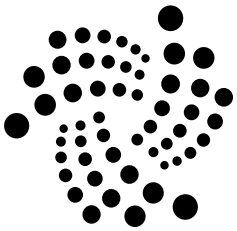
Hackathon REPORT

IOTAPAD



23 September 2024

Elvis Konjoh (99%)
Tariq Naeem(0.5%)
Emily Priyadarshini(0.5%)

Development of a Vesting for an overall Launchpad Module on IOTA

Introduction

The goal of this project was to build a decentralized application (dApp) on the IOTA platform that could facilitate the vesting of any token using different vesting strategies. Our team, with minimal prior knowledge of the IOTA language and platform, sought to implement a functional and minimal version of a token vesting system that could later be expanded for broader use cases. This report outlines the challenges we faced, the solutions we implemented, and the potential future directions for this project. Pull Request #4

Objectives

The primary objective of this project was to create a versatile vesting module for tokens using the IOTA blockchain, as well as to integrate it into a launchpad system that would allow any token to be launched with customizable vesting options. In particular, we aimed to:

1. Develop a flexible and modular vesting system.
2. Integrate the vesting system into a launchpad to allow for seamless token distribution.
3. Write and execute unit tests to validate the system.
4. Ensure that the module is adaptable for future extensions and broader use cases.

Despite the initial steep learning curve, we were able to achieve a minimal but fully functional version of the vesting and launchpad module.

Implementation

Vesting Module

The vesting module was designed to handle multiple vesting strategies, allowing tokens to be distributed to users over time according to a predefined schedule. Vesting is a crucial feature in many blockchain projects, as it ensures that tokens are gradually released to participants instead of being made fully available at once. This helps to prevent large-scale token dumps and maintains market stability.

The **Vesting Module** is responsible for managing the release of tokens over time based on predefined vesting strategies. Below is a brief explanation of how it works, followed by the state diagram.

Key States:

- **Initialization**: The vesting contract is created, defining the total amount of tokens and the vesting strategy (e.g., linear or time-frame based).
- **Active Vesting**: Tokens are being released to users based on the vesting schedule.
- **Completed Vesting**: All tokens have been fully vested and released to the users.

How It Works:

1. **Create Vesting**: The administrator creates a vesting contract with a chosen strategy, time frame, and amount of tokens.
2. **Vesting in Progress**: As time passes, the contract releases tokens to users based on the vesting schedule. The state will continue as "vesting in progress" until all tokens have been released.
3. **Releasing Tokens**: Periodically, users can claim their vested tokens.
4. **Vesting Complete**: Once all tokens have been fully released, the contract reaches the "Vesting Complete" state.

