Version 18/12/2025

Project77 Whitepaper

Table of Contents

Disclaimer

Whitepaper is subject to change without prior notice. This document is for informational purposes only and does not constitute investment, financial, legal or tax advice. Using the token is at your own risk, and cryptocurrency investments carry the risk of partial or total loss. Details about used protocols and investments made by caretakers will be announced when funds used in this manner will exceed 100.000$. Caretakers are not liable for third-party risks such as protocol hacks, exchange bankruptcy or investments being non profitable because they are the owner of it, in exchange buyers obtained tokens Project77. 100% TGE refers to time between TGE and Listing, meaning if there is allocation with 100% TGE their beneficiaries will be able to receive/claim tokens before listing. Mission is subject to change if new Caretakers chosen in voting in 2030 decided to change project mission from initial one.

1 About

Project77 (later “$P77”) is a public, non-profit, open and fully decentralized cryptocurrency token deployed on the Base Mainnet (Chain ID 8453). $P77 smart contract has no owner, is non-upgradeable and has no functions that could benefit any individual such as buy/sell tax, mint, blacklist, freeze or transfer without approval. All 77 million tokens were minted at Token Generation Event (later “TGE”). In early stage token relies on individuals committed to serve its mission and help fulfill Roadmap’s milestones.
There are two types of roles: one Main Caretaker and four Caretakers. Main caretaker is a guard of token reserve and its mission, Caretakers serve purpose of providing structure allowing $P77 to grow such as website, smart contracts. They manage social channels and are responsible of making investments to obtain funds for token burning.
Initial Main Caretaker is token deployer, initial Caretakers are core contributors.
In Q1 of 2030 there will be voting in which token Burners will decide who is the next Main Caretaker and who are the next Caretakers. Rules and details are provided in Voting Procedure section.

2 Mission

$P77 is a minimalist token that is heading to ultimate zero supply.
This can be achieved by several factors:

- Minimization of operational cost being less than 100$ per year for whole $P77 ecosystem, including but not limited to smart contracts, website, promoting campaigns. These costs are fully paid by private Caretakers funds.
- All funds/fees obtained by Caretakers via token Presale or providing liquidity will be used in lending protocols/investments providing in a long-term funds for constant buybacks and burning.
- Unsold tokens from Presale allocation and unclaimed tokens from Airdrop allocation will be burned.
- $P77 smart contract has no mint function, minting new tokens is not possible.
- $P77 Tokenomics is designed that all unused tokens will be burned.
- $P77 has no allocation for team or CEX exchanges, significantly reducing supply.

Combining all these factors ensure that $P77’s mission is realistic and structurally enforced.

3 Burning Mechanism

$P77 has implemented a public Burn (0x42966c68) function in its smart contract which allows to remove permanently tokens from effective supply. How does it work? Any $P77 holder can call this function via BaseScan or in the future on our website. This function transfers tokens from caller’s wallet to 0x0000000000000000000000000000000000000000 (later “0x0”) EVM address. 0x0 address is a wallet without owner, ensuring burned tokens will stay there forever, Burn function can not be reversed.

4 Voting Procedure

$P77 community members can obtain Discord roles by burning tokens, which allow them to vote and submit proposals. Types of roles:

- VOTE X1, role can be obtained by burning at least 1.000 $P77. It grants the right to vote with x1 multiplier in votings.
- VOTE X10, role can be obtained by burning at least 10.000 $P77. It grants the right to vote with x10 multiplier in votings.
- VOTE X100, role can be obtained by burning at least 100.000 $P77. It grants the right to vote with x100 multiplier in votings and to submit one proposal per month.
- Adept, role can be obtained by burning at least 77.000 $P77. It grants the right to be a candidate in voting which will decide next Caretakers.
- Main Adept, role can be obtained by burning at least 777.000 $P77. It grants the right to be candidate in voting which will decide next Main Caretaker.

After a member meets the requirement, it is required to open support ticket on our Discord server. In order to verify wallet ownership, Main Caretaker may request small transfer up to 100 $P77 which will be refunded, after this member will obtain role. Please note that address from this transfer will be linked to member's discord ID in database, role transfer is not possible, each wallet can be used only once to obtain each role. Updating roles is allowed for example member with VOTE X1 role by burning additional 9.000 $P77 can obtain VOTE X10 role. Only one VOTE role can be active on member's account.

