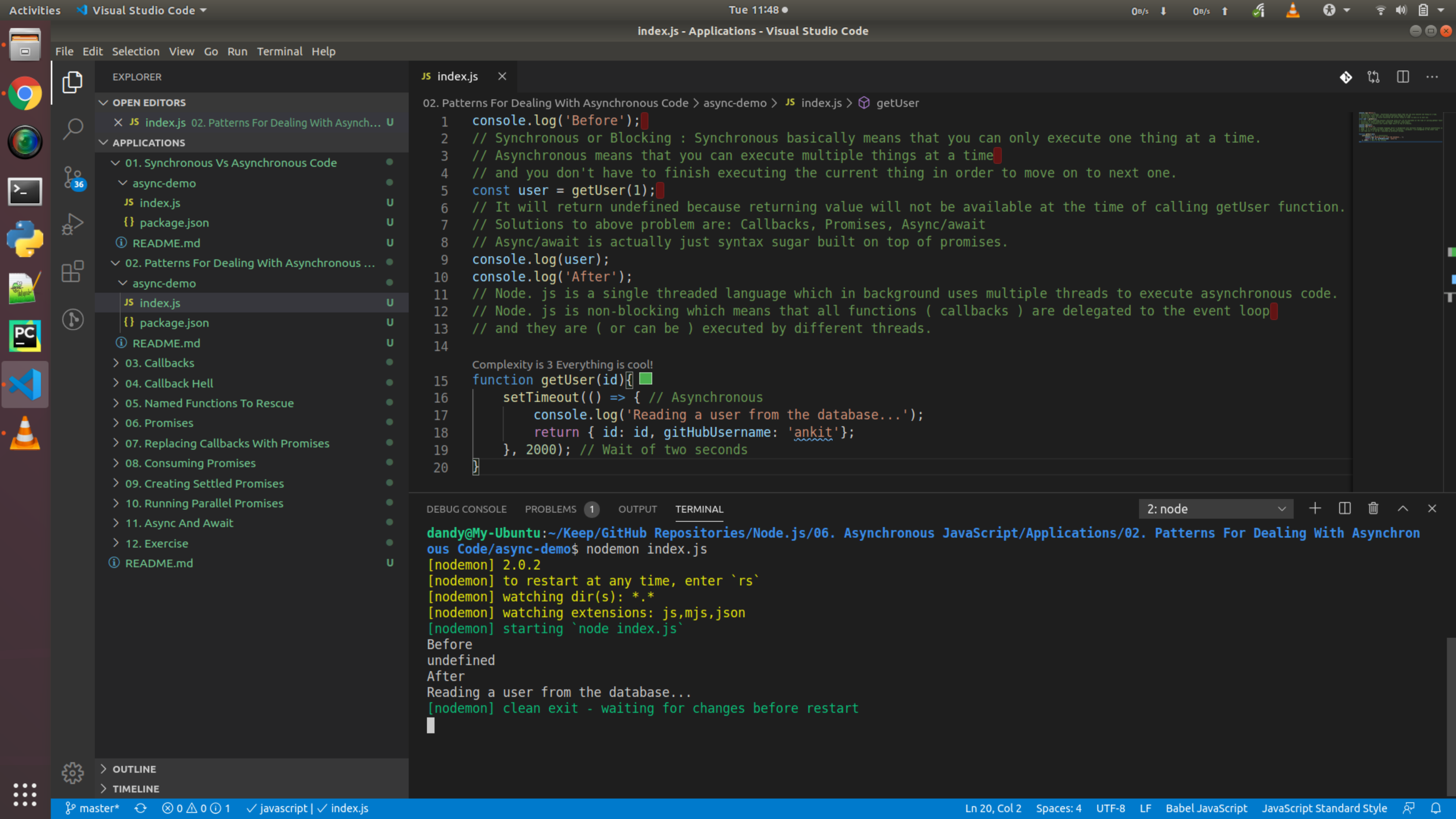
Async Patterns

Mosh Hamedani
programmingwithmosh.com



```
console.log('Before');
// Synchronous or Blocking : Synchronous basically means that you can only execute one thing at a time.
// Asynchronous means that you can execute multiple things at a time
// and you don't have to finish executing the current thing in order to move on to next one.
const user = getUser(1);
// It will return undefined because returning value will not be available at the time of calling getUser function.
// Solutions to above problem are: Callbacks, Promises, Async/await
// Async/await is actually just syntax sugar built on top of promises.
console.log(user);
console.log('After');
// Node. js is a single threaded language which in background uses multiple threads to execute asynchronous code.
// Node. js is non-blocking which means that all functions ( callbacks ) are delegated to the event loop
// and they are ( or can be ) executed by different threads.

Complexity is 3 Everything is cool!
function getUser(id){
    setTimeout(() => { // Asynchronous
        console.log('Reading a user from the database...');
        return { id: id, gitHubUsername: 'ankit'};
    }, 2000); // Wait of two seconds
}
```

```
dandy@My-Ubuntu:~/Keep/GitHub Repositories/Node.js/06. Asynchronous JavaScript/Applications/02. Patterns For Dealing With Asynchron
ous Code/async-demo$ nodemon index.js
[nodemon] 2.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Before
undefined
After
Reading a user from the database...
[nodemon] clean exit - waiting for changes before restart
```