****

**BURSA TECHNICAL UNIVERSITY**

**FACULTY OF ENGINEERING AND NATURAL SCIENCES**

**COMPUTER ENGINEERING DEPARTMENT**

**GRADUATION PROJECT REPORT**

**BLOCKCHAIN APPLICATION FOR SMART FARMING**

**Hasan Ergin Balta**

**Advisor : Yrd. Doc. Dr. İzzet Fatih Şentürk**

**January 2020**

**BLOCKCHAIN APPLICATION FOR SMART FARMING**

**SUMMARY**

Project topic is blockchain application for smart farming. I used blockchain technology in this project. Customers can see where the products on shelves come from easily. The aim of the project is to show people how the products they consume come from the field to the market. There are 3 types of users in the project. These are customers, users and miners. When all users enter the system, they connect to other users in the system through peer-to-peer network connection. The main users using the system are customers. Customers search any product inside the system by using product name. Without any customer input in the system, customers only search for products in the search section. Users can only prepare blocks to be added to the chain and send them to the miners. we can think that miners are servers. if too more miners in system, systems have too more servers. Customers and users connect any miners for access in the system. All data traffic happens through packets. All the packages come to the miners and processed there. Miners do not need any manual operation. The system does all the miners work in the back. Human error was thus minimized and saved from time. I developed this project with Python programming language and pythons modules. Only users register and enter this system. Unlike normal block chain systems, miners can only determine the company to use and add them to the database. Customers just search the products. They don’t need to create account for searching.

**INTRODUCTION**

Contrary to common knowledge, block chain applications use peer-to-peer architecture as opposed to client-server architecture. With the introduction of the bitcoin into our lives, we met with blockchain. Nowadays, although blockchain applications are popular around the world, blockchain has just begun to be heard in our country. What we call a block chain is actually a data structure. The block chain, which has become popular in the field of crypto money mining, has begun to and will continue to evolve into different sectors today. The traditional architecture of the World Wide Web uses a client-server network. In this case, the server keeps all the required information in one place so that it is easy to update, due to the server being a centralized database controlled by a number of administrators with permissions. Unlike the client-server applications, data is stored in any users in system, not in a single centre. Any users can access system data easily and if any user wants to be a miner(like database). user just send a be miner request to any miner all nodes are connected with each other. Blockchain-based, traceable tracking systems that are used by large corporate companies like Walmart are almost non-existent in our country. For example, Walmart uses blockchain to track products on their shelves. shows where and how a product comes to the market shelf. Contrary to common knowledge, there is a Peer-to-Peer network architecture in the background instead of Client-Server architecture, and every user in the system can view the data entered in the system freely.

When we look at the structure of the block chain consists of blocks and each block in the chain, keeps the hash information of the previous block, indicating that it is connected to the previous block. In this case it provides us with more secure data storage. Blocs are consist of 4 part in this system. Hash, Previous Hash, Transaction and Timestamp. Can be thought of as an encrypted private id. Hash is data encrypted with certain standards. sha256 and sha1 algorithms are generally used. Sha256 hash algorithm used in this project. Transaction is the actual data stored in the chain. It is transactionId, product id, product name, product number, from place, to place and operation date in this system. Users send transaction adding request to all miners. All the miners to whom the request come, enter race for add transaction in to chain. This race called Proof of Work(PoW). There is a hash standard and usually changes periodically. In Bitcoin, miners enter data in to an hash function continuously and even use brute force for enter this data. The first 10 digits of the output of this function must be zero to win the race and it should be done within 10 minutes. If any miner done before others it can add transaction in to blockchain. But before adding, winner sends win message to other miners and if over the half of the other miners proof the transaction, winner can add it in to blockchain. However, Works are more different than bitcoin in this system. The race rule of this system is, if there are 3 zeros in the part of a randomly generated number after the point, the miner who catches that number wins. The winner send win message to other miners and miner adds transaction in to blockchain.

**File Structure**

There are 4 folder in this project. these are main, UI, services and datas. There is main.py file in this directory. Users UI files are in UI folder. Those are Miner.py, Seacrh.py, SignUp.py and User.py. All users operation functions are inside Services folder. MinerService.py, SearchService.py, UserService.py, LogInService.py are inside this folder. All the functional works in the project are in these files. Finally Datas folder store the all of the system datas. There are 3 json(JavaScript Object Notation) files in there which are blockchain.json, userInformation.json and transactionQueue.json. blockchain.json is the most important file for this system because the main system datas in its. Blockchain.json store blockchain. userInformation.json just store users information and TransactionQueue.json store transaction datas.



