The principal component of this app would probably be the normalized spline position (npp),

which returns a float value from [0,1], with 12 digit precision, that represents how far ahead you're on the AI Spline (the racing line that AI drivers would drive). Generally, a 0 would mean the starting line

and 1 a finish line, but not necessarily, a starting line could be set at 0.2 and the finish line could be set at 0.9 for example, it depends very much on how the developer for that map decided to do

the AI Spline.

When you're setting up the sectors, you're pressing buttons on the track where you would like to

place the end of a sector. What the button does at that time of it being pressed, is that

it stores the value of the npp at that time, and lets call this "the checkpoint of sector i",

where i goes from 1 to n, n being the total number of sectors.

How recording of time works. as you drive on the track after configuring your sectors, the game

compares your npp to the value of the checkpoint for that respective sector you are in. The particular condition is: if npp >= sector_checkpoint. It is needed to use >= instead of a direct equality, ==, because the game doesn't report for every single value, so if you check for exactly a value, there is a high chance it will be missed because the npp value will be higher.

think about it like this:

a car driving with 20 km/h

```
.     .        .

x---------------x-----------x
```

a car driving with 200km/h

```
.                 .            .

x---------------x-----------x
```

Let’s say that x represent some checkpoints and the dots (.) display where you're getting information from the game engine.

As you can see, if you drive faster, the car travels a lot more in a single game tick, due to this, you can be before a checkpoint in a game tick, and be after it in the next game tick, but never actually be right on top of it in any game tick.

Therefore a >= comparison becomes necessary.

This comparison is also the reason why you must first go pits after setting up a track configuration.

Let’s say you didn't need to go to pits. You just finished your configuration and you are at a npp value of 0.9. At the moment you configure your last sector, the app would start to check npp >= sector_checkpoint, which would be true for all sectors, since you placed them behind you, resulting in all your times being recorded as 0, the game thinks you already passed all of them in, more or less an instant.

How the last sector as finish works:

Since the highest npp value possible is 1, if you create a sector checkpoint with a value higher than 1, it would never be reached. But, you can make an additional check after a player has entered a new lap, that checks if all the available sectors have been passed, if for example, you put the last sector checkpoint at a value of 2, this sector would never be passed normally, so therefore, the only time when it’s gonna be passed, it’s when the player enters a new lap and the app sees that not all sectors have been passed. At that time the app calculates the time for the last sector, resulting in this sector being completely equal to the finish line.

Another hurdle was the fact that, originally i developed this app with the idea that a player was gonna drive it only in a practice-pits session. Because, when you reset the session, it would drop you into pits. But every other session would place you in different locations, such as: before the finish line, way before the finish line etc... Making my section that checks for a session restart to not work. Anyhow, I added some extra functionality for this to work. It now stores the 3d position and npp value of the player when they first enter the game, to have some baseline values for a starting position. Which then the app compares to the current 3d position and progress of the car when resetting, to make the sure the car reseted completely when resetting a session, thus making it work for any type of session, regardless of if the player starts in pits or not.

Such issues with the app thinking you already passed the sector were solved by various means, but i added just in case a safety net that limits the maximum difference between the sector checkpoint and the npp value to 0.1, this is meant only for cases where maybe the engine decides to do something funny and break, like falling through the map, or some other random alike stuff.

Another thought was: "How do I implement support for touge/hill climb maps?" such as trento bondone and the likes. The problem here being that after you cross the line in such a map, your npp value would sit at 1, or a very high value. This resulting in the next game tick thinking you passed all the sectors before you, just like how I explained above with the needed >= instead of the == comparison. The solution to this is quite simple but, there are some unpractical drawbacks. I simply added an extra check for when entering a new lap, that the npp value is lower than 0.3, this would wait for the player to go to pits in a touge/hill climb map, resulting in not overwritten timers. The only "problem" here technically would be if some map had the pits at a value of 0.3, but it’s pretty much unthinkable, so it’s a good trade-off.