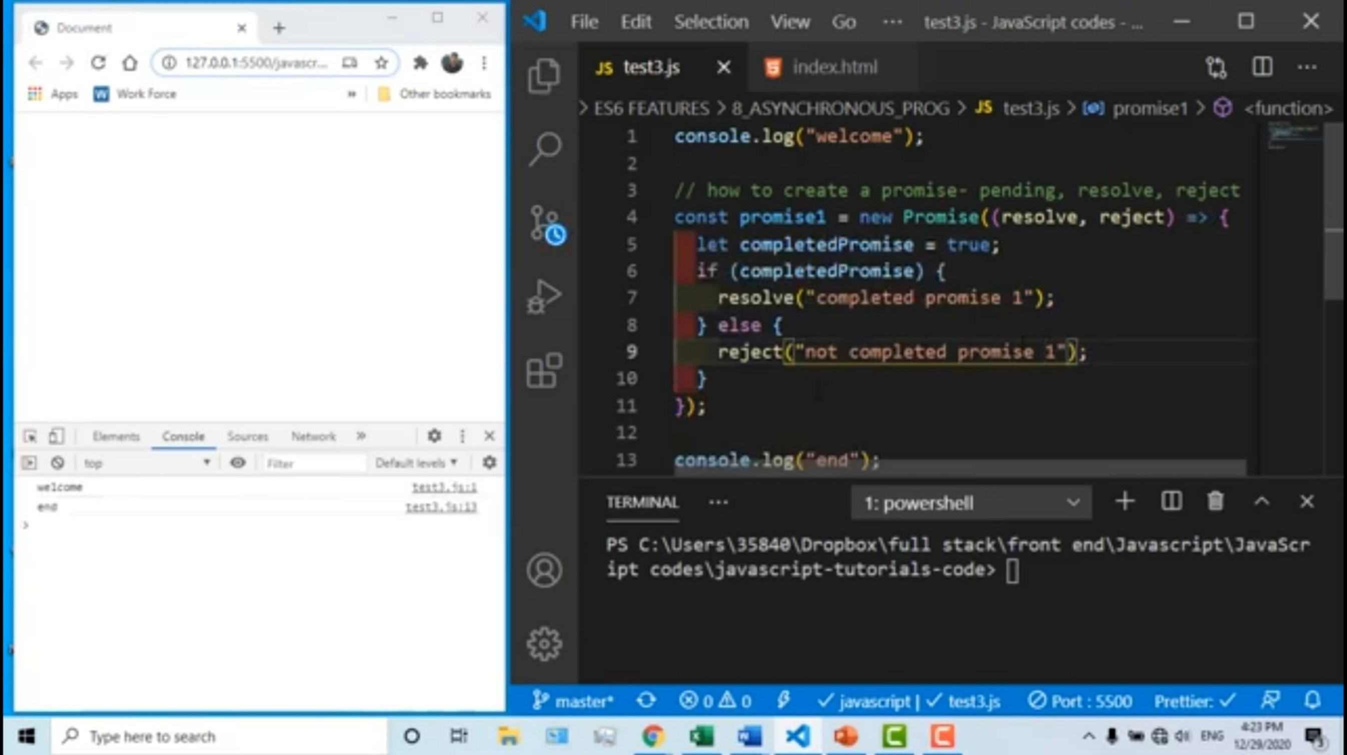
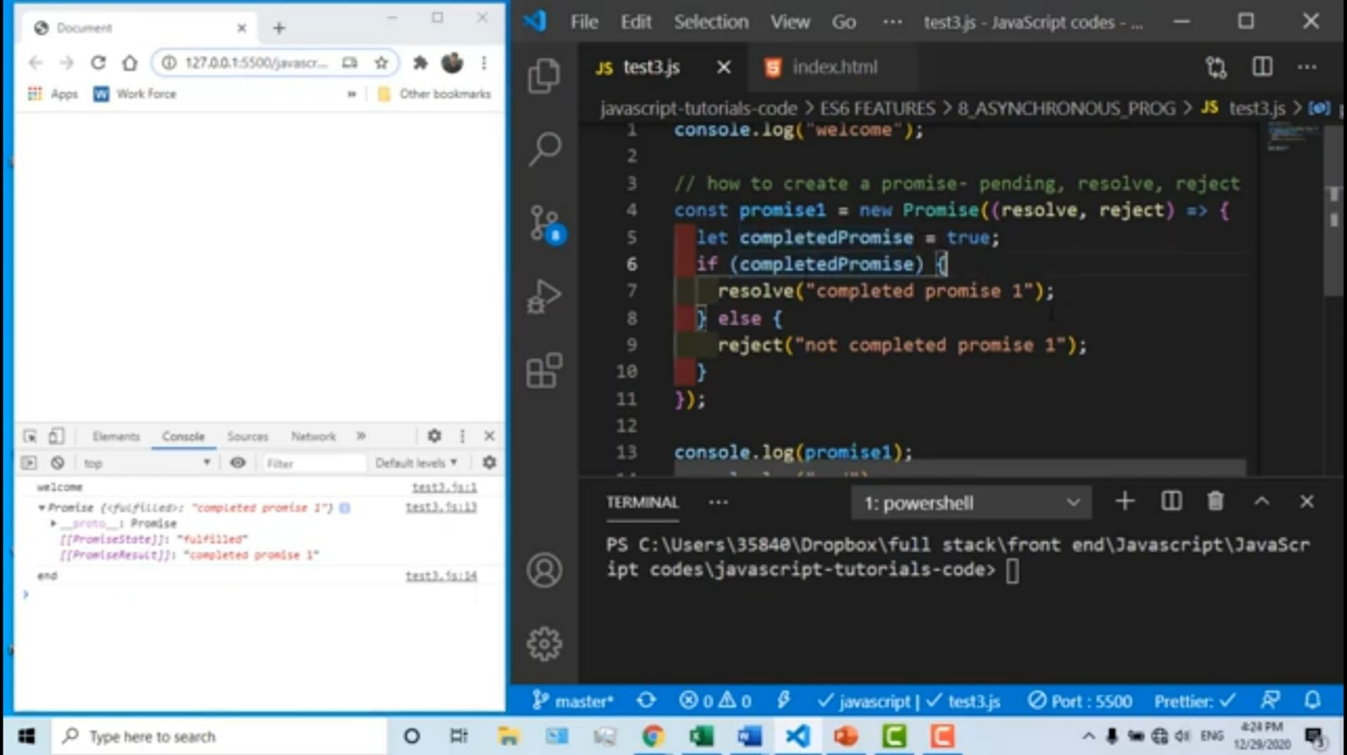
- How to create a promise
- How to use a promise
- How to run multiple promise - all()
- race()
- Promise chaining



```
welcome                                  test3.js:1
end                                      test3.js:13
>
```

```javascript
console.log("welcome");

// how to create a promise- pending, resolve, reject
const promise1 = new Promise((resolve, reject) => {
  let completedPromise = true;
  if (completedPromise) {
    resolve("completed promise 1");
  } else {
    reject("not completed promise 1");
  }
});

console.log("end");
```

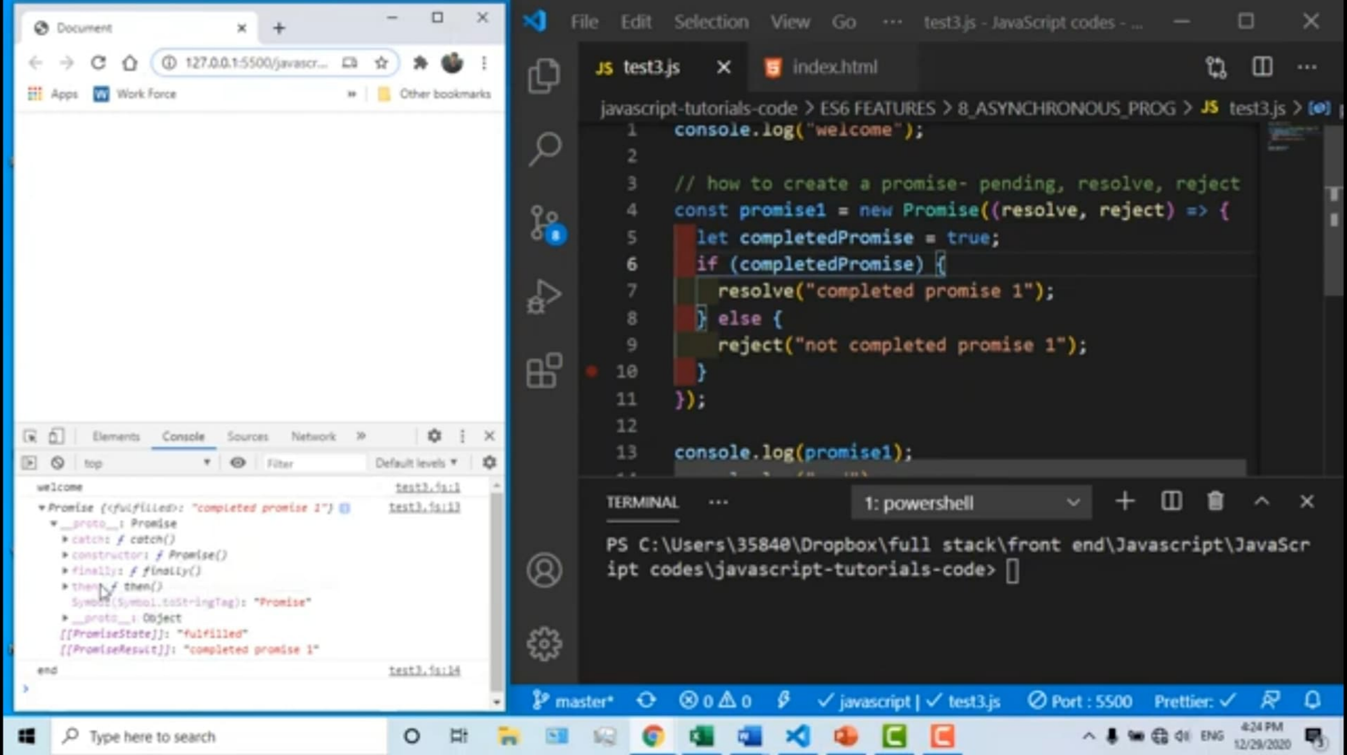
```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code>
```



```
console.log("welcome");

// how to create a promise- pending, resolve, reject