**Datas**

**1-) blockchain.json**



**2-) userInformation.json**

****

**Packet Types**

All data traffic is happening with packets. Each packet consists of 3 parts which is **Packet Type, Packet Data** and **Sender Information(**sender host and sender port**).** Packet types are **TRN, SRC, BLK, WNN, INF and ENT.** TRN is header of transaction, SRC is header of search packet send by customers, WNN is header of the packet which is send by winner miner to other online miners after transaction adding race, BLK is the header of the after winner miner sends WNN packet, is a package that sends a block chain piece to other online miners. ENT header of the users login packets. userId and password are inside ENT packet and users send this packet to any online miners for enter the system. INF header of the packet to send for register to system. This packet data are userId, password,name,surname,company,status.

**Services**

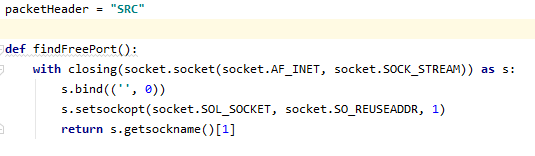
There are 4 services in this project. Those are SearchService, MinerService, LogInService and UserService. SearchServce has all search operations functions. LogIn service has routing function for log in operation and LGN packet creator function. UserService has all users operation functions. MinerService has all miners operation functions.

**1-)SearchService**

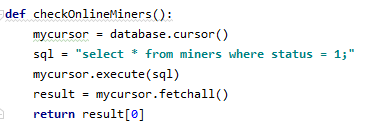
Firstly **mysql.connector** modeule, **socket** module and **closing from contextlib** imported for this service. Mysql.connector module used for connect and send sql query to miners table in mysql database. Socket module and closing module used for finding free port for customer. When customer search any product system get customers host and give it free port and miner send response to this host and port.



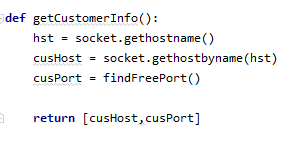
Database is mysql connector object. connect mysql database by using **connect()** function. Parameters of this function are connection information. Username, server host, server port, password and database name.



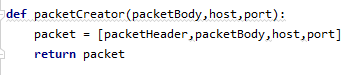
Here **packetHeader** and **findFreePort()** function. This function find free port to connect miner for search operation. Closing module and socket module used in this function.



**checkOnlineMiners()** function find any online miner to connect for search operation. Mysql.connector module used in this function. Created select query to miners table and this query result is miners which status is 1 ( online). Send query to mysql database by used Execute(sql) function. Result is return output of this query comes from database. And this function return one row from miners table. Customer know that which miner online for search operation by this function.



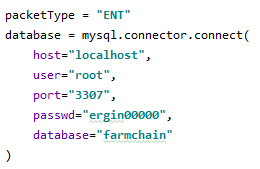
Here is **getCustomerInfo()** function. This function returns a list and customer host and customer port is attribute of this list. Socket module used for find customer host. And findFreePort function used for give free port to customer for search.



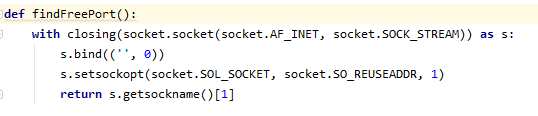
**packetCreator()** function is created search packet to send miner for search product. This function returns a list and packet header, packet body and sender information are inside this list. Packet header is “SRC” and packet body is product name. this packet send to miners for search operation.

**2-)LogIn Service**

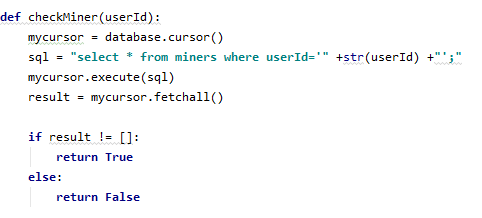
Mysql.connector, socket, closing from contextlib and json modules imported firstly. This service has login operation functions and also has packetType variable. packetType is “ENT” for this service.



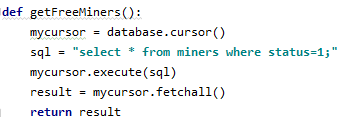
database is mysql database connection object. this object is inside all services and used **connect()** function from mysql.connector modules for connect database.



