



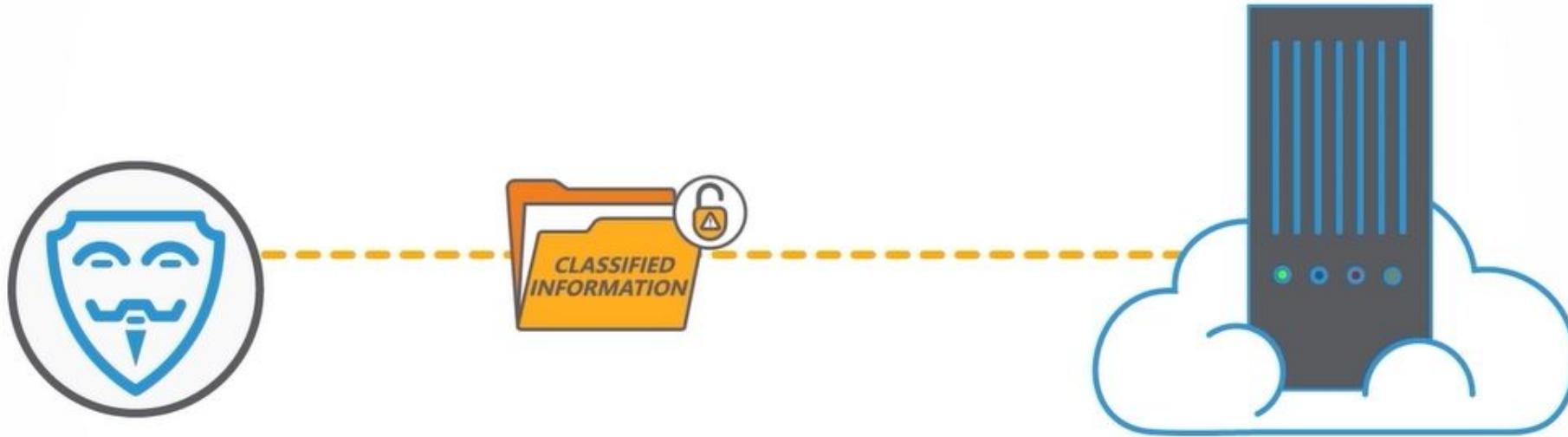
**SECURE
CODE
WARRIOR**

INSUFFICIENT VALIDATION

The Insufficient Validation of input parameters could let attackers send in data that allows them to bypass security procedures,



