**Creating integration tests**

**Integration testing**

We have finished the unit tests. Amazing!

But that doesn't mean we are finished with testing. Unit tests are just the first step. If you remember what we learned some lessons ago there are 4 types of tests:

1. Unit tests - Basic tests that check the functionality
2. Integration tests - We test our deployment scripts and other components of our contracts
3. Forked tests - Pseudo staging
4. Staging tests - We run tests on a mainnet/testnet

We won't cover staging tests for now, but these are very important. People often deploy their entire protocols on testnets or cheap mainnets like Polygon to properly check how their protocols behave in production environments.

Testnets are not always fun to interact with (just remember the gigantic refactoring we had to do in the previous lesson), but they are the closest thing to the mainnet we have. To check the recommended testnet please access the [Foundry Full Course repo](https://github.com/Cyfrin/foundry-full-course-cu).

Back to integration testing, we already started testing the deployment script given that in the RaffleTest.t.sol::setUp function we used the script to deploy our Raffle contract. A better approach would have been to create an InteractionsTest.t.sol and test the deploy scripts first, then use them in the RaffleTest.t.sol.

We will not do staging tests because writing scripts that do a lot of waiting is a bit tricky. Foundry is phenomenal in testing things that are 100% on chain, but given that we are using Chainlink VRF and some things that happen off-chain, we will struggle to do these staging tests.

If you want to try your hand in writing some tests go ahead and write the InteractionsTest.t.sol tests. These should cover at least the DeployRaffle.s.sol. If you want to score extra points and be a true champion, then cover all the scripts.