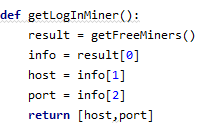
This function find free port for log in operation.



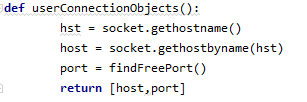
This method returns **True** or **False**. It checks the logged in user is miner or not. Firstly created sql query and it show row from miners table. if logged in users id is in miners table, result is that row but if there is no user in table result is empty.



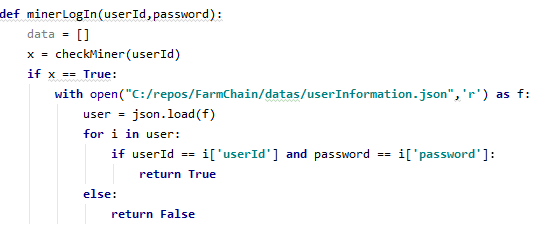
This method return all miners whose status is 1. While doing this operation, a query was prepared first, all rows with status 1 were assigned to the result variable with the select query.



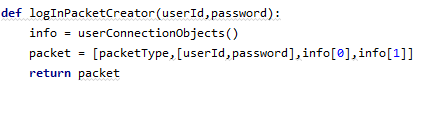
this method returns miner information that the ENT package will be sent to. getFreeMiners() called in result and also result is a list. This method returns first index of result list.



this method returns logged in user host and port for connect to miner by using socket.



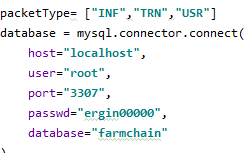
Only miners use this method for log in. here **checkMiner**() called and assigned x variable. If x is True logged in user is miner and userId and password checked **userInformation.json** file. Json module used in this method and it returns **True** or **False.**



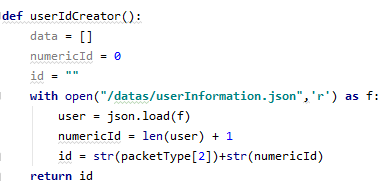
Prepare ENT packet for log in operation. This packet is important part of log in operation.

**3-)User Service**

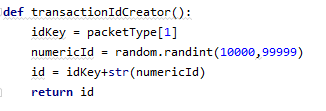
Firstly mysql.connector, random, json, socket and closing from contextlib modules imported for this service.



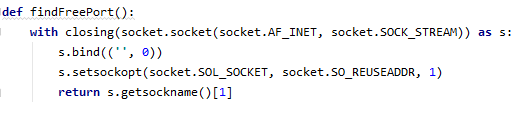
Packet type is all packet header send by the user. INF, TRN and USR types store a packetType list. Database is **connect()** function of mysql.connector module and it provides database connection. Parameters of connect function are host and port of local database server, username and password of database and database name.



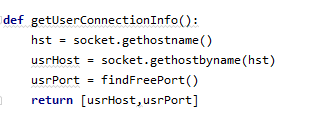
This function returns an user id for users. User id has two part. First part is USR key and second part is numeric part. Numeric part increases automatically. First step, read userInformation.json file and write it into user list. Numeric part is length of user list + 1. After USR key and numeric id combined in id string and function return value is id string.



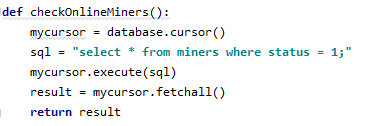
All transaction has randomly generated ids and this id used to get regular. There are two part of transaction id. First part is TRN key, second part is numeric part. Numeric part generated randomly and random module used for this. **Randint()** function generate random number in random module. Parameters of this function 10000 and 99999. It means generate random number between 10000 and 99999. TRN key and numeric id combined in id string and function returns id.



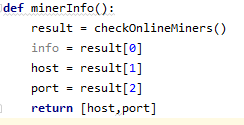
This function find free port in local machine for connection. Closing and socket modules used in this function and it returns any free port number.



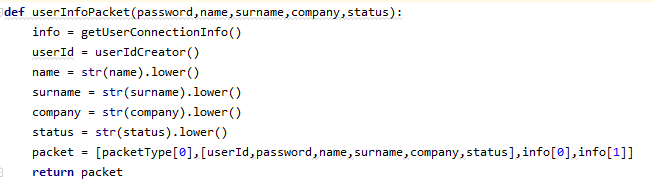
This function returns user host and port numbers for connection. Socket module used for get user host and **findFreePort()** function used for user port. User host and user port combined a list and this list return value of this function.



