

ASSP: Automated strategy-based stock processors

System Design Specification (SDS)

<Version 1.2>

<Date 2021.11.23>

王泽华 2130233077

林睿翀 2130233046

曾健洪 2130233029

万祥来 2130233074

Artificial Intelligence and Digital Media Programme

United International College



1. Document Change Log

<i>Change Date</i>	<i>Changed By</i>	<i>Version</i>	<i>Change Description</i>
17/11/2021	王泽华	1.0	Prepared Document
18/11/2021	王泽华, 林睿翀	1.1	Construct the data structure and logic flow
19/11/2021	万祥来	1.1	Point out the strategy to show the processing code
19/11/2021	曾健洪	1.1	Draw the control flow
19/11/2021	王泽华, 林睿翀	1.2	Update the code of getting data and show how to connect to Android



2. Table of Contents

1.	DOCUMENT CHANGE LOG	2
2.	TABLE OF CONTENTS	3
3.	DESIGN OVERVIEW	4
4.	TOOLS AND STANDARDS	5
4.1.	Tools	5
4.2.	Standards	5
5.	USER INTERFACE DESIGN	6
5.1.	Usage Scenario 1	6
5.2.	Usage Scenario 2	19
6.	DATABASE DESIGN	20
7.	DIAGRAMS	21



3. Design Overview

<Please provide a general overview of the project in terms of the product or service that is being built, so that anyone reading this section is able to gain an understanding of what functionality is provided and how this functionality will be technically implemented. Mention any partners that may be involved in the effort.>

该项目的设计，是想通过计算机的优势，帮助人们更科学、更理性地炒股。在这个时代，资金的流动越来越频繁，货币每天都在贬值。因此，再像从前那样把钱攥在手里是不现实的事情，人们越来越愿意通过风险投资来获得额外的收益。炒股便是众多风险投资中比较常见的一种，一旦掌握一定的规律，它可以很轻松地帮助你获取金钱。炒股，顾名思义就是对股票进行炒作，股票可以理解为某家公司的股份也就是对这个公司的投资，简单理解股票就是一个可以进行买卖的东西，并且在市场上股票的价格都会有一定的波动，而我们要做的事情也很简单，便是在低价位的时候进行买入，在高价位的时候卖出，从而通过动动手指便可以轻松赚钱。随着互联网的发展，针对这一方面衍生出了许多量化交易平台，在那些平台上，我们可以通过编写代码构造策略，然后进行回归分析，然后坐收小额回报。但是对于实际应用到自己的现实生活中仍旧不太理想，因为平台会施加各种各样的限制。而我们做这个项目就是为了打破这个枷锁，让每个人自己可以具现化自己的想法，并真正的运用到实际的股票交易当中。在我们设计的程序中，人不需要实时的盯着股票进行买入和卖出，而是可以使用预定的策略进行完全自动化操作，即便你白天出去上课或是上班，在自己家里的电脑上，机器也可以实时地监控股价，并帮助各位进行股票的买入卖出。

具体的设计中，主要涉及到两个问题：

1. 股票的选择

选择一支优秀的股票是十分重要的，这与策略的难易程度没有直接关系，即便我们使用的是最简单的策略，但只要这个策略能帮助我们选择出足够优秀的股票，也可以获得比那些很复杂策略更高的收益。

首先，我们得收集各支股票的数据，获取股票数据的渠道很多，可以爬虫，也可以直接到某些地方下载，我们设计的程序是直接调用接口（tushare）进行数据获取的。

其次，是对各支股票的历史数据进行分析，这个步骤我们没有使用特别复杂的策略，一是不存在所谓的最好的策略，在不同的时间段，不同的经济环境存在不同的炒股策略；二是同学都可能有更好的想法，那么大家也都可以通过自定义策略的方式来炒股。我们之所以对数据的分析没有以封装好的形式呈现出来，是因为这些模块虽然简便，但是可扩展性比较差，当有同学想使用更复杂的方法时，会非常不方便。我们这里实现了一个比较简单的策略作为展示，并向大家介绍如何编写自己的股票筛选策略。

2. 股票的择时买入/价格买入

这里的买入存在两点，第一点是股票的买入价格，因为股票的价格是当天会发生变动的，因此当天并不一定能够买入你想要买到的那个价位的股票，因此需要程序来实时进行监控，以免漏掉买入的时机。和其他程序不太一样的地方是：当我们选择买入时，人并不在电脑屏幕前进行操作，而是按照已经制定好的策略由程序自动操作，人在此时可以在任何地方做任何事情，无需担心自己股票的买入和卖出状况。



4. Tools and Standards

4.1. Tools

<Specify which technologies will be used to develop the application, e.g. Python packages such as Numpy, Pandas, etc.>

Python 包:

numpy

pandas

tushare

apscheduler

uiautomator2

easyquotation

pymysql(可选)

sqlalchemy(可选)

电脑端(通用):

adb

安卓模拟器/手机

网络要求:

wifi

4.2. Standards

<Specify what standards will be followed in this project, e.g. runs on a Windows platform, etc.>

理论上在 windows Linux Mac 平台均可使用



5. User Interface Design

5.1. Usage Scenario 1

<Put screen images here to describe each usage scenario step by step. Name each screen Figure-n and use this name in your description. Under each screen, write a paragraph to explain the objects and the actions on that screen. The purpose is to make the reader understand what is on the screen, and what will happen when any action is taken on the screen.>

