ASYNC / AWAIT

```
CONSOLE.LOG("ASYNC AWAIT, STARTING"); //
```

Printed 1st

// This async function will return a promise.
ASYNC FUNCTION HARRY() {
CONSOLE.LOG("INSIDE HARRY FUNCTION"); // Printed 3RD
CONST RESPONSE = AWAIT FETCH("HTTPS ---- "); // await means, it will be resolved asynchronously till then the program can leave the function and do the further work if any.
CONSOLE.LOG("BEFORE RESPONSE"); // Printed 7th
CONST USERS = AWAIT RESPONSE.JSON(); // Now, there are no further works to do, so, it will simply wait for the await to convert response in JSON, and then execute it.
CONSOLE.LOG("USERS RESOLVED"); // Printed 8th
RETURN USERS; // So, function is finally resolved so, it will run the line a.THEN(——) & and print the data.
}

CONSOLE.LOG("BEFORE CALLING HARRY"); // Printed 2ND
LET A = HARR();
CONSOLE.LOG("AFTER CALLING HARRY"); // Printed 4th
CONSOLE.LOG(a); // Printed 5th
a.THEN(DATA => CONSOLE.LOG(DATA)) // Printed 9th

Page No.
Date

CONSOLE.LOG("LAST LINE OF THIS JS FILE!"); // Printed 6th

→ Benefits of using an async function:
→ Debugging using promises sometimes is very hard because the debugger will not step over the asynchronous code. But the Async/Await makes this very easy because it is just like synchronous code to the compiler.
→ As we can see from the above example, the code looks very simple compared to the code using plain promises, with chaining and callback functions.
→ Error handling is simpler in async functions.