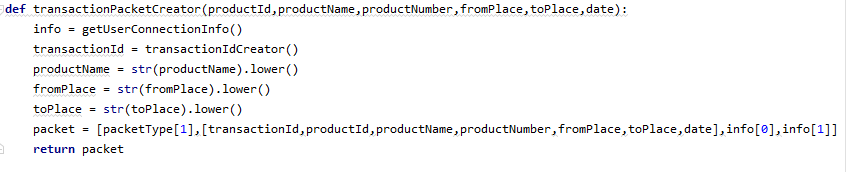
checkOnlineMiners() function, get miners whose status is equal 1 (True) for connection. Simple sql select query created and after execution of this query, result is rows of miners table in result list. And this function returns result list.



**minerInfo()** used for get one miner host and port for connection. **checkOnlineMiners()** function called in here and this function only select first index of result list and weed out host and port and combine them into list. This function returns this list.



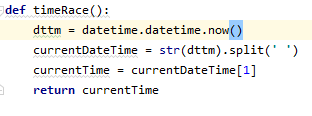
userInformationPacket() is created user register packet. There are password, namei surname, company and status in this packet and this packet send all miners. Called userIdCreator() for userId, and other information will be taken from user. Status is position in the company.



This function created a transaction packet and this packet will be added blockchain by miner. productid, productName, productNumber, fromPlace, toPlace and date are parameters of this function and taken from user. transactionIdCreator() function called for created transaction id. All of these variables combine a packet list. This function returns packet list.

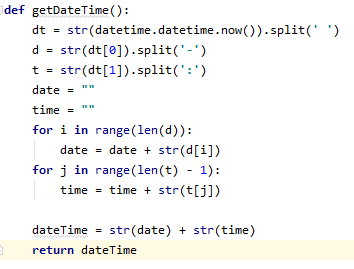
**4-) Miner Service**

Random, json, hashlib, datetime, socket, mysql.connector and closing from contextlib modules imported firstly for this service. This service separate packets and do operations for its packet. All packets comes to miners and miners do all operations.

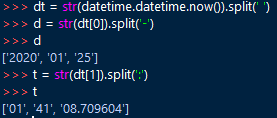


timeRace() function is time race between miners after proof of work operation. After proof of work operation, winner miner send with win message this functions output. Used datetime module for get current time.

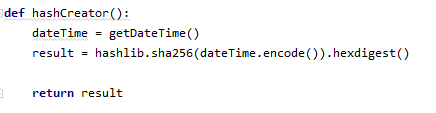
 dttm format is here. Split it and write it in currentDateTime list. CurrentTime is 1st index of this list and this function returns it.



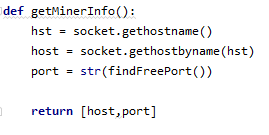
This function returns current time as string. Dt is a list and it store current date and time. First index of list store current date, last index of list store current time. D is list of first index of dt lists by split ‘-‘. T is list of last index of dt by split ’:’.



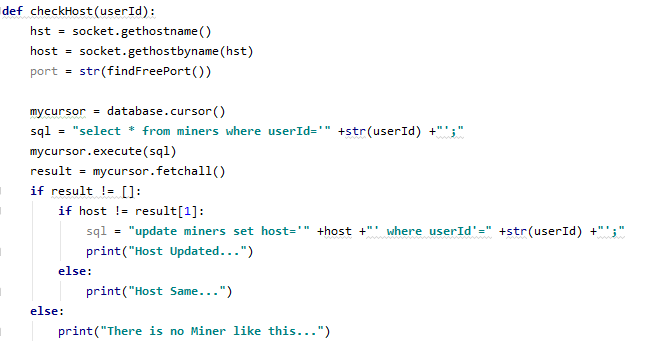
And this function returns combine d and t into string.



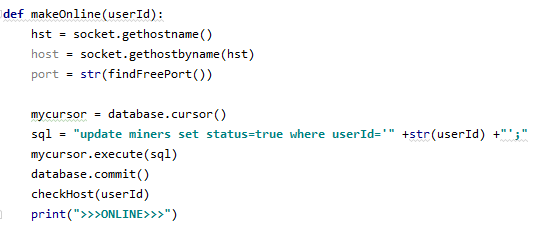
Hash is very important term for blockchain. All transactions have hash value and all hash are different to another. This function returns hash value. The parameter of the hash function is a string and extracts that string in an encrypted form. SHA256 algorithm used for create hash. Hashlib module in python has sha256() function and parameter of this function is string. For this project hash string is current date time for this reason called getDateTime() function for get current date time. Result encrypts date time and returns hexdigest.



