RociFi Protocol Whitepaper

"Perhaps the largest financial value built directly on reputation is credit and uncollateralized lending. Currently, the web3 ecosystem cannot replicate simple forms of uncollateralized lending" - Vitalik Buterin

1 Intro

RociFi protocol is a DeFi primitive for permissionless under-collateralized lending. Built on Ethereum, RociFi will launch initially on Polygon. RociFi is reinventing credit to be more inclusive while increasing capital efficiency in DeFi by leveraging on-chain behavior and reputation to issue under-collateralized loans.

As the protocol grows in adoption, it will function similarly to a Layer 1 whereby DeFi and TradFi firms can build new capital markets services atop RociFi, thus creating a decentralized credit economy.

RociFi enables borrowers with clean EVM-compatible on-chain history and reputation to borrow with reduced or zero collateral while offering lenders a high, stable deposit yield. Initially, RociFi's approved assets for lending and collateral will be USDC, DAI, USDT, ETH, and WBTC.

1.1 Current issues

Current DeFi lending suffers from two problems:

- "One size fits all" model
- Lack of "smart" loan parameters

1.1.1 One size fits all

The "one size fits all" model forces DeFi users into the same terms, with no option for customization based upon specific needs. RociFi's protocol design allows unique customization and flexibility of loan terms for borrowers and risk preferences for lenders.

At launch, 30-day fixed rate loans with a 5-day grace period (no instant liquidation) for under-collateralized borrowers will be offered.

1.1.2 Lack of smart loan parameters

The lack of “smart” loan parameters unfairly penalizes creditworthy borrowers with the same loan terms as less creditworthy ones, leading to decreased capital efficiency and revenue. RociFi’s credit risk scoring enables the facilitation of under-collateralized and zero-collateral loans, thus enhancing capital efficiency and revenue.

2 Key architectural components

2.1 Overview

The key components of the RociFi architecture define a 24/7/365 credit market that is interconnected with other DeFi systems, providing ultimate capital efficiency and liquidity for users. For example, via the RociFi smart contract architecture, lenders are able to invest their capital without lock ups despite fixed term loans issued.

The RociFi protocol architecture relies on a set of loosely coupled internal components with the minimum of hard dependencies between them. This allows for RociFi’s various smart contracts to connect with those of external DeFi protocols and primitives without being reliant upon any one of them.

Below is a taxonomy reference guide to the primary tokens and actors in the ecosystem.

TOKEN	**AKA**	**FUNCTIONS**
Bond	ERC-1155 smart contract	Bonds are minted and staked when a new loan is created and unstaked and burned when the loan is repaid.
Debt token	rDAI, rUSDC, rUSDT	An ERC-20 token that represents the underlying asset/risk class e.g rDAI3 corresponds to DAI asset pool and risk score 3 debt token
ACTOR	**AKA**	**FUNCTIONS**
Borrower	n/a	Loan request triggers bond token request
Lender	Depositor/Staker	Fulfills bond request initiated by borrower
DAO Contributor	vexROCI token investor	Stakes vexROCI token for governance participation

There are three main smart contracts that comprise the RociFi lending architecture:

- **Payment contract:** Defines the details of the loan, including the loan-to-value ratio, the interest rate, what the penalties are, what collateral is required and how that collateral may be liquidated.
- **Bond contract:** This is the tokenized debt coupon that allows lenders to finance loans (by buying the bonds), collect on payments (by exchanging in payment contract) and accrue interest (by staking bonds).
- **Investor contract:** This is the abstraction contract that acts like an investment bank buying various tranches of bonds defined within its parameters and managing these loans. It splits its profits with its investors (the "lenders").

Note: *Any of these smart contract components can be upgraded or replaced without affecting other parts of the system.*

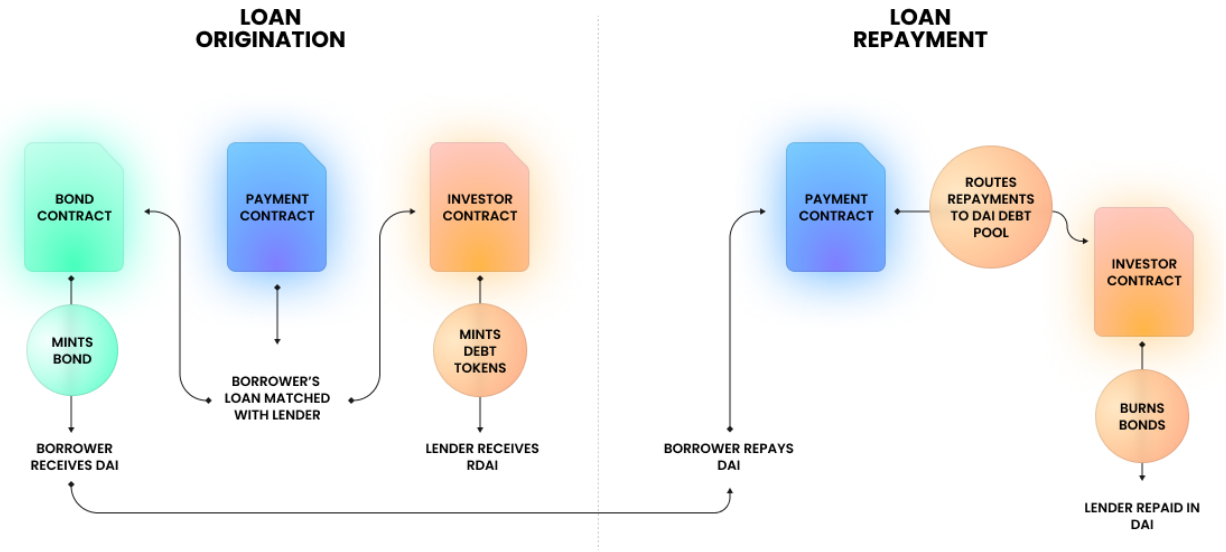

Figure 1. RociFi protocol architecture

RociFi contracts provide infinite customization possibilities to the structure of loan products to fit individual needs of DeFi users, ranging from a 5-day loan with a fixed rate to a 90-day loan with a 15-day repayment interval that borrows against your NFT collectibles and LP shares in a Uniswap liquidity pool, for example.

2.2 Actors

Borrowers - participants who want to raise money against a certain amount of collateral.

Lenders - participants who want to lend money to receive returns by holding ERC-20 debt tokens.

Liquidators - participants who earn money by liquidating delinquent loans and receive collateral in return.

