commodity_backingtrack_system

基于区块链的商品溯源系统

运行方式

1. 环境配置

1.1 Python获取

需从python官网下载python环境，下载地址：

Python Releases for Windows | Python.org

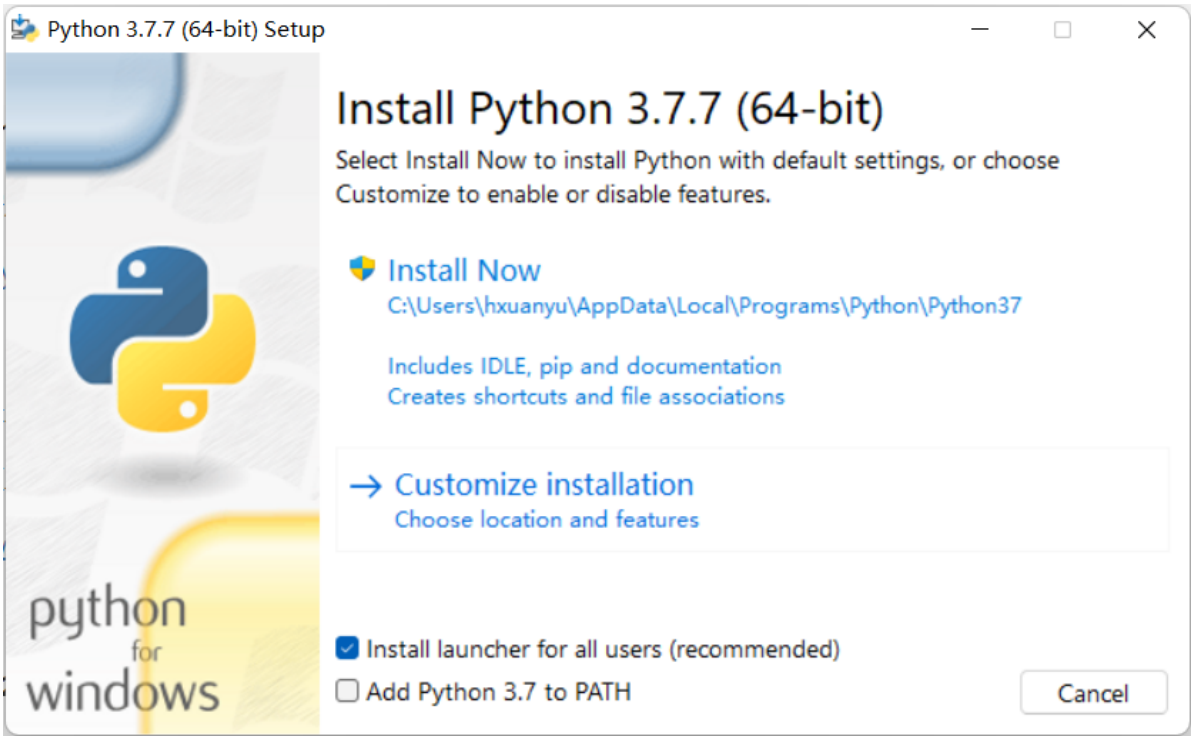
进入页面后找到3.7.7版本下载即可

这里给出windows环境的二进制安装包下载链接：