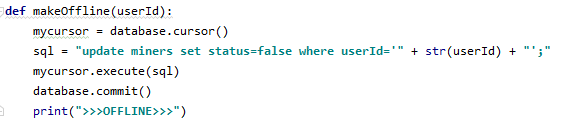
This function returns miners host and port. When any miners enter system, miners host can change and if miners host change, this function get current host of miners. And called findFreePort() function for find a free port for miner. This function returns list and attribute of that list is host and port. Used gethostbyname() function from socket module for get hostname.



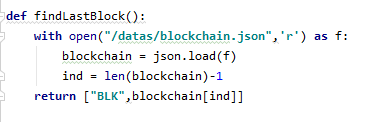
This function checked miners host and port while any miner enter into system. Search by userId into miners tabele in database and after execution of this query result is empty, print that message. However, if result is not empty checked host and port of that miner. If miners host and port are different from database, update host and port into database and print host updated message.



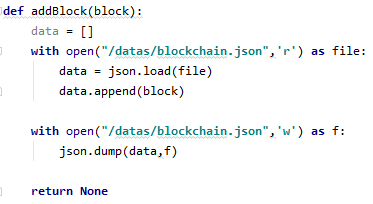
When any miner enter system, miner should update status from 0 to 1 in database. This function creates sql update query and after execution miner status will be 1 and miner is online.



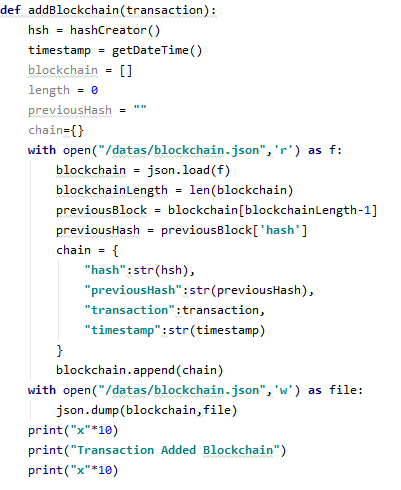
When any miner logging out the system, miner should update status from 1 to 0. Sql is update query and it changes status and make it offline. After execution print message to screen.



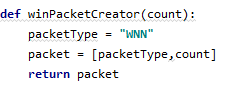
After proof of work operation, winner miner send added block to other miners. This function return ‘BLK’ packet. BLK packet has last added block in blockchain. open blockchain.json file and read them. Blockchain is a list and ind is index of last added block.



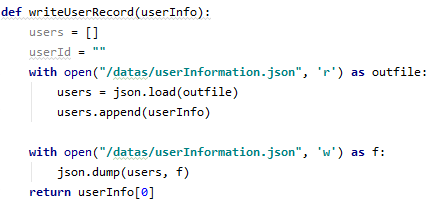
When miner receive BLK packet, this function add it into blockchain. parameter of this function is a list. Block is body of BLK packet. Firstly blockchain.json file read. Block is appended data list and data overwritten to blockchain.json. this function returns none , it just add.



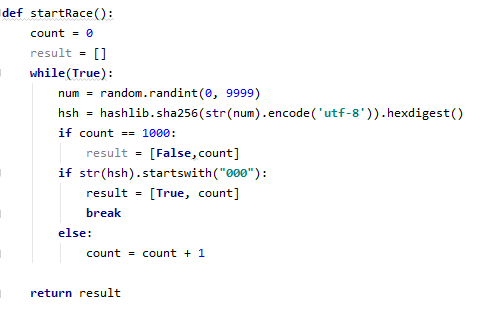
This function is very important function for this project. Shortly, addBlockchain() function add hash to received transaction and write it into blockchain.json file. Parameter of this function is is transaction list. Firstly hash is created by called hashCreator() function . timestamp created by called getDateTime() function. Read blockchain.json and write it into blockchain list. Chain dictionary created and that dictionary appended into blockchain list and this blockchain list overwrite into blockchain.json file. Print message to screen.



