智能合约

前端搜索界面（html）

交互（python：web3.py 或者js：web3.js）

和云服务器的接口？

前提：IPFS（将文件映射为唯一字符串）

基本算法：

1 密钥生成（这里采用对称加密算法 密钥可以通过区块链传播（私钥认证））

2 索引算法：索引直接保存在云服务器上

3 陷门生成算法：由查询方生成

智能合约函数：

1 用户提交押金

2 服务器请求检索

3 服务器返回检索结果

4 用户请求结果：包括智能合约对服务器结果的验证

最后用户解密

Python：

1基本算法PEKS：

* 1. 密钥生成
  2. 加密
  3. 陷门生成算法：由查询方生成
  4. 解密

2与html交互（django框架？）

2.1 注册时生成（两个密钥对）+（用户名，指向密钥对的下标）

2.2 输入用户名，请求公钥进行关键字加密

2.3 点击查询时，请求私钥进行trapdoor的生成

2.4 传回文件给用户（直接解密）

3和智能合约交互web3.py

3.1 注册生成的东西传给智能合约

3.2 请求公钥进行关键字加密

3.3 请求私钥进行trapdoor的生成

3.4 请求rsa私钥进行文件解密

pypbc安装：

<https://blog.csdn.net/KoalaZB/article/details/89499048>

<https://blog.csdn.net/bubbleliar/article/details/101548630>

Pypbc代码：

<https://blog.csdn.net/weixin_34318272/article/details/91811741?utm_medium=distribute.pc_relevant.none-task-blog-OPENSEARCH-1&depth_1-utm_source=distribute.pc_relevant.none-task-blog-OPENSEARCH-1>

pypbc的使用：

<https://blog.csdn.net/qq_14834491/article/details/100657640>

<https://github.com/debatem1/pypbc/blob/master/test.py>

ubuntu安装python-django的配置环境：

<https://blog.csdn.net/qq_42092076/article/details/100796208>

Pycharm+django：

<https://blog.csdn.net/luhu124541/article/details/81606347?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-1&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromBaidu-1>

python django 之 前端：

<https://blog.csdn.net/ABoLuoTaMei/article/details/100690206>

Django 中使用 ajax 请求：

<https://tendcode.com/article/django-ajax/>

Django教程（包含项目）：

<https://www.liujiangblog.com/course/django/85>

<https://www.liujiangblog.com/course/django/109>

Django对用户提交的表单数据进行处理（菜鸟教程）：

<https://www.runoob.com/django/django-form.html>

django官方文档:

<https://docs.djangoproject.com/zh-hans/3.0/>

表单：

<https://docs.djangoproject.com/zh-hans/3.0/intro/tutorial04/>

将views.py中的数据传递到前端html页面，并展示：

<https://blog.csdn.net/songlh1234/article/details/81185036?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-24&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-24>

Web3.py：

0.官方文档：<https://web3py.readthedocs.io/en/latest/>

1.<https://www.mscto.com/blockchain/258474.html>

2.<https://blog.csdn.net/mongo_node/article/details/80954816?utm_medium=distribute.pc_relevant.none-task-blog-baidujs-7>

3.<https://blog.csdn.net/qq_41907714/article/details/103852873?utm_medium=distribute.pc_relevant.none-task-blog-OPENSEARCH-1&depth_1-utm_source=distribute.pc_relevant.none-task-blog-OPENSEARCH-1>

python类库web3.py概述:

<https://cnodejs.org/topic/5b4ea54d4b9cd4200f942099>

4.Python Web3.py和Infura Windows中设置托管以太坊节点：

<http://blog.hubwiz.com/2018/10/31/ethereum-python-node-windows/>

1. python开发以太坊应用（使用web3）：

<https://blog.csdn.net/fei13971414170/article/details/82891641?utm_medium=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-2.nonecase&depth_1-utm_source=distribute.pc_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-2.nonecase>

