HYDRAPAD

WWW.HYDRAPAD.COM

A platform for web3 builders

HYDRAPAD

A Next Gen Launchpad

Gateway to onboard Users & Projects in web3 space

Whitepaper

www.hydrapad.com

hello@hydrapad.com

Abstract

Hydra Launchpad is integrated with top tier security features with Bonded Curve Presale Contract with live trading for users & investors. A Bonded Curve Presale Launchpad differs from a Normal Presale Launchpad primarily in how the token price behaves during the presale and the mechanics of token distribution. While both types of launchpads help raise funds for blockchain projects by allowing early investors to buy tokens before they are listed publicly, their key difference lies in the pricing model, investors benefits and incentive structure. Our platform addresses key pain points in blockchain fundraising for startups, investors, and developers.

Product Scope

- **Presale Launchpad:** A simple and intuitive interface for startups to create, manage, and customize presales for their tokens, enabling projects to raise funds in a decentralized manner while ensuring transparency and fairness.

- **Bonded Curve Mechanism:** The bonded curve will ensure a fair pricing structure during token presales. The curve dynamically adjusts the price based on the quantity of tokens sold, incentivizing early participation while maintaining price stability and reducing manipulation.

- **Live Token Trading:** Investors can trade presale tokens in real-time, allowing for instant liquidity during the presale phase, providing a market-driven approach that benefits both the startup and the investor.

- **Security and Transparency:** Hydrapad ensures transparency through decentralized smart contracts that govern token sales, reducing risks of fraud and increasing user trust. All transactions are verifiable and auditable on the blockchain.

- **Token Creation:** Startups can easily create custom tokens for their projects directly through the Hydrapad platform. This feature enables the creation of tokens with predefined tokenomics, facilitating the setup of presales in a seamless and user-friendly manner.

- **Token Locking:** Projects and investors can lock tokens for a defined period, enhancing trust and preventing early sell-offs. Token locking can be applied to both presale funds and tokens issued during the presale phase, ensuring long-term commitment from the project team and investors.

- **Portfolio Management for Users:** Users can manage and track their investments and token holdings in a comprehensive portfolio dashboard. This feature will provide real-time updates on presale participation, token performance, and overall portfolio value, making it easier for users to monitor their assets.

- **Airdrops:** Hydrapad allows users to easily conduct token airdrops as part of their marketing and community-building strategies. This feature allows projects to distribute tokens to a targeted group of users, helping to build momentum and reward loyal supporters.

- **Secure Liquidity Management:** Hydrapad incorporates secure liquidity management tools, enabling projects to provide liquidity pools for their tokens and ensuring that tokens are readily available for trading. This feature also allows projects to manage the liquidity of presale funds in a secure, transparent manner, ensuring proper token flow and liquidity during the presale phase.

Problem Statement

Many startups face challenges when raising initial capital, especially in decentralized ecosystems where token presales can be manipulated by large investors or whales. Traditional presale methods suffer from inefficiencies, such as front-running and unequal access to token allocations. Moreover, investors are often uncertain about the token pricing mechanisms and their long-term value.

- **Liquidity Issues:**
Traditional presales often lack liquidity, as tokens are locked until later stages. Our platform allows live trading of presale tokens, providing liquidity and flexibility to investors.

- **Volatile Pricing:**
Presale token prices can be unpredictable and manipulated. We solve this with a bonded curve pricing model that gradually adjusts token prices based on supply and demand, ensuring fairness and stability.

- **Fees/cost:**
Early fundraising can be costly for startups building their own project. Our platform ensures low to No fees for presale creation and scalable transactions, making presales more accessible.

- **Lack of Transparency:**
Many investors face trust issues in presales. Our transparent bonded curve model ensures fair pricing, fostering trust among participants.

Solution & Benefits

- **Decentralized Capital Raising:**
The Bonded Curve Presale contract is a trustless and decentralized way to raise funds. It eliminates the need for centralized authorities or intermediaries, empowering project creators to directly raise funds from their community.

- **Global Accessibility:**
Anyone with an internet connection can participate in these presales, providing global access to Web3 fundraising. This democratizes the ability for users to support and invest in projects regardless of geographic location.

- **Building Communities Around Projects:**
The contract encourages community-driven participation, as early backers are rewarded with lower prices and a stronger stake in the project's success. This builds an invested community around

the project that has both financial and reputational incentives to see it succeed.

- **Integration with DeFi:**
The contract integrates seamlessly with DeFi protocols like Uniswap and Balancer, which are foundational to the Web3 ecosystem. By creating liquidity pools post-presale, the project ensures that tokens have immediate utility and market value, fostering broader DeFi adoption.