const promise1 = new Promise((resolve, reject) => {
  let completedPromise = true;
  if (completedPromise) {
    resolve("completed promise 1");
  } else {
    reject("not completed promise 1");
  }
});

console.log(promise1);
```

```
welcome                                                        test3.js:1
▾Promise {<fulfilled>: "completed promise 1"}                  test3.js:13
  ▸ __proto__: Promise
   [[PromiseState]]: "fulfilled"
   [[PromiseResult]]: "completed promise 1"
end                                                            test3.js:14
>
```

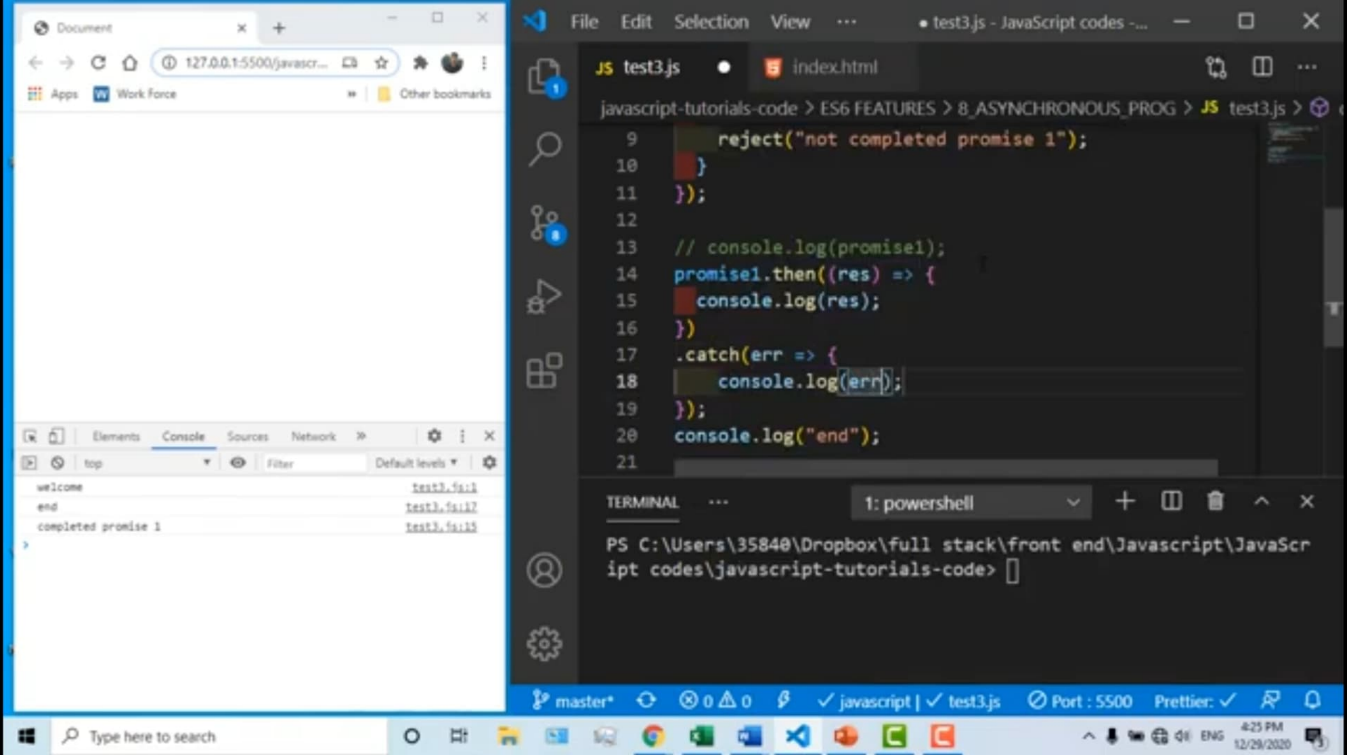
```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScript codes\javascript-tutorials-code>
```



```
console.log("welcome");

// how to create a promise- pending, resolve, reject
