从零开始的 web 开发入门

- 1 基础体验
- 2 进阶
- 3 推荐学习路线和一些学习资源
- 4 常见问题

1 基础体验

网站搭建初体验——使用已有的框架和模板

- 整个指引可参考https://lider7214.cn/2022/11/09/B2-3-1web%E5%AD%A6%E4%B9%A0%E4%B8%8E%E5%8D%9A%E5%AE%A2%E6%90%AD%E5%BB%BA%E6%80%BB%E7%BB%93/中第二部分
- 网站框架是Hexo，可通过阅读相关的博客贴和官网文档https://hexo.io/zh-cn/docs/
- 当有了个人网站之后需要发布文章，文章的用markdown格式书写的，初步的使用可参考相关互联网教程和https://docs.net9.org/basic/markdown/

2 进阶

From ustc 王子博
Hypercube@0x01.me

讲解视频：http://ftp.lug.ustc.edu.cn/weekly_party/2021.12.18_Web_Development/obs-record.mp4

网站工作流程

https://hack.lug.ustc.edu.cn/board/?group=ustc

协议：HTTPS　　域名：hack.lug.ustc.edu.cn　　路径：/board/?group=ustc

IP 地址：对域名进行 DNS 解析得到 202.38.95.102
端口：HTTPS 默认为 443

方法：GET
路径：/board/?group=ustc
请求头：

Accept: text/html;q=0.9,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Host: hack.lug.ustc.edu.cn
User-Agent: Mozilla/5.0 (X11; Linux x86
…

状态码：200 (OK)
响应头：
Content-Encoding: gzip
Content-Type: text/html; charset=utf-8
Date: Sat, 18 Dec 2021 00:00:00 GMT
Server: openresty
…
响应正文：
<!DOCTYPE html>
<html lang=”zh-hans”>
…

网站需要什么？

域名	白嫖：github.io / home.ustc.edu.cn 购买：阿里云 / Cloudflare / GoDaddy
服务器	白嫖：GitHub Pages / home.ustc.edu.cn 购买：DigitalOcean / AWS Lightsail 购买：阿里云 / 腾讯云 / AWS CDN：Cloudflare
服务器端程序	静态文件：Nginx 编程：Lua / PHP / Python / Ruby / C / Haskell 数据库：SQLite / PostgreSQL
HTML、CSS	直接写：（vanilla CSS、vanilla JS）
浏览器端程序（JavaScript）	库：Bootstrap / Pure CSS / Ant Design JS 框架：Vue.js / React

