What is Moonwell?

Moonwell is an open lending & borrowing DeFi protocol built on Moonbeam and Moonriver.

Moonwell Apollo, Moonwell's first deployment, has been deployed on Kusama's **Moonriver** network, the sister parachain of Polkadot's **Moonbeam**. Moonriver is where new products are expected to be incubated and developed prior to being deployed on Moonbeam.

Moonwell Artemis, Moonwell's second deployment, is now live on Polkadot's **Moonbeam** network!

> Join the social channels to freely interact with the community, or if there are any additional questions.

Moonbeam & Moonriver

Why Polkadot & Kusama?

Polkadot is a Layer 0 blockchain that allows other blockchains (parachains) to seamlessly communicate and send assets to each other. Polkadot's interoperability, security, and low transaction fees make it the ideal infrastructure for the multi-chain world of Web 3.0.

Kusama is Polkadot's canary network. Kusama's lower barrier of entry relative to Polkadot has resulted in rapid innovation and growth with no signs of slowing down.

> The Polkadot and Kusama ecosystems are commonly referenced together as the "**DotSama**" ecosystem.

Why Moonbeam?

Moonbeam is the first fully Ethereum Virtual Machine (EVM) compatible parachain on Polkadot.

Moonbeam combines the best of both worlds:

1. Polkadot's scalability & interoperability.
2. Ethereum's Ease-of-Use & developer community.

Users, developers, and infrastructure providers already familiar with Ethereum can easily use, deploy, and build on Moonbeam while benefiting from the scalability and interoperability of Polkadot.

Moonwell Artemis, our Moonbeam deployment, will be activated by the Moonwell community once favorable economic conditions on Moonbeam are met.

Why Moonriver?

Moonriver is Moonbeam's sister network on Kusama.

As the first EVM compatible parachain on Kusama, Moonriver has attracted numerous projects, developers, and users.

Moonwell Apollo, our Moonriver deployment, will be our first deployment given Moonriver's already sufficient DEX liquidity and Chainlink oracle support. Moonriver is where new products are expected to be incubated and developed prior to being deployed on Moonbeam.

👥 Team

Luke Youngblood - Engineering

- Former Coinbase Sr. Staff Engineer & AWS Principal Engineer
- Built the Coinbase staking rewards infrastructure
- Built the Tezos validator node infrastructure & launched the Tezos network
- Built the Coinbase price oracle, providing Compound protocol with secure price feeds

Lucas Kozinski - Operations & Finance

- Chief Operating Officer (COO) at Tokensoft Inc
- Program Lead for the Tezos Betanet Launch
- Former Management Consultant with Point B

Moonwell's core team also includes several former Coinbase and Google blockchain engineers specialized in DeFi and security. Moonwell is also partnered with (RBL), the same team that built Benqi, the lending & borrowing protocol on Avalanche. RBL provides some of the most experienced developers and designers in the crypto space.

Moderators

Telegram Handle	Discord Handle
@Chrizzled	chrizy#3492
@Fechuky	Fechuky_25#1261
@lFeil	Fei#4375
@Richnd	Richnd#1047
@GB_14	GB#6376
@Tommmymlt	Molot#4905
@Kanbeok	Kanbeok#7308

⚠ We will **never** contact you first!
We will **never** ask you to send funds!
We will **never** ask for your seed phrase or passwords!
Please be aware of scammers!
Moonwell only provides support in Discord or Telegram.

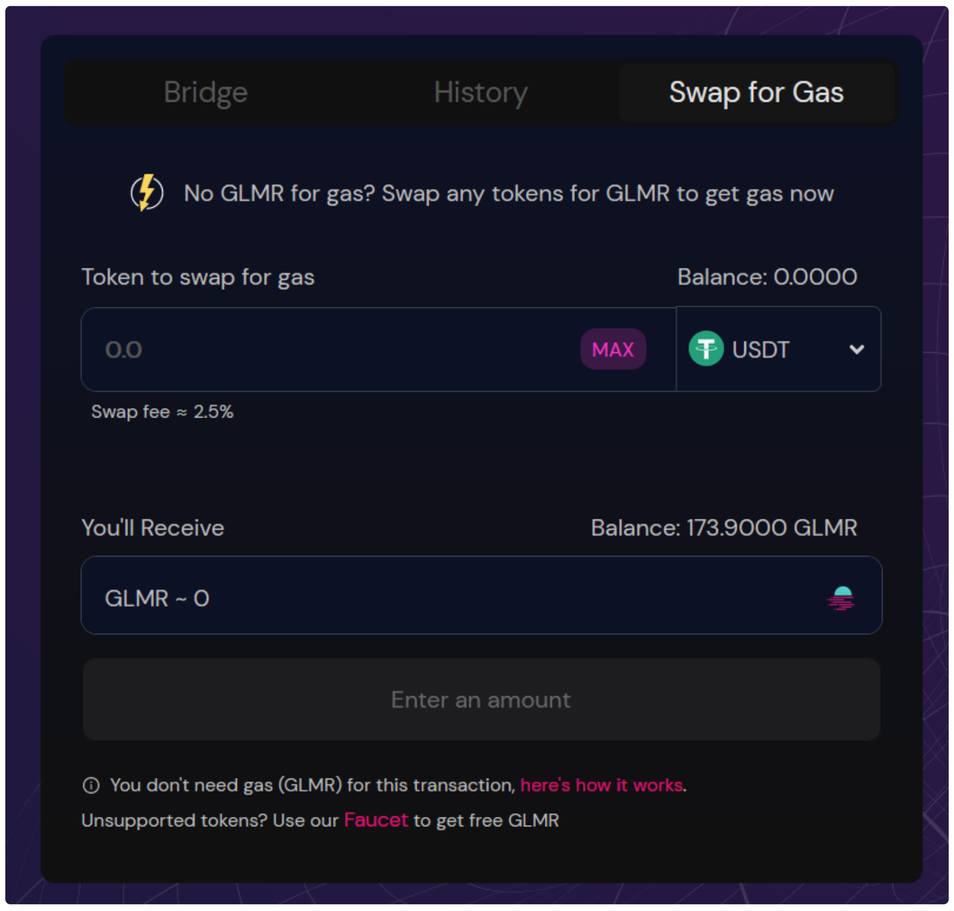
ⓘ If you need technical support, please message one of the moderators via Telegram or Discord.

Using Moonbeam & Moonriver

Acquiring GLMR for Initial Moonbeam Transactions

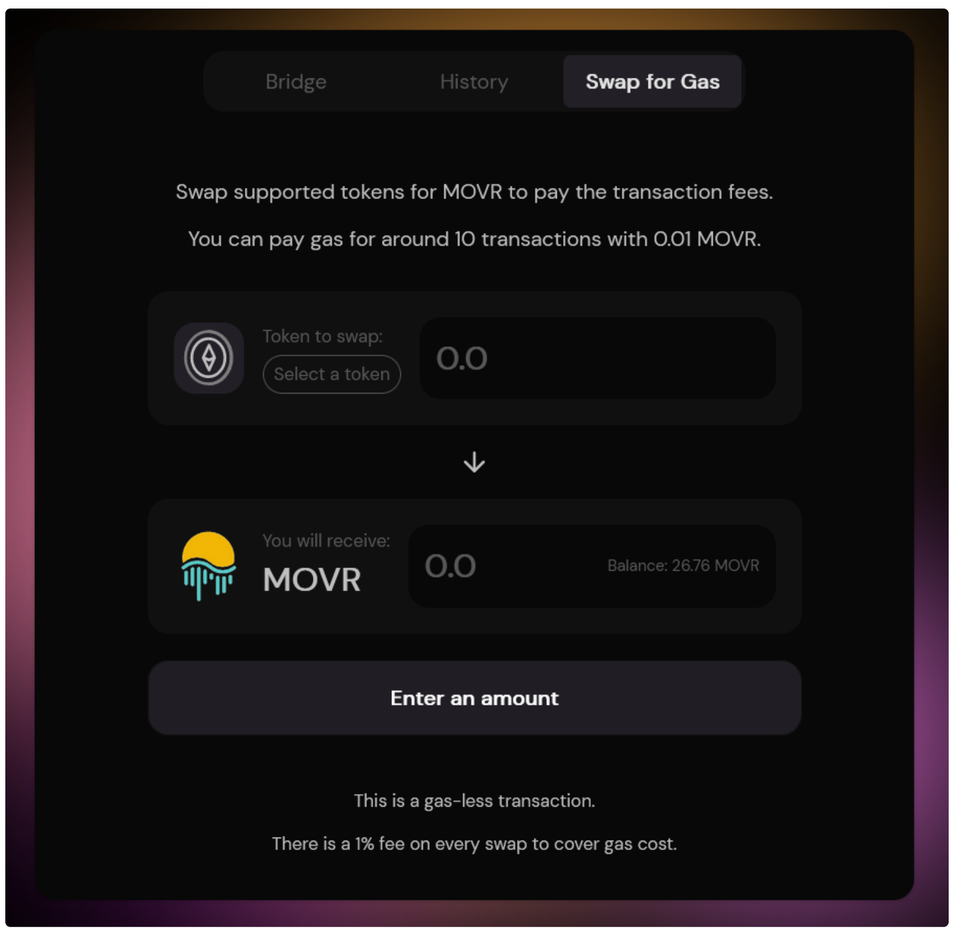
Several bridges will deposit a nominal amount to cover the initial transactions fees. In the event that it is not enough, or a bridge was used that does not provide a starting transaction fee, there is a gasless "Swap for Gas" feature located at StellaSwap.

Please click here to access the "Swap for Gas" feature.



Aquiring MOVR for Initial Moonriver Transacations

If you require MOVR for initial transactions on Moonriver, please click here to access Solarbeam's "Swap for Gas" feature.



Where to Swap Tokens on Moonbeam and Moonriver

Moonbeam:

StellaSwap | Documentation

Zenlink | Documentation

Moonriver:

Solarbeam | Documentation

Huckleberry Finance | Documentation

Zenlink | Documentation

Bridging Assets

> This page is not an endorsement of any of the bridging protocols described below, and any such bridging protocols are described purely as illustrative examples. Always do your own research about the risks of the bridging protocols described before interacting.

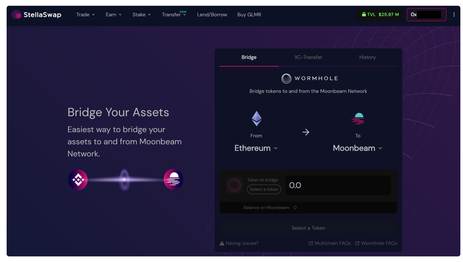
Bridging Assets to Moonbeam

Users starting out on Moonbeam network for the first time will need to bridge their assets from other chains in order to use dapps in the ecosystem. The bridge used in this example will be the native bridge on the StellaSwap DEX. StellaSwap's simplified and integrated front-end bridge, built over the **Wormhole** protocol, allows for users to easily bridge over their assets from Ethereum Mainnet and BNB Smart Chain.

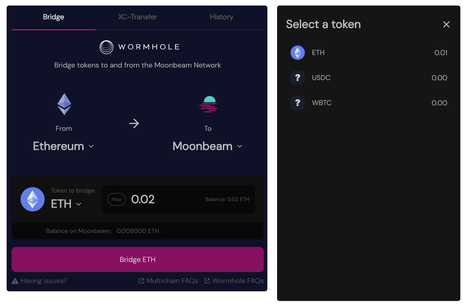
StellaSwap's native bridge currently supports:

- **USDC**, **ETH**, and **WBTC** from Ethereum Mainnet
- **BUSD** from BNB Smart Chain

Step 1: Connect your wallet by clicking `CONNECT` at the top right of the page. After connecting, select the origin network. This is where tokens will be bridged from. StellaSwap's bridge currently supports bridging to Moonbeam from Ethereum and BNB Smart Chain (BSC). For this example, we will be selecting Ethereum.

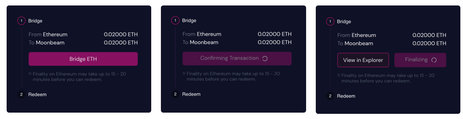


Step 2: Select the `Token to bridge`, input the amount you would like to bridge over, and click `Bridge`.

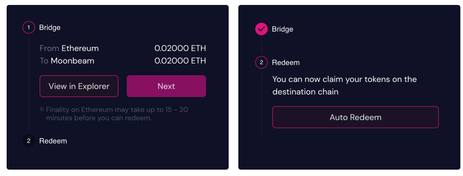


Step 3: Click `Bridge` once more in the pop-up modal and sign the wallet transaction when prompted. After the transaction has been confirmed in wallet, you will be able to click `View in Explorer` to check the status.

> Note that it may take up to 15-20 minutes before the bridging has completed and you are able to redeem your tokens.



Step 4: Once the bridge transaction has been finalized, click `Next` and then `Auto Redeem` . After clicking `Auto Redeem`, you should find your new Wormhole-wrapped assets in your wallet!



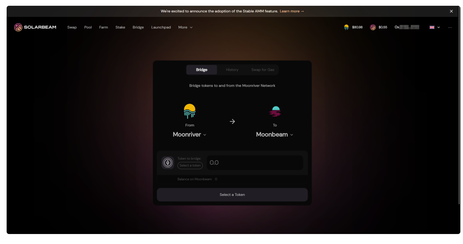
Bridging Assets to Moonriver

Users starting out on Moonriver network for the first time will need to bridge their assets from other chains in order to use dapps in the ecosystem. There are several bridge options available supporting a wide range of assets.

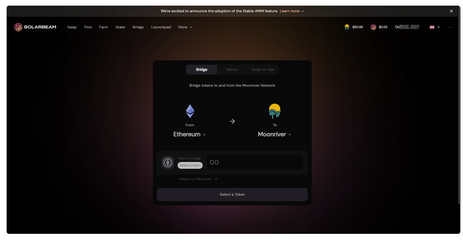
The bridge used in this example will be the natively integrated bridge powered by Multichain on Solarbeam DEX.

Solarbeam and Solarflare have a simplified and integrated front-end bridge, built over the **Multichain** protocol. The current interface uses v2 **Multichain** assets.

Step 1: Select origin network. This is the network where tokens will be bridged from.

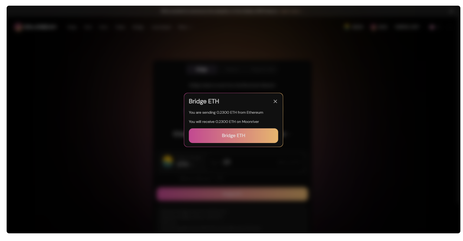


Step 2: Select the destination network. In this case, the destination network is "Moonriver"



Step 3: Select the token to be bridged and input the amount you wish to bridge.

