Friday 17 January 2025

As I had developed more and more web applications, I found it was time-consumed and extremely to locate mistakes when these systems were becoming larger. To tackle these issues, I adopted TDD,(acrnym) which was an effective way to write code, when I was writing code. Surprisingly, all of my colleagues were neither eager to understand nor wanted no learn this approach which had been invented two decades ago and had been accepted by many software engineers around world. They prefered to do work in a traditional way which had been outdated or less effective. Let me draw an analogy between a software engineer and an athelte. Nowadays, sportsmen run faster and jump higher than those decades ago because they are trained with constantly developing scientific methods. In the fast-paced programming world, it is exactly the same.

The obstacles we have encountered were faced by some exceptional engineers and they had made effort to tackle these problems and shared the patterns and solutions on the Internet for free or on books which cost a little money. More and more developers have learnt from them and managed to write robust, extendable, maintainable programme with less mistakes. Though I don’t expertise in TDD or other kind of approaches, I can write better code than before after learning them. Unfornately, my peers don’t want to learn with me. I am not strongly critical of them as I was the same about a year ago. Slowly and gradually, when I know more and more advanced skills I can write sophiscated applications for customers.

***Improved by My Tutor***

As I had developed more and more web applications, I found out that it was extremely time-consuming to locate mistakes when these systems were becoming larger. To tackle this issues, I adopted TDD, which was an effective way to write codes.

(next sentence = please add commas)

.....had made efforts to tackle......  
....which costed a little money

Though I'm not an expert in TDD or other kind of approaches, I can write better codes than before, after learning them.

Slowly and gradually, when I knew more and more advanced skills, I can now write sophisticated applications for customers.