Solution to Pepper Supply Chain Finance Based on Blockchain Technology

team members:
221025144 Yuzhen Jiang
221041004 Chang Liu
221041035 Xinran Zhang
221041040 Huanli Wu
221041041 Yining Zhong

The Chinese University of Hong Kong, Shenzhen
Longgang, Shenzhen

Abstract

Our report mainly includes two parts: design of supply-chain financing based on Blockchain technology and deployment of blockchain application based on hyperledger fabric extension in VScode.

In the part of the blockchain system design, according to the business model, we drew the UML use case diagram, which discusses the permissions of different entities; ER diagram, which reflects the information association of different entities and ledgers; Data Flow Diagram, which shows the data flow in the blockchain; Information Flow Diagram, which reflects the whole process information from loan application to repayment. A clear understanding of the business model and the design of the blockchain system lay a solid foundation for the implementation of our blockchain system.

In the part of the deployment of our blockchain application, we chose hyperledger fabric frame first, drew the Blockchain Network Diagram, which shows the structure of the blockchain systems; We used java programming languages to define the participants, assets, and transactions, from which we basically realized our business proposal. And we put the operational documentation in the appendix to better show you details of how to use our applications.

Contents

1 Sponsor of the Project

The pepper planting area of Tuocheng county is stable at 400,000 mu all year round. The annual trading volume of dried pepper reaches 700,000 tons. The trading volume of pepper market is more than 10 billion yuan, which is the largest pepper trading market in China. The Red Cube Industrial Internet platform is an industrial Internet demonstration project of " Rural revitalization: 100 counties producing 100 products" planned and promoted by the Industrial Internet Special Committee of China Association of Small and Medium Enterprises. Through smart cloud warehouse management, it creates a pepper industrial ecosystem integrating research, production, supply, marketing and trade.

At the same time, Red Cube Is also our instructor. We have learned from its business framework to build a business model of inventory pledge in the pepper industry chain, and plan to use blockchain to empower the industry.

2 Background and Introduction

2.1 Industry Description

Supply Chain Finance (SCF) is a professional field of credit business of commercial banks (bank level), and also a financing channel for enterprises, especially smes (enterprise level).

Specifically, it means that the bank provides financing and other settlement and financial management services to the customers (core enterprises), and at the same time provides convenience for the timely collection of loans to the suppliers of these customers, or provides advance payment and inventory financing services to their distributors. (To put it simply, it is a financing model in which banks connect core enterprises with upstream and downstream enterprises to provide flexible financial products and services.)

In the traditional sense, inventory pledge refers to the movable property pledge business with the participation of logistics enterprises, in which enterprises in need of financing take their own inventory as pledges, pledge it to the capital provider enterprises, and transfer the pledges to the logistics enterprises with the legal inventory qualification for storage, so as to obtain the credit loan.

In terms of our project, we focused on the inventory pledge mode in supply chain finance.

2.2 Industry Pain Points and Challenges

Inventory Financing involves a lot of participants such as pledgors, warehouses, financiers, inspection services firms, etc. This complex network requires a centralized Supply Chain Finance platform governed by a trusted intermediary to manage collaboration, communication, information exchanges and financial needs between participants. However, such centralized systems often face challenges including dependency, transparency, cost and time efficiency. Therefore, the goal of our project is to build a system using Blockchain technology to address these challenges, while keeping intact the benefits of Supply Chain Finance. The pain points of traditional Inventory Financing platforms are as follows:

(1) Lack of Effective Risk Control Measures
Financial institutions cannot fully trust the authenticity and validity of enterprise information on the supply chain finance platform, and there may be fraud or malicious manipulation including false warehouse receipts, repeated pledges and etc.

(2) Approval Issues
In order to verify the enterprises background, financial institutions will invest heavily in manpower and resources to verify the authenticity of the information flow, business flow, cash flow and logistics in a multi-dimensional way, which reduces the businss efficiency of supply chain finance.

(3) Data Validity Issues
Supply Chain Finance platforms usually need to rely on third-party authorities such as notary offices to witness and improve the authority of the data. But it will inevitably increase transaction costs, affect efficiency.

(4) System Vulnerability
Normally, the system applications, transaction data and account data of supply chain finance platforms are stored centrally and maintained independently by the platform itself. This centralized storage mode leads to potential vulnerabilites. The entire platform may be paralyzed and data may be lost if attacked, which affects the continuity and reliability of system services.

(5) Data-sharing Issues
The data standards among participants are not unified, and the realization of data interaction requires each participant to carry out system transformation. Some core enterprises are unwilling to open the ERP system for the sake of system security.

3 Design of Blockchain Application

3.1 Usecase

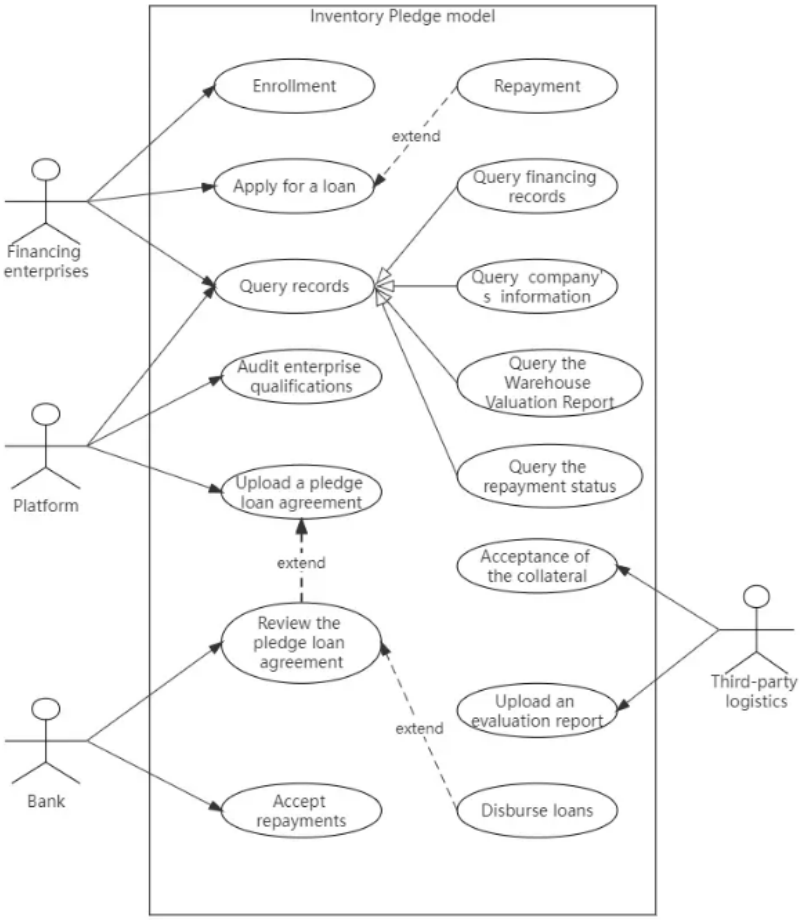


Figure 1: Use Case diagram

- **Enrollment:** Enterprises can enroll and create a record, which will contain enterprise credit and other information.
- **Query records:** Enterprises can query their financing records, company' s information, the Warehouse Valuation report and whether they repay or not. The platform can also check this information.
- **Audit enterprise qualificaiton:** The platform judges the enterprise' s credit and relevant qualifications to determine whether or not to borrow.
- **Review the pledge loan agreement:** The bank reviews the agreement then disburse loans.
- **Acceptance of the collateral:** The warehouse evaluates the goods from the financing enterprises and the issue reports and provide supervision services.

3.2 Entity-Relationship Diagram

3.2.1 ER model

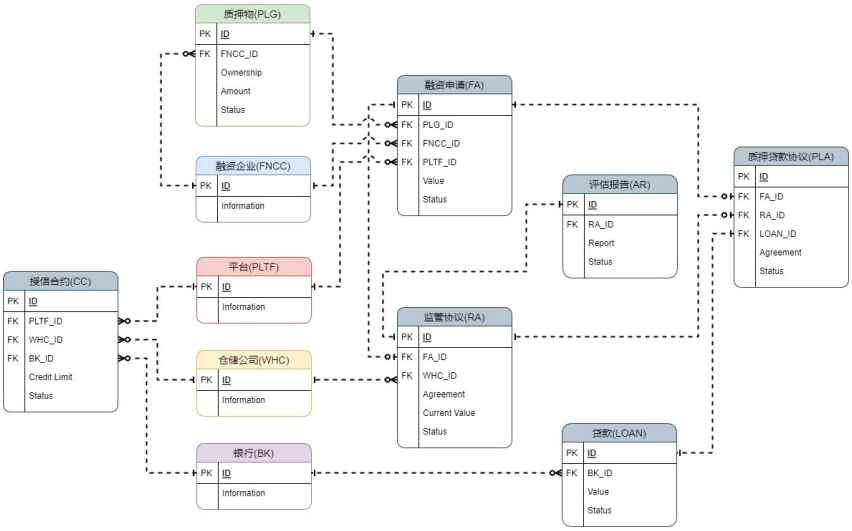


Figure 2: Use Case diagram

As we can see from the figure above, we designed 11 bodies throughout the process. There are four users, and seven intermediate principals.

There are 4 main accounts, including pepper enterprises, platform, warehouse and banks. At the same time, the pledge used for financing is abstracted into an entity, with ownership, quantity, state and other attributes.

In addition, there are other entities in the ER diagram that correspond to things that are produced in the process.

Firstly, The platform, the storage company and the bank will sign the credit extension contract(CC), including the credit limit, contract status and other attributes.

Secondly, the financing enterprise initiates the financing application (FA), which contains the application amount, application status and other attributes.

After the application is approved, the platform and the financing enterprise sign the Pledge Supervision Agreement (RA) with the storage company according to the application, which contains the contents of the pledge agreement, the current value of the pledge, the status of the agreement and other attributes.

When the storage company receives the pledge for the first time, it will evaluate the real situation of the goods and submit the appraisal report(AR) to the platform. The appraisal report entity includes the report content, report status and other attributes.

After reviewing and passing the appraisal report, the platform will formally sign the pledge loan agreement PLA to issue loans to financing enterprises. The agreement entity includes the agreement content, agreement status and other attributes,.

Finally, we added "loan entity", including the lending bank, loan status and other attributes, for real-time understanding of the loan status.

In the following project, we mainly focus on the 4 main entities–enterprises, platform, warehouse and banks.

3.2.2 Relational Schema

The relations between each entity are as followed.

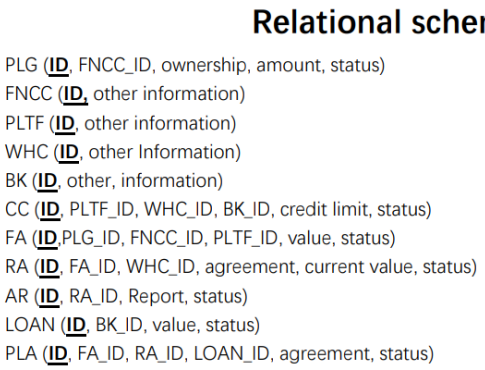


Figure 3: Use Case diagram

The primary key of each principal is its own ID. Pledges and various other protocols and reports have the primary key of another entities as keys.

