The statement “Don’t leave security to the end” is one that many coders have certainly heard of in the past. I’m sure that many of these coders see this statement almost as if it nagging them to worry about security, but it is much much more than that. A coder should always have “don’t leave security to the end” in the back of their mind always. Three of the main coding principles about security are “Architect and Design for Security Policies”, “Use Effective Quality Assurance Techniques”, and “Adopt a Secure Coding Standard.” If you do leave security to the end, you are not following these three principles.

Firstly, your job as a coder is to ensure that your code is readable by yourself and the machine that will read it. In both cases, none of the things you write should be ambiguous. What is written should only be able to be read one way and one way only. This is the idea behind adopting a secure coding standard. By having a secure coding standard to begin with, you will always have some degree of security, but if your code is convoluted and/or ambiguous, then you lose that layer of security.

That, however, is only the first step that you can take to prevent threats. Like all other scenarios when it comes to coding, you should be coding with “Defense in Depth” in mind. The second layer of security you have to prevent threats is to design your code with security policies in mind. At this point, not only do you have solid, unambiguous code, but you would also now be able to write code with security in mind, which makes it even stronger. This is the second step you can take to prevent threats when writing code.

The last step (not truly the last step, but the last one I will mention here) is using effective quality assurance techniques. By ensuring that your code runs properly and is quality code, you add another layer of security on top of your other layers. You can run unit tests to ensure that your code is not going to result in some unexpected error or other flaw. You can do this as you

code instead of when you finish so that you can find issues as soon as they arise instead of later when it's more difficult and more costly to find, address, and fix the issues.

All of these three things together make a grand plan that allows you to intrinsically build security into your design from the ground up. By employing all these different principles and methods, you are utilizing “Defense in Depth” in your code already. This allows you to find, address, and circumvent issues as soon as they come up or even before they come up, depending on how you plan your code ahead.