Automatic Liquidity Pool Creation

- **Ensuring Liquidity:**
One of the major challenges in DeFi projects is liquidity. The contract automatically wraps the ETH raised and pairs it with the presale token to create a liquidity pool on decentralized exchanges. This feature ensures that the token is tradable immediately after the presale ends, providing liquidity to the market and enabling price discovery.

No Control Over Funds by the Creator (No Centralized Control)

- **Presale Funds Are Secure:**
In a typical rug pull, the project creator retains control over the raised funds and may drain them at any time. In this contract, the funds raised (in ETH) are not directly controlled by the project creator; rather, they are securely handled by the contract itself.

- **Automatic Fees and Refunds:**
The presale includes automated functions for fee collection, token distribution, and refunds. If the presale fails (e.g., fundraising target is not met), users can automatically reclaim their ETH in proportion to their contribution, preventing the project creator from absconding with the funds.

KPIs for Impact Estimation:

We believe that our Dapp integration will bring measurable value to both our platform and your ecosystem. Key performance indicators (KPIs) to estimate the impact include:

● **Transaction Volume:** Volume-based metrics such as the Total Gross Transaction Volume (GTV) and send/receive transaction volume on the platform.

● **User Growth:** We project to onboard 100,000 active users by the end of Year 1, including investors and project teams, driving adoption of presale transactions.

● **Liquidity and Trading Activity:** We are expecting that new projects launch will lead to increased trading volume and liquidity within your ecosystem.

Development Roadmap

Phase 1 - Research & Design (Completed)

- Finalize the project architecture and design
- Research bonding curve mechanisms and presale dynamics
- Collaborate with ecosystem developers for integration strategies

Phase 2 - Initial Platform integration (5 weeks)

- Develop the functionalities of front-end user interface and backend infrastructure.

Phase 3 - Contract Development & Security Audits(5 weeks)

- Develop the bonded curve mechanism into the presale contract
- Factory Contract development for Token Creation

- Regular Presale Contract
- Locking/Vesting Contract
- Airdrops Contract
- Conduct initial tests with selected partners and projects
- Begin user testing of the presale and trading platform

Phase 4 – Beta Testing (4 weeks)

- Launch beta platform for public use
- Perform extensive security and performance testing

Phase 5 – Mainnet Launch & Community Engagement (2 weeks)

- Finalize platform for mainnet launch
- Focus on user acquisition, education, and community building
- Host a series of AMAs and events to promote Hydrapad and educate the ecosystem about bonded curve presale models

Bonded Curve Math

A Bonded Curve is a pricing mechanism where the price of the token increases as more tokens are purchased. It can be mathematically represented by a curve that is dependent on the total supply of tokens (i.e., the amount of tokens in circulation).

The general form of a Bonded Curve pricing function is:

P(S)=a×SbP(S) = a \times S^bP(S)=a×Sb

Where:

- P(S)P(S)P(S) is the price of the token when the total supply is SSS.
- aaa is a constant that defines the initial price when S=1S = 1S=1.

- bbb is a parameter that controls the steepness of the curve. If b>1b > 1b>1, the price increases more rapidly as tokens are purchased.

Example: Linear Bonding Curve

For simplicity, let's assume a linear bonded curve where the price increases by a fixed amount per token.

P(S)=a+b×SP(S) = a + b \times SP(S)=a+b×S

Where:

- P(S)P(S)P(S) is the price at supply SSS.
- aaa is the base price.
- bbb is the price increment per token.

This linear curve means that each subsequent token purchased increases the price by the same fixed amount.

Example: Exponential Bonding Curve

For more complexity, an exponential curve could be used, where the price increases more sharply as more tokens are purchased.

P(S)=a×eb×SP(S) = a \times e^{b \times S}P(S)=a×eb×S

Where:

- P(S)P(S)P(S) is the price at supply SSS.
- eee is Euler's number (~2.718).
- aaa is a constant.
- bbb controls the growth rate.

This exponential model will have a slower initial price increase, but as the supply grows, the price can increase significantly.

=================

The math behind the bonding curve are based on an integral of a line that starts from (0,0).

The line's equation is l=a*s, where a is constant and s in the supply.

To determine the price someone needs to pay in order to change the supply from s1 to s2 we calculate the integral of l from s1 to s2.