Now, here are the ER relationships between different entities:

- one FNCC may have zero or more PLG; one PLG just belong to one FNCC.
- one FNCC/PLG/PLTF may have zero or more FA; one FA only corresponds to one FNCC/PLG/PLTF
- one PLTF/WHC/BK has zero or more CC; one CC only to one PLTF/WHC/BK
- one WHC may have zero or more RA; one RA just correspond to one WHC
- one FA may have 0 or one RA; one RA just correspond to one FA
- RA and AR correspond one by one
- one BK may have 0 or more LOAN; one LOAN only comes from one BK
- one FA/RA may correspond to 0 or one PLA; one PLA just correspond to one FA/RA
- LOAN and PLA correspond one by one

3.3 Data Flow Diagram

Figure 4: 0 Diagram

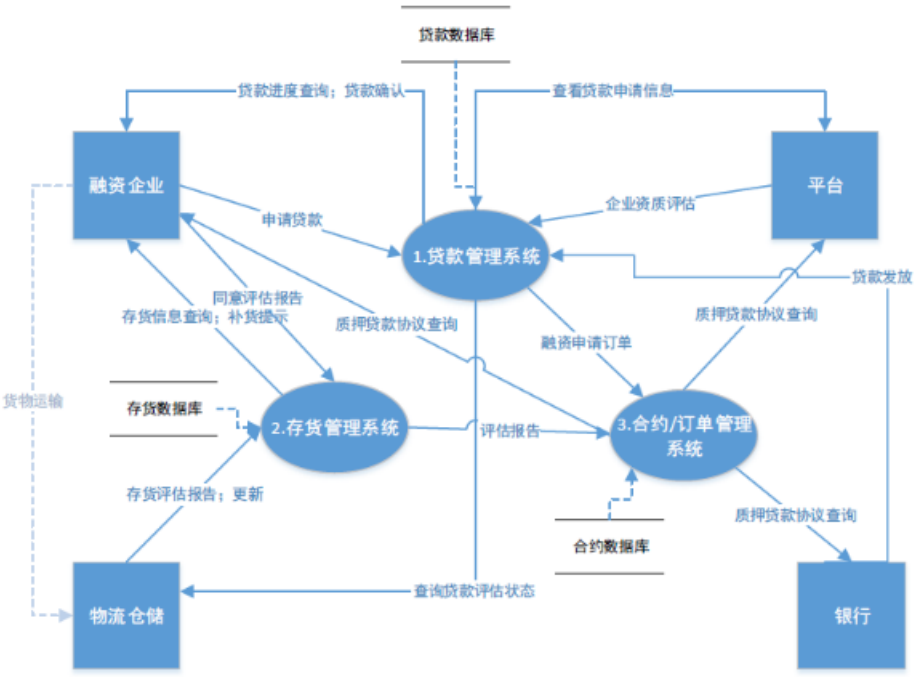


Figure 5: 1 Diagram

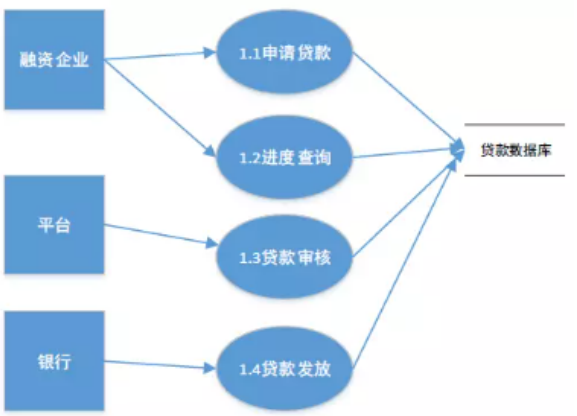


Figure 6: 2 Diagram - 1

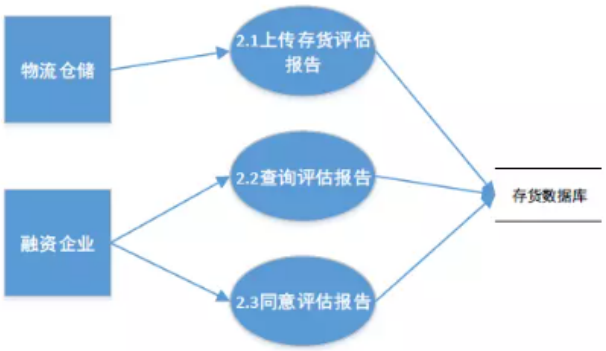


Figure 7: 2 Diagram - 2

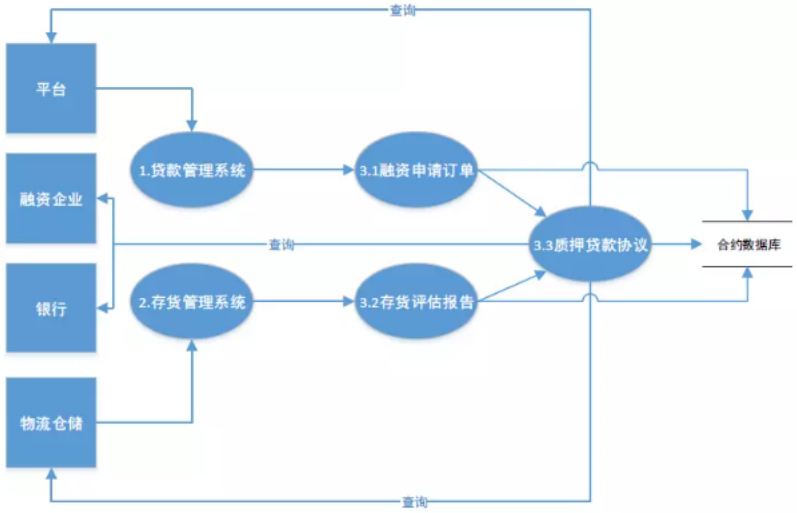


Figure 8: 2 Diagram - 3

- There are four users in our DFD (Data Flow Diagram). They are financing enterprises, platforms, storage companies, and banks. In the 0 diagram, we show the data flow between the four users and the whole blockchain system.

- In the 1 diagram, we broke the whole system into threes subsystems: loan managing system, inventory managing system, and order/contract managing system. It is worth noting that the data flow between the three systems is achieved by the consensus algorithm and smart contract. And Integrating banks directly into the system makes sure the loan process is transparent.

- Decomposing the three subsystems again, we get 2 Diagram. It contains some operations that entities will do in the three systems, which is similar to the UML use case diagram. These operations will be all recorded into the corresponding database.

3.4 Information Flow Diagram

In this part, for the requirement of following project task, we split the whole data flow process into two parts – Platform Credit Application and Inventory Financing Information Flow Diagram.

In the part1, we find that only 3 entities are included – except pepper enterprises. In the part22, the pepper Enterprises apply for loans from the platform; The platform and warehouse confirm the enterprise credit and pledge states, and release the amount.

Because pepper enterprises are not included in step 1, we choose step 2 to start our lifecycle.

3.4.1 Platform Credit Application

Small pepper enterprises always have high credit risk in the banking system, and it is difficult to directly finance from the bank. Therefore, as the intermediary of the whole process, the platform needs to apply for credit from the bank firstly and then confirm the credit limit of the platform's loan.

The process is shown in the fig 4. In this part, platform apply for credit from the bank.

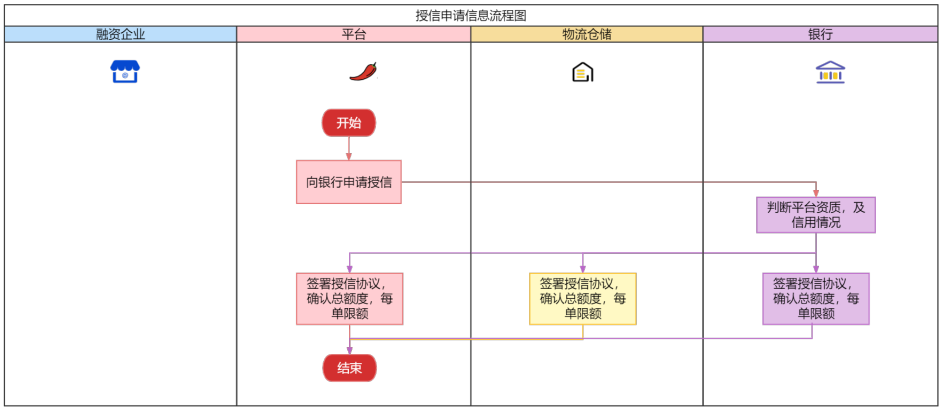


Figure 9: Platform Credit Application Information flow Diagram

3.4.2 Inventory Financing Information Flow Diagram

Now, let's focus on the second step–how the enterprises apply for the loan. The process is shown in fig5.

Here are the specific steps.

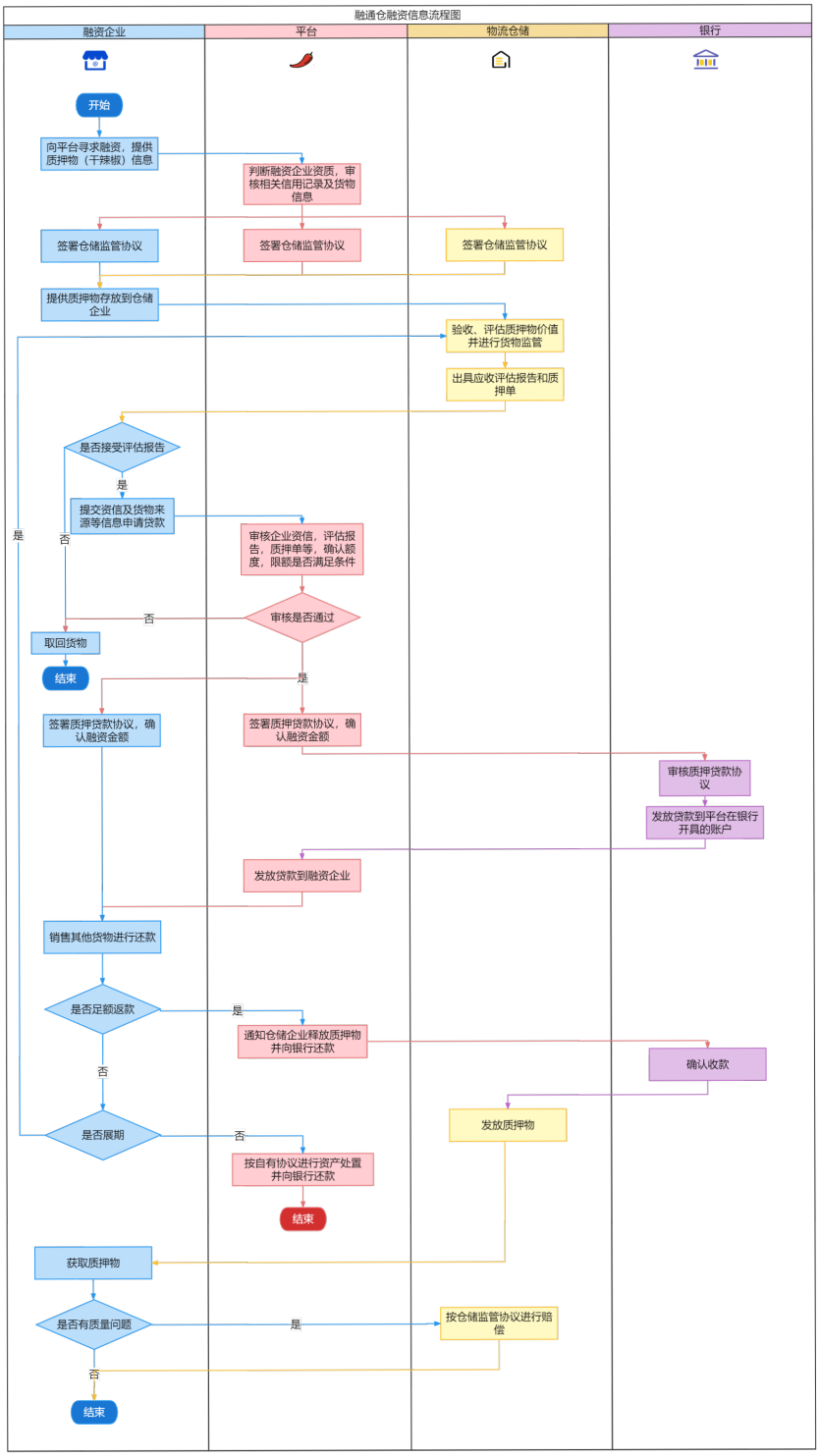


Figure 10: FTW Information Flow Diagram

- When pepper enterprises have financing needs, they submit applications and provide pledge information in the system.
- The platform preliminarily judges the enterprise's credit and relevant qualifications, and signs a warehousing supervision agreement with the financing enterprise and the warehouse.

