

模块化拆分

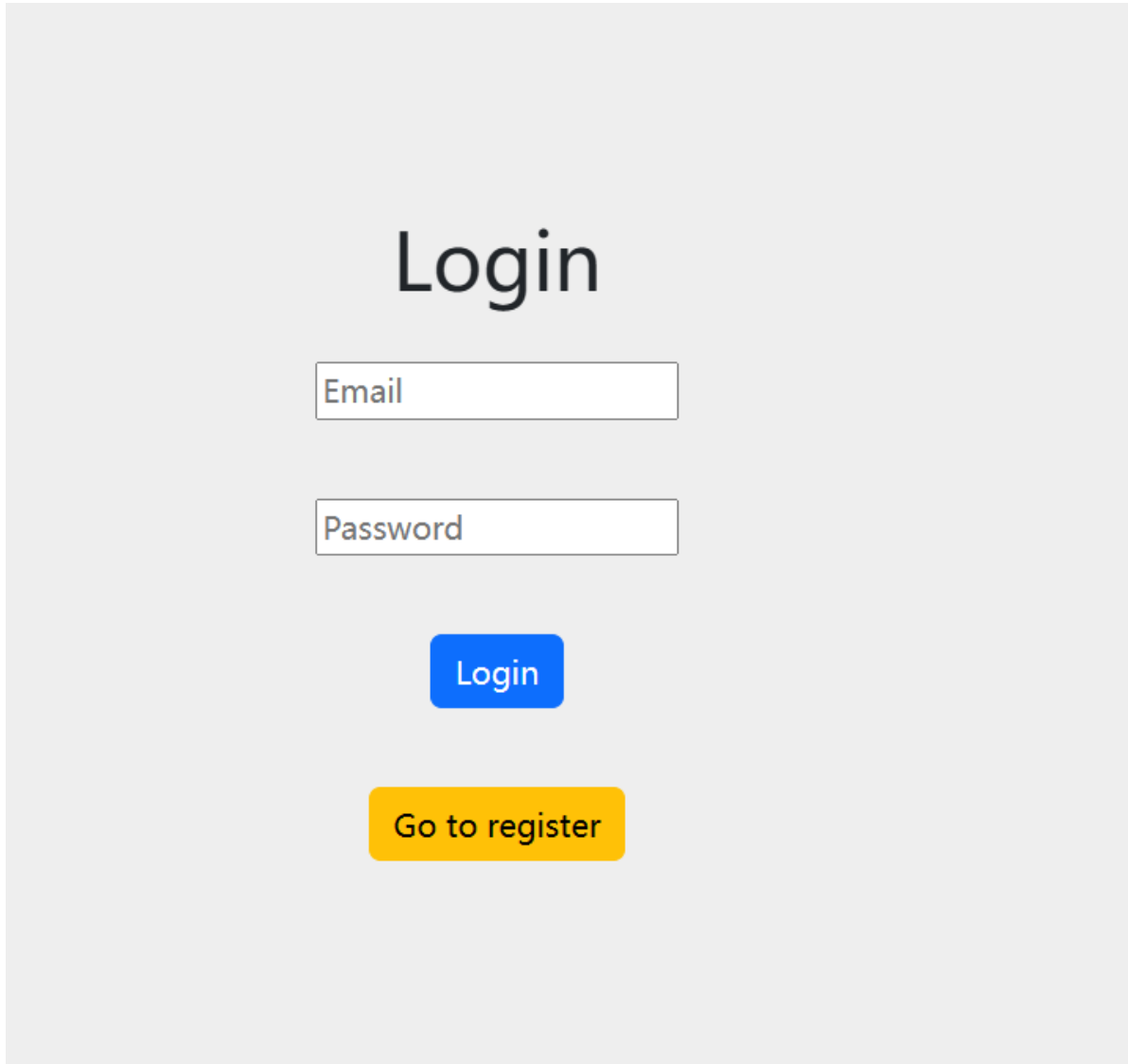
根据题意将网站主要分为三个模块，分别为登录注册、消息、频道

登录注册

将登录注册分别用login和register表实现，且登录注册时可以相互跳转，符合实际应用。

且密码输入处均设有隐私保护功能，可实现*号输入。

登录界面：

A login interface mockup on a light gray background. At the top, the word "Login" is displayed in a large, black, sans-serif font. Below it, there are two white input fields with thin gray borders. The first field is labeled "Email" in a small, gray, sans-serif font. The second field is labeled "Password" in a small, gray, sans-serif font. Below the input fields, there are two buttons. The first button is blue with rounded corners and the word "Login" in white, sans-serif font. The second button is yellow with rounded corners and the text "Go to register" in black, sans-serif font.