Step 4: Approve the bridge on the site and in your wallet.



Transactions using this bridge can sometimes take a few minutes. If the bridge attempt does not complete after 15 - 30 minutes, try to manually push your transaction. For any further problems, refer to Solarbeam's documentation.

Supply

Supplying assets to Moonwell

Supplying Assets

Users on Moonwell Apollo and Artemis are able to supply their assets to single-sided liquidity pools, or Money Markets, in order to start earning a variable interest rate, which is based on a block-by-block basis. Users are also able to leverage their supplied assets to borrow other tokens from the protocol.

Before supplying, determine if the token you wish to utilize is supported. Some bridges will have their own version of an asset which may not be supported on Moonwell. For a list of assets supported on Moonwell, please (click here).

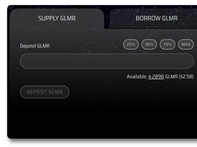
> **Moonwell Apollo** supports multi (Multichain, formerly Anyswap) assets.
> **Moonwell Artemis** supports .wh (Wormhole) assets.

If a user is wanting to supply an incompatible version of the asset, it may be possible to swap it at a decentralized exchange if there is sufficient liquidity. Otherwise, the only solution is to bridge back to the origin network, and then return on a compatible bridge or with a compatible asset.

> Please be aware that in rare situations, utilization can reach 100% which would result in being **unable** to withdraw tokens until there is sufficient liquidity (such as loan repayments, or additional liquidity being supplied).

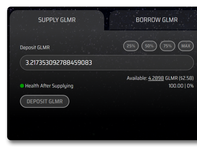
Step 1: Navigate to "Supply" tab

For this demonstration, GLMR (Moonbeam's native token) will be used.



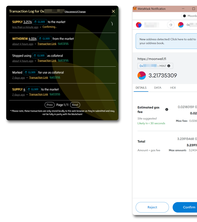
Step 2: Select amount of asset to supply

The Health Factor Value status will fluctuate depending on the amount of assets supplied compared to the amount borrowed. In this demonstration, the health factor is 100 as there are no outstanding loans.



Step 3: Confirm in Metamask

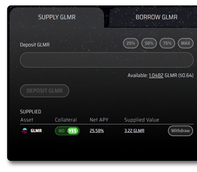
The user transaction history will display, beside the wallet transaction request. Confirm the transaction in wallet. Additionally, the transaction history can be closed or re-opened by the button in the bottom right corner of the page.



Success! Transaction completed.

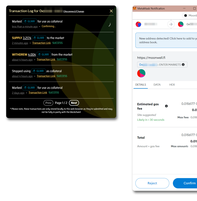
Enable Collateral

Step 1: Go to "Supply" tab, toggle "Yes" Collateral

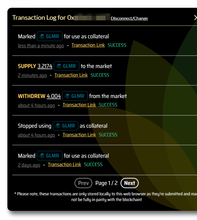


Step 2: Confirm using asset as collateral

Step 3: Confirm in Metamask



Success!



Withdraw Funds

Step 1: Navigate to the "Supply" tab.

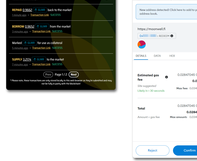
Click on the "Withdraw" button to open the modal window for withdrawing funds.

Step 2: Select the amount you wish to withdraw

In the modal window, select the amount you wish to withdraw. In this case, "MAX" was selected in order to withdraw the full amount.

Step 3: Approve the withdrawal transaction in Metamask

Confirm and sign the transaction in Metamask. All previous transactions can be viewed in the transaction log.



? Supplying FAQ

How do I supply assets?

Please review the instructions located here.

What is the source of earnings?

Lenders receive continuous earnings which are dependent on certain market conditions from the borrower payment interest fees.

Interest rates are shared between suppliers, which are derived from borrowers fees and corresponds roughly to average borrow rate * utilization rate. A higher utilization of the reserve yields higher earnings for depositors.

> Each asset has an individual market of supply and demand, and APY which fluctuate over time.

Currently, there are 2 types of rewards for supplying assets: **Supply APY**, and **Distribution APY**.

Supply APY automatically accrues in the smart contract, and the rewards are in the token supplied based on fees paid by borrowers. For example: supplying **GLMR** will accrue rewards in **GLMR**.

Distribution APY are additional rewards, which need to be claimed in the rewards panel. These are additional rewards that are earned over a period of time.

Protocol	Reward	Source
Artemis	GLMR	Moonbeam Ignite
Artemis	WELL	Moonwell Artemis token
Apollo	MFAM	Moonwell Apollo token
Apollo	MOVR	First MOVR Advantage

Is there a minimum or maximum amount?

There is no minimum or maximum amounts to supply. However, take in to account the gas fee and transaction costs which may skew your expected earnings when you deposit very low amounts.

How do I withdraw?

Please review the instructions located here.

Ensure there is enough unborrowed liquidity before you withdraw. If there is an insufficient balance, you will need to wait for more liquidity before withdrawing.

Can I opt-out my asset from being used as a collateral?

Yes. You can opt out of individual assets from being used as collateral in the “Supply” section of the dashboard.

You will earn rewards from supplying assets whether you have enabled collateral or not.

Borrow

Borrowing assets on Moonwell

Moonwell allows users to leverage or "collateralize" their supplied assets in order to borrow other assets. Once your supply has been deposited and you have enabled collateral, you can obtain an over collateralized loan from the protocol. The maximum borrowable amount will be based on the value of the supplied collateral, as well as the parameters established in the protocol parameters.

Before starting, please ensure "collateral" is enabled in the "Supply" tab. Click here for instructions. For the purpose of this demonstration, the GLMR asset will be used.

About Your Health Factor Value

The health of the loan is determined by the Health Factor Value, but is otherwise a perpetual open loan. Please be aware that there is an interest based fee which can impact your Health Factor Value and put your loan at risk of liquidation.

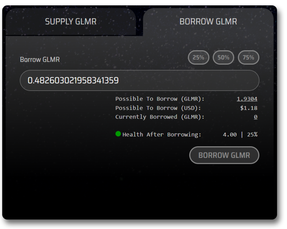
> Please regularly monitor your loan health, and ensure your score is always safe.
> There is **no** notification system at present to notify you if your loan is at risk.

> If you don't repay your loan, and your Health Factor Value drops too low, a liquidation event will be triggered to repay the loan, and you will lose the value of your loan. So it's important to only borrow what you can afford to repay, and continuously monitor your Health Factor Value.

Step 1: Navigate to "Borrow" tab

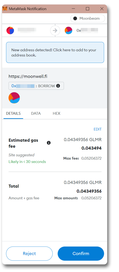
The borrow tab will display an estimated health factor value adjustment based on the amount borrowed.

Select, or input, the amount you wish to borrow.

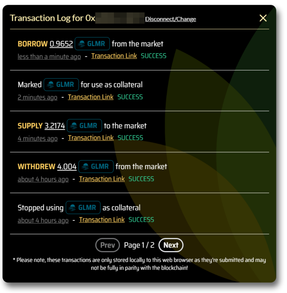


Step 2: Confirm transaction in Metamask

Approve the transaction in Metamask. A transaction log will also appear which will show details about previous transactions.



Success!



> Your dashboard will now have additional information displaying health factor, loan composition, and more.

Repaying Borrowed Assets

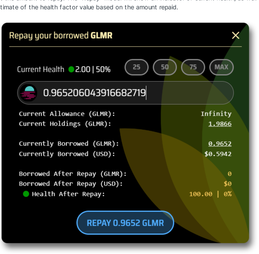
In order to keep the Health Factor Value at a safe value, it is recommended to pay back the loan.

Step 1: Navigate to the "Borrow" tab and click "Repay"

Open loans can be viewed in the "Borrow" tab. Repaying the loan is also available in the "Borrow" tab. Clicking "Repay" will open up a modal window with further instructions.

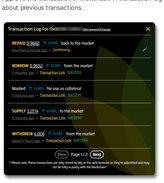
Step 2: Choose amount to repay

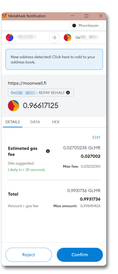
Select the amount to repay. The "Repay" modal will show an indicator of current health, as well as an estimate of the health factor value based on the amount repaid.



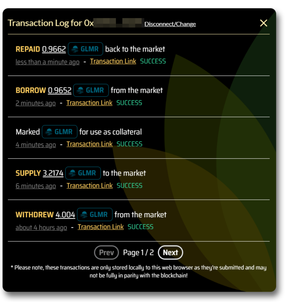
Step 3: Approve the transaction in Metamask

Approve the transaction in Metamask. A transaction log will also appear which will show details about previous transactions.





Success!



Borrowing FAQ

What is the primary use case of borrowing?

Borrowing allows users to obtain liquidity from their assets, without having to close their position on the assets. This allows a user that is longing their assets with upside to borrow against it for any reason they choose.

How do I borrow?

To get started with borrowing, a user must first deposit their assets to be used as collateral. For step by step instructions, please click here.

How much I can borrow?

The potential borrowing amount is dependent on the value of assets deposited, user health factor, as well as the available liquidity. If there is not enough liquidity, or if your health factor drops below a certain threshold, you will not be able to borrow. For more information regarding collateral parameters, please click here.

Are the interest rates stable?

No. The rates fluctuate and are subjected to rebalancing over a longer term as a response to changes in market conditions, supply and demand, overall market conditions, and other factors. They may or may not be optimal rates.

Interest rates are determined by the borrowing rate, as determined by the supply and demand ratio of the asset.

> Please be diligent in monitoring your health factor. Fees from borrowing rates can add up, and put your loan at risk of liquidation.

Why can't I borrow a particular asset?

There are a few reasons:

No collateral factor: this means there is not enough liquidity on the blockchain to safely enable borrowing of the asset

Not enough liquidity: this means there is insufficient liquidity. Waiting for more liquidity to be supplied will allow you to borrow.

Health Factor

What is the health factor?

The “health factor” metric is a representation of the status of your assets (supplied vs borrowed), and the underlying value of those assets. A higher value indicates a healthier collateral coverage asset ratio and a safer margin against liquidation. If the health factor reaches a specific threshold, the assets are at risk of triggering a liquidation event. If the health factor falls below the threshold, there is a high likelihood of liquidation occurring.

If you want more information about the collateral parameters as well as health factor calculations, please click here.

Where can I find my health factor value?

The health factor is located at the top of the screen, under the “Your Balances” card.

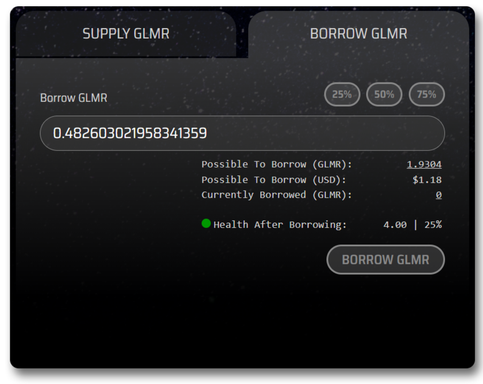
Additionally, the health factor will display in the “Borrow” window, which displays a dynamic estimate based on how much will be borrowed.

What happens when my health factor is reduced?

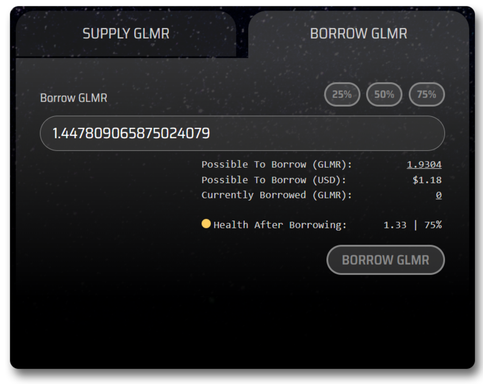
The health factor value will increase or decrease in response to the fluctuation of value of the assets. If the health factor increases, the borrow position is improved and there is a greater margin of safety from the liquidation threshold.

In the event of the value of the collateralized assets ratio decreasing, the health factor is reduced which increases the risk of liquidation.

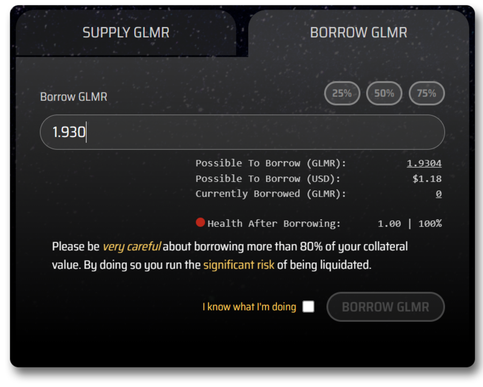
A healthy value is indicated with a green indicator.



As the user approaches a higher risk score, it will be indicated with a yellow indicator.



As a user approaches a higher risk of liquidation, the indicator becomes red. Additionally, the user is prompted to acknowledge the higher risk of liquidation before they can borrow.



How do I improve my health factor?

The health factor value can be improved by the following:

1. **Closing the balance by repaying the loan.** This will have the most impact on the health factor value.
2. **Adding more collateral.** This will have some impact on the health factor value, but not as much as repaying the loan balance

When do I need to pay back the loan?

There is no set time for repayments. The loan repayment is determined by the health factor value. As long as the health factor stays above 1, the loan remains open in perpetuity. When the health factor value falls below 1, it will trigger a liquidation event of the users assets to recover the loan.

> ⓘ Please note that while there is no repayment time, interest will still accrue which can decrease the health factor value over time, and can put the assets at risk for liquidation.

Repayments

How much would I pay in interest?

Interest rates are determined by the borrowing rate, as determined by the supply and demand ratio of the asset.

When do I need to pay back the loan?

The loan is perpetual, so there is no fixed time period. The factor depends entirely on whether your position is safe. This allows users to borrow for an undefined period of time. As time passes, accrued interest grows and could affect your health factor value, which may put your assets at risk of becoming liquidated.

How do I pay back the loan?

Please review the instructions located here.

👀 Liquidations

What is Liquidation?

A liquidation occurs when the health factor of a borrower drops below a particular threshold (health factor value of "1"), or when ones borrow limit reaches 100% due to the collateral being unable to cover one's loan value. The user's health factor is determined by your collateral vs loan value. When your collateral decreases in value, or the borrowed debt increases in value, this will influence your score.

When a liquidation occurs, up to 60% of the borrower's debt is used to repay the loan. That value (50% Close Factor + 10% Liquidation Incentive), is taken from the available collateral. After a liquidation, the amount liquidated from the debt is repaid.