Standard Voting Procedure

Main Caretaker or members holding VOTE X100 role may submit proposals that can be either accepted or declined by community members with voting roles: VOTE X1, VOTE X10, VOTE X100. Each voting will have voting period of 24 hours up to 3 days. Voting will take place via surveys in dedicated Discord channels, after voting period ends all votes will be counted and multiplied by voting roles multipliers. Vote can be cast in favor or against proposal. Voting is binding if at least 51% of total weighted votes are in favor.

Main Caretaker and Caretakers Voting Procedure

In 2030 there will be two votings but it is only possible to cast votes in favor. This time community will decide who is next Main Caretaker and four Caretakers. To be candidate, member needs Adept or Main Adept role before 31.12.2029. Four most popular candidates in Caretakers voting will be the next four Caretakers. The most popular candidate in Main Caretaker voting will be the next Main Caretaker. Each Caretaker will receive 25% of investment funds. Main Caretaker will receive full access to deployer wallet with $P77 reserve (27.000.000 tokens).

Voting Procedure will be reworked to be more decentralized in Q2 2026

5 Tokenomics

Tokenomics is one of the most important factors while looking before making investment in cryptocurrency, we made sure that our Tokenomics represents something what we would invest in. Both 100% unlocks for airdrop and Presale showing confidence in project, no irrational allocations, no giving millions of tokens for CEX exchanges so they can dump them the next second they received them and no referrals in airdrop and Presale making it fair for everyone.
Simplified version of Tokenomics in form of graph can be found at home page.

Allocations (Total 77 millions tokens):

- Reserve 27m (approx. 35.06%) cliff 3 years***.

Usage of reserve will be determined by main caretaker chosen in voting in 2030, until then it remains locked in smart contract.

- Liquidity 9m (approx. 11.69%) TGE 100%

Liquidity allocation allows caretakers to provide liquidity on DEX.

- Presale 9m (approx. 11.69%) TGE 100%

Presale allows caretakers to obtain capital for investments that will allow burnings, unsold tokens will be burned.

- Airdrop 9m (approx. 11.69%) TGE 100%

Airdrop serves purpose as a big thank you for early supporters, more than 10% of total supply and 100% TGE shows that we respect those who support us.

- Launchpads/Private 9m (approx. 11.69%) TBD

This is allocation to acquire additional funds for investments. More details will be shared closer to listing date.

- Future incentives 5m (approx. 6.49%) cliff 1 year

This allocation will be used but not limited to future airdrops, staking rewards, burning, community campaigns.

- Marketing 5m (approx. 6.49%) 100% TGE

This allocation will be solely used for marketing purposes such as paid campaigns, airdrops, burning.

- Core Contributors 4m (approx. 5.19%) 100% TGE

This allocation is for early contributors that allowed this project to happen in the first place, smallest allocation yet huge amount of great job done by them.

6 Airdrop

$P77 Airdrop campaign launched on 14/12/2025. Airdrop campaign will end in Q4 2026.
Airdrop’s tasks will take place on 2 platforms. First platform is Zealy where our campaign is already live, second platform is $P77 website. Tasks on website will be launched in Q2 2026. Participating in airdrop is free, participating in airdrop campaigns does not guarantee receiving airdrop, Caretakers reserve the right to filter out sibils, spam accounts and will put requirements that will be announced in Q1 2026.

7 Presale

$P77 Presale will launch in Q1 2026 and end in Q4 2026. Purchased tokens in presale are not refundable, in exchange of funds buyers obtained $P77 meaning the caretakers have full rights over the funds. Tokens bought in presale will be airdropped to buyers wallet, there is no option to change designated wallet. Tokens are being sold at 269.500$ FDV making one $P77 worth approx. 0,0035$. Tokens can be bought in exchange of USDC on Base mainnet. Tokens bought in presale will be rounded down. For example if someone bought 1100.80 $P77, 1100 tokens will be airdropped.