注册界面：

Register Page

Email

Name

Password

Confirm Password

Register

频道

登录之后会显示当前所有的频道，优先显示已经加入的频道。频道简单信息展示包括频道名称和公共情况。若你在频道内则会显示leave离开按钮，若不在则会显示join加入按钮。最上方有频道创建按钮，填写频道名称、频道简介以及私有是否就可创建自己的频道。

log-out

CreateChannel

sdxcxzc

public

leave

public channel

public

leave

123

private

join

111

private

join

© Slackr

频道创建：

Create Channel

name

private☐true☒false

description

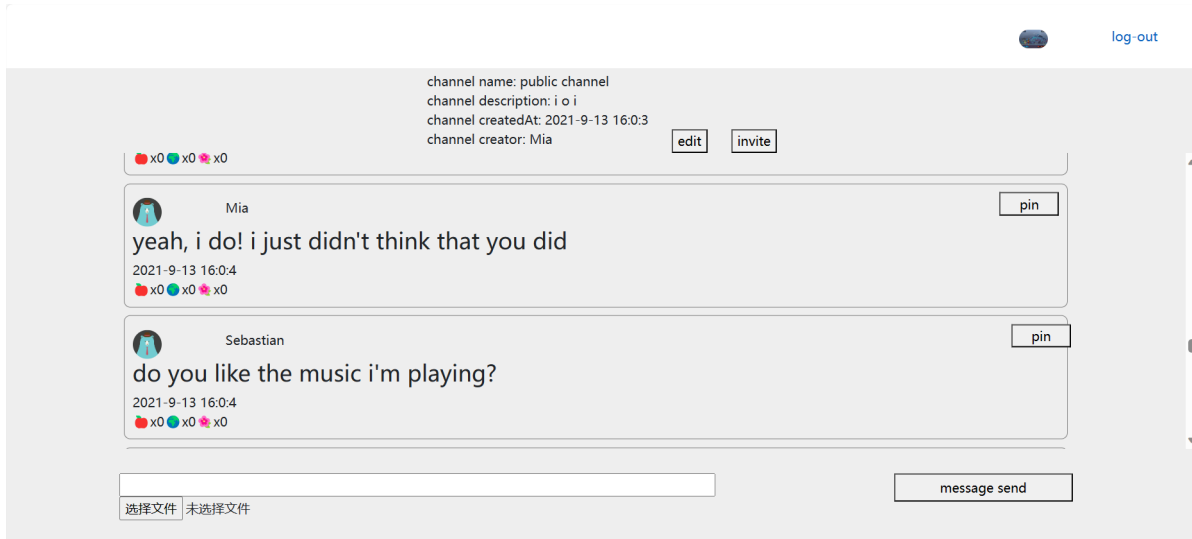
CreateChannel

© Slackr

消息

选择频道后，频道的消息都会直接显示。每条消息包括消息创建人，发送时间，编辑时间（如果有）以及消息内容。频道里的每个人可以对消息进行互动（三种方式），也可以取消互动操作，每条消息也会显示对应的互动数目。同时，每个人还可以将消息置顶。置顶的消息会标蓝且优先显示在最上面。你可以对自己的消息进行修改或者删除。消息支持图片上传。

