CoinExchange_CryptoExchange_Java

Public

Watch 75 Fork 1.1k Star 1k

master 57 branches 0 tags Go to file Add file Code

jammy928 Update README.md 8adf508 on Nov 5, 2021 10 commits

00_framework	初始化提交	3 years ago
01_wallet_rpc	初始化提交	3 years ago
02_App_Android	初始化提交	3 years ago
03_APP_IOS	初始化提交	3 years ago
04_Web_Admin	初始化提交	3 years ago
05_Web_Front	初始化提交	3 years ago
06_ExchangeRobot	初始化提交	3 years ago
09_DOC	初始化提交	3 years ago
DEVELOP.md	初始化提交	3 years ago
LICENSE	初始化提交	3 years ago
README.md	Update README.md	2 years ago
管理后台截图.md	初始化提交	3 years ago

README.md

Crypto-Exchange / Coin-Exchange

Maybe The best open source core code exchange in the entire net, the architecture/code quality is visible.

I think this may be the best choice for you to build an exchange or secondary development

> Statement 1: I have been working in the new company. I will take the time to update some descriptive things here so that everyone can compile, build, and develop
> Statement 2: The APP source code and the trading robot source code are not open source (provided for a fee), if necessary, email: 837385225@qq.com

I am a chinese， so chinese is my mother language, But I can also use both English and Japanese to communicate with you.

Introduction

This project is a bitcoin exchange based on Java (SpringCloud) | BTC exchange | ETH exchange | digital currency exchange | trading platform | matching trading engine. This project is based on the development of Spring Cloud microservices, which can be used to build and secondary development digital currency exchanges, and has a complete system component.

- Match trading engine.
- Background management (back-end + front-end)
- Frontend (transaction page, event page, personal center, etc.))
- Native Android APP source code
- Native Apple APP source code
- Currency wallet RPC source code
- Multi Language Support（Simple Chinese、English）

About

The best Java open source crypto currency exchange platform, bitcoin exchange based on Java | BTC exchange | ETH exchange | digital currency exchange | trading platform | matching trading engine. This project is based on the development of Spring Cloud microservices and can be used to build and secondary development of digital currency exchanges...

cryptocurrency otc cryptocurrency-exchanges coinexchange

Readme
Apache-2.0 license
Activity
1k stars
75 watching
1.1k forks
Report repository