const promise1 = new Promise((resolve, reject) => {
  let completedPromise = true;
  if (completedPromise) {
    resolve("completed promise 1");
  } else {
    reject("not completed promise 1");
  }
});

console.log(promise1);
```

```
welcome                                                    test3.js:1
▼Promise {<fulfilled>: "completed promise 1"}              test3.js:13
  ▼__proto__: Promise
    ▶catch: ƒ catch()
    ▶constructor: ƒ Promise()
    ▶finally: ƒ finally()
    ▶then: ƒ then()
     Symbol(Symbol.toStringTag): "Promise"
    ▶__proto__: Object
   [[PromiseState]]: "fulfilled"
   [[PromiseResult]]: "completed promise 1"
end                                                        test3.js:14
>
```

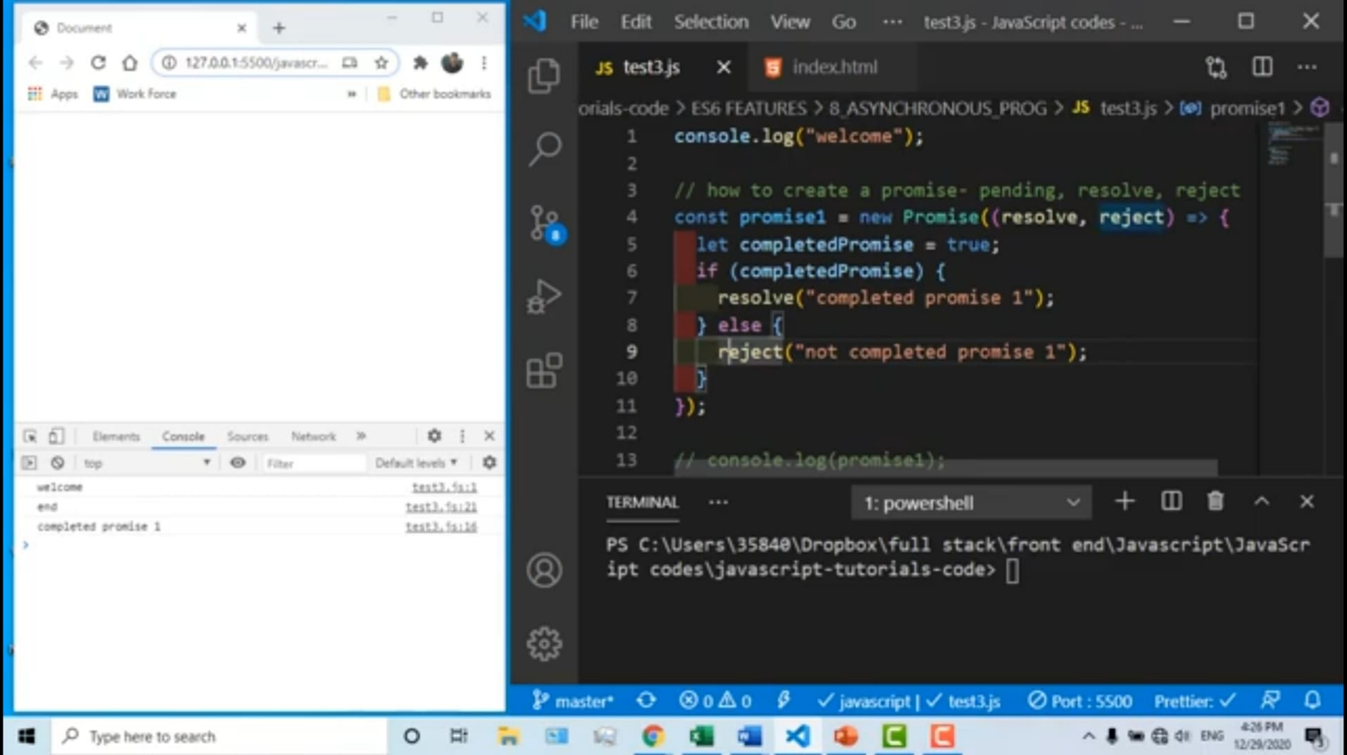
```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code> 