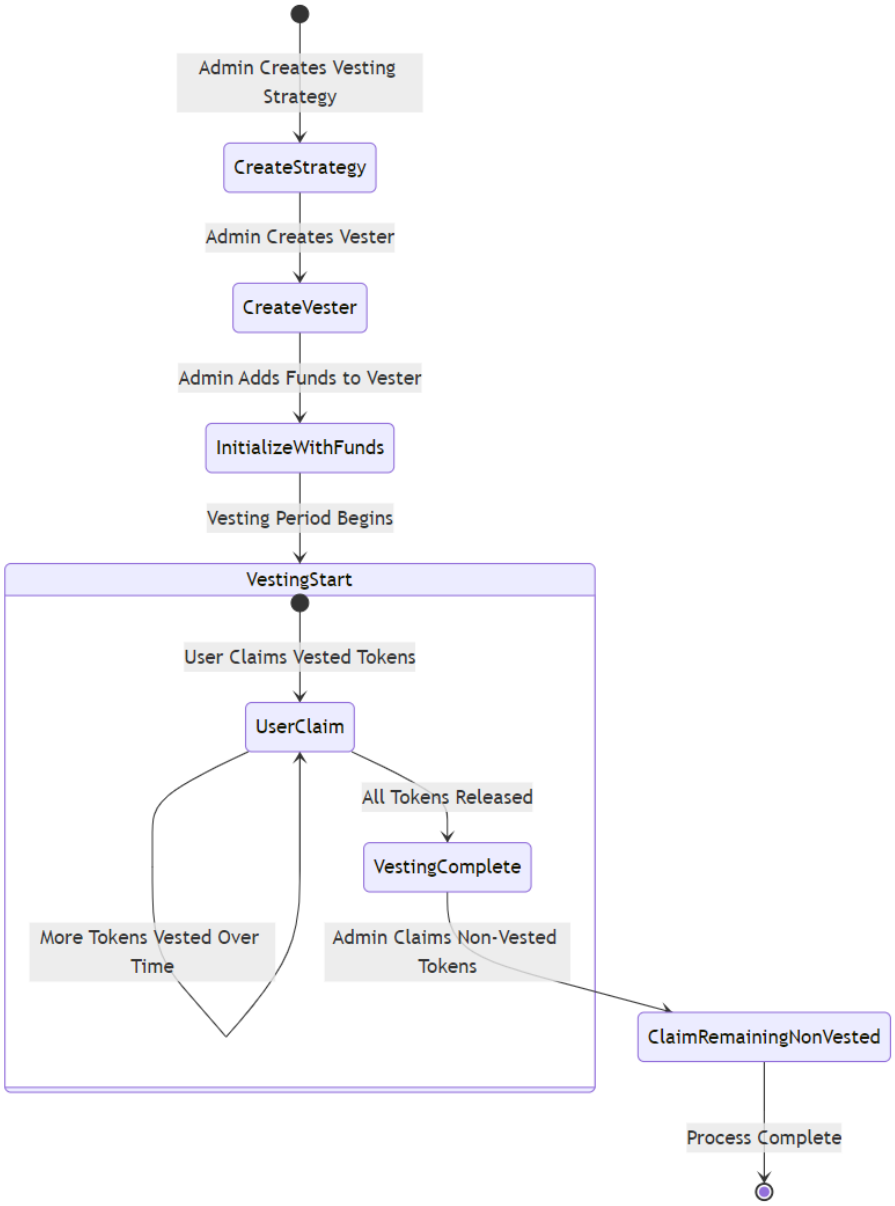


Figure 1: Workflow for the vesting module

Core Features of the Vesting Module

a. **Vesting Strategies**: We implemented several strategies, such as linear vesting and time-frame-based vesting. The linear vesting strategy allows tokens to be released uniformly over a set period. Time-frame-based vesting enables tokens to be released in phases, with specific percentages tied to predefined time frames.

b. **Customizable Parameters**: The module allows for flexibility in configuring the duration, percentage release schedule, and other vesting parameters, making it adaptable to various use cases.

c. **Minting Support**: We integrated a minting function, which allows tokens to be minted over time instead of simply being held in a contract. This feature is particularly useful for projects that need to manage supply inflation dynamically.
d. **Modular Design**: The vesting module was developed in a way that allows for future expansion. New vesting strategies or token-related features can be added without significant refactoring of the codebase.

Launchpad Module

The second part of the project involved integrating the vesting module into a launchpad system. This launchpad system allows projects to launch new tokens with customizable vesting schedules, simplifying the token distribution process for developers.

The **Launchpad Module** facilitates the token sale and manages participants. It integrates the **Vesting Module** to ensure that tokens are vested after the launchpad sale is completed.

Key States:

- **Launchpad Creation**: A new launchpad is created with parameters like token allocation, start and end times, and the conversion rate.
- **Launchpad Active**: The token sale is ongoing, and participants can register and stake tokens.
- **Launchpad Closed**: The sale is closed, and no further participation is allowed.
- **Vesting Creation**: A vesting contract is created for participants based on their contributions.
- **Token Claim**: The raised tokens are claimed according to the vesting schedule.

How It Works:

a. **Create Launchpad**: The administrator defines the launchpad's total allocation, vesting schedule, conversion rate, and duration.
b. **Register Participation**: Users can register for the token sale by staking tokens.
c. **Close Launchpad**: Once the sale ends, the launchpad is closed, and participants are no longer allowed to stake tokens.
d. **Create Vesting**: After the sale, the administrator creates a vesting contract based on participants' token stakes.
e. **Claim Tokens**: Participants can claim their tokens based on the vesting schedule defined earlier.