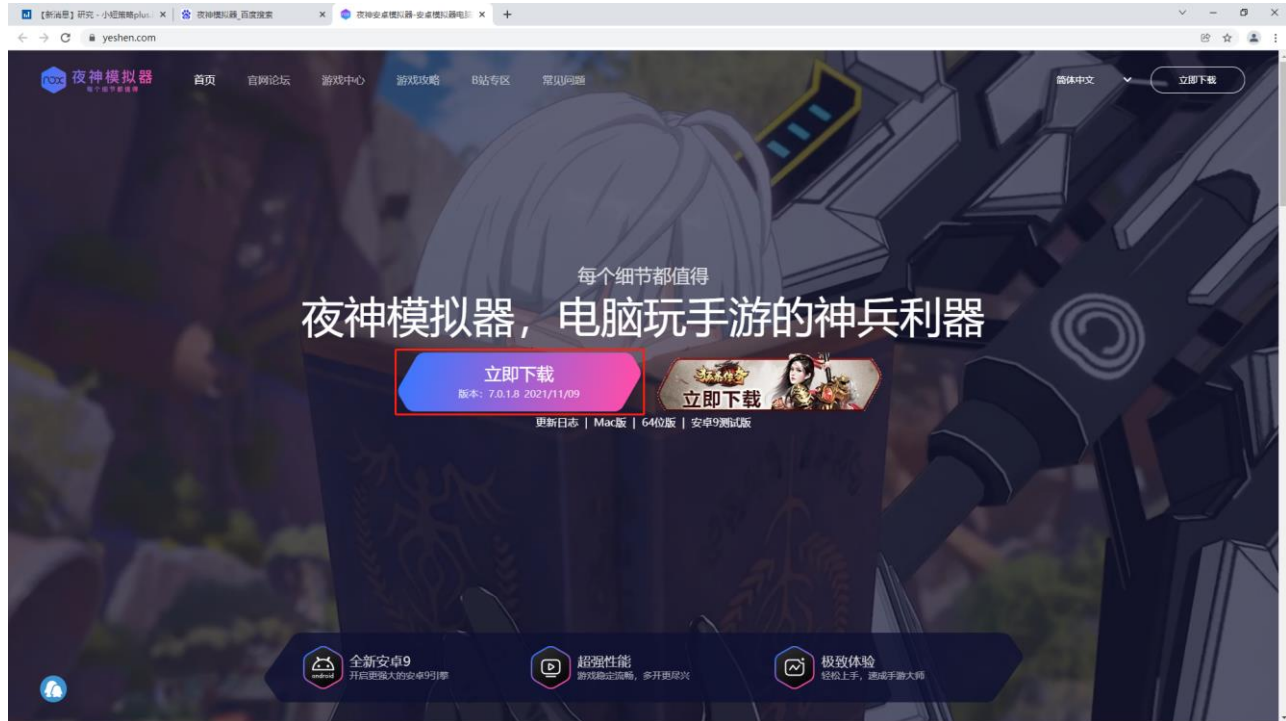


Figure 1

这里介绍在电脑端使用安卓模拟器的方法:

