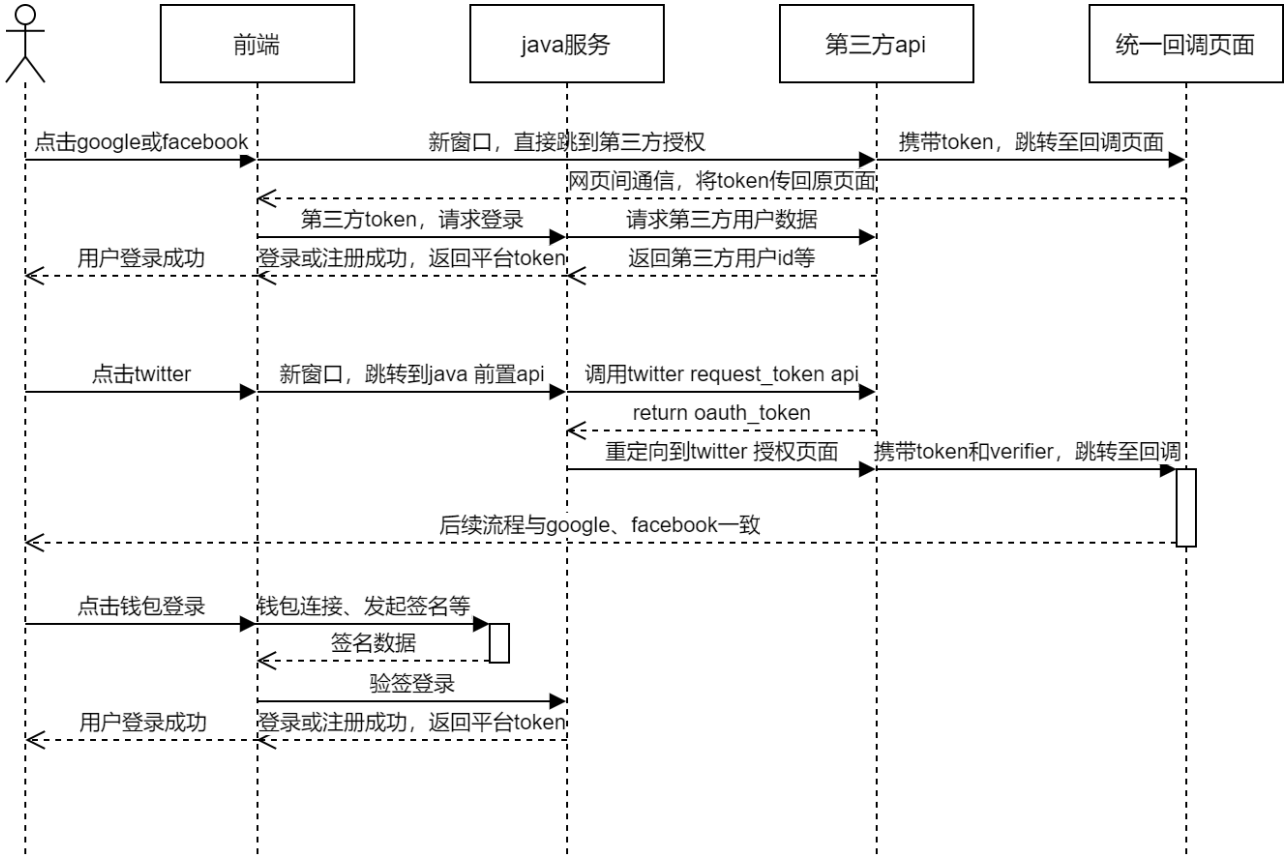
交互流程

- 通过前端请求第三方平台授权，获取到第三方平台的token或其他校验字段，发送给服务器接口
- 服务端向第三方平台api发起请求，验证相应参数，获取第三方的用户身份
- 如果在我们平台有对应映射的用户，则返回平台token；如果没有则注册用户，并返回平台token。



接口请求参数

Method: POST

Content-Type: application/x-www-form-urlencoded

Body:

```
{
    type: 'Google', // 可能的值：'Google'|'Facebook'|'Twitter'|'Wallet'
    token: 'xxxxxxxxxx...', // 对应平台的access_token，Twitter平台的oauth_token
    verifier: 'xxxxxxxxxx...', // 仅Twitter，即Twitter的oauth_verifier
    // 以下字段仅type为Wallet时传入
    signature: 'xxxxxxxxxxxxxxxxxxxx...', // 签名字符串
    message: 'Hello, Welcome ....',  // 原始文本
    address: '0x1234567845678...',  // 钱包address
    timestamp: ''
}
```

