项目实战01

课堂目标

1. UI库选型思路
2. 全家桶融会贯通vue-router+vuex
3. 前端登录和权限控制
4. 深入理解令牌认证机制
5. 前后端交互
6. 解决跨域问题

预习资源

1. UI库：cube-ui
2. 后端接口编写：koa
3. 请求后端接口：axios

   着重关注请求、响应拦截
4. 令牌机制：Bearer Token
5. 代理配置、mock数据：vue-cli配置指南、webpack配置指南

开发环境

1. vscode下载
2. node.js下载

知识点

1. UI库选型
2. 登录模块开发
   - cube-ui表单使用
   - 校验规则文档

```
  <cube-form
    :model="model"
    :schema="schema"
    @submit="handleLogin"
    @validate="haneldValidate"
  >
  </cube-form>
```

3. 登录接口模拟

```
app.get("/api/login", function(req,res){
  const {username, passwd} = req.query
  if(username=='kaikeba' && passwd =='123'){
    res.json({
      code:0,
      token:'kaiekbazhenbucuo-'+(new Date().getTime()+1000*60)
    })
  }else{
    res.json({
      code:1,
      message:"用户名或者密码错误"
    })
  }
})
```

4. 登录状态保存vuex

```
import Vue from 'vue'
import Vuex from 'vuex'

Vue.use(Vuex)

export default new Vuex.Store({
  state: {
    token:""
  },
  mutations: {
    settoken(state,token){
      state.token = token
    }
  },
  actions: {

  }
})
```

5. axios拦截的使用

```
axios.interceptors.request.use(
    config=>{
      if(store.state.token){
```

```
        config.headers.token = store.state.token
      }
      return config
    }
)

axios.interceptors.response.use(
    response=>{
      if(response.status==200){
        const data = response.data
        if(data.code==-1){
          // 登录过期了
          // 注销登录 清空vuex和localstorage
          store.commit('settoken','')
          localStorage.removeItem('token')
          // 跳转login
          router.replace({
            path:'login'
          })
          // store.commit(LOGOUT,'')
        }
        return data
      }
      return response
    }
)
```

6. 深入理解令牌机制bearer token
7. 真实接口实现koa
8. 请求代理

```
devServer: {
    proxy: {
      '/api': {
        target: 'http://localhost:3000',   //代理目标
        changeOrigin: true, // 是否改变请求源
        pathRewrite: { // 路径是否重写
          '^/api': '/api'
        }
      }
    }
}
```