**Implementing Vrf Fulfil**

To work with the Chainlink VRF (Verifiable Random Function) in Solidity, we need to inherit functions from an **abstract contract** called [VRFConsumerBaseV2Plus](https://github.com/smartcontractkit/chainlink-brownie-contracts/blob/12393bd475bd60c222ff12e75c0f68effe1bbaaf/contracts/src/v0.8/vrf/dev/VRFConsumerBaseV2Plus.sol). Abstract contracts can contain both defined and undefined functions, such as:

function fulfillRandomWords(uint256 requestId, uint256[] calldata randomWords) internal virtual;

* When we call the Raffle::performUpkeep function, we send a request for a **random number** to the VRF coordinator, using the s\_vrfCoordinator variable inherited from VRFConsumerBaseV2Plus. This request involves passing a VRFV2PlusClient.RandomWordsRequest struct to the requestRandomWords method, which generates a **request ID**.
* After a certain number of block confirmations, the Chainlink Node will generate a random number and call the VRFConsumerBaseV2Plus::rawFulfillRandomWords function. This function validates the caller address and then invokes the fulfillRandomWords function in our Raffle contract.

🗒️ **NOTE**  
Since VRFConsumerBaseV2Plus::fulfillRandomWords is marked as virtual, we need to **override** it in its child contract. This requires defining the actions to take when the random number is returned, such as selecting a winner and distributing the prize.

Here’s how you override the fulfillRandomWords function:

function fulfillRandomWords(uint256, /\* requestId \*/ uint256[] calldata randomWords) internal override {

//pick a winner here, send him the reward and reset the raffle

}