koa
- 在 koa使用 next作中间件

6.2 安装依赖

```
yarn add koa koa-router
```

6.3 client\index.js

client\index.js

```
let Koa = require('koa');
let Router = require('koa-router');
const next = require('next');
const dev = process.env.NODE_ENV !== 'production';
const app = next({ dev: true });
const handler = app.getRequestHandler();
app.prepare().then(() => {
    const server = new Koa();
    let router = new Router();
    server.use(router.routes());
    server.use(async (ctx, next) => {
        await handler(ctx.req, ctx.res);
        ctx.response = false;
    });
    server.listen(3000, () => console.log('server started at port 3000'));
});
```

6.4 package.json

package.json

```
  "scripts": {
+   "client": "nodemon client",
    "build": "next build",
    "start": "next start"
  },
```

7.调用接口

- 添加用户
- 查看用户列表
- 当服务渲染时, getInitialProps将会把数据序列化，就像 JSON.stringify
- 所以确保 getInitialProps返回的是一个普通 JS 对象，而不是Date, Map 或 Set类型
- 当页面初始化加载时, getInitialProps只会加载在服务端。只有当路由跳转（Link组件跳转或 API 方法跳转）时，客户端才会执行 getInitialProps
- getInitialProps将不能使用在子组件中。只能使用在 pages页面中
- getInitialProps入参对象 (https://nextjs.frontendx.cn/docs/#%E8%B7%AF%E7%94%B1)
- 如果需要注入 pathname, query 或 asPath到你组件中，你可以使用 withRouter

7.1 utils\axios.tsx

utils\axios.tsx

```
import axios from 'axios';
axios.defaults.withCredentials = true
const instance = axios.create({
    baseURL: 'http://localhost:4000'
})
export default instance;
```

7.2 pages\user\add.tsx

pages\user\add.tsx

```
import UserLayout from './index';
import router from 'next/router';
+import { Form, Input, Button, Icon, message } from 'antd';
+import axios from '../../utils/axios';
function UserAdd(props) {
+    async function handleSubmit(event) {
+        event.preventDefault();
+        let values = props.form.getFieldsValue();
+        let response = await axios.post('/api/register', values);
+        if (response.data.code === 0) {
+            router.push('/user/list');
+        } else {
+            message.error('添加用户失败');
+        }
+    }
    const { getFieldDecorator } = props.form;
    return (

+
                {props.currentUser && 创建者:{props.currentUser.username}}

                    {getFieldDecorator('username', {
                        rules: [{ required: true, message: 'Please input your username!' }],
                    })(
                        }
                            placeholder="Username"
                        />,
                    )}

                    {getFieldDecorator('password', {
                        rules: [{ required: true, message: 'Please input your Password!' }],
                    })(
                        }
                            type="password"
                            placeholder="Password"
                        />,
                    )}

                        添加用户

    )
}
export default Form.create({ name: 'UserAdd' })(UserAdd);
```

7.3 user\list.tsx

pages\user\list.tsx

```
import UserLayout from './index';
import { List } from 'antd';
import Link from 'next/link';
+import axios from '../../utils/axios';
function UseList(props) {
    console.log('4.UseList.render', props);
    return (

            用户列表}
                footer={共计多少{props.list.length}个用户}
                bordered
                dataSource={props.list}
                renderItem={(item: any) => (

                        {item.username}

                )}
            />

    )
}

UseList.getInitialProps = async (ctx) => {
    if (ctx.req) { ctx.req = 'req'; }
    if (ctx.res) { ctx.res = 'res'; }
    console.log('2.UseList.getInitialProps ctx', ctx);
+    let response = await axios({ url: '/api/users', method: 'GET' });
+    return { list: response.data.data };
}

export default UseList;
```