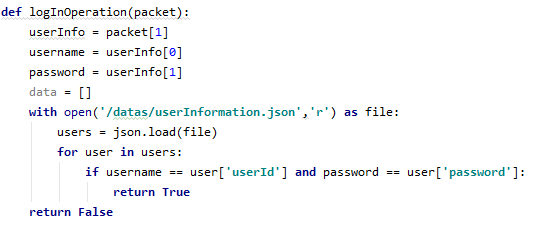
After proof of work operation winner miner create a WNN packet and this packet send other miners. winnPacketCreator() returns WNN packet list. Parameter of this function is finishing time of race condition.



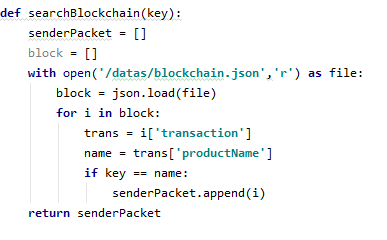
Parameter of this function is userInfo list. When user registered, miners receive USR packet. This function add USR packets datas into userInformation.json file and returns userId of new user.



When transaction comes to miners, startRace() function started. This function proof of work function. random number created between 0 and 9999 and that number convert int to string. Hsh is a hash by using sha256 algorithm. Generated random number was put into this function. there is one condition. The first three digits of the hash formed must be zero. The first three digits continue to generate a random number until it catches zero and continues to put it in the hash alogitization. If he runs out within the specified time, he will win the race. and returns the result value. The result value is a list. The elements of this list have a boolean value and when they finish.



Packet is coming from users. Parameter of this function is ENT packet. Function separate ENT packet and get userId and password from packet. This function checks this information from the userInformation.json file and returns a boolean value.



When SRC packet comes to miner, this function activated. Key is product name of SRC packet. Sender packet is search result. Firstly, blockchain.json file readed and key searched in that file. If key is inside of any block of blockchain, that block append to senderPacket list. After this operation this function returns senderPacket list.



The most important function is packetSeperator(). Parameter of this function is packet. Packet is coming from users, customers or other miners. This function goal is separate and process packets. Packet[0] is packet header, packet[1] is packet body, packet[2] sender host and packet[3] is sender port.

If packet type SRC, call searchBlockchain() function and append the function output in response list. There are sender of packet host and port information in response packet. And if packet type SR, function returns that response list.

If packet type ENT, call logInOperation() function and packet is also parameter of its. And output of that function and sender of packet host and port information append in response list. If packet type ENT, function returns that response list.

If packet type INF, call writeUserRecord() function. output of this function and sender of packet host and port information append in response list. If packet type is INF, function returns that response list.

If packet type is WNN, packet[0] is WNN, packet[1] is finished time of proof of work. Called timeRace() function for miner’s time and if WNN’s time is greater than miner’s time, function returns false. Else returns true.

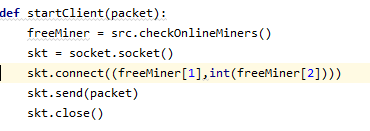
If packet type is BLK, only called addBlock() function and packet[1] is added blockchain.

**UI**

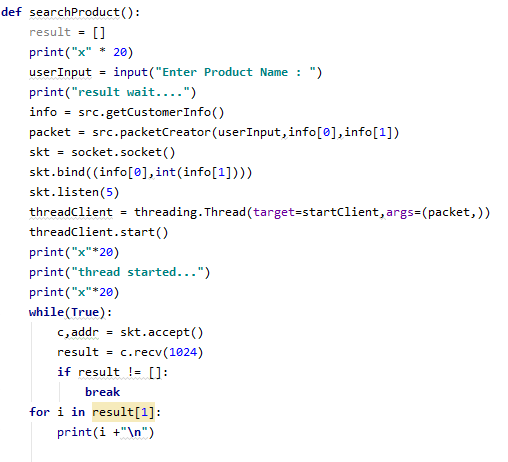
There are project screen in this folder. Miner.py, search.py, signUp.py and user.py files in this folder. Miner.py is miner user screen. All miner operations work automatically. For this reason miner just log in the system. Search.py file is search screen. Just customer or who want to search product use this screen. Just enter product name and search product. User.py is normal users screen. User just send transaction to miners but users must log in the system first. signUp.py is register screen. If any one wants to be a normal user if it has products for adding blockchain.