Releases

No releases published

Packages

No packages published

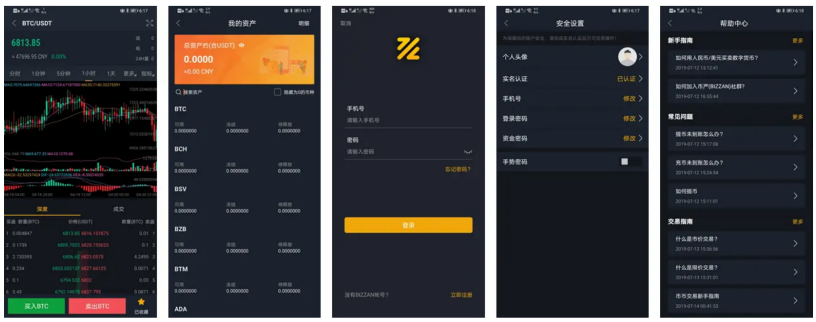
Deployments 5

github-pages 2 years ago

+ 4 deployments

Languages

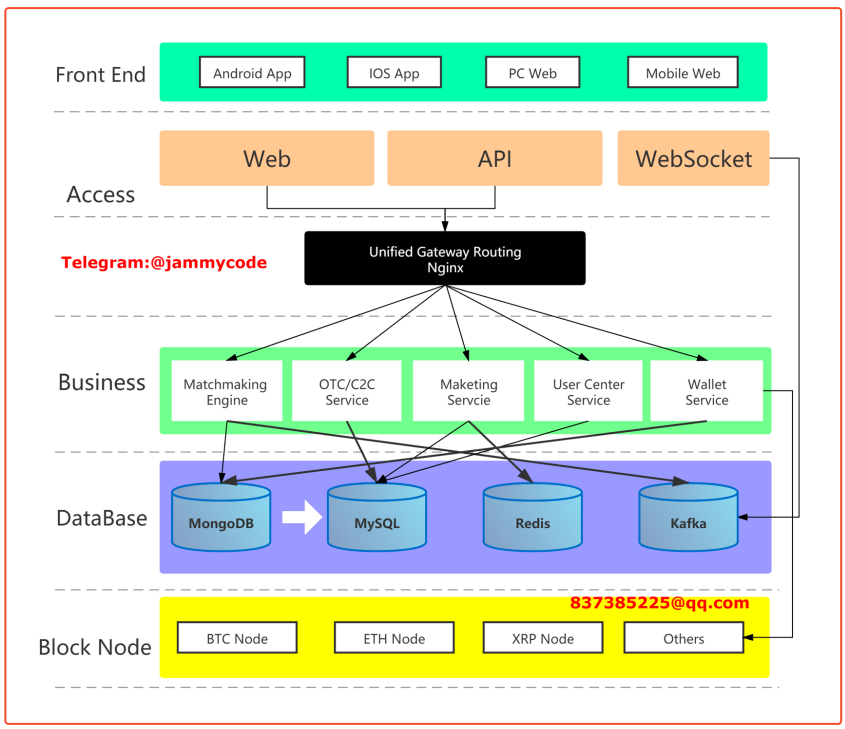
Java 49.9% Vue 33.3% CSS 9.0% JavaScript 6.9% Less 0.5% FreeMarker 0.2% Other 0.2%



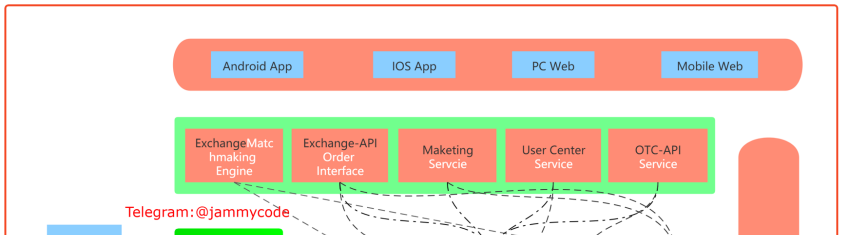
System architecture overview

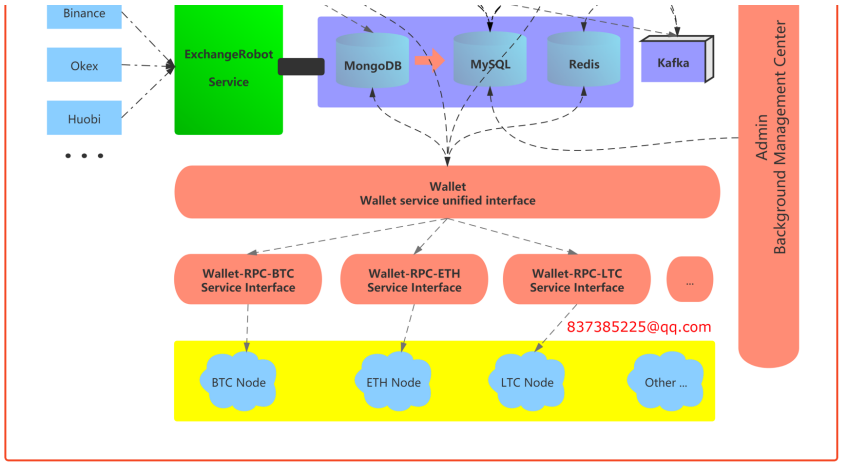
Just draw a few sketches, just look at it。。。