gain access to restricted information,

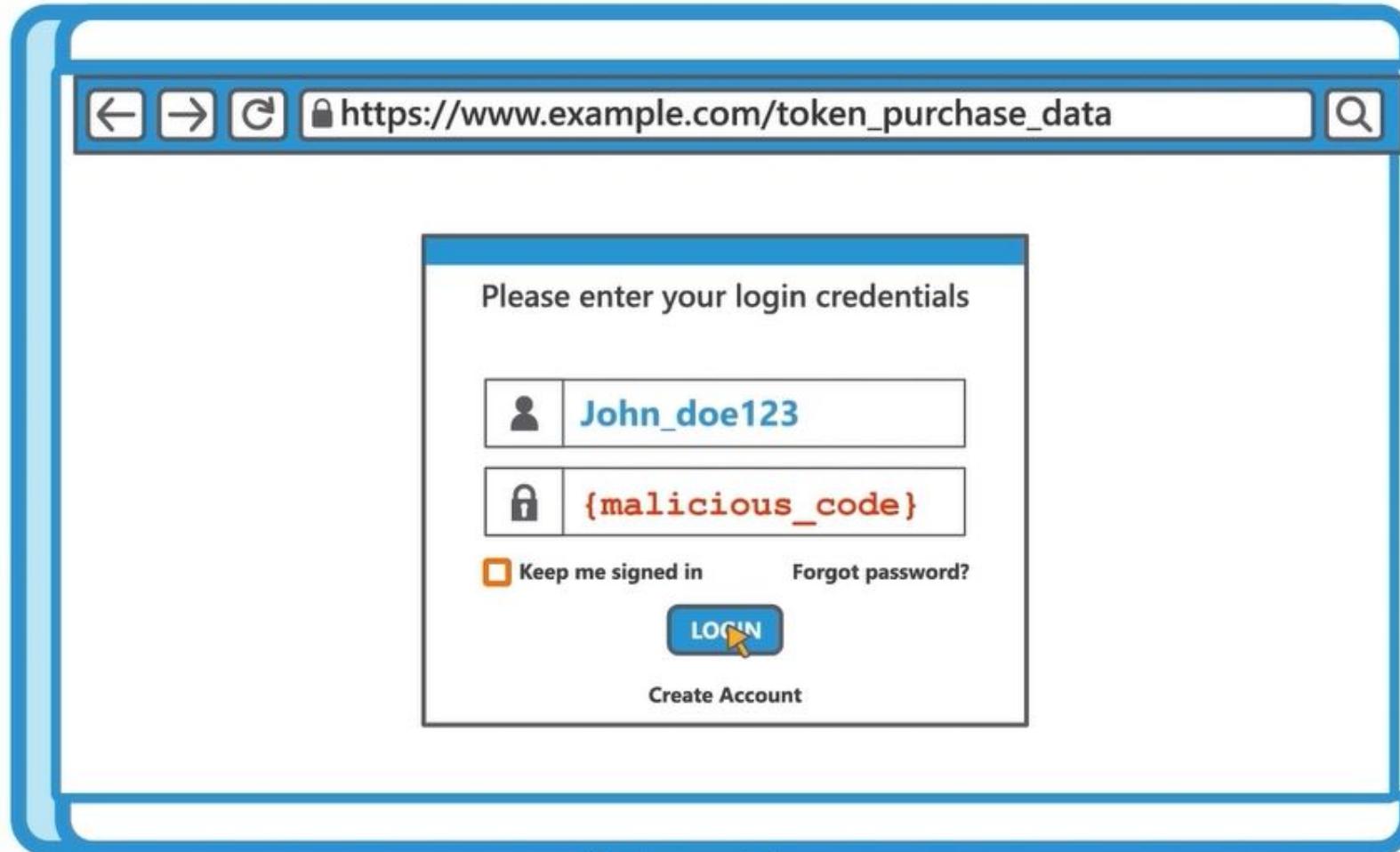


or cause services to crash.

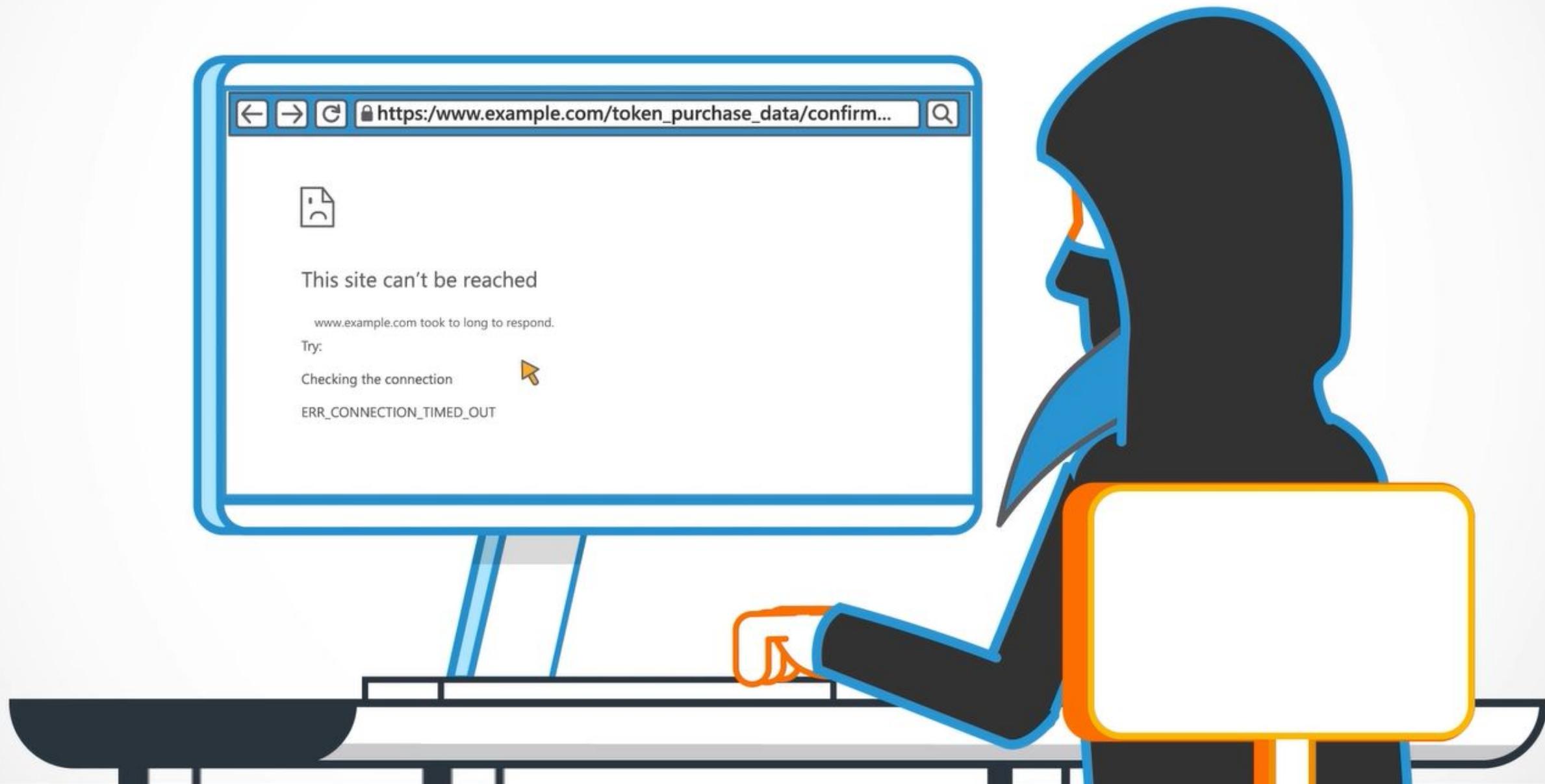


WHERE DOES THIS
VULNERABILITY OCCUR?