第一步 Figure1: 下载一个安卓模拟器, 下载后自行选择安装位置。





Figure 2

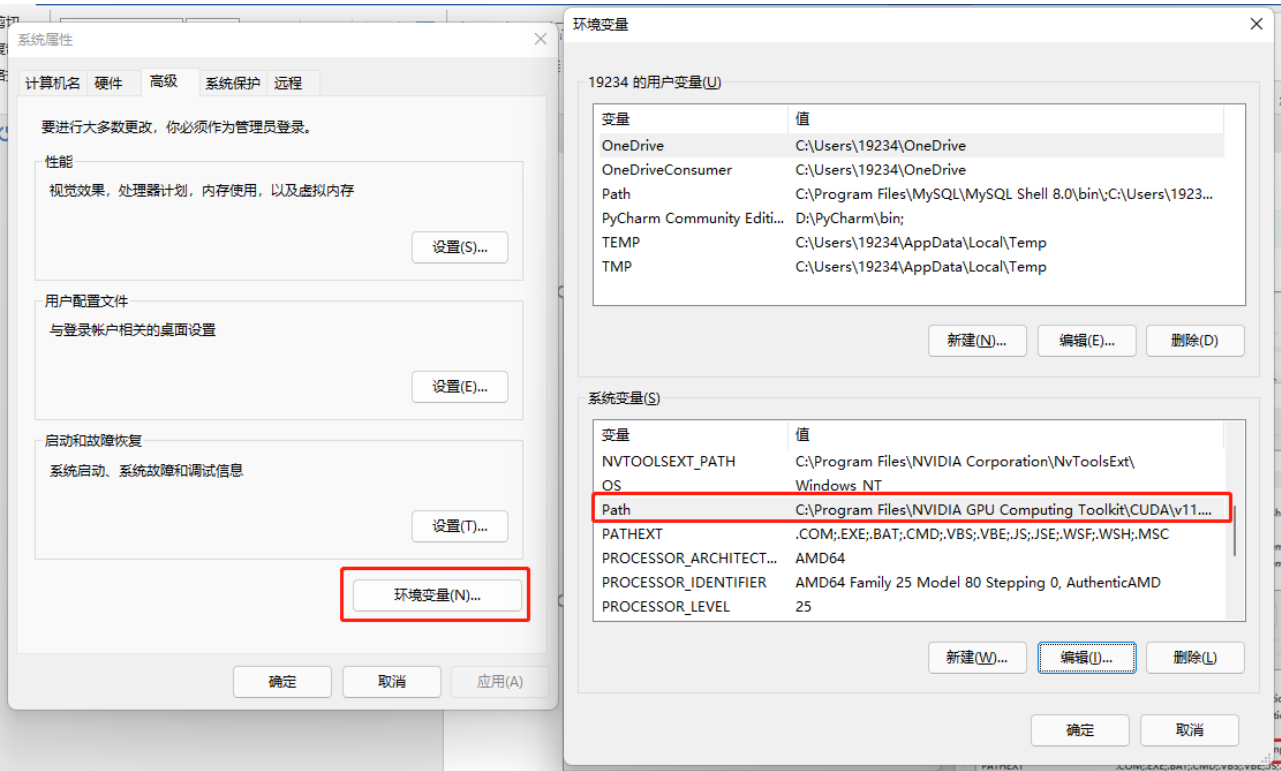


Figure 3

第二步: Figure2-3 找到环境变量-系统变量中的 Path



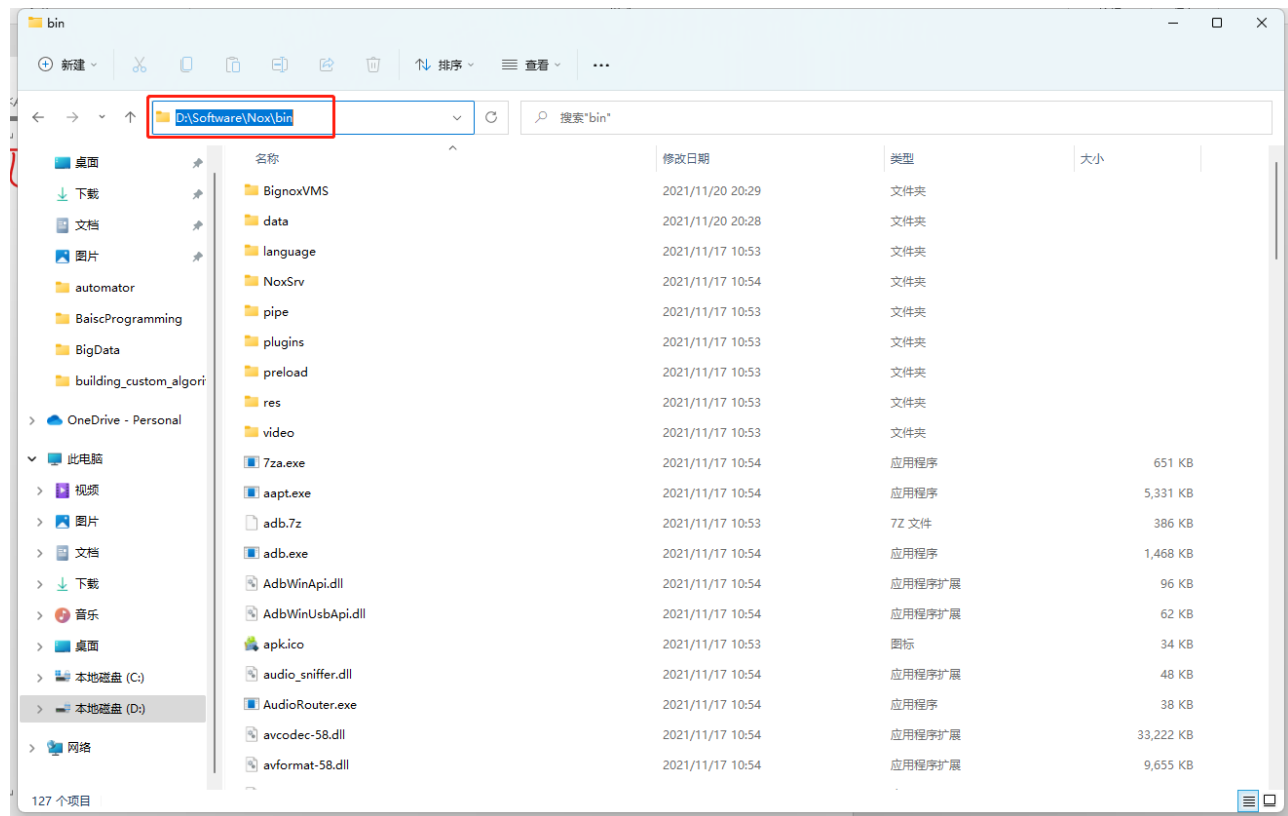


Figure 4



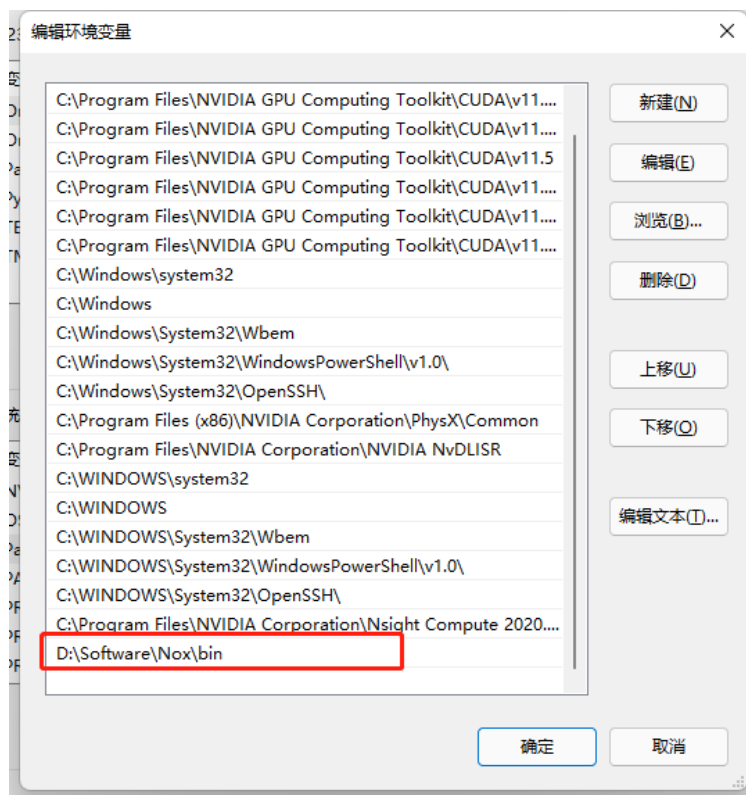


Figure 5

Figure4-5 将路径复制到环境变量中，点击确定。

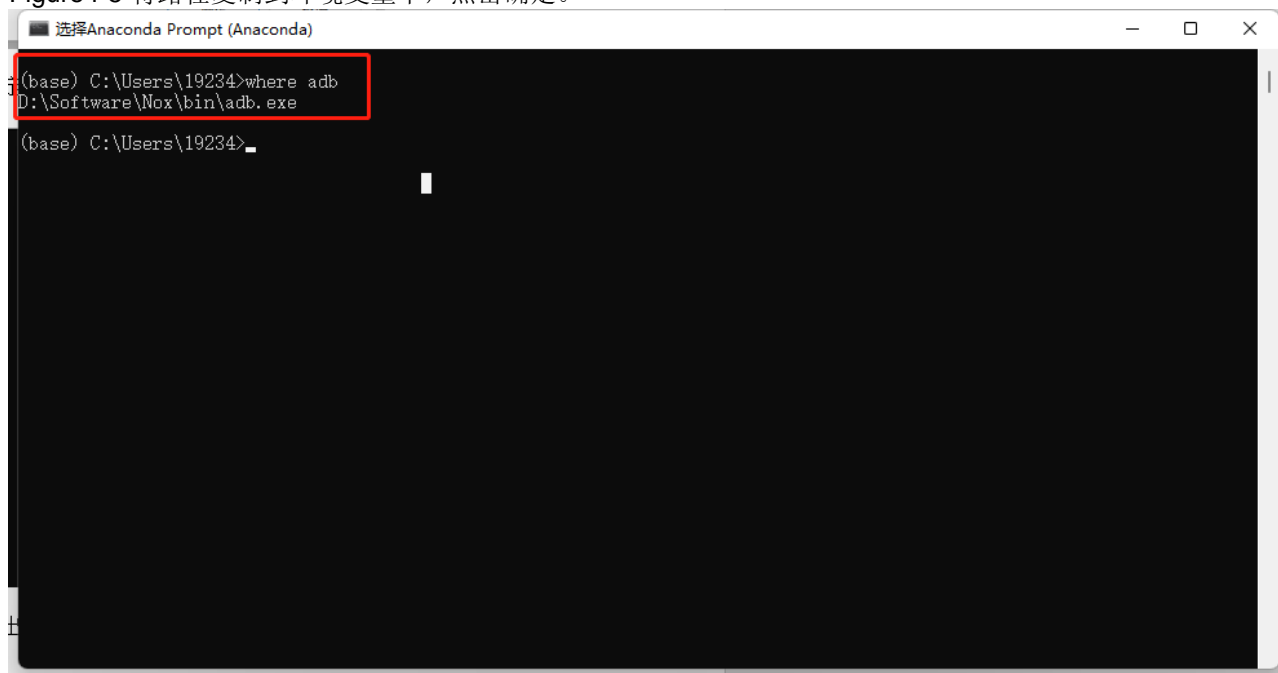


Figure 6

此时在 cmd 命令行中输入 where adb 应显示出刚才添加的路径(Figure 6)