- The pepper enterprise shall deliver the pledges to the warehouse according to the agreement. The warehouse shall evaluate, issue reports and pledge documents, and provide goods supervision services.
- If the evaluation report is approved by the pepper enterprise, it shall be submitted to the platform for formal examination and decision on whether to lend.
- If approved, the pepper enterprise shall sign the inventory financing agreement with the platform, and the bank will review the agreement and remit the money to the platform, and then transferred to the enterprise.
- Finally, the enterprise sells other goods for repayment. If the repayment is successful, the platform will return the pledge, and if the repayment fails, the platform will consider whether to postpone the financing agreement or dispose of the goods for repayment.

4 Business Process and Life cycle

Based on the above Information Flow Diagram, we developed the life cycle which is shown in fig 6.

The entire life cycle has two components: the States and Transactions. The transitions of states are achieved through transactions between different actors in the life cycle. In the part, We use rounded boxes to represent states and use arrows to represent transactions.

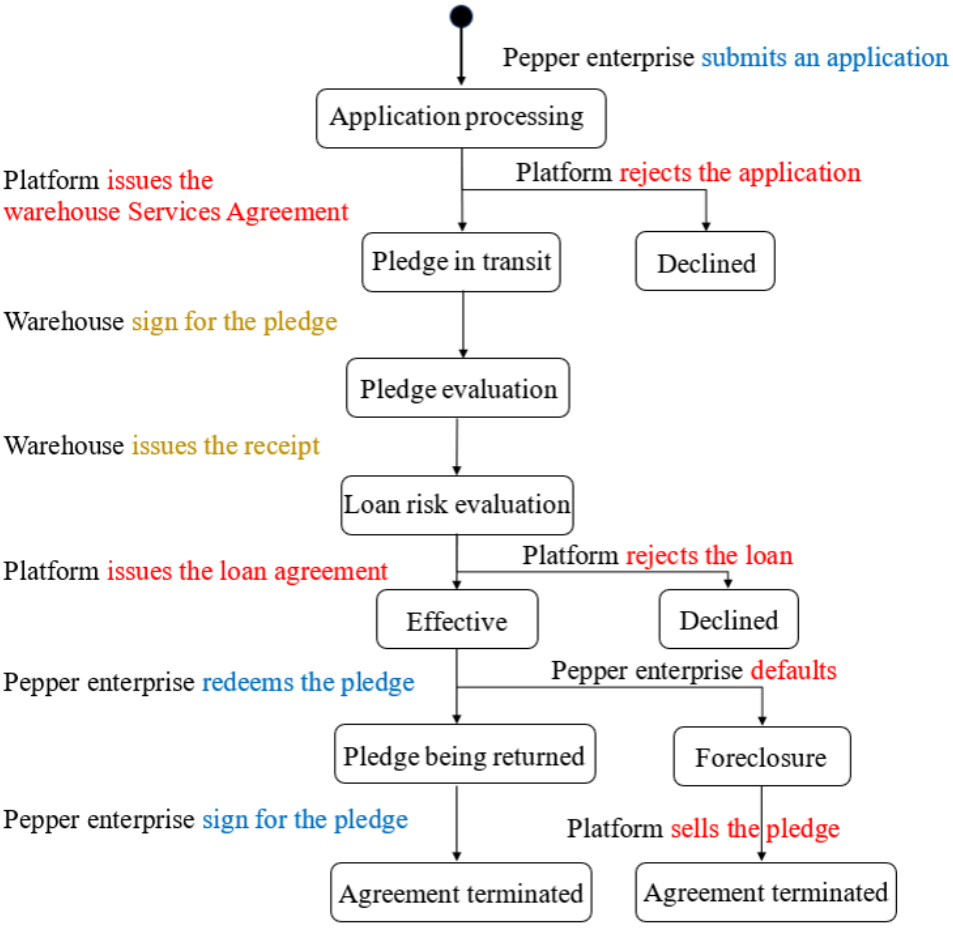


Figure 11: life cycle

We can see from the picture that, there are mainly 7 transactions and they achieve different states. During each transaction, there are some actions of each subject. Also, based on the application process, we can split the whole life cycle into 2 part2. The first part includes the application process and the second part includes the process after the agreement of loan.

4.1 application process

Firstly, let's talk about the application process. In this part, the process from the a enterprise's application for a loan to its final approval will be included.

- Firstly, the pepper enterprises submit their application and the state changes to 'application processing' .
- Secondly, The platform will conduct a preliminary review of the application. If the application is approved, the storage supervision agreement will be issued and endorsed by the pepper enterprise and the storage company ——the state is called 'pledge in transit' . If the application is rejected, the application state changes to declined.
- Thirdly, after the pledge arrives at the warehouse, the storage company will sign for the receipt and the process enters the 'pledge evaluation' stage.
- Fourthly, after the evaluation, the warehouse issues a pledge sheet, and the platform evaluates the pepper enterprise's qualification and risk according to the pledge sheet and makes decisions whether we should lend money to this enterprise.
- Fifthly, if the evaluation result of the platform passes, the final loan agreement will be issued, indicating that the three parties have reached an agreement on the loan amount, repayment date and other aspects, and the agreement will enter into force. If the evaluation result is not passed, the application state changes to declined.

4.2 after the agreement of loan

The above is the application process. Then we focus on the process after the agreement of loan.

4.2.1 normal process

The normal process is: While the agreement is in force, the pepper enterprise repayments by selling other goods. If the payment is made successfully before the repayment date, the pledge will be returned to the pepper enterprise, and after the enterprise signs and receives the pledge, the agreement terminate.

There are 2 states in the normal process.

- The pepper enterprise repayments by selling other goods. If the payment is made successfully before the repayment date, the pledge will be returned to the pepper enterprise. And the state changes to 'pledge being returned'.
- After the enterprise signs and receives the pledge, the agreement terminate and the state changes to 'Agreement terminated'.

4.2.2 special case

But in some special cases, the pepper enterprise cannot pay the loan, and it will lose the ownership of the pledge. The platform sells the pledge and pays the bank back. Then the agreement terminate.

- The pepper enterprise cannot pay the loan, the enterprise loses the ownership of the pledge. The state changes to 'Foreclosure'.
- The platform gets the ownership of pledge and sells the pledge to pay the bank back. Then the agreement terminate and state changes to 'Agreement terminated'.

5 Ledger State and Structure

In the above life cycle, we can abstract the 2 parts. The process in first part can be abstracted as ‘Through some transactions, the enterprise application status changes from 'Application Processing' to 'Effective' . And the second part can be abstracted as process from 'Effective' to 'Agreement terminated'. We will mainly focus on the first part and give 2 samples of Ledger State and Structure as shown in fig 12.

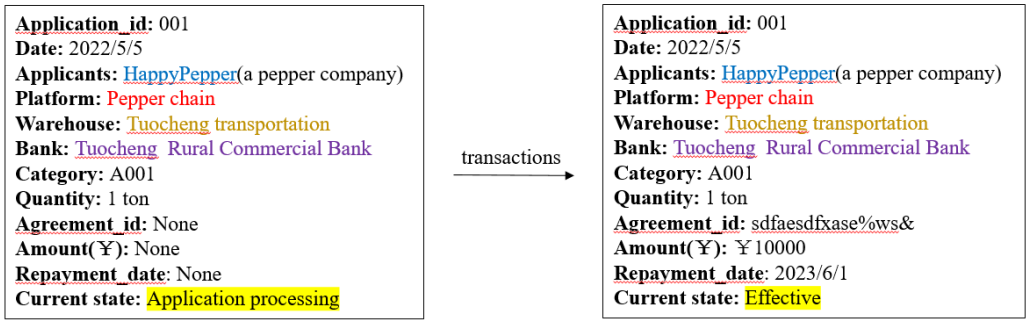


Figure 12: Ledger State and Structure

The structure includes some attributes :

- Application_id
- Date
- Applicants: the pepper company who applies for the loan
- Platform:
- Warehouse:
- Bank:
- Category:
- Quantity: the quantity of the pledge
- Agreement_id
- Amount($): the amount of the loan. When the application has not been passed, the amount is 'None'. When the application has been passed and the limit has been confirmed, the amount is a finite number.
- Repayment_date: the agreed repayment date. When the application has not been passed, the date is 'None'. When the application has been passed, the date is an exact date.
- Current state

6 Implementation of Blockchain Application

6.1 Blockchain Network Diagram

In our blockchain network, there are four participants (R1: warehouse company, R2: platform, R3: pepper enterprise, R4: bank), a virtue system manager, a channel, peer nodes for each participant, a ledger, and a smart contract. Ordering service node O5 will give users R5 defined by a Certificate Authority CA5, set of rights over the resources in the whole network N as described by policies contained inside a network configuration NC5. R5 is a virtue system manager and will not participate in the transactions. Organization R5 updates the network configuration to make R2, the platform an administrator too. Then R2

and R5 have equal rights over the network configuration, like they can define a consortium to make transaction within them.

The platform defines a consortium X1 that contains four members based on our business needs, the organizations R1-R4, representing warehouse company, platform, pepper enterprise, and banks. This consortium definition is stored in the network configuration NC5. CA1 to CA4 are the respective Certificate Authorities for these organizations. Consortium X1 can create one channel C1 for their private transaction, this channel is governed by a channel configuration CC1, which is managed by the four participants who have equal rights over C1. All the four peer nodes P1-P4 have rights to read or write on this channel becasue the four different orgnizations take on different responsibilities to change the state of assets in the network.

Administrator of the platform will create a chaincode packages (containing smart contract) and install it onto peer node P2. It then must approve the chaincode definition for S1. After pepper enterprises, warehouse, and bank approve the definition of the chaincode, Chaincode can be use by the four organizations. Client applications invoke a smart contract by sending transaction proposals like query and update data to Peer nodes. It access the ledger via smart contract, to generate transactions that will be endorsed by four organization.