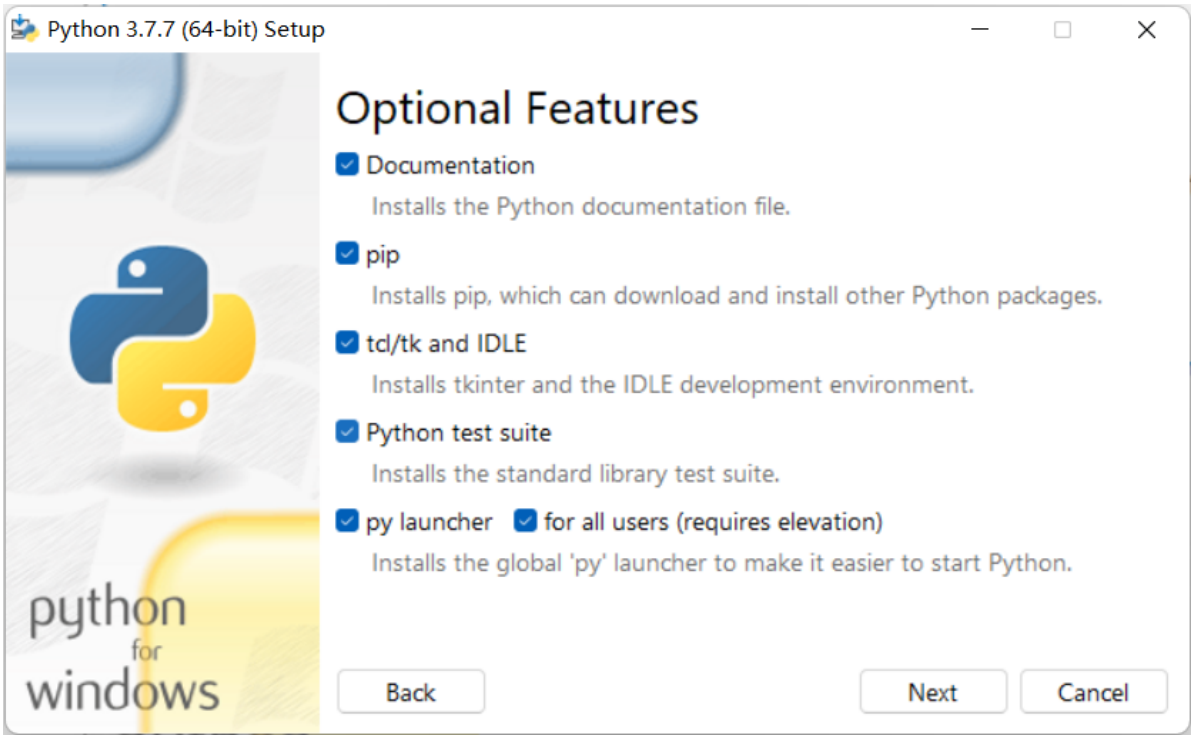
python-3.7.7-amd64.exe

1.2 Python安装

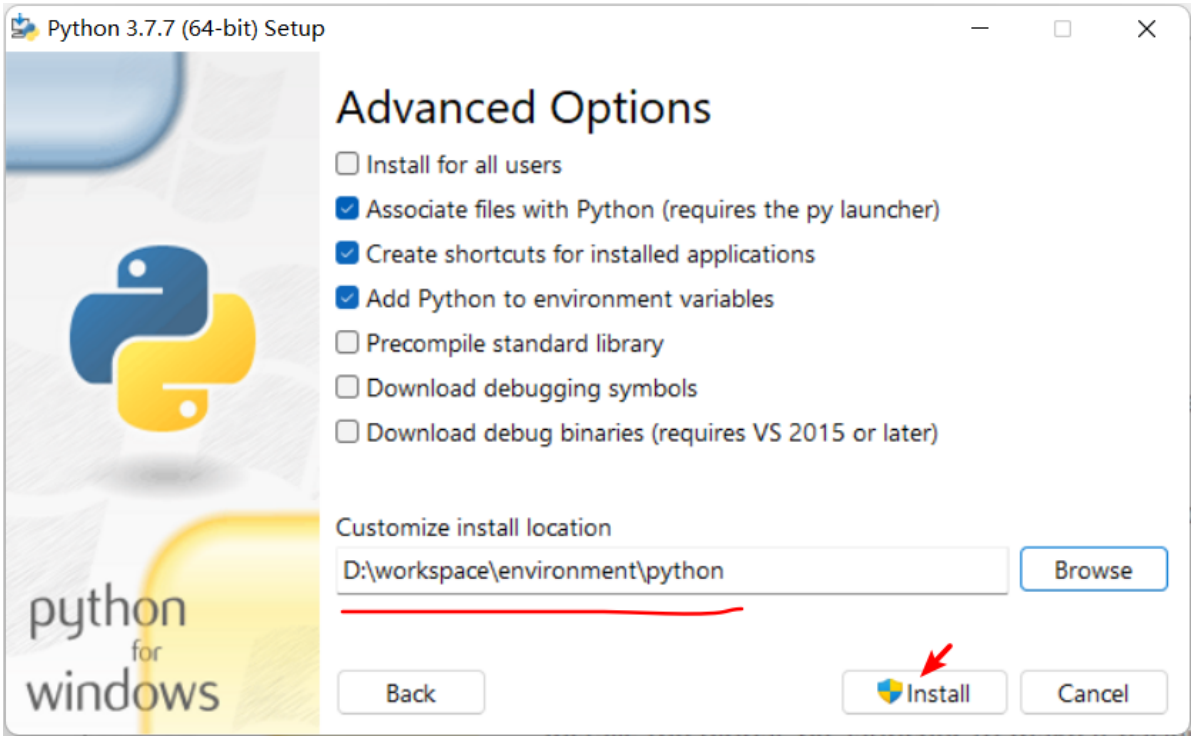
运行下载好的安装包，



> 选择下面的`Customize installation`进行自定义安装，可以勾选下方的`Add Python 3.7 to PATH`选项，将python可执行文件添加至系统变量