8.用户详情

8.1 知识点

- 路由跳转传参数
- 模块和组件懒加载
- 自定义服务端路由

8.2 user\detail.tsx

pages\user\detail.tsx

```
import React, { useState } from 'react';
import UserLayout from './';
import { withRouter } from 'next/router';
import { Button } from 'antd';
import axios from '../../utils/axios';
import dynamic from 'next/dynamic';
const UserInfo = dynamic(import('../../components/UserInfo'))

function UserDetail(props) {
  let [show, setShow] = useState(false);
  return (
    <UserLayout>
      <p>ID: {props.router.query.id}p>
      <Button onClick={() => setShow(!show)}>显示/隐藏Button>
      {
        show && <UserInfo user={props.user} />
      }
    UserLayout>
  )
}
UserDetail.getInitialProps = async (ctx) => {
  let response = await axios({ url: `/api/users/${ctx.query.id}`, method: 'GET' });
  return { user: response.data.data };
}
export default withRouter(UserDetail);
```

8.2 components\UserInfo.tsx

components\UserInfo.tsx

```
import { useState } from 'react';
import { Button } from 'antd';
function UserInfo(props) {
    let [created, setCreated] = useState(props.user.created);
    async function changeFormat() {
        let moment = await import('moment');
        setCreated(moment.default(props.user.created).fromNow());
    }
    return (
        <>
            <p>用户名:{props.user.username}p>
            <p>密码:{props.user.password}p>
            <p>创建时间:{created}<Button onClick={changeFormat}>切换为相对时间Button>p>
        </>
    )
}
export default UserInfo;
```

8.3 client\index.js

client\index.js

```
let Koa = require('koa');
let Router = require('koa-router');
const next = require('next');
const dev = process.env.NODE_ENV !== 'production';
const app = next({ dev: true });
const handler = app.getRequestHandler();
app.prepare().then(() => {
    const server = new Koa();
    let router = new Router();
+    router.get('/user/detail/:id', async (ctx, next) => {
+        const id = ctx.params.id;
+        await handler(ctx.req, ctx.res, {
+            pathname: '/user/detail',
+            query: { id }
+        });
+        ctx.response = false;
+    });
    server.use(router.routes());
    server.use(async (ctx, next) => {
        await handler(ctx.req, ctx.res);
        ctx.response = false;
    });
    server.listen(3000, () => console.log('server started at port 3000'));
});
```

9.登录

9.1 pages\login.tsx

```
import router from 'next/router';
import { Form, Input, Button, Icon, message } from 'antd';
import axios from '../utils/axios';
function Login(props) {
    const { getFieldDecorator } = props.form;
    async function handleSubmit(event) {
        event.preventDefault();
        let values = props.form.getFieldsValue();
        let response = await axios.post('/api/login', values);
        if (response.data.code === 0) {
            router.push('/');
        } else {
            message.error('登录失败');
        }
    }
    return (
        <Form onSubmit={handleSubmit} className="login-form" style={{ maxWidth: '300px',margin:'20px auto' }}>
            <Form.Item>
                {getFieldDecorator('username', {
                    rules: [{ required: true, message: 'Please input your username!' }],
                })(
                    <Input
                        prefix={<Icon type="user" style={{ color: 'rgba(0,0,0,.25)' }} />}
                        placeholder="Username"
                    />,
                )}
            Form.Item>
            <Form.Item>
                {getFieldDecorator('password', {
                    rules: [{ required: true, message: 'Please input your Password!' }],
                })(
                    <Input
                        prefix={<Icon type="lock" style={{ color: 'rgba(0,0,0,.25)' }} />}
                        type="password"
                        placeholder="Password"
                    />,
                )}
            Form.Item>
            <Form.Item>
                <Button type="primary" htmlType="submit" className="login-form-button">
                    登录
                Button>
            Form.Item>
        Form>
    )
}
const WrappedLogin = Form.create({ name: 'Login' })(Login);
export default WrappedLogin;
```