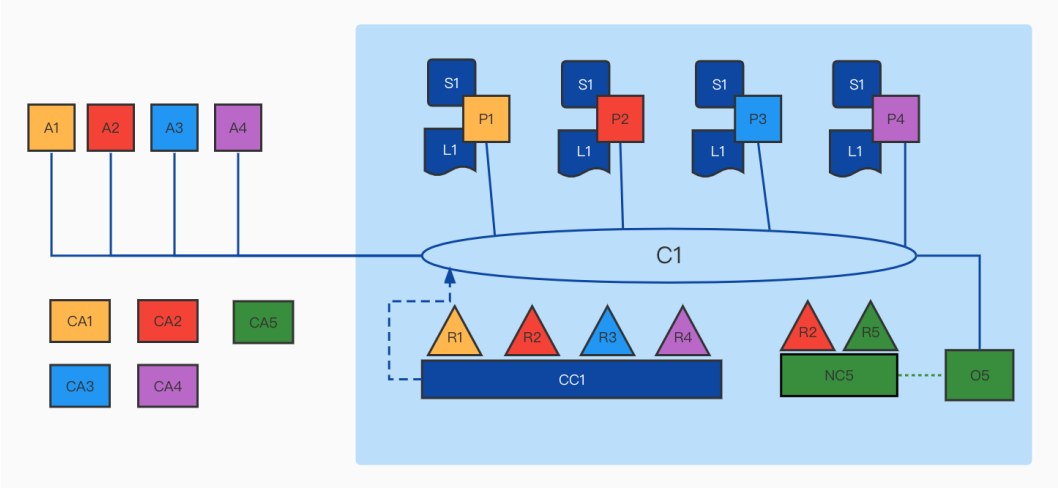


Figure 13: Code for Enterprise

6.2 Description on Bussiness Network

6.2.1 Hyperledger Fabric

Hyperledger fabric is an open source project from the Linux foundation. It is a modular blockchain framework and the actual standard adopted by the enterprise blockchain platform. As the basis for developing enterprise applications and industry solutions, open modular architecture uses plug and play components to meet the requirements of various use cases. In our project, we use IBM blockchain platform plug-in based on vscode to interact.

6.2.2 Participants File

There are four participants in our network, pepper enterprises, platform, warehouse, and banks. Each participant contains its basic information like ID, name, telephone number, and address to identify itself. For platform, we record more details about them like guaranteeAmount for platform to evaluate if summation the debt larger than loan amount the bank gives to it.

```
export class Enterprise {
    public enterpriseId: string;
    public enterpriseName: string;
    public mspId: string;

    constructor(enterpriseId: string, enterpriseName: string, mspId: string) {
        this.enterpriseId = enterpriseId;
        this.enterpriseName = enterpriseName;
        this.mspId = mspId;
    }
}
export class Platform {
    public platformId: string;
    public platformName: string;
    public mspId: string;
    public guarenteeAmount?: number;

    constructor(platformId: string, platformName: string, mspId: string) {
        this.platformId = platformId;
        this.platformName = platformName;
        this.mspId = mspId;
    }
}
export class Bank {
    public bankId: string;
    public bankName: string;
    public mspId: string;

    constructor(bankId: string, bankName: string, mspId: string) {
        this.bankId = bankId;
        this.bankName = bankName;
        this.mspId = mspId;
    }
}
export class Warehouse {
    public warehouseId: string;
    public warehouseName: string;
    public mspId: string;

    constructor(warehouseId: string, warehouseName: string, mspId: string) {
        this.warehouseId = warehouseId;
        this.warehouseName = warehouseName;
        this.mspId = mspId;
    }
}
```

Figure 14: Code for participant

6.2.3 Asset File

The key asset in our network is "loan application form" , which contains the information of pledge and loans. The required data includes applicaionID, CreateDate, CurrentState (imply the above several states: Application processing, Pledge in transit, Pledge evaluation, Decline, Loan risk evaluation, Effective, Pledge being returned, Foreclosure, Agreement terminated). CreaterEnterpriseID, PlatformID. The optional data contains warehouseID, bankID, application state, warehouseServiceAgreement, RiskEvaluation, loanAgreement, and so on. These information will be updated by different transcation defined later. Take applicatioState as an example, when the pepper industry use our system to apply for a loan, script file will call the transaction submitApplication to update the value of applicationState

from none to application processing. And the pepper enterprise can query this information immediately.

```
import { Object } from 'fabric-contract-api';

export const enum states {
    processing = "Application Processing",
    transit = "Pledge in Transit",
    ple_eval = "Pledge Evaluation",
    risk_eval = "Loan Risk Evaluation",
    effective = "Effective",
    returned = "Pledge being Returned",
    foreclosure = "Foreclosure",
    terminated = "Agreement Terminated",
    declined = "Declined"
}

@Object()
export class LoanApplication {

    public applicationId: string;
    public createDate: string;
    public currentState: states;
    public pledgeAmount: number;
    public creatorEnterpriseId: string;
    public platformId: string;
    public warehouseId?: string;
    public bankId?: string;

    public rejectedDate?: string;
    public issueWSADate?: string;
    public signDate?: string;
    public issueReceiptDate?: string;
    public issueLoanAgreementDate?: string;
    public redeemDate?: string;
    public signPledgeDate?: string;
    public defaultDate?: string;
    public sellPledgeDate?: string;
    public storageWarehouseId?: string;
    public lenderBankId?: string;
    public application_state?: string;
    public warehouseServiceAgreement?: string;
    public riskEvaluation?: string;
    public loanAgreement?: string;
    public queryDate?: string;
    public loanAmount?: number;
    public loanState?: string;
    public pledgePrice?: string;
    public sellPrice?: string;

    constructor(
        applicationId: string,
        creatorEnterpriseId: string,
        platformId: string,
        createDate: string,
        pledgeAmount: number
    ) {
        this.applicationId = applicationId;
        this.creatorEnterpriseId = creatorEnterpriseId;
        this.platformId = platformId;
        this.createDate = createDate;
        this.currentState = states.processing;
        this.pledgeAmount = pledgeAmount;
    }
}
```

Figure 15: Code for asset

6.2.4 Transaction Files

We define seven transaction based on our business needs, which are submitApplication, redeemPledge, signPledge, rejectApplication, issueWSAgreement, claimDefault, and issueReceipt. For different participants, they have different rights to access these functions.

Table 1: Different functions for different organizations

Entity	Function
enterprise	submitApplication, redeemPledge, signPledge, claimDefault, queryApplicationStateById, queryAllApplication
platform	initPlatformIdentity,rejectApplication,issueWSAgreement, issueLoanAgreement,sellPledge,queryAllApplication
bank	initBankIdentity,queryAllApplication
warehouse	initWarehouseIdentity,signPledge, issueReceipt,queryAllApplication

What should mentioned is that in the above 8 transactions, signPledge, rejectApplication are related to the changing of pledge ownership, between the enterprises and the platforms.

IssueWSAgreement will assign warehouse for the enterprise to transport and store. The price of pepper will be updated by warehouse using transaction priceChange, which will ensure the total value of the pledge will be not less than the amount of loan.

RedeemPledge and ClaimDefault are related with the loanState, when pepper enterprises redeem the pledge, they must pay the loan back to bank first, so the loanState will change

from loaning to repayment completed. DeclareDefault is called when the enterprises can not pay then loan back in time, so the loanState will change from loaning to repayment failed.

```
/*****************************************************************
 *
 * Methods for Enterprise
 *
 *****************************************************************/
    /**
     * 初始化融资企业信息
     */
    @Transaction(true)
    public async ETP_initEnterpriseIdentity(ctx: Context, enterpriseId: string, enterpriseName: string): Promise<string> {
        await this._checkIdentity(ctx, participantIdentity.enterprise);
        const mspId: string = await this._getMSPID(ctx);
        const enterprise: Enterprise = new Enterprise(enterpriseId, enterpriseName, mspId);
        await ctx.stub.putState(stateKeys.enterpriseKey, Buffer.from(JSON.stringify(enterprise)));
        return `Successfully initialize enterprise with id ${enterpriseId} and name ${enterpriseName}`;
    }

    @Transaction(true)
    public async ETP_submitApplication(
        ctx: Context,
        platformID: string,
        createDate: string,
        loanAmount: number,
        pledgeAmount: number
    ): Promise<string> {
        await this._checkIdentity(ctx, participantIdentity.enterprise);
        const enterprise: Enterprise = await this._getEnterprise(ctx);
        const appNum: number = await this._getAppNum(ctx);
        const applicationId: string = String(appNum + 1);
        const ifExist: boolean = await this._applicationExists(ctx, applicationId);
        if (ifExist) {
            throw new Error(`The application ${applicationId} already exists`);
        }
        const application = new LoanApplication(
            applicationId,
            enterprise.enterpriseId,
            platformID,
            createDate,
            pledgeAmount
        )
        application.loanAmount = loanAmount;
        await this._putStateFromObj(ctx, applicationId, application);
        console.info(`Successfully create application ${applicationId}`);
        await this._putStateFromStr(ctx, stateKeys.appNumKey, String(appNum + 1));
        return `Successfully create application ${applicationId}`;
    }
    @Transaction(true)
    public async ETP_redeemPledge(ctx: Context, applicationId: string, redeemDate: string): Promise<string> {
        await this._checkIdentity(ctx, participantIdentity.enterprise);
        const application: LoanApplication = await this._getApplicationById(ctx, applicationId);
        if (application.creatorEnterpriseId != (await this._getEnterprise(ctx)).enterpriseId) {
            throw new Error(`Enterprise Id dose not match.`);
        }
        if (application.currentState != states.effective) {
            throw new Error(`Can only submit this transaction with state ${states.effective}.`);
        }
        await this._updateApplication(application, states.returned);
        application.redeemDate = redeemDate;
        await this._putStateFromObj(ctx, applicationId, application);
        return `Successfully redeem pledge for application ${applicationId}`;
    }

    @Transaction(true)
    public async ETP_signPledge(ctx: Context, applicationId: string, signPledgeDate: string): Promise<string> {
        await this._checkIdentity(ctx, participantIdentity.enterprise);
        const application: LoanApplication = await this._getApplicationById(ctx, applicationId);
        if (application.creatorEnterpriseId != (await this._getEnterprise(ctx)).enterpriseId) {
            throw new Error(`Enterprise Id dose not match.`);
        }
        if (application.currentState != states.returned) {
            throw new Error(`Can only submit this transaction with state ${states.returned}.`);
        }
        await this._updateApplication(application, states.terminated);
        application.signPledgeDate = signPledgeDate;
        await this._putStateFromObj(ctx, applicationId, application);
        return `Successfully sign pledge for application ${applicationId}`;
    }

    @Transaction(true)
    public async ETP_claimDefault(ctx: Context, applicationId: string, defaultDate: string): Promise<string> {
        await this._checkIdentity(ctx, participantIdentity.enterprise);
        const application: LoanApplication = await this._getApplicationById(ctx, applicationId);
        if (application.creatorEnterpriseId != (await this._getEnterprise(ctx)).enterpriseId) {
            throw new Error(`Enterprise Id dose not match.`);
        }
        if (application.currentState != states.effective) {
            throw new Error(`Can only submit this transaction with state ${states.effective}.`);
        }
        await this._updateApplication(application, states.foreclosure);
        application.defaultDate = defaultDate;
        await this._putStateFromObj(ctx, applicationId, application);
        return `Successfully claim default for application ${applicationId}`;
    }
```

Figure 16: Code for transaction(enterprises)

6.2.5 Query Files

In the query file, several query rules were defined by us for users. For the pepper enterprises, they may use the **ETPqueryApplicationStateById** to check the state of their loan applications. The query function asks the pepper enterprises to enter Application Id, and for each participant who submits a query, we verify their identity type and identity ID to ensure that only enterprises in the business network can use this query functionality. If the participant passes the verification, the function will return the currentState corresponding to the application ID.

```
public async ETP_queryApplicationStateById(ctx: Context, applicationId: string): Promise<string> {
    await this._checkIdentity(ctx, participantIdentity.enterprise);
    const application: LoanApplication = await this._getApplicationById(ctx, applicationId);
    if (application.creatorEnterpriseId !== (await this._getEnterprise(ctx)).enterpriseId) {
        throw new Error(`This is an application created by other enterprise.`);
    }
    const result = {
        "Application Id": applicationId,
        "Current State": application.currentState
    }
    return JSON.stringify(result);
}
```

Figure 17: Code for ETPqueryApplicationStateById

The pepper enterprises can also use the **ETPqueryApplication** to query all the information of their application without entering a specific application ID.

```
public async ETP_queryAllApplication(ctx: Context): Promise<string> {
    await this._checkIdentity(ctx, participantIdentity.enterprise);
    const startKey = '';
    const endKey = '';
    const allResults = [];
    for await (const {key, value} of ctx.stub.getStateByRange(startKey, endKey)) {
        var application: LoanApplication;
        try {
            application = JSON.parse(value.toString()) as LoanApplication;
        } catch (err) {
            console.log(err);
            continue;
        }
        if (!keySet.has(key) && application.creatorEnterpriseId == (await this._getEnterprise(ctx)).enterpriseId) {
            allResults.push({
                "Application Id": application.applicationId,
                "Application Info": application
            });
        }
    }
    return JSON.stringify(allResults);
}
```

Figure 18: Code for ETPqueryApplication

The return value will be the Application Id, create Date, Platform Id, Pledge Amount and other information, which are shown in the figure below. You can see that the status of Application 1 is "Application Processing" .