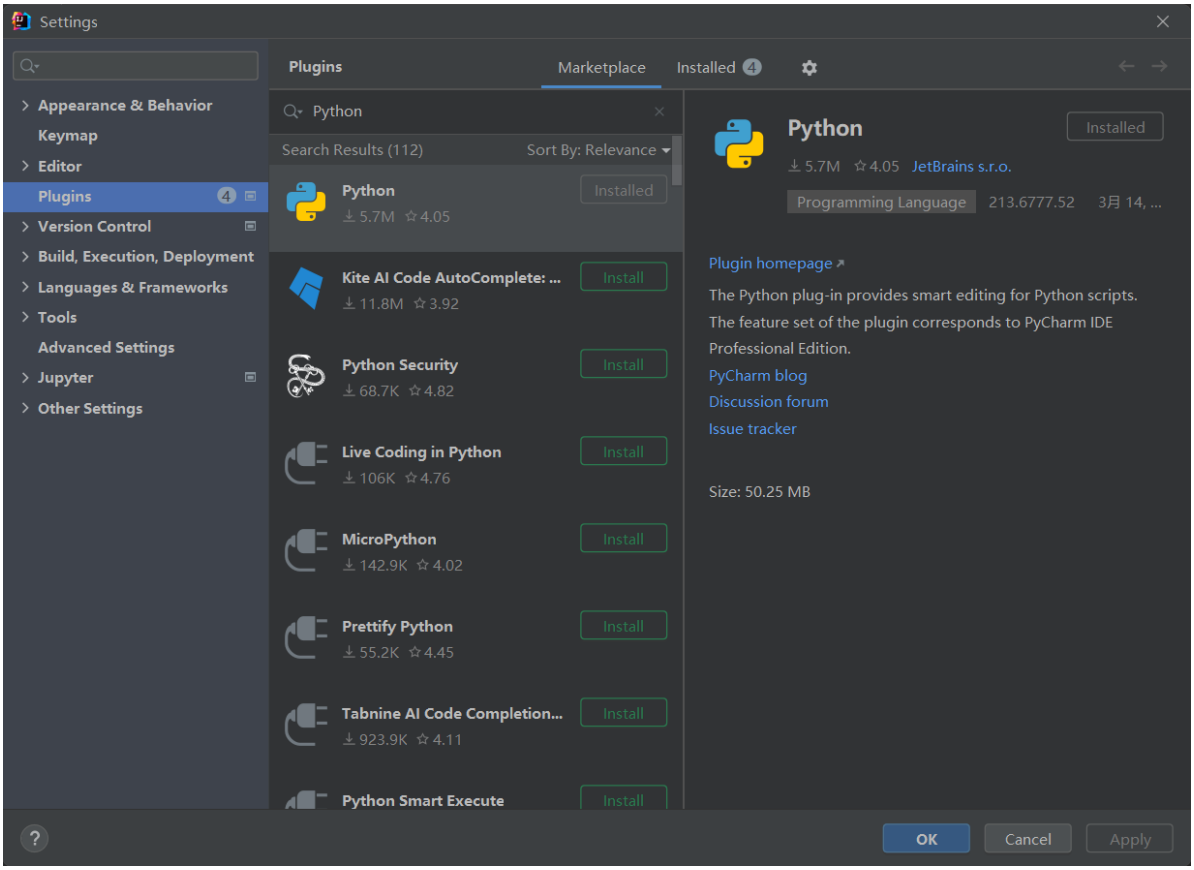
本页选择默认即可



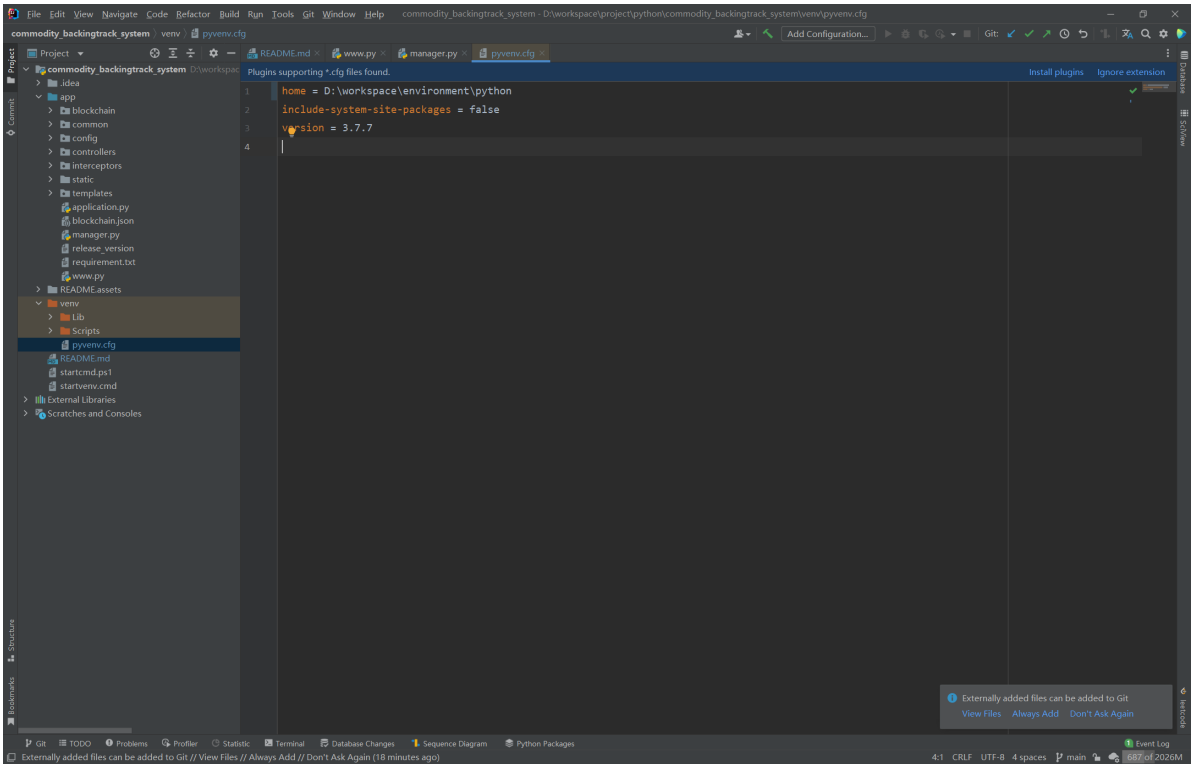
本页面选项可默认，下面的安装地址如果需要可以进行修改，但是一定要记住这个地址，后续配置需要用到。设置完成后点击Install按钮进行安装。