It can occur in any function that requires external inputs to complete its operation.

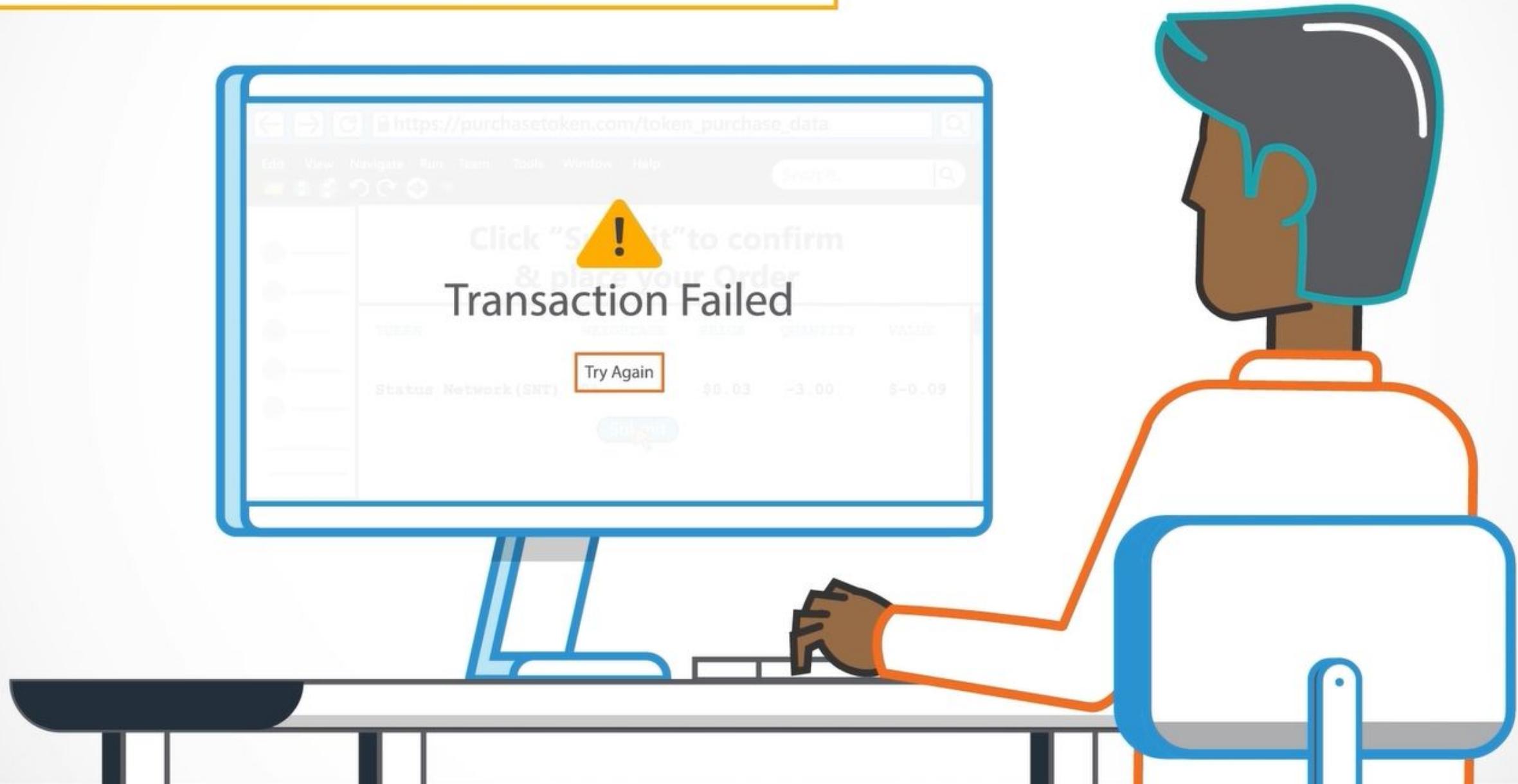


When attacked, these operations may fail in unexpected ways or throw exceptions.



