Blockchain Assignment 3 Write-up

The contract inherits from ERC20Capped, so the maximum token supply is defined and set in the constructor upon deployment. The cap is immutably enforced by the mint function, and that is used by the mint and airdrop functions. For the pausable transfers, it’s inheriting from ERC20Pausable, and it causes it to get the ability to half all transfers in an emergency. It uses pause and unpause functions to trigger the function. For the access control roles, it uses AccessControl for a flexible and secure permissioning system. It uses MINTER\_ROLE for creating new tokens and PAUSER\_ROLE for managing the pausable state.

For the gas saving, the single batch airdrop consumed 109,180 gas, but all three consumed a total of 110,448 gas. It saved around 1.15% gas, or around 1,268 gas. The reason it saved a little was because of the base transaction cost. Every transaction has an overhead cost of 21,000 gas, so the method of using 3 separate transfers paid the cost 3 times, where the batch only paid it once. The savings were only realized by consolidating the fixed transaction overhead into a single operation.

There were several challenges encountered. My project was blocked by npm errors, including “MODULE\_NOT\_FOUND” for the hardhat-chai-matchers and an ERESOLVE dependency conflict. I had to do a clean installation and use the –legacy-peer-deps flag to resolve the version mismatch. I also had a variable mismatch with some places saying PRIVKEY and others saying PRIVATE\_KEY, so to fix it, I changed them all to PRIVATE\_KEY. I also had an error with the deploy due to internet connectivity. It didn’t trust the website since I was on a public wifi network at work.