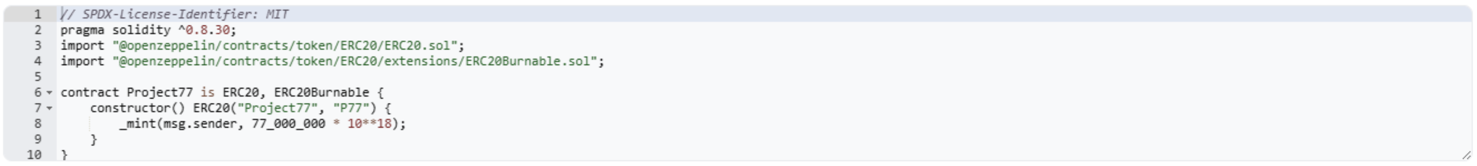
8 Roadmap

$P77 Roadmap is build for long-term development. Simplified version of Roadmap in graphic form be found at home page.
Q3 2025+
Start of building website, making social accounts, discord server nad testing smart contrracts of token, presale, claim, airdrop and lock. $P77 litepaper is public.
Q2 2026
Website fully launched, airdrop and presale are fully live, token deployed on blockchain. Voting Procedures are now decentralized.
Q1 2027
Airdrop claiming, burning starts, token listed on DEX, Presale tokens airdropped, start of 3 year $P77 Reserve cliff. Voting mechanism is live
Q1 2030+
Reducing total supply to less than 25m tokens (excluding reserve), Voting for next Main Caretaker and Caretakers, future is in their hands.

9 Smart contracts

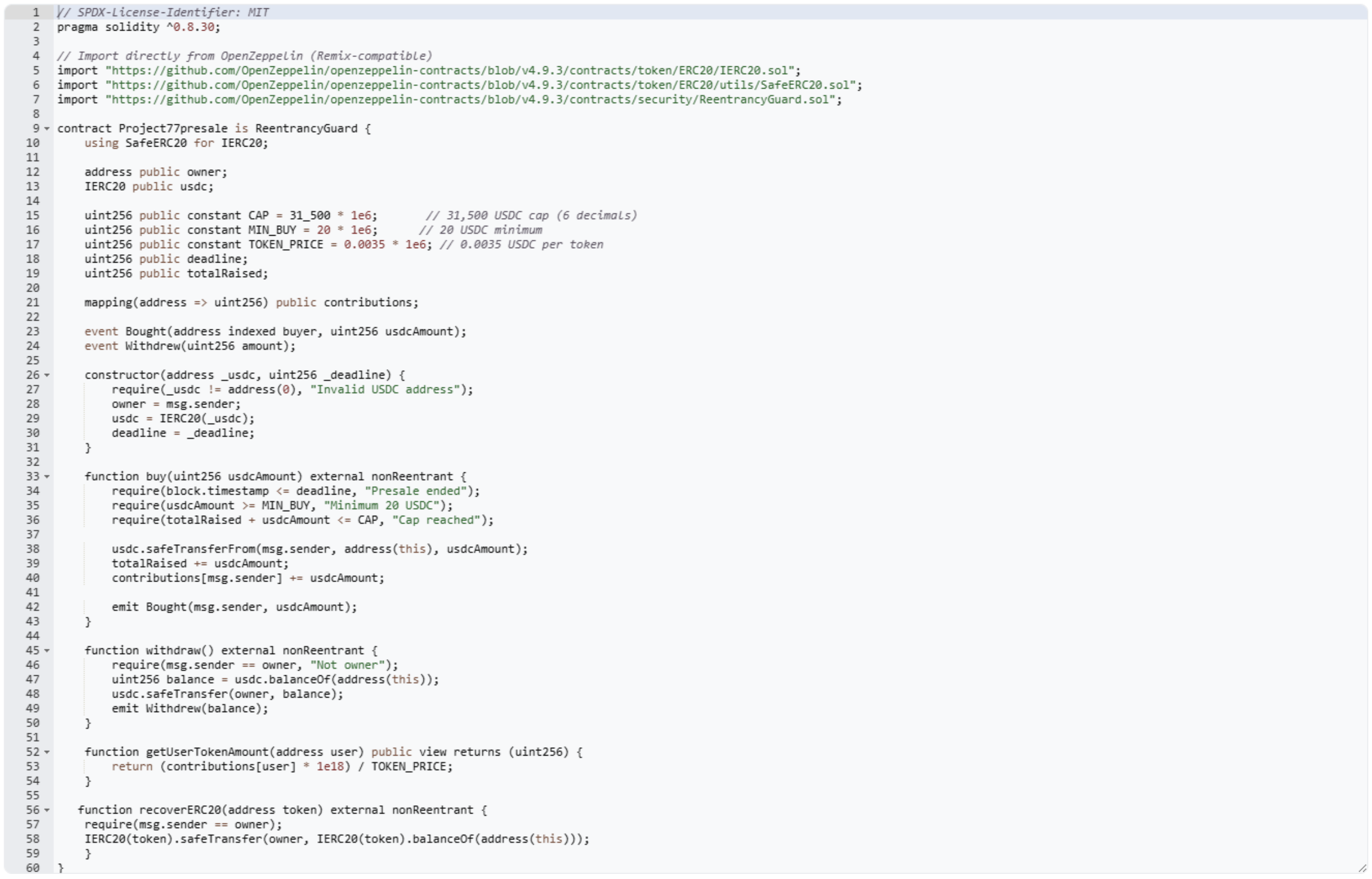
All $P77 smart contracts deployed by Main Caretaker are public and verified on blockchain.