10.集成redux

10.1 安装依赖

```
yarn add redux react-redux
```

10.2 pages_app.tsx

pages_app.tsx

```
import App from 'next/app';
import Link from 'next/link';
+import { Layout, Menu, Icon, Avatar } from 'antd';
import 'antd/dist/antd.css';
import { withRouter } from 'next/router';
+import axios from '../utils/axios';
+import initStore from '../store';
+import { Provider } from 'react-redux';
+import * as TYEPS from '../store/action-types';
+const { Header, Footer, Content } = Layout;
+declare global {
+    interface Window {
+        _redux_store_: any
+    }
+}
+function getStore(initialState) {
+    if (typeof window == 'undefined') {
+        return initStore(initialState);
+    } else {
+        if (!window._redux_store_) {
+            window._redux_store_ = initStore(initialState);
+        }
+        return window._redux_store_;
+    }
+}
class LayoutApp extends App {
+    store: any
+    constructor(props) {
+        super(props);
+        this.store = getStore(props.initialState);
+    }
+    static async getInitialProps({ Component, ctx }) {
+        let store = getStore({});
+        let currentUser;
+        let options: any = { url: '/api/currentUser' };
+        if (ctx.req&&ctx.req.headers.cookie) {
+            (options.headers = options.headers || {}).cookie = ctx.req.headers.cookie;
+        }
+        let response = await axios(options);
+        if (response.data.code == 0) {
+            currentUser = response.data.data;
+            store.dispatch({ type: TYEPS.SET_USER_INFO, payload: currentUser });
+        }
        let pageProps = {};
        if (Component.getInitialProps)
            pageProps = await Component.getInitialProps(ctx);

+        return { pageProps, currentUser, initialState: store.getState() };
    }
    render() {
        let { Component } = this.props as any;
        let pathname = this.props.router.pathname;
        pathname = '/' + (pathname.split('/')[1]);
        return (
+

                    {`a{display:inline-block!important;}`}

                                首页

                                 用户管理

                                个人中心

                                登录

+                        {
+                            this.props.currentUser &&
+                                {this.props.currentUser.username}
+
+                        }

                    @copyright 珠峰架构

+
        )
    }
}
export default withRouter(LayoutApp);
```

10.3 user\add.tsx

pages\user\add.tsx

```
import UserLayout from './index';
import router from 'next/router';
import { Form, Input, Button, Icon, message } from 'antd';
import axios from '../../utils/axios';
+import { connect } from 'react-redux';
function UserAdd(props) {
    async function handleSubmit(event) {
        event.preventDefault();
        let values = props.form.getFieldsValue();
        let response = await axios.post('/api/register', values);
        if (response.data.code
            router.push('/user/list');
        } else {
            message.error('添加用户失败');
        }
    }
    const { getFieldDecorator } = props.form;
    return (

                    {getFieldDecorator('username', {
                        rules: [{ required: true, message: 'Please input your username!' }],
                    })(
                        }
                            placeholder="Username"
                        />,
                    )}

                    {getFieldDecorator('password', {
                        rules: [{ required: true, message: 'Please input your Password!' }],
                    })(
                        }
                            type="password"
                            placeholder="Password"
                        />,
                    )}

                        添加用户

    )
}
+const WrappedUserAdd = Form.create({ name: 'UserAdd' })(UserAdd);
+export default WrappedUserAdd;
```

10.4 action-types.tsx

store\action-types.tsx

```
export const SET_USER_INFO = 'SET_USER_INFO';
```

10.5 store\index.tsx

store\index.tsx

```
import { createStore } from 'redux';
import * as TYPES from './action-types';
let initState = {
    currentUser: null
}
const reducer = (state = initState, action) => {
    switch (action.type) {
        case TYPES.SET_USER_INFO:
            return { currentUser: action.payload }
        default:
            return state;
    }
}

export default function initStore(initialState) {
    const store = createStore(reducer, initialState);
    return store;
}
```

11.loading效果

- 路由事件 (https://nextjs.frontendx.cn/docs/#%E8%B7%AF%E7%94%B1)