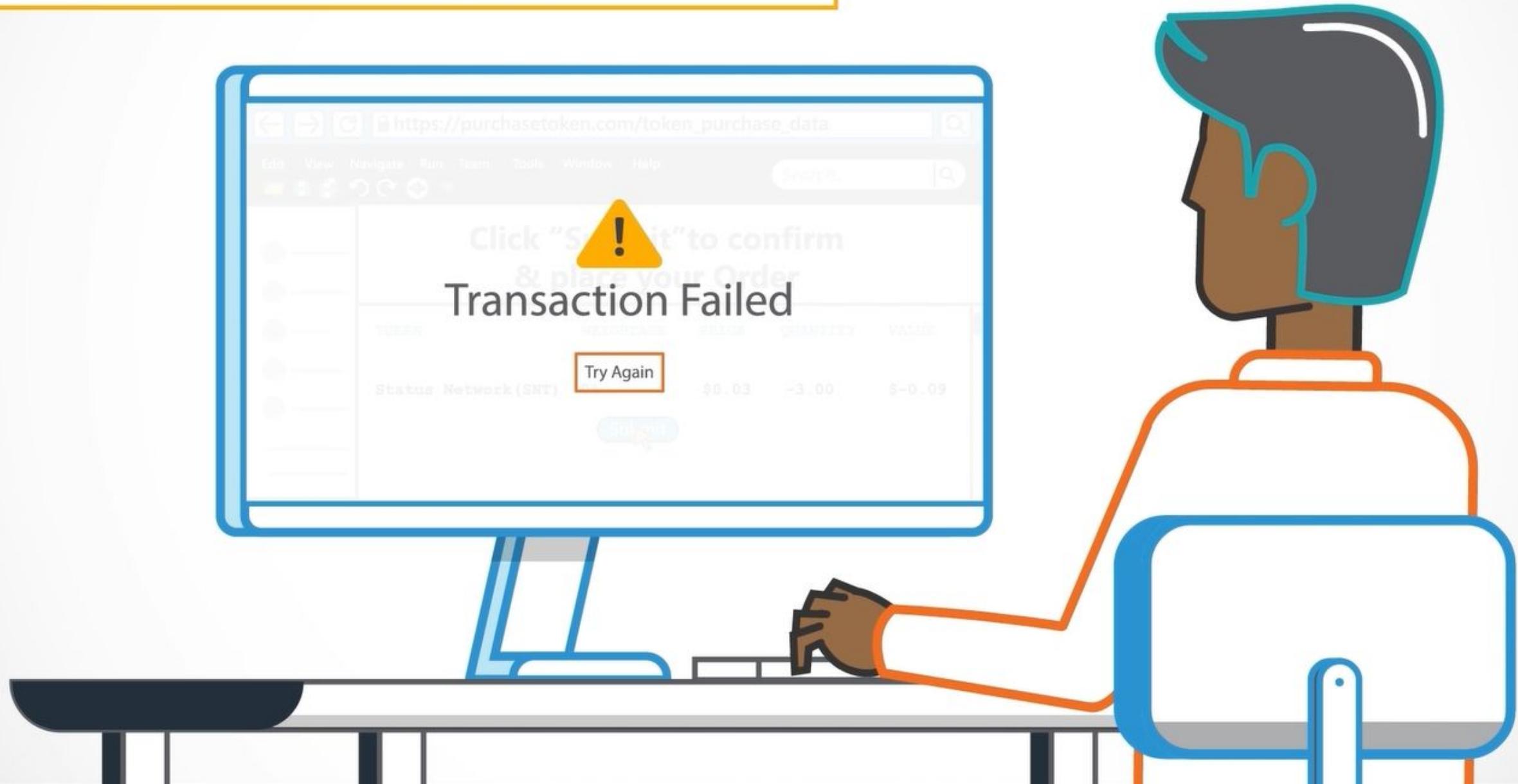
A common example is when users supply negative values in transactions, and data processing fails as a result.

-3349.62



A common example is when users supply negative values in transactions, and data processing fails as a result.

