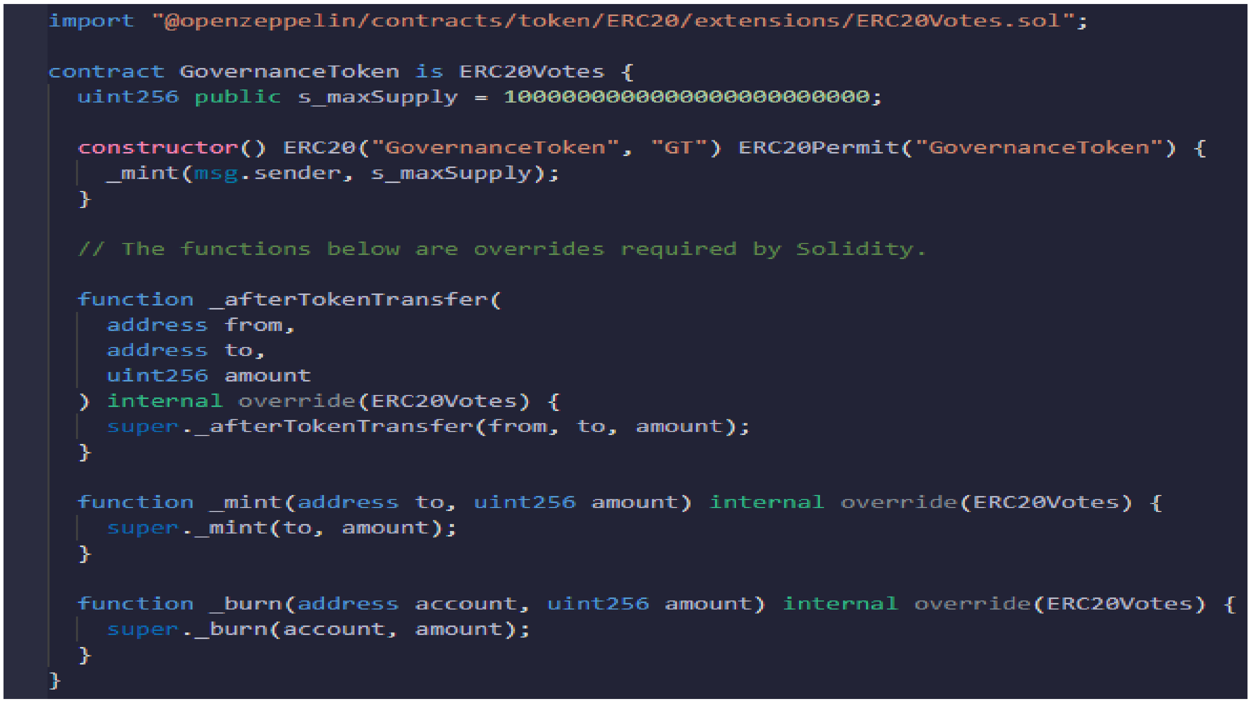
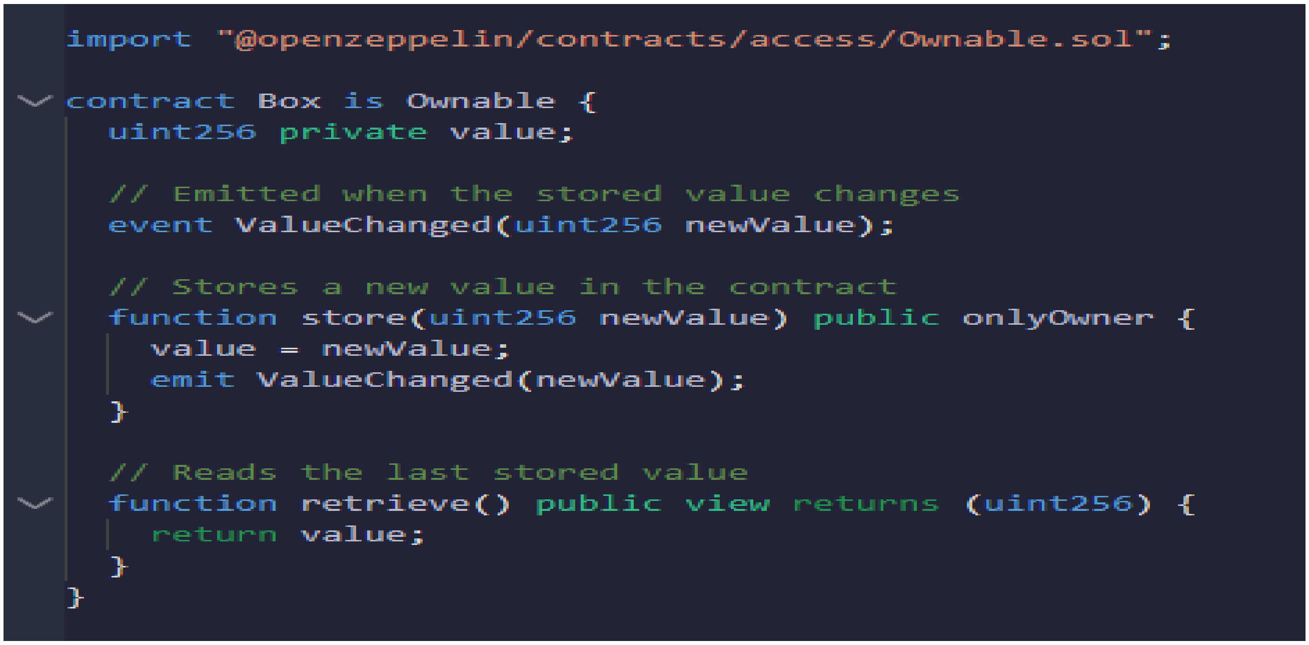
Hardhat DAOs



```
import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol";

contract GovernanceToken is ERC20Votes {
  uint256 public s_maxSupply = 1000000000000000000000000;

  constructor() ERC20("GovernanceToken", "GT") ERC20Permit("GovernanceToken") {
    _mint(msg.sender, s_maxSupply);
  }

  // The functions below are overrides required by Solidity.

  function _afterTokenTransfer(
    address from,
    address to,
    uint256 amount
  ) internal override(ERC20Votes) {
    super._afterTokenTransfer(from, to, amount);
  }

  function _mint(address to, uint256 amount) internal override(ERC20Votes) {
    super._mint(to, amount);
  }

  function _burn(address account, uint256 amount) internal override(ERC20Votes) {
    super._burn(account, amount);
  }
}
```



```
import "@openzeppelin/contracts/access/Ownable.sol";

contract Box is Ownable {
  uint256 private value;

  // Emitted when the stored value changes
  event ValueChanged(uint256 newValue);

  // Stores a new value in the contract
  function store(uint256 newValue) public onlyOwner {
    value = newValue;
    emit ValueChanged(newValue);
  }

  // Reads the last stored value
  function retrieve() public view returns (uint256) {
    return value;
  }
}
```