2. 在IDEA中运行

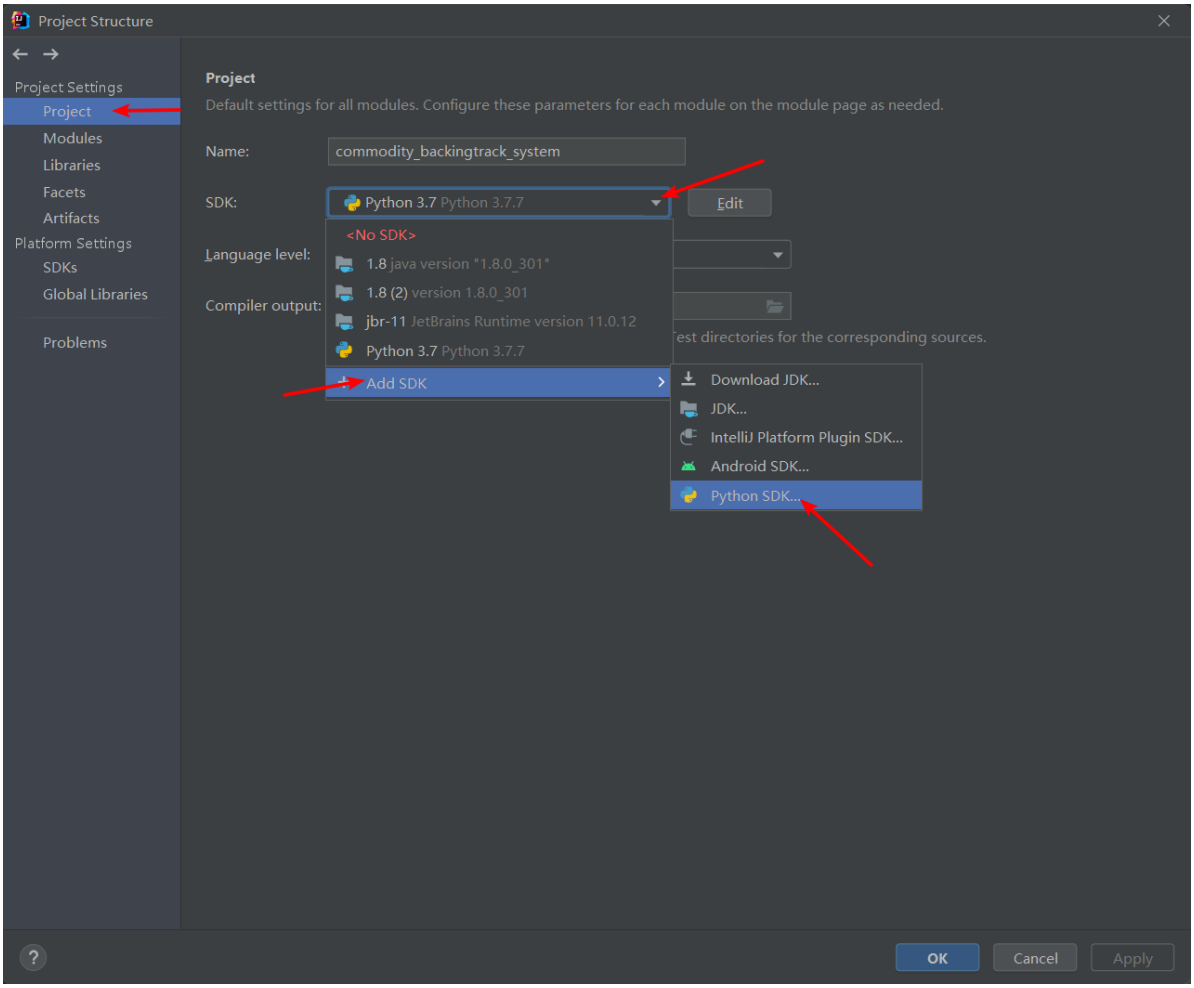
使用IDEA打开项目，点击File>Setting>Plugins打开插件安装界面，搜索Python插件并安装