Figure 7

打开安卓模拟器，搜索并下载同花顺(一款软件)，如 Figure7 所示

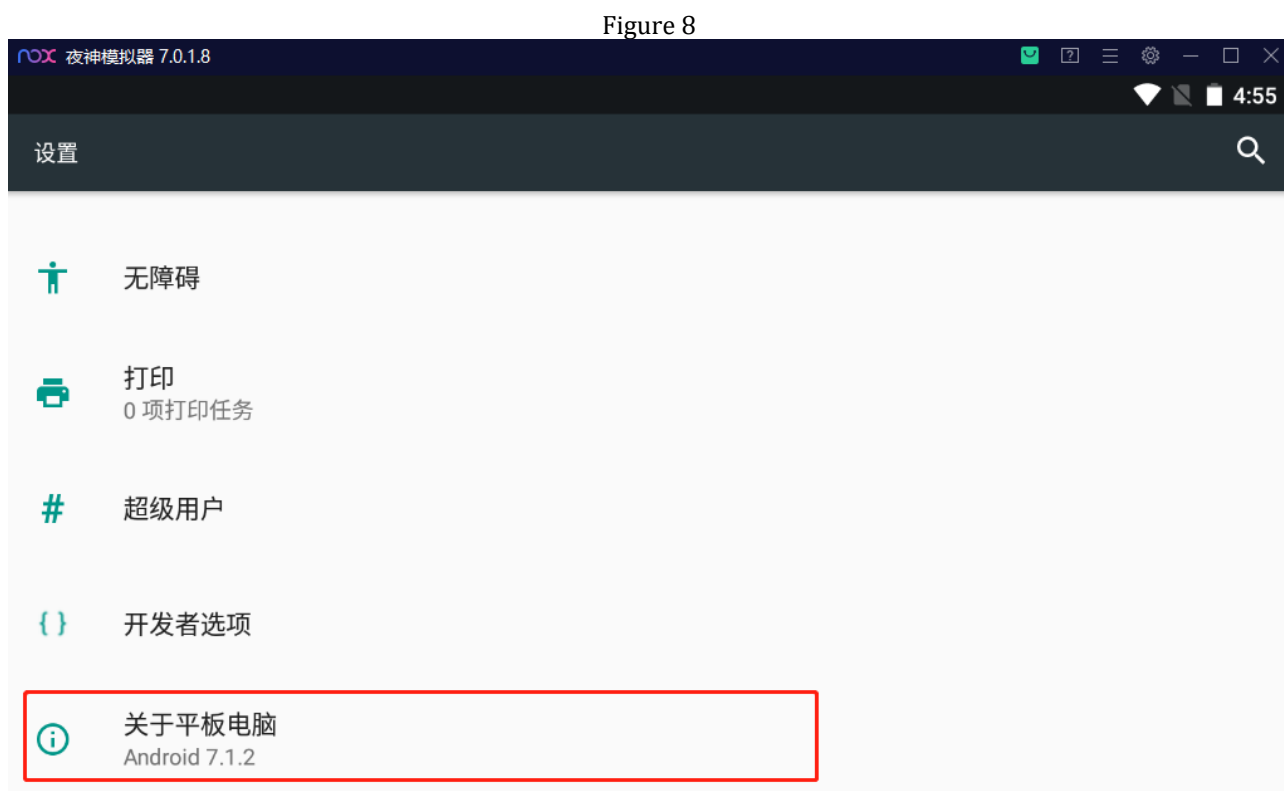


Figure 9



Figure9-12 在安卓模拟器应用内打开设置，点开关于平板电脑，拉到下面找到版本号，连续点击直至提示出现开发者选项，而后返回设置界面即可看到开发者选项，点击进入，并打开 **Usb 调试**。