Liquidation can be avoided by routinely monitoring the health factor, and ensuring it is as high as possible. If the health factor is lower, a user can avoid liquidation by raising the health factor. This includes depositing more collateral, or paying part of the loan. Repayments will typically increase health factor more than additional deposits. Users must also routinely monitor the health factor to avoid liquidation.

What Are Liquidators?

Liquidation mining is highly competitive, but available to anyone who wishes to participate.

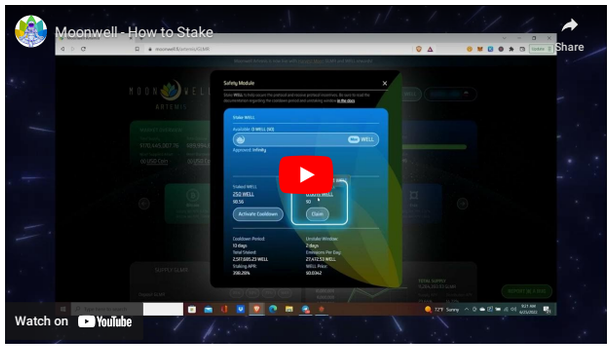
Liquidators are a vital part of the borrow and lending ecosystem. Liquidators protect depositors and help offset loan defaults on borrowers. This is accomplished by turning the discounted collateral into a loan.

How do I avoid liquidation?

Liquidation events are triggered by the health factor declining below the threshold. Repaying the loan, or depositing more assets, can increase your health factor. Repaying the loan will improve the health factor value greater than depositing more assets.

Safety Module

Stake WELL and MFAM tokens to receive increased rewards while also securing the Moonwell protocol via the Safety Module.



What is a Safety Module?

Moonwell's primary mechanism for securing the protocol is the **Safety Module** – a smart contract that allows users to stake their assets in order to protect and mitigate against "Shortfall Events".

A **Shortfall Event** occurs when there is a deficit in the markets of the Moonwell ecosystem. When this happens, the Safety Module is used to cover the losses by selling the assets needed to mitigate the deficit.

Contributing to the Safety Module allows users to obtain additional incentives in return for a tokenized position which can be freely used in the protocol (for example: Governance). However, the original asset will always be at some form of risk if a Shortfall Event does occur.

> Users are able to redeem their token by activating the "cooldown period" and waiting 10 days. After the cooldown period, they will have a window of 2 days to redeem their tokens.

> If tokens are not claimed within the 2 day window, users will have to reactivate the cooldown period and wait another 10 days. Please be sure to add the date to your calendar.

How do I claim rewards?

While WELL or MFAM are staked in the safety module, you will accrue rewards in the form of additional WELL/MFAM tokens.

You will continue to accrue WELL and MFAM rewards during a cooldown period.

These rewards can be claimed at any time.

How do I Unstake?

1. To unstake WELL/MFAM tokens, the cooldown period needs to be activated first.
2. Once the **cooldown period** has been activated, you will have to **wait 10 days** until you can unstake your WELL or MFAM.
3. Once the cooldown period of 10 days is complete, you will have a **2-day window** to unstake your tokens.

> If tokens are not unstaked within the 2-day window, you will need to activate the cooldown period again!

What is a shortfall event?

A Shortfall Event is an incident that causes an unexpected loss of capital. Despite Moonwell's emphasis on solid security measures and software development practices, there remains a non zero risk of an event or exploit that causes funds to be lost. The Safety Module serves as a last line of defense to help mitigate the damage from these types of occurences.

Below are three examples of potential shortfall events.

Smart contract exploit: A smart contract bug or hack that causes capital loss. As a first line of defense, all Moonwell smart contracts go through multiple rigorous audits by Halborn Security. Additionally, Moonwell has engaged in an ongoing and continuous service agreement with Halborn Security.

Liquidation failure: An asset being used as collateral fails due to low liquidity or liquidators failing to perform liquidations efficiently. To combat this potentiality, Moonwell only allows for an asset to be collateralized once there is sufficient liquidity across the parachain to liquidate even the largest of borrowers. Markets undergo scrutiny and assessment both for contract security of the asset, as well as economic assessment. Typically, a market will require TVL in the amount capable of liquidating the largest position held on the protocol.

Oracle failure: The oracle network fails to properly update prices during extreme market conditions or network congestion, causing improper liquidations due to inaccurate price data. To secure the protocol against this threat, Moonwell uses Chainlink — the most secure and decentralized oracle network on the market.

What is the risk of staking?

There are precautionary measures taken to harden the Moonwell protocol against risks, particularly with potential shortfall events. Despite these efforts, if a Shortfall Event did occur, up to 30% of the assets staked in the Safety Module would be sold to cover the ensuing deficit. Although Shortfall Events are far from daily occurrences, it is still crucial to understand and weigh the potential risks before staking.

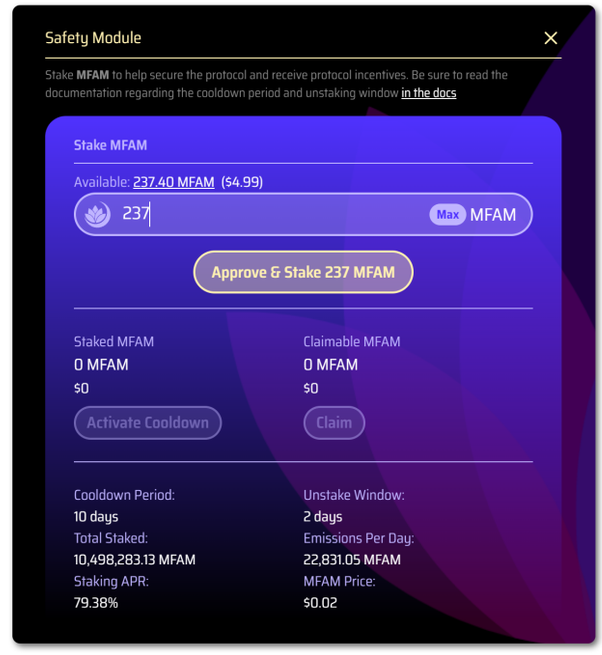
How do I stake my WELL or MFAM

Step 1: Navigate to the Safety Module

Click the `STAKE MFAM` or `STAKE WELL` button located at the top right corner of either the Moonwell Apollo (MFAM) or Artemis (WELL) dapps to open Safety Module window.

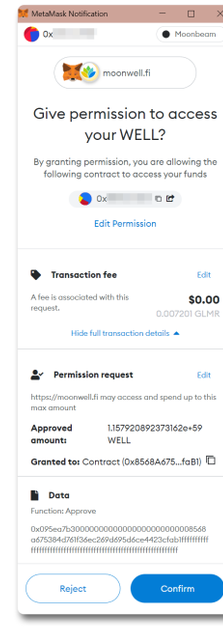
Step 2: Select the amount to stake

Enter the amount you would like to stake and click "Approve and Stake".



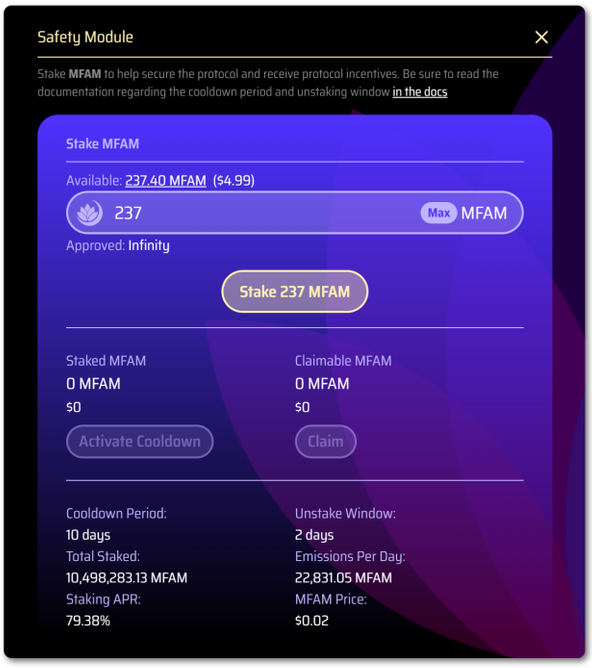
Step 3: Grant permission to Safety Module contract

Confirm and sign the transaction in Metamask to grant permission to the Moonwell Safety Module smart contract to access your funds. Transaction history can be viewed in the transaction log.



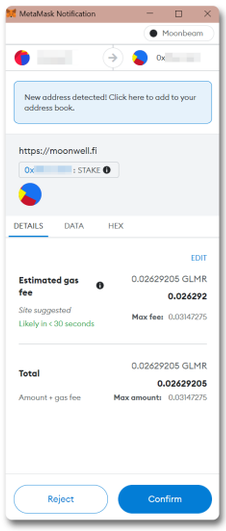
Step 4: Stake WELL or MFAM

Click "Stake (x amount) WELL/MFAM" to initiate staking



Step 5: Approve staking transaction

Confirm the transaction in Metamask. Transaction history can be viewed in the transaction log.

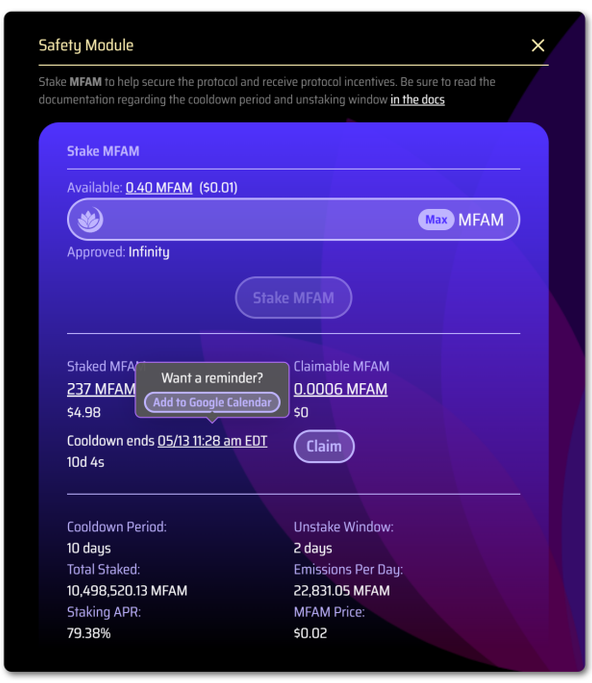


Success!

You have successfully staked WELL or MFAM into the Moonwell Safety Module!

After staking MFAM tokens, you will receive an equivalent amount of stkMFAM tokens. The stkMFAM tokens are a 1:1 representation of your staked MFAM, like a receipt.

Please note that staked WELL/MFAM tokens will be temporarily locked until you initiate the **cooldown period**.

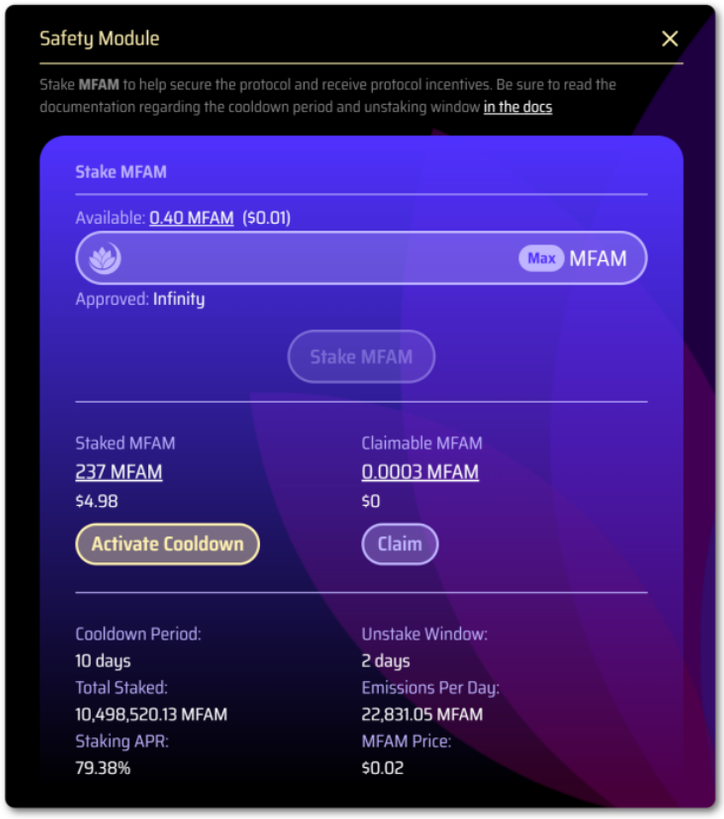


Set a Google Calendar alert to remind you when your cooldown period has concluded!

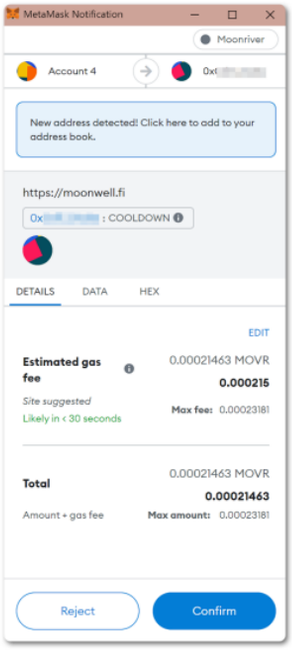
How do I unstake my WELL or MFAM?

Step 1: Activate Cooldown

To unstake tokens from the Safety Module, the cooldown period must first be activated.



Step 2: Confirm the transaction



Once the cooldown period is activated, you will have to wait 10 days until you can unstake your WELL or MFAM.

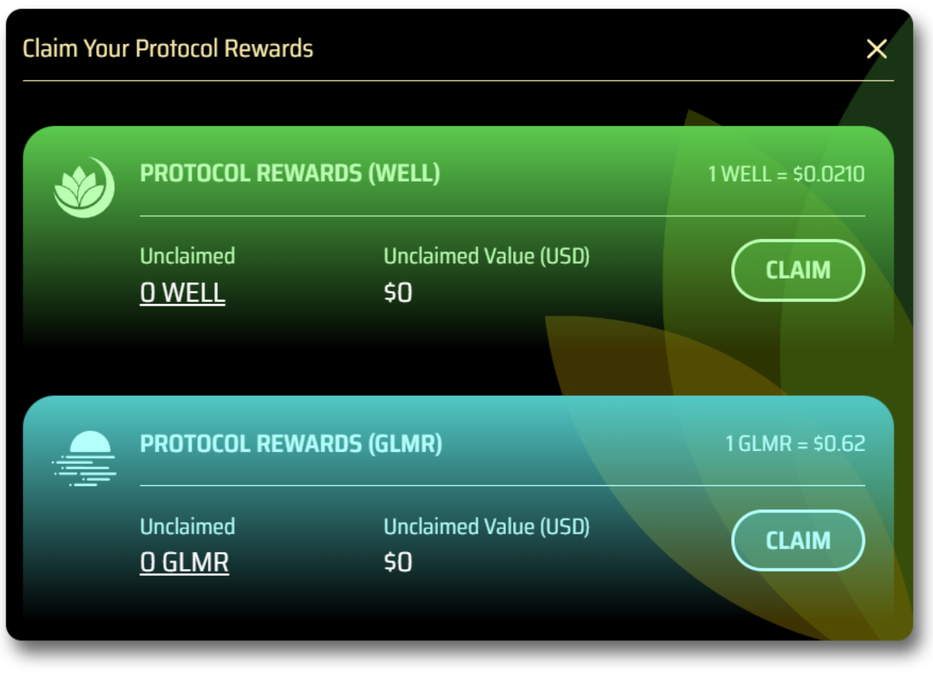
Once the cooldown period of 10 days is completed, you will have a 2-day window to unstake your tokens.