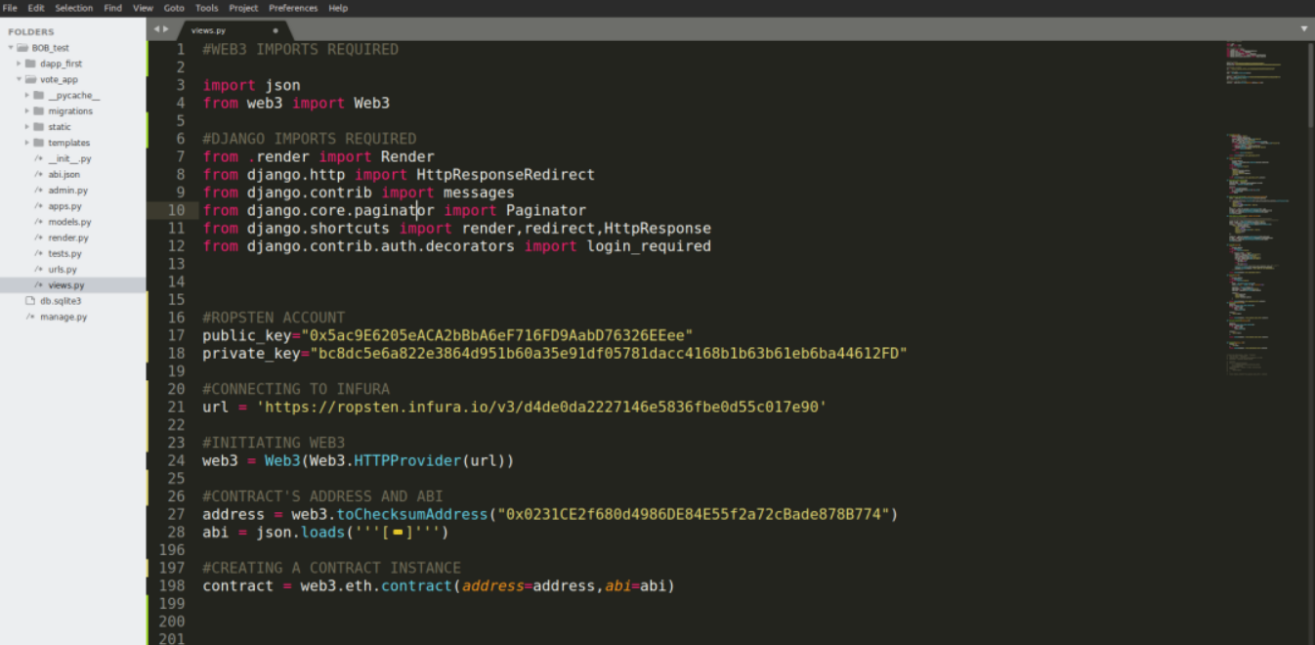
6.python如何用以太坊web3客户端（不知道在讲啥）：

<http://blog.hubwiz.com/2018/08/10/ethereum-python-client-dev/>

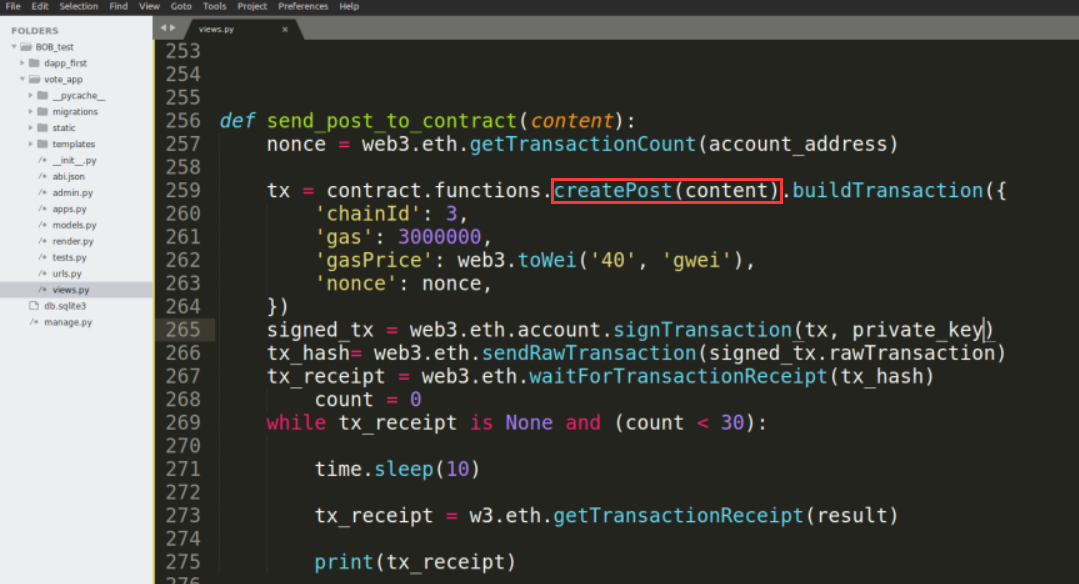