事件 触发时机 routeChangeStart(url) 路由开始切换时触发 routeChangeComplete(url) 完成路由切换时触发 routeChangeError(err, url) 路由切换报错时触发 beforeHistoryChange(url) 浏览器 history 模式开始切换时触发 hashChangeStart(url) 开始切换 hash 值但是没有切换页面路由时触发 hashChangeComplete(url) 完成切换 hash 值但是没有切换页面路由时触发

11.1 pages_app.tsx

```
import App from 'next/app';
import Link from 'next/link';
import { Layout, Menu, Icon, Avatar } from 'antd';
import 'antd/dist/antd.css';
import { withRouter } from 'next/router';
import axios from '../utils/axios';
import initStore from '../store';
import { Provider } from 'react-redux';
import * as TYEPS from '../store/action-types';
+import router from 'next/router';
+import { Spin } from 'antd';
const { Header, Footer, Content } = Layout;
declare global {
    interface Window {
        _redux_store_: any
    }
}
function getStore(initialState) {
    if (typeof window == 'undefined') {
        return initStore(initialState);
    } else {
        if (!window._redux_store_) {
            window._redux_store_ = initStore(initialState);
        }
        return window._redux_store_;
    }
}
class LayoutApp extends App {
    store: any
+    state = { loading: false }
+    routeChangeStart: any
+    routeChangeComplete: any
    constructor(props) {
        super(props);
        this.store = getStore(props.initialState);
    }
    static async getInitialProps({ Component, ctx }) {
        let store = getStore({});
        let currentUser;
        let options: any = { url: '/api/currentUser' };
        if (ctx.req) {
            (options.headers = options.headers || {}).cookie = ctx.req.headers.cookie;
        }
        let response = await axios(options);
        if (response.data.code == 0) {
            currentUser = response.data.data;
            store.dispatch({ type: TYEPS.SET_USER_INFO, payload: currentUser });
        }
        let pageProps = {};
        if (Component.getInitialProps)
            pageProps = await Component.getInitialProps(ctx);

        return { pageProps, currentUser, initialState: store.getState() };
    }
+    componentDidMount() {
+        this.routeChangeStart = (url) => {
+            this.setState({ loading: true });
+        };
+        router.events.on('routeChangeStart', this.routeChangeStart);
+        this.routeChangeComplete = (url) => {
+            this.setState({ loading: false });
+        };
+        router.events.on('routeChangeComplete', this.routeChangeComplete);
+    }
    componentWillUnmount() {
        router.events.off('routeChangeStart', this.routeChangeStart)
        router.events.off('routeChangeStart', this.routeChangeComplete)
    }
    render() {
        let { Component, pageProps, currentUser } = this.props as any;
        let pathname = this.props.router.pathname;
        pathname = '/' + (pathname.split('/')[1]);
        return (

                    {`a{display:inline-block!important;}`}

                                首页

                                 用户管理

                                个人中心

                                登录

+                        {
+                            this.props.currentUser &&
+                                {this.props.currentUser.username}
+
                        }

                    {
                        this.state.loading ?  :
                    }
                    @copyright 珠峰架构

        )
    }
}
export default withRouter(LayoutApp);
```

12.受保护路由

12.1 pages\profile.tsx

pages\profile.tsx

```
import router from 'next/router';
import { Button, Layout } from 'antd';
+import axios from '../utils/axios';
const { Content } = Layout;
+function Profile(props) {
    return (

+           当前登录用户:{props.currentUser.username}
             router.back()}>返回

    )
}
+Profile.getInitialProps = async function (ctx) {
+    let options: any = { url: '/api/currentUser' };
+    if (ctx.req&&ctx.req.headers.cookie) {
+        (options.headers = options.headers || {}).cookie = ctx.req.headers.cookie;
+    }
+    let response = await axios(options);
+    if (response.data.code == 0) {
+        return {};
+    } else {
+        if (ctx.req) {
+            ctx.res.writeHead(303, { Location: '/login' })
+            ctx.res.end()
+        } else {
+            router.push('/login');
+        }
+        return {};
+    }
+}
+
+const WrappedProfile = connect(
+    state => state
+)(Profile);
+export default WrappedProfile;
```

