**Async Rust Summary**

Recently the stabilization of rust async feature (async/.await) the asynchronous coding in rust is easy to do within a more efficient way and use fewer resources. Before going deep about the Async rust, first of all, let’s take an introduction about the asynchronous programming. Asynchronous programming basically is a technique in computer science to run the operations concurrently or simultaneously in the application. Here the question arises why we do asynchronous approach when we have a simple flow of coding? Although in every situation we don’t go through the asynchronous programming the developers know when they use or not. Like in the internet of things aspect we have different sensors at different locations and we want continuously monitoring of data so in this case, the asynchronous approach will definitely give appropriate results instead of normal flow because we want concurrent processing to get the continuous monitoring.

In normal threaded applications, we have sperate threads for a particular operation, the threads are designed to run different task multiple times so it is difficult to share data between the threads and jump from one thread to other thread and also if some thread just sits in and do nothing only uses system resources these all overhead were facing in threaded applications. The asynchronous rust overcomes this overhead by running multiple operations on the same thread also asynchronous rust is more faster and use fewer resources.

To make function asynchronous in rust we have to follow some steps:

1- access to async\_std trait this will pull all dependencies at compile time.

2- write async keyword before the function otherwise the rust compiler doesn’t consider the function in the async environment.

3- need an executor on which future (describe letter on) will run.