Overall structure

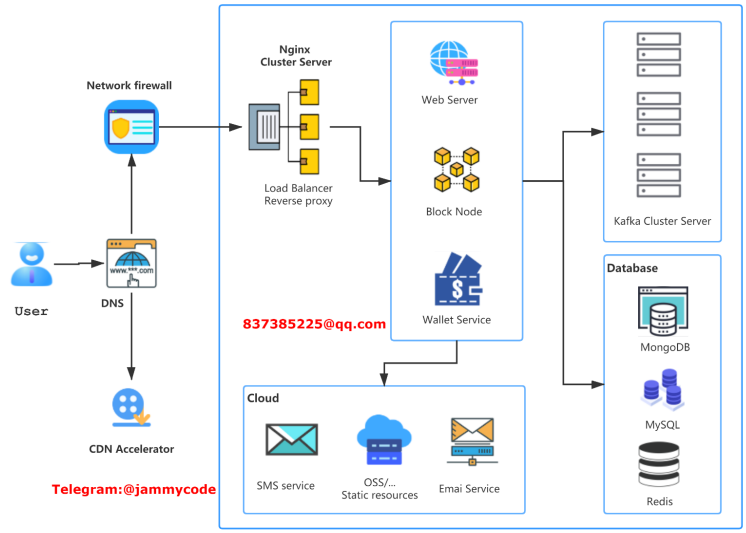


Logical architecture

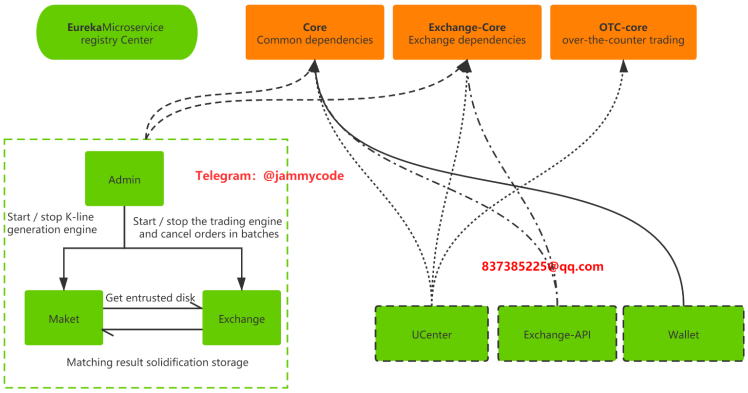




Deployment architecture



Dependencies



System demonstration video

PC front end (user web end)：https://gitee.com/cexchange/CoinExchange/attach_files

Mobile APP：https://gitee.com/cexchange/CoinExchange/attach_files

Management background：https://gitee.com/cexchange/CoinExchange/attach_files

Development Reference

Development Reference Document：https://gitee.com/cexchange/CoinExchange/blob/master/DEVELOP.md

Manage background screenshots：https://gitee.com/cexchange/CoinExchange/tree/master/09_DOC/管理后台截图

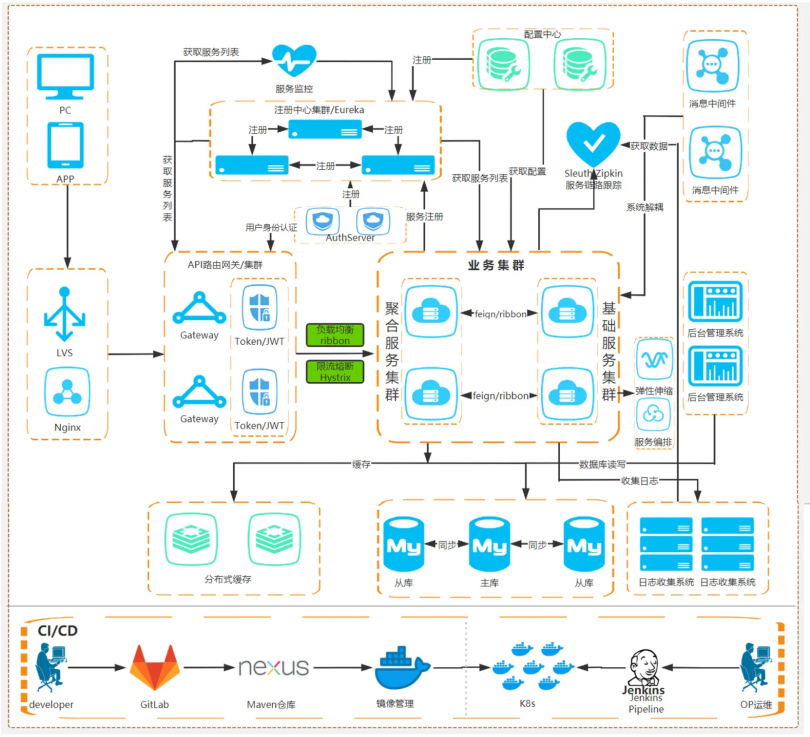
About server configuration and deployment

If you want to build an exchange system on your computer or cloud server, I have prepared some basic deployment manuals. Of course, installing software on linux/unix is not a simple matter. You need to have a certain Linux basics and command line skills, but also the courage and patience to solve problems, I wish you success!