13.自定义Document

13.1 pages_document.js

pages_document.js

```
import Document, { Html, Head, Main, NextScript } from 'next/document';
class CustomDocument extends Document {
    static async getInitialProps(ctx) {
        const props = await Document.getInitialProps(ctx);
        return { ...props };
    }
    render() {
        return (
            <Html>
                <Head>
                    <style>
                        {
                            `
                            *{
                                padding:0;
                                margin:0;
                            }
                            `
                        }
                    style>
                Head>
                <body>
                    <Main />
                    <NextScript />
                body>
            Html>
        )
    }
}
export default CustomDocument;
```

13.2 pages\index.tsx

pages\index.tsx

```
import router from 'next/router'
import { Button, Layout } from 'antd';
import Head from 'next/head'
const { Content } = Layout;
export default function () {
    return (
        <>
            <Head>
                <title>首页title>
                <meta name="description" content="这是首页" />
            Head>
            <Content style={{ margin: '20px auto' }}>
                <div>Homediv>
                <Button onClick={() => router.push('/user')}>/userButton>
            Content>
        </>
    )
}
```

14.服务端

```
yarn add express  body-parser  cors express-session connect-mongo mongoose
```

14.1 api\index.js

```
let express = require("express");
let bodyParser = require("body-parser");
let cors = require("cors");
let Models = require('./db');
let session = require("express-session");
let config = require('./config');
let MongoStore = require('connect-mongo')(session);
let app = express();
app.use(
    cors({
        origin: ['http://localhost:3000'],
        credentials: true,
        allowedHeaders: "Content-Type,Authorization",
        methods: "GET,HEAD,PUT,PATCH,POST,DELETE,OPTIONS"
    })
);
app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());
app.use(
    session({
        secret: config.secret,
        resave: false,
        saveUninitialized: true,
        store: new MongoStore({
            url: config.dbUrl,
            mongoOptions: {
                useNewUrlParser: true,
                useUnifiedTopology: true
            }
        })
    })
);
app.get('/api/users', async (req, res) => {
    let users = await Models.UserModel.find();
    res.send({ code: 0, data: users });
});
app.get('/api/users/:id', async (req, res) => {
    let user = await Models.UserModel.findById(req.params.id);
    res.send({ code: 0, data: user });
});
app.post('/api/register', async (req, res) => {
    let user = req.body;
    user = await Models.UserModel.create(user);
    res.send({ code: 0, data: user });
});
app.post('/api/login', async (req, res) => {
    let user = req.body;
    let dbUser = await Models.UserModel.findOne(user);
    if (dbUser) {
        req.session.currentUser = dbUser;
        res.send({ code: 0, data: dbUser });
    } else {
        res.send({ code: 1, error: '登录失败' });
    }
});

app.get('/api/currentUser', async (req, res) => {
    let currentUser = req.session.currentUser;
    if (currentUser) {
        res.send({ code: 0, data: currentUser });
    } else {
        res.send({ code: 1, error: '当前用户未登录' });
    }
});
app.listen(4000, () => {
    console.log('服务器在4000端口启动!');
});
```

14.2 api\config.js

api\config.js

```
module.exports = {
    secret: 'zhufeng',
    dbUrl: "mongodb://127.0.0.1/zhufengnext2"
}
```

14.3 api\db.js

api\db.js

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
const ObjectId = Schema.Types.ObjectId;
const config = require('./config');
const conn = mongoose.createConnection(config.dbUrl, { useNewUrlParser: true, useUnifiedTopology: true });
const UserModel = conn.model('User', new Schema({
    username: { type: String },
    password: { type: String }
}, { timestamps: { createdAt: 'created', updatedAt: 'updated' } }));

module.exports = {
    UserModel
}
```