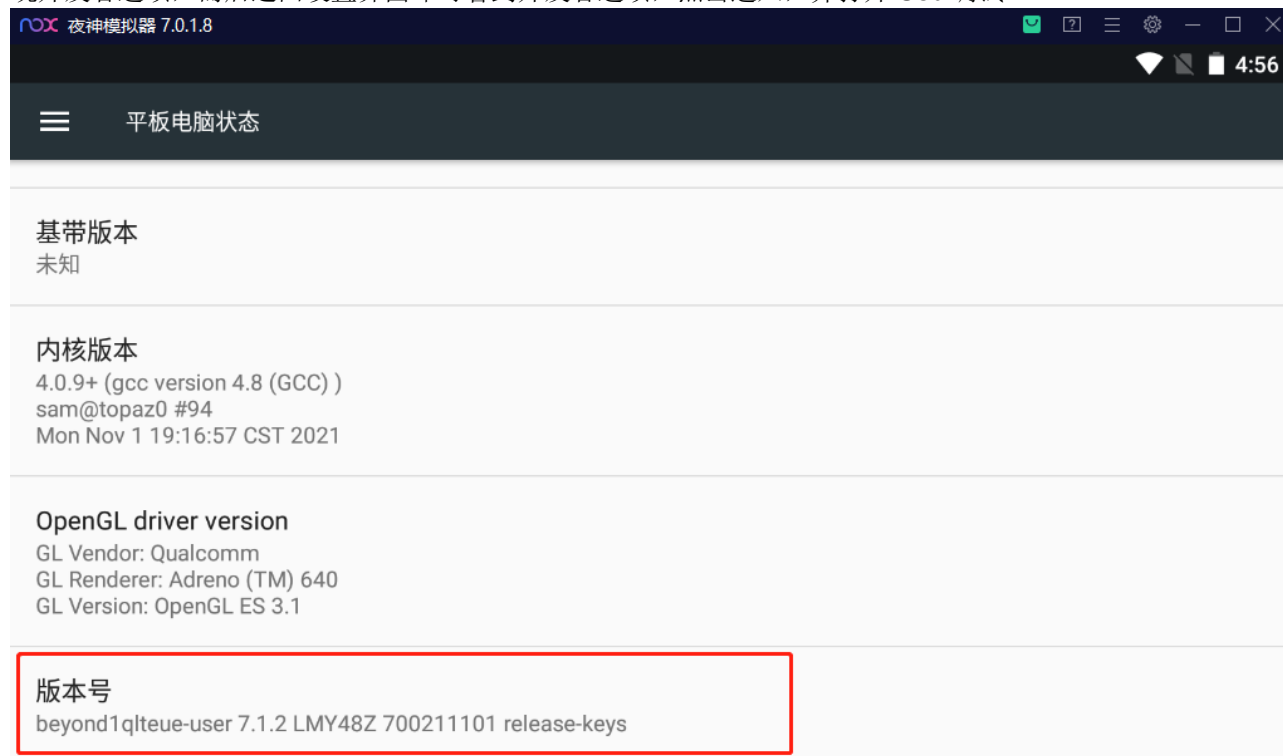


Figure 10



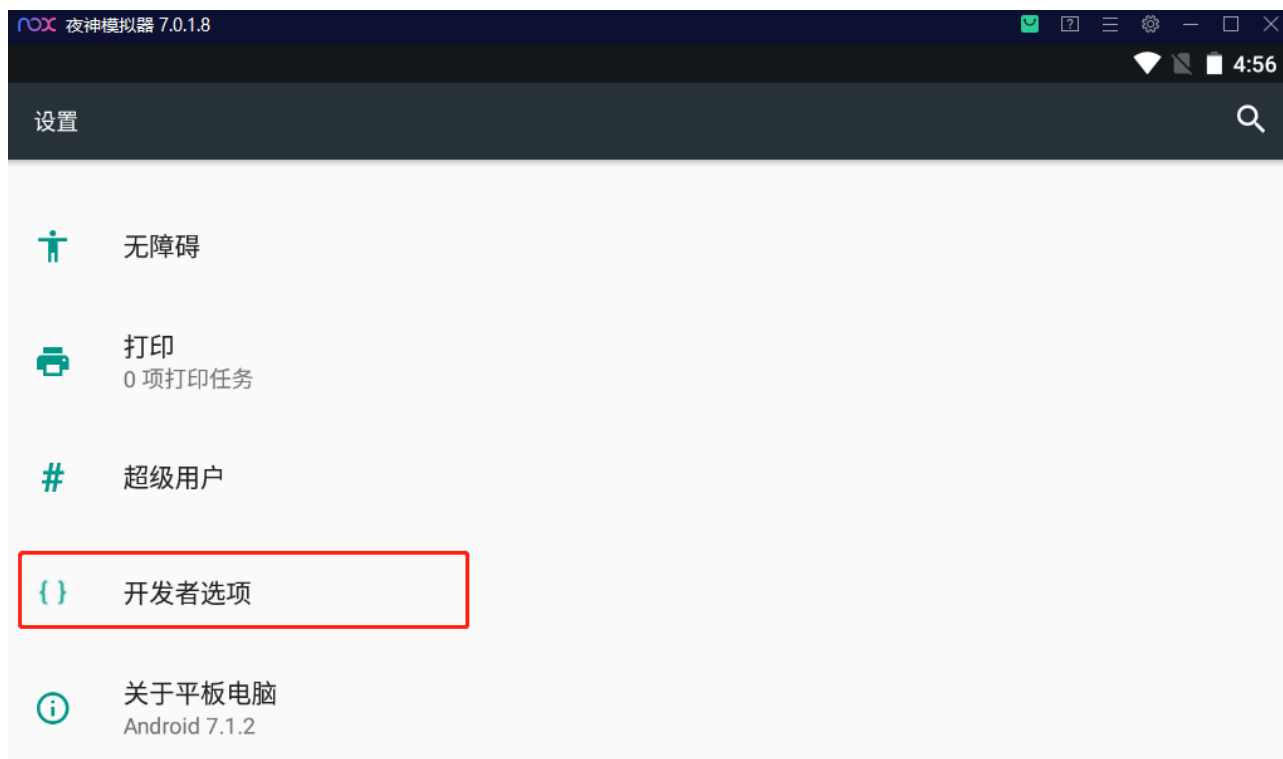


Figure 11



Figure 12





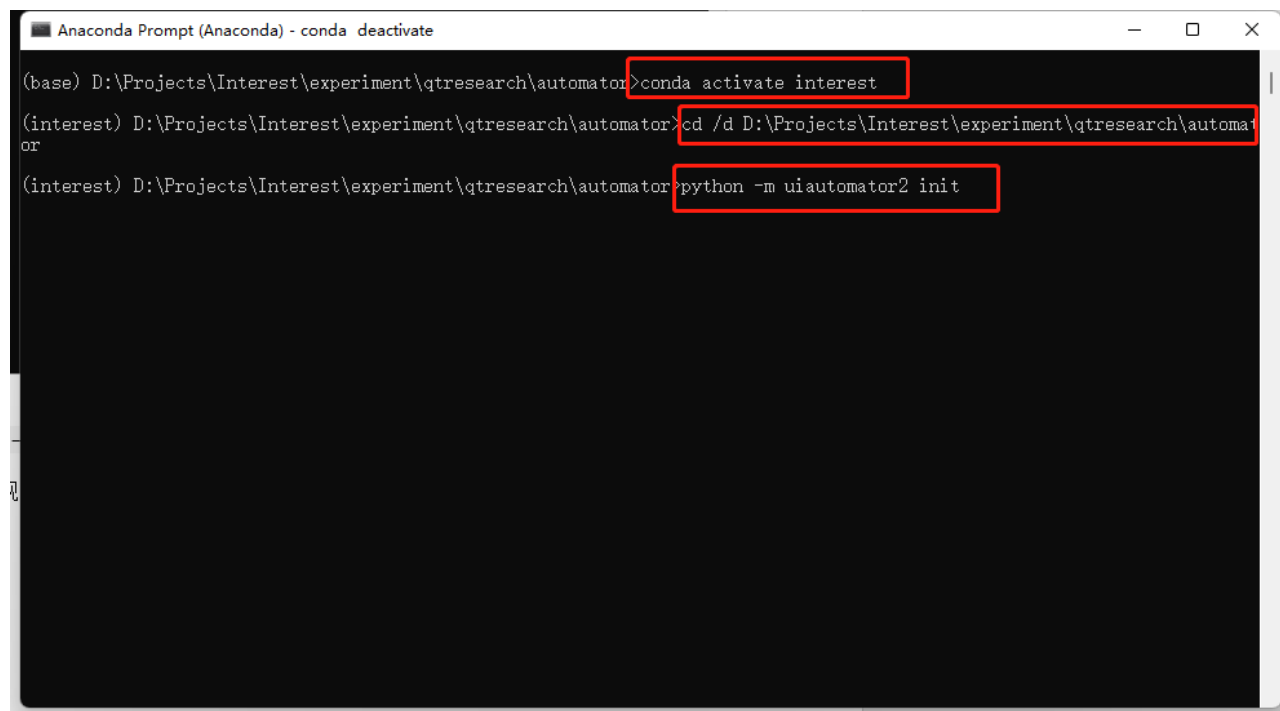
Figure 13



Figure 14

Figure13-14:在安卓模拟器右上角打开模拟器的设置，然后点击手机选择开启网络桥接模式。





```
Anaconda Prompt (Anaconda) - conda deactivate
(base) D:\Projects\Interest\experiment\qtresearch\automator>conda activate interest
(interest) D:\Projects\Interest\experiment\qtresearch\automator>cd /d D:\Projects\Interest\experiment\qtresearch\automator
(interest) D:\Projects\Interest\experiment\qtresearch\automator>python -m uiautomator2 init
```

