Incentive Mechanism:

For this project, to facilitate more prosumers to take part in trading energies, an incentive mechanism based on the number of transactions is implemented. After registering with the main contract, a prosumer would either send a buy or a sell request. We will declare a threshold for the number of transactions a user has to make in order to be able to receive incentives, in this implementation the threshold is taken as 5 transactions. Once the prosumer takes part in 5 transactions, a reward point is awarded to the prosumer and the transaction count is reset to 0. In order to get another reward point the prosumer will have to again take part in trading 5 times by sending buy or sell requests. The prosumer can use these reward points to exchange with actual ethers stored in the smart contract. These reward ethers are accumulated by charging a certain amount of fees (10% in our case) when a trade occurs. However there might be a deficit in the amount of reward ethers if no transactions have been made, this is taken care of by adding a modifier. So whenever a prosumer tries to convert their reward points to ethers but there are less reward ethers accumulated in the smart contract then they will receive a prompt to try again later after more trades have taken place within the energy trading environment. This incentive mechanism will obviously encourage more prosumers to take part in energy trading since the incentives are received after completing a certain amount of transactions. We have to be careful of not keeping the threshold value for the transactions too high since this might discourage the prosumers to take part in energy trading. Also the reward points should have a fair and a reasonable exchange rate.

The concept of the incentive mechanism proposed in this project was designed based on the key requirements outlined in the paper "A Review of Incentive Mechanisms in Peer-to-Peer Systems" by Kan Zhang, Nick Antonopoulos, and Zaigham Mahmood from the School of Computing at the University of Derby in the United Kingdom. According to the paper, incentive mechanisms should aim to influence nodes' behaviors in a certain manner to increase the utility of the system. The proposed incentive mechanism in this project covers most of the requirements discussed in the paper, including decentralization, incentive, service diversity, adaptability, and light weight. The proposed mechanism utilizes the concept of a "fixed-contribution" incentive, where a fixed number of transactions must be completed before receiving incentives. However, it should be noted that the penalty requirement was not addressed in this incentive mechanism due to it being out of scope for the current coursework. Nevertheless, it may be considered for future developments to reduce the number of free riders in the system.

Overall, the incentive mechanism proposed in this project aligns with the key requirements outlined in the referenced paper and provides a means to encourage prosumers to participate in energy trading.

Improvements:

In order to enhance the proposed mechanism and improve the functionality of the peer-to-peer energy trading system, several features, new structural attributes, and modifiers were added. Two queues were implemented to keep track of buyer and seller requests. When a new prosumer submits a request to buy or sell energy, the system attempts to match them with a buyer or seller from the queue who can fulfill their energy needs. If no suitable match is found, the prosumer's request is added to the appropriate queue. Once a proper match has been made between a buyer and seller, the prosumer is removed from the queue (in the case of a seller, they are only removed from the queue once their surplus energy status becomes zero).Another addition to the system was the inclusion of a new attribute in the struct, called 'isRegistered.' This attribute is set to false by default, but becomes true once a prosumer is registered in the system. This attribute is necessary to keep track of which prosumers are registered and which are not.

Furthermore, an additional check was added to the checkFunds modifier. In addition to verifying that the prosumer has sufficient funds to cover the cost of the current energy request, this modifier now checks the prosumer's past pending buy requests and ensures that the balance available in their account is greater than or equal to the combination of the cost of the current energy request and the cost of any unfulfilled pending energy requests made in the past. These enhancements and additions aim to improve the efficiency and reliability of the energy trading system and ensure that prosumers are able to participate effectively and efficiently in the trading process.

Table Implementing the scenario:

The following table was created to demonstrate the implementation of the scenario outlined in the coursework document, using smart contracts that incorporate an incentive mechanism. As a result, a fee of 10% was deducted from buyers during energy trades. Initially, all user accounts were registered and had 20 ethers deposited into them. Buy and sell requests ranging from 1 to 6 and 1 to 8 units, respectively, were randomly generated to demonstrate the functionality of the smart contracts. These prosumer interactions were conducted by interacting with the main contract, which in turn sent appropriate requests to the P2P smart contract.

Serial number	Account ID	Before Energy Trading		After Energy Trading	
		Energy Status	Balance (ethers)	Energy Status	Balance (ethers)
1	0x47d7993370DC6033d53aFE6206Cf3Ef00518F488	0	20	0	20
2	0x4e865E48D85CeEdF56B0b915feB5a946773C54BD	-2	20	0	17.8
3	0xdfc2e0BCF64E82c0751Fea4e35E7626bBDd86f55	3	20	1	22
4	0x2b2736DC8Ae736639554a7227DEbA4B9DD76288E	7	20	0	27
5	0x59312FD2b0350DEB9F23eC13A3c4bf566881FFb2	-4	20	0	15.6
6	0x00Cba4Cfe4237F7F379611f67f70a7Bb89946535	6	20	0	26
7	0x830ff95d0b426aD7729897771303De1897214f44	-6	20	0	13.4
8	0xFEf436849d795d4a5B51e48F93016c4cD1996152	2	20	0	22
9	0x069CD5d4C5f2F450FF7BBAC7991f70418C5cCCac	-4	20	0	15.6
10	0xfaeC0b6bA5fA3313835704ab1145D4E2F0f7f0fF	5	20	1	24
11	0xb0F3aC73B60C58F91B104fD2b6e382687597E98D	-6	20	0	13.4
12	0x29642958af271966Ff4FBE73267bD28DF4d5f638	-3	20	0	16.7
13	0x89291C69CEb3D4B27f4E56Ce59afeCBEf27c9c0E	8	20	0	28
14	0xCEDEB337d18D971aD0F307Cc6650aB1F591CBBB7	-2	20	0	17.8
15	0x07c717dC68fF8e3b9a678266e005Fb96bF8d7482	7	20	4	23
16	0xd5e9f7ED9572305932a43e4539168535411F0b75	-2	20	0	17.8
17	0x90e8604d336bc2b2f24390EB7B6b1F833ba224DF	1	20	1	20
18	0xCb4a506D6e0BA8568c7B24b5eC348318d8590FCA	6	20	6	20
19	0xe09005fE9f84d9ED64E3c3A245B028074570C1a6	1	20	1	20
20	0xC0399866347178be85573f4AF7052ADE3C70adD6	-3	20	0	16.7