If the tokens are not unstaked within the 2-day window, you will need to activate the cooldown period again!

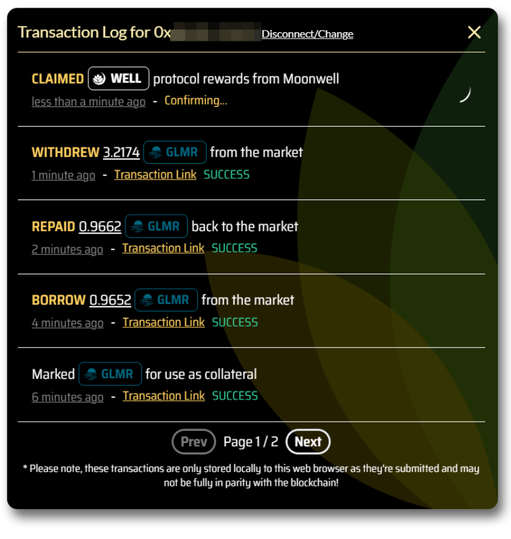
Claim Rewards

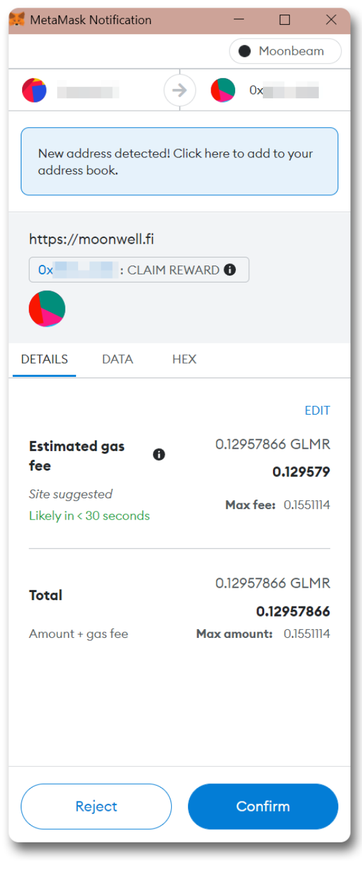
Step 1: Click "Rewards"

Step 2: Click "Claim"

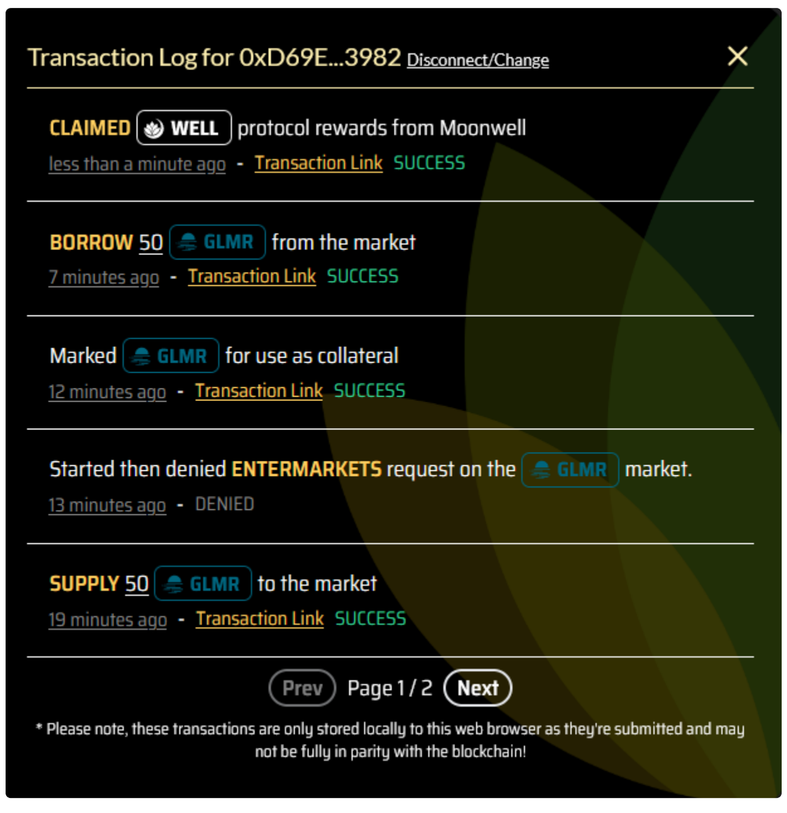


Step 3: Confirm in Metamask





Success!



😎 WELL

WELL is the native governance token of the Moonwell Artemis protocol. WELL can be staked in the Moonwell Safety Module, which backstops the Moonwell Artemis protocol in the event of a shortfall event. WELL can also be delegated to an ERC-20 address and used to participate in Moonwell Governance.

Contract Address

0x511aB53F793683763E5a8829738301368a2411E3

WELL Transparency Report

https://medium.com/lunartechfdn/well-transparency-report-8dfeb91f5176

Where Can I Buy & Sell WELL?

StellaSwap

KuCoin

Huobi

Gate.io

> ⚠ Buy and Sell at your own risk

Where can I farm WELL?

https://app.stellaswap.com/farm

🥰 MFAM

MFAM is the native governance token of the Moonwell Apollo protocol. MFAM can be staked in the Moonwell Safety Module, which backstops the Moonwell Apollo protocol in the event of a shortfall event. WELL can also be delegated to an ERC-20 address and used to participate in Moonwell Governance.

> ⓘ MFAM stands for "Moonriver Family" ❤.

Contract Address

0xbb8d88bcd9749636bc4d2be22aac4bb3b01a58f1

MFAM Transparency Report

https://medium.com/lunartechfdn/moonwell-apollo-mfam-token-transparency-report-52845c7aed7f

Where can I buy and sell MFAM?

Solarbeam DEX

MEXC

> ⚠ Buy and Sell at your own risk

Where can I farm MFAM?

https://app.solarbeam.io/farm

? Artemis FAQ

What is the Moonwell Artemis token?

WELL is the native governance token of the Moonwell Artemis protocol. WELL can be staked in the Moonwell Safety Module, which backstops the Moonwell Artemis protocol in the event of a shortfall event. WELL can also be delegated to an ERC-20 address and utilized in Moonwell Governance voting and proposal creation.

What is Moonwell Artemis?

Moonwell Artemis is Moonwell's deployment on Moonbeam.

What can we do on Moonwell Artemis?

Users can lend and borrow 7 different assets (BUSD.wh, USDC.wh, BTC.wh, ETH.wh, GLMR, xcDOT, and FRAX)

Why not X-token support?

To safely support an asset, there needs to be sufficient DEX liquidity and Chainlink Oracle support

Why StellaSwap as a DEX partner?

StellaSwap is the leading native DEX on Moonbeam.

Is Moonwell audited?

Moonwell is partnered with and audited by Halborn to ensure that top security practices are always followed.

Why Moonbeam?

Moonbeam is the first fully EVM compatible parachain on Polkadot. It combines Polkadot's scalability & interoperability with Ethereum's ease-of-use & developer community.

What does Moonwell represent?

Moonwell is the overarching brand for our two separate deployments on Moonriver and Moonbeam represented by Moonwell Apollo and Moonwell Artemis, respectively.

Moonwell represents wellness in all forms - financial, physical, and mental.

Why Apollo & Artemis?

Space mission theme

Apollo represents the first missions to the Moon and Moonwell's first deployment on Moonriver.

Artemis represents NASA's return to the moon in 2022 and Moonwell's later deployment on Moonbeam.

Greek mythology theme

Kusama is the canary network of Polkadot and Moonriver is the sister network of Moonbeam.

Artemis is the twin sister of Apollo and goddess of the Moon in Greek mythology.

? Apollo FAQ

What is the Moonwell Apollo token?

MFAM is the native governance token of the Moonwell Apollo protocol. MFAM can be staked in the Moonwell Safety Module, which backstops the Moonwell Apollo protocol in the event of a shortfall event. MFAM can also be delegated to an ERC-20 address and utilized in Moonwell Governance voting and proposal creation.

What is Moonwell Apollo?

Moonwell Apollo is Moonwell's deployment on Moonriver.

What can we do on Moonwell Apollo?

Users can lend and borrow 6 different assets (MOVR, BTC, WETH, USDC, USDT, FRAX)

Which bridge should I use?

https://app.solarbeam.io/bridge

Why not X-token support?

To safely support an asset, there needs to be sufficient DEX liquidity and Chainlink Oracle support

Why Solarbeam as a DEX partner?

Solarbeam is the leading native DEX on Moonriver.

Why Moonriver?

Moonriver is the first fully EVM compatible blockchain on Kusama. It already has sufficient DEX liquidity and Chainlink oracle support, two crucial requirements for a lending & borrowing protocol.

Is Moonwell audited?

Moonwell is partnered with and audited by Halborn to ensure that top security practices are always followed.

What does Moonwell represent?

Moonwell is the overarching brand for our two separate deployments on Moonriver and Moonbeam represented by Moonwell Apollo and Moonwell Artemis, respectively.

Moonwell represents wellness in all forms - financial, physical, and mental.

Why Apollo & Artemis?

Space mission theme

Apollo represents the first missions to the Moon and Moonwell's first deployment on Moonriver.

Artemis represents NASA's return to the moon in 2022 and Moonwell's later deployment on Moonbeam.

Greek mythology theme

Kusama is the canary network of Polkadot and Moonriver is the sister network of Moonbeam.

Artemis is the twin sister of Apollo and goddess of the Moon in Greek mythology.

Governance

Learn more about Moonwell Governance

The Moonwell protocol is governed and upgraded through the use of both on-chain and off-chain governance structures. Voting power and governance rights over Moonwell Apollo and Artemis are derived from the protocols' native governance tokens, MFAM and WELL. Once delegated, these tokens can be used to propose changes to Moonwell and vote on the proposals of others.

The central hub for governance and proposal-related discourse is the Moonwell Governance Forum. Anyone can register for an account on the forum, and all discussions are publicly available. The Moonwell Discord is also a great place to share ideas with other community members and stay up to date on the latest proposal and voting information.

On-chain governance proposals are voted on through the Governance Portal. Off-chain governance proposals and signaling votes are conducted from the Snapshot voting portals.

Further information regarding Moonwell Governance can be found on the following sub-pages:

- Delegate
- Proposals
- Voting
- Parameters
- FAQ

Delegate

Delegate MFAM and WELL tokens to activate their inherent voting power

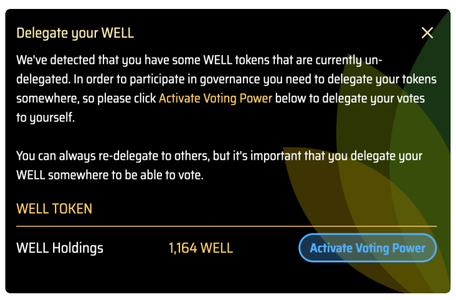
Introduction

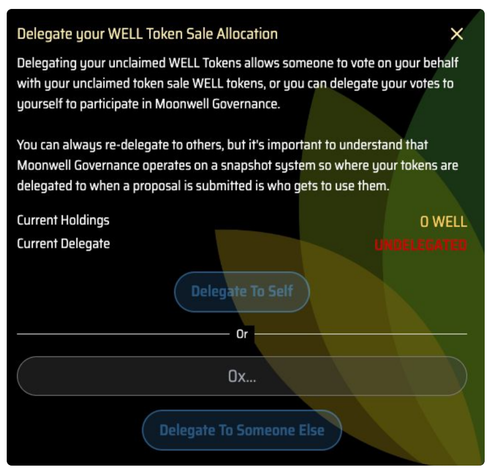
In order to propose changes to the protocol(s) and participate in voting, one must first delegate their WELL or MFAM tokens. When you delegate your governance tokens, you're giving whomever you delegate to the ability to vote with said tokens (if you "activate your voting power", you're delegating your WELL/MFAM to your own address). Users may only delegate to a single address at a time, be it their own or that of another who will represent them in governance and can change delegates at any time. Votes are delegated from the current block and onward, until the sender delegates again, or transfers their WELL or MFAM.

> **Unless delegated, WELL/MFAM tokens will not be able to be used for governance voting.**

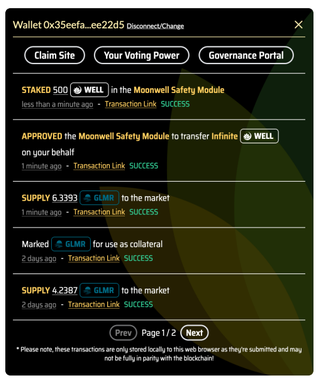
How to Delegate

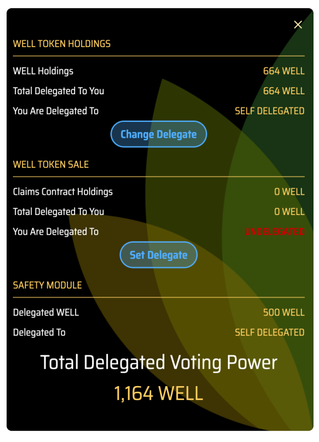
WELL and MFAM token holders are able to delegate their tokens from the Moonwell Artemis and Moonwell Apollo dapps, Governance Portal, and Claims Site. Upon visiting these sites while holding un-delegated WELL or MFAM, users will be prompted to delegate. By clicking "Activate Voting Power", you will delegate your WELL/MFAM to your own wallet address.

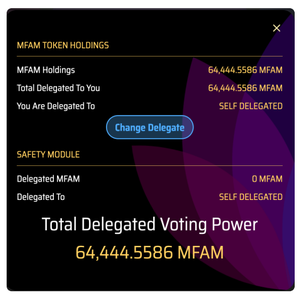




From the transaction log (which you can access by clicking your wallet address in the top right), users will see a `Your Voting Power` button. Clicking this will bring up a modal that shows ones total delegated voting power and a breakdown of where the power comes from. This total voting power displayed is an aggregate of delegated WELL or MFAM, delegated claims, and the amount staked in the Safety Module. From this modal, you're also able to assign to which address their voting power is delegated to.







