# 153.https://stackoverflow.com/questions/72655473/tokenuri-is-not-defined-no-undef

**T:**'tokenURI' is not defined no-undef

**Q:**I am creating a NFT minter website using the following tutorial https://docs.alchemy.com/alchemy/tutorials/nft-minter.I have followed the tutorial to the last step. I experience an error "tokenURI' is not defined no-undef" when I am trying to run this code in localhost. Do you have any suggestions to resolve the issue?  
  
import React from 'react';import ReactDOM from 'react-dom';import {pinJSONToIPFS} from './pinata.js'require('dotenv').config();const alchemyKey = process.env.REACT\_APP\_ALCHEMY\_KEY;const { createAlchemyWeb3 } = require("@alch/alchemy-web3");const web3 = createAlchemyWeb3(alchemyKey); const contractABI = require('../contract-abi.json')const contractAddress = "0x4C4a07F737Bf57F6632B6CAB089B78f62385aCaE";export const mintNFT = async(url, name, description) => { //error handling if (url.trim() == "" || (name.trim() == "" || description.trim() == "")) { return { success: false, status: "❗Please make sure all fields are completed before minting.", } } //make metadata const metadata = new Object(); metadata.name = name; metadata.image = url; metadata.description = description; //make pinata call const pinataResponse = await pinJSONToIPFS(metadata); if (!pinataResponse.success) { return { success: false, status: "😢 Something went wrong while uploading your tokenURI.", } } const tokenURI = pinataResponse.pinataUrl; }window.contract = await new web3.eth.Contract(contractABI, contractAddress);//set up your Ethereum transactionconst transactionParameters = { to: contractAddress, // Required except during contract publications. from: window.ethereum.selectedAddress, // must match user's active address. 'data': window.contract.methods.mintNFT(window.ethereum.selectedAddress, tokenURI).encodeABI()//make call to NFT smart contract };//sign the transaction via Metamasktry {const txHash = await window.ethereum .request({ method: 'eth\_sendTransaction', params: [transactionParameters], });return { success: true, status: "✅ Check out your transaction on Etherscan: https://ropsten.etherscan.io/tx/" + txHash}} catch (error) {return { success: false, status: "😥 Something went wrong: " + error.message}}export const connectWallet = async () => { if (window.ethereum) { try { const addressArray = await window.ethereum.request({ method: "eth\_requestAccounts", }); const obj = { status: "👆🏽 Write a message in the text-field above.", address: addressArray[0], }; return obj; } catch (err) { return { address: "", status: "😥 " + err.message, }; } } else { return { address: "", status: ( <span> <p> {" "} 🦊{" "} <a target="\_blank" href={`https://metamask.io/download.html`}> You must install Metamask, a virtual Ethereum wallet, in your browser. </a> </p> </span> ), }; } }; export const getCurrentWalletConnected = async () => { if (window.ethereum) { try { const addressArray = await window.ethereum.request({ method: "eth\_accounts", }); if (addressArray.length > 0) { return { address: addressArray[0], status: "👆🏽 Write a message in the text-field above.", }; } else { return { address: "", status: "🦊 Connect to Metamask using the top right button.", }; } } catch (err) { return { address: "", status: "😥 " + err.message, }; } } else { return { address: "", status: ( <span> <p> {" "} 🦊{" "} <a target="\_blank" href={`https://metamask.io/download.html`}> You must install Metamask, a virtual Ethereum wallet, in your browser. </a> </p> </span> ), }; } };  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

1 **Answer**

**A1:**After checking the docs you linked, it seems you're putting this code outside the function, when instead it should be inside.  
  
  
  
import React from 'react';import { pinJSONToIPFS } from './pinata.js'require('dotenv').config();const alchemyKey = process.env.REACT\_APP\_ALCHEMY\_KEY;const { createAlchemyWeb3 } = require("@alch/alchemy-web3");const web3 = createAlchemyWeb3(alchemyKey);const contractABI = require('../contract-abi.json')const contractAddress = "0x4C4a07F737Bf57F6632B6CAB089B78f62385aCaE";let tokenURI;export const mintNFT = async (url, name, description) => { //error handling if (url.trim() == "" || (name.trim() == "" || description.trim() == "")) { return { success: false, status: "❗Please make sure all fields are completed before minting.", } } //make metadata const metadata = new Object(); metadata.name = name; metadata.image = url; metadata.description = description; //make pinata call const pinataResponse = await pinJSONToIPFS(metadata); if (!pinataResponse.success) { return { success: false, status: "😢 Something went wrong while uploading your tokenURI.", } } const tokenURI = pinataResponse.pinataUrl; window.contract = await new web3.eth.Contract(contractABI, contractAddress); //set up your Ethereum transaction const transactionParameters = { to: contractAddress, // Required except during contract publications. from: window.ethereum.selectedAddress, // must match user's active address. 'data': window.contract.methods.mintNFT(window.ethereum.selectedAddress, tokenURI).encodeABI()//make call to NFT smart contract }; //sign the transaction via Metamask try { const txHash = await window.ethereum .request({ method: 'eth\_sendTransaction', params: [transactionParameters], }); return { success: true, status: "✅ Check out your transaction on Etherscan: https://ropsten.etherscan.io/tx/" + txHash } } catch (error) { return { success: false, status: "😥 Something went wrong: " + error.message } }}export const connectWallet = async () => { if (window.ethereum) { try { const addressArray = await window.ethereum.request({ method: "eth\_requestAccounts", }); const obj = { status: "👆🏽 Write a message in the text-field above.", address: addressArray[0], }; return obj; } catch (err) { return { address: "", status: "😥 " + err.message, }; } } else { return { address: "", status: ( <span> <p> {" "} 🦊{" "} <a target="\_blank" href={`https://metamask.io/download.html`}> You must install Metamask, a virtual Ethereum wallet, in your browser. </a> </p> </span> ), }; }};export const getCurrentWalletConnected = async () => { if (window.ethereum) { try { const addressArray = await window.ethereum.request({ method: "eth\_accounts", }); if (addressArray.length > 0) { return { address: addressArray[0], status: "👆🏽 Write a message in the text-field above.", }; } else { return { address: "", status: "🦊 Connect to Metamask using the top right button.", }; } } catch (err) { return { address: "", status: "😥 " + err.message, }; } } else { return { address: "", status: ( <span> <p> {" "} 🦊{" "} <a target="\_blank" href={`https://metamask.io/download.html`}> You must install Metamask, a virtual Ethereum wallet, in your browser. </a> </p> </span> ), }; }};  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]