7.用 Django 开发基于以太坊智能合约的 DApp：

<http://vlambda.com/wz_7iKZWtJc5jO.html>

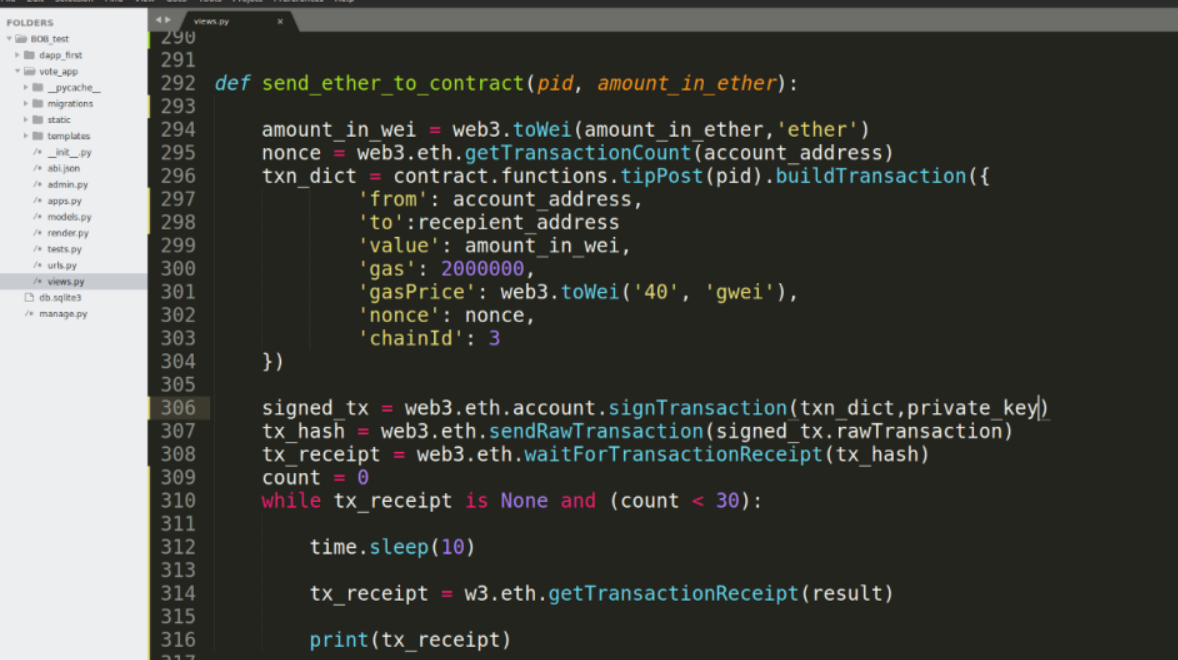
8.<https://ctolib.com/topics-137119.html>



