The goal of this project is to produce a proof of concept implementation of a Solidity proxy wallet. Unlike most authentication in Ethereum contracts, the address of the transaction sender is arbitrary but it includes a signed message that can be authenticated as being from either the account owner or a dApp for whom the account owner has created a session with permissions.

See
https://ethereum.stackexchange.com/questions/1777/workflow-on-signing-a-string-with-private-key-followed-by-signature-verificatio
https://ethereum.stackexchange.com/questions/15364/ecrecover-from-geth-and-web3-eth-sign
https://github.com/shine2lay/SmartContracts/blob/master/contracts/ERC1077/ExecuteSignedMessage.sol for more details on the general concept of signed messages.

Requirements:

- Specify a hard-coded list of administrator addresses - this is only required for the account initialization step
- A username (specified during initialization)
- A public key (specified during initialization)
- Initialization method (admin only) - specify username and public key used to sign messages authenticated by the account owner.

- dApp Pairing method - includes authenticated message from account owner (signed with their private key) indicating that a dApp with the private session key corresponding to a provided pubic session key has access to up to X ETH funds for Y time.
  - Verify that message is signed by owner of the saved account public key
  - Verify that account has X funds
  - Save public session key and permission info

- dApp Action method - includes authenticated message from dApp (corresponds to saved public session key) requesting a transaction to be sent to address A of ETH amount B.
  - Verify that message is signed by owner of the saved session public key
  - Verify that account has B funds
  - Send transaction of amount B to address A as requested

- Should include passing unit tests

Notes:

- If possible, track the amount of gas costs incurred from all legitimate transactions and even better if the gas costs can be tracked for the specific addresses that incurred those costs. The gasleft() function can do this, although it's not possible to make 100% accurate.
- There is some administrative functionality, as well as handling for non-ETH tokens that is not being included in this version.

- We can use send method to deposit funds to the contract unless there is a reason why that isn’t possible.