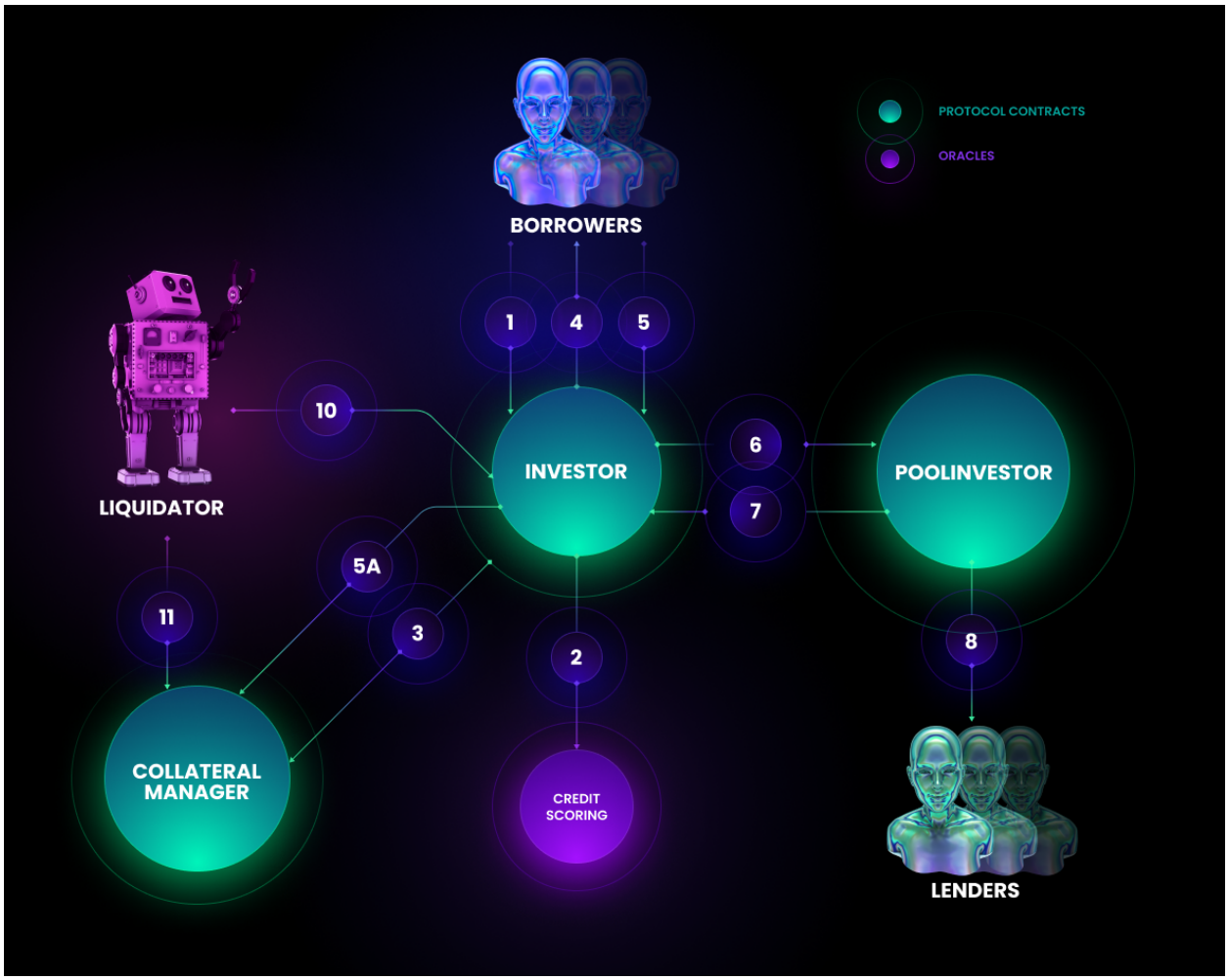

Figure 2. RociFi Smart contract architecture

Lifecycle of fulfilling a DAI loan with a Borrower credit rated 3 (rDAI3):

1. The borrower requests a loan and **Bonds** are created by the **Payment contract**
2. The **Investor contract** buys these bonds
3. Upon purchase of borrower's bonds, the Investor releases DAI to the borrower
4. Loan repayments are made to the **Payments contract**, in DAI, and the **Investor** collects DAI payments from there and distributes it to Lenders
5. Investor burns bonds as payments are made
6. *Concurrently,* **Lender** deposits DAI into **rDAI3 Debt Pool**
7. Lender receives rDAI3 **Debt Tokens**
8. As Borrower repay DAI loan to investor contract, it burns bonds
9. Lender receives interest payments in DAI via Payments contract
10. **Collateral Manager** invests the collateral externally for yield
11. **Liquidators** search for defaulted or distressed loans in the Collateral contract to buy

2.3 Components

2.3.1 Bonds

RociFi lending platform uses the concept of non-fungible loans on the ERC-1155 standard known as **Bond tokens**.

In a real off-chain world, giving loans in cash is scalable only up to a point. As the number of borrowers and lenders grow, and parameters of the loan become more complex (think maturity date, borrower risk profile, collateral type, etc), the bond issuer needs a sophisticated balance sheet.

Similarly, DeFi fungible debt tokens need a complex balance logic. This prevents mixing loans of the different maturity dates or risks in the same pool or makes implementation utterly convoluted at the very least as every loan should be created and maintained as a data structure on the contract.

RociFi protocol solves this problem by introducing the **Bond token**. Bond is compatible with the ERC-1155 multi-token standard. It allows the minting of multiple non-fungible assets (NFTs) from a single contract which contains a summary (the metadata) of the loan information.

In the traditional DeFi lending world, protocols operate with fungible loans which usually are represented by the platform token (e.g. aToken for Aave, cToken for Compound), commonly known as Liquidity Provider, or LP tokens. These LP tokens represent a 1:1 value to the underlying asset but don't do not carry any specific information about the loan, thus are limited in functionality.

RociFi replaces these LP tokens with **Debt tokens** which enable the creation of smart loans and the co-mingling of loans into bundles based on their credit risk rating. We will go into Debt tokens in more detail in Section 2.4.1.

Bond working example

Bonds can be owned by any address, transferred and traded. Bonds are minted and burned by the RociFi protocol.

As per their off-chain counterparts, RociFi Bonds essentially are debt obligations, where 1 debt token represents 1 of the smallest denominations of the principal currency. For example, a bond might represent information like this:

Alice's Bond
Minted 09/23/2021
Maturity date 09/23/2022
Loan of 10,000.00 USDC
10% Annual interest rate
Payments must be made monthly

Alice's loan is for 10,000 USDC. And USDC is an ERC-20 token with 6 decimal places. Meaning that 10,000,000,000 of these bonds are minted, one for each of the lowest denominations of USDC (0.000001). Alice can then sell these bonds to a protocol at a value of 0.000001 USDC each to raise a total of 10,000 USDC.

Bonds do not ever change face value, instead, they function like a zero-coupon bond and change in quantity. If a protocol bought 100,000 of Alice's bonds then that 10% interest rate would mint 10,000 new bonds after a year, representing the accrual.

Bonds are staked in the Bonds contract (ERC-1155 smart contract). While staked, they accrue compounding interest as defined by the interest rate. While bonds are an NFT memorial of the loan terms, *the interest is always repaid in the borrowed asset* (in the above instance USDC) not in any other currency or asset.

Lifecycle of the bond:

1. Alice requested the loan of 10,000 USDC for 12 months with 10% APR.
2. After checking collateral value, credit score and assigning LTV, the protocol mints 10,000,000,000 bonds to Alice
3. Current Alice bond price is 1,000,000 bonds per 1 USDC
4. Protocol sends 10,000 USDC to Alice in exchange for all her bonds
5. In 12 months Alice repays 11,000 USDC
6. Protocol burns all of Alice's 11,000,000,000 bonds
7. If Alice should miss the repayment and hence, the loan becomes delinquent, external liquidators can buy and sell her collateral at a discount

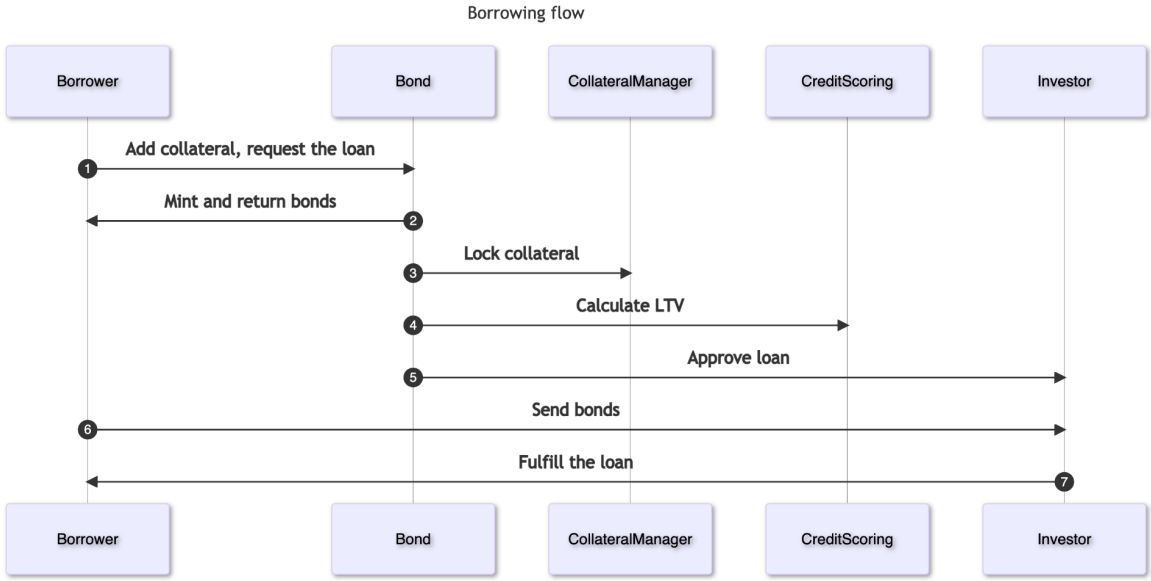

Figure 3. Borrowing flow

2.3.2 Lending pools

Lending pools are the interface for lenders to deposit and withdraw their capital.

Each pool contains the underlying asset/risk class ERC-20 debt token and corresponding asset token. For 10 risk classes and 3 assets there will be 30 pools. For example, rDAI3 corresponds to DAI asset pool and risk score 3 debt token.

At launch, lending pools will be bucketed based on risk as to not dilute liquidity, e.g. credit scores 1-3 (low risk pool), 4-6 (mid risk pool), and 7-10 (high risk pool).

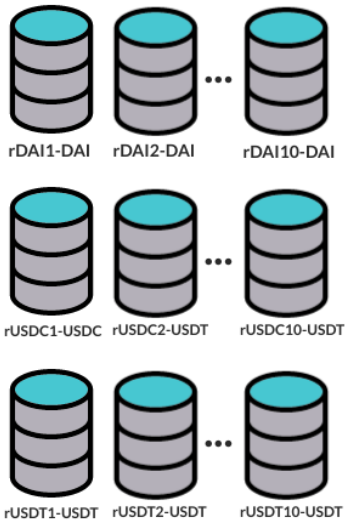

Figure 4. Lending pools

RociFi debt tokens can be redeemed by the protocol (if liquidity exists) or exchanged on external AMMs. The latter allows lenders to sell their ERC-20 debt tokens (e.g. rDAI) and withdraw their capital at any time, even if insufficient liquidity exists in the lending pools.

