**Where to go from here**

**ENTER**

**-In this course we've given you a overview of many of the aspects of PHP unit testing. Remember, it's for you. I**

**t's a lot of work and you have believe in what you're doing in order to create proper unit testing.**

**We've shown a lot of ways that, you can write bad tests and get that nice green light but the fact is that this is for you and if you believe that it can reduce the maintenance workload later on down the road you'll be motivated to write good tests. Code coverage is just a tool to help you improve your code.**

**ENTER**

**A 100 percent means nothing, unless they're well written tests that anticipate the unordinary.**

**ENTER**

**The best unit tests are the result of making the right decisions. If something is not testable it needs to be refactored.**

**ENTER**

**Also, don't get caught up with testing the database too much that's a battle for a different day. PHP unit testing is about testing the code. It's efficiency, it's maintainability, and that it gives us what we expect.**

**ENTER**

**Also, be prepared to work with other frameworks. Selenium is software that does automated web browser testing and Travis does continuous integration.**

**PHP unit is a tool and it's been around for a long time and it works well, but it works very well when utilized with the other tools that help you give a more complete test driven development environment. If you like phpunit, perhaps you will choose phpspec for you behavior driven development needs**

**Also, when you work, replicate the live environment for your project. In this course we used Mamp which is not the industry standard.**

**Generally you'd use a virtual machine with Vagrant, or something like that.**

**ENTER**

**Also, use source control, git.If you don't know what that is, you need to learn it just so that, you don't waste your time writing tests that get overwritten.**

**ENTER**

**Make it an edict of yours that every time a function is added, you add a test for that function or you write the test before the function is added but those, from now on, go in unison. Again, make sure the existing code is testable and lastly, have tests for both success and failure cases.**

**We want a lot of our tests to fail we want to anticipate a failure and that's how we can write better unit tests.**