So, P = (a/2) * (s1 + s2) * max(s2-s1, s1-s2)

line (l)

P (The price you pay
to change the supply
from s1 to s2)

l=a*s

a

s1   s2   supply (s)
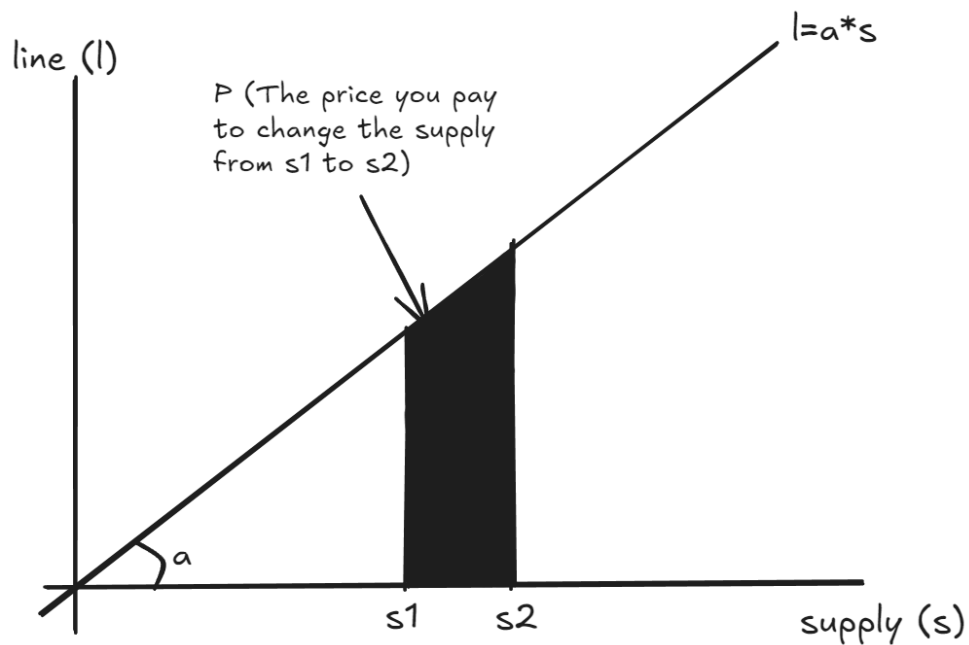
Smart Contract Functions with Security Features

Below are the basic smart contract functions that could be used to implement the Bonded Curve presale, along with key security features to protect users and ensure a secure launch:

```
uint256 public totalSupply;
uint256 public priceBase;
uint256 public priceIncrement;
uint256 public presaleEndTime;
address public owner;
mapping(address => uint256) public userBalances;
```

totalSupply: The total amount of tokens sold in the presale.

priceBase: The initial price of the token.

priceIncrement: The rate at which the price increases with each token purchase.

presaleEndTime: The time the presale will end.

owner: The address of the contract owner.

userBalances: A mapping to store the balance of tokens purchased by each user.

Constructor

The constructor sets the initial values for the presale, such as the price base, price increment, and end time.

```
constructor(uint256 _priceBase, uint256 _priceIncrement, uint256
_presaleDuration) {
    priceBase = _priceBase;
    priceIncrement = _priceIncrement;
    presaleEndTime = block.timestamp + _presaleDuration;
```

```
    owner = msg.sender;
    totalSupply = 0;
}
```

Bonded Curve Calculation Function

This function calculates the price of the token based on the
total supply using a linear formula for simplicity:

```
function getPrice(uint256 _supply) public view returns (uint256) {
    return priceBase + priceIncrement * _supply;
}
```

For an **exponential curve**, you could use:

```
function getPrice(uint256 _supply) public view returns (uint256) {
    return priceBase * (e ** (priceIncrement * _supply));
}
```

Buy Tokens Function

This function allows users to buy tokens at the current price
based on the total supply. It ensures that the presale is not
over, and it updates the user's balance.

```
function buyTokens(uint256 _amount) public payable {
    require(block.timestamp < presaleEndTime, "Presale has ended");
    uint256 currentPrice = getPrice(totalSupply);
    uint256 cost = currentPrice * _amount;

    require(msg.value >= cost, "Insufficient funds");

    userBalances[msg.sender] += _amount;
    totalSupply += _amount;

    // Refund any excess ETH
    if (msg.value > cost) {
        payable(msg.sender).transfer(msg.value - cost);
```

```
    }
}
```

Security Considerations:

- Ensure that `msg.value` is checked against the calculated cost to prevent underpayment.
- Refund any overpayment to ensure fairness.
- Prevent manipulation of the total supply by locking presale contract functions post-sale.