- Server Configuration Reference Manual
- Installation Basic Environment Manual
- Service deployment script
- Install MySql Manual
- Install Redis Manual
- Install Zookeeper Manual
- Install Kafka Manual
- nstall Mongodb manual
- Building a BTC wallet node manual
- Building ETH Wallet Node Manual
- Manual of Building USDT Wallet Node

About SpringCloud

Spring Cloud is an ordered collection of frameworks. It uses Spring Boot's development convenience to subtly simplify the development of distributed system infrastructure, such as service discovery registration, configuration center, message bus, load balancing, circuit breaker, data monitoring, etc., can be done using Spring Boot's development style One click to start and deploy. Spring Cloud does not repeat the manufacturing of wheels. It just combines the mature and practical service frameworks developed by various companies. The re-encapsulation through the Spring Boot style shields the complex configuration and implementation principles, and finally gives the development The author has left a set of distributed system development kits that are simple to understand, easy to deploy, and easy to maintain. In general, a complete Spring Cloud framework should be as shown in the following figure:



If you are unfamiliar with Spring Cloud, you can simply learn about Spring Cloud related tutorials first, so that you will come back to this project and it will be easier to get started. As a reminder, because the Springcloud framework diagram is a complete architecture, we will tailor some content appropriately during development to make development and deployment faster, so there are some discrepancies.

About Matchmaking Trading Engine

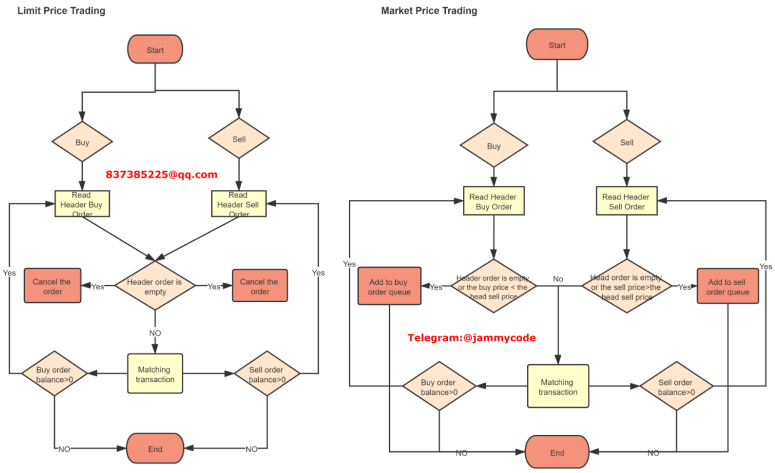
The system uses memory matching for the transaction queue, Kafka is used for matching order information transmission, MongoDB persists the order transaction details, and MySQL records the overall order transaction. Among them, the 01_Framework/Exchange project is mainly responsible for memory matching, and the 01_Framework/Market project is mainly responsible for order transaction persistence, market generation, market push and other services, including:

- K-line data, the intervals are: 1 minute, 5 minutes, 15 minutes, 30 minutes, 1 hour, 1 day, 1 week, 1 month
- Market depth data for all trading pairs
- The latest prices of all trading pairs
- Recently traded pairs

Modes supported by memory matching transactions

- Matching of limit order and limit order
- Matching market orders and limit orders
- Matching limit orders with market orders
- Matching market orders with market orders

Limit & Market Order Processing Logic



Note: This picture is a long time ago, the logic in the latest code is more complicated

Other features supported by match engine

In addition to the ordinary matching functions of limit and market prices, the matching trading engine of this system also introduces an active transaction mode. By setting the trading start time, initial issuance volume, initial issuance price, and activity of trading pairs (such as BTC/USDT) Modes and other parameters can formulate a wealth of matching transaction modes to meet different matching modes.

for example

The exchange is expected to launch the trading pair AAA/USDT at 12:00:00 on August 8, 2020, but as a newly launched currency, how can it work without activity? The project party or the exchange decided to come up with 10,000 AAA at a price of 0.0001USDT (market price: 0.0005) for everyone to snap up. The system supports the setting of such activities.