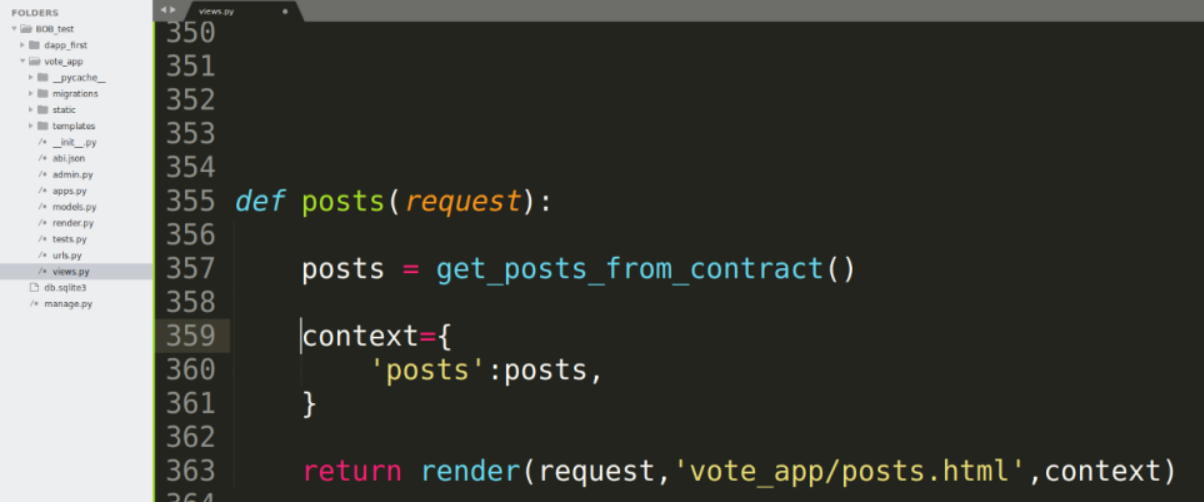
调用事务：



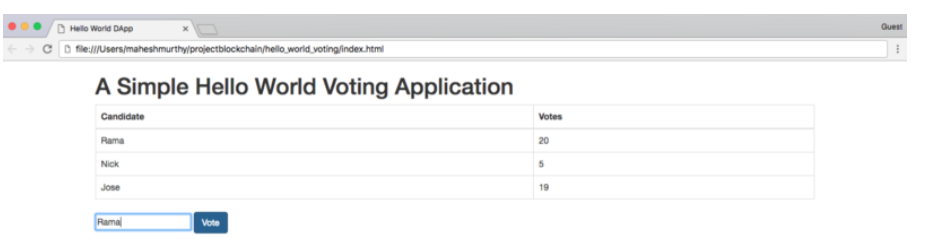
付款：



将posts\_data传递到模板的函数：



[如何在以太坊上搭建一个Dapp？](https://www.cnblogs.com/bigben0123/p/9176259.html)



