ProofNet

A SECURE BLOCKCHAIN DATABASE FOR SMALL BUSINESSES

FINNEGAN O'KEEFE, OLIVER OLSON, PETER HASEMEIER

CURRENT MARKET

The core idea of the financial management structure is fragmented, giving small businesses too many options and leading them to rely on themselves for financial management support. With that, a lot of the time, these financial tools that are offered can be inefficient for smaller businesses. Smaller businesses rely on multiple disconnected platforms (i.e., payments, bookkeeping, taxes), creating a highly fragmented market. Many of the tools involved in these markets are either too simple, lacking automation and protection, or too complex, meaning they are tailored towards the bigger enterprises, not small businesses.

KEY ACTIVITIES

What ProofNet actually does is continuous bookkeeping, creating a real-time database of different financial transactions. Things that are included in ProofNet are automated reconciliation between invoices and expenses, tax calculations based on multiple factors (i.e., jurisdiction), creating and then verifying invoices, and creating an immutable record of all actions internally and externally. Trust is workflow automation and blockchain. With blockchain, all of this data and the transactions put forth are fully trusted and immutable.

USE CASES

This idea came up when I was speaking with my dad, who owns a small dry ice blasting business. Going through multiple conversations and asking his personal opinion on financial management within a small business, we came up with real use cases that could be implemented through blockchain in his company.

Automated Bookkeeping: He runs a business by himself, and currently does everything by hand. In our slides, we labeled this as “personal records” or pen and paper. There are a few positives and many negatives to personal record keeping. Showing the app we created,

automated bookkeeping, we thought would save him time. Having a set amount of variables that he would still need to plug in, and using all the key activities listed above, he would save around 10 hours a month.

Verified Financial History: Blockchain allows one to have an ID code to track all of their financial matters. This can enable different companies to see the financial history of a smaller business. Lots of bigger businesses go off their name brand, which brings in customers; smaller businesses don't have that privilege, especially when first starting. With that being said, blockchain keeps all data in one place while also maximizing security. Given a wallet ID, our dry ice blasting company can show that to a potential client, proving their work through financials. The trust behind that wallet ID comes from the fact that most transactions are pulled directly from bank feeds and payment processors like PayPal rather than entered by the business owner manually. Because those records originate with third-party financial institutions, a client or bank reviewing the wallet ID can see that the data was not self-reported and could not have been altered after the fact. For transactions that are entered manually, the wallet ID will clearly label them as self-reported so that whoever is reviewing the financial history can see exactly which records came from a third-party source and which ones were entered by the business owner themselves. This way the validation process does not depend entirely on the business owner, and anyone looking at the wallet ID can make their own judgment about how much of the data is independently verified

Finally, Expense Reconciliation: Human error will always be a thing. However, blockchain and our coding will allow internal expenses to match external outputs. Things like reports, receipts, or statements will all be in the blockchain itself, so when future data is implemented, continuous financial management can occur. Transactions are cross-checked against uploaded receipts, payment records, and bank deposits to identify missing or inconsistent entries. Because records

are timestamped and stored on the blockchain ledger, suspicious edits or deleted transactions can be flagged automatically for review.

TARGET MARKETS

Our target market is mainly small businesses in the United States. There are around 33 million small businesses in the US, so there are a lot of potential customers. We are mainly focused on three different groups of people.

The first group is solo operators and really small businesses with around 1-10 employees. These are people like plumbers, electricians, contractors, and other tradespeople. They make real money, but they don't have the budget to hire an accountant or bookkeeper. Most of them are using spreadsheets or just recording by hand, which is exactly what my dad does. ProofNet would save them a lot of time every month and give them better records than they have now.

The second group is small businesses that are trying to get new clients or get a loan from a bank. These businesses need to show that they are financially trustworthy but they don't have a good way to do that right now. With ProofNet's wallet ID they can share their financial history with whoever needs to see it and it's verified through blockchain so nobody can say it was faked.

The third group is businesses in industries that have a lot of rules and regulations, like food service or healthcare. These businesses need thorough financial records to stay compliant and ProofNet helps with that, too.

IMPLEMENTATION

We plan to build ProofNet in three phases over the course of about a year.

