# 315.https://stackoverflow.com/questions/71669119/how-to-directly-interact-with-already-deployed-smart-contracts

**T:**How to directly interact with already deployed smart contracts?

**Q:**I am tasked with interacting with a function in a smart contract that has already been deployed. I know the contract address and the function signature and I have a solution that uses interfaces. However, this function I am interacting with then sends an NFT to msg.sender which in this case is the contract with the interface solution as opposed to my personal account.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
//SPDX-License-Identifier: Unlicensepragma solidity ^0.8.4;interface ITargetContract { function addWhitelist(bytes32 \_something, string memory \_id) external;}contract MyContract { function addWhitelist( address \_t, bytes32 \_something, string memory \_id ) public { ITargetContract(\_t).addWhitelist(\_something, \_id); }}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
So to avoid this NFT being sent to my contract, can I interact with the deployed contract directly with my personal account if I only know the contract address and function signature?

1 **Answer**

**A1:**The only case when msg.sender is relayed is delegatecall. However, it also uses storage of the caller contract (and not of the called one), so it's mostly used in proxy contracts.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
There's no way to both relay the msg.sender and use the storage of the called contract.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Assuming the target contract implements the ERC721 standard, you can implement the onERC721Received() function and resend the token to the end user after you've received it.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
contract MyContract { mapping (string => address) tokenUsers; function addWhitelist( address \_t, bytes32 \_something, string memory \_id ) public { // store the end user address by the token ID tokenUsers[\_id] = msg.sender; ITargetContract(\_t).addCandidateIdToWhitelist(\_something, \_id); } function onERC721Received(address \_operator, address \_from, uint256 \_tokenId, bytes \_data) external returns(bytes4) { // resend the token from your contract to the user ITargetContract.safeTransferFrom( address(this), tokenUsers[\_tokenId], // TODO: transfer functions accept the token ID as uint, not as string \_stringToUint(\_tokenId) ); }}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

**C1:**Hi Petr, thank you for your response. This link shows my implementation with interfaces (ropsten.etherscan.io/tx/…), and then an implementation where it appears people are interacting with the contract (0x006eCb9efFA194df5ce192AF1e15eFBaa8a3c87b) directly(ropsten.etherscan.io/tx/…). I just wonder how they did it without a contract in the middle?