In addition, if the project party or the exchange decides to take out 10,000 AAAs to issue at the price of 0.0001USDT, I don't want everyone to snap up, but hope that all users who recharge USDT can divide 10,000 AAAs on average. This system also supports the setting of this activity .

to sum up

n short, this system supports a highly customized matching mode. At the same time, you can also develop your own matching transaction mode, just by modifying the matching logic in the Exchange project.

About the technical composition

- Backend development：Spring、SpringMVC、SpringData、SpringCloud、SpringBoot
- DataBase：Mysql、Mongodb
- Other：redis、kafka、Ali OSS、Tencent Captcha
- Frontend：Vue、iView、less

Demo website

https://www.bizzan.com

This was done for the customer, but later the customer stopped operating, so this website was left, because I don't have server permissions, so this website may not be accessible at any time.

Building a test site requires purchasing several cloud servers, which cost a lot, so I did not set up a test station myself, but the system is complete and has passed the commercial and practical operation test for nearly a year.

> About trading robots
> Trading robots are automatic trading programs that can automatically trade based on external market conditions, so that the exchange's trading pair prices are consistent with the external, preventing losses caused by some users "moving bricks".

==========

System operating environment

Demo website

https://www.bizzan.com

This was done for the customer, but later the customer stopped operating, so this website was left, because I don't have server permissions, so this website may not be accessible at any time.

Building a test site requires purchasing several cloud servers, which cost a lot, so I did not set up a test station myself, but the system is complete and has passed the commercial and practical operation test for nearly a year.

> About trading robots
> Trading robots are automatic trading programs that can automatically trade based on external market

> Trading robots are automatic trading programs that can automatically trade based on external market conditions, so that the exchange's trading pair prices are consistent with the external, preventing losses caused by some users "moving bricks".

==========

System operating environment

1. Centos 6.8
2. MySQL 5.5.16
3. Redis-x64-3.2.100
4. Mongodb 3.6.13
5. kafka_2.11-2.2.1
6. nginx-1.16.0
7. JRE 8u241
8. JDK 1.8
9. Vue
10. Zookeeper

Recommended configuration for production environment

SERVER	Recommended configuration	Running services	Remarks
Server 1	8 Cores 16GB RAM	Exchange	The service only occupies memory. It is the only log that consumes memory
	100G HDD	Market	
		Cloud	
		Exchange-Api	
		UCenter	
		Chat	
		Wallet	
Server 2	8Cores 16GB RAM	USDT, BTC	The nodes occupy hard disk space
	1T+HDD	ETH Node	
		Other Node	
Server 3	8Cores 16GB RAM	Wallet RPC-BTC	The service occupies memory. The only thing that consumes hard disk space is the log
	40-60GB HDD	Wallet RPC-ETH	
		Wallet RPC-Other	
Server 4	4Cores 8GB RAM	Nginx Reverse proxy	Only as a web static file access portal
	60G HDD	Web Front end agent	The domain name needs to be bound to port 80 of this

Other Cloud Service
MySQL Cloud Server
Kafka Message Queue
Redis cache database
SMS Cloud
Mail server
aliyun OSS