$P77 smart contract:



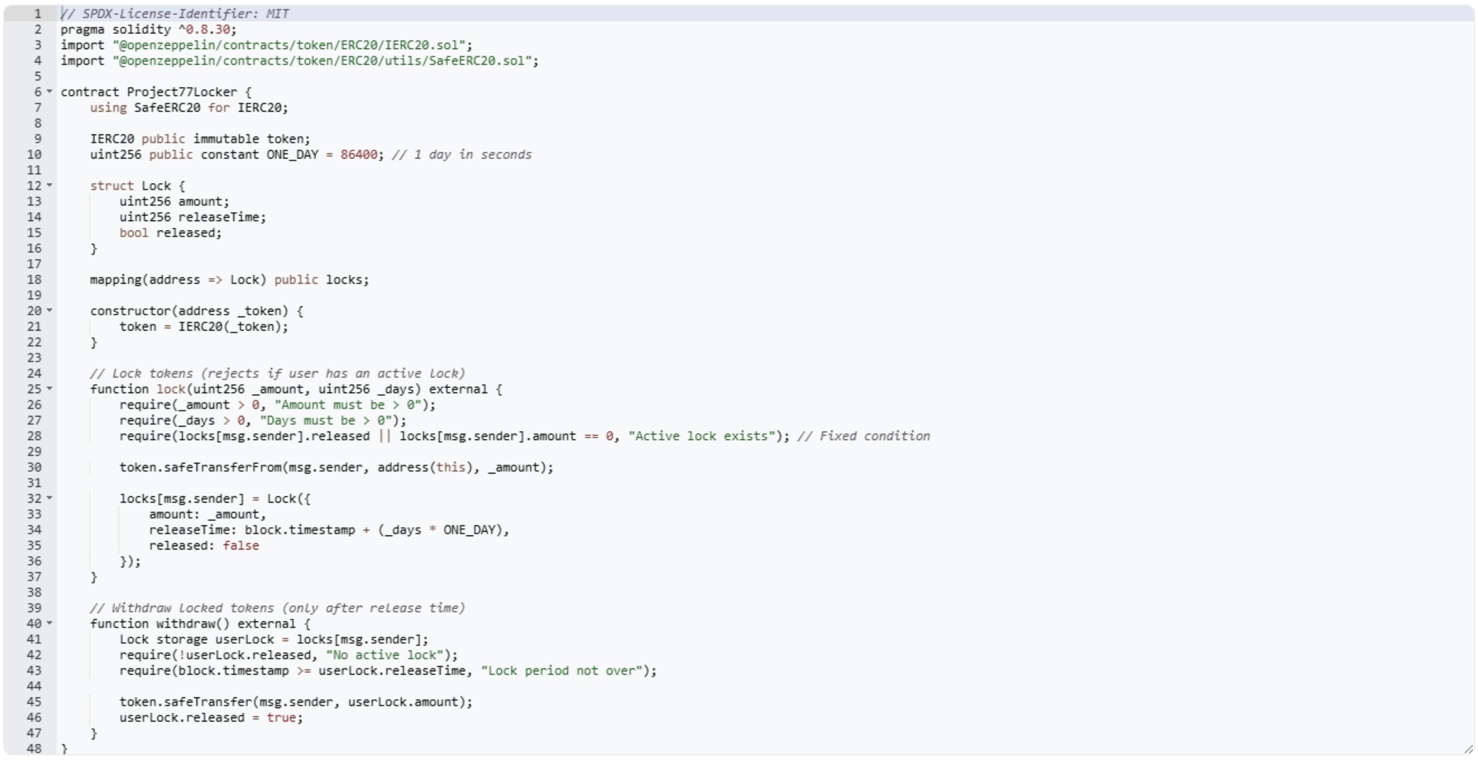
```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.30;
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol";

contract Project77 is ERC20, ERC20Burnable {
    constructor() ERC20("Project77", "P77") {
        _mint(msg.sender, 77_000_000 * 10**18);
    }
}
```