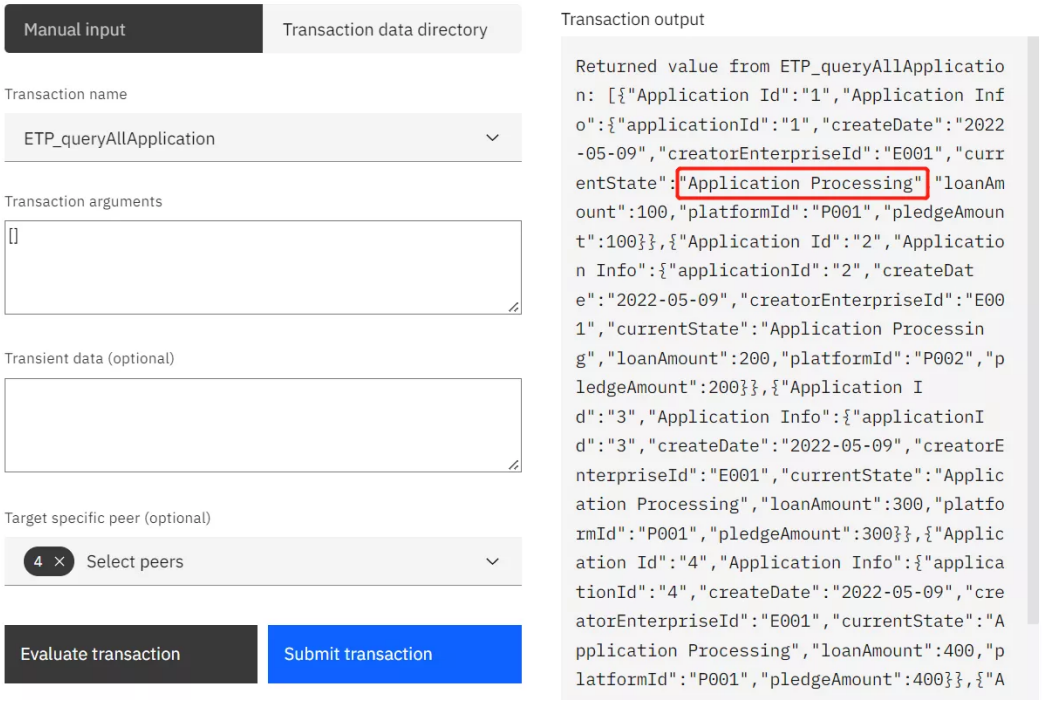


Figure 19: Result for ETPqueryApplication

Similar functions can be used to query application records and information related to other participants. The function for Platforms, Warehouses and Banks are P**PLTqueryAllApplication**, **WHSqueryAllApplication** and **BNKqueryAllApplication**, respectively. Corresponding code and return value of these functions are shown below:

Figure 20: Code and result for PLTqueryAllApplication

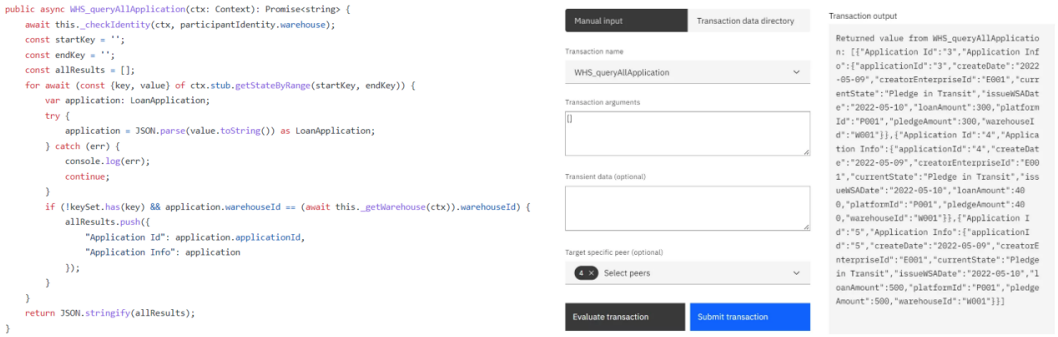


Figure 21: Code and result for WHSqueryAllApplication

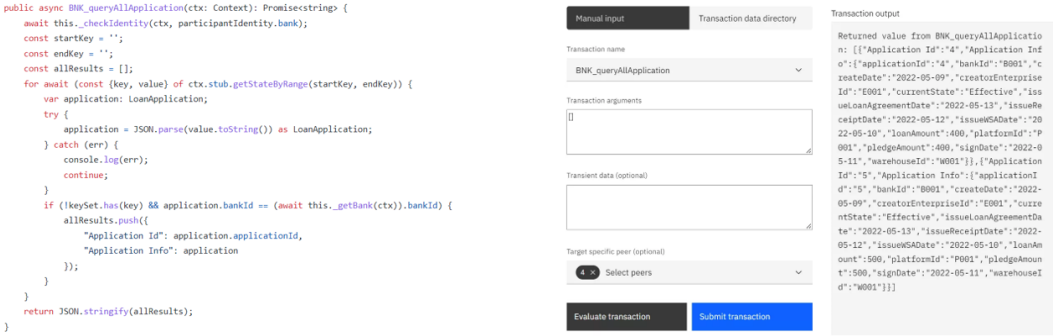


Figure 22: Code and result for BNKqueryAllApplication

7 Future Work and Possible Extensions

7.1 Advantages of Our Design and Implementation

7.1.1 Data reliability

Due to the immutability of the blockchain, all the data on the chain will be saved permanently and reliably. Especially in our application scenario, the consensus mechanism makes everyone trust each other, and the counterfeiting cost of any party is huge. It further ensures the safety and reliability of the whole process.

7.1.2 Data promptness

For many enterprises, they need financing in a short time to keep the company running smoothly. Considering past patterns, the processes are not only time-consuming but also cumbersome. Each operation link still simply relies on paper documents like the financing pledge and pledge loan agreement. However, our project can help to speed up the whole process. At the same time, it also reduces the time cost and operation cost of each participant.

7.1.3 Business integrity

Due to the integration of the actual business experience of professionals from real enterprises, we try our best to make the whole business process closer to reality. The business process is more complete and rich because we also consider many special situations, such as default when the enterprise cannot repay on time. At the same time, it is convenient and easy to expand functions on the basis, which also makes the whole system more robust and stable.

7.2 Drawbacks of the Design and Implementation

The interactive page is relatively simple, and the graphical user interface needs to be improved. Due to time constraints, the main interactions are completed by relying on the existing components of hyperledger itself. At the same time, if we want to apply it to practice, also a lot of modifications closer to the actual scene are need. For example, consider whether the participants of the platform can directly exit the chain and only need the other three parties in the supply chain to conduct transactions, without intermediators. In addition, we need to consider more query requirements and design more aggregate queries to enrich business scenarios.

7.3 Future Work and Possible Extensions

- Add more enterprise background information involved
- Enrich GUI interface with user friendly options and cool buttons.
- Allow multiple banks to join the blockchain and provide financing services for different types of enterprises
- Improve server throughput and response speed
- Develop and deploy mobile applications

8 Potential Impact of the Project

8.1 Impacts on supply chain finance

8.1.1 Improve Intermediary

The information system of the core enterprise cannot fully integrate all the transaction information of upstream and downstream enterprises. Banks have limited access to information, so they can neither get more information nor discern whether it is genuine. Due to lack of a trust , there exist the intermediaries

8.1.2 Reduce Manual Processing

The signing of agreements, the disbursement of funds and so on are all manual operations. Disputes can be costly. Thanks for blockchain, these costs can be greatly reduces.

8.1.3 Increase Trust

Financing can only proceed if there is trust on both sides of the agreement. Because of immutability and reliability, the platform built on the blockchain can determine the identity of trading parties and identify patients, hospitals, medical institutions and insurance companies, which can be said to be a smart contract.

8.1.4 Authentication

Successful financing requires the signing of multiple agreements, e.g. pledge loan agreement and warehouse supervision agreement and so on. And different identities have different authority. Our project can provide authority and identity authentication automatically.

8.2 Impacts on other supply chains

Blockchain can effectively solve the phenomenon of information island. Based on the big data analysis of the supply chain, blockchain can provide more information sources, provide high-quality data information, effectively reduce the risk of data leakage, and ensure the security, effectiveness and credibility of big data in the supply chain. Effectively get through the fragmentation of raw material procurement, production, logistics, sales, supervision and other information in the supply chain, and establish a supply chain information trading platform based on big data credit. With the popularization of blockchain technology, the

digital economy under smart supply chain will be more authentic. With the in-depth application and development of blockchain in the field of supply chain, the future digital economy society will become more fair and transparent.

9 Evaluation letter from our sponsor

Dear Group5 members of Blockchain Application Course,

On behalf of contact person in charge of the Internet upgrade of many agricultural industries, I would like to thank you for your efforts. I still remember our meeting when we discussed the business logic, and now you have deployed your design,Congratulations! I express great interest in your blockchain-based supply chain financing solution. Here is my opinion about your project.

The supply chain financing solution based on blockchain technology is designed to satisfy the urgent needs for financing of many small pepper enterprises. The basic requirement improve the funding efficiency, which means let the enterprises receive the loan as soon as possible once they send the pledge to the platforms. The application you design can help us achieve this goal.

What impressed me most is your good design at the inventory pledge application stage. I believe this will help the can improve our existing process by making it more specific. Also, The good property of blockchain also provides me new insight of how it could be used in state changing of application. This can be very useful because the immutable property of the ledgers can be used in anti-fraud, basis for Internal supervision.

Your project define a detailed model between four entities and different access to the pledge data, which is theoretically acceptable. But in reality, the bank needs to have more permission to know the current state of the pledge in order to their risk management. This should be considered in your future enhancement. And I also recommend you to take the privacy transaction of each enterprise into consideration. After all, no enterprise wants to let peers know about its financial situation.

Given all of the above, I appreciate the your efforts and useful design. And I hope that we can apply blockchain technology to solve the current problems as soon as possible.