File directory description

00_framework

└———admin Background management API

└———bitrade-job Task management（Empty）

└———chat OTC Chat

└———cloud SpringCloud Eureka

└———core Core

└———exchange match trading

└———exchange-api order API

└———exchange-core order core

└———jar Other lib

└———market market API、K-line service

└———otc-api OTC trade API

└———otc-core OTC core

└———sql SQL script

└———ucenter-api user API

└———wallet wallet

01_wallet_rpc

└———bitcoin

└———bsv

└———btm

└———eos

└———erc-eusdt

└——erc-token（可对接各种ERC20币种）

└——eth

└——ltc

└——usdt

02_App_Android

03_App_IOS

04_Web_Admin

05_Web_Front

Use tutorial

1. Prepare the mysql database and create a database named "xxxx"
2. Prepare redis cache database
3. Prepare kafka streaming environment (first configure to run zookper, then configure to run kafka)
4. Prepare mongodb database environment, create users admin, xxxx, create bitrade database
5. Prepare Alibaba Cloud OSS (modify the place to be configured in the project)
6. Prepare nginx and modify the configuration file (optional, need to be configured for official launch)
7. Modify the configuration file in the framework code to prepare the environment configuration parameters
8. Compile to generate jar executable file
9. Run cloud.jar (microservices registration center)
10. Run exchange.jar (match trading engine)
11. Run market.jar (Quote Center, need to wait for Exchange.jar to fully start)
12. Run ucenter.jar (User Center)
13. Run other modules (wallet.jar, chat.jar, otc-api.jar, etc.)
14. Open mysql and import the xxxxxxx.sql file in the sql folder in the framework code. Note that if the trigger sql reports an error, you need to add a trigger for the wallet table
15. Run the front-end vue project
16. Run the back-end vue project
17. Run wallet RPC
18. Run the automated trading robot program (the code in this part is not uploaded, but it does not affect)
19. Run the Admin project (the service does not depend on other services, so you can just run this project and directly view the background)

Technical Support

This digital currency trading system is a project developed by my company for an exchange. The exchange has ceased operations due to team reasons, and our company was disbanded in February. Since I participated in the project, I was responsible for overall R&D management, architecture design and customer docking, so I mastered all the codes.

There are some places that need special attention in the use of the function of this system, such as other operations after the new transaction pair, improper operation will cause data disorder errors.

I can provide paid technical assistance and use training guidance!

Email：877070886@qq.com

Precautions

When the memory is insufficient, enter top in the Linux console to view the java process occupies a lot of memory (a java process occupies more than 1G), because there are many jar packages to run, so you need to control the memory used by certain jar packages, you can choose A few less resource intensive projects are as follows:

```
java -jar -Xms128m -Xmx128m -Xmn200m -Xss256k  admin-api.jar

java -jar -Xms512m -Xmx512m -Xmn200m -Xss256k  cloud.jar

java -jar -Xms512m -Xmx512m -Xmn200m -Xss256k  wallet.jar
```

About Mail & SMS

1. This system supports the operation status of the system for sending emails and short messages
2. System notification/alarm support: user registration, user authentication, user recharge/withdrawal, currency RPC operation status, system resource usage monitoring, etc. 24 kinds of monitoring

Questions about database scripts

Some reported that there is no complete SQL file. This is because the successfully compiled Jar will automatically map the Entity to a database structure after the first run. The SQL in the project only completes some database structures that Springcloud cannot complete. The automatic database configuration is located in the application.properties configuration file:

```
#jpa
spring.jpa.show-sql=true
spring.data.jpa.repositories.enabled=true
spring.jpa.hibernate.ddl-auto=update
```

spring.jpa.hibernate.ddl-auto=update This configuration will automatically update the database structure

spring.jpa.hibernate.ddl-auto=update This configuration will automatically update the database structure.

Core function description (user side)

- [x] 1. Registration/login/real-name authentication/audit (currently only supports mobile phones, secondary development can be added to the mail, very simple)
- [x] 2. Banner/Announcement/Help/Customized page (Banner supports separate settings for PC and APP, helps support various classification modes)
- [x] 3. Fiat currency C2C transaction/fiat currency OTC transaction (supports two fiat currency models, the platform can undertake C2C fiat currency exchange in the early stage of the project, and OTC transactions can be opened later)
- [x] 4. Coin trading (support for limit order, market order, and other commission modes can be added for secondary development)
- [x] 5. Invite registered/promotion partners (support for ranking statistics on the number of invited promoters and commissions by day, week and month)
- [x] 6. Innovation Lab (there are many supported functions in this part, and itemized explanations. In addition, the APP does not support this function for the time being)

> - [x] 6-1. First launch panic buying activity mode (for example, when issuing a new trading pair, a certain number of currencies can be set on the trading pair for panic buying)
> - [x] 6-2. The first distribution mode of activity (for example, before issuing the BTC/USDT trading pair, the official took out 5BTC for the activity, and divided the BTC equally according to how much USDT the user recharged mortgaged)
> - [x] 6-3. Panic buying mode (for example, before the ZZZ/USDT trading pair is issued, the ZZZ currency price is 5USDT, and the official issuance activity price is 0.5USDT, you can use this mode)
> - [x] 6-4. Control panel sharing mode (such as 6-3, only average distribution)
> - [x] 6-5. Mining machine activity mode (support users to mortgage a certain amount of currency, and the official promise to return a certain amount of currency every month)

