

Garden Fleet (GF) Yellow Paper

The Garden Fleet (GF) Token is a Celo-based ReFi (Regenerative Finance) token designed to fund and govern a decentralized fleet of low-cost ships that remove plastic waste from coastlines and reforest eroded beaches with mangroves. This Yellow Paper provides technical details, including tokenomics, smart contract logic, staking mechanisms, governance structures, and liquidity management.

Garden Fleet: Regenerative Solution for Coastal Restoration and Waste Management

Abu-Bakr Harakat

03/01/2025



i In addition to this Yellow Paper, a White Paper is available that explains the Garden Fleet Project in detail, along with an ESG Services Paper that outlines our service offerings for corporations looking to improve their ESG ratings.

Technical Overview

Smart Contract Implementation

The GF Token is an **ERC-20** compliant token on the **Celo blockchain**, enabling staking, governance, and impact-based rewards. The smart contract ensures secure and transparent token transactions, leveraging built-in functions for transferring tokens, managing staking balances, and governing DAO proposals.

The contract includes key functions for such as **automated staking rewards**, **DAO voting**, and **impact-based NFT issuance**. It is designed to prevent double-spending and unauthorized token minting, ensuring security through **gas-efficient** transactions and **multi-signature** governance.

Token Rewards

NFT: Ocean Guardian

Users who stake GF tokens to fund **plastic waste cleanup operations** receive Ocean Guardian NFTs. These NFTs serve as verifiable proof of waste removal efforts and ecosystem restoration. Benefits include:

- Blockchain-verified certification of plastic waste removal efforts.
- Enhanced ESG compliance credentials for corporate stakeholders.
- Potential trade or resale value as proof-of-impact assets.

NFT: Mangrove Guardian

This reward is issued to users who stake GF tokens to support **mangrove reforestation efforts**. Each NFT represents a verified contribution to planting and maintaining coastal mangroves. The benefits include:

- Blockchain-verified certification of beach mangrove forestation.
- Enhanced ESG compliance credentials for corporate stakeholders.
- Potential trade or resale value as proof-of-impact assets.

Token Rewards

Stakers receive **GF token rewards** based on their contribution to ecosystem restoration. The staking mechanism provides the following incentives:

- Drop rewards issued periodically based on participation in staking pools.
- The ability to convert staking rewards into NFTs for added verification and ESG impact tracking.

Smart Contract Logic

1. Token Allocation

Defines how the total supply of tokens is allocated to different functions within the ecosystem, ensuring stability and fairness.

```
# Token Allocations
self.dao_treasury = 100_000_000 # 10% reserved for community governance funding
self.liquidity_reserve = 30_000_000 # 3% set aside for exchange liquidity
self.fleet_operations_reserve = 350_000_000 # 35% for coastal restoration & ESG services
self.staking_rewards_pool = 150_000_000 # 15% allocated to staking incentives
self.ico_supply = 350_000_000 # 35% made available for public sale (ICO)
self.team_allocation = 20_000_000 # 2% set aside for project team and advisors
```

These allocations ensure that sufficient funds are directed toward ecosystem sustainability, incentivizing participation while maintaining operational efficiency.

2. Transfer Mechanism

This function allows users to transfer GF tokens securely between wallets.

```
def transfer(self, sender, receiver, amount):
    """ Transfer GF Tokens between users """
    if self.balances.get(sender, 0) >= amount:
        self.balances[sender] -= amount
        self.balances[receiver] = self.balances.get(receiver, 0) + amount
        return True
    return False
```

The function ensures that the sender has enough tokens before making a transfer. If not, the transaction is rejected.

3. Staking Mechanism

This function enables users to stake their GF tokens to support ecosystem activities like coastal cleanup and reforestation.

```
def stake(self, user, amount, usdt_value, pool_type):
    """ Stake GF Tokens for Cleanup & Reforestation (Funds Go to Fleet Wallet) """
    if self.balances.get(user, 0) >= amount:
        self.balances[user] -= amount
        self.staked_balances[user] = self.staked_balances.get(user, 0) + amount
        self.staking_start_time[user] = time.time()
        self.investment_in_usdt[user] = usdt_value

        # 100% of staking investment is sent to the Fleet Wallet
        self.fleet_wallet += usdt_value

        self.issue_nft_reward(user, usdt_value, pool_type)
        return True
    return False
```

The function ensures that users have enough balance before staking. Once staked, the investment is locked and used for fleet operations.