Note: *The platform will cap lender's deposits at launch in order to mitigate risk.*

Figure 5 represents a simplified flow diagram for lending pool selection based on lender's risk preference and borrowers' bonds routed to commensurately rated pools.

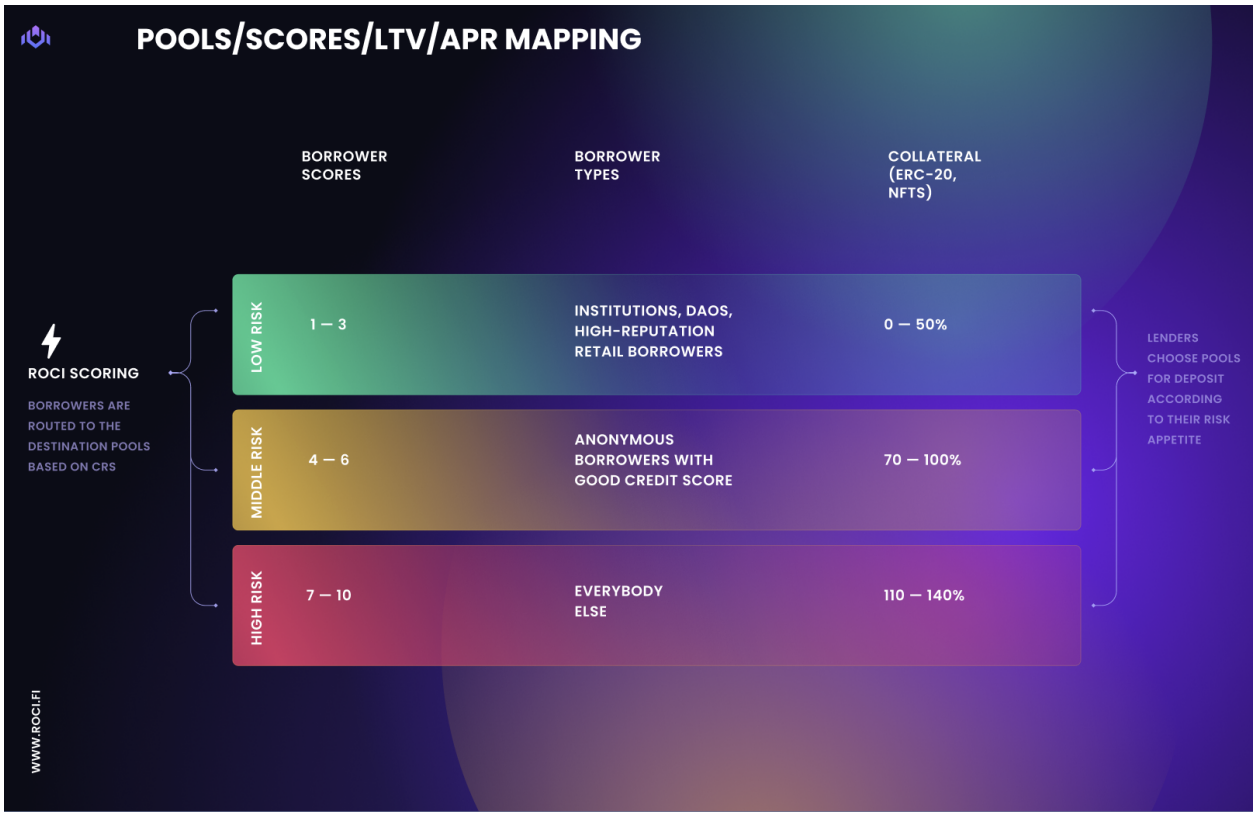

Figure 5. Lending pool selection flow

2.3.3 Investor

The **Investor smart contract** provides the heartbeat for the system. Its purpose is twofold:

- Ensure proper liquidity for the asset and its debt token in the corresponding lending pool
- Orchestrate payout to borrowers from the lending pool and route repayments from the borrowers

Once a loan is approved, the Investor contract buys the borrower's bonds and releases the asset being borrowed. During repayment, the Investor contract gradually burns the bonds as the borrowed asset is repaid.

The Investor manages protocol liquidity while it also fulfills the loans and accepts repayments. Eventually, maintenance and customization of the Investor contract and all smart contracts will be managed by the RociFi DAO.

There are as many investment contracts as there are lending pools (e.g. for 3 assets and 10 risk classes there will be 30 contracts).

The **PoolInvestor contract** is a child contract of the Investor parent contract. As such PoolInvestor decides which loans are to be funded and matches them with the commensurate asset and risk pool.

2.3.4 Collateral manager

The Collateral Manager component is invoked when collateral needs to be transferred from the borrower to the protocol before issuing the loan. It locks and escrows collateral until the loan is either repaid or liquidated.

In the first version, RociFi's Collateral Manager (CM) accepts only ERC-20 as collateral.

2.3.6 Liquidation engine

This component is responsible for the liquidation mechanics. It allows for 3rd-party liquidations via API or liquidation by the protocol. It can be upgraded into a full-fledged loan marketplace, where interested parties trade loans and collateral.

Liquidation occurs when a borrower's loan is overdue (under-collateralized) or its LTV is higher than a predetermined threshold (over-collateralized).

The first and simplest liquidation strategy is liquidation by the protocol. In this case collateral from the insolvent borrower is sold for the underlying asset and acquired debt tokens are burned.

The second strategy is liquidation by a 3rd party. In this case, delinquent loans can be repaid by anyone, not just a borrower. So, for example, if Bob's loan is overdue, Alice can send a request to liquidate it. After repaying the loan at a discount dependent on how far out from maturity the loan is (the further out the greater the discount), Alice can grab Bob's collateral.

2.4 Tokens

RociFi smart contracts operate with the following tokens:

- **Debt token** is a representation of the lender's share in the lending pool. It is an ERC-20 token with elastic supply, minted and burned on demand. It is managed initially by the protocol admins and in future by the DAO.

- **Bond token** is a representation of the loan. It is an ERC-1155 non-fungible token that corresponds to a specific loan and is parameterized by principal amount, collateral type and amount and the loan maturity date. It is managed by the protocol admins or DAO.

- **Asset token** is a token used for borrowing. It is an ERC-20 stablecoin (DAI, USDC, USDT and alike).

2.4.1 Debt tokens

Denominated as rDAI, rUSDC, rUSDT, **Debt tokens** are minted and issued to a lender whenever they make a deposit. Simply put, RociFi takes a lender's DAI into the pool and gives them rDAI in return.

These tokens can be thought allegorically as shares (debt tokens) of a fund (investor) that manages the many different loans (bonds).

Debt tokens are minted and burned by the protocol as per the borrowing parameters set out in the Payment contract. The debt tokens can be safely stored, transferred or traded and are managed initially by the protocol admins and, in future, by the DAO.

As mentioned earlier, the innovation of debt tokens over their LP token counterpart on other DeFi platforms is that they enable the co-mingling of loans of the same risk profile, i.e. loans with different maturities, interest rates, and collateral *but* with the same credit risk score.

This is done by pairing RociFi's credit scoring system with the Investor (parent) smart contract to create numerous (child) **PoolInvestor contracts** that buy up and invest in bonds across assets and risk profiles.

The PoolInvestor contracts issue the ERC-20 debt tokens to lenders which entitles them to their portion of the lending pool rewards.

Lenders can then deposit to a particular lending pool rated Low, Mid, High, without having to sift through and inspect many individual loans that fit their investing style.

Note: *The debt token holder will always receive payments in the underlying asset, e.g. DAI, USDC, USDT etc.*

Debt tokens are swapped within pools or exchanged via external AMM/DEX. For example, rDAI3 token would correspond to the pool with DAI as the main asset and the risk class 3, which suggests returns up to 10% APY.

The protocol tokenizes its contract with the lender via debt tokens, hence it creates a new kind of liquid asset: a bond granting its holder the right to withdraw the deposit plus interest when there's enough liquidity in the protocol.

This asset can also be traded on markets that are external to the protocol, e.g. DEXes like Curve, Uniswap and Balancer, which allows lenders to exit their position by trading their debt tokens on the open market even if there's a lack of liquidity on the protocol. The discount for exiting in this situation would be determined by the market, which is a feature, not a bug.

Note: *the aforementioned describes a decentralized bond market, which is revolutionary in DeFi.*

The discount is a function of current market conditions and debt token holder's perceived default risk of the underlying bonds backing the lending pool. By not dictating or estimating this discount inside the protocol or making the protocol depend on the correctness of this estimation will allow for better stability of the system under changing market conditions and less exploitability via market manipulation.