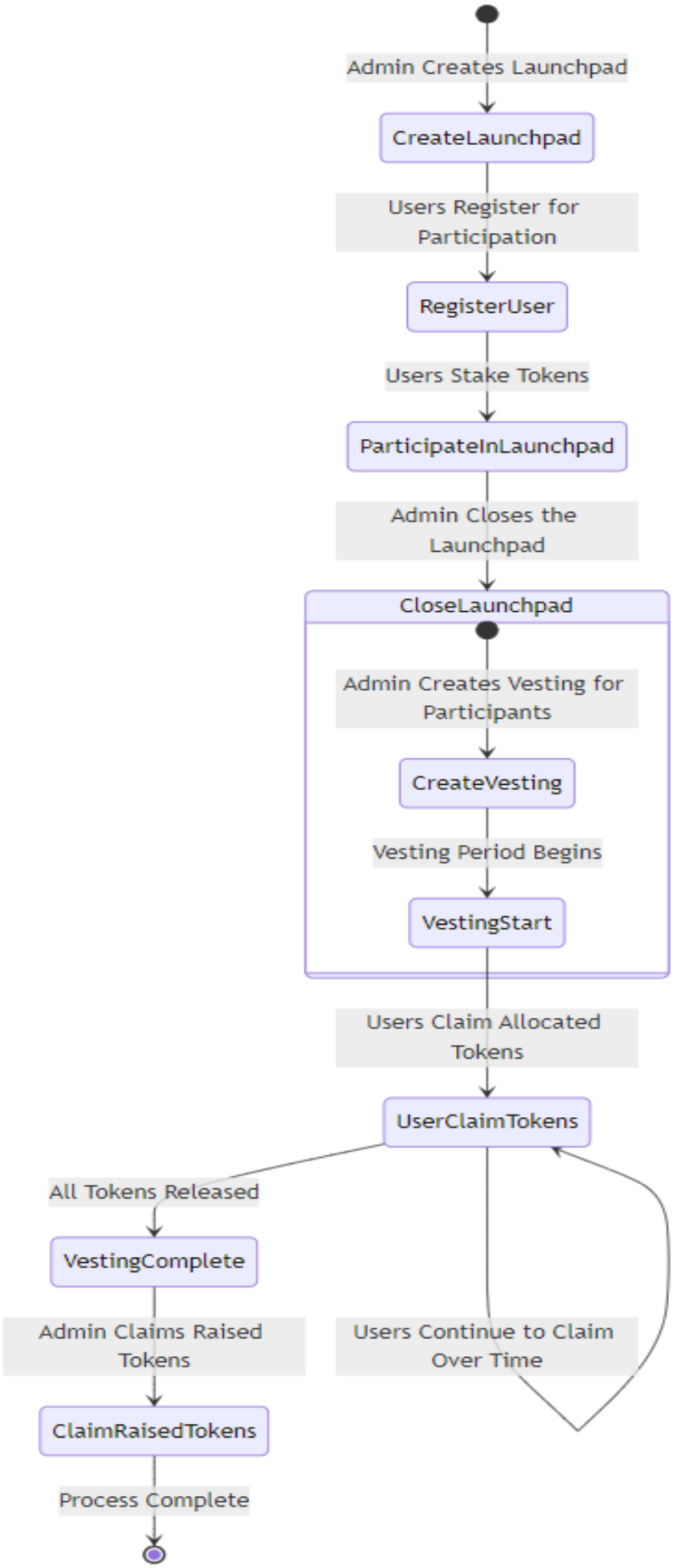


Figure 2 : Workflow of the launchpad

Core Features of the Launchpad Module

a. **Token Distribution**: The launchpad system ensures that participants in a token sale receive their tokens according to the vesting schedule defined during the setup process. This helps ensure fairness and prevent early investors from dumping large amounts of tokens.
b. **Dynamic Token Conversion**: The system allows users to contribute tokens at a predefined conversion rate. For example, if a user stakes 1 IOTA, they might receive 20 units of the new token, depending on the defined conversion rate. This conversion mechanism was designed with flexibility in mind, using integer ratios to handle fractional rates.
c. **Administrative Controls**: The launchpad includes features for administrators, allowing them to close the token sale, manage participants, and initiate vesting. This ensures that the system can be controlled and managed effectively.
d. **Integration with Vesting Module**: The launchpad uses the vesting module to ensure that token sales and distribution are tied to the vesting strategy selected during the token launch. This integration guarantees that all distributed tokens follow the vesting rules, providing trust and transparency to participants.

Unit Testing

To validate the functionality of both the vesting and launchpad modules, we implemented unit tests. These tests were designed to cover all core aspects of the system, including:

- The correct calculation of vesting amounts.
- The release of tokens according to the vesting schedule.
- The ability of the launchpad to handle multiple participants.
- The overall correctness of the conversion mechanism between the token being sold and the token being received.

Test Results

The unit tests confirmed that our modules work as expected:

1. **Vesting Release**: The vesting module released tokens correctly according to both linear and time-frame-based strategies.

2. **Launchpad Functionality**: The launchpad correctly allocated tokens to participants and distributed them according to the selected vesting schedule.
3. **Error Handling**: The system handled errors, such as attempts to participate after the sale ended or attempts to withdraw tokens before they were vested.