Phase 1 (Months 1-4): The first thing we need to do is build the actual blockchain and the main dashboard that users will see. This includes the ledger where all transactions get stored, the interface where business owners can log their income and expenses, and the wallet ID system. By the end of this phase we want to have a working app that people can actually use, even if it still requires some manual entry. We would test this with my dad's business first.

Phase 2 (Months 5-8): Once the basic product is working we want to add automation. This means connecting to payment apps like Stripe and PayPal so transactions import automatically, adding bank feed connections, and building the tax calculator that figures out the right tax rates based on where the business is located. After this phase most of the bookkeeping should basically take care of itself.

Phase 3 (Months 9-12): The last phase is about adding features that help businesses grow. This includes the shareable financial summary portal, a mobile app so people can use it from the job site, and tools for generating reports that banks and clients expect. We also want to add a way for businesses to bring in their old records from spreadsheets.

REVENUE STREAM

ProofNet will make money through a monthly subscription model. We have three different pricing tiers depending on what features the business needs.

The Starter plan is $19 per month and includes the blockchain ledger, manual entry, the wallet ID, and basic reporting. This is for businesses that are just starting out or just want the basics.

The Professional plan is $49 per month and includes everything in Starter plus bank feed connections, the tax calculator, automated reconciliation, and invoice tools. This is probably where most of our customers will end up.

The Business plan is $99 per month and includes everything plus audit reports, the mobile app, priority customer support, and the ability to have multiple users on the same account.

On top of subscriptions we also plan to charge a small fee (around $0.10) per verified invoice for high volume users and a one-time fee for generating special reports for bank loans. If we can get 500 Starter customers, 200 Professional customers, and 50 Business customers by the end of year one that would be almost $25,000 a month in revenue which would be enough to keep the company running and keep building new features.

COMPETITION

There are already some products out there that do parts of what ProofNet does. The main competitors are QuickBooks, FreshBooks, Wave, and Xero.

QuickBooks is the biggest one and has a lot of features but it is expensive and complicated for a one-person business. It also doesn't have any blockchain so the records aren't verified the same way ours would be.

FreshBooks is simpler and good for invoicing but it doesn't have deep bookkeeping features and there is no way to share your financial history with clients in a trusted format.

Wave is free which is nice but you have to do almost everything manually and there is no automation or blockchain component at all.

Xero is popular with bigger small businesses and has good bank integrations but it is designed for companies that have actual staff and is too complicated and expensive for solo operators.

None of these competitors offer blockchain-based verification or the wallet ID credentialing feature that we think is one of ProofNet's biggest selling points.

COMPETITIVE ADVANTAGE

ProofNet has three main advantages over the competition.

First, our records are immutable. Because we use blockchain, once a transaction is recorded it can't be changed or deleted. This is something that regular accounting software just can't offer. A business owner can prove to anyone that their financial history is real and hasn't been tampered with.

Second, we are built specifically for small businesses. Big companies can use enterprise blockchain tools, but those are costly and are too complicated for a one-person plumbing or landscaping business. And regular accounting apps don't have the trust layer that blockchain provides. ProofNet is designed to sit in the middle and be both affordable and trustworthy.

Third, the wallet ID turns your bookkeeping into something you can actually use to grow your business. Instead of just tracking your expenses for tax season, ProofNet gives you a way to show potential clients or banks that you are financially reliable. This is a big deal for small businesses that don't have a well-known brand name to rely on.

Data Privacy and Security

Because ProofNet handles sensitive financial data, security is a core part of how the platform is built. All data is encrypted both when it is stored and when it is being transmitted between the user and our servers. Users log in with multi-factor authentication, and on the Business plan, account owners can control what each team member is allowed to see or do. Personal information like names, addresses, and tax IDs is stored in a separate database from the blockchain ledger, so it can be deleted if a customer requests it without affecting the integrity of the records. We will also obtain cyber liability insurance and complete a security review before accepting any real customer data.

POTENTIAL RISKS

There are some risks we need to think about with ProofNet.

Technical Risk: Blockchain is more complicated to build than a regular database and could have performance issues as we get more users. There is also the issue of smart contract bugs, which are hard to fix after the fact because of how immutable the ledger is. We plan to use a permissioned blockchain instead of a public one to keep costs and speed manageable, and we will store only transaction hashes on-chain while keeping full documents in a separate encrypted database. We will also have the smart contract code reviewed before launch.