Becoming a Delegate

If you are interested in becoming a delegate and representing others in Moonwell Governance, create a post in the Delegation Pitch category of the forums. This is where you will be able to introduce yourself, share your vision for the protocol, define your framework for voting on proposals, and expound upon why others should delegate to you. Before posting, please review the Delegate Guidelines.

> **For further information regarding delegation, please visit the Delegate FAQs.**

📜 Proposals

Propose upgrades and changes to Moonwell

Introduction

Through the proposal process, the Moonwell community is able to recommend, discuss, vote upon and implement changes to the Moonwell protocol without requiring any sole individual or contributing team to activate those changes.

Ideation and Forum Discussion

In general, proposals should start with a post in the Moonwell Governance Forum. Before submitting a proposal to the forum, please consult the proposal guidelines. If you're unable to distill your idea down into a focused and structured proposal, share it first in the ideas subcategory of the forums or in the #governance channel of our Discord.

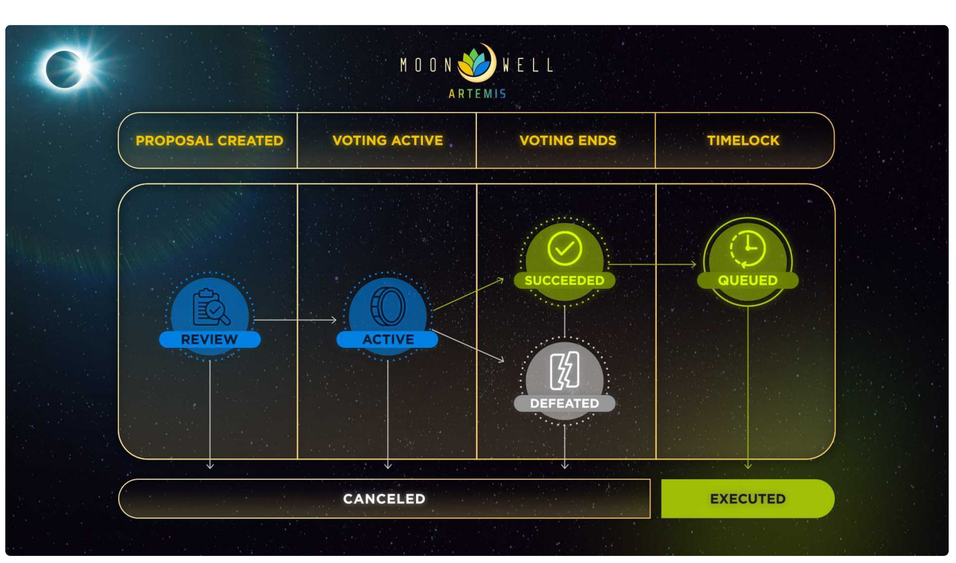
Anyone can register for an account on the forum or join the Moonwell Discord. The community welcomes thoughtful proposals and discourse about those proposals from everyone, whether or not they are a WELL or MFAM token holder.

Off-Chain Voting (Snapshot)

For proposals that will result in substantial changes to the Moonwell protocol, it is recommended that the proposal author first gauges community interest by capturing an off-chain "signaling vote." Signaling votes are non-binding, since they don't have the ability to directly change the protocol, however, they are a valuable means of measuring sentiment on substantive issues. Signal voting and voting on proposals that do not require upgrades or changes to the Moonwell Artemis code base utilize the Snapshot Voting Portal.

On-Chain Voting (MIP)

The process of implementing changes to the Moonwell protocol begins with a proposal, known as a **Moonwell Improvement Proposal**, or MIP**.** These on-chain proposals possess both a description and code that will be executed on the Moonbeam network following its passage. The Governance Portal makes it easy to browse and vote upon MIPs that are currently undergoing active voting. The full proposal process has been illustrated in the diagram below.



Proposal Process

Proposal Creation

A Moonwell Improvement Proposal can be created by any wallet that has enough voting power to meet the proposal threshold, which is currently set at 400,000 votes for Artemis and 500,000 votes for Apollo. The proposal author must submit the description and code, which consists of a set of up to 25 operations that will be executed if the proposal passes. Once the proposal has been created, the active voting period begins after a small delay of 60 seconds.

Voting Period

The initial voting period is 3 days, during which any wallet with voting privileges can register a vote of `YAY`, `NAY`, or `ABSTAIN`.

At the end of the voting period, a proposal will be considered to have succeeded if it meets the following criteria:

- Proposal has received a majority of Yay votes (greater than 50% `YAY` when compared to `NAY`)
- Proposal has received quorum of at least 100,000,000 total votes (WELL)
  - *Moonwell Apollo uses a dynamic "floating quorum", currently set at 20,000,000 votes (MFAM)

Quorum is a measure of all `YAY`, `NAY`, and `ABSTAIN` votes on a proposal, therefore it is possible for `ABSTAIN` votes to assist a proposal in reaching quorum and passing successfully. If a proposal does not achieve both of these requirements during the 3 day voting period, it will be considered defeated, and cannot be executed.

Timelock

After a proposal succeeds, it will be queued in the Timelock, which introduces a mandatory 24 hour delay before the proposal is able to be executed. This is intended to give the community time to react to a proposal before it is executed.

Execution

Once the Timelock delay has elapsed the proposal can now be executed by any account on the Moonbeam network, at which point all of the operations included in the proposal will be performed.

Voting

How to vote on governance proposals

Introduction

Moonwell Governance leverages both on-chain and off-chain voting mechanisms. Below is a step-by-step guide showing how to review and vote upon active proposals from the Governance Portal (on-chain) and the snapshot portals for Apollo and Artemis (off-chain).

Prerequisites

In order to participate in voting you will need:

- Web-3 wallet
- Delegated WELL/MFAM tokens
- GLMR/MOVR for gas (does not apply for Snapshot voting)
- An active proposal to vote on

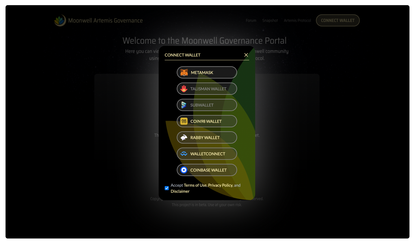
> Your WELL/MFAM must be delegated prior to a proposal becoming active in order for your votes to count toward the current proposal.

On-Chain Voting

Once a Moonwell Improvement Proposal has made it through the proposal process and has been submitted, it will appear on the Governance Portal. Here you can view all current and former MIPs.

Step 1 - Connecting to Governance Portal

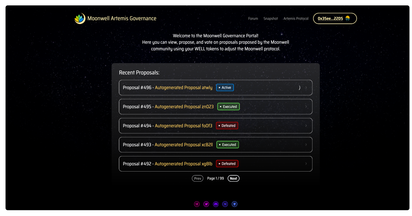
- Navigate to https://gov.moonwell.fi
- Upon connecting to the Governance Portal, a pop-up modal will ask you to connect your Web3 wallet
- Accept the Terms of Use, Privacy Policy, and Disclaimer
- Connect wallet



Step 2 - Review Governance Portal

After connecting your wallet, you'll see the main components of the Governance Portal.

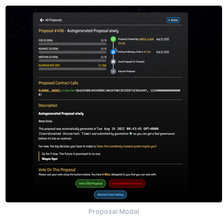
- Your voting wallet in the top right
  - Your total voting power can be viewed by clicking on your voting wallet
- Links to the Governance Forums, Snapshot Portal, and the Moonwell dApps
- Current and former proposals



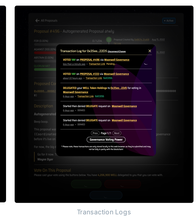
Step 3 - Voting

- If a proposal is currently live for voting, it will say `Active` next to the title. Upon clicking on an active proposal, a pop-up modal will appear showing proposal details.
  - Current vote tally
  - Time remaining
  - Proposed contract call
  - Proposal description
  - User voting power
  - Vote `FOR`, `AGAINST`, and `ABSTAIN` buttons
- To vote on a proposal, simply click on the voting action you want to choose. For the sake of this guide, we'll be voting `FOR` this proposal.
- After choosing a voting direction, you'll be presented with a transaction log modal showing your voting direction and prompted to confirm the transaction via wallet.
- Confirm transaction in wallet
- Once your transaction has been completed, you'll see the confirmed state in the transaction logs and proposal modal.
- You've successfully voted!

> Note that you can only vote once per proposal. Make it count!



Proposal Modal



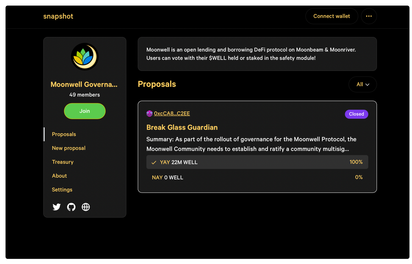
Transaction Logs

Off-Chain Voting (Snapshot)

Signal voting and voting on proposals that do not require upgrades or changes to the Moonwell Apollo/Artemis codebase utilize snapshot voting portals.

Step 1 - Connecting to Snapshot Portal

- Navigate to either the Moonwell Artemis snapshot portal or the Moonwell Apollo snapshot portal.
- Upon connecting to the snapshot portal, a pop-up modal will ask you to connect your wallet
- Select and connect wallet



Step 2 - Voting

- Select an active proposal
- Select the options you would like to vote for, in order of preference
- Click on the "Vote" button and sign the message via your wallet
- You've successfully voted!

> The "Vote" button will only unlock and be clickable once all options have been selected, in order of user preference.

Parameters

Governance parameters

> Note that the governance parameters outlined below can be modified by the Moonwell community through the voting process.

Artemis Parameters

Description	Off-chain. Signal voting and changes not related to core protocol parameters	On-chain. Changes to the core code base of the protocol
Proposal Threshold	200,000 votes	400,000 votes
Quorum	10,000,000 votes	100,000,000 votes
Voting Period	3 days	3 days
Voting Delay	None	1 minute

Apollo Parameters

Description	Off-chain. Signal voting and changes not related to core protocol parameters	On-chain. Changes to the core code base of the protocol
Proposal Threshold	250,000 votes	500,000 votes
Quorum	5,000,000 votes	40,000,000 votes
Voting Period	3 days	3 days
Voting Delay	None	1 minute

Glossary

Proposal Threshold - The minimum number of votes required for an account to create a proposal.

Quorum - The required minimum number of votes in support of a proposal for it to succeed.

Voting Period - The duration of voting on a proposal

Voting Delay - The amount of time to wait before voting on a proposal may begin.

FAQ

Governance FAQs

- Who can participate in governance?

Delegate FAQ

- What does it mean to delegate?
- Why would I want to delegate my voting power to someone else?
- If I delegate to another address will I retain ownership of my WELL/MFAM?
- Am I able to modify to whom my WELL or MFAM is delegated to?
- Upon acquiring additional WELL/MFAM tokens, do I need to delegate again?
- Will my Safety Module staked tokens count towards my voting power?
- Will my unclaimed sale tokens count towards my voting power?

Proposal FAQ

- Where can I share my ideas and proposals?
- Are there any rules or guidelines for posting proposals on the forum?
- What sort of changes can we propose?
- What is the threshold for a proposal to pass?

📜 Contracts

Moonwell Smart Contract Addresses

WELL Contract Address

0x511aB53F793683763E5a8829738301368a2411E3

MFAM Contract Address

0xbb8d88bcd9749636bc4d2be22aac4bb3b01a58f1

Artemis Contracts

Name	Contract Address
Comptroller	0x8E00D5e02E65A19337Cdba98bbA9F84d4186a180
ChainlinkOracle	0xED301cd3EB27217BDB05C4E9B820a8A3c8B665f9
Maximillion	0xe5Ef9310cC7E3437bAD83466675f24FD62A380c3
EcosystemReserve	0x7793E08Eb4525309C46C9BA394cE33361A167ba4
EcosystemReserve Controller	0xCa889f40aae37FFf165BccF69aeF1E82b5C511B9
JumpRateModel	0x1Ce7e4928943d6A4820375eBe737204dc1E73755
GovernorAlpha	0xfc4DFB17101A12C5CEc5eeDd8E92B5b16557666d
breakGlassGuardian	0x5402447a0db03EeE98c98b924F7d346bd19cdD17
Timelock	0x3a9249d70dCb4A4E9ef4f3AF99a3A130452ec19B
mGLMR	0x091608f4e4a15335145be0A279483C0f8E4c7955
mxcDOT	0xD22Da948c0aB3A27f5570b604f3ADef5F68211C3
mxcUSDT	0x42A96C0681B74838eC525AdbD13c37f66388f289
mFRAX	0x1C55649f73CDA2f72CEf3DD6C5CA3d49EFcF484C
mUSDC.wh	0x744b1756e7651c6D57f5311767EAFE5E931D615b
mETH.wh	0xb6c94b3A378537300387B57ab1cC0d2083f9AeaC
mWBTC.wh	0xaaa20c5a584a9fECdFEDD71E46DA7858B774A9ce
stkWELL	0x8568A675384d761f36eC269D695d6Ce4423cfaB1

Apollo Contracts

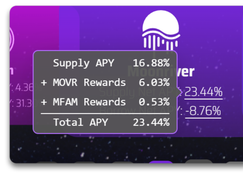
Name	Contract Address
Comptroller	0x0b7a0EAA884849c6Af7a129e899536dDDcA4905E
ChainlinkOracle	0x892bE716Dcf0A6199677F355f45ba8CC123BAF60
Maximillion	0x1650C0AD9483158f9e240fd58d0E173807A80CcC
EcosystemReserve	0xbA17581Bb6d89954B42fB84294e476e97588908B
EcosystemReserve Controller	0xD94F826C17e870a6327B7b1de6B43C5a9Ef21044
JumpRateModel	0xC862A3af64a8d3C146E6c505a18c2B6c6a6601bf
GovernorAlpha	0x2BE2e230e89c59c8E20E633C524AD2De246e7370
breakGlassGuardian	0x5DeD9d1025a158554Ab19540Ae83182d890Bb8DB
Timelock	0x04e6322D196E0E4cCBb2610dd8B8f2871E160bd7
mMOVR	0x6a1A771C7826596652daDC9145fEAaE62b1cd07f
mxcKSM	0xa0d116513bd0b8f3f14e6ea41556c6ec34688e0f
mETH	0x6503D905338e2ebB550c9eC39Ced525b612E77aE
mUSDC	0xd0670AEe3698F66e2D4dAf071EB9c690d978BFA8
mUSDT	0x36918B66F9A3eC7a59d0007D8458DB17bDffBF21
mFRAX	0x93Ef8B7c6171BaB1C0A51092B2c9da8dc2ba0e9D
mWBTC	0x6E745367F4Ad2b3da7339aee65dC85d416614D90
StakedMFAM	0xCd76e63f3AbFA864c53b4B98F57c1aA6539FDa3a

Deprecated Markets

mUSDC.mad	0x02e9081DfadD37A852F9a73C4d7d69e615E61334
mETH.mad	0xc3090f41Eb54A7f18587FD6651d4D3ab477b07a4
mBTC.mad	0x24A9d8f1f350d59cB0368D3d52A77dB29c833D1D
mBUSD.wh	0x298f2E346b82D69a473BF25f329BDF869e17dEc8

‼ Protocol Information

Each asset on Moonwell will have a breakdown of the rewards, as shown in the following diagram. For the purpose of this documentation, Moonriver (MOVR) asset was used as an example.