```



```
        reject("not completed promise 1");
    }
});

// console.log(promise1);
promise1.then((res) => {
  console.log(res);
})
.catch(err => {
    console.log(err);
});
console.log("end");
```

```
welcome                    test3.js:1
end                        test3.js:17
completed promise 1        test3.js:15
```

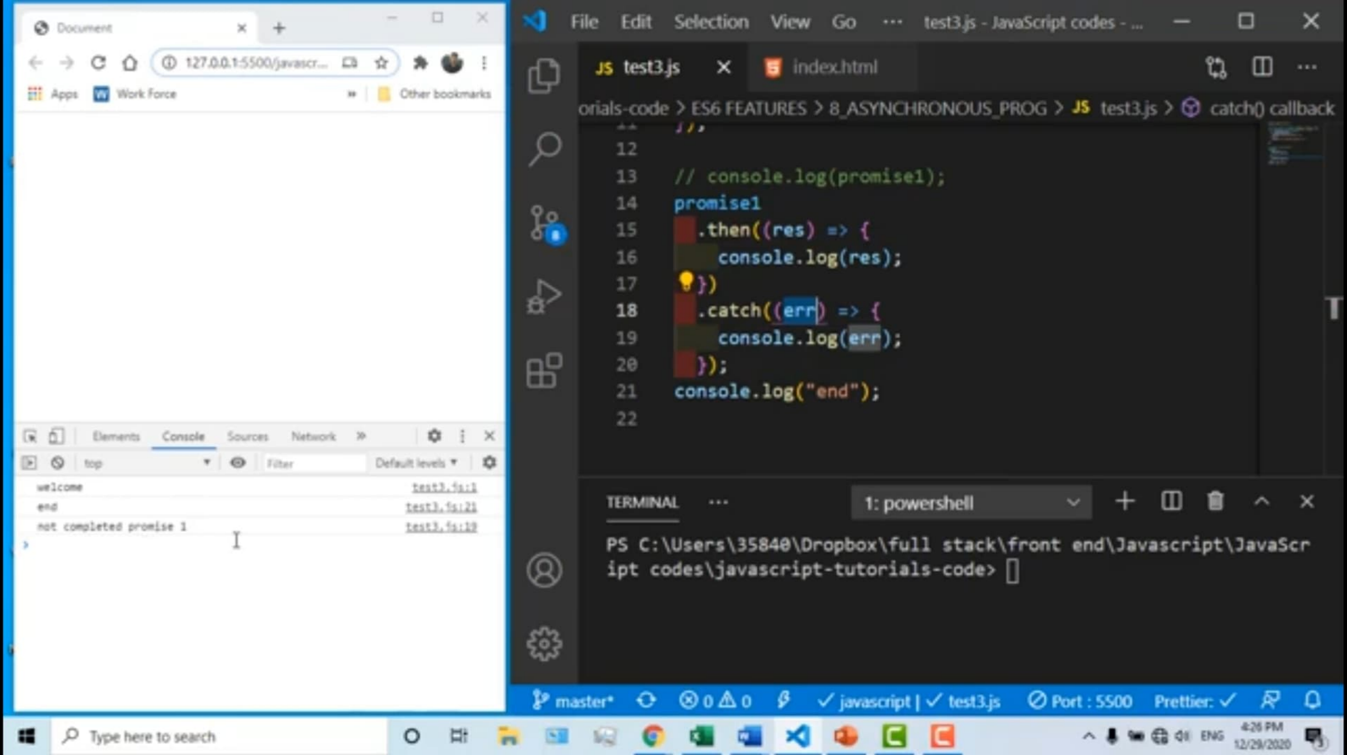
```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code> 
```



```
console.log("welcome");

// how to create a promise- pending, resolve, reject
const promise1 = new Promise((resolve, reject) => {
  let completedPromise = true;
  if (completedPromise) {
    resolve("completed promise 1");
  } else {
    reject("not completed promise 1");
  }
});

// console.log(promise1);
```

```
welcome                                    test3.js:1