Token Locking & Vesting

To prevent rugpulls and sudden selling of tokens, use a **vesting schedule**.

```
mapping(address => uint256) public lockedTokens;
mapping(address => uint256) public vestingStartTime;
function startVesting(address _user, uint256 _amount, uint256
_vestingDuration) external onlyOwner {
    require(userBalances[_user] >= _amount, "Insufficient balance");
    lockedTokens[_user] += _amount;
    vestingStartTime[_user] = block.timestamp;
}
function releaseVestedTokens() public {
    require(lockedTokens[msg.sender] > 0, "No tokens to release");
    uint256 elapsedTime = block.timestamp - vestingStartTime[msg.sender];
    uint256 vestedAmount = (lockedTokens[msg.sender] * elapsedTime) /
vestingDuration;
    require(vestedAmount <= lockedTokens[msg.sender], "All tokens are not
vested yet");

    userBalances[msg.sender] += vestedAmount;
    lockedTokens[msg.sender] -= vestedAmount;
}
```

Security Considerations:

- Ensure only the owner can start the vesting process.
- Users can only withdraw tokens in accordance with the vesting schedule.

Anti-Rugpull Measures

- **Liquidity Lock**: Lock the liquidity to ensure that the project cannot pull the liquidity.
- **Ownership Transfer**: Transfer ownership of the presale contract to a multisig wallet or community governance mechanism after the presale ends.

```solidity
function lockLiquidity(uint256 _amount) external onlyOwner {
    // Ensure liquidity is locked by sending it to a secure wallet or smart contract
    // Use a smart contract or multisig wallet to lock liquidity for a specified period
}
```

Access Control & Security

- **Modifiers**:
    - **onlyOwner**: Ensures that only the contract owner can perform certain actions (e.g., liquidity lock, end presale).

```solidity
modifier onlyOwner() {
    require(msg.sender == owner, "Not authorized");
    _;
}
```

Reentrancy Protection: Use **ReentrancyGuard** to prevent reentrancy attacks during fund transfers.

```solidity
import "@openzeppelin/contracts/security/ReentrancyGuard.sol";
```

```solidity
contract Presale is ReentrancyGuard {
    // Your contract code here
}
```

The **Bonded Curve** pricing mechanism is a great tool for launching tokens in a fair and transparent manner. The security features implemented in the smart contract—such as token locking, vesting schedules, anti-rugpull systems, and access control—ensure the integrity of the platform and protection for both investors and project teams.

As you expand the functionality of the platform, more complex bonding curve models (like logarithmic or more advanced exponential models) and additional security features (e.g., DAO governance, multi-signature wallets for liquidity management) can be incorporated for greater scalability and trust.

Our bonded curve presale launchpad is designed to provide a secure, transparent, and user-friendly platform for blockchain projects and investors. By incorporating innovative features like bonded curve pricing, token locking and vesting, anti-rugpull systems, and decentralized identity verification, we aim to set new standards in the way presales and token launches are conducted. Our goal is to build a trustworthy ecosystem that fosters the growth of the Web3 space while ensuring investor protection and long-term success for projects.

Community DAO Governors

The Hydrapad team is composed of experienced blockchain developers, Community Builders and entrepreneurs with a strong background in the web3 ecosystem.

- 1. **Angela Steffens** is an engineer with 20+ years of experience in Python, JavaScript, Rust, Solidity, and WASM, specializing in decentralized scalable applications. She has a history as an engineering lead and as a principal engineer in a successful startup exit. Most recently she's been a core team member at Unlock Protocol, is currently an Unlock DAO steward and does consulting in growth level startups helping with engineering team building and scaling systems architecture.

- 2. **Matthew Lopez** is a blockchain strategist skilled in ecosystem growth and community adoption. He specializes in creating sustainable models for scalable blockchain solutions. With extensive experience in education, hackathons, and financial inclusion, he bridges emerging technologies with underserved region

Hydrapad aims to foster a more open, dynamic, and fair environment for both project creators and investors in the Web3 ecosystem. It offers decentralized fundraising with a transparent mechanism, flexible participation, and a fair exit strategy for participants, all of which enhance the overall adoption and growth of decentralized finance (DeFi) projects and web3 space.

We are confident that with our Community driven approach, Hydrapad can drive adoption and provide long-term value and sustainable DeFi ecosystems.

Please feel free to reach out with any questions or for
additional documentation.

Warm regards,
Hydrapad Team
hello@hydrapad.com
www.hydrapad.com