Adoption Risk: A lot of small business owners, especially older tradespeople, might not want to switch from how they already do things, and the word "blockchain" can sound confusing or technical to someone who just wants to track their invoices. We plan to make the app really easy to use so the blockchain side is invisible to the user, and we will offer a free trial so people can try it without any commitment. The fact that we already have a real small business willing to try it (my dad's company) is a good start.

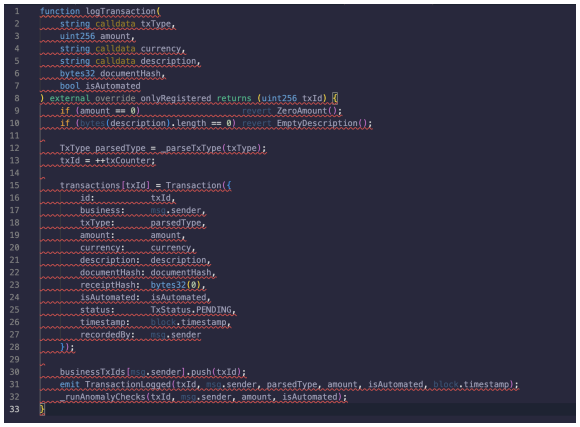
Legal Risk: Financial data has a lot of privacy laws around it, and blockchain creates unique issues when it comes to things like the right to delete your data, since the whole point of a blockchain is that you can't change or remove anything. We plan to handle this by only storing personal information like names and tax IDs in a regular database that can be deleted, and putting only anonymous transaction hashes on the blockchain. That way if someone asks us to delete their data we can do that without touching the ledger. We would also need to talk to a lawyer before launching to make sure we are doing everything right.

Competition Risk: If ProofNet starts getting popular, bigger companies like Intuit could just add blockchain features to QuickBooks. They have way more money and customers than us. Our best defense against this is to move fast and build a loyal user base before that happens, and to focus on solo operators and really small businesses that big enterprise tools will never actually be designed for. The wallet ID also creates a switching cost because the longer a business uses ProofNet, the more financial history they build up, which makes it harder to walk away from.

CODE IMPLEMENTATION

The following code excerpts are taken from ProofNet's core smart contract, written in Solidity for deployment on a permissioned blockchain network. Each function corresponds directly to one of the key activities described above.

Transaction Logging: Every financial record that enters ProofNet is written to the ledger through the logTransaction function. The isAutomated flag is what distinguishes a transaction that came from a third-party source like Stripe or a bank feed from one that was entered manually by the business owner. This is the technical foundation of the wallet ID credentialing system. Anyone reviewing a business's financial history can see exactly which records were independently sourced and which were self-reported.



```
function logTransaction(
    string calldata txType,
    uint256 amount,
    string calldata currency,
    string calldata description,
    bytes32 documentHash,
    bool isAutomated
) external override onlyRegistered returns (uint256 txId) {
    if (amount == 0)                          revert ZeroAmount();
    if (bytes(description).length == 0) revert EmptyDescription();

    TxType parsedType = _parseTxType(txType);
    txId = ++txCounter;

    transactions[txId] = Transaction({
        id:           txId,
        business:     msg.sender,
        txType:       parsedType,
        amount:       amount,
        currency:     currency,
        description:  description,
        documentHash: documentHash,
        receiptHash:  bytes32(0),
        isAutomated:  isAutomated,
        status:       TxStatus.PENDING,
        timestamp:    block.timestamp,
        recordedBy:   msg.sender
    });

    businessTxIds[msg.sender].push(txId);
    emit TransactionLogged(txId, msg.sender, parsedType, amount, isAutomated, block.timestamp);
    _runAnomalyChecks(txId, msg.sender, amount, isAutomated);
}
```