end                                        test3.js:21
completed promise 1                        test3.js:16
```

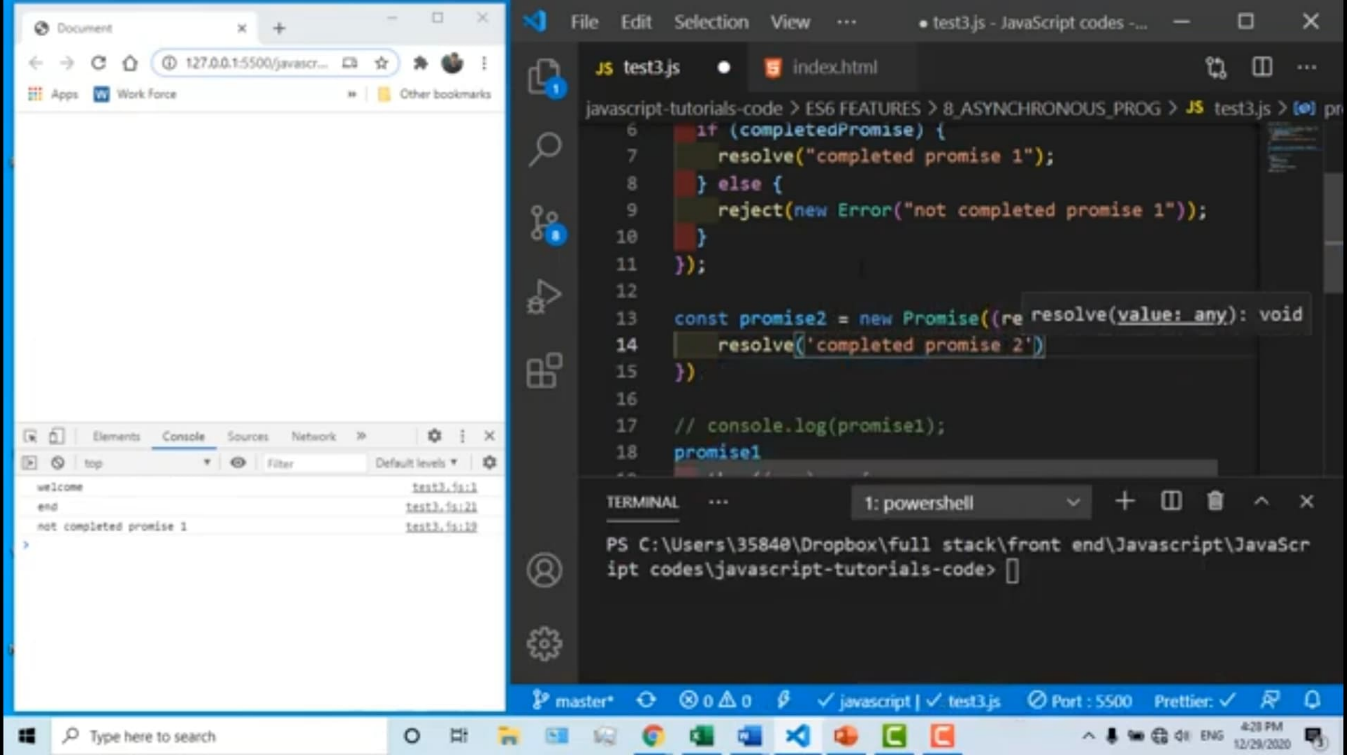
```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code>
```



```
welcome                                  test3.js:1
end                                      test3.js:21
not completed promise 1                  test3.js:19
```

```
});

// console.log(promise1);
promise1
  .then((res) => {
    console.log(res);
  })
  .catch((err) => {
    console.log(err);
  });
console.log("end");
```

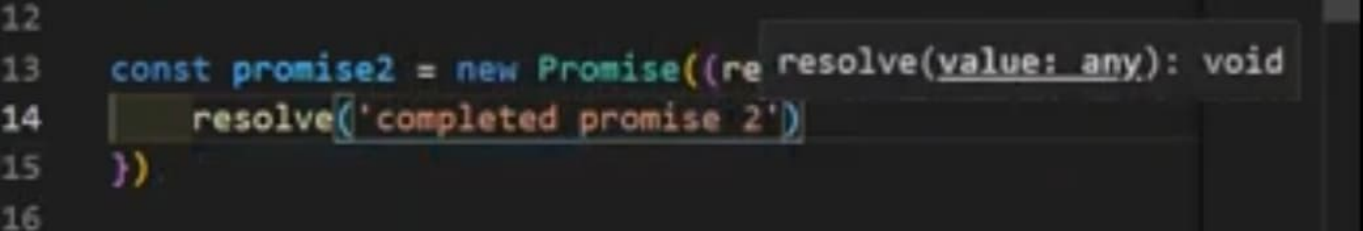
```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code>
```



```
  if (completedPromise) {
    resolve("completed promise 1");
  } else {
    reject(new Error("not completed promise 1"));
