```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract LandRegistry {
    // Structure to represent a land parcel
    struct Land {
        uint256 id;
        string location;
        uint256 area; // Area in square meters
        address owner;
        bool registered;
    }

    // Mapping from land ID to Land details
    mapping(uint256 => Land) public lands;

    // Event emitted when a new land is registered
    event LandRegistered(uint256 indexed landId, string location, uint256 area, address indexed
owner);

    // Event emitted when land ownership is transferred
    event OwnershipTransferred(uint256 indexed landId, address indexed oldOwner, address
indexed newOwner);

    // Register a new land parcel
    function registerLand(uint256 _id, string memory _location, uint256 _area) public {
        require(!lands[_id].registered, "Land already registered");

        lands[_id] = Land({
            id: _id,
            location: _location,
            area: _area,
            owner: msg.sender,
            registered: true
        });

        emit LandRegistered(_id, _location, _area, msg.sender);
    }

    // Transfer ownership of a land parcel
    function transferOwnership(uint256 _id, address _newOwner) public {
        Land storage land = lands[_id];
        require(land.registered, "Land not registered");
```

```
        require(land.owner == msg.sender, "Only the owner can transfer ownership");

        address oldOwner = land.owner;
        land.owner = _newOwner;

        emit OwnershipTransferred(_id, oldOwner, _newOwner);
    }

    // Get land details
    function getLand(uint256 _id) public view returns (uint256, string memory, uint256, address,
bool) {
        Land memory land = lands[_id];
        require(land.registered, "Land not registered");

        return (land.id, land.location, land.area, land.owner, land.registered);
    }

    // Check if land is registered
    function isLandRegistered(uint256 _id) public view returns (bool) {
        return lands[_id].registered;
    }
}
```