- [x] 7. Red envelope function (support platform and official to issue a certain amount of currency red envelope, this function is suitable for user fission)
- [x] 8. Various basic management such as user asset management, flow management, commission management, real name management

Core function description (management side)

- [x] 1. Summary (view platform operating data, including transaction amount, registered number, recharge, etc.)
- [x] 2. Member management (member information management, member real name verification, member real name management, member balance management, member recharge/freeze balance, etc.)
- [x] 3. Invitation management (member invitation information, member invitation ranking management)
- [x] 4. CTC management (CTC order management, flow management, acceptor management)
- [x] 5. Content management (PC advertising management, APP advertising management, announcement management, help management)
- [x] 6. Financial management (charge and withdrawal management, financial flow management, reconciliation management, currency wallet balance management)
- [x] 7. Currency management (new trading pair, management trading pair, new trading robot, setting trading robot parameters, setting market engine/trading engine, revoking all orders)
- [x] 8. Event management (new activity, mining machine subscription, panic buying/division management)
- [x] 9. Red envelope management (platform red envelope management, user red envelope management)
- [x] 10. System management (role management, department management, user management, authority management, currency management, RPC management, version management)
- [x] 11. Margin management (this function is considered in design, but not used during actual operation)
- [x] 12. OTC management (advertising management, order management, OTC currency management, surrender management, etc., this function has not been verified by actual operation)

About blockchain wallet RPC

This project provides two wallet docking methods, one is self-built node + blockchain browser, and the other is third-party wallet docking. If you want to use a self-built node or blockchain browser, you can directly compile the code in 00_framework. If you want to use a third-party wallet for docking, you can download the project files of the YouDun wallet in the 07_Uduncloud folder and copy them to 00_framework.

After you get the code, you can not connect the blockchain nodes during the debugging and running of this project, which will not have much impact; even if you do not connect the blockchain nodes, you can deploy one of them with matching transaction function Trading platform (just that users cannot recharge their wallet addresses).

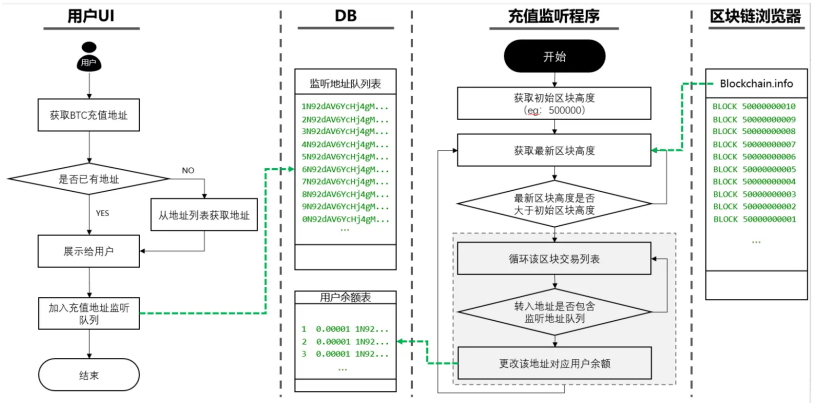
When you are gradually familiar with the entire system, and have a certain basic reserve of blockchain operation principles, node construction, and blockchain browser, you can start to study the projects under the **01_wallet_rpc** folder. Each currency corresponds to different data access methods. Most blockchain projects have the same or very similar wallet operation methods. For example, Bitcoin derivatives such as BTC, LTC, BCH, BSV, BCD, etc., have almost the same API operations. The same; another example is ETH, when you master the operation of a contract currency, the operation of other digital currencies based on ETH is almost the same. So, basically when you spend time understanding one, you understand a bunch of currencies.