Figure 15

Figure15: 打开 Anaconda Prompt 进入自定义环境(这里默认大家已经安装好所需要的 Python 包), 进入代码对应目录, 输入 `python -m uiautomator2 init`

Figure16-17: 执行完毕后, 出现 **Successful**, 并可以在模拟器中找到一个叫 **ATX** 的应用, 说明安装完成



```

Anaconda Prompt (Anaconda) - conda deactivate

[I 211123 17:00:24 init:375] abi:x86 not supported well, skip install minicap
[D 211123 17:00:24 init:256] apk-debug package-info: None
[D 211123 17:00:24 init:257] apk-debug-test package-info: {'package_name': 'com.github.uiautomator.test', 'version_name':
', 'version_code': '', 'flags': ['DEBUGGABLE', 'HAS_CODE', 'ALLOW_CLEAR_USER_DATA', 'ALLOW_BACKUP'], 'first_install_t
time': datetime.datetime(2021, 11, 21, 10, 4, 48), 'last_update_time': datetime.datetime(2021, 11, 21, 10, 4, 48), 'signa
ture': 'ae17cd86'}
[I 211123 17:00:24 init:385] Install com.github.uiautomator, com.github.uiautomator.test 2.3.3
[D 211123 17:00:24 init:167] Shell: ('pm', 'uninstall', 'com.github.uiautomator')
[D 211123 17:00:24 init:167] Shell: ('pm', 'uninstall', 'com.github.uiautomator.test')
[D 211123 17:00:24 init:59] Use cached assets: C:\Users\19234\.uiautomator2\cache\app-uiautomator.apk-d3f17174fb\app-ua
utomator.apk
[D 211123 17:00:24 init:238] Push to /data/local/tmp/app-uiautomator.apk:0644
[D 211123 17:00:24 init:167] Shell: ('pm', 'install', '-r', '-t', '/data/local/tmp/app-uiautomator.apk')
[I 211123 17:00:25 init:331] - app-uiautomator.apk installed
[D 211123 17:00:25 init:59] Use cached assets: C:\Users\19234\.uiautomator2\cache\app-uiautomator-test.apk-652bf9e13c\ap
pp-uiautomator-test.apk
[D 211123 17:00:25 init:238] Push to /data/local/tmp/app-uiautomator-test.apk:0644
[D 211123 17:00:25 init:167] Shell: ('pm', 'install', '-r', '-t', '/data/local/tmp/app-uiautomator-test.apk')
[I 211123 17:00:26 init:331] - app-uiautomator-test.apk installed
[D 211123 17:00:26 init:167] Shell: ('/data/local/tmp/atx-agent', 'server', '--stop')
[D 211123 17:00:26 init:295] Real version: [0, 10, 0], Expect version: [0, 10, 0]
[D 211123 17:00:26 init:167] Shell: ('/data/local/tmp/atx-agent', 'server', '--nouia', '-d', '--addr', '127.0.0.1:7912')

[I 211123 17:00:26 init:350] Check atx-agent version
[D 211123 17:00:26 init:359] Forward: local:tcp:55629 -> remote:tcp:7912
[D 211123 17:00:26 init:359] Forward: local:tcp:55629 -> remote:tcp:7912
[D 211123 17:00:26 init:362] atx-agent version 0.10.0
[D 211123 17:00:26 init:365] device wlan ip: 192.168.1.107
Successfully init AdbDevice(serial=127.0.0.1:62001)

(interest) D:\Projects\Interest\experiment\qtresearch\automator>