4. NFT Issuance Logic

Issues NFTs to stakers based on their contributions.

```
def issue_nft_reward(self, user, usdt_value, pool_type):
    """ Issue NFT Rewards Based on Staking Pool with USDT Investment Data """
    nft_types = {
        "TreePlanting": "Mangrove Guardian NFT",
        "OceanCleaning": "Ocean Guardian NFT"
    }
    if pool_type in nft_types:
        return f"{nft_types[pool_type]} | Investment: {usdt_value} USDT issued to {user}"
    return "Invalid Pool Type"
```

NFT rewards serve as proof of environmental impact and incentivize users to contribute to sustainability efforts.

5. DAO Governance Logic

Token holders to participate in governance by voting on fund allocations.

```
def dao_proposal(self, proposal_id, proposed_amount):
    """ DAO Treasury Proposal Handling (Absolute Transparency) """
    if proposed_amount <= self.dao_treasury:
        return f"Proposal {proposal_id} approved, {proposed_amount} GF allocated. Publicly
trackable on-chain."
    return f"Proposal {proposal_id} denied due to insufficient funds."
```


This function ensures that DAO proposals are transparent and community-driven, with clear accountability.

6. Market Stability Logic

To prevent excessive price volatility, this function manages liquidity reserves.

```
def provide_liquidity(self, exchange, amount):
    """ Provide Liquidity to Exchanges (Paired with USDT) """
    if self.liquidity_reserve >= amount:
        self.liquidity_reserve -= amount
        return f"{amount} GF added to {exchange} liquidity pool (USDT Pairing Only)"
    return "Insufficient liquidity reserve"
```

By maintaining liquidity reserves, this function ensures smoother trading and stable pricing for GF tokens.

-  The Steering Committee can intervene during financial crises by allocating funds from the Garden Fleet wallet to the reserves, ensuring stability.

7. Drop Rewards Distribution

Drop rewards are distributed monthly to token holders as an additional incentive for long-term participation.

```
def distribute_drop_rewards(self):
    """ Distribute Monthly Token Drop Rewards """
    for user in self.balances:
        if self.balances[user] > 0:
            reward = self.balances[user] * 0.01
            self.balances[user] += reward
    return "Monthly drop rewards distributed"
```