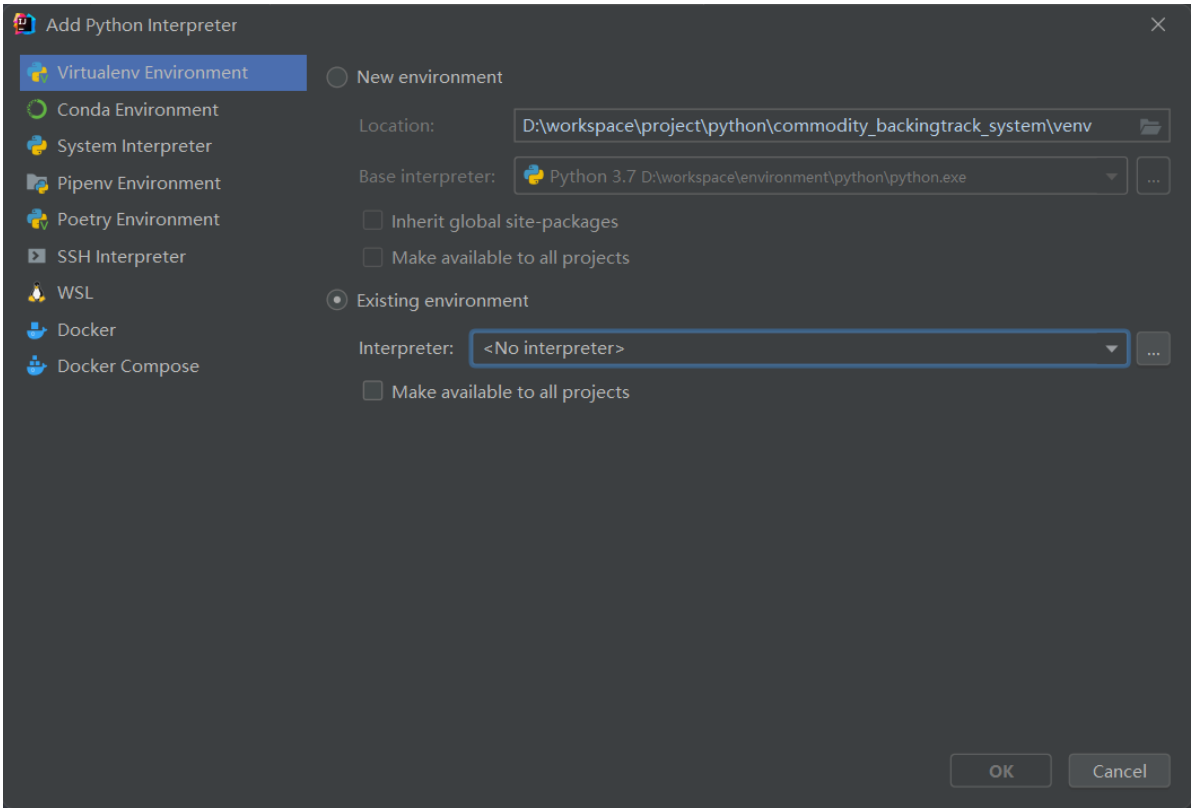
打开`commodity_backingtrack_system\venv`目录下的`pyvenv.cfg`文件，修改其第一行内容为上面python安装的地址



随后依次点击 File > Project Structure > Project ，打开项目设置界面，并下拉SDK一栏，找到 Add SDK，并选择 PythonSDK