These results validated the core idea behind our project and provided a strong foundation for future development.

```
INCLUDING DEPENDENCY Iota
INCLUDING DEPENDENCY MoveStdlib
BUILDING iotapad
Running Move unit tests
[ PASS    ] 0x0::vesting::test_time_frame_based_releasable_amount
[ PASS    ] 0x0::vesting_test::test_add_supply
[ PASS    ] 0x0::iotapad_tests::test_create_launchpad_invalid_parameters
[ PASS    ] 0x0::vesting_test::test_collect_not_vested_coins
[ PASS    ] 0x0::iotapad_tests::test_create_launchpad_valid
[ PASS    ] 0x0::vesting_test::test_create_dynamic_strategy
[ PASS    ] 0x0::iotapad_tests::test_create_vesting_contract_fail_not_admin
[ PASS    ] 0x0::vesting_test::test_create_dynamic_vesting_with_timeframes
[ PASS    ] 0x0::iotapad_tests::test_create_vesting_contract_fail_still_active
[ PASS    ] 0x0::vesting_test::test_create_linear_strategy
[ PASS    ] 0x0::iotapad_tests::test_create_vesting_contract_success
[ PASS    ] 0x0::vesting_test::test_create_linear_vesting_with_duration
[ PASS    ] 0x0::iotapad_tests::test_launchpad_lifecycle
[ PASS    ] 0x0::vesting_test::test_duration_based_releasable_amount
[ PASS    ] 0x0::iotapad_tests::test_register_duplicate_participation
[ PASS    ] 0x0::vesting_test::test_invalid_timeframes_and_percentages
[ PASS    ] 0x0::iotapad_tests::test_register_launchpad_success
[ PASS    ] 0x0::vesting_test::test_mint_and_vest_fail_without_treasury
[ PASS    ] 0x0::vesting_test::test_mint_and_vest_success
[ PASS    ] 0x0::vesting_test::test_release_coins_by_coinbase
[ PASS    ] 0x0::vesting_test::test_release_coins_linear_vesting
[ PASS    ] 0x0::vesting_test::test_release_coins_timeframes
Test result: OK. Total tests: 22; passed: 22; failed: 0
```

Deliverable: Launchpad and Vesting Package

The deliverable for this project consists of a comprehensive package that includes two core modules—**Launchpad** and **Vesting**—as well as unit tests for each module. The purpose of this package is to provide a reusable and extendable system that can be integrated into any project for token vesting and launchpad functionalities.

1. Vesting Module

The **Vesting Module** provides a robust solution for token vesting with multiple strategies. It can handle various types of vesting schedules, including linear and dynamic, and can mint tokens as part of the vesting process. The core functionalities include:

- **Create Strategy**: Different strategies for token vesting can be defined, including time-based releases and dynamic allocation.
- **Create Vester**: This function allows the creation of a vesting contract with specific parameters, such as start time, duration, and vesting strategy.
- **Initialize with Funds**: After creating the vesting contract, it can be initialized with tokens to be distributed according to the vesting schedule.
- **Vesting Start**: Once initialized, the vesting period begins, and users can start claiming their vested tokens periodically.
- **User Claim**: Users can claim their vested tokens as time progresses, according to the vesting strategy defined in the contract.
- **Vesting Complete**: After the entire vesting period has passed and all tokens have been claimed, the vesting process is considered complete.
- **Claim Non-Vested Tokens**: The admin can claim any remaining, non-vested tokens after the vesting period has concluded.

This module is designed to be easily integrated into any blockchain project, providing a standardized way of handling token vesting for various scenarios.

2. Launchpad Module

The **Launchpad Module** enables the creation of a decentralized token launchpad where users can stake tokens in exchange for participation in a token sale or allocation. The main components of the module include:

- **Create Launchpad**: Admins can create a new launchpad, defining the parameters such as conversion rate, start and end times, total allocation, and the recipient of the raised funds.
- **User Registration**: Users can register for the launchpad to indicate their intention to participate. The registration process ensures that only approved participants can stake tokens.
- **User Participation**: Once registered, users can stake their tokens to participate in the launchpad. Their share of the token allocation is calculated based on the amount staked and the total allocation.
- **Close Launchpad**: The admin can close the launchpad once the sale or allocation period has ended, preventing further participation.

- **Create Vesting for Participants**: After closing the launchpad, the admin can create a vesting schedule for the participants, allowing them to claim their allocated tokens over time.
- **User Token Claims**: Participants can claim their allocated tokens as per the vesting schedule.
- **Claim Raised Tokens**: Admins can claim the tokens raised in the launchpad according to the vesting schedule, ensuring a fair release of funds.