个人建站：
Hexo框架

举例

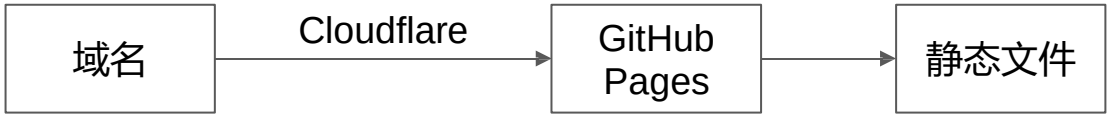
ustc.life

域名经过 Cloudflare（CDN）指向 GitHub Pages

服务器白嫖 GitHub Pages，实际文件存储在 GitHub

基本是 vanilla HTML + CSS + 一点 JS

用了 Font Awesome 的图标



举例

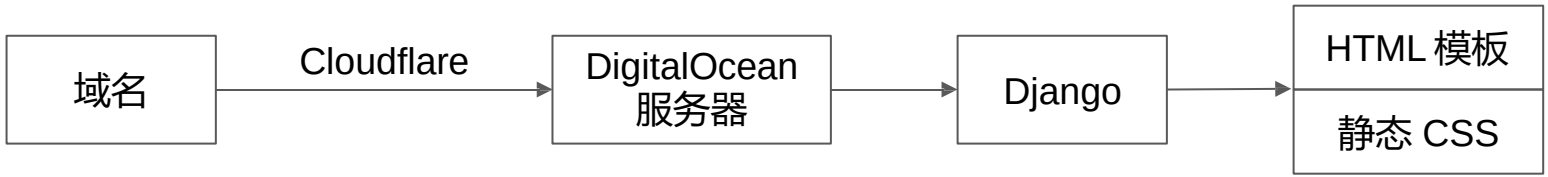
USTC QQ 号证明

域名经过 Cloudflare 指向 DigitalOcean 服务器

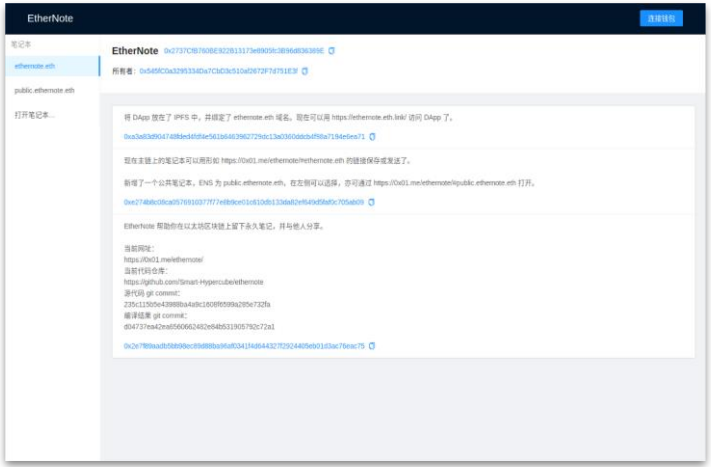
服务器直接 Django runserver

用 Django 的模板和静态文件功能

用 Bootstrap 美化



举例



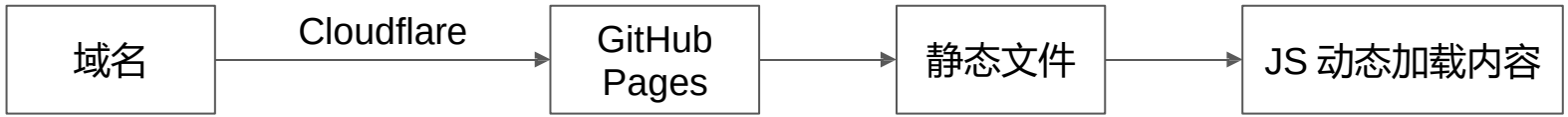
EtherNote

纯前端（数据在区块链上）

仍然白嫖 Cloudflare + GitHub Pages

前端用 React + Ant Design

用 Web3.js 读写以太坊区块链



举例

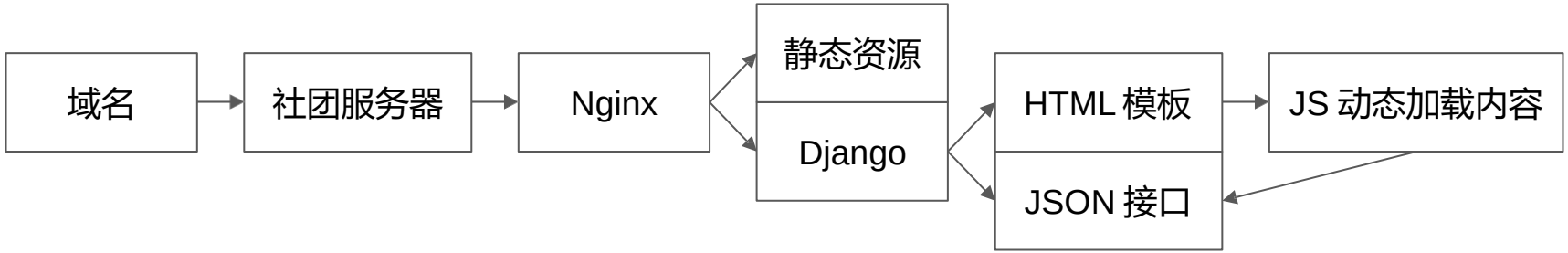
Hackergame

域名指向社团的专门服务器

“正确”部署 Django（uWSGI + Nginx + PostgreSQL）

一部分数据通过模板嵌入页面，一部分动态加载

前端用了 Pure CSS + Vue.js



举例

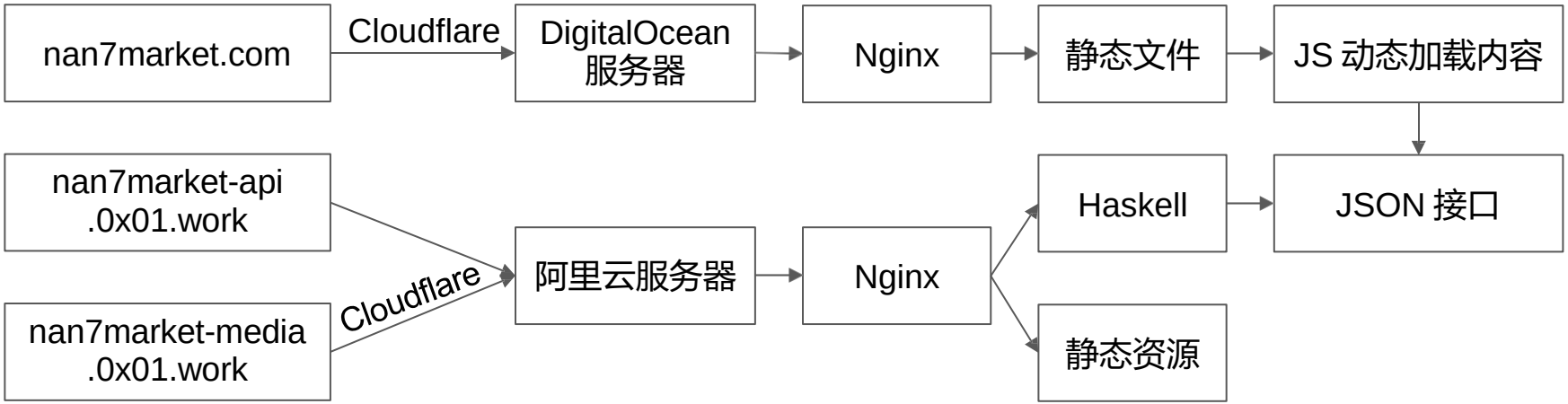
南七集市

三个域名是为了解决访问延迟、备案、流量费用等多个问题

完全前后端分离

后端是 Haskell + Yesod 写的纯 JSON API

前端是 React + Ant Design Mobile 写的单页应用



推荐学习路线

1. 自己写 HTML[1]、CSS[1] 做静态网页，用免费方案部署
2. 用 Django[2] 写网站，用 SQLite 数据库，runserver 简易部署
3. 用 Nginx[3]、uWSGI、PostgreSQL 等组件更“正确”地部署
4. 学习 JavaScript[1][4] 和 ES6[5]，写浏览器端程序
5. 学习 Vue.js、React、Ant Design 等“现代”前端框架，用 <script> 标签简易部署
6. 学习 npm / Yarn、webpack 等“现代”前端开发工具

一些学习资源

[1]: 菜鸟教程 https://www.runoob.com/ 和 https://developer.mozilla.org/zh-CN/docs/Learn/Getting_started_with_the_web

[2]: Django 官方文档教程 https://docs.djangoproject.com/en/4.0/intro/

[3]: Nginx 配置工具 https://nginxconfig.io/