  }
});

const promise2 = new Promise((re resolve(value: any): void
    resolve('completed promise 2')
})

// console.log(promise1);
promise1
```

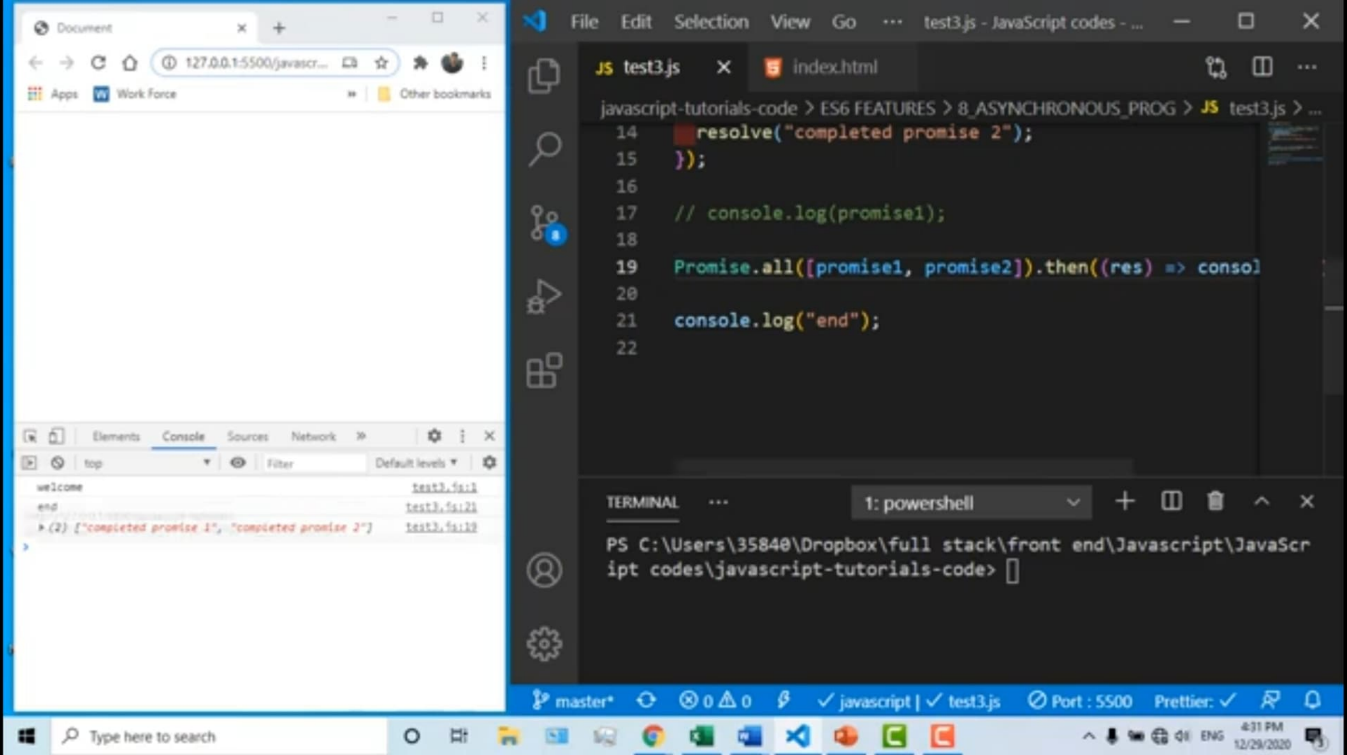
```
welcome                          test3.js:1
end                              test3.js:21
not completed promise 1          test3.js:19
>
```

```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code> 
```



```
const promise2 = new Promise((re
    resolve('completed promise 2')
})
```

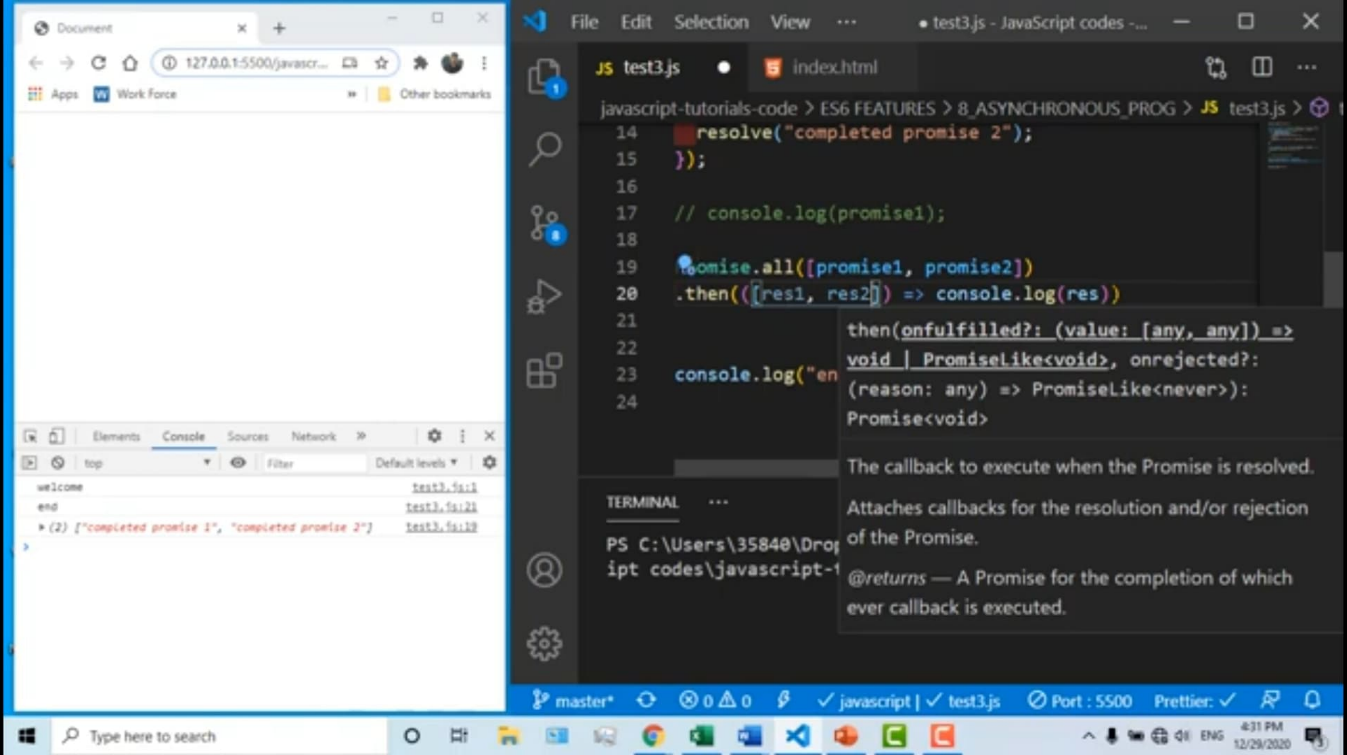
```
resolve(value: any): void
```



```
    resolve("completed promise 2");
});

// console.log(promise1);

Promise.all([promise1, promise2]).then((res) => consol

console.log("end");
```

```
welcome                                                  test3.js:1