Lifecycle of the debt token:

- Alice has a credit score of 3 and wants to borrow 10,000 DAI from the risk class 3 pool at 10% APY by selling bonds to the Investor 3 contract (see "Bond" section for more details)
- If this loan is approved, the Investor contract needs 10,000 DAI to fulfill the request
- Investor contract takes 10,000 DAI from the corresponding pool and mints 10,000 rDAI debt tokens
- Alice gets 10,000 DAI, and her lender receives 10,000 rDAI

- On maturity date, Alice repays 11,000 DAI back to the Investor contract
- rDAI tokens appreciate in price, and all lenders who hold debt tokens profit

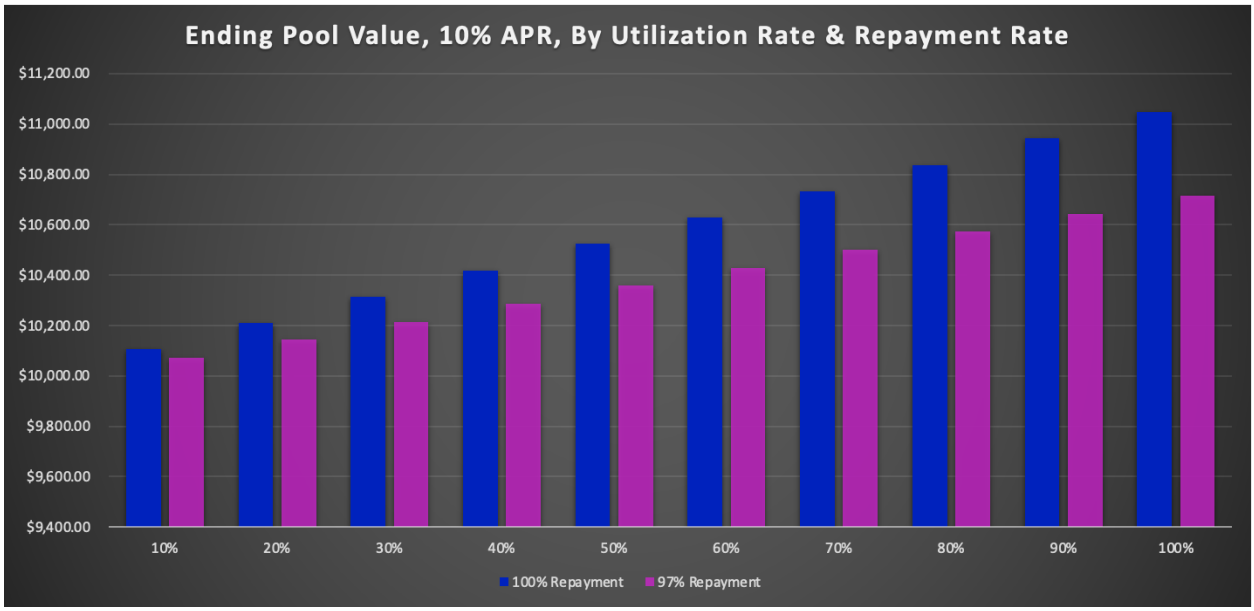

Figure 6. Hypothetical lending pool returns based on utilization and repayment rate

2.4.2 Lending pool formula

Each lending pool – Low, Mid, and High risk – implements a distinct debt token. Upon depositing payment tokens (where DT = debt tokens):

```
DT_TO_MINT = DT_TOTAL_SUPPLY * DEPOSIT_AMOUNT / CURRENT_POOL_VALUE
```

DT_TO_MINT of debt tokens are minted to the depositor, increasing the debt token total supply to:

```
DT_TOTAL_SUPPLY = DT_TOTAL_SUPPLY + DT_TO_MINT
```

Current lending pool value used in the formula includes all outstanding debt.

Holding DT_AMOUNT debt tokens entitles you to owning a share of the respective lending pool calculated as:

```
POOL_SHARE = DT_AMOUNT / DT_TOTAL_SUPPLY
```

Burning DT_AMOUNT of debt tokens allows you to withdraw

```
TOKEN_AMOUNT = CURRENT_POOL_VALUE * DT_AMOUNT / DT_TOTAL_SUPPLY
```

Payment token from the pool, if sufficient liquidity exists. If not, debt tokens can be traded via the secondary market.

Given Bonds follow a zero-coupon bond (ZCB) formula, the value of debt tokens are a function of the Bonds interest rate and utilization rate in the lending pool, and ownership percentage of the pool by each depositor.

For example, assuming the low risk lending pool value starts at 1000 USDC with 10% APR, 100% utilization, and two equally sized depositors (50%/50% ownership), the value of lending pool after 1 year:

$$Lending\ Pool_{low\ risk} Value\ =\ 1000\ USDC\ *\ 10\%\ APR\ =\ 1100\ USDC$$

Assuming depositor A and B both own 100 debt tokens each, the per debt token price equals:

$$Depositor_A\ Debt\ Token\ Value\ =\ 1100\ USDC\ *\ 50\%\ pool\ ownership\ /100\ =\ \$5.50\ USDC$$
$$Depositor_B\ Debt\ Token\ Value\ =\ 1100\ USDC\ *\ 50\%\ pool\ ownership\ /100\ =\ \$5.50\ USDC$$

3. Credit, Reputation, and Trust Verification (NFCS)

In order to borrow on RociFi, users must mint an ERC-1155 token, **NFCS** (**Non-Fungible Credit Score**). The **NFCS** proves ownership of the address bundle that the borrower wishes to be credit scored.

The NFCS operates similarly to Experian, i.e. the score belongs to the user but the user does not control it, making it integrable into other Web3 protocols.

The need for a unique, non-transferable on-chain credential that unlocks greater utility and safety across Web3 has grown in interest since Vitalik Buterin's recent post about "Soulbound" tokens. RociFi's NFCS was designed to be that and more by becoming Web3's first, verified token for credit, reputation, and trust (CRT) of a user's on-chain identity, i.e. *proof of CRT.*

3.1 Credit Scores

RociFi's credit scores are based on risk, meaning the lower the better. RociFi's credit scale is 1–10 where 1 is the lowest credit risk (best score) and 10 is the highest credit risk (worst score).

At launch, good credit scores (1–6) provide access to under-collateralized loans while riskier ones (7–10) receive over-collateralized loans at LTVs above market rates.

The protocol assesses a borrower's likelihood of defaulting on their debts by analyzing DeFi transaction history and behavior, coupled with Web3.0 reputation data points.

A lower credit score means lower default risk which enables lower collateralization ratios and borrowing costs (APR).

In general, if a user has responsibly participated in DeFi, i.e. lending/borrowing/ liquidity provisioning/DEXes, and has a solid onchain reputation, they will most likely receive a good score.

3.2 Credit Risk Management

RociFi's credit risk management is a holistic approach designed to maximize the protocol's margin of safety, i.e. bad borrowers will eventually be given loans and defaults will occur.

However, the goal is to keep defaults to small, manageable amounts so as to learn from them rather than becoming catastrophic.

The three main components analyzed during the credit scoring process are:

- **Credit risk:** the borrower is able to repay the loan, i.e. creditworthiness
- **Fraud risk**: the borrower is willing to repay the loan, i.e. trustworthiness
- **Reputation risk**: the borrower has something to lose in the event of failing to repay the loan, i.e. social recourse