**1-) Search Screen**

Firstly imported searchService.py file for using its function, imported threading modüle for start client and server simultaneously and importded socket modüle for network operations.



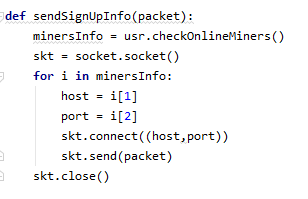
This function start client and send search packet to any free miner. Called checkOnlineMiners() function into searchService for freeMiner. Skt is socket. Connect free miner with skt.connect(). Send src packet with skt.send() and lastly close socket. Parameter of this function is src packet.



Search screen is inside this function. Simple python console user interface used and thread and socket server started in this function. userInput is product name entered by customer. Called getCustomerInfo in searchService for getting customer host and port to connection. Called packetCreator() and parameters of packet creator are userInfo and customer host and port. Skt is socket. Create socket server with skt.bind() and listen network. threadClient is search thread and target of this thread is startClient() function. While server start and listen network, client connect and send data simultaneously by using thread. After that waiting reply from the miner and print reply data.

**2-) SignUp Screen**

Firstly imported userService.py, socket and threading.

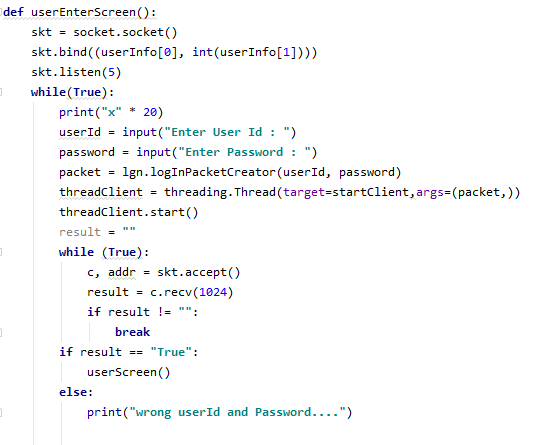


sendSignUpIfo() function, send sign up packet to all miners. Called checkOnlineMiners() function for online miners information. Skt is socket and for all miners, connetc all miners with skt.connect() by using miners host and port and send paket to all. Close socket with skt.close()

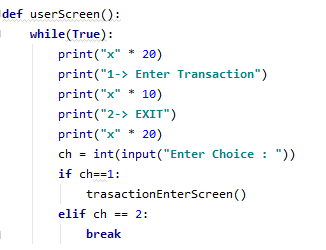


signUpScreen is simple python console user interface. Password , name, surname, company and status getting from the user and called userInfoPacket() function from userService and parameters of that function are everything taken from the user. called getUserConnectionInfo() from user sevice for getting user host and port. Created thread for send packet to miners and target of that thread if sendSignUpInfo function. Skt is socket and start server by using skt.bind(). Thread startded with thread.start() and sending packet and wait reply message. Miners send userId to new user. and print userId.

**3-) User Screen**



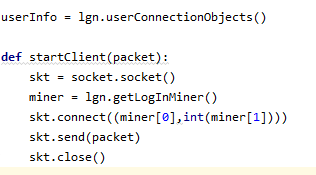
User log in screen. Skt is socket and start socket server with skt.bind() and listen network. userId and password taken from the user. packet is logIn packet. Called logInPacketCreator() for log in packet. threadClient is thread and target of that thread is startClient() function and thread started with shtreadClient.start(). Skt.accept is accept connection. Miner send enter result to user by result. If result is true called userScreen(). İf result is false print message and again enter userId and password.



This screen is simple python console user interface. There are 2 options for users. 1 enter transaction and other exit. Called transactionEnterScreen() for entering transaction.



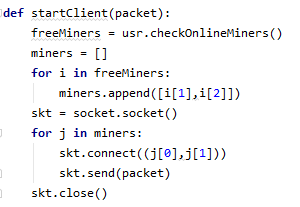
User enter transaction information and send transaction to all online miners with this function. Transaction information taken from user with input function and called transactionPacketCreator() function for creating transaction packet. Skt is socket and start socket server by using skt.bind(). ThreadClient is thread and target of that thread is startClient() function. Stread started with threadClient.start() wait response from miners. Result is reply packet from miners and print result at the end.