Mr, Shen

20 May, 2022

A Appendix

操作说明

1 登录服务器

因为我们的应用的交互是基于VSCode的IBM Blockchain Platform插件，所以需要使用VSCode登录服务器进行操作。
使用VSCode的远程资源管理器登录服务器：

```
ssh ibm@8.219.67.160
psw: ibm
```

参考：https://zhuanlan.zhihu.com/p/141205262

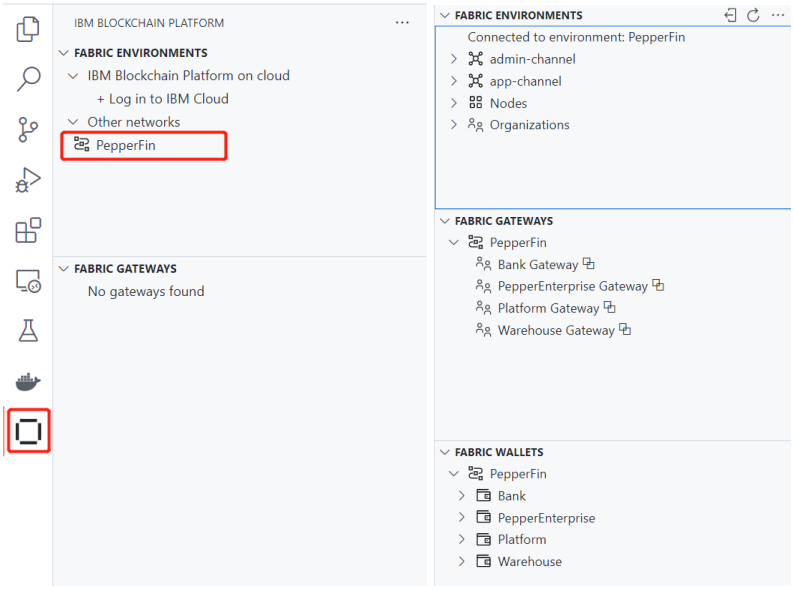
登陆不上可以使用手机热点。

2 初始化、启动和连接fabric网络

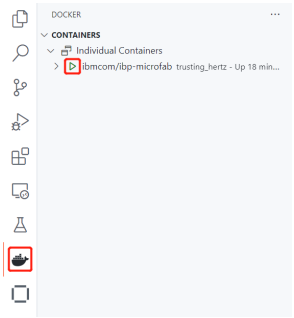
1. 打开VSCode集成终端，输入指令以初始化和启动fabric网络

```
source /home/ibm/Document/PepperFin-ts/run-network.sh
```

2. 进入IBM Blockchain Platform插件，点击 `PepperFin` ，连接到fabric网络

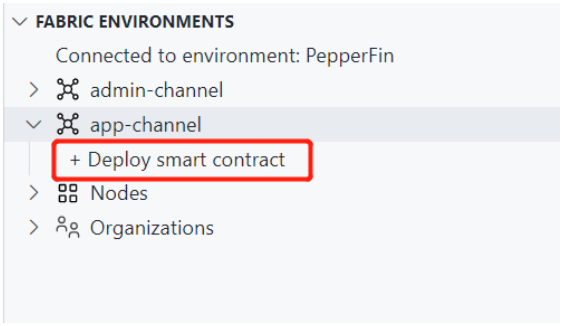


如果连接不上，检查docker容器是否已经启动，否则重复步骤1。

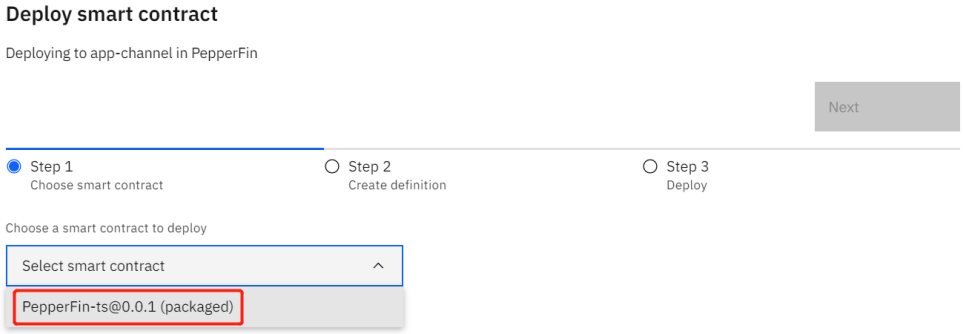


3 部署链码

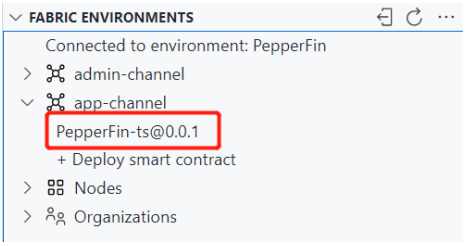
1. 点击 `FABRIC ENVIRONMENTS` 窗口下的 `app-channel` -> `+ Deploy smart contract`



2. 在弹出的 `Deploy smart contract` 窗口中，选择已经打包好的链码 `PepperFin-ts@0.0.1(packaged)` ，点击 `next` ； `Step2` 和 `Step3` 无需配置，选择 `next` 和 `Deploy` ，稍等片刻，系统将会把链码部署在网络上。

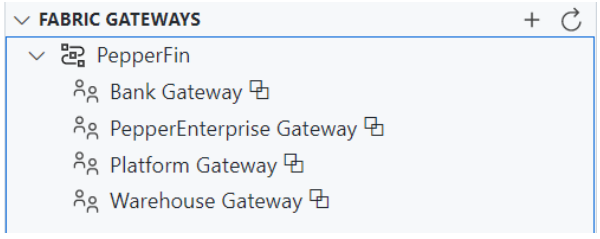


部署成功后， `app-channel` 下会出现 `PepperFin-ts@0.0.1` 。

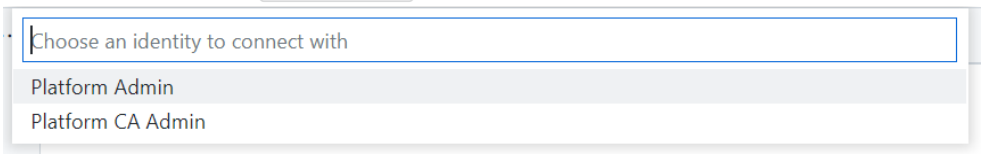


4 登录与Transaction的使用

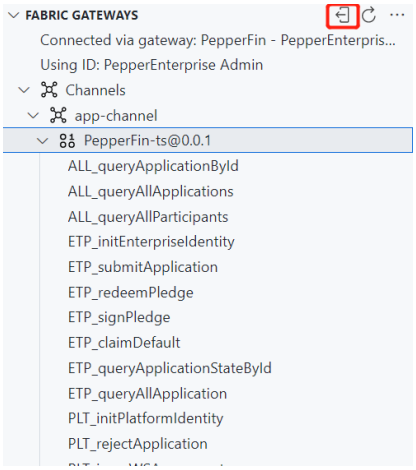
本网络中一共有四个 `org` ，代表参与一个业务的四方企业， `FABRIC GATEWAYS` 栏下的四个Gateway分别是他们访问fabric网络的通道。以不同的身份登录会有不同的操作权限。



以Platform为例，如果要以Platform的身份登陆网络，访问智能合约，点击 `Platform Gateway` ，在上方的弹窗中选择 `Platform Admin` ：



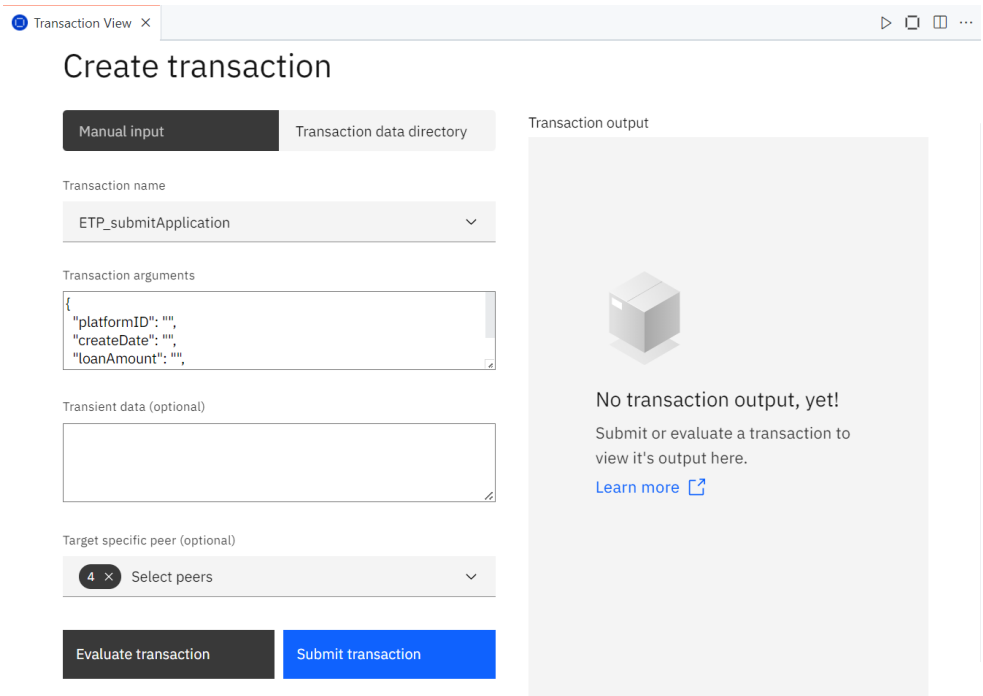
登录后即可以Platform的身份访问各个通道中的智能合约，点击右上角按钮可以退出登录。