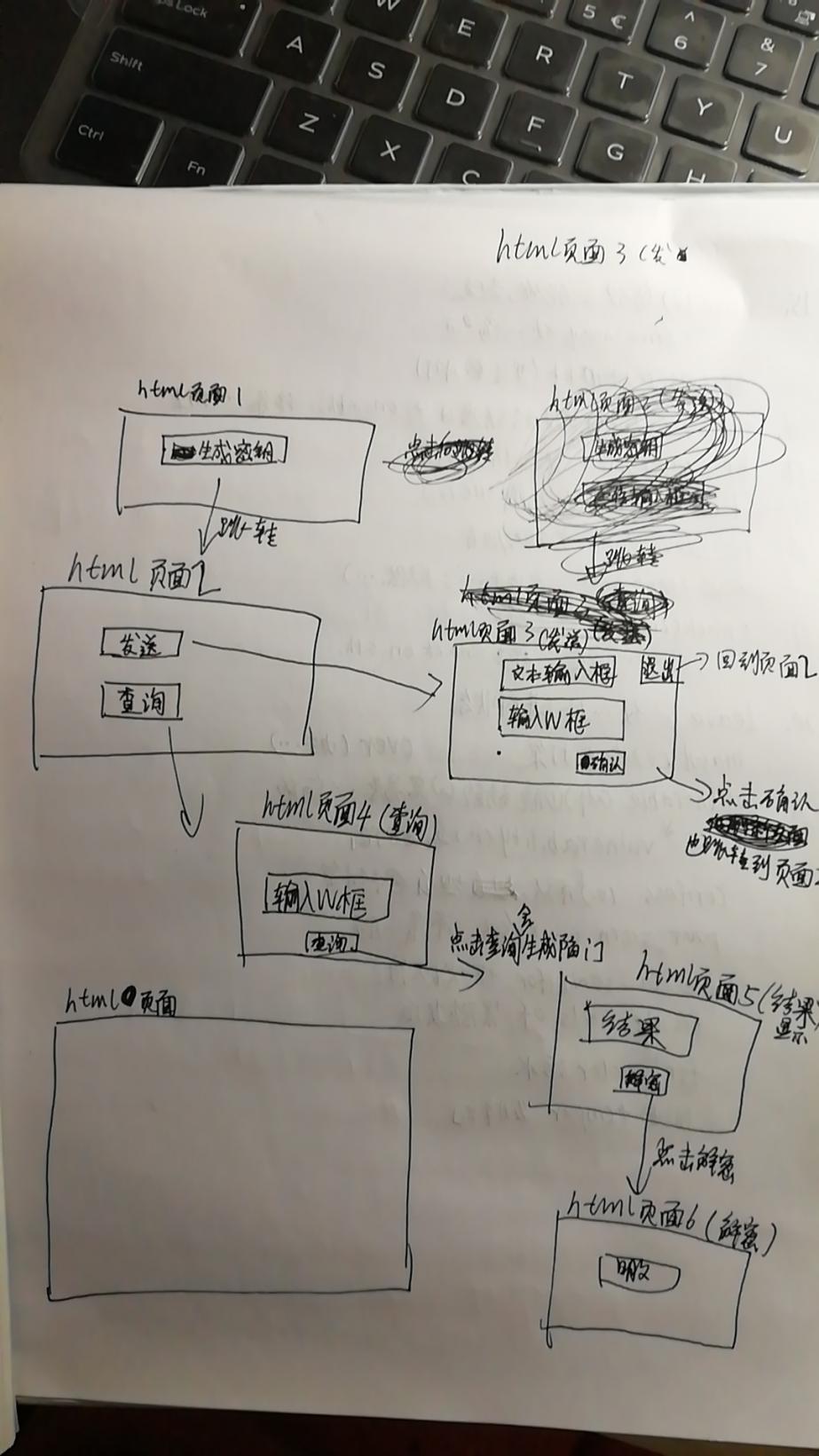
<https://www.cnblogs.com/bigben0123/p/9176259.html>

搭建以太坊私有链并两台电脑间节点连接：

<https://blog.csdn.net/wahaha13168/article/details/81028494>

Html需修改部分：  
1) 注册页面新增一个确认密码

1. 各个页面因为新加massage，所以板式要调整一下
2. 新增退出登录按键
3. Send页面的类似search的按钮不明显
4. Search页面接收到密文之后，提供按钮解密



Js与python结合：

<https://blog.csdn.net/a312863063/article/details/87898349>

Python CGI编程：

<https://www.runoob.com/python3/python3-cgi-programming.html>

Python\_前端网页+前后端交互（django框架）：

<https://blog.csdn.net/Grandaunt/article/details/84632791>

Django框架：

<https://www.jb51.net/article/120522.htm>

Windows下使用python-web3.py进行以太坊Dapp开发笔记--第1篇：

<https://blog.csdn.net/qq_41907714/article/details/103852873?depth_1-utm_source=distribute.pc_relevant.none-task-blog-OPENSEARCH-1&utm_source=distribute.pc_relevant.none-task-blog-OPENSEARCH-1>

Python以太坊智能合约开发指南：

<http://blog.hubwiz.com/2018/12/14/ethereum-python-smartcontract/>

以太坊智能合约：

<https://github.com/Fabsqrt/Junshan/tree/master/Blockchain>