The wallet operation scheme used in this project is also different, and as far as possible to show you different usages:

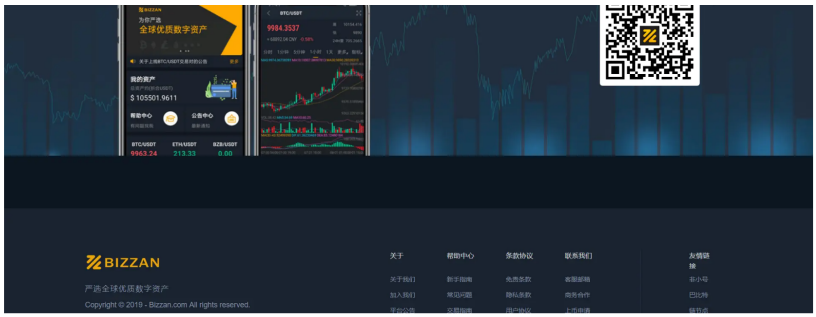
- Self-built full nodes such as BTC and USDT now require almost 300G of hard disk space;
- Like ETH, self-built light nodes are used (Reference Article), because the full node requires too much hard disk space;
- Such as BCH, BSV, etc., use a third-party blockchain browser to obtain data;
- Like XRP, the official has provided an interface to access block data ([Ripple API GitHub address](https://github.com/ripple/ripple-lib/))

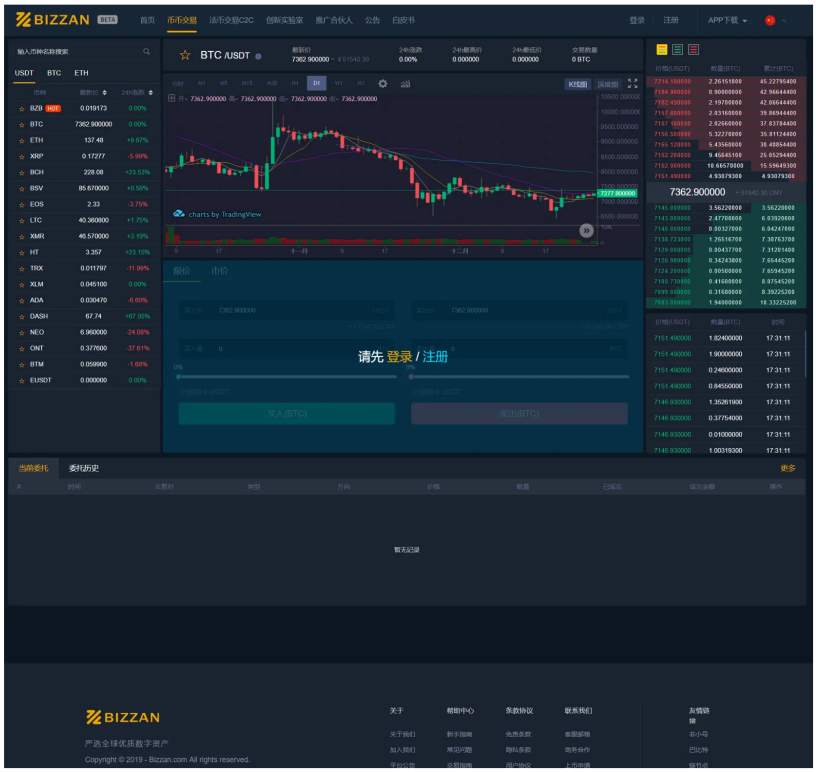
Generally speaking, when the amount of funds exchanged by the exchange is not large, you can explore it yourself, but when the amount of funds exchanged is large, if you are not confident about operating your wallet, you can also use a third-party wallet service Of course, this requires you to negotiate with the wallet service provider to pay an annual fee or something.

The following figure is a brief explanatory diagram of the user recharge monitoring logic, just take a look at it:



System display (PC front end)


全球优质数字资产
$ 105501.9611
BIZZAN
严选全球优质数字资产
Copyright © 2019 - Bizzan.com All rights reserved.


BIZZAN
首页
币币交易
法币交易C2C
创新实验室
推广合伙人
公告
白皮书
登录
注册
APP下载
BTC /USDT
请先 登录 / 注册
当前委托
委托历史
暂无记录
BIZZAN
严选全球优质数字资产
Copyright © 2019 - Bizzan.com All rights reserved.


BIZZAN
首页
币币交易
法币交易C2C
创新实验室
推广合伙人
公告
白皮书
登录
注册
APP下载
一键买卖 ● 平台托管 ● 安全放心
由币严平台托管数字资产，保障用户资产安全
USDT交易
买入价
7
CNY
卖出价
6.93
CNY
温馨提示