接口返回值

Content-Type: application/json

```
// 目前暂定方案
{
    accessToken: 'xxxxxxxxx',
    refreshToken: 'xxxxxxxxxxxxxx'
}
```

接口中实现第三方平台的校验

Google:

直接基于token，进行Get请求，获取用户信息（用户名和id）

https://www.googleapis.com/oauth2/v3/userinfo?alt=json&access_token=[access_token]

Facebook:

直接基于token，进行Get请求，获取用户信息（用户名和id）

https://graph.facebook.com/v11.0/me?access_token=[access_token]

更多用户信息的api：https://developers.facebook.com/docs/graph-api/reference/user#fields

Twitter:

基于token和verifier参数，调用twitter的oauth/access_token接口，即可获得twitter的user_id和screen_name

流程参考：https://developer.twitter.com/en/docs/authentication/guides/log-in-with-twitter#convert-request-to-access-token

具体api参考：https://developer.twitter.com/en/docs/authentication/api-reference/access_token

Wallet:

java参考：https://blog.csdn.net/weixin_40901926/article/details/122943926

fingerNft 参考：

```
  // fingernft-mapi\fingernft-api-farm\src\main\java\com\fingerchar\api\utils\DappCryptoUtil.java

  public static boolean validate(String signature, String message, String address) {

    if(StringUtils.isEmpty(message)) {
      return false;
```

```
    }

    String prefix = PERSONAL_MESSAGE_PREFIX + message.length();

    byte[] msgHash = Hash.sha3((prefix + message).getBytes());

    byte[] signatureBytes = Numeric.hexStringToByteArray(signature);

    byte v = signatureBytes[64];
    if (v < 27) {
      v += 27;
    }
    SignatureData sd = new SignatureData(v, Arrays.copyOfRange(signatureBytes, 0, 32), Arrays.copyOfRange(signatureBytes, 32, 64));
    String addressRecovered = null;
    boolean match = false;
    for (int i = 0; i < 4; i++) {
      BigInteger publicKey = Sign.recoverFromSignature((byte) i, new ECDSASignature(new BigInteger(1, sd.getR()), new BigInteger(1, sd.getS())), msgHash);
      if (publicKey != null) {
        addressRecovered = "0x" + Keys.getAddress(publicKey);
        if (addressRecovered.toLowerCase().equals(address.toLowerCase())) {
          match = true;
          break;
        }
      }
    }
    return match;
  }
```

其他接口需求

1、twitter 请求授权的前置接口

Method: POST

Query: redirect=[统一回调地址]

返回值:

请求正常时，返回状态码302，重定向到twitter授权链接

异常时，返回状态码302，重定向到redirect指定的回调地址

```
// 参考实现：Nodejs
    const requestData = {
      url: tokenUrl,
      method: 'POST',
      data: {
        oauth_callback: redirect
      }
    };

    try {
      const response = await fetch(requestData.url, {
        method: requestData.method,
        body: JSON.stringify(requestData.data),
        headers: oauthTwitter().toHeader(oauthTwitter().authorize(requestData)) as any
      });
      const text = await response.text();
      const { oauth_token, oauth_token_secret, oauth_callback_confirmed } = url.parse(`?${text}`, true).query;



      if (oauth_callback_confirmed !== "true") {
        throw new Error("Missing `oauth_callback_confirmed` parameter in response (is the callback URL approved for this client application?)");
      }

      res.redirect(`${authUrl}?oauth_token=${oauth_token}`);
    } catch (e) {
      res.redirect(`${redirect}?error=${(e as Error).message}`);
    }
```

参考文档：https://developer.twitter.com/en/docs/authentication/guides/log-in-with-twitter#obtain-a-request-token

接口细节文档：https://developer.twitter.com/en/docs/authentication/api-reference/request_token

2、统一回调url 及页面部署

统一回调地址，是接收从第三方平台重定向回来的参数的地址。为了便于与原发起页面通信，使参数能够传回原页面，直接使用html+js的页面接受数据，并使用网页间通信，回传至原页面。
由于是纯静态的内容，页面可以直接部署到oss，使用cdn加速，配合运维做一个域名指向。比如：
https://auth-callback.soundsright.com