-3349.62



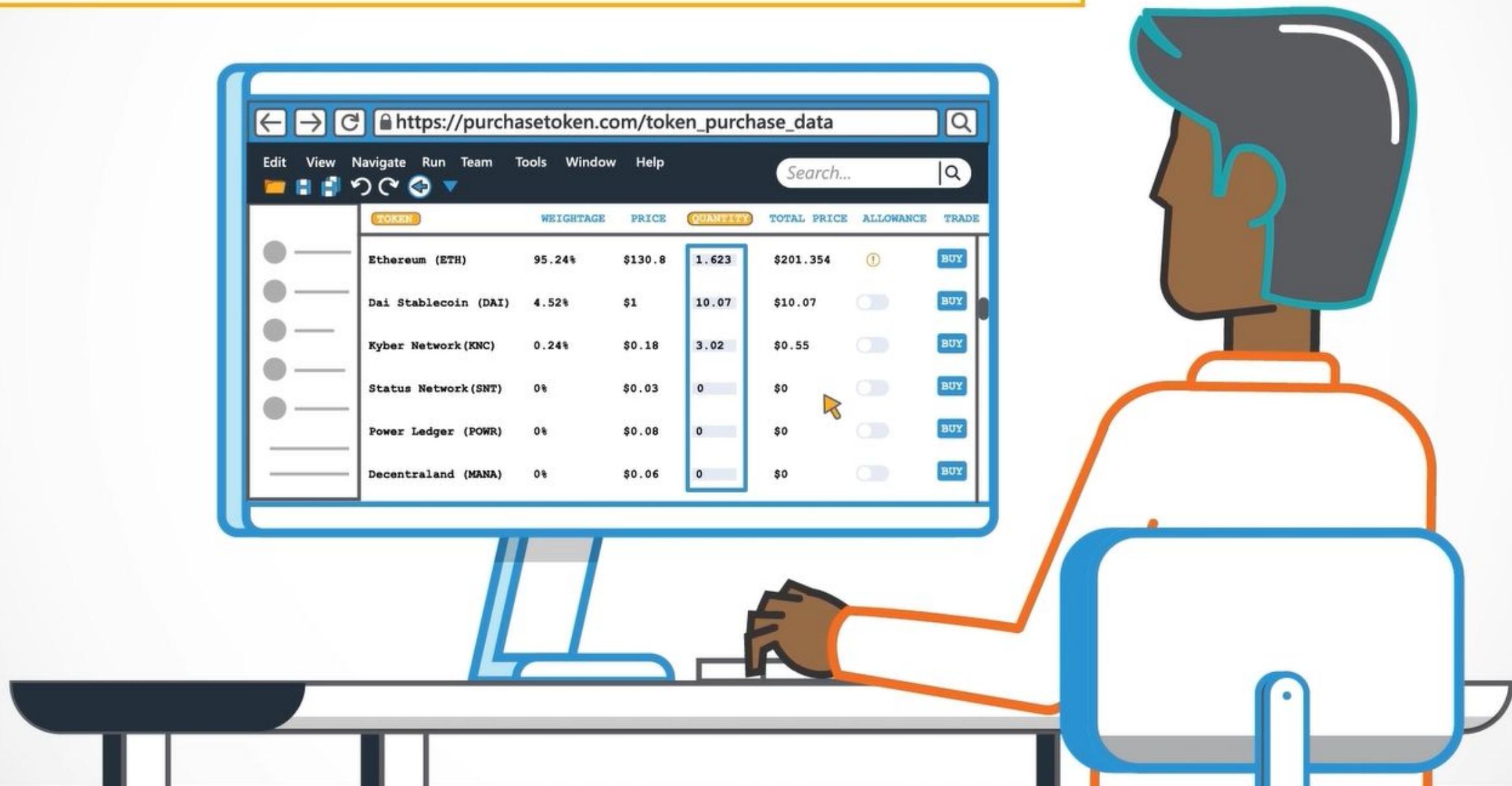
LET'S LOOK AT AN EXAMPLE

In this application, users can pre-purchase tokens to access functionality in the application.

The screenshot shows a web browser window with the URL https://purchasetoken.com/token_purchase_data. The page displays a table of tokens for purchase, with a cursor pointing at the allowance switch for the Power Ledger (POWR) row.

TOKEN	WEIGHTAGE	PRICE	QUANTITY	TOTAL PRICE	ALLOWANCE	TRADE
Ethereum (ETH)	95.24%	\$130.8	1.623	\$201.354	<input type="checkbox"/>	<button>BUY</button>
Dai Stablecoin (DAI)	4.52%	\$1	10.07	\$10.07	<input type="checkbox"/>	<button>BUY</button>
Kyber Network (KNC)	0.24%	\$0.18	3.02	\$0.55	<input type="checkbox"/>	<button>BUY</button>
Status Network (SNT)	0%	\$0.03	0	\$0	<input type="checkbox"/>	<button>BUY</button>
Power Ledger (POWR)	0%	\$0.08	0	\$0	<input checked="" type="checkbox"/>	<button>BUY</button>
Decentraland (MANA)	0%	\$0.06	0	\$0	<input type="checkbox"/>	<button>BUY</button>

The application asks the user how many tokens they want to spend on this transaction in the platform, and provides a free text field for a response.



Unfortunately, the application does not validate the value received from the user.