Anomaly Detection: This function runs automatically every time a manual transaction is logged. It checks for two things: whether the amount is unusually large, and whether the business has logged an excessive number of manual entries in a single day. Both of these patterns could indicate attempted manipulation of the financial record. Transactions that trigger either check are flagged for review and marked as such in the ledger, so the flag becomes part of the permanent record.

```
function _runAnomalyChecks(
    uint256 txId,
    address business,
    uint256 amount,
    bool isAutomated
) internal {
    if (isAutomated) return;

    if (amount >= largeManualEntryThreshold) {
        transactions[txId].status = TxStatus.FLAGGED;
        emit TransactionFlagged(txId, business,
            "Large manual entry exceeds threshold — pending review");
        return;
    }

    uint256 today = block.timestamp / 86400;
    dailyManualCount[business][today]++;
    if (dailyManualCount[business][today] > maxDailyManualEntries) {
        transactions[txId].status = TxStatus.FLAGGED;
        emit TransactionFlagged(txId, business,
            "Daily manual entry limit exceeded — pending review");
    }
}
```

Wallet ID Summary**:** This is the function that powers the wallet ID credentialing feature. When a business shares their wallet ID with a potential client or a bank, this is the data they are seeing. It breaks down the total transaction history into verified, manual, and flagged counts separately, so whoever is reviewing it can make their own judgment about how much of the record came from independent sources versus the business owner themselves.

```
function getWalletSummary(address business)
    external view override
    returns (
        uint256 totalTransactions,
        uint256 verifiedCount,
        uint256 manualCount,
        uint256 flaggedCount,
        uint256 memberSince
    )
{
    Business storage b = businesses[business];
    if (!b.registered) revert BusinessNotRegistered();

    uint256[] storage ids = businessTxIds[business];
    totalTransactions = ids.length;
    memberSince = b.registeredAt;

    for (uint256 i = 0; i < ids.length; i++) {
        Transaction storage t = transactions[ids[i]];
        if (t.status == TxStatus.VERIFIED ||
            t.status == TxStatus.RECONCILED) verifiedCount++;
        if (!t.isAutomated) manualCount++;
        if (t.status == TxStatus.FLAGGED)   flaggedCount++;
    }
}
```

Expense Reconciliation: This function is how ProofNet automatically matches internal records against bank feed data. A validator node runs it periodically, passing in a set of hashes from the business's connected bank account or payment processor. Any transaction that has a matching hash gets marked as reconciled. Any manual entry that has no match in the bank feed gets flagged automatically, without requiring a human to review every line item individually.

```
function reconcile(address business, bytes32[] calldata bankHashes)
    external onlyValidator
{
    if (!businesses[business].registered) revert BusinessNotRegistered();

    uint256[] storage ids = businessTxIds[business];
    uint256 matched = 0;
    uint256 flagged = 0;

    for (uint256 i = 0; i < ids.length; i++) {
        Transaction storage t = transactions[ids[i]];
        if (t.status != TxStatus.PENDING) continue;

        bool found = false;
        for (uint256 j = 0; j < bankHashes.length; j++) {
            if (t.documentHash == bankHashes[j]) {
                t.receiptHash = bankHashes[j];
                t.status      = TxStatus.RECONCILED;
                found = true;
                matched++;
                break;
            }
        }

        if (!found && !t.isAutomated) {
            t.status = TxStatus.FLAGGED;
            flagged++;
            emit TransactionFlagged(ids[i], business,
                "No matching bank record for manual entry");
        }
    }

    emit ReconciliationCompleted(business, matched, flagged);
}
```

REFLECTION

After presenting ProofNet, we received feedback that pointed out a few areas we needed to develop further. The main things that were flagged were that we needed to be clearer about how invoices are tracked and what stops someone from leaving out transactions or entering fake ones, how we are protecting customer financial data, and how the validation process works without just relying on the business owner to verify their own records. We also needed to be more specific about our mitigations for each risk rather than just identifying the problems.

In response to that feedback, we expanded the Expense Reconciliation section to explain how transactions are automatically cross-checked against bank records and how suspicious entries get flagged without needing a human to review every line. We added a Data Privacy and Security section that covers how data is encrypted, how users log in, and how we handle the right-to-deletion problem that comes with using a blockchain. We also expanded the Verified Financial History section to explain that the wallet ID's credibility comes from the fact that most transactions are sourced directly from third-party platforms like bank feeds and payment processors, and that any manually entered records are clearly labeled as self-reported so whoever is reviewing the wallet ID can judge for themselves how much of the data is independently verified. The Potential Risks section was updated to include actual mitigations for each risk rather than just naming them. Finally, we added the Code Implementation section to show how the key features we described throughout the paper are actually built into the smart contract.