end                                                      test3.js:21
▸ (2) ["completed promise 1", "completed promise 2"]     test3.js:19
```

```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code>
```



```
welcome                                              test3.js:1
end                                                  test3.js:21
▸ (2) ["completed promise 1", "completed promise 2"]  test3.js:19
>
```

```
    resolve("completed promise 2");
});

// console.log(promise1);

Promise.all([promise1, promise2])
.then(([res1, res2]) => console.log(res))


console.log("en
```

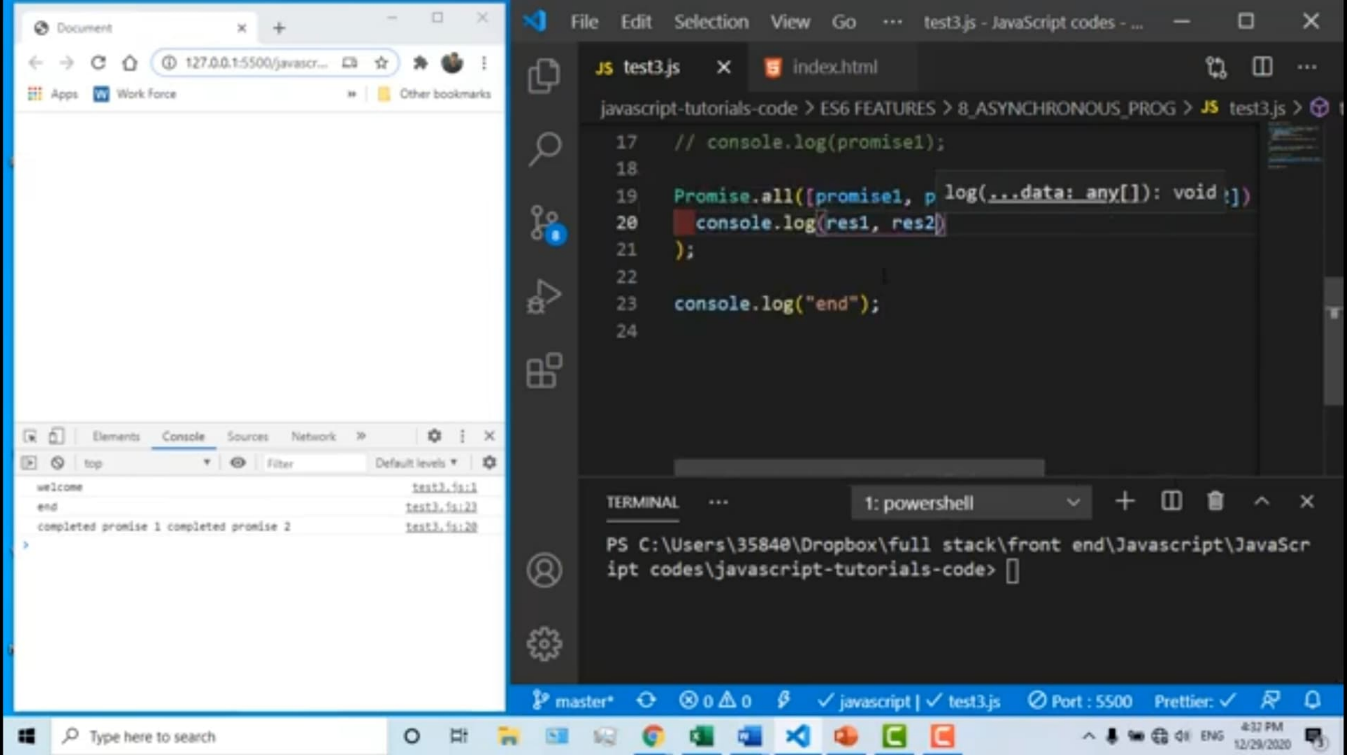
```
then(onfulfilled?: (value: [any, any]) =>
void | PromiseLike<void>, onrejected?:
(reason: any) => PromiseLike<never>):
Promise<void>
```

The callback to execute when the Promise is resolved.

