Token Economy Project Guide

Build Your Own Token Ecosystem

BSc Blockchain, Crypto Economy & NFTs

FS2026

Learning Objectives

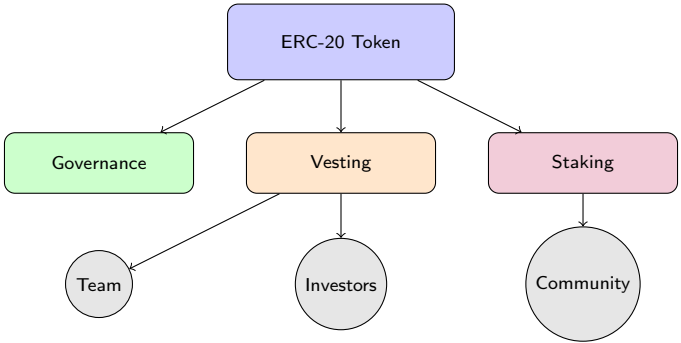
By completing this project, you will:

1. Master the ERC-20 token standard
2. Implement governance mechanisms
3. Design vesting schedules for token distribution
4. Create staking rewards systems

Related Lessons: L17, L19, L29, L30, L31

Problem: How do we create programmable money that aligns stakeholder incentives?

Project Architecture



Four components: Base token, governance voting, vesting schedules, staking rewards

The ERC-20 Standard

Core Functions:

- `totalSupply()` - Total tokens
- `balanceOf(addr)` - Account balance
- `transfer(to, amt)` - Send tokens
- `approve(spender, amt)` - Allow spending
- `allowance(owner, spender)` - Check allowance
- `transferFrom(from, to, amt)` - Delegated transfer

Events:

- `Transfer(from, to, amount)` - Emitted on any transfer
- `Approval(owner, spender, amount)` - Emitted on approval

ERC-20 is the foundation for all fungible tokens on Ethereum

Basic Token Implementation

```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract MyToken is ERC20, Ownable {
    constructor(
        string memory name,
        string memory symbol,
        uint256 initialSupply
    ) ERC20(name, symbol) Ownable(msg.sender) {
        _mint(msg.sender, initialSupply * 10**decimals());
    }

    function mint(address to, uint256 amount) public onlyOwner {
        _mint(to, amount);
    }
}
```

Using OpenZeppelin reduces bugs and saves development time

Governance Design

Why Token Governance?

- Align decision-making with token ownership
- Decentralize protocol control over time
- Enable community participation

Key Concepts:

Voting Power

- 1 token = 1 vote
- Delegation allowed
- Snapshot at proposal creation

Proposal Lifecycle

- Propose → Vote → Execute
- Quorum requirements
- Time delays for safety

Governance tokens transform passive holders into active participants

Vesting Schedules

Purpose: Align long-term incentives by gradually releasing tokens

Category	Cliff	Vesting	Total
Team	12 months	36 months	48 months
Advisors	6 months	18 months	24 months
Investors	6 months	24 months	30 months
Community	0 months	12 months	12 months

Key Terms:

- **Cliff**: Period before any tokens vest
- **Linear vesting**: Equal amounts released each period
- **Beneficiary**: Address receiving vested tokens

Vesting prevents token dumps and signals long-term commitment

Staking Mechanics

How Staking Works:

1. User deposits (stakes) tokens into contract
2. Contract tracks pro-rata share of rewards
3. User can claim rewards or withdraw anytime

Reward Calculation:

$$\text{User Reward} = \text{User Stake} \times \frac{\text{Total Rewards}}{\text{Total Staked}}$$

Design Parameters:

- Reward rate (tokens/second)
- Lock period (optional)
- Early withdrawal penalty (optional)

Staking incentivizes long-term holding and reduces circulating supply

Token Economics

Distribution Example:

- Team: 15% (4-year vest)
- Investors: 20% (2-year vest)
- Treasury: 20% (DAO controlled)
- Community: 25% (rewards)
- Liquidity: 20% (DEX pools)

Critical Questions:

- What utility does the token provide?
- How do you prevent sell pressure?
- What drives long-term demand?

Emission Schedule:

- Year 1: 40% of total
- Year 2: 25% of total
- Year 3: 20% of total
- Year 4+: 15% of total

Good tokenomics balances supply/demand and creates sustainable incentives

Apply the 6 Questions:

Question	Token Economy Answer
PROBLEM	Coordinate stakeholders without central authority
INCENTIVES	Token rewards align user and protocol interests
BENEFITS/COSTS	Users get yield; protocol gets locked liquidity
FAILURE MODE	Inflation spirals, governance attacks, exit scams
DESIGN CHOICES	Fixed vs dynamic supply, voting mechanisms
ALTERNATIVES	Traditional equity, profit-sharing models

Every design decision has trade-offs - make them explicit

Implementation Checklist

Phase 1: Basic Token

- ☐ Deploy ERC-20 with initial supply
- ☐ Test transfer and approve functions
- ☐ Add mint/burn capabilities

Phase 2: Vesting

- ☐ Create vesting contract
- ☐ Configure schedules for each category
- ☐ Test release mechanics

Phase 3: Staking

- ☐ Deploy staking pool
- ☐ Fund reward pool
- ☐ Verify reward calculations

Phase 4: Governance

- ☐ Add voting extension
- ☐ Create governor contract
- ☐ Test proposal lifecycle

Resources

Documentation:

- OpenZeppelin Contracts: `docs.openzeppelin.com`
- EIP-20 Standard: `eips.ethereum.org/EIPS/eip-20`
- Compound Governance: `compound.finance/docs`

Project Materials:

- Jupyter Notebook: `projects/notebooks/01_token_economy.ipynb`
- Web Guide: `digital-ai-finance.github.io/Cryptoeconomics-Blockchain/projects/token-economy/`

Related Course Content:

- L17: Token Standards
- L19: Token Design
- L29-31: Tokenomics modules