The **Launchpad Module** ensures a smooth and secure process for token distribution through a decentralized mechanism, allowing projects to launch tokens while managing their vesting schedule.

3. Unit Tests

To ensure the reliability and correctness of both modules, extensive **unit tests** have been written for each functionality. The tests verify:

- Correct creation of vesting strategies and vesting contracts.
- Proper initialization and distribution of tokens through the vesting process.
- Accurate registration and participation in the launchpad.
- Correct handling of token allocations and vesting schedules.
- Robust error handling for invalid inputs or unauthorized actions.

These tests provide a safety net and ensure that both modules can be integrated into a larger system with confidence.

Integration and Use

Both the **Vesting** and **Launchpad** modules are designed to be highly modular and reusable. They can be integrated as standalone components or in combination within any blockchain-based project that requires token vesting and a launchpad for decentralized token distribution. The package provides flexibility for adding new vesting strategies or customizing the launchpad parameters, making it suitable for a wide range of applications, from ICOs to community-driven token launches.

Challenges

While the project was successful, we encountered several challenges along the way. These challenges highlighted the learning curve associated with IOTA's platform and the complexities of blockchain development in general.

1. Lack of Clear Documentation

One of the most significant challenges was the lack of clear and up-to-date documentation for IOTA's SDK and language. As newcomers to the platform, we struggled to find reliable resources on how to set up the development environment, run tests, and deploy the contract on IOTA's testnet. The available documentation often assumed a certain level of familiarity with the platform, which made it difficult for us to get started.

Overcoming This Challenge

We relied heavily on trial and error, as well as community resources such as online forums and chats. Notably, we were surprised to receive timely assistance from Mirko Zichichi, even late at night. His support was invaluable in helping us resolve several roadblocks.

2. Limited Knowledge of IOTA and Smart Contracts

Our team had minimal prior experience with IOTA and its smart contract language. The syntax, structure, and paradigms of IOTA’s development environment were unfamiliar, leading to a slow initial development process.

Overcoming This Challenge

We adopted an incremental approach, starting with basic functionality and gradually adding complexity. By focusing on small, manageable tasks, we were able to learn the system step by step, eventually implementing the full vesting and launchpad modules.

3. Testing and Debugging

Another challenge we faced was the lack of proper tooling for debugging smart contracts on IOTA. While we implemented unit tests to ensure correctness, debugging the actual contract execution on the network was more difficult than expected.

Overcoming This Challenge

We wrote extensive test cases and incorporated as many edge cases as possible. This helped us identify potential issues before deploying the contract, reducing the need for extensive post-deployment debugging.

4. Time Constraints

Given the time limitations of the hackathon, we had to prioritize certain features over others. While we achieved our primary objectives, we were unable to implement several planned extensions.

Future Directions

The current implementation represents a minimal version of the vesting and launchpad modules. There are several ways in which the project could be expanded:

1. Additional Vesting Strategies

While we implemented linear and time-frame-based vesting strategies, there are many other types of vesting that could be added to the module. For example:

- **Cliff Vesting**: Tokens are held for a specific period before being gradually released.
- **Percentage-based Vesting**: Tokens are released in percentages at specific milestones.

2. Multi-token Support

Currently, the system supports vesting for one token at a time. Adding multi-token support would allow projects to vest multiple tokens simultaneously, increasing the flexibility and usability of the module.

3. Improved User Interface

An enhanced user interface could make the system more accessible to non-technical users. For example, integrating a web-based front-end that allows users to interact with the vesting and launchpad systems would simplify the process for end-users.

4. Improved Documentation and Tutorials

One of the major barriers we faced was the lack of clear documentation. By contributing to the documentation and creating tutorials for future developers, we can help lower the barrier to entry for others looking to build on IOTA.

Conclusion

This project was an ambitious undertaking, especially given our minimal prior knowledge of IOTA and its platform. However, through perseverance and collaboration, we were able to successfully implement a functional vesting and launchpad module.

The result is a flexible system that can be used to vest any token using a variety of strategies. We validated our approach through comprehensive unit testing, and the results confirmed that the system works as expected.

While we faced several challenges, including limited documentation and a steep learning curve, we were able to overcome these obstacles and deliver a working solution. The

project has significant potential for future expansion, and we hope that it can serve as a foundation for more complex decentralized applications in the IOTA ecosystem.

Lastly, we would like to express our gratitude to the hackathon committee and especially to Mirko Zichichi for his support during the development process. His assistance was critical in helping us meet our project goals.