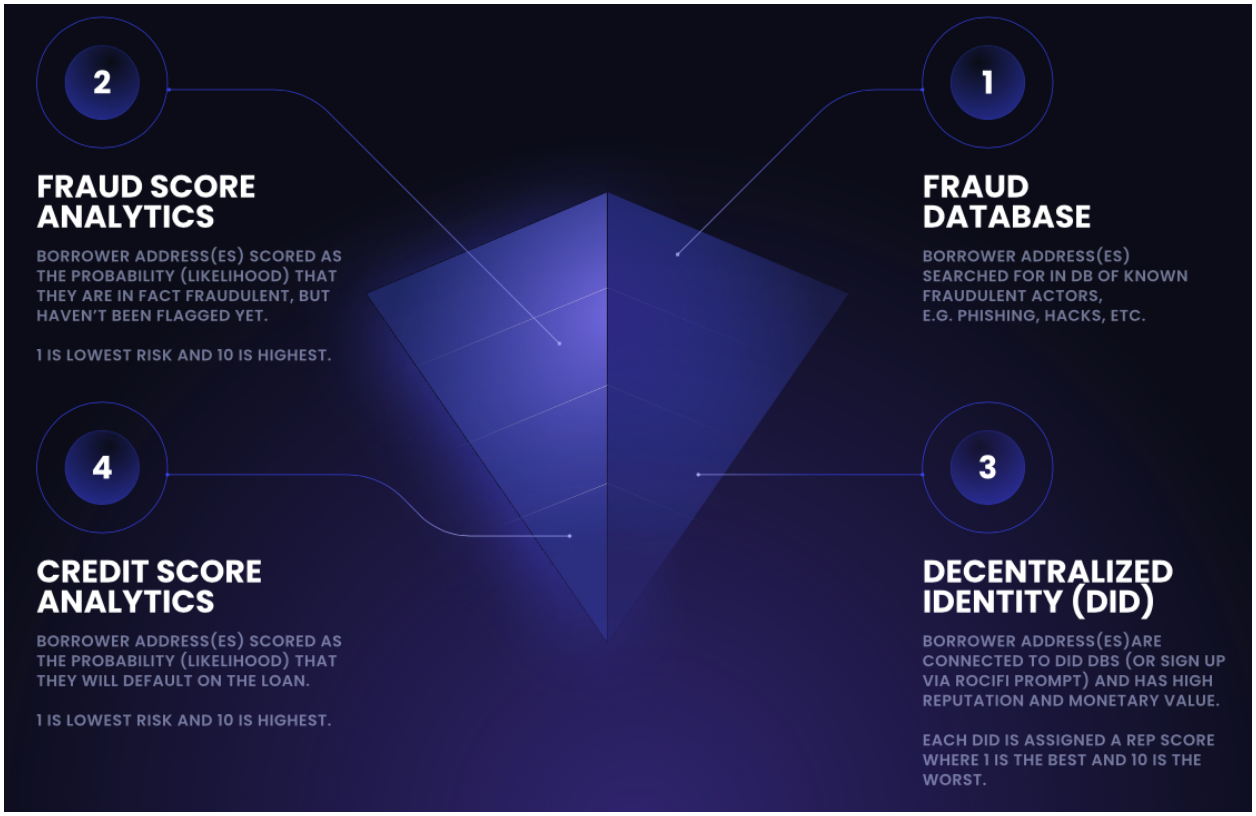

Figure 7. Credit Scoring Process Funnel

3.3 Borrower interest rate

The borrower's interest rate is a derivative of the Capital Asset Pricing Model (CAPM) to compensate lenders for the risks associated with under-collateralized lending.

$$Borrower\ Interest\ Rate = R_f + R_p + V_c$$

$$R_f = risk\ free\ rate$$

$$R_p = risk\ premium$$

$$V_c = volatility\ charge$$

Risk free rate is the interest rate charged for the same asset over a similar duration, risk free. Aave's stable yield provides a decent proxy for the time being despite limitations.

Market Risk Premium is the additional interest charged to a borrower based on an estimate of the market's desired rate of return at a defined level of risk. Specifically, if the market's desired APR at a specific level of risk is R_l and the market risk premium as R_p, then RociFi seeks to provide an estimate of R_p such that $R_l = R_f + V_c + R_p$.

In order to do this, RociFi will utilize a linear model of the following form

$$R_l = R_b + \frac{U - U^*}{U^*} R_s$$

$$R_b = R_f + V_c$$

In other words, the market risk premium ties out the base lending rate of R_b with the market's desired lending rate R_l. ***Effectively, this acts as spread on top of the base rate for which would make the loan attractive to a lender based on opportunity costs and perceived default risk.*** Readers may notice this formula is similar to AAVE's borrower interest rate (and it is), however as RociFi's base rate is dynamic, we allow for a negative adjustment applied to the rate in order to reflect the potential that RociFi's base rate is overestimating the desired rate of return by the market.

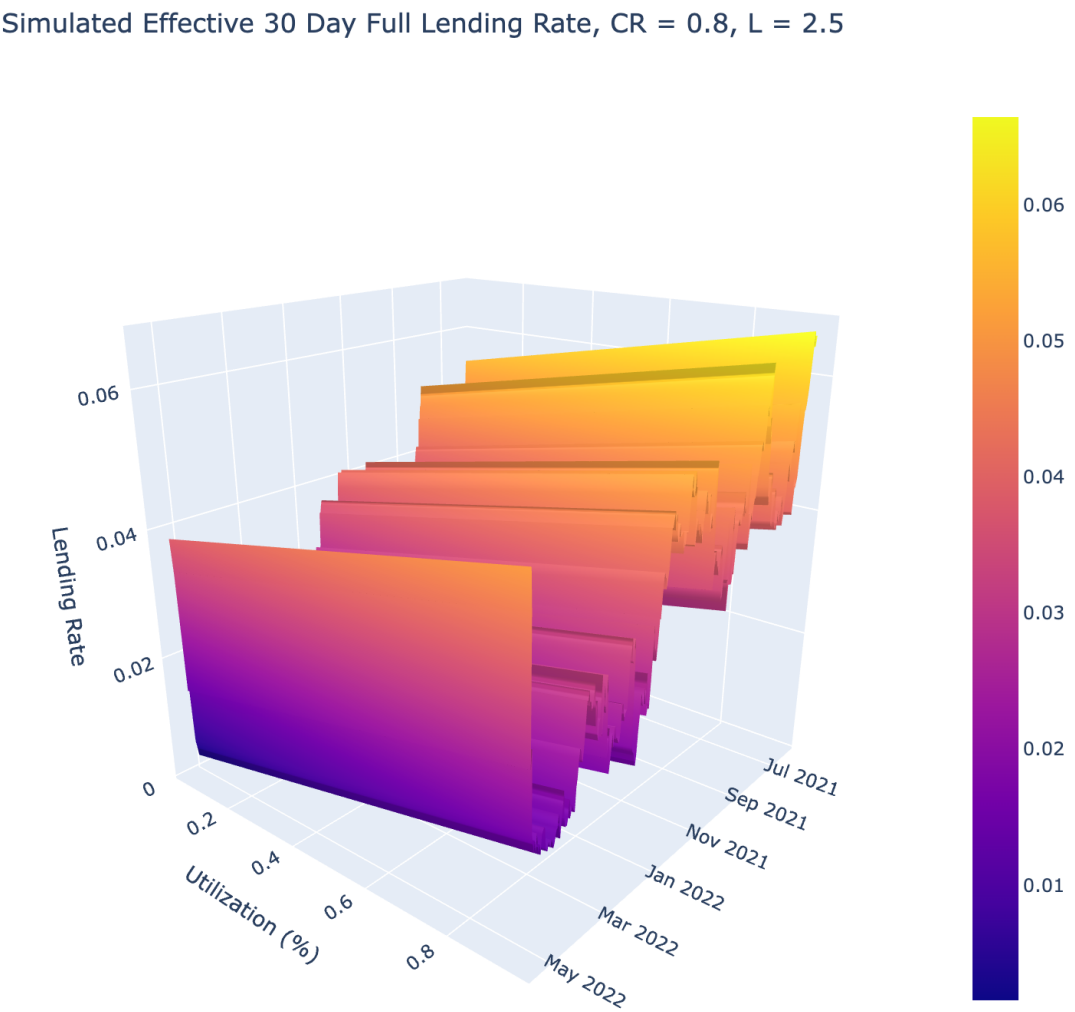

Figure 8. Effective 30 Day Full Lending Rate

The above example shows the full lending rate RociFi would charge if $U^* = 0.5$ and $R_s = 0.08$. As we can see, the full lending rate gets adjusted up as the utilization rate goes higher than optimal, and likewise lower if below U^*. This dynamic helps encourage (reduce) supply and decrease (increase) demand to keep the pool utilization optimal.

At launch, RociFi plans to provide a fixed spread applied to each risk pool's base rate. Each lending pool's risk premium will be adjusted based upon supply and demand for credit at those rates. As RociFi collects more data, we will utilize data about those utilization rates and regress this against fixed spread assessed in order to implement a dynamic Market Risk Premium Model. This will help ensure that market perceived opportunity costs and default risks are properly priced into the return for each pool.

Volatility Charge is charging the borrower the cost to hedge the value of their collateral in the event of a price drop sufficient for the borrower to put their collateral to RociFi, and forgo their credit & web3 reputation. Since this cost to hedge is being derived from implied volatility of traded options on the underlying collateral asset, we refer to this measure of risk as the "Volatility Charge".

The Volatility Charge is a function of the borrowed collateral ratio "R", LTV Threshold "L", and the annualized implied volatility of the relevant market "S". The most important concept here is "L", the LTV Threshold. "L" reflects the threshold, where if the LTV of the loan were to rise so high, the borrower would choose to walk away from the loan, i.e. Walkaway Risk.

To arrive at the effective lending rate, we can use the AAVE USDC Deposit APY as a proxy for a risk free rate, and to that add the vol charge. In practice we also add a market risk premium spread to this total rate to reflect the profit margin that ROCI lenders would like to earn in light of opportunity costs and perceived default risk.

Below we show what the effective lending rate would have been, assuming an 80% Collateral Ratio, with no risk premium added.