8. GF Token Python Syntax Pseudo-Code

```
# Token Allocations

self.dao_treasury = 100_000_000 # 10% for DAO Treasury
self.liquidity_reserve = 30_000_000 # 3% for liquidity
self.fleet_operations_reserve = 350_000_000 # 35% for coastal restoration & ESG
services

self.staking_rewards_pool = 150_000_000 # 15% for staking rewards
self.ico_supply = 350_000_000 # 35% for ICO allocation
self.team_allocation = 20_000_000 # 2% for team and advisors

# Fleet Reserve Wallet

(100% of investment in staking goes here)
self.fleet_wallet = 0

# Steering Committee Authorized Wallets

self.steering_committee = set() # List of authorized addresses

# Vesting Schedules

self.vesting_schedules = {
    "ico": {"total": 350_000_000, "initial_unlock": 0.25, "vested_years": 2},
    "fleet_reserve": {"total": 350_000_000, "vested_years": 10},
    "staking_rewards": {"total": 150_000_000, "vesting_period": "monthly"},
    "dao_treasury": {"total": 100_000_000, "community_controlled": True},
    "team": {"total": 20_000_000, "cliff": 1, "vesting_years": 4}
}

def transfer(self, sender, receiver, amount):
    """ Transfer GF Tokens between users """
    if self.balances.get(sender, 0) >= amount:
        self.balances[sender] -= amount
        self.balances[receiver] = self.balances.get(receiver, 0) + amount
        return True
    return False

def stake(self, user, amount, usdt_value, pool_type):
    """ Stake GF Tokens for Cleanup & Reforestation (Funds Go to Fleet Wallet) """
    if self.balances.get(user, 0) >= amount:
        self.balances[user] -= amount
        self.staked_balances[user] = self.staked_balances.get(user, 0) + amount
        self.staking_start_time[user] = time.time()
        self.investment_in_usdt[user] = usdt_value

        # 100% of staking investment is sent to the Fleet Wallet
        self.fleet_wallet += usdt_value

        self.issue_nft_reward(user, usdt_value, pool_type)
        return True
    return False

def issue_nft_reward(self, user, usdt_value, pool_type):
    """ Issue NFT Rewards Based on Staking Pool with USDT Investment Data """
    nft_types = {
        "TreePlanting": "Mangrove Guardian NFT",
        "OceanCleaning": "Ocean Guardian NFT"
    }
    if pool_type in nft_types:
        return f"{nft_types[pool_type]} | Investment: {usdt_value} USDT issued to {user}"
    return "Invalid Pool Type"

def distribute_staking_rewards(self):
    """ Distribute Monthly Staking Rewards (5%) """
    for user, amount_staked in self.staked_balances.items():
        reward = amount_staked * 0.05
        self.balances[user] = self.balances.get(user, 0) + reward
        self.staking_rewards_pool -= reward
    return "Monthly staking rewards distributed"

def distribute_drop_rewards(self):
    """ Distribute Monthly Token Drop Rewards (1%) to Holders """
    for user in self.balances:
        if self.balances[user] > 0:
            reward = self.balances[user] * 0.01
            self.balances[user] += reward
    return "Monthly drop rewards distributed"

def dao_proposal(self, proposal_id, proposed_amount):
    """ DAO Treasury Proposal Handling (Absolute Transparency) """
    if proposed_amount <= self.dao_treasury:
        return f"Proposal {proposal_id} approved, {proposed_amount} GF allocated. Publicly trackable on-chain."
    return f"Proposal {proposal_id} denied due to insufficient funds."

def steering_committee_reward(self, user, amount):
    """ Steering Committee Distributes Extra Token Rewards (Airdrops) """
    if user in self.balances and amount <= self.staking_rewards_pool:
        self.balances[user] += amount
        self.staking_rewards_pool -= amount
        return f"Steering Committee distributed {amount} GF to {user}"
    return "Insufficient balance in rewards pool or invalid user"

def release_vested_tokens(self, category):
    """ Release Vesting Funds Based on Schedule """
    if category in self.vesting_schedules:
        schedule = self.vesting_schedules[category]
        if "vested_years" in schedule:
            yearly_release = schedule["total"] / schedule["vested_years"]
            return f"{yearly_release} GF released for {category}"
        elif "vesting_period" in schedule and schedule["vesting_period"] == "monthly":
            monthly_release = schedule["total"] / 120 # 10 years = 120 months
            return f"{monthly_release} GF released for {category} this month"
    return "Invalid vesting category"

def provide_liquidity(self, exchange, amount):
    """ Provide Liquidity to Exchanges (Paired with USDT) """
    if self.liquidity_reserve >= amount:
        self.liquidity_reserve -= amount
        return f"{amount} GF added to {exchange} liquidity pool (USDT Pairing Only)"
    return "Insufficient liquidity reserve"
```

Conclusion

The **Garden Fleet (GF) Token** represents a scalable, blockchain-powered solution for funding real-world environmental restoration. Through **ReFi incentives, staking rewards, and DAO governance**, GF ensures sustainable impact while maintaining financial viability. By integrating on-chain impact tracking, verifiable ESG compliance, and decentralized governance, GF stands as an innovative model for environmental sustainability and blockchain transparency.

Prior to deployment, all smart contract code will be **externally audited and penetration tested** to ensure security and robustness. This process will verify that the token's functionality is **secure, free from vulnerabilities, and fully aligned with industry best practices**.