Protocol Math

Supply APY Equation

Supply APY

Supply APY is the interest rate paid to users supplying asset. Users earn a percentage in the asset supplied, which automatically accrues.

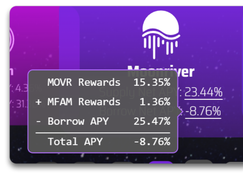
The amount is determined by the value in "Supply APY" – a rate that fluctuates depending on the rate of supplied assets vs borrowed assets (market utilization). In this example, a user supplying MOVR to Moonwell Apollo, may earn MOVR tokens.

A higher supply with low borrowing may have a lower supply APY, while a higher borrow with lower supply may have a higher supply APY in order to incentivize supplying.

Supply APY Example

Borrow APY

Borrow APY is the rate attached to users borrowing an asset, and indicates the rates a user will need to pay for their loan. The rate is currently offset by rewards from the distribution APY (currently rewarded in MOVR and MFAM on Apollo, GLMR and WELL on Artemis). In some rare situations, the rate may also be favorable enough so that users might earn rewards for borrowing in addition to supplying

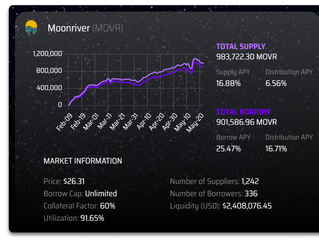


Borrow APY Example

Distribution APY

Moonwell currently offers MOVR and MFAM rewards on Apollo, and GLMR and WELL on Artemis. The rate of these rewards is broken down on the Moonwell page by hovering over the asset card. The combined rate is seen on the chart for the asset:

These rewards are adjusted based on market conditions, asset value, and reflect tokens provided through a grant through the First MOVR Advantage for MOVR rewards and Moonbeam Ignite for GLMR rewards; MFAM rewards as outlined in the Transparency Reports for Apollo, and WELL rewards for Artemis.



Distribution APY Example

Complex Example

Collateral Factor

The maximum that can be borrowed on a particular asset.

Collateral Factor Example

Reserve Factor

The percentage of the borrower's interest that goes to the Moonwell protocol reserves.

Reserve Factor Example

Close Factor & Liquidation Incentive

Close Factor

The maximum amount of an outstanding borrow that can be closed in a single liquidation event.

Close Factor Example

Liquidation Incentive

The additional collateral given to liquidators as an incentive to perform liquidations and keep the protocol solvent. A portion of this is given to the collateral mToken reserves as determined by the protocol seize share to reduce the risk of insolvency from cascading liquidations.

The Liquidation Incentive is equivalent to 10% of an underwater accounts' outstanding borrow, of which 30% returns to the protocol reserves.

Artemis Asset Risk Parameters

Asset	Collateral Factor	Reserve Factor	Close Factor	Borrow Cap
GLMR	62%	25%	50%	22,555,000 GLMR
xcDOT	63%	25%	50%	918,000 xcDOT
xcUSDT	53%	15%	50%	1,200,000 xcUSDT
FRAX	59.5%	15%	50%	5,250,000 FRAX
WETH.wh	50%	25%	50%	1100 ETH.wh
WBTC.wh	35%	25%	50%	110 WBTC.wh
USDC.wh	64%	15%	50%	2,600,000 USDC.wh
DEPRECATED				
BUSD.wh	0%	99%	50%	325,000 BUSD
WETH.mad	0%	100%	50%	Unlimited
WBTC.mad	0%	100%	50%	Unlimited
USDC.mad	0%	100%	50%	Unlimited

Artemis Interest Rate Curve Parameters

Asset	Base	Kink	Multiplier	Jump Multiplier
GLMR	0.02	0.6	0.15	3
xcDOT	0.02	0.6	0.15	3
WETH.wh	0.02	0.6	0.15	3
WBTC.wh	0.02	0.6	0.15	3
USDC.wh	0	0.8	0.05	2.5
FRAX	0	0.8	0.05	2.5
xcUSDT	0	0.8	0.05	2.5

Apollo Asset Risk Parameters

Asset	Collateral Factor	Reserve Factor	Close Factor	Borrow Cap
MOVR	60%	25%	50%	120,000 MOVR
xcKSM	60%	25%	50%	19,000 xcKSM
FRAX	58%	15%	50%	1,000,000 FRAX
WETH.multi	64%	25%	50%	700 WETH.multi
WBTC.multi	0%	25%	50%	4 BTC.multi
USDC.multi	68%	15%	50%	9,727,000 USDC.multi
USDT.multi	46%	15%	50%	600,000 USDT.multi

Apollo Interest Rate Curve Parameters

Asset	Base	Kink	Multiplier	Jump Multiplier
MOVR	0.02	0.6	0.15	3
xcKSM	0.02	0.6	0.15	3
WETH.multi	0.02	0.6	0.15	3
WBTC.multi	0.02	0.6	0.15	3
USDC.multi	0	0.8	0.05	2.5
USDT.multi	0	0.8	0.05	2.5
FRAX	0	0.8	0.05	2.5

mTokens

mTokens are an ERC-20 compliant token that is used to track positions within the Moonwell protocol across markets. mTokens are transferrable and fungible, and can be redeemed for an underlying position assuming there is sufficient liquidity and that position represented by those tokens haven't been marked for use as collateral.

Each market supported by the protocol (GLMR, Frax, etc) is represented by an mToken, and the mToken is responsible for:

- Taking new deposits (minting mTokens)
- Withdrawing assets from the protocol (redeeming/burning mTokens)
- Escrowing borrow requests for a specific asset
- Handling repayments for a specific market
- Handling liquidations for specific market positions
- Calculating reserve and seize shares to increase protocol reserves

When you liquidate an underwater loan on Moonwell, you pay off a user's underwater position and are able to seize a portion or the entirety of their mTokens.

Contract Interactions

Public Functions

Mint

This function is responsible for taking in ERC20 tokens or native assets and dispatching mTokens which represent the supplied position

Redeem

This function is responsible for taking in mTokens and sending out the underlying asset back to the caller

RedeemUnderlying

Works exactly like `Redeem`, but instead allows you specify the amount of underlying tokens you'd like back (saving you from having to calculate/use the mToken `exchangeRate`)

Borrow

Borrows a specific amount of assets from the protocol - only allowed if you have supplied (minted) collateral and called `enterMarket` with it, and you have sufficient liquidity to borrow the requested asset

RepayBorrow

Repays a borrow position the caller owns within the market

RepayBorrowBehalf

Repays a borrow position on behalf of someone else

LiquidateBorrow

Liquidates a borrower's position who is underwater/has a heath factor below 1. See **Liquidations** for more info

SweepToken

Sweeps accidentally sent ERC-20 tokens to the protocol `admin`

AddReserves

Add a specified amount of liquidity to the protocol's reserves

BalanceOfUnderlying

Get the balance of an mToken holder denominated in the underlying asset instead of the mToken balance (accruing interest and calculating the balance from the exchange rate)

GetAccountSnapshot

Gets a user's `mTokenBalance`, `borrowBalance`, and `exchangeRateMantissa` - useful for determining a user's current positions

BorrowRatePerTimestamp

Gets the borrower side interest rate per second on a specific market

SupplyRatePerTimestamp

Gets the supplier side interest rate per second on a specific market

TotalBorrowsCurrent

Gets the total amount of outstanding borrows for a market

Standard ERC-20 Functions

mTokens are ERC-20 complaint, so they expose the same interface around `transfer`, `balanceOf`, etc as any other ERC-20 token

Admin Functions

> Only the protocol admin can call these functions!

SetPendingAdmin / AcceptAdmin

A two step process to transfer admin control to an address

ReduceReserves

Withdraw reserve liquidity to the `admin` address

👮 The Comptroller

The Moonwell Comptroller is the central brain to the entire Moonwell protocol - it's responsible for the following functions

- Disbursing protocol rewards
- Determining liquidity for lenders based on price oracles
- Listing new markets
- Storing and Updating Risk Parameters
- Authorizing liquidations (determining if a position is liquidatable)
- Authorizing minting/supplying to the protocol
- Authorizing redemptions/withdraws from the protocol
- Authorizing loan repayments
- Authorizing mToken movements

More details can be found at the following sub-pages:

Contract Interactions

Risk Parameters

Guardian Roles

Contract Interactions

Public Functions

Enter Market

By calling this function you'll specify the mTokens that you wish to "enter" the market for. This marks your supplied position in a market to be used for collateral calculations, and is a pre-requirement for borrowing assets from the protocol.

Exit Market

Does the opposite of "Enter Market" - marks a supplied position as no longer to be used for collateral calculations. Can not occur while actively borrowing against this market and cannot succeed if doing so will plunge your position underwater.

Claim Reward

The Claim Reward function in the comptroller sends the invoker the owed protocol rewards for their supply/borrow activity.

Admin Functions

> ⚠ Only the protocol admin can call these functions!

Set Price Oracle

Sets the price oracle to be used by the protocol

Set Pause Guardian

Sets the address of the pause guardian used across the protocol

Set Gas Amount

Sets the amount of gas sent with any native token sends from the protocol, useful to support things like Gnosis Safe, but should be set to the minimum threshold necessary to hedge against reentrancy issues.

Set Mint Paused

Disables minting for a specified market

Set Borrow Paused

Disables borrowing for a specified market

Set Transfer Paused

Disables transfers of mTokens between two addresses. Does *NOT* prevent supplying, borrowing, repaying, or liquidations.

Set Seize Paused

Disables liquidations across the entire protocol

Set WELL Address

Sets the WELL address used for sending out rewards

GrantWell

Sends a specific amount of well from the comptroller to a specified address

🚦 Risk Parameters

The following **risk parameters** are adjustable within the Moonwell Comptroller. Most of these correspond with the Comptroller Storage variables.

Collateral Factor

The Collateral Factor is the percentage of a supplied asset that is treated as liquidity, and is set **per market**. If `CF=0.6` for an asset, then every $1 of that asset supplied allows for $0.60 of borrowing power within the protocol.

Stored in the `collateralFactorMantissa` variable and set via this function.

Price Oracle

This is the price oracle contract address used by the protocol to determine the prices of specific assets within the market.

Stored in the `oracle` variable, and set via this function.

Close Factor

The Close Factor is the percentage of a specific borrowers position that can be liquidated in one transaction.

Stored in the `closeFactorMantissa` variable and set via this function.

Liquidation Incentive

The Liquidation Incentive is the discount on collateral that a liquidator receives for liquidating a position.

Stored in the `liquidationIncentiveMantissa` variable and set via this function.

Market Borrow Cap

The Market Borrow Cap is the maximum amount of borrows allowed for a specific market.

Stored in the `borrowCaps` variable and set via this function.

Gas Amount

The Comptroller Gas Amount parameter sets the amount of gas that a transaction to send the native token on a chain will use - this is helpful to enable things like withdraws to smart contracts like Gnosis Safe, but if set too high can be an avenue for re-entrancy issues within the protocol.

Stored in the `gasAmount` variable and set via this function.

Reward Speed

The Comptroller is responsible for disbursing rewards, and supports different emission speeds for borrowers and suppliers. These speeds are set per market and per emission token where `rewardType 0 = WELL`, and `1 = GLMR`

The reward speeds for a market are stored in the `supplyRewardSpeeds` and `borrowRewardSpeeds` variables and set via this function.

Guardian Roles

The Comptroller has a number of guardian roles within it, each that can be set to distinct addresses and fulfill specific security roles within the protocol.

Mint Guardian

The Mint Guardian is responsible for pausing minting (supplying assets) for a specific market within the protocol. This is useful if you ever want to stop liquidity from entering the markets.

Borrow Guardian

The Borrow Guardian is responsible for pausing borrowing for a specific market within the protocol. This is useful if you ever want to stop assets from being borrowed from a specific market.

Seize Guardian

The Seize Guardian is responsible for pausing liquidations **across the entire protocol**. This is useful if you ever want to stop liquidations from occurring, but keep in mind that liquidations are part of the protocol staying solvent and removing debt that's at risk of becoming uncollateralized from the system.

Transfer Guardian

The Transfer Guardian is responsible for halting transfers of mTokens between addresses. The mTokens you're given in exchange for supplying liquidity are a representation of your position within the Moonwell protocol, and since they're standard ERC-20 tokens they can be transferred just like any other token. This pause guardian stops all transfers though, which might be helpful if there's contagion risk via this mechanism for some reason. It is important to note that this does not stop minting, borrowing, redeeming, or liquidations within the protocol since those use different logical paths with their own guardians.

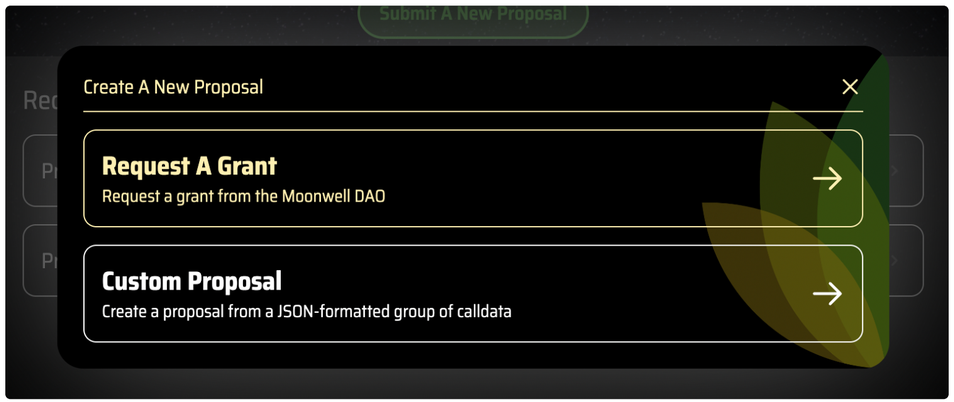
Borrow Cap Guardian

The Borrow Cap Guardian is responsible for setting the borrow caps on a specific market via the _setMarketBorrowCaps function. This is useful if you wanted to delegate the adjustment of the borrow caps across markets to someone who specializes in these sorts of adjustments without giving them administrative access across the rest of the protocol.