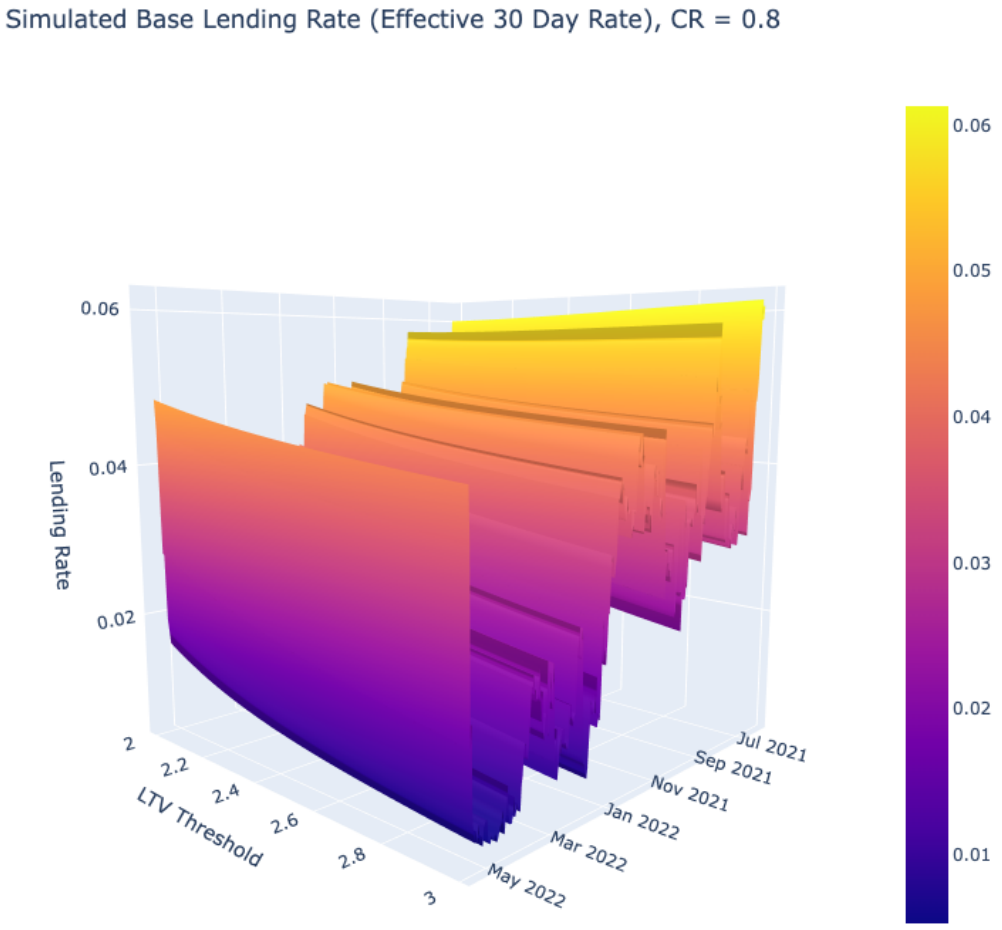

Figure 9. Effective 30 Day Base Lending Rate (No Market Risk Premium Added)

On the x axis we show the timestamp, the y axis represents the LTV Threshold "L", and the z axis represents the effective lending rate in decimal (0.04 == 4%). As we can see the effective lending rate will increase during times of market stress such as Summer '21, Winter '22, and Spring '22. Having our interest rates include risk based on market volatility ensures that lending rates offered remain consistent with risks present in the broader market.

The exact computation of the Volatility Charge is the product of careful mathematics based on the assumptions made about borrower behavior. Below we are able to provide the community with an approximation based on a regression set up with reasonable assumptions for "R", "L", "S".

$$V_C \approx max(0.21803969 - 0.57549647 * R - 0.09633753 * L + 0.85737744 * S, 0)$$

$$L = borrower's\ LTV\ threshold$$
$$R = borrower's\ Collateral\ Ratio$$
$$S = annualized\ implied\ volatility\ of\ the\ market$$

This linear approximation will be reasonable, within 3.5% APR of the actual Volatility Charge on average, for "L" 200% to 300%, for R between 60% to 100%, and for S consistent with implied volatility observed in the market over the last 2 years for ETH.

3.4 Social Recourse

Over decades, village co-op-style banking has proven remarkably resilient with fewer defaults and overindebtedness partly because people know each other and effectively lend at more personalized rates. In this setting, the baker and the farmer earn their reputation over years of hard work supplying food, and, if they defaulted, their reputation would be severely damaged as word would quickly spread.

This is a form of **social recourse**.

RociFi is building the same dynamic in Web3 whereby lending pools operate as the village bank and **NFCS** operates as word of mouth reputation and credit score.

Social recourse rewards or removes actors from the global village based upon their actions towards the common good. At its core, social recourse is designed to incentivize positive, cooperative behavior while strongly disincentivizing negative ones such as loan defaults.

Upon being eligible for an under-collateralized loan, the user must agree to the terms and conditions that include exposure of their provided information in event of a default. Upon non-repayment, RociFi will disseminate the defaulter's information across social media and community channels.

Additionally, the broader RociFi community will be incentivized to share the defaulter's information across their network channels as well. The defaulter will be banned from RociFi forever while the loan remains delinquent.

The end goal is to create a firewall to remove bad actors from the RociFi ecosystem via onchain reputation-based recourse. The thesis behind this is – "do you believe

your onchain identity and reputation will be more or less valuable in 5 years?" If the former, you'll do whatever it takes to guard and grow it with good behavior.

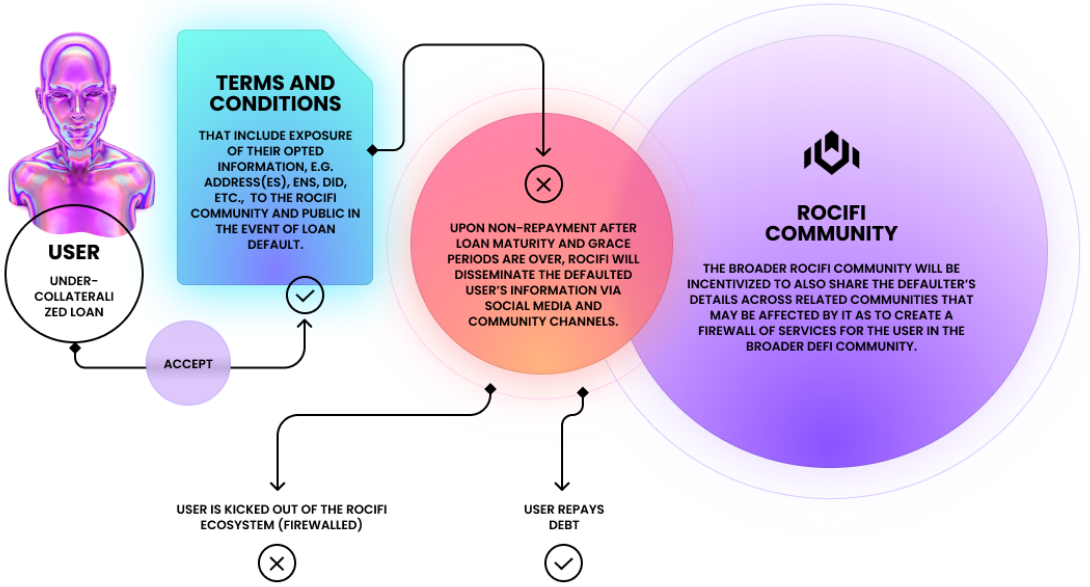

Figure 10. Social recourse process flow

3.5 NFCS minting flow

- Alice enters RociFi app and connects her wallet (MetaMask in this example)
- She adds multiple addresses to give her the best credit score
- Signs a small transaction to prove she's the owner of said addresses
- Mints her NFCS token and generates her credit score
- Alice's NFCS is stored in her MetaMask wallet for later use
- Alice connects her existing NFCS to RociFi app for new loan requests

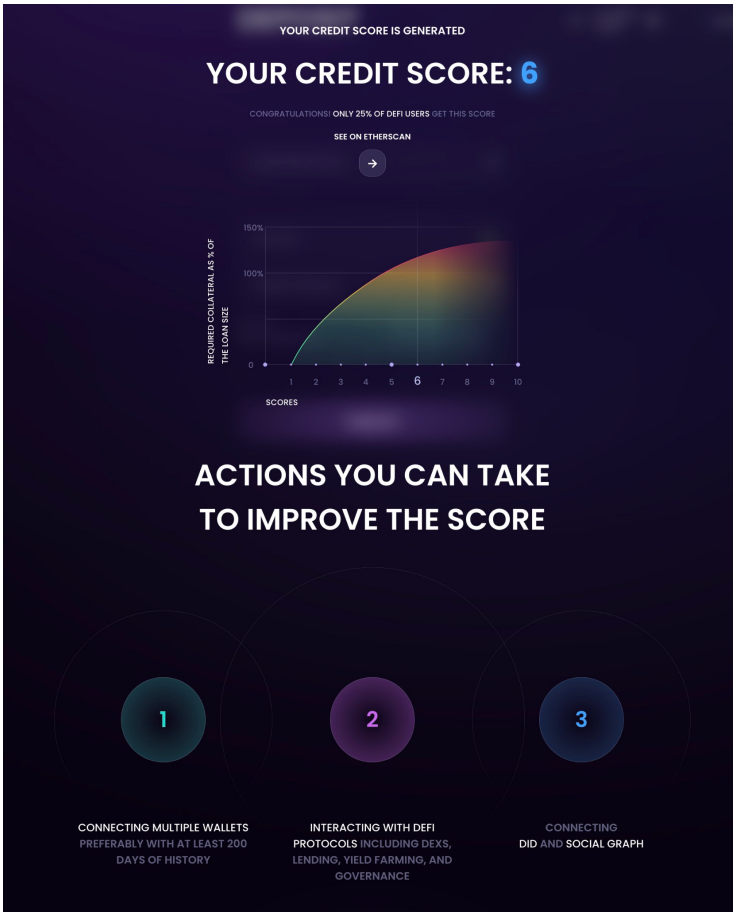

Figure 11. NFCS minting process

3.6 Credit risk oracle

The credit risk oracle component connects to the off-chain credit scoring analytics and price feeds to generate a loan's LTV and user's credit score. This logic is abstracted away from the Investor and Bond. The interface is generic, so any scoring logic can be plugged in (as we decentralize scoring inputs from DAO members).

