Axios 请求实例配置

1 导入依赖

```
import axios from 'axios';
import { ElMessage } from 'element-plus';
import { useTokenStore } from '@/stores/token.js';
import router from '@/router';
```

2 创建 Axios 实例

```
const baseURL = '/api';
const instance = axios.create({ baseURL });
```

3 添加请求拦截器

在发送请求之前，检查并添加 `token` 以进行身份验证。

```
instance.interceptors.request.use(
    (config) => {
        const tokenStore = useTokenStore();
        if (tokenStore.token) {
            config.headers.Authorization = tokenStore.token;
        }
        return config;
    },
    (err) => {
        return Promise.reject(err);
    }
);
```

3.1 功能说明

- **Token 添加**: 在请求头中添加 `Authorization` 字段，附带用户的 `token`。
- **错误处理**: 如果请求配置出错，将错误传递到后续处理。

4 添加响应拦截器

在接收到服务器响应后，根据业务逻辑处理响应结果。

```
instance.interceptors.response.use(
    (result) => {
        if (result.data.code === 0) {
            return result.data;
        } else {
            ElMessage.error(result.data.msg ? result.data.msg : '服务异常');
            return Promise.reject(result.data);
        }
    },
    (err) => {
        if (err.response.status === 401) {
            ElMessage.error('请先登录');
            router.push('/login');
        } else {
            ElMessage.error('服务异常');
```

```
        }
        return Promise.reject(err);
    }
);
```

4.1 功能说明

- **成功状态处理**: 如果状态码为 `0`，则返回业务数据。
- **失败状态处理**: 根据返回的状态码显示错误信息，未登录时跳转到登录页面。

5 导出 Axios 实例

```
export default instance;
```