在打开的界面中，保持左侧选项卡为Virtualenv Environment不变，右侧选择Existing environment



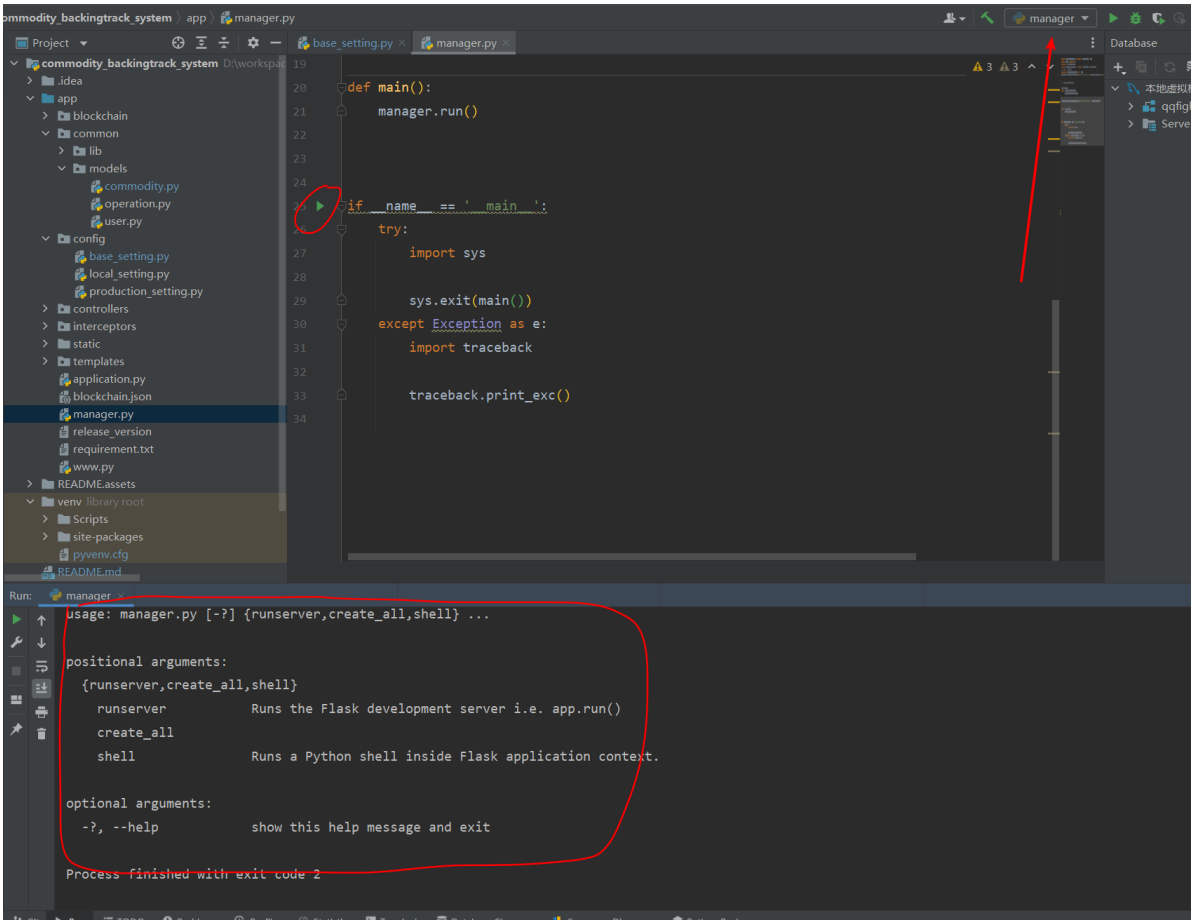
点击下拉框右侧的三个点，选择当前项目下的venv文件夹中的python.exe文件，之后全程点击OK即可，等待IDEA完成加载。

3. 项目启动前的配置

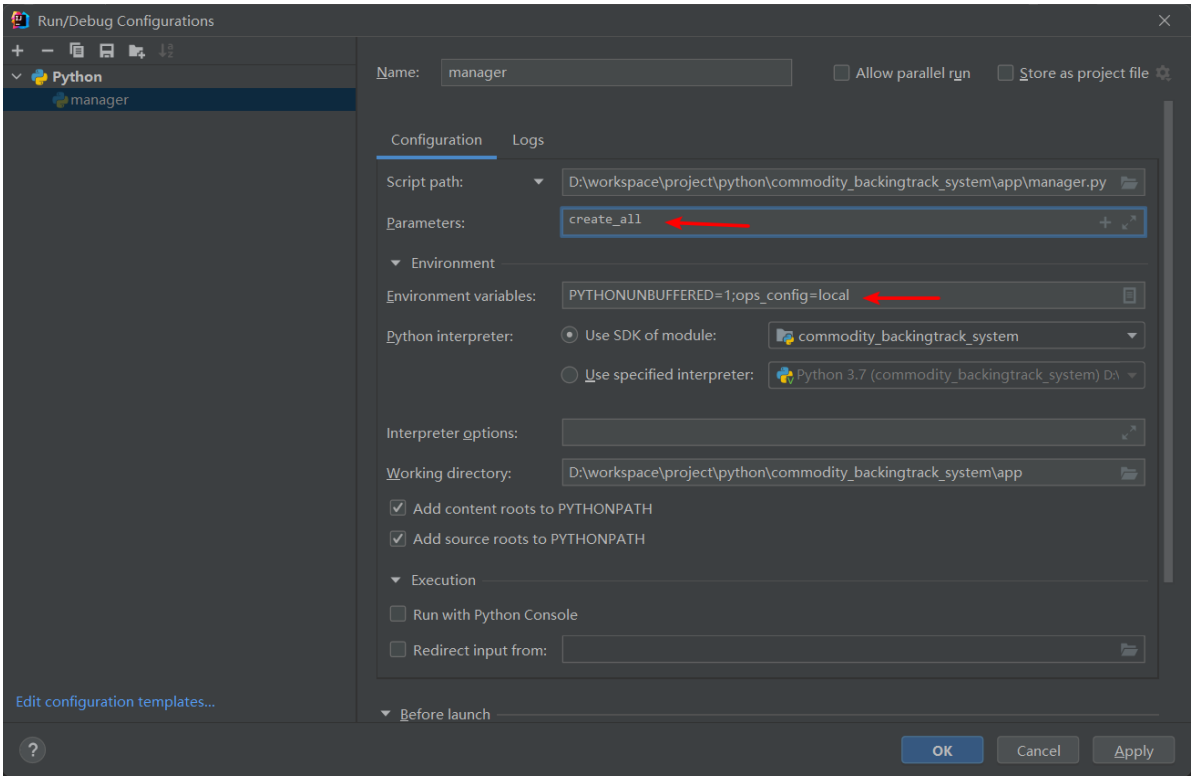
- 数据库配置：修改项目的`app/config/local_setting.py`文件中的`SQLALCHEMY_DATABASE_URI`字段，可参照原有数据格式进行修改，即：

```
SQLALCHEMY_DATABASE_URI = "mysql://用户名:密码@数据库地址/数据库名"  # 数据库连接配置
```