The off-chain credit scoring analytics works by plugging on-chain data collected about a user into a ML model that returns a probability that a particular user, if given access to a loan, would have their loan result in default. This probability is then rescaled into our credit scores.

Our model has been trained using millions of data points collected from various DeFi protocols. The performance shown below is highly competitive as compared to traditional credit risk models.

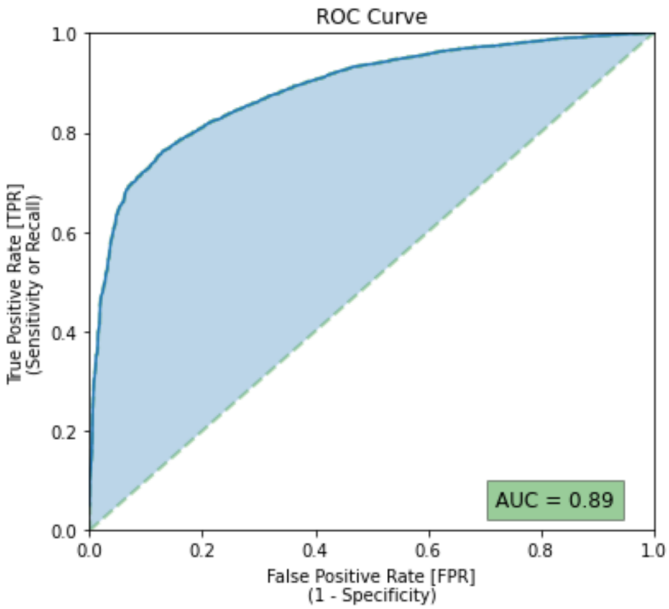

Figure 12. Credit model ROC curve and AUC

Above is the ROC Curve generated by RociFi's current credit risk model. An ROC curve looks at a model's True Positive Rate (Sensitivity of Recall) vs the model's False positive rate (1 - Specificity). For our model, AUC equals 0.89, which is considered strong by conventional metrics.

Similar to our credit risk model, RociFi is also careful to consider another source of credit risk: Fraud. There are multiple possible sources of fraud including exploits, scams, and phishing. Addresses associated with these types of fraud activities are considered by RociFi to be more likely to simply run away with disbursed funds if given access to credit. For that reason, RociFi has invested a lot of time and resources into building out our off-chain fraud analytics based on modern approaches that combine Graph Theory and Machine Learning. To date, RociFi has built a Graph Network of addresses exceeding 96GB in size and growing by the day. From this graph we are able to mine a set of features about a particular user's address, and run it through our Fraud Classifier. On out-of-sample data, we are able to achieve Recall of roughly 96%.

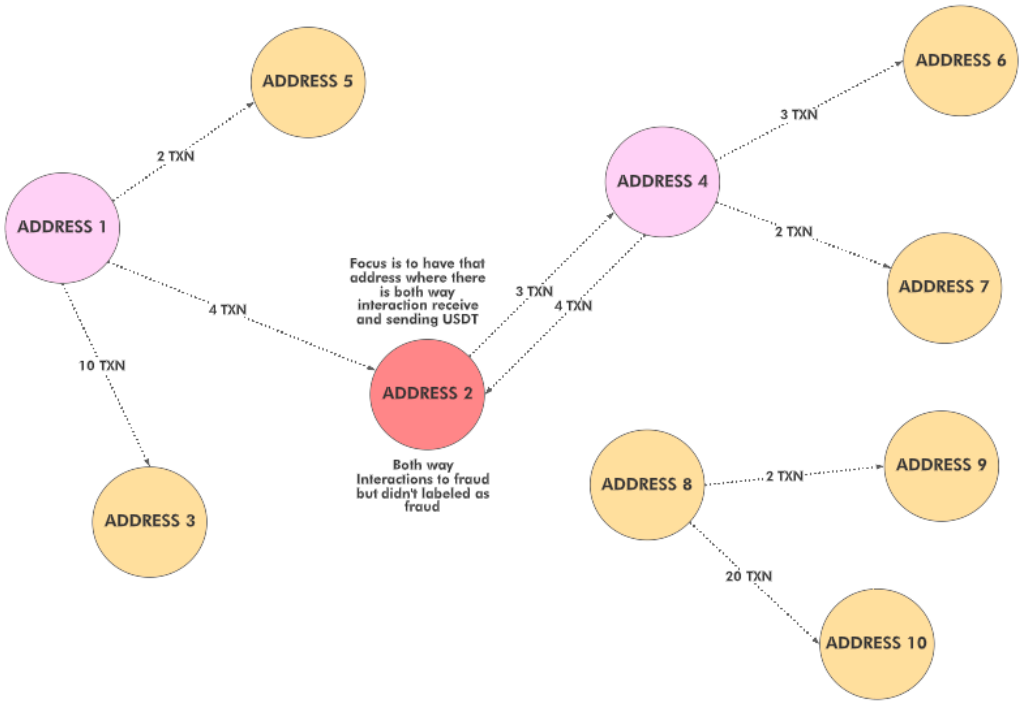

Figure 13. Fraudulent address identification

Figure 11 demonstrates how we attempt to identify fraud in our network graph. Here the red node “Address 2” represents a fraudulent address. We see that Address 1 has sent 4 TXNs to “Address 2” and Address 4 has sent 4 TXNs to “Address 2”, while “Address 2” has sent 3 TXNs to Address 2. Based on the interactions that “Address 2” has had with its neighbors (1 and 4) and those neighbors interactions with their neighbors, using data from these transactions we are able to induce the probability that “Address 2” is fraud, along with addresses 1 & 4. These probabilities are then rescaled to a Fraud Score that runs from 1-10. Similar to our credit score, a score of “1” represents the lowest risk of being Fraud and a score of 10 represents the highest probability of being fraud.

In the event we do not have data for a particular address in our fraud graph, RociFi has also invested substantial time and resources into building a transactional level model which attempts to learn a decision boundary for fraudulent transactions directly from the transactional level data. This model has been trained on millions of transactions mined from hundreds of thousands of active & known fraudulent addresses and as a result is able to achieve Recall of roughly 90%. Similar to our Graph based classifier, this model will produce a probability of a transaction being fraudulent. We scale this from 1-10 to get a Fraud Score.

In the worst case scenario, if there is no available data for an address in our Fraud DB, and if insufficient transaction level information, a Fraud Score of 10 is assigned. This is done out of an abundance of caution.

3.7 On-chain reputation

The NFCS acts as an additional measure for DeFi users to build their on-chain reputation and onchain decentralized IDs (DiD); especially when RociFi's NFCS is linked to user's ENS accounts, social graph, NFT holdings, and more.