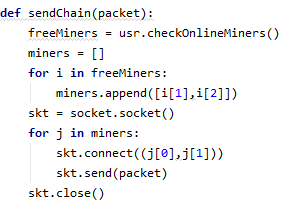
Called userConnectionObjects() from log in service in userInfo. For startClient, skt is socket and called getLogInMiner from logIn service in miner list. Connect that miner with skt.connect() and send logIn packet with skt.send() to miner at the end, close socket with skt.close().

**4-) Miner Screen**

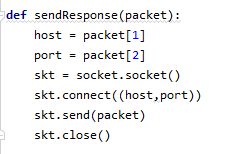
Imported MinerService, LogInService, UserService, threding modüle and socket module.



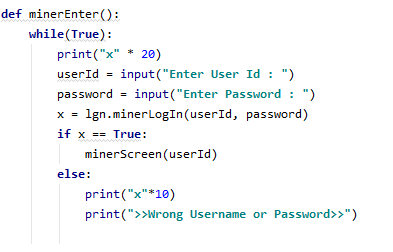
This function send win packet to all miners. Called checkOnlineMiners from user service in freeMiners list and i is elements of that list. İ[1] and i[2] is host and port of miner and append each miners host and port in miners list. Skt is socket and connect and send packet to all miner which is element of miners list.



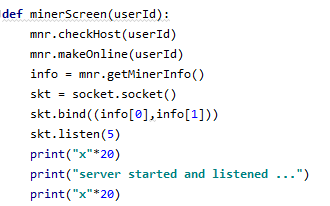
Winner miner send chain to all miners which are join race after add transaction in blockchain.called checkOnlineminers() from userService for miners information. Each element of miner list is 1. And 2. Index of freeMiners list and connect and send to all miners which are inside miners list by using skt.connect() and skt.send().



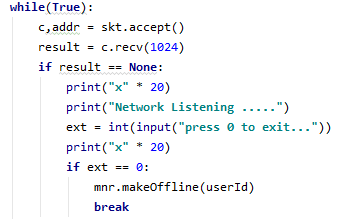
This function send logIn response and search response to users and customers. Seperate host and port into packet and connect that host and port. After connection send packet to that host and port.



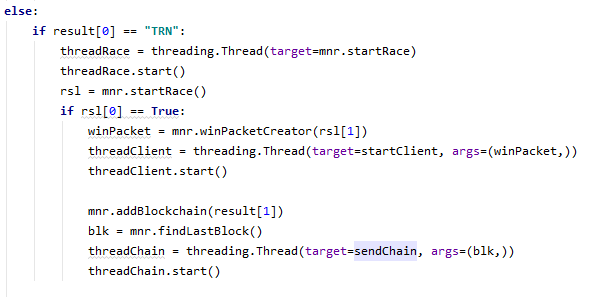
Miner enter the system with this function. userId and passwor taken from miner in infinite while loop and called minerLogIn function from logIn service in x. İf x is true, miner can enter and called minerScreen() function but x is false miner must enter userId and password again.



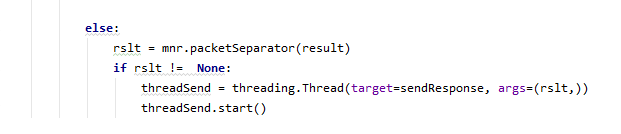
Called checkHost() and makeOnline() functions from minerService. Called getMinerInfo() from miner service for info list. Created socket server by using skt.bind() and listen network. Print message “server started and listened”.



Accept any connection with skt.accept() in infinite while loop and result is receiving data. İf result is empty, print message and if miner wants to exit system press 0 to exit. İf miner press 0, called makeOffline() function and break loop.



İf result is not empty, seperate result[0]. Result[0] is header of receiving packet. İf packet header is “TRN”, threadRace is thread and target of that thread is startRace() function from minerService and tread started with threadRace.start(). Rsl is output of startRace() funtion. İf first index of rls list is True, called winPcketCreator() and parameter of this function is rsl[1]. threadClient is another thread and target of this thread is startClient() function. And start thread with threadClient.start(). Called addBlockChain() from miner service and parameter of this function is 1. İndex of result list. Called findLastBlock() from miner service in blk variable. threadChain is another thread and target of that thread is sendChain() function and start thread with threadChain.start().



İf result[0] is not a “TRN”, called packetSeperator() from miner service for seperate receiving packet. Result list is paremeter of packetSeperater function and output of that function is rslt variable. İf rslt is not empty, threadSend thread created. Target of threadSend thread is sendResponse() function and start thread with threadSend.start().