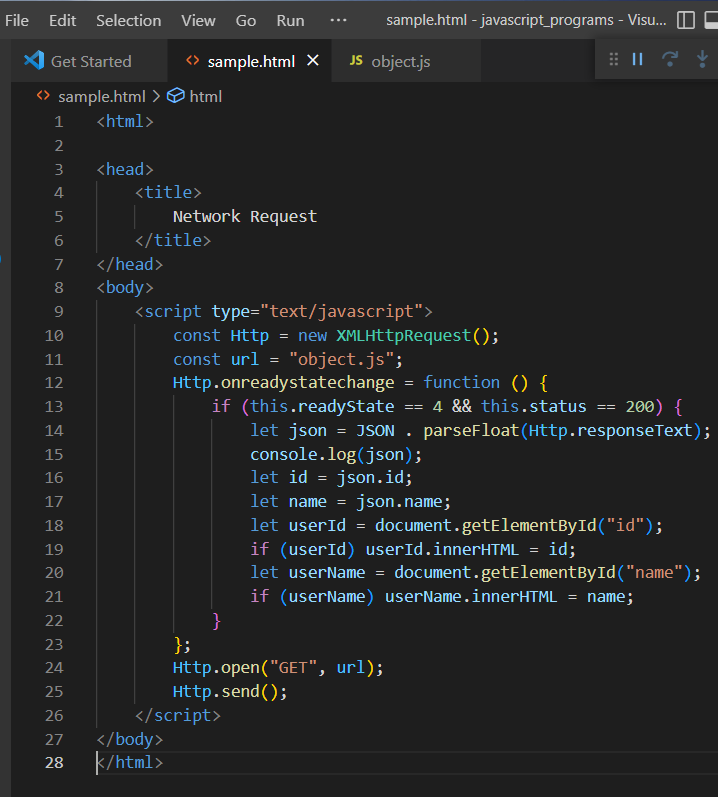
**Chapter-20 Network request**

Applications dealing with back-end communicates via HTTP requests. We can call the url with the GET or POST method.



whenHTTP request is executed using open and send methods,the onreadystatechange event will be fired 4 times. When the value is 4 and the status is 200, the connection is established successfully.

**Callback Hell**:

Callbacks are functions that return after an event is executed. These callbacks were extensively used as a tool to execute asynchronous calls. 

A series of calls written this way can cause the problem of dependency.

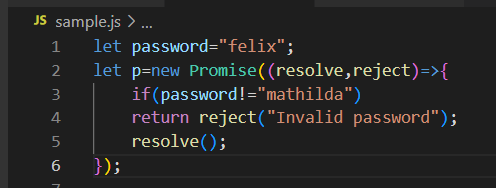
This is called **callback hell.**

**Promises**:

Using just callbacks in some situations where one event depends on results returned from another event might result in a complicated code.

In this case Promises are used. The promises take two parameter.

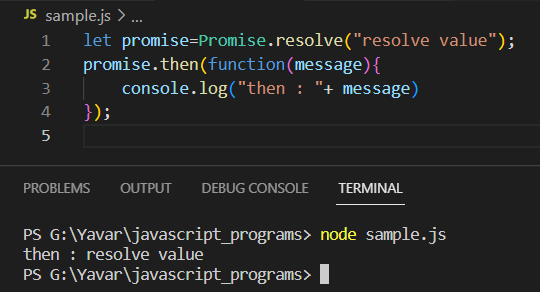
* Resolve
* Reject



A Promise object provides a pattern for checking whether an action fails or succeeds.

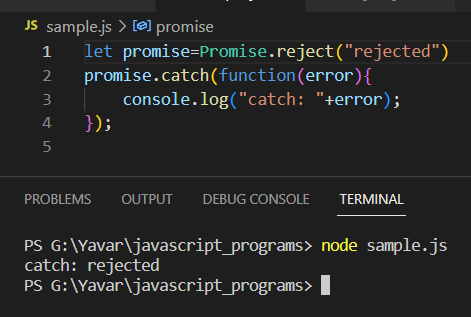
.**then:**

The resolve method indicates that the promise has been successfully fulfilled and contains a return value. This is carried out by **then** method.



