Introduction

Now that you have undergone training in identifying your language issues and understanding them, you are ready to correct them. In this stage, you will identify your errors, correct them, and explain your correction. Explaining your correction is an important part of this step, as it helps both you and your teacher to understand your thinking behind the correction.

Correction

Choose a sample of your writing of about 500 words. This could be something you're working on at the moment, or a section of your ESA report. Identify errors related to one of the two most important writing issues you are focusing on in Phase 2. Do not directly correct the errors in the text. Instead, use the insert comment function in word **to add your correction**, and **add the reason for the correction**. Here is an example:

Unmanned Aerial Vehicles (UAVs) are defined as @type of aircraft without on-board pilots. 
Thus, UAVs are expected to fly in the air and move on the ground autonomously. As the UAVs 
the type 
Reason: specific aircraft 
Reply 
Resolve 

**Next**, in the same document, identify and correct errors related to **the other** most important writing issue you are focusing on in Phase 1, using the insert comment function.

Please watch [this video](https://web.microsoftstream.com/video/72896c0a-ea33-4978-a832-4b0ceab6e4ce?list=studio&referrer=https:%2F%2Fukc-onenote.officeapps.live.com%2F) , which shows you exactly what you need to do.

# Your text:

In [15], an alternative Sybil attack protection mechanism named Proof-of-Stake (PoS) was proposed. In case of PoS [18] [19], instead of requiring miners to do a tremendous amount of computational work, the probability of validating blocks relies on the stake nodes have. Namely, the more stakes a node has, the higher chance this node can create a block. Once the block is approved by the blockchain, a certain amount of transaction fees will be issued to this validator. A coin age based PoS-PoW consensus was first implemented to Peercoin in [18] to find validator. In Peercoin, the chance of being selected depends on the product of amount of coins and the time that a user has held these coins. Once a block is appended to the chain successfully, the age timer of this block will be set as zero to avoid reusing these coins. However, the drawback of this method is that it is very dangerous for the system. Because even one malicious node possesses sufficient coins for a long time, it can take control or destroy the network.

A variant of PoS named Delegated Proof-of-Stake (DPoS) is employed in BitShares [20] EOS.IO [21] and Cosmos [22], in which not all nodes are allowed to create blocks. Every node holding stakes votes for its trusted witnesses whose work is to generate and validate blocks as the representative of all nodes. Because there are fewer nodes participating in block issuing and validating work, the generation and confirmation speed of blocks are accelerated. To deal with Byzantine problem and reduce the time complexity of Byzantine Fault Tolerance (BFT) algorithm, Practical Byzantine Fault Tolerance (PBFT) was introduced [23], [24]. Roughly speaking, the deterministic consensus is reached based on a mutual transmission of information across nodes in BFT algorithm, while in PBFT algorithm, the complexity of sharing data among nodes is deduced from exponential to polynomial [25]. The algorithm works well if no more than 1/3 of the nodes are malicious. However, even if fewer than 1/3 of the nodes are malicious, even if the consensus mechanism is robust, these malicious nodes may cause other vulnerabilities. Blockchain can be divided into two types according to the available access to the ledger [26] [27]. The first one is permissionless Blockchain, which is open for anyone and based on PoW/PoS mentioned above. The second one is permissioned Blockchains, which is typically operated by one or more organizations and only preselected participants can be involved in validating transactions. The above mentioned PBFT consensus algorithm is commonly used in permissioned Blockchains [28].