```

Figure 16

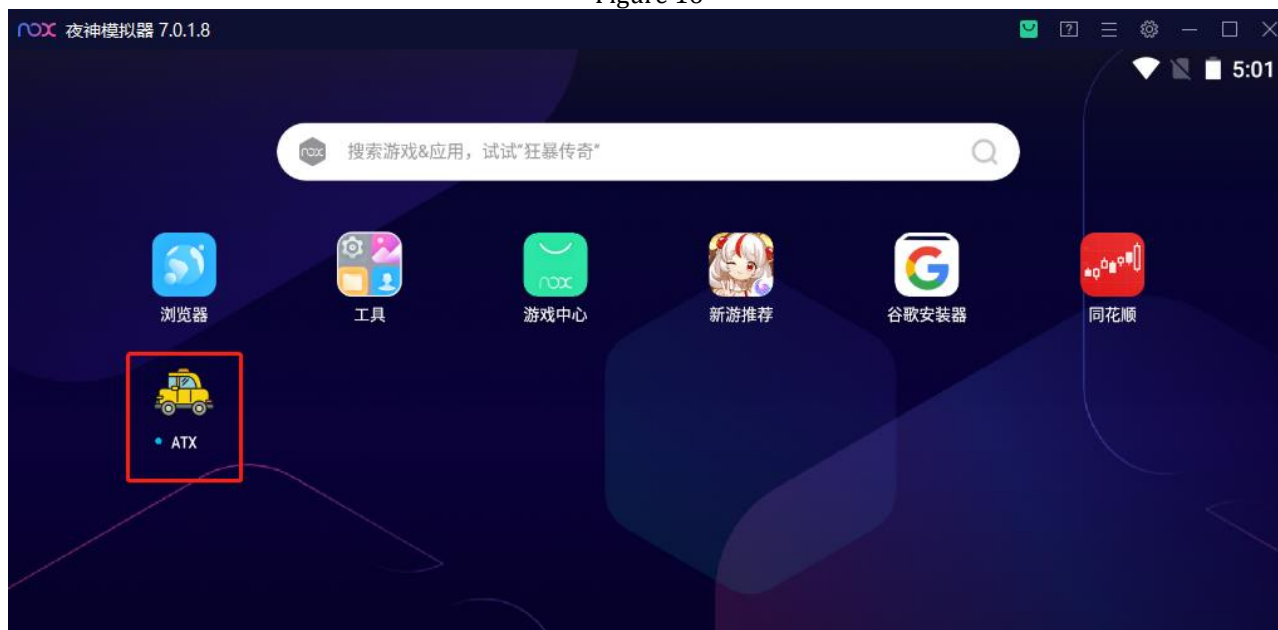


Figure 17





Figure 18

Figure18: 打开 ATX 点击启动 UIAUTOMATOR,可以看到上方 UIAutomator Running 显示为正常服务状态说明启动成功。



Figure 19





Figure 20

Figure 19-20: 打开同花顺，创建并登录自己的账号。



Figure 21

Figure21: 切换至模拟交易区



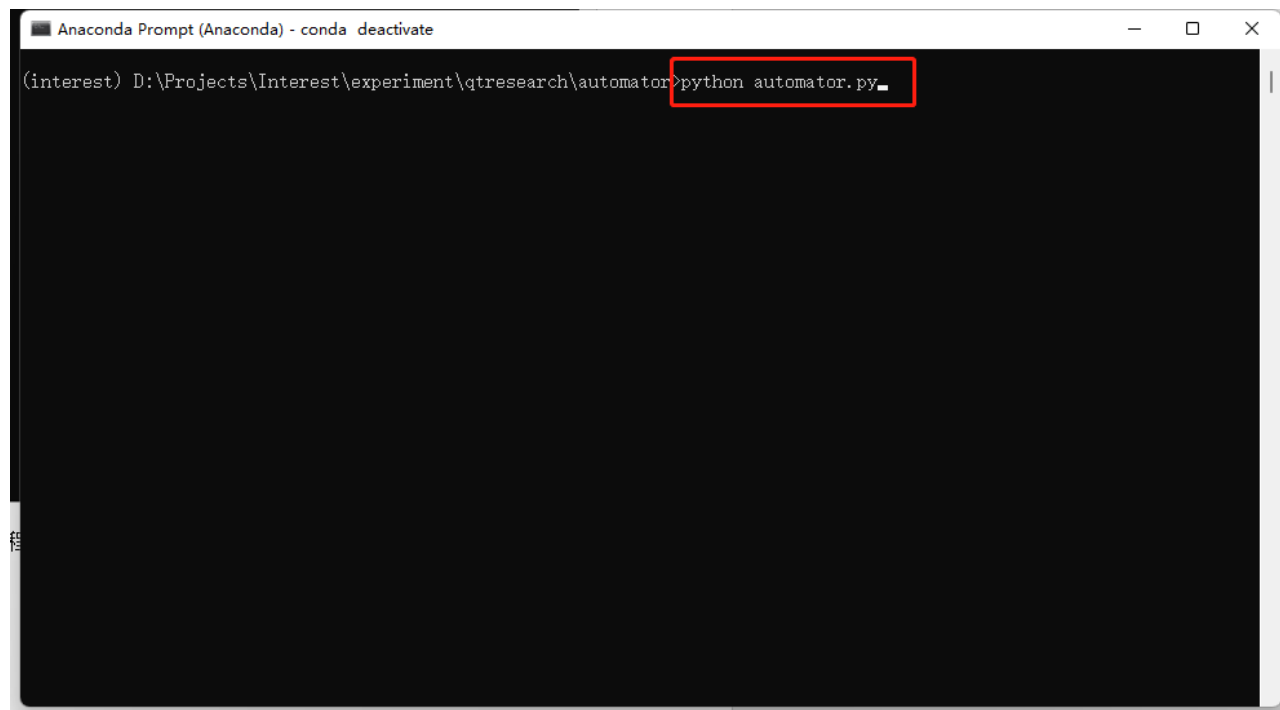


Figure 22

Figure22-23: 输入测试代码，可以看到应用程序中会自动操作进行对应股票的买入

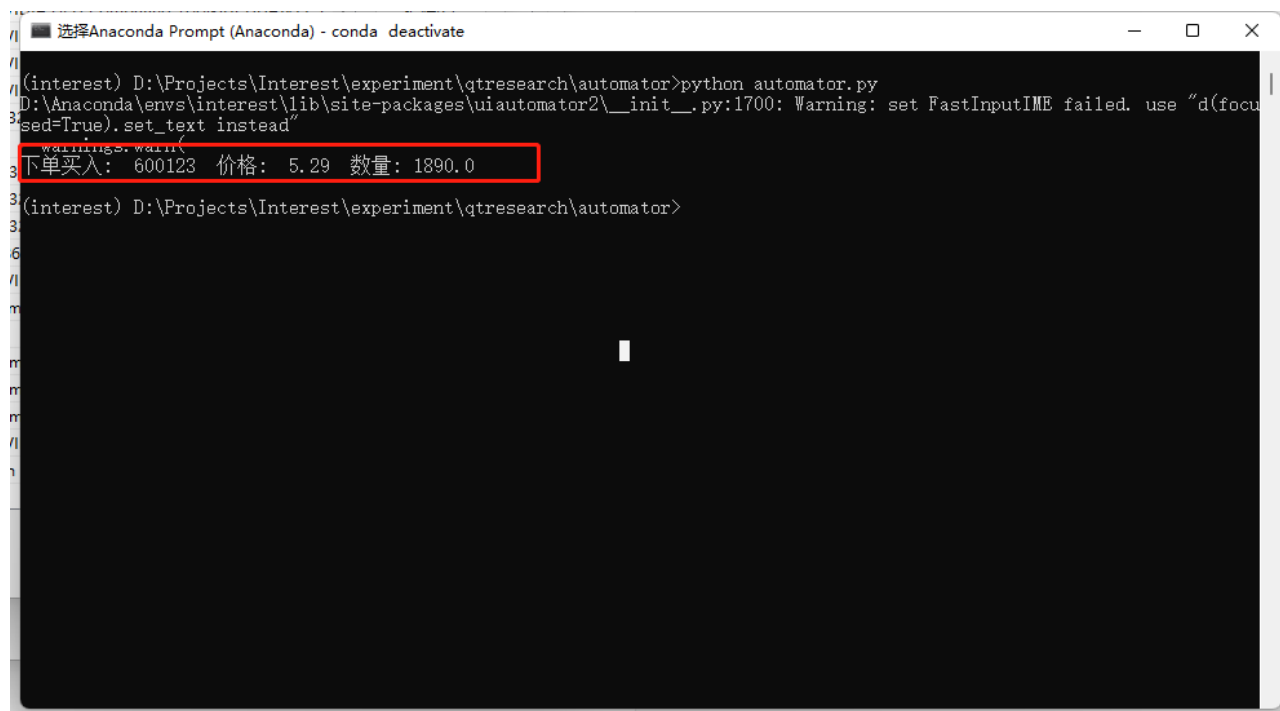
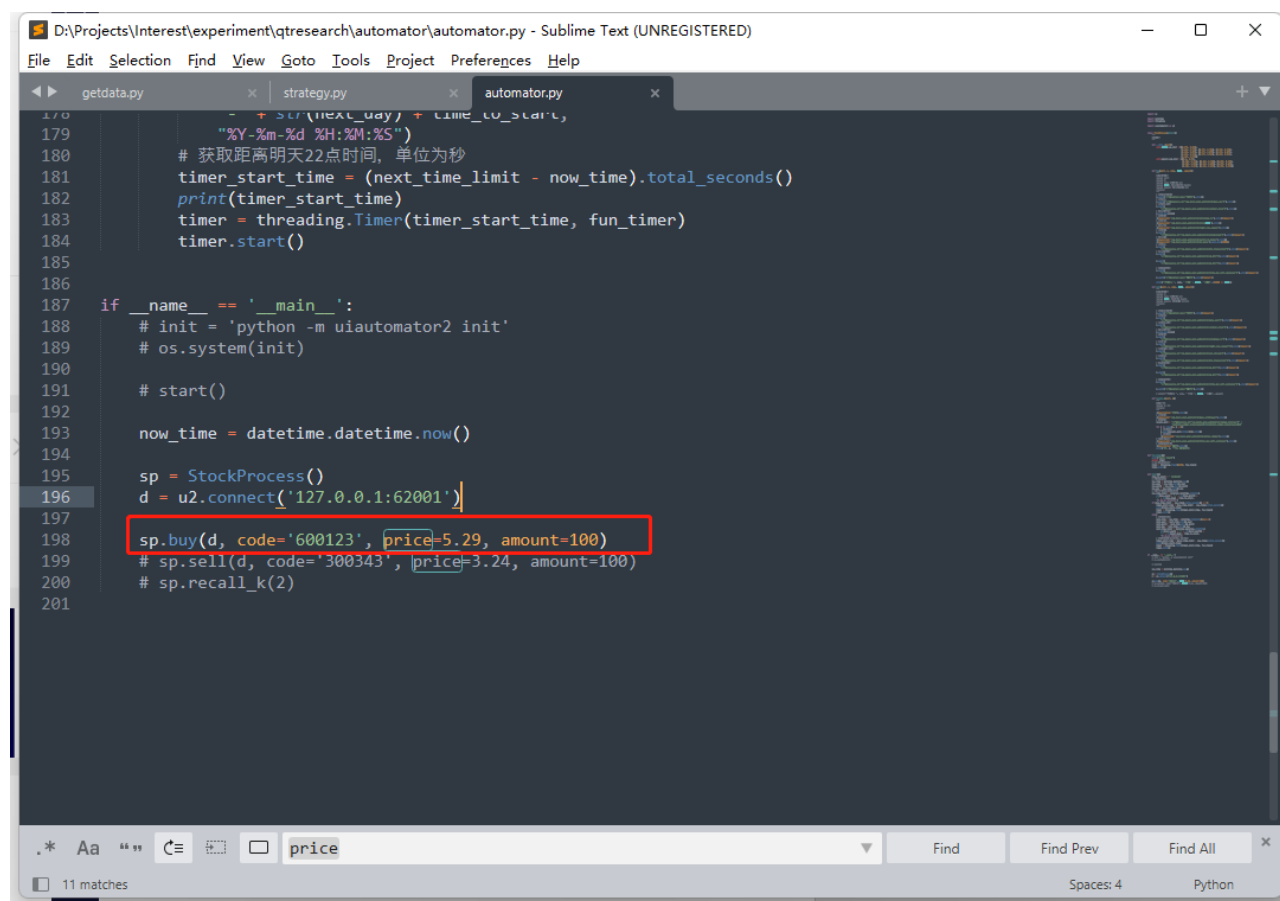