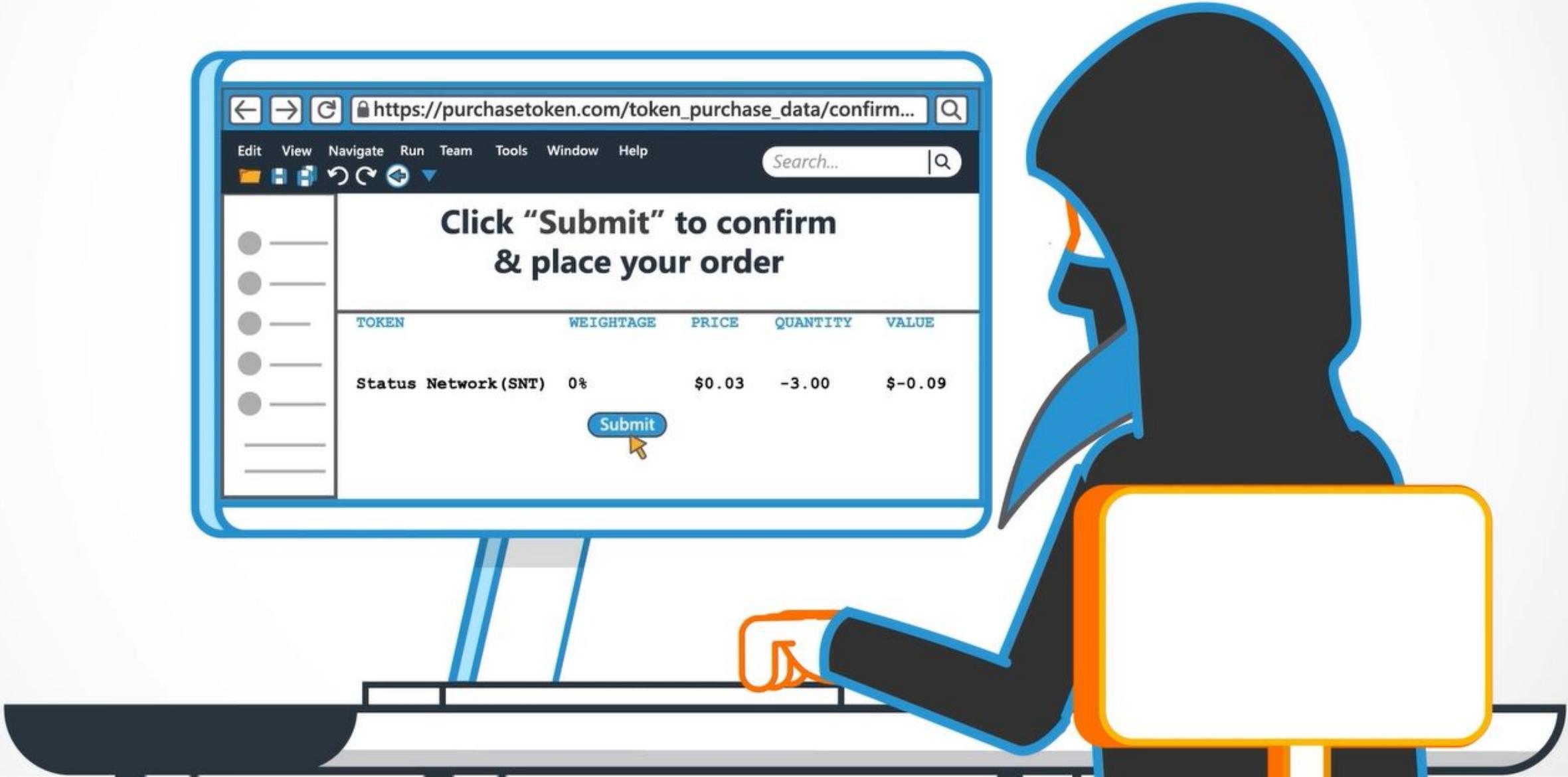
NO INPUT VALIDATION



The screenshot shows a web browser window with the URL https://purchasetoken.com/token_purchase_data. The page displays a table of tokens with columns: TOKEN, WEIGHTAGE, PRICE, QUANTITY, TOTAL PRICE, ALLOWANCE, and TRADE. The 'TOTAL PRICE' column for Status Network (SNT) contains the value '\$-0.09', which is highlighted with a red border. A blue 'BUY' button is located in the TRADE column for this row. An orange arrow points to the 'BUY' button for SNT, indicating that clicking it would result in an invalid transaction due to the negative price.

TOKEN	WEIGHTAGE	PRICE	QUANTITY	TOTAL PRICE	ALLOWANCE	TRADE
Ethereum (ETH)	95.24%	\$130.8	1.623	\$201.354	!	<button>BUY</button>
Dai Stablecoin (DAI)	4.52%	\$1	10.07	\$10.07	<input type="checkbox"/>	<button>BUY</button>
Kyber Network (KNC)	0.24%	\$0.18	3.02	\$0.55	<input type="checkbox"/>	<button>BUY</button>
Status Network (SNT)	0%	\$0.03	-3.00	\$-0.09	<input type="checkbox"/>	<button>BUY</button>
Power Ledger (POWR)	0%	\$0.08	0	\$0	<input type="checkbox"/>	<button>BUY</button>
Decentraland (MANA)	0%	\$0.06	0	\$0	<input type="checkbox"/>	<button>BUY</button>

Realizing this, an attacker provides a negative value into the form and presses “Submit”.