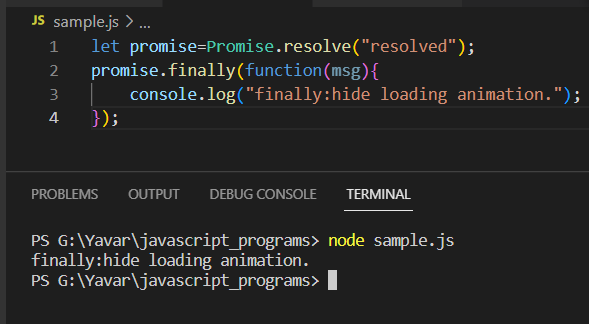
**.catch:**

The catch method responds only to reject method.



**.finally:**

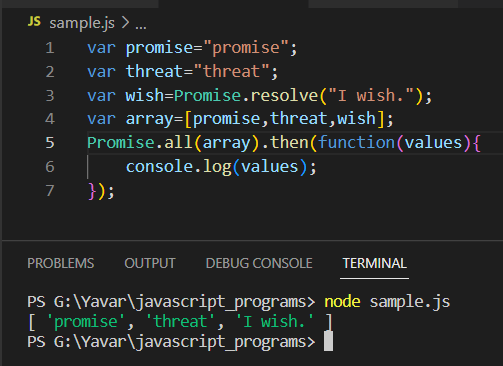
The finally method is executed regardless of whether event succeeded with resolve method or failed with reject method.



If a condition isn’t met and the promise is rejected then catch method will not be called instead finally method is called.

Promise.all:

We can resolve multiple promises at once using a single call to the Promise.all method:



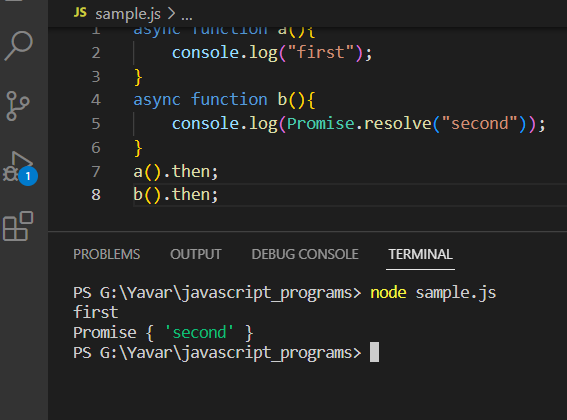
Generally when a promise is used all the three methods(.then,.catch and .finally) are used together.

**async/await:**

Promise-based code suffers similar issues faced by callbacks. This can be resolved by **async.**

In a normal function, the function call will be executed in the order in which they are called. But in case of async calls is little different.

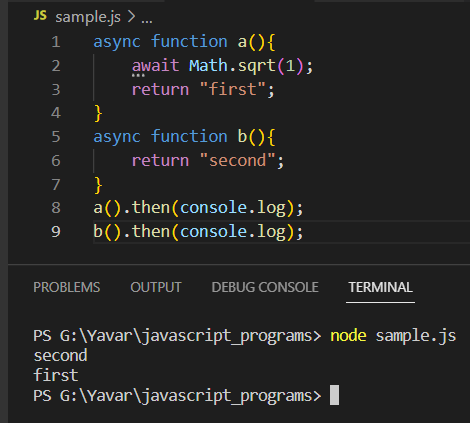
The async functions will return a promise object.



Await:

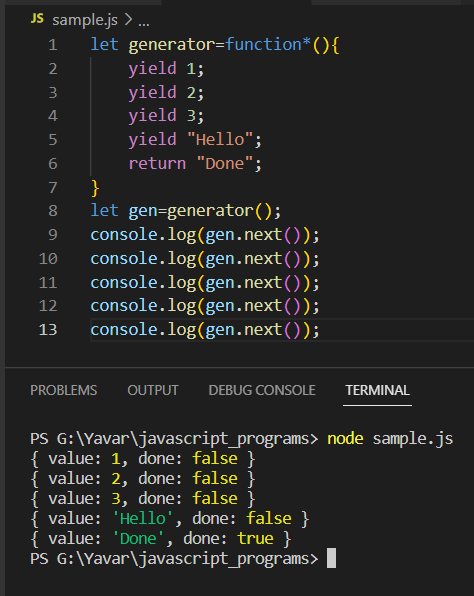
The async and await keywords are used in combination with each other.

Usage of await outside of async function will generate an error.



Generators:

They are similar to async. It is defined by adding a star(\*) character to the function definition. It comes with the yield keyword.



The generators dont require a return value but if there is one it will be treated as the final value.