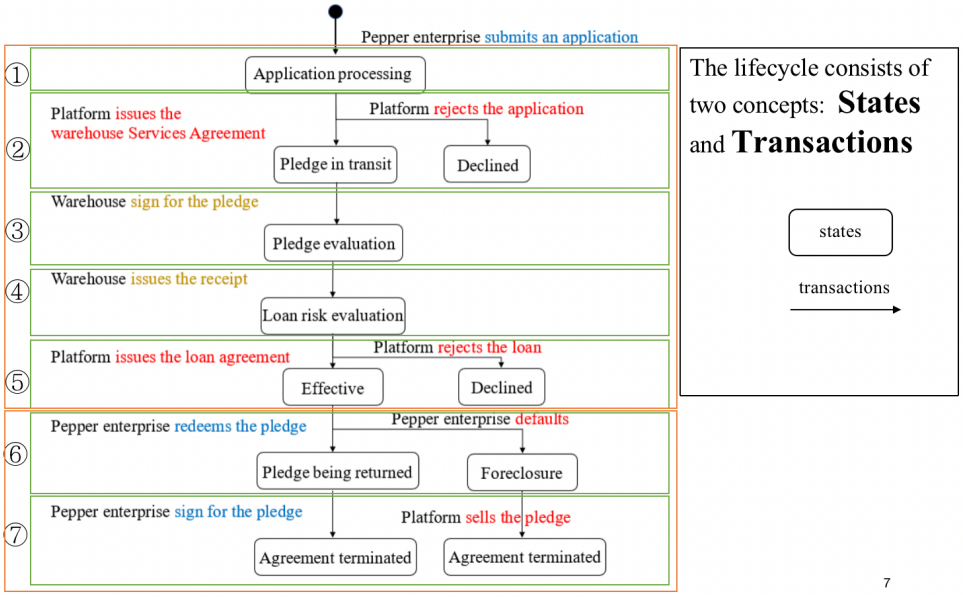
其中，Transaction的前缀代表使用权限，登陆身份不匹配时则会报错：

- `ALL`：所有身份均可使用
- `ETP`：仅限融资企业使用
- `PLT`：仅限平台使用
- `BNK`：仅限银行使用
- `WHS`：仅限仓储使用

点击任何一个Transaction，会弹出 `Transaction View` 窗口，在这个窗口下可以进行交易的测试。



5 业务流程示例

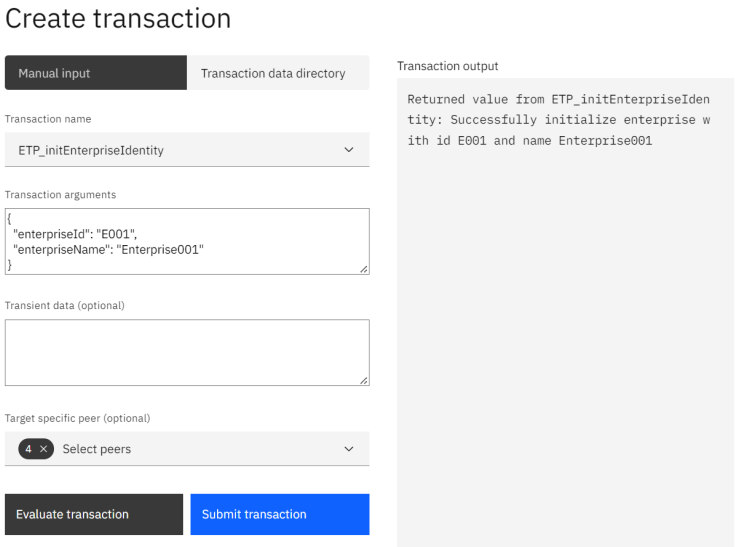


5.1 初始化参与企业信息

分别以四个Participants的身份登录，使用 `[前缀]_init[身份]Identity` 方法，初始化信息。以 `PepperEnterprise` 为例，登录 `PepperEnterprise Admin` ，点击 `ETP_initEnterpriseIdentity` 方法，在 `Transaction arguments` 中输入

```
{
  "enterpriseId": "E001",
  "enterpriseName": "Enterprise001"
}
```

点击 `Submit transaction` ，右边 `Transaction output` 窗口中会提示信息录入成功。



类似地，接着分别登录 `Bank Admin` ， `Platform Admin` ， `Warehouse Admin` ，使用以下数据初始化各自企业信息。

- `BNK_initBankIdentity` ：

```
{
  "bankId": "B001",
  "bankName": "Bank001"
}
```

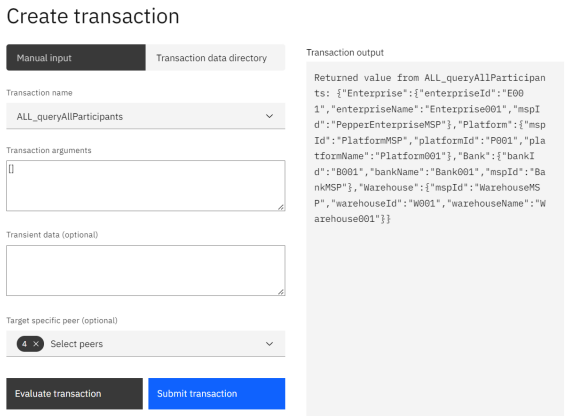
- `PLT_initPlatformIdentity` ：

```
{
  "platformId": "P001",
  "platformName": "Platform001"
}
```

- `WHS_initWarehouseIdentity` ：

```
{
  "warehouseId": "W001",
  "warehouseName": "Warehouse001"
}
```

完成所有的初始化后，以任意身份登录，可以使用 `ALL_queryAllParticipants` 方法查询到所有的参与企业信息：



5.2 融资企业 – 提交申请

> 以融资企业身份登录。

融资企业可以使用 `ETP_submitApplication` 方法提交融资申请。
创建以下五个申请用作示例：

```
{
  "platformID": "P001",
  "createDate": "2022-05-09",
  "loanAmount": 100,
  "pledgeAmount": 100
}
```

```
{
  "platformID": "P002",
  "createDate": "2022-05-09",
  "loanAmount": 200,
  "pledgeAmount": 200
}
```

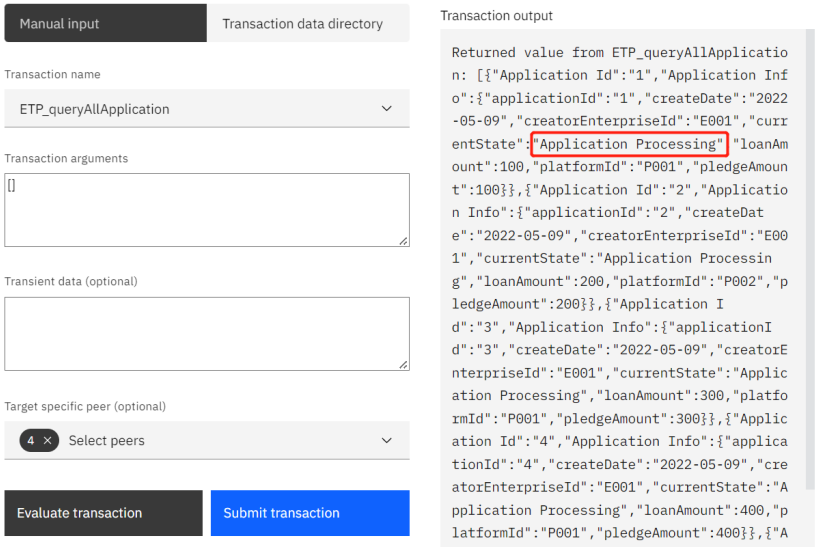
```
{
  "platformID": "P001",
```

```
  "createDate": "2022-05-09",
  "loanAmount": 300,
  "pledgeAmount": 300
}
```

```
{
  "platformID": "P001",
  "createDate": "2022-05-09",
  "loanAmount": 400,
  "pledgeAmount": 400
}
```

```
{
  "platformID": "P001",
  "createDate": "2022-05-09",
  "loanAmount": 500,
  "pledgeAmount": 500
}
```

创建成功后，可以使用 `ETP_queryAllApplication` 方法查询改企业创建的所有申请：

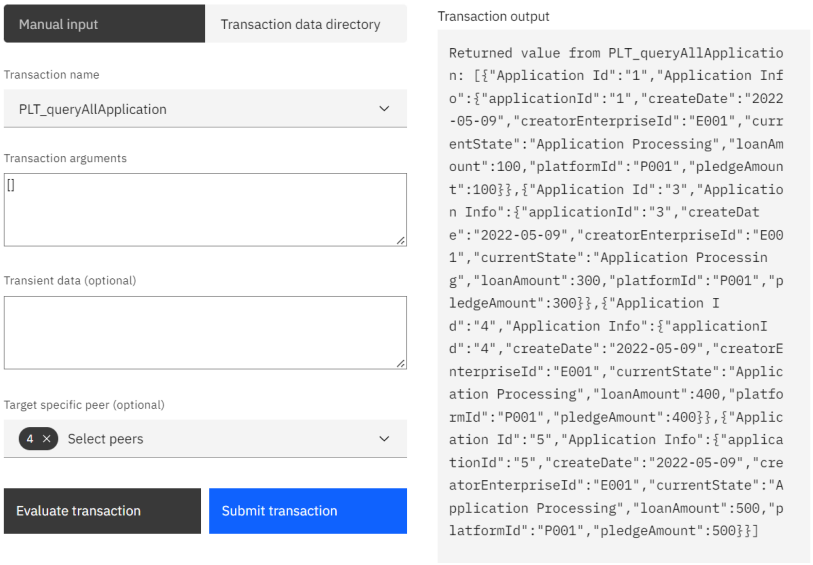


可以看到，所有新创建的申请都处于 `Application Processing` 状态。

5.3 平台 – 处理申请

> 以平台身份登录。

平台可以使用 `PLT_queryAllApplication` 方法查询所有向当前平台提交的申请：



可以看到，申请2由于申请平台是”P002”，而当前平台是”P001”，所以没有出现在当前查询结果中。

拒绝申请

平台可以使用 `PLT_rejectApplication` 方法拒绝融资企业的申请。
示例：拒绝申请1

```
{
  "applicationId": "1",
  "rejectDate": "2022-05-10"
}
```

重新查询申请，此时申请1状态变为 `declined` ，新增 `rejectedDate` 属性：

```
n: [{"Application Id":"1","Application Info":{"applicationId":"1","createDate":"2022-05-09","creatorEnterpriseId":"E001","currentState":"Declined","loanAmount":100,"platformId":"P001","pledgeAmount":100,"rejectedDate":"2022-05-10"}},{"Application I
```

注：每个transaction都有申请当前状态的检查，当申请处于不支持当前transaction的状态时，会提示错误。

同意申请，并向仓储企业申请服务

平台使用 `PLT_issueWSAgreement` 方法同意申请，并向仓储公司申请服务。
示例：同意申请3、4和5，并向仓储公司”W001”申请服务

```
{
  "applicationId": "3",
  "issueWSADate": "2022-05-10",
  "warehouseId": "W001"
}
```

```
{
  "applicationId": "4",
  "issueWSADate": "2022-05-10",
```

```
  "warehouseId": "W001"
}
```

```
{
  "applicationId": "5",
  "issueWSADate": "2022-05-10",
  "warehouseId": "W001"
}
```

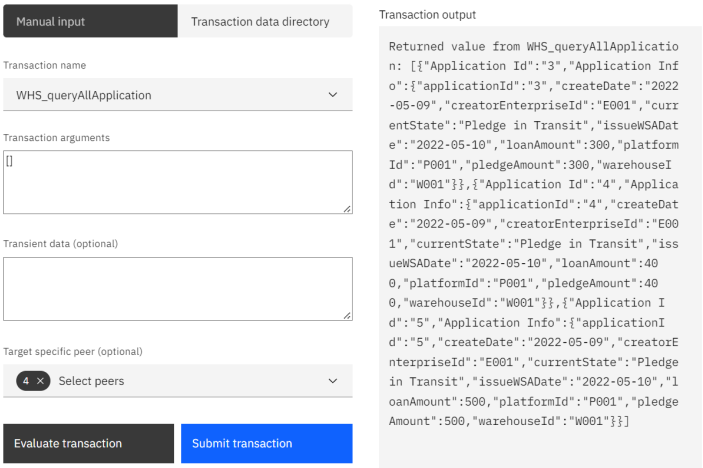
此时，三个申请的状态变为 `Pledge in Transit` ，代表抵押物正在向仓储公司转运。


```
edDate":"2022-05-10"}},{"Application I
d":"3","Application Info":{"applicationI
d":"3","createDate":"2022-05-09","creatorE
nterpriseId":"E001","currentState":"Pledge
in Transit","issueWSADate":"2022-05-10","l
oanAmount":300,"platformId":"P001","pledge
Amount":300,"warehouseId":"W001"}},{"Appli
cation Id":"4","Application Info":{"applic
ationId":"4","createDate":"2022-05-09","cr
eatorEnterpriseId":"E001","currentStat
e":"Pledge in Transit","issueWSADate":"202
2-05-10","loanAmount":400,"platformId":"P0
01","pledgeAmount":400,"warehouseId":"W00
1"}},{"Application Id":"5","Application In
fo":{"applicationId":"5","createDate":"202
2-05-09","creatorEnterpriseId":"E001","cur
rentState":"Pledge in Transit","issueWSADa
te":"2022-05-10","loanAmount":500,"platfor
mId":"P001","pledgeAmount":500,"warehouseI
d":"W001"}}]
```