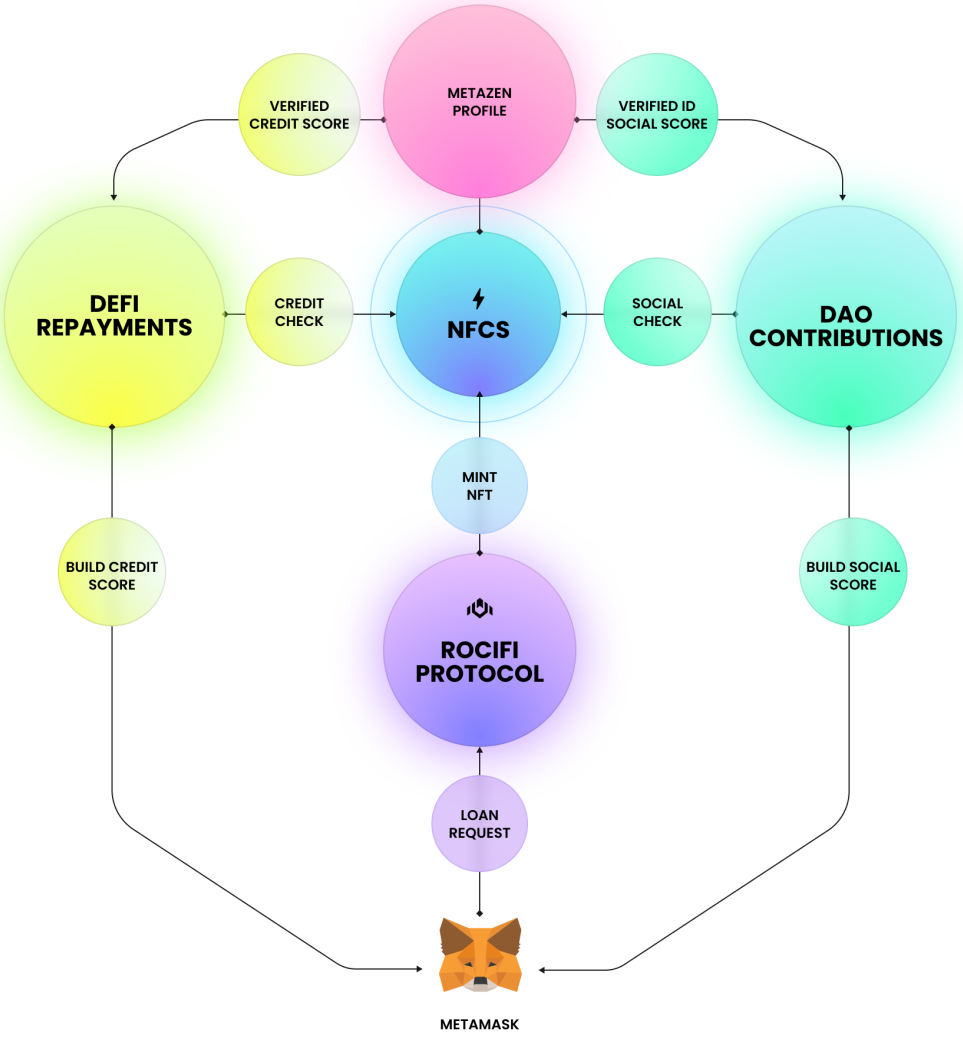

Figure 14. NFCS reputation flow

- Jane requests a loan from RociFi via her MetaMask wallet
- RociFi mints NFCS based on Jane's MetaMask transactions in DeFi and other Web3 Dapps
- NFCS carries credit and social checks on Jane's MetaMask and Web3 profile
- The NFCS contributes as verified badge of Jane's decentralized ID (metazenship) hosted on a DID protocol such as BrightID or SelfID
- If Jane's NFCS is of high quality the value flows back to her MetaMask wallet via lower borrowing rates in DeFi, more lending opportunities and higher reputation or voting power in DAO governance
- Jane's DAO contributor proposals are approved due to NFCS reputation

4 RociFi ecosystem

4.1 ROCI token

ROCI is the governance token of the RociFi ecosystem and is the staking asset used to backstop the protocol's solvency. ROCI holders who stake their tokens into vexROCI will be entitled to a share of protocol revenue, governance voting, and early access to new products. The total ROCI supply will be fixed at 1 billion.

4.2 vexROCI

RociFi utilizes a staking program using the vexROCI token (vote-escrowed ROCI). This is a non-transferable asset that accrues protocol revenue and encourages long-term stakeholders by rewarding vexROCI holders with more governance rights the longer they stake.

Assuming four year lockup period:

1 ROCI = 1 vexROCI

To earn vexROCI, a user stakes their ROCI between one week to four years and receives vexROCI in return. The process cannot be reversed, meaning that once you stake ROCI for vexROCI, the holder will not have access to their underlying ROCI tokens until the lockup period ends.

To incentivize longer lockups, the amount of vexROCI received is proportional to how long the user locks up their ROCI.

ROCI lockup term	vexROCI reward
4 year	1 vexROCI
2 year	0.5 vexROCI
1 year	0.25 vexROCI
1 month	0.02 vexROCI

Note: *vexROCI holders don't necessarily have to be lenders on the platform.*

vexROCI holders are entitled to:

- Governance rights
- 15% of protocol revenue
- Boosted ROCI rewards via staking incentive program

Four components of vexROCI:

- **Governance:** Once full DAO governance is implemented, 1 vexROCI = 1 governance vote. Only vexROCI holders can participate in governance; including decisions on asset listing, risk parameters, treasury spending and more.

- **Fees:** vexROCI holders will receive a share of protocol revenue. Similar to SushiSwap's SushiBar contract, this will be done by using the revenue to buy ROCI on the open market and adding it to the vexROCI pool. Initially, this yield will be supplemented with our staking rewards program at the beginning of the protocol.

- **Withdrawals:** A withdrawal fee of 1% will be applied to vexROCI stakers when they unstake their capital; taken in ROCI. The proceeds will be re-allocated to the vexROCI pool.

- **Backstop:** vexROCI holders will backstop protocol solvency in the event of a capital shortfall from defaults in our lending pools.

Note: *In v1 launch, vexROCI holders will be responsible for backstopping lender pools. However, in v2 lenders will bear any losses as the protocol rolls out risk management tools, e.g. credit default swaps, tranches, and insurance.*

4.3 DAO

Once the platform is launched, RociFi will implement a strategy and timeline for turning control of the protocol over to the community. Once implemented, protocol decision-making will be governed solely by DAO voting, which is limited to vexROCI holders.

Additionally, the DAO will implement a plan for decentralizing the credit risk scoring process so that the final score is a result of multiple, trusted credit scoring nodes from around the world. It's possible to envision an armada of credit analysts that are both approved by the community and holders of vexROCI, managing the credit scoring oracle process.

5. Risks and mitigation

Within DeFi, lending protocols have four main risk factors: security, governance, oracle, and market. We will walk through each as it relates to RociFi.

5.1 Security risks

Security risk concerns the correct execution of smart contract code that stores supplied assets, manages borrowed assets, and liquidates bad loans.

Such risks are assessed by cybersecurity code auditors who focus on ensuring that the implementation of the contract exactly matches its high-level specifications.

RociFi protocol code is audited by two external auditors, Chainsulting and Certik.

Every asset-storing contract is equipped with circuit breakers which allow protocol admin or DAO to pause it and disable any money transfers. There is a global borrowing limit across assets which caps the amount of borrowing per day, as well as per user on RociFi.

5.1.1 Bonds

Bond is an ERC-1155 transferable token that can be swapped via Investor contract to an ERC-20 token. Every valid bond is a claim for ERC-20 tokens lent by the protocol and every bond is customized to the lender's terms of his ERC-20 lent to the protocol.

However, if an attacker steals the bonds from the borrower or gets control over the Bond contract, it can exploit the protocol.

To mitigate this attack vector we have abstracted the code of the Bond contract away from any one owner. In other words, bonds are neither held by the lender or any parent smart contract.

5.1.2 Investor

Investor contracts fulfill loans by buying the borrower's bonds and releasing the asset being borrowed. This rich functionality can make them a target for exploits. Attackers will try to trick the Investor into paying non-existing loans, forging bonds or repaying less than the principal sum.

To mitigate this, the Investor contract is audited and closely monitored. Furthermore, the Investor won't keep any funds in the contract.

Additionally, in future versions, the protocol could tap into Ethereum layer 2 solutions that offer privacy such as ZK-Roll-ups or Aztec to obfuscate trades within the pools, i.e. "dark pools".

5.1.3 Collateral Manager

Collateral Manager contacts have access to significant funds as they handle borrower collateral and invest it in various platforms (a feature in future versions) to earn a yield for the borrower, thus reducing their borrow rate.

The Collateral Manager releases funds and transfers them to the borrower when the loan is repaid.

This rich functionality makes it a target for various exploits. Attackers can trick it into releasing collateral regardless of loan status, forge ownership, or use vulnerabilities from other platforms.

To mitigate this, the contract is audited and closely monitored. The Collateral Manager allows transfers only within the parameters of 'amount' and 'time' as set in the smart contract, so in the worst-case scenario losses will be limited.

5.2 Governance risk

Governance risk deals with management-related issues such as administrator mismanagement, poor voter participation, and concentration of voting power.

To mitigate this, RociFi governance will have strict security procedures around key ownership. Every contract with admin access will use multisig. This means that no one person can access smart contracts and make changes to them, they can only be accessed in consensus with the other 'key holders'.

After Mainnet launch, a clear strategy and timeline for moving to the full DAO ownership will be implemented in order to make the protocol decentralized and permissionless.

The end-goal for DAO governance is for users to become the ultimate owners and managers of the platform. We believe the best way to do this is by approaching Web3 like a 'global village' where the following stakeholders are shared 'property owners' of the RociFi ecosystem:

- Lenders
- Borrowers
- Stakers
- DAO contributors

Together they will be responsible for keeping credit and social checks in balance.

5.3 Market risk

5.3.1 Oracles and liquidations

Manipulation of the off-chain credit score API or on-chain price feeds can force the protocol to liquidate loans regardless of status, causing a loss of customer funds. Or, attackers may create scam loans with a fraudulent credit score.

RociFi uses Chainlink for its price feeds, which serve as an industry standard and have a variety of security specifications.

In case of an oracle exploit, the protocol admin will use a circuit breaker to prevent serious losses. Limits per borrower also ensures several malicious borrowers won't affect liquidity in a large way.