消息列表



消息编辑



通用方法封装

将常用的API进行封装，实现request函数，之后的前后端交互均使用此函数。

```
import { BACKEND_PORT } from "./config.js";

let baseUrl = "http://127.0.0.1:" + BACKEND_PORT;

export default function request (url, params, config) {
  const initConfigs = {
    method: "GET",
    params: null,
    body: null,
    headers: {},
  };
};
```

```

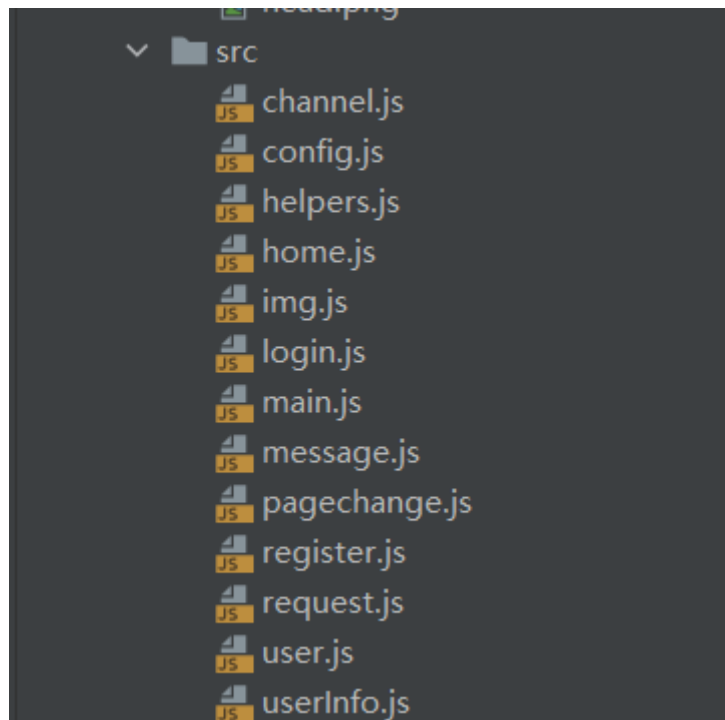
config = Object.assign(initConfigs, config);
url = baseUrl + url;
let { method } = config;
config.method = method.toUpperCase();
// with token
let token = localStorage.getItem("token");
if (token) config.headers["Authorization"] = token;
config.headers["Content-Type"] = "application/json";

let paramsRes = "";
if (/^(POST|PUT|PATCH|DELETE)$/i.test(config.method)) {
  if (params != null) params = JSON.stringify(params);
  config.body = params;
} else {
  for (const key in params) {
    const item = params[key];
    paramsRes += `&${key}=${item}`;
  }
}
if (paramsRes) url = url + `?${paramsRes}`;
return fetch(url, config)
  .then((response) => {
    let { status } = response;
    // judge response status
    if (status >= 200 && status < 400) {
      return response.json();
    }
    return Promise.reject({
      code: "STATUS ERROR",
      status,
      response: response.json(),
    });
  })
  .catch((reason) => {
    if (reason && reason.code === "STATUS ERROR") {
      reason.response.then((res) => {
        alert( res.error);
      });
    }
    return Promise.reject(reason);
  })
  .finally(() => {
  });
}