📜 Governance Proposal Generator

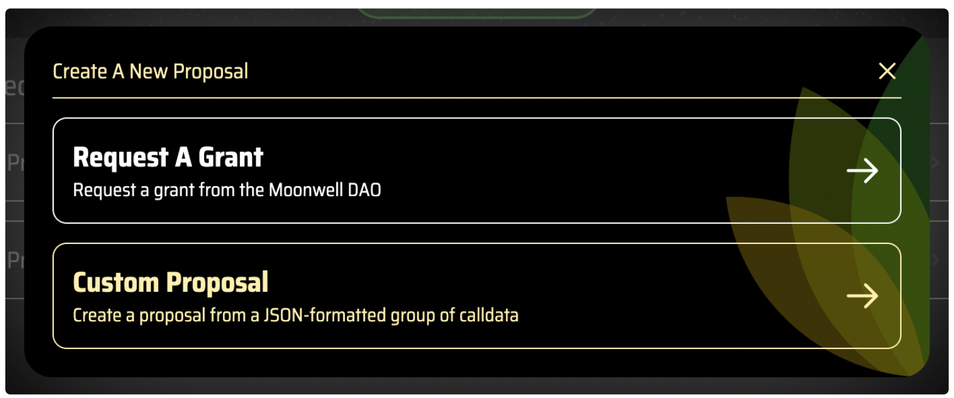
There are currently a couple ways to generate proposals within the Moonwell governance ecosystem

When you want to apply for a grant, you can click on the "**Request A Grant**" workflow.



Requesting A Grant

If you want to submit a custom proposal to do other/arbitrary things, you can use the "**Custom Proposal**" workflow.



Submitting a Custom Proposal

Custom Proposal

The Custom Proposal workflow allows you to submit a JSON bundle of contract calls as part of your proposal, and have them be used when submitting your proposal on-chain.

An example of where this might be useful would be MIP-2, which makes a number of adjustments to the protocol's risk parameters, as well as pays out **100,000 WELL** to the proposal submitter. https://gov.moonwell.fi/proposal/2

JSON Parameters

Each proposal needs to populate an array of values - `targets`, `values`, `signatures`, and `callDatas`. Proposal steps are combined based on index, so the first call will be `targets[0]`, `values[0]`, `signatures[0]`, and `callDatas[0]`, the second call will be `targets[1]`, `values[1]`, `signatures[1]`, `callDatas[1]`, etc.

Key Name	Usage
`targets`	An array of contract addresses to interact with - Ex: `["0x091608f4e4a15335145be0A279483C0f8E4c7955", "0xffffffff1fcacbd218edc0eba20fc2308c778080"]`
`values`	An array of NATIVE (`GLMR` / `MOVR`) values to send along with the request. Usually is set to 0 - Ex: `[0, 0]`
`signatures`	An array of function signatures for a target and callData. Ex: `["transferFrom(address,address,uint256)", "transferFrom(address,address,uint256)"]`
`callDatas`	The ABI encoded callData for a function signature - Ex: `["0x000000000000000000000000000000000000000000000000375b83deb52b6eb740000", "0x000000000000000000000000000000000000000000000000375b83deb52b6eb740000"]`

An example blob would look something like this, which is a single `transferFrom` request that will send `4,182,693 WELL` from `6972f25ab3fc425eaf719721f0ebd1cdb58ee451` to `7793e08eb4525309c46c9ba394ce33361a167ba4`:

```
{
  targets: [
    "0x511aB53F793683763E5a8829738301368a2411E3"
  ],
  values: [
    0
  ],
  signatures: [
    "transferFrom(address,address,uint256)"
  ],
  callDatas: [
    "0x000000000000000000000000000000000000000000000000375b83deb52b6eb740000"
  ]
}
```

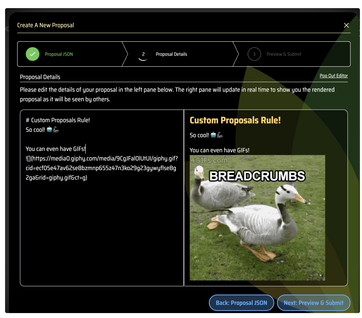
In order to generate a payload like this, you can use the following example for how to format things:

```
const ethers = require('ethers');
const moonwellJS = require('@moonwell-fi/moonwell.js');

const WELL_RECIPIENTS = {
    '0xa320c822A8e62d75cc6020E3c878931f122077': 100_000,
    '0x8b61D581f826F91854Fd3dCfdFd1408A7c59d249': 250_000,
}

// Generate a payload to send the specified amounts of WELL to the desired recipients (WELL_
;(async () => {
    // JSON payload
    const payload = {
        targets: [], signatures: [], values: [], callDatas: [],
    }
    // 18 digit mantissa
    const mantissa = ethers.BigNumber.from(10).pow(18)

    const wellContract = new ethers.Contract(
        // Get WELL token address from moonwell.js
        moonwellJS.moonwellContracts.moonbeam.GOV_TOKEN,
        // Fill this in with a full ABI from moonscan for other functions!
        ["function transferFrom(address from, address to, uint256 value) returns (bool)"],
    )

    for (const [wallet, amount] of Object.entries(WELL_RECIPIENTS)){
        // Use `populateTransaction` to just return an encoded transaction without sending t
        const populatedTx = await wellContract.populateTransaction.transferFrom(
            '0xf130e4946F862F2c6CA3d007D51C216B8908e006', // DEV GRANT Multisig
            wallet,
            ethers.BigNumber.from(amount).mul(mantissa)
        )

        // Target == WELL token
        payload.targets.push(wellContract.address)
        // Values = 0 GLMR sent
        payload.values.push(0)
        // Signatures = Get signature for `transferFrom` function
        payload.signatures.push(
            wellContract.interface.getFunction('transferFrom').format()
        )
        // CallDatas = Get call data for arguments.
        // IMPORTANT: You need to slice off the function selector from your call args
        payload.callDatas.push('0x' + populatedTx.data.slice(10))
    }

    // Log formatted proposal JSON
    console.log( JSON.stringify(payload, null, 2) )
})();
```

Proposal Details

Once you've generated and added your proposal JSON, you'll be taken to the editor to wordsmith your proposal. The editor accepts markdown and will give you an immediate preview on what things will look like when submitted.

> If you need more space, click the "Pop Out Editor" for a full-screen markdown editor.



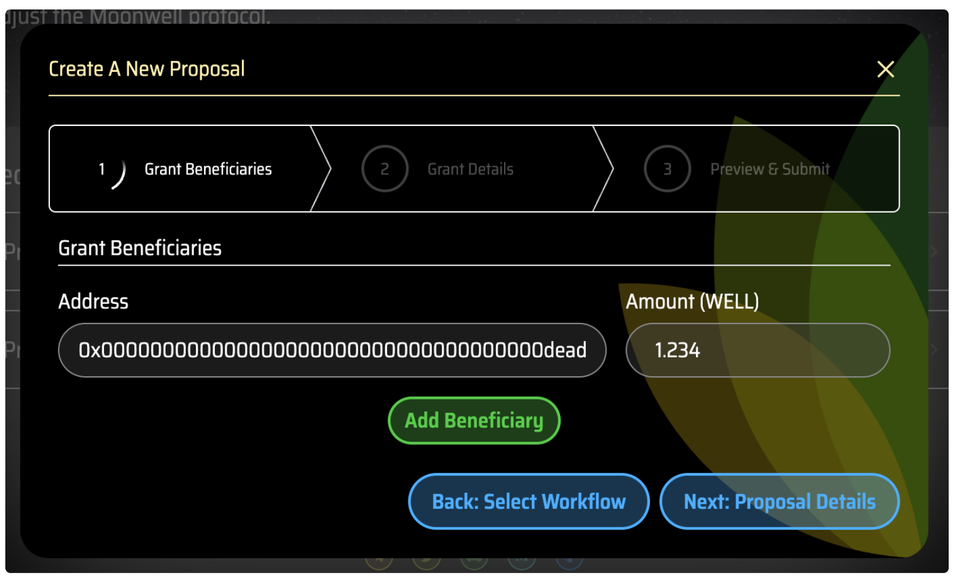
Once things look good, the last step is to preview your proposal, which will give you a detailed breakdown of the calls made and a formatted view of your proposal.

If things look good there, you can click "Submit Proposal For Voting", which will make a metamask request and broadcast your custom proposal to the DAO for a vote!

🙏 Requesting A Grant

In order to request a grant, you can follow the "Request a Grant" workflow

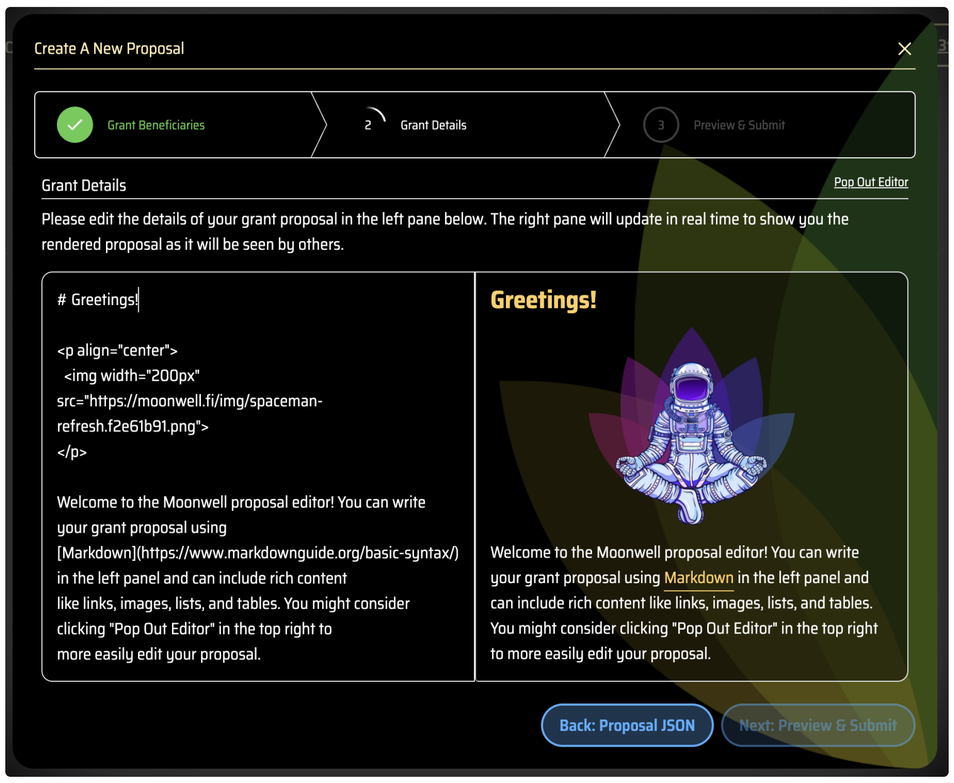
On the first screen you can add any number of grant beneficiaries for the submitted proposal.



Proposal Details

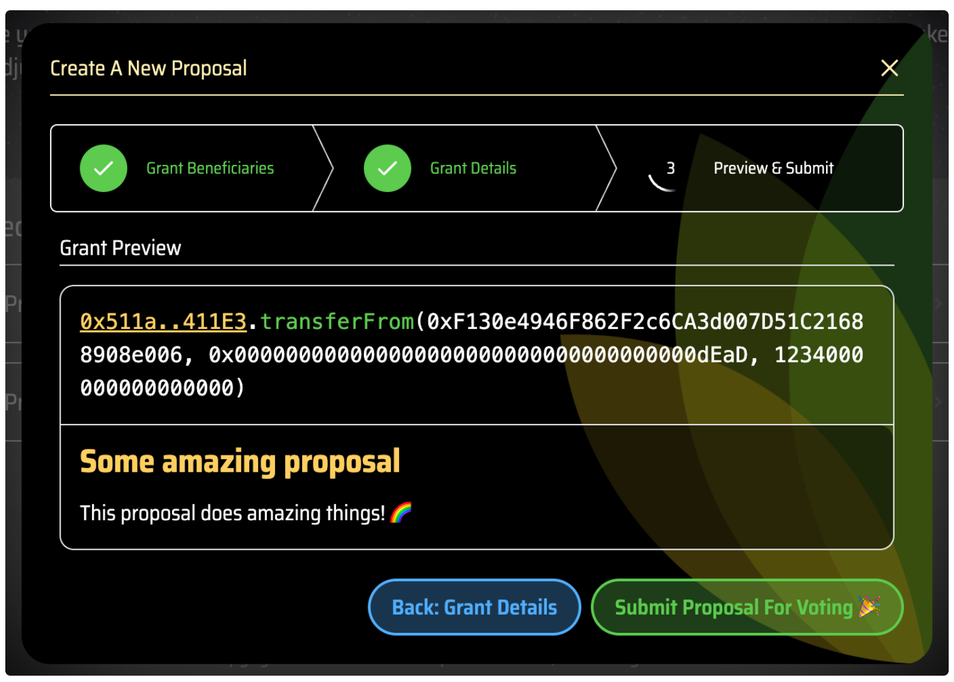
Next, you can type out the details of your proposal using the Moonwell Proposal Editor. It supports Markdown, and gives you immediate feedback for how the proposal will be rendered.

> ⓘ If you need more space, click the "Pop Out Editor" for a full-screen markdown editor.



The only thing left to do is to preview your grant proposal and make sure things look correct.

> ⓘ If you want help double checking the actual contract calls as part of your proposal, feel free to come to our discord and we'll be happy to take a look!



Once you click "Submit Proposal For Voting", you'll be prompted to sign a transaction to broadcast your grant proposal on-chain!

Moonwell Artemis GraphQL

The Graph Protocol is a decentralized network dedicated to indexing and serving up blockchain data as events happen on-chain. The Moonwell frontend uses this extensively to speed up load times and make a better use experience (though it's also designed to still work if the subgraph is unavailable for some reason).

A Moonwell community member/contributor using the handle `0xbe1` did the lion's share of the work and you can find the subgraph for Moonwell Artemis deployed at https://thegraph.com/hosted-service/subgraph/moonwell-fi/moonwell-moonbeam.

