```solidity
// SPDX-License-Identifier: MIT
// OpenZeppelin Contracts (last updated v4.5.0) (access/AccessControlEnumerable.sol)

pragma solidity ^0.8.0;

import "./IAccessControlEnumerable.sol";
import "./AccessControl.sol";
import "../utils/structs/EnumerableSet.sol";

/**
 * @dev Extension of {AccessControl} that allows enumerating the members of each role.
 */
abstract contract AccessControlEnumerable is IAccessControlEnumerable, AccessControl {
    using EnumerableSet for EnumerableSet.AddressSet;

    mapping(bytes32 => EnumerableSet.AddressSet) private _roleMembers;

    /**
     * @dev See {IERC165-supportsInterface}.
     */
    function supportsInterface(bytes4 interfaceId) public view virtual override returns (bool) {
        return interfaceId == type(IAccessControlEnumerable).interfaceId || super.supportsInterface(interfaceId);
    }

    /**
     * @dev Returns one of the accounts that have `role`. `index` must be a
     * value between 0 and {getRoleMemberCount}, non-inclusive.
     *
     * Role bearers are not sorted in any particular way, and their ordering may
     * change at any point.
     *
     * WARNING: When using {getRoleMember} and {getRoleMemberCount}, make sure
     * you perform all queries on the same block. See the following
     * https://forum.openzeppelin.com/t/iterating-over-elements-on-enumerableset-in-openzeppelin-contracts/2296[forum post]
     * for more information.
     */
    function getRoleMember(bytes32 role, uint256 index) public view virtual override returns (address) {
        return _roleMembers[role].at(index);
    }

    /**
     * @dev Returns the number of accounts that have `role`. Can be used
     * together with {getRoleMember} to enumerate all bearers of a role.
     */
    function getRoleMemberCount(bytes32 role) public view virtual override returns (uint256) {
        return _roleMembers[role].length();
    }

    /**
     * @dev Overload {_grantRole} to track enumerable memberships
     */
    function _grantRole(bytes32 role, address account) internal virtual override {
        super._grantRole(role, account);
        _roleMembers[role].add(account);
    }

    /**
     * @dev Overload {_revokeRole} to track enumerable memberships
     */
    function _revokeRole(bytes32 role, address account) internal virtual override {
        super._revokeRole(role, account);
        _roleMembers[role].remove(account);
    }
}

```