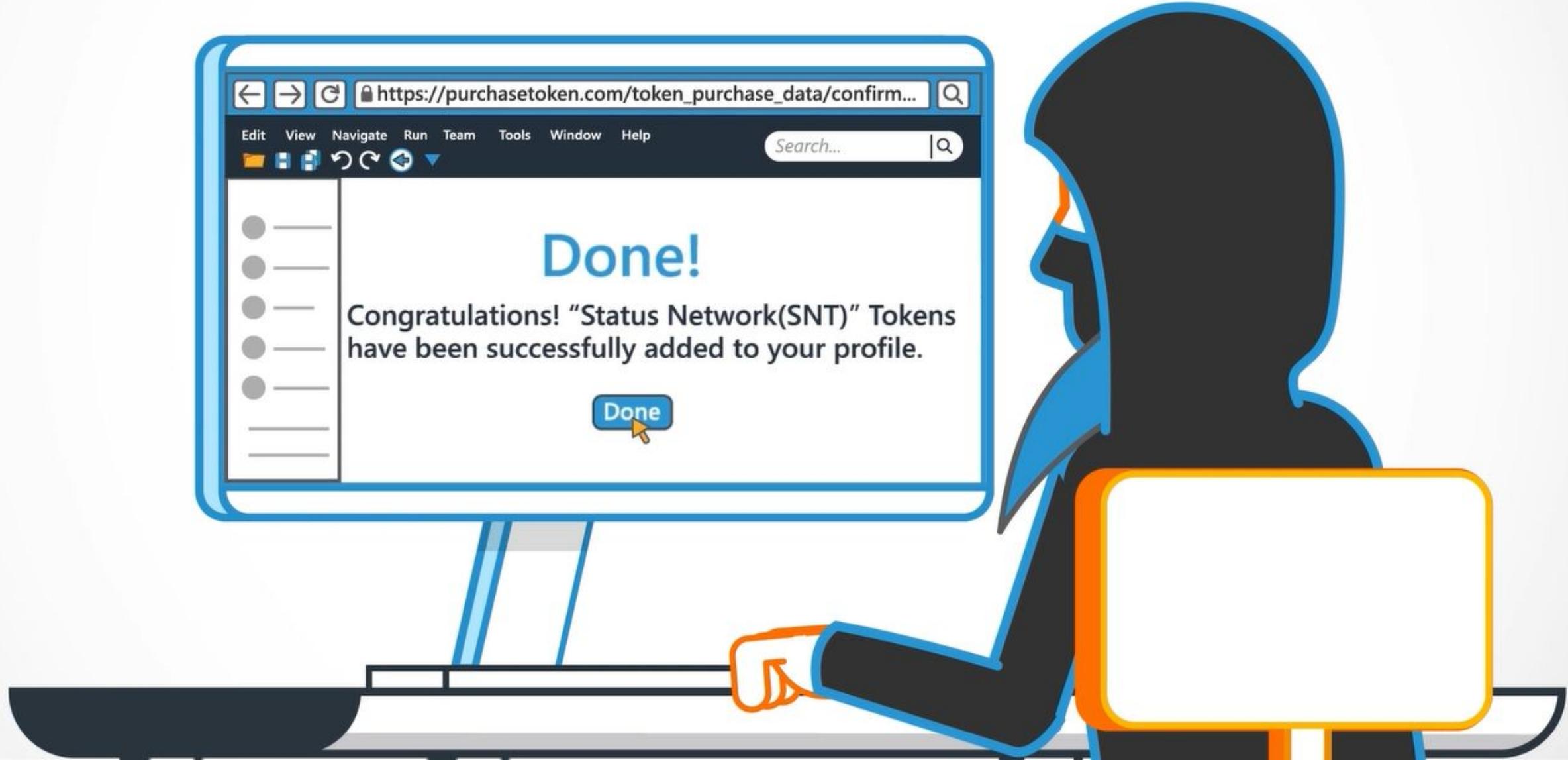
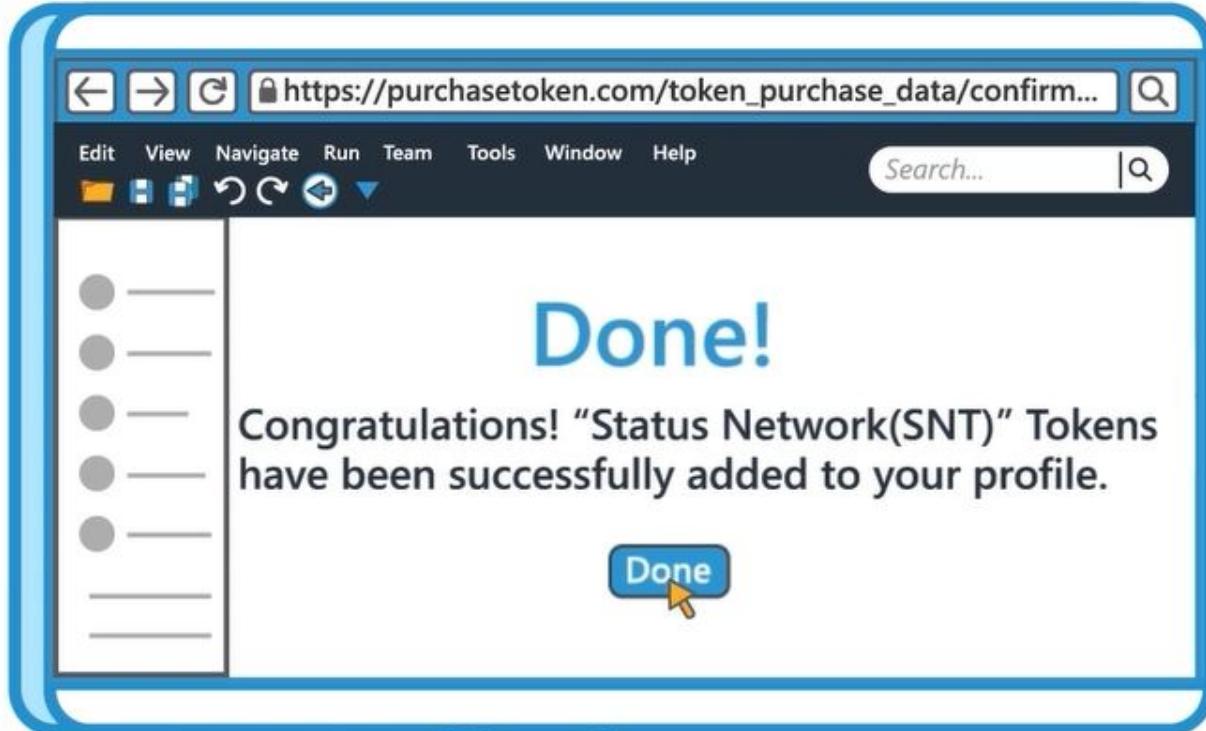