$P77 Presale smart contract:



```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.30;

// Import directly from OpenZeppelin (Remix-compatible)
import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.9.3/contracts/token/ERC20/IERC20.sol";
import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.9.3/contracts/token/ERC20/utils/SafeERC20.sol";
import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v4.9.3/contracts/security/ReentrancyGuard.sol";

contract Project77presale is ReentrancyGuard {
    using SafeERC20 for IERC20;

    address public owner;
    IERC20 public usdc;

    uint256 public constant CAP = 31_500 * 1e6;       // 31,500 USDC cap (6 decimals)
    uint256 public constant MIN_BUY = 20 * 1e6;      // 20 USDC minimum
    uint256 public constant TOKEN_PRICE = 0.0035 * 1e6; // 0.0035 USDC per token
    uint256 public deadline;
    uint256 public totalRaised;

    mapping(address => uint256) public contributions;

    event Bought(address indexed buyer, uint256 usdcAmount);
    event Withdrew(uint256 amount);

    constructor(address _usdc, uint256 _deadline) {
        require(_usdc != address(0), "Invalid USDC address");
        owner = msg.sender;
        usdc = IERC20(_usdc);
        deadline = _deadline;
    }

    function buy(uint256 usdcAmount) external nonReentrant {
        require(block.timestamp <= deadline, "Presale ended");
        require(usdcAmount >= MIN_BUY, "Minimum 20 USDC");
        require(totalRaised + usdcAmount <= CAP, "Cap reached");

        usdc.safeTransferFrom(msg.sender, address(this), usdcAmount);
        totalRaised += usdcAmount;
        contributions[msg.sender] += usdcAmount;

        emit Bought(msg.sender, usdcAmount);
    }

    function withdraw() external nonReentrant {
        require(msg.sender == owner, "Not owner");
        uint256 balance = usdc.balanceOf(address(this));
        usdc.safeTransfer(owner, balance);
        emit Withdrew(balance);
    }

    function getUserTokenAmount(address user) public view returns (uint256) {
        return (contributions[user] * 1e18) / TOKEN_PRICE;
    }

   function recoverERC20(address token) external nonReentrant {
    require(msg.sender == owner);
    IERC20(token).safeTransfer(owner, IERC20(token).balanceOf(address(this)));
    }
}
```

$P77 Token Locker smart contract:



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.30;
import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
import "@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol";

contract Project77Locker {
    using SafeERC20 for IERC20;

    IERC20 public immutable token;
    uint256 public constant ONE_DAY = 86400; // 1 day in seconds

    struct Lock {
        uint256 amount;
        uint256 releaseTime;
        bool released;
    }

    mapping(address => Lock) public locks;

    constructor(address _token) {
        token = IERC20(_token);
    }

    // Lock tokens (rejects if user has an active lock)
    function lock(uint256 _amount, uint256 _days) external {
        require(_amount > 0, "Amount must be > 0");
        require(_days > 0, "Days must be > 0");
        require(locks[msg.sender].released || locks[msg.sender].amount == 0, "Active lock exists"); // Fixed condition

        token.safeTransferFrom(msg.sender, address(this), _amount);

        locks[msg.sender] = Lock({
            amount: _amount,
            releaseTime: block.timestamp + (_days * ONE_DAY),
            released: false
        });
    }

    // Withdraw locked tokens (only after release time)
    function withdraw() external {
        Lock storage userLock = locks[msg.sender];
        require(!userLock.released, "No active lock");
        require(block.timestamp >= userLock.releaseTime, "Lock period not over");

        token.safeTransfer(msg.sender, userLock.amount);
        userLock.released = true;
    }
}
```

10 Addresses

Token contract: 0x60Cd0c438EbEe7736af69E0d2C902e9892548E82
Presale contract: 0x80f104c0275B9726b78DB4D329104Cfb2947B1d5
Claim contract: TBA
Lock contract: TBA

Reserve wallet: 0xaEd538A9c1f15380F569810B879ff65a78684D12
Wallet providing liquidity: TBA
Wallet with Presale and Lauchpad tokens: TBA
Future incentives wallet: TBA
Marketing wallet: TBA