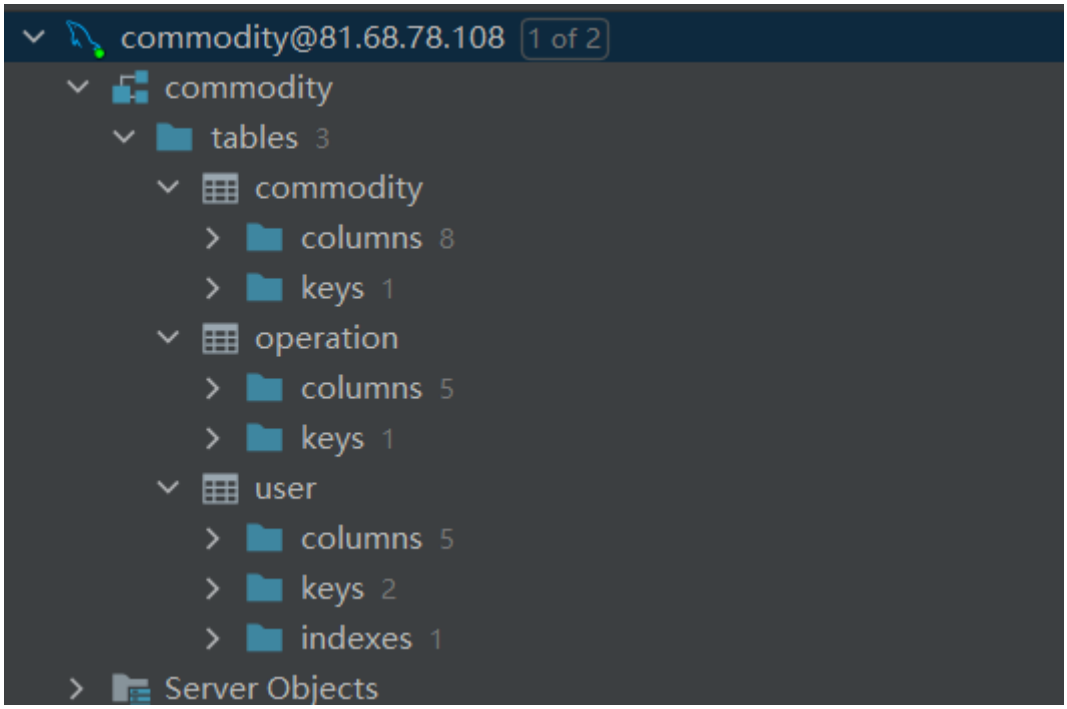
- 监听端口配置：继续修改`local_setting.py`文件中的DOMAN字段，修改`www`字段为运行程序的机器所在的局域网IP，如果本地运行则修改为`http://127.0.0.1:5000`即可。
- 数据库建表：打开`app/manager.py`，找到IDEA左侧显示的运行按钮，点击运行，此时会发现运行失败，但是右上角出现了运行配置，此时点击下拉框，选择Edit Configuration选项，编辑运行配置



编辑Parameters框，填入参数“create_all”， 并在下面的`Environment variables`中追加：`;ops_config=local`，表示添加环境变量键值对，应用启动时会根据这个配置加载`xxx_setting.py`配置文件。