```

模块划分

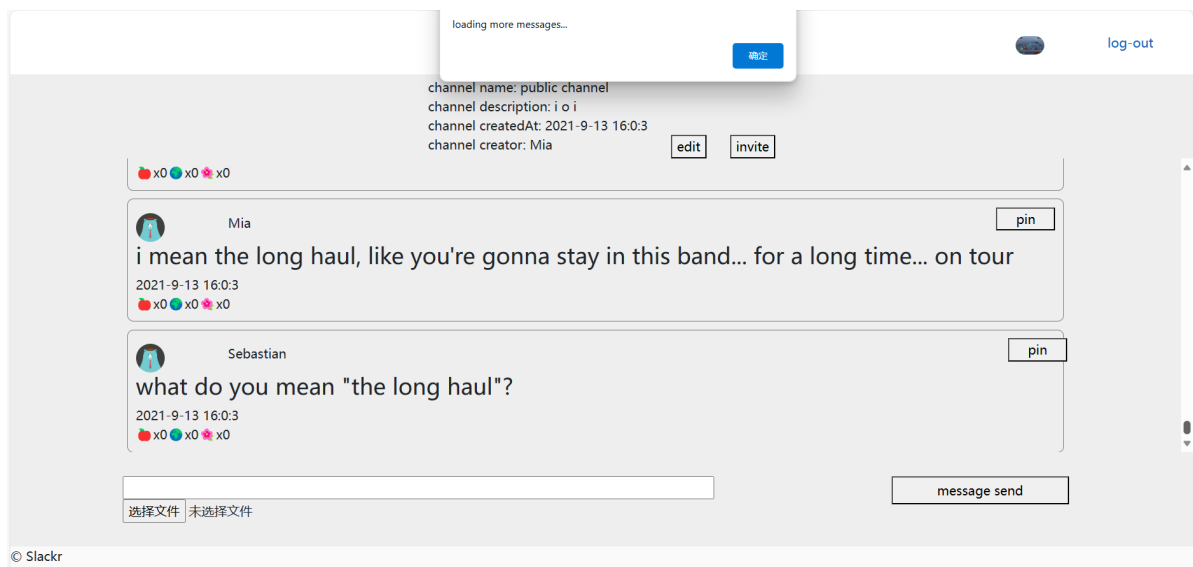
对不同的模块实现不同的JS文件，实现方法的可分离性，提高项目的内聚性以及降低耦合性。



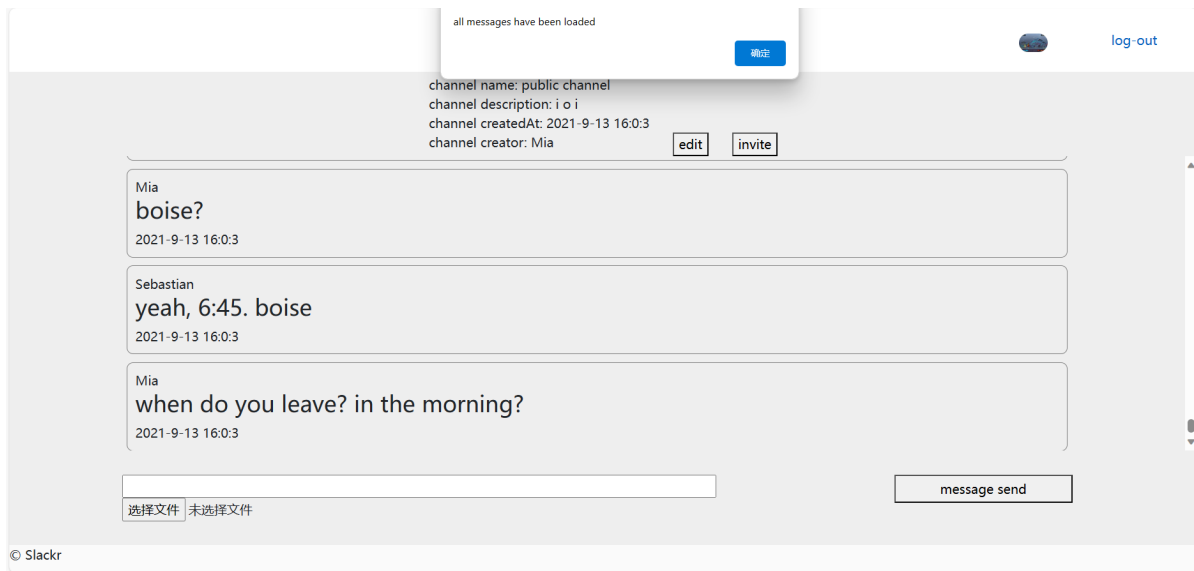
优化体验

实现了无限滚动（2.6），用户的消息不再需要分页，并且在消息加载和加载完后都会有提示。

还有消息可供加载时：



消息加载完时：



移动端适配

使用vw和vh进行css布局来实现各种屏幕的适配。