5.4 仓储 – 签收货物

> 以仓储身份登录。

仓储可以通过 `WHS_queryAllApplication` 方法查询所有与该仓储公司相关申请：



可以看到，申请3、4、5出现在查询结果中。

仓储确认收货后，使用 `WHS_signPledge` 方法签收抵押物。

示例：签收申请3、4、5

```
{
```

```
  "applicationId": "3",
  "signDate": "2022-05-11"
}
```

```
{
  "applicationId": "4",
  "signDate": "2022-05-11"
}
```

```
{
  "applicationId": "5",
  "signDate": "2022-05-11"
}
```

此时，申请状态变为 `Pledge Evaluation` ，表示抵押物正在评估中。


```
n: [{"Application Id":"3","Application Inf
o":{"applicationId":"3","createDate":"2022
-05-09","creatorEnterpriseId":"E001","curr
entState":"Pledge Evaluation","issueWSADat
e":"2022-05-10","loanAmount":300,"platform
Id":"P001","pledgeAmount":300,"signDat
e":"2022-05-11","warehouseId":"W001"}},{"A
pplication Id":"4","Application Info":{"ap
plicationId":"4","createDate":"2022-05-0
9","creatorEnterpriseId":"E001","currentSt
ate":"Pledge Evaluation","issueWSADate":"2
022-05-10","loanAmount":400,"platformI
d":"P001","pledgeAmount":400,"signDate":"2
022-05-11","warehouseId":"W001"}},{"Applic
ation Id":"5","Application Info":{"applica
tionId":"5","createDate":"2022-05-09","cre
atorEnterpriseId":"E001","currentState":"P
ledge Evaluation","issueWSADate":"2022-05-
10","loanAmount":500,"platformId":"P00
1","pledgeAmount":500,"signDate":"2022-05-
11","warehouseId":"W001"}]
```


5.5 仓储 – 出具评估报告

仓储对货物的评估完成后，使用 `WHS_issueReceipt` 方法出具评估报告，将申请状态变为 `Loan Risk Evaluation` ，代表此时将由平台评估贷款申请的风险。

示例：出具申请3、4、5的评估报告

```
{
  "applicationId": "3",
  "issueDate": "2022-05-12"
}
```

```
{
  "applicationId": "4",
  "issueDate": "2022-05-12"
}
```

```
{
  "applicationId": "5",
```

```
  "issueDate": "2022-05-12"
}
```


```
n: [{"Application Id":"3","Application Inf
o":{"applicationId":"3","createDate":"2022
-05-09","creatorEnterpriseId":"E001","curr
entState":"Loan Risk Evaluation","issueRec
eiptDate":"2022-05-12","issueWSADate":"202
2-05-10","loanAmount":300,"platformId":"P0
01","pledgeAmount":300,"signDate":"2022-05
-11","warehouseId":"W001"}},{"Application
Id":"4","Application Info":{"applicationI
d":"4","createDate":"2022-05-09","creatorE
nterpriseId":"E001","currentState":"Loan R
isk Evaluation","issueReceiptDate":"2022-0
5-12","issueWSADate":"2022-05-10","loanAmo
unt":400,"platformId":"P001","pledgeAmoun
t":400,"signDate":"2022-05-11","warehouseI
d":"W001"}},{"Application Id":"5","Applica
tion Info":{"applicationId":"5","createDat
e":"2022-05-09","creatorEnterpriseId":"E00
1","currentState":"Loan Risk Evaluatio
n","issueReceiptDate":"2022-05-12","issueW
SADate":"2022-05-10","loanAmount":500,"pla
tformId":"P001","pledgeAmount":500,"signDa
te":"2022-05-11","warehouseId":"W001"}}]
```


5.6 平台 – 根据评估结果处理申请

> 以平台身份登录

拒绝申请

平台可以通过 `PLT_rejectApplication` 方法拒绝申请，申请状态变为 `Declined` 。
示例：拒绝申请3

```
{
  "applicationId": "3",
  "rejectDate": "2022-05-13"
}
```


```
d":"3","Application Info":{"applicationI
d":"3","createDate":"2022-05-09","creatorE
nterpriseId":"E001","currentState":"Declin
ed","issueReceiptDate":"2022-05-12","issue
WSADate":"2022-05-10","loanAmount":300,"pl
atformId":"P001","pledgeAmount":300,"rejec
tedDate":"2022-05-13","signDate":"2022-05-
11","warehouseId":"W001"}},{"Application I
```


同意申请，发放贷款

平台可以通过 `PLT_issueLoanAgreement` 方法同意申请并发放贷款，需填入发放贷款的银行id，此时申请状态变为 `Effective` 。
示例：同意申请4、5

```
{
  "applicationId": "4",
  "issueLoanAgreementDate": "2022-05-13",
  "bankId": "B001"
```

```
}
```

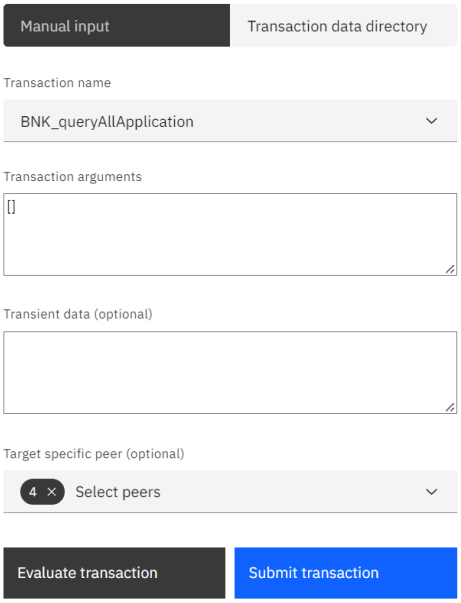
```
{
  "applicationId": "5",
  "issueLoanAgreementDate": "2022-05-13",
  "bankId": "B001"
}
```


```
11","warehouseId":"W001"}},{"Application Id":"4","Application Info":{"applicationId":"4","bankId":"B001","createDate":"2022-05-09","creatorEnterpriseId":"E001","currentState":"Effective","issueLoanAgreementDate":"2022-05-13","issueReceiptDate":"2022-05-12","issueWSADate":"2022-05-10","loanAmount":400,"platformId":"P001","pledgeAmount":400,"signDate":"2022-05-11","warehouseId":"W001"}},{"Application Id":"5","Application Info":{"applicationId":"5","bankId":"B001","createDate":"2022-05-09","creatorEnterpriseId":"E001","currentState":"Effective","issueLoanAgreementDate":"2022-05-13","issueReceiptDate":"2022-05-12","issueWSADate":"2022-05-10","loanAmount":500,"platformId":"P001","pledgeAmount":500,"signDate":"2022-05-11","warehouseId":"W001"}}]
```


5.7 银行 – 查询贷款申请

> 登录银行账户。

银行可以使用 `BNK_queryAllApplication` 方法查询所有当前银行放出的贷款申请



Transaction output


```
Returned value from BNK_queryAllApplication: [{"Application Id":"4","Application Info":{"applicationId":"4","bankId":"B001","createDate":"2022-05-09","creatorEnterpriseId":"E001","currentState":"Effective","issueLoanAgreementDate":"2022-05-13","issueReceiptDate":"2022-05-12","issueWSADate":"2022-05-10","loanAmount":400,"platformId":"P001","pledgeAmount":400,"signDate":"2022-05-11","warehouseId":"W001"}},{"Application Id":"5","Application Info":{"applicationId":"5","bankId":"B001","createDate":"2022-05-09","creatorEnterpriseId":"E001","currentState":"Effective","issueLoanAgreementDate":"2022-05-13","issueReceiptDate":"2022-05-12","issueWSADate":"2022-05-10","loanAmount":500,"platformId":"P001","pledgeAmount":500,"signDate":"2022-05-11","warehouseId":"W001"}}]
```


可以看到申请4、5出现在了查询结果中。

5.8 企业 – 还款

> 登录企业账户。

1. 对于贷款生效中的申请，企业可以使用 `ETP_redeemPledge` 方法还款并赎回抵押物，此时申请状态变为 `Pledge being Returned` ，表示抵押物正在返还中。
   示例：申请4还款

```
{
  "applicationId": "4",
```

```
  "redeemDate": "2022-05-14"
}
```


```
e":"2022-05-11","warehouseId":"W001"}},{"A
pplication Id":"4","Application Info":{"ap
plicationId":"4","bankId":"B001","createDa
te":"2022-05-09","creatorEnterpriseId":"E0
01","currentState":"Pledge being Returne
d","issueLoanAgreementDate":"2022-05-1
3","issueReceiptDate":"2022-05-12","issueW
SADate":"2022-05-10","loanAmount":400,"pla
tformId":"P001","pledgeAmount":400,"redeem
Date":"2022-05-14","signDate":"2022-05-1
1","warehouseId":"W001"}},{"Application I
```


2. 收到抵押物后，企业可以使用 `ETP_signPledge` 方法签收货物，此时申请状态变为 `Terminated` ，代表业务已结束。
   示例：签收申请4

```
{
  "applicationId": "4",
  "signPledgeDate": "2022-05-15"
}
```


```
pplication Id":"4","Application Info":{"ap
plicationId":"4","bankId":"B001","createDa
te":"2022-05-09","creatorEnterpriseId":"E0
01","currentState":"Agreement Terminate
d","issueLoanAgreementDate":"2022-05-1
3","issueReceiptDate":"2022-05-12","issueW
SADate":"2022-05-10","loanAmount":400,"pla
tformId":"P001","pledgeAmount":400,"redeem
Date":"2022-05-14","signDate":"2022-05-1
1","signPledgeDate":"2022-05-15","warehous
eId":"W001"}},{"Application Id":"5","Appli
```


5.9 企业 – 违约

对于贷款生效中的申请，企业可以使用 `ETP_claimDefault` 方法宣布违约，此时申请状态变为 `Foreclosure` ，表示抵押物已归平台所有。
示例：申请5违约

```
{
  "applicationId": "5",
  "defaultDate": "2022-05-14"
}
```


```
eId":"W001"}},{"Application Id":"5","Appli
cation Info":{"applicationId":"5","bankI
d":"B001","createDate":"2022-05-09","creat
orEnterpriseId":"E001","currentState":"For
eclosure","defaultDate":"2022-05-14","issu
eLoanAgreementDate":"2022-05-13","issueRec
eiptDate":"2022-05-12","issueWSADate":"202
2-05-10","loanAmount":500,"platformId":"P0
01","pledgeAmount":500,"signDate":"2022-05
-11","warehouseId":"W001"}}]
```


5.10 平台 – 转卖违约抵押物

> 登录平台账户

对于违约的申请，平台可以使用 `PLT_sellPledge` 方法出售抵押物，此时申请状态变为 `Terminated` ，代表业务已经结束。
示例：申请5出售抵押物

```
{
  "applicationId": "5",
  "sellPledgeDate": "2022-05-15"
}
```


```
{"Application Id":"5","Application Info":
{"applicationId":"5","bankId":"B001","crea
teDate":"2022-05-09","creatorEnterpriseI
d":"E001","currentState":"Agreement Termin
ated","defaultDate":"2022-05-14","issueLoa
nAgreementDate":"2022-05-13","issueReceipt
Date":"2022-05-12","issueWSADate":"2022-05
-10","loanAmount":500,"platformId":"P00
1","pledgeAmount":500,"sellPledgeDate":"20
22-05-15","signDate":"2022-05-11","warehou
seId":"W001"}}]
```