Figure 23





```
178     + str(next_day) + time_to_start,
179     "%Y-%m-%d %H:%M:%S")
180     # 获取距离明天22点时间, 单位为秒
181     timer_start_time = (next_time_limit - now_time).total_seconds()
182     print(timer_start_time)
183     timer = threading.Timer(timer_start_time, fun_timer)
184     timer.start()
185
186
187 if __name__ == '__main__':
188     # init = 'python -m uiautomator2 init'
189     # os.system(init)
190
191     # start()
192
193     now_time = datetime.datetime.now()
194
195     sp = StockProcess()
196     d = u2.connect('127.0.0.1:62001')
197
198     sp.buy(d, code='600123', price=5.29, amount=100)
199     # sp.sell(d, code='300343', price=3.24, amount=100)
200     # sp.recall_k(2)
201
```

Figure 24

Figure24-25: 用户可在程序中设定自己买卖的条件与时机并制定自己的策略(获取数据在 `getdata.py` 中, 编写策略在 `strategy.py` 中)

5.2. Usage Scenario 2

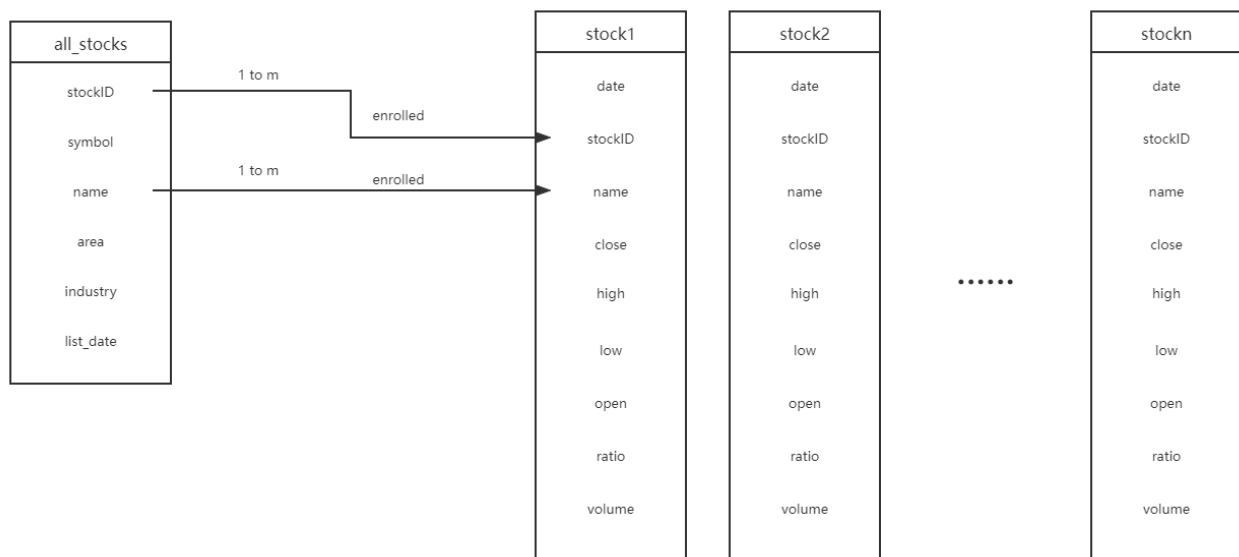
<Do the same as Usage Scenario 1. Add additional usage scenario n below as needed.>



6. Database Design (If Applicable)

<Describe your database design here. You can put your ER diagram in this section.>

我们计划使用 **Mysql** 作为数据库来存储股票信息和各个股票的历史数据，用户可以自定义获取数据的途径这里只展示其中一种方式。



7. Diagrams

<C Programming course: Control Flow diagram; Java: Class Diagram