The image shows a stylized illustration of a person sitting at a desk, viewed from the side. The person has dark hair and is wearing a blue hoodie. They are interacting with a computer monitor which displays a web browser window. The browser's address bar shows the URL https://purchasetoken.com/token_purchase_data/confirm.... The main content of the page is a confirmation message: "Click ‘Submit’ to confirm & place your order". Below this message is a table with the following data:

TOKEN	WEIGHTAGE	PRICE	QUANTITY	VALUE
Status Network (SNT)	0%	\$0.03	-3.00	\$-0.09

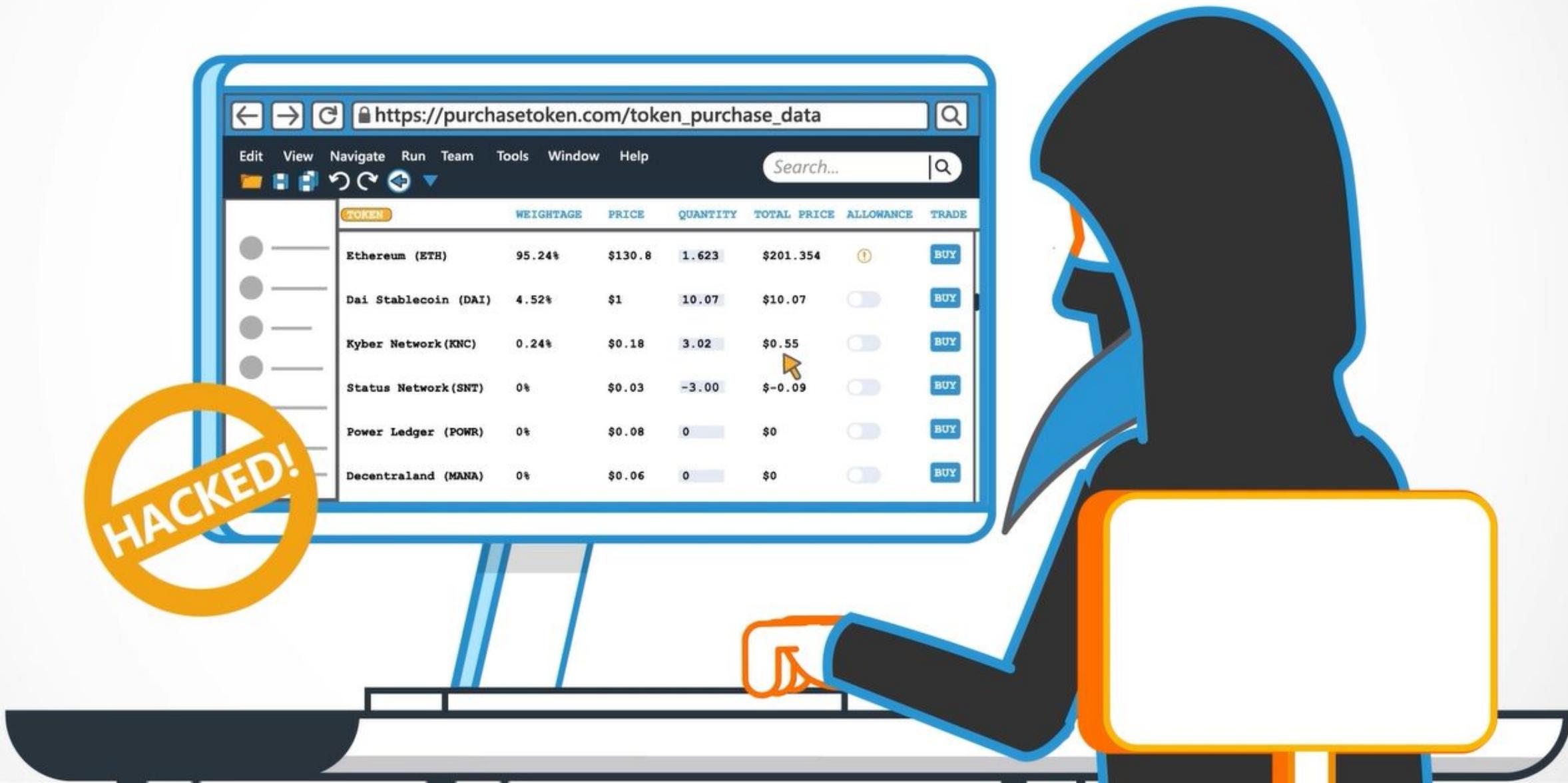
A blue "Submit" button is located at the bottom of the form. A cursor arrow is pointing towards this button.

Now, instead of removing the entered amount of tokens,
the application adds the tokens to the attackers profile.



\$\$\$

The attacker can use this to make purchases in the application without paying, from now now on.



To prevent Insufficient Validation, developers should :

- ④ Consider all potential areas where inputs can enter the software and assume all input is malicious
- ④ Consider using an allowlist with valid input parameters
- ④ Ensure any security checks performed on the client side are duplicated on the server side
- ④ Always convert inputs into the appropriate, expected data type

Congratulations, you have now completed this module!



**SECURE
CODE
WARRIOR**