# 140.https://stackoverflow.com/questions/72695638/getting-error-erc721-operator-query-for-non-existent-token

**T:**Getting error Erc721 : operator Query for non existent token

**Q:**I am making a nft rental contractWhen calling setUser or any function in the contract error is generated -- ERC721: operator query for nonexistent token.Even if the token is minted .Any guess why?The contract is as followshttps://github.com/shristivyas/SmartContracts/blob/main/nft-renting.sol

1 **Answer**

**A1:**The error is thrown by the OpenZeppelin \_isApprovedOrOwner() function when the token ID is not minted.  
  
There's no function to mint a token in your current implementation.  
  
Proof of concept: When the token exists, the error no longer appears.  
  
function setUser(uint256 tokenId, address user, uint64 expires) public virtual override(IERC4907){ if (\_exists(tokenId) == false) { \_mint(msg.sender, tokenId); // mint the token if it doesn't exist yet } require(\_isApprovedOrOwner(msg.sender, tokenId), "ERC4907: transfer caller is not owner nor approved"); UserInfo storage info = \_users[tokenId]; info.user = user; info.expires = expires; emit UpdateUser(tokenId, user, expires);}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

**C1:**I am just a beginnner . So can you tell me what if the token is minted by any other contract , and we pass the token id as argument in this contract?

**C2:**@GauravMishra In this context, it doesn't really matter by which address is the token minted - but it's important on what contract is the token minted. Your snippet and my answer are valid only for tokens minted on this contract. If you want to validate an owner of a specific token ID on an external collection contract, you need to invoke their ownerOf() and getApproved() public functions. Example: if(otherCollection.ownerOf(tokenId) == msg.sender || otherCollection.getApproved(tokenId) == msg.sender) { } where otherCollection is a pointer to the external contract.