再次运行项目，如果数据库连接正常的话，会成功创建三个数据表：



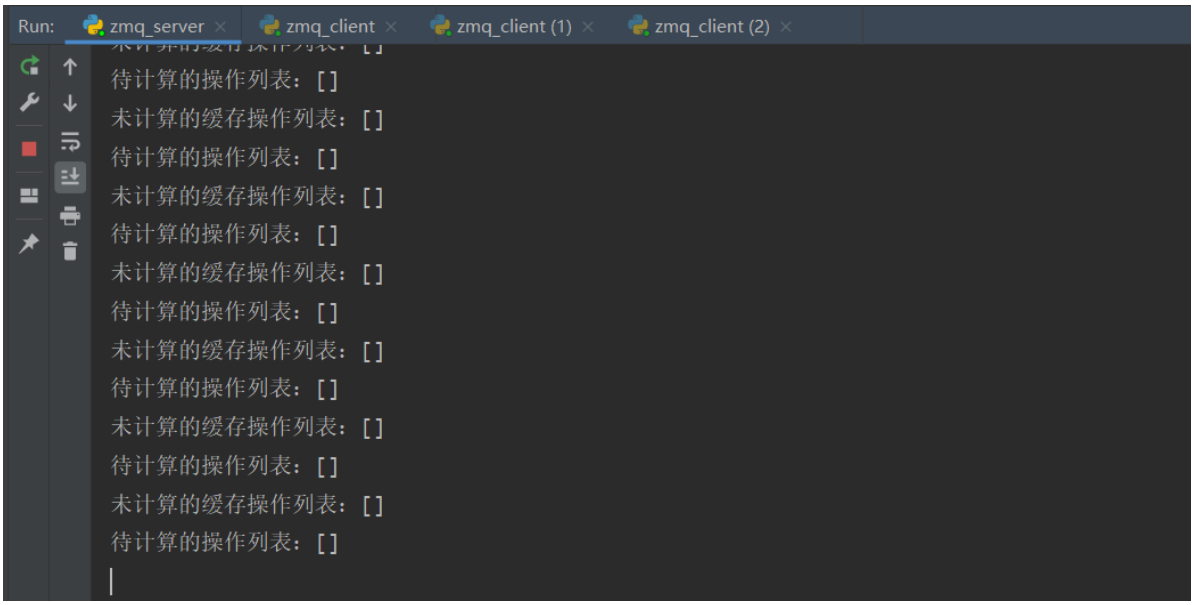
如果建表失败，也可以使用如下建表语句手动创建数据表：

```
create table commodity
(
    id        int auto_increment
        primary key,
    name      varchar(32) default '' not null,
    origin    varchar(32) default '' not null,
    seller    varchar(32) default '' not null,
    trans     varchar(32) default '' not null,
    warehouse varchar(32) default '' not null,
    buyer     varchar(32) default '' not null,
    status    varchar(32) default '' not null
);

create table operation
(
    id           int auto_increment
        primary key,
    user_id      int          not null,
    type         varchar(255) not null,
    commodity_id int          not null,
    date         datetime     not null
);

create table user
(
    id          int auto_increment
        primary key,
    name        varchar(30) not null,
    type        varchar(10) not null,
    password    varchar(32) not null,
    regist_time datetime    not null,
    constraint name
        unique (name)
);
```

4. 正式运行项目

4.1 区块链服务端及客户端启动

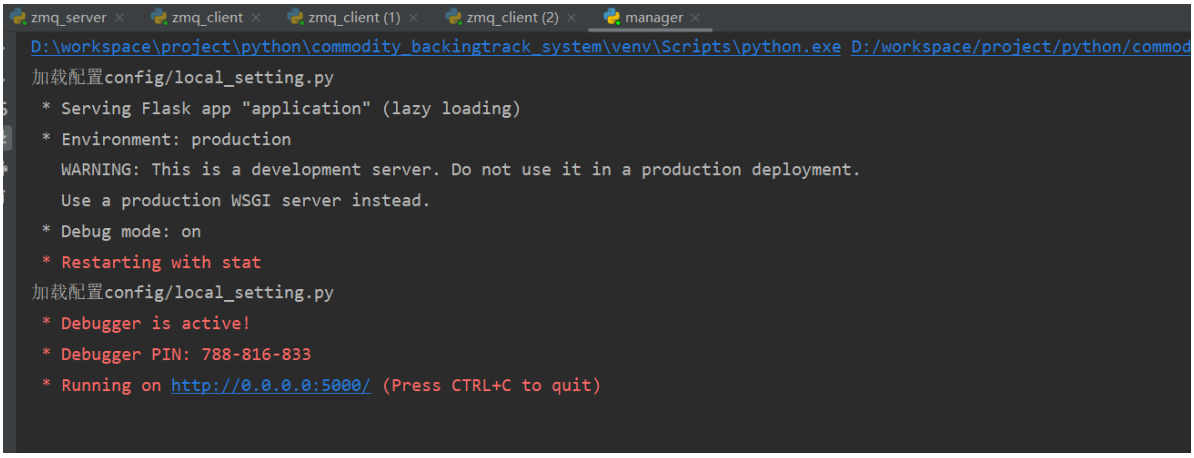
打开`/app/blockchain`目录，其中`server`为服务端，`clients`文件夹为客户端集群，`client`文件夹为客户端源码，运行时不需要此文件夹中的内容，其他客户端均由此文件夹复制生成，内部为三个完全一致地客户端节点，打开`zmq_server.py`或`zmq_client.py`文件后，下方运行入口左边会显示运行小三角，点击即可直接运行。



此时下方已经运行了1个服务节点和3个客户端节点。

4.2 启动管理系统

修改之前的manager运行配置，将`create_all`参数改为`runserver`启动参数后，点击运行即可



此时应用已经启动，可在浏览器中访问`http://127.0.0.1:5000`打开系统，开始前需要进行账号注册操作，共有4种账户类型，分别代表商品流转过程中经历的不同节点，商品的状态以及可以操作的权限可参考代码。商品的状态发生改变时，会被区块链服务端接收到，并由客户端进行工作量计算，得到结果后服务端会选择占大多数的证明为最终结果写入区块链，区块链保存在json中，每个节点均保存一个备份。

点击溯源按钮后，服务端会从区块链上查询商品的操作记录并进行展示。