Likewise, if you want the Moonwell Apollo/Moonriver subgraph you can find it here https://thegraph.com/hosted-service/subgraph/moonwell-fi/moonwell-moonriver

All updates to the subgraph are done in a public Github repo at https://github.com/moonwell-fi/moonwell-subgraph/

🏃 Getting Started

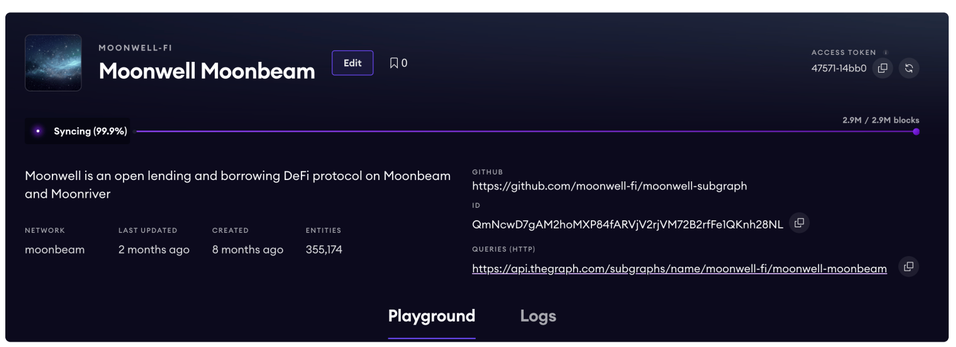
🧮 Tutorial: Querying Liquidations

🤖 Tutorial: Querying the subgraph programatically

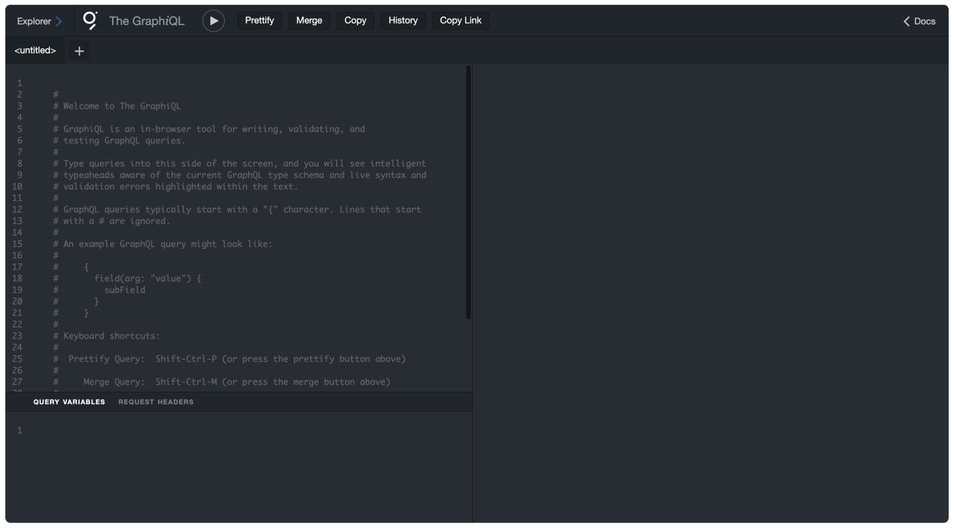
🏃 Getting Started

Getting started with the Moonwell Subgraph is easy thanks to some spectacular tooling offered by the graph protocol.

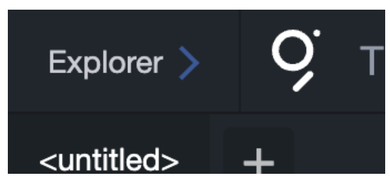
There are a number of ways to interact with the subgraph, but the easiest way to start is by simply visiting the "queries" URL by following the "queries" URL in the bottom right on the subgraph's page found here - https://thegraph.com/hosted-service/subgraph/moonwell-fi/moonwell-moonbeam

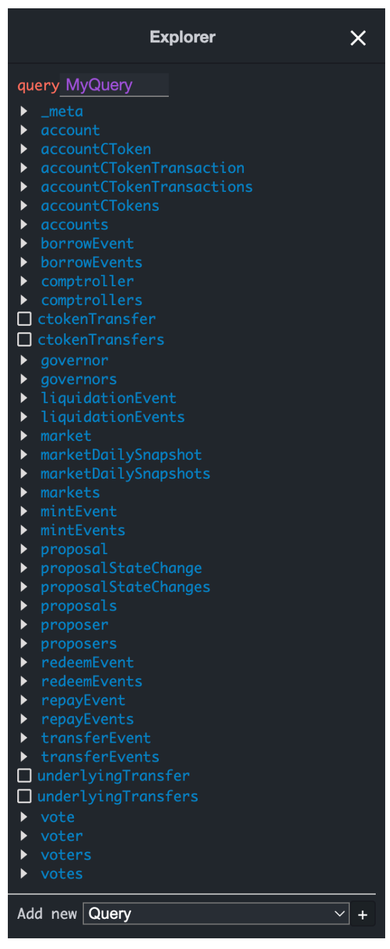


When you click on that, you're presented with the following interface, which gives you a mini-tutorial about how to write queries against the data that's indexed.



But how to find what queries are possible? If you click the "Explorer" button in the top left, you'll get a list of all data that's indexed by this subgraph.





This can be somewhat confusing to work with the first time you use it, so the following page is a simple tutorial about using these tools to accomplish something in a real way.

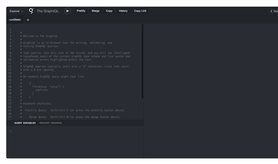
Tutorial: Querying Liquidations

This small tutorial outlines the process that a developer might take if they wanted to query some piece of data indexed by the subgraph.

For this example we'll be querying a list of liquidations and ordering them by date (with the latest liquidations showing first), but this process is a generic workflow that can apply to basically any data you may want to query.

First Steps

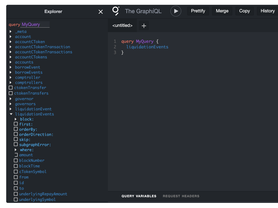
First, we need to visit https://api.thegraph.com/subgraphs/name/moonwell-fi/moonwell-moonbeam/graphql which will look like this (note you may have previously-cached queries displayed).



From there, pop out the "Explorer" tool, giving you a view like this

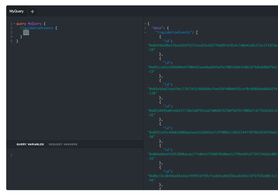
On the *right*, if you go down the list you can see both `liquidationEvent` and `liquidationEvents`. The difference between the two is that you can use `liquidationEvent` to search for a particular event *by ID*, while `liquidationEvents` gives you the ability to query *all* events in the database.

For our usecase, since we're looking for *all* liquidations ordered by date, we'll be using `liquidationEvents`. Clicking on that expands the object's fields, and pre-populates the query for you.



First Query

At this point if you click "go" (looks like ▶ at the top of the interface) then it'll fire off this query and return you a list of found objects. In this case it'll just return each object's ID and nothing else, in some undetermined order. You may see that it auto-fills `id` in your query since we never asked it to retrieve any fields.



Great, so we have a list of IDs, and if you look closely at them, they are actually in the format `${TX_HASH}-${transactionLogIndex}`. You can see how those IDs are constructed in the subgraph processing logic itself here - https://github.com/moonwell-fi/moonwell-subgraph/blob/ea823807632e689a93cdd5a5752b7b1078b9453c/src/ctoken.ts#L298-L301.

So if we wanted to look at the first liquidation returned on moonscan, we'd copy the first half of the ID and paste it into moonscan like this - https://moonscan.io/tx/0x0244be9be1fbae918f52f3ceaa51b3d377bb851fef814c7e0a42a9c57ec1f1870e

Using other tools can be incredibly helpful to validate what you think you're seeing, other contextual data that's not captured by the subgraph, etc.

More Data

Back to the task at hand, what other data would we like instead of just the ID? Here are our options:

Ignoring the top part in bold, we see:

- amount
- blockNumber
- blockTime
- tCollateralTokens
- tokenSymbol
- from
- to
- underlyingRepayAmount
- underlyingSymbol

So let's check all of those and click go again and see what happens:

Much more useful data comes back this time!

Controlling the query

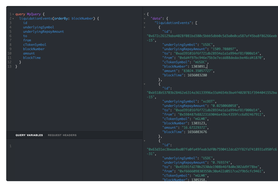
Now that we can go find the fields we want, we also want to filter things, order things, etc. This can be done using the fields that are bold. Some are complicated/nested queries like `block`, while others are simple, like `orderBy`).

Since we're trying to get a list of liquidations ordered by when they happened, the `orderBy` field is the one we're looking for. Clicking that, we get a list of fields we can order our query by. Since we're interested in *historical* time-based ordering, either `blockNumber` or `blockTime` should work for our needs.

By selecting things on the left, the query is auto-updated to order things by block number now.

```
liquidationEvents(orderBy: blockNumber) {
```

Re-running our query, we see that now things are sorted by `blockTime`!



Multiple filters

One small issue though, this shows us *all* liquidations since the start of the protocol, in order. We could paginate through the results (more on this later), but a better solution would be to add another filter to the query, specifically a `orderDirection`, which defaults to ascending.

Following the same process as before, we'll add in the `orderDirection` parameter, and click on `desc` to signify that we want results ordered by block number, but with the largest block number first (i.e. latest events first).

Running our revised query indeed gives us our list of liquidations by block number with the latest ones being first. Feel free to experiment with other filters, things like `where` are extremely powerful and let you write queries that pare down the entire dataset to just what you're looking for.

Extra: Querying more than 100 results

This query works great! However, for performance reasons the maximum returned results for any given query is 100 items. What if we wanted to go get the prior 250 liquidations that have happened? Or the entire list? That's where the `skip` parameter comes in.

For demonstration purposes I'll be using `skip: 1` but in the real world you'd likely use `skip: 100` and recursively query the data store, breaking when you receive a page of less than 100 results.

As with the other queries by clicking "skip" on the left, and putting 1 in the field it'll auto-update our query, and when we click "execute" the first result will be skipped, showing us items `2-101` instead of the default `1-100`.

🤖 Tutorial: Querying the subgraph programatically

The previous tutorial covered the process of *constructing* a query that provides useful data, but what if you want to actually use it in a program?

This tutorial will cover what it looks like to take a query from the user-friendly tooling and use it directly in your application.

The following query (taken from the previous tutorial) will be used, and will display the most recent 100 liquidations within the protocol:

```
query queryLiquidations {
  liquidationEvents(orderBy: blockNumber, orderDirection: desc) {
    id
    underlyingSymbol
    underlyingRepayAmount
    to
    from
    cTokenSymbol
    blockNumber
    amount
    blockTime
  }
}
```

Interacting programmatically

To actually send requests you'll need to use the following URL (`subgraphURL` below):

https://api.thegraph.com/subgraphs/name/moonwell-fi/moonwell-moonbeam

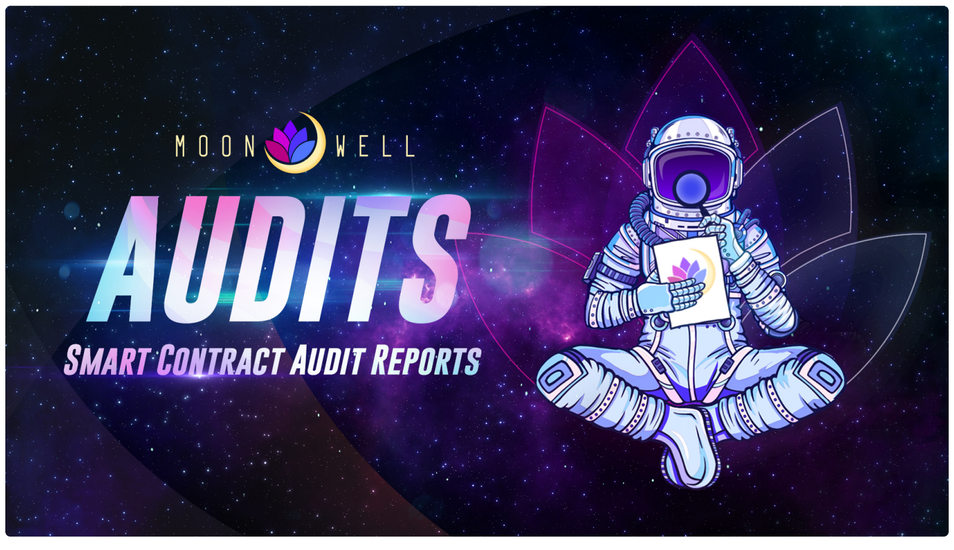
Below are some reference implementations that should be helpful, if you'd like to see a language not listed here come request it in the discord!

Javascript | Python

```
// Query GraphQL from Javascript using the Axios library

const axios = require("axios")

const subgraphURL = 'https://api.thegraph.com/subgraphs/name/moonwell-fi/moonwell-moor
const query = `
  query queryLiquidations {
    liquidationEvents(orderBy: blockNumber, orderDirection: desc) {
      id
      underlyingSymbol
      underlyingRepayAmount
      to
      from
      cTokenSymbol
      blockNumber
      amount
      blockTime
    }
  }
`

async function fetchResults(){
  const result = await axios.post(
    subgraphURL,
    { "query": query }
  )
  // Do something with `result.data.liquidationEvents` here
  console.log(result.data)
}

// Kick off the fetchResults async function
fetchResults().then(() => {
  console.log("Done!")
})
```

🔎 Audits

Smart Contract Audit Reports Conducted by Halborn Security



Moonwell smart contracts **always** undergo a comprehensive audit by Halborn Security prior to deployment, with the aim of significantly reducing the likelihood of security vulnerabilities in the protocol

Audit Reports

Moonwell Apollo v1

- Initial deployment of the Moonwell Apollo protocol on Moonriver

Moonwell Apollo v2

- Current deployment of the Moonwell Apollo protocol
- Added governance module and updated comptroller

Moonwell Artemis v1

- Initial deployment of the Moonwell Artemis protocol on Moonbeam

Moonwell Artemis v2

- Current deployment of the Moonwell Artemis protocol
- Added governance module and updated comptroller

Moonwell Safety Module Audit Report

- Currently deployed iteration of the Safety Module smart contract

Github Releases

You can access both present and previous versions of the Moonwell protocol, along with their corresponding smart contracts, in the releases section of the Moonwell Github repository.

To verify the information, cross-reference the Commit IDs mentioned in Section 1.4 of the audit reports with the commits available on Moonwell's Github page.

Bug Bounty

Moonwell leverages Immunefi for its bug bounty program.

Risks

Moonwell is committed to reducing the risks associated with smart contracts and liquidations. To achieve this, the protocol's smart contracts undergo ongoing audits by Halborn Security.

However, it is important to note that no cryptocurrency protocol can be entirely risk-free, and inherent risks remain in relation to smart contracts and liquidations.