[4]: JavaScript 教程 https://wangdoc.com/javascript/

[5]: ES6 教程 https://wangdoc.com/es6/

常见问题：备案

区分三个不同概念：ICP 备案、公安备案、经营性 ICP 许可证

所有从中国大陆接入互联网的机器，如果在某个域名上提供网站服务，就必须在网络接入商处完成 ICP 备案。

不用备案的情况（满足任何一条即可）：

- 不从中国大陆接入互联网（GoDaddy）
- 不提供网站服务
- 不在域名上提供网站服务

一个备案号可以对应多个网络接入商的“接入备案”

- 第一次接入备案时会新产生备案号
- 后续接入备案还用之前的备案号
- 最后一个接入备案被注销时，备案号被注销

常见问题：安全

现代网站和浏览器涉及很多复杂的安全问题，但了解基础知识 + 使用主流工具可以帮助避免绝大多数问题。

请求方法和参数

- GET、POST、…
- query、application/x-www-form-urlencoded、multipart/form-data、application/json

CORS（Cross-Origin Resource Sharing，跨域资源共享）

XSS（Cross Site Scripting，跨站脚本）

CSRF（Cross-Site Request Forgery，跨站请求伪造）

Clickjacking（点击劫持）

常见问题：认证与鉴权

认证（authentication）是指验证用户的身份

鉴权（authorization）是指判断用户能否执行特定操作

一般来说，认证成功后，用户的后续请求都要携带认证结果，这需要一种会话（session）实现

认证方法：密码、验证码、第三方登录、签名、客户端证书、……

会话实现：

- 浏览器端存储 session ID / 签名 / 加密并签名
- 利用 cookie / bearer token
- 滚动刷新机制
- revoke 机制

鉴权模型：权限、组、属主、超级管理员、规则集、角色、……

常见问题：前后端分离

一般地说，“前后端分离”指浏览器加载一个静态网站，所有动态数据都用 JS 向后端服务器获取。

好处：

- 访问体验可以更好，不会每操作一下都导致页面刷新
- 易于分别编写两部分的程序
- 避免拼接 HTML 产生安全或性能问题 问

题：

- 没什么好的接口设计范式！

接口设计时常见的问题：

- 多行查询（查询前 100 名的用户个人信息）
- 多列查询（查询所有文章的标题、作者、摘要）
- 联合查询（查询某个商品的所有评论，以及每条评论的发布者信息）
- 分页

后端内部的“前后端”

Hackergame 项目尝试采用的设计

后端：

- 每个模块负责一部分业务逻辑
- 负责鉴权
- 每个模块有自己私有的存储，外部不得直接访问
- 每个模块暴露一套接口，保证内部鉴权逻辑和数据一致性

前端：

- 负责和 HTTP、HTML 等概念打交道
- 负责认证
- 调用后端接口时要提供用户，无法绕过后端的鉴权逻辑
- 理论上可以换成任何能进行认证、接受用户的技术，比如 Telegram bot