Attaches callbacks for the resolution and/or rejection of the Promise.

@returns — A Promise for the completion of which ever callback is executed.

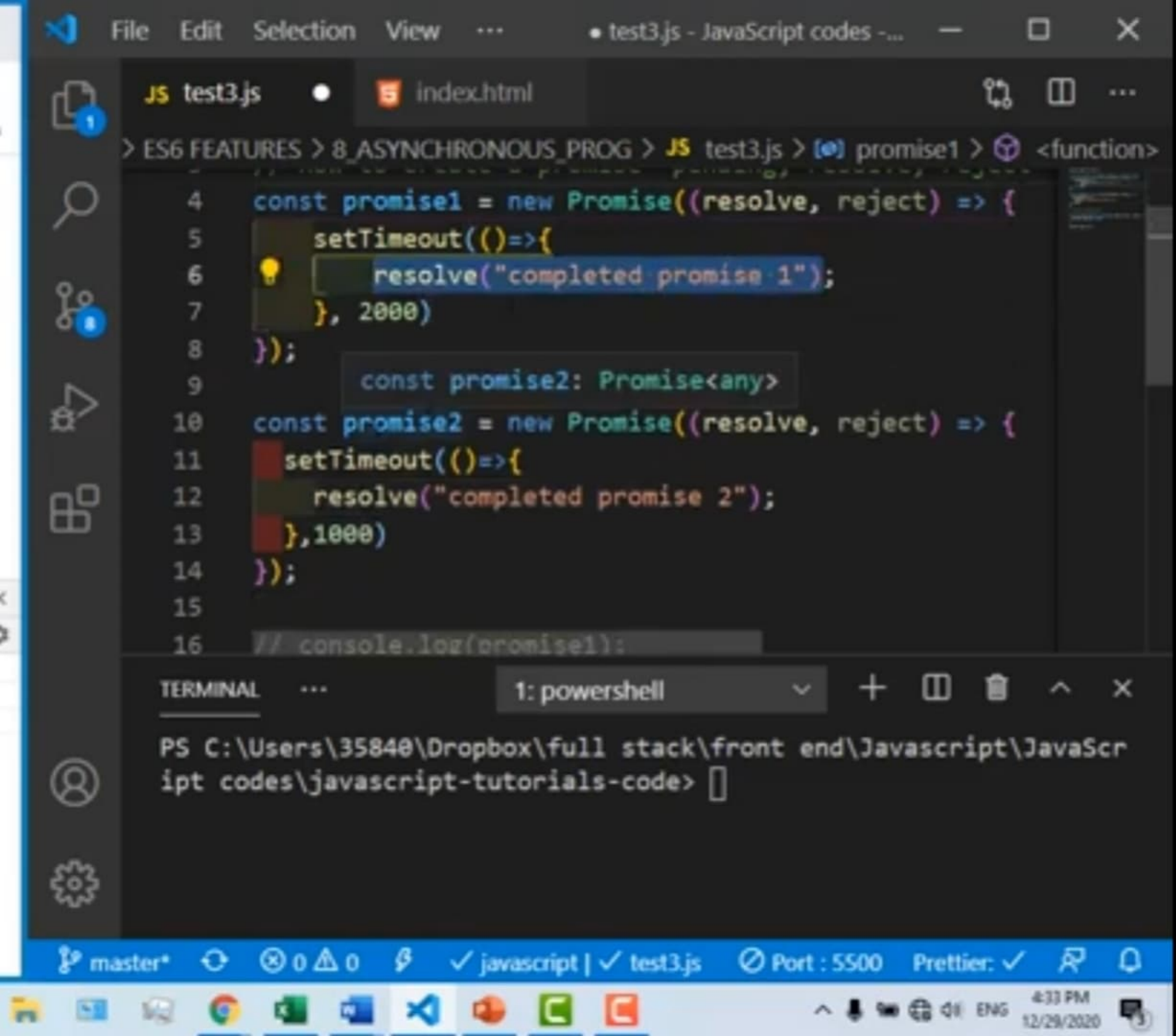
```
PS C:\Users\35840\Drop
ipt codes\javascript-t
```



```
// console.log(promise1);

Promise.all([promise1, p                                    2])
  console.log(res1, res2)
);

console.log("end");

```

```
welcome                                         test3.js:1
end                                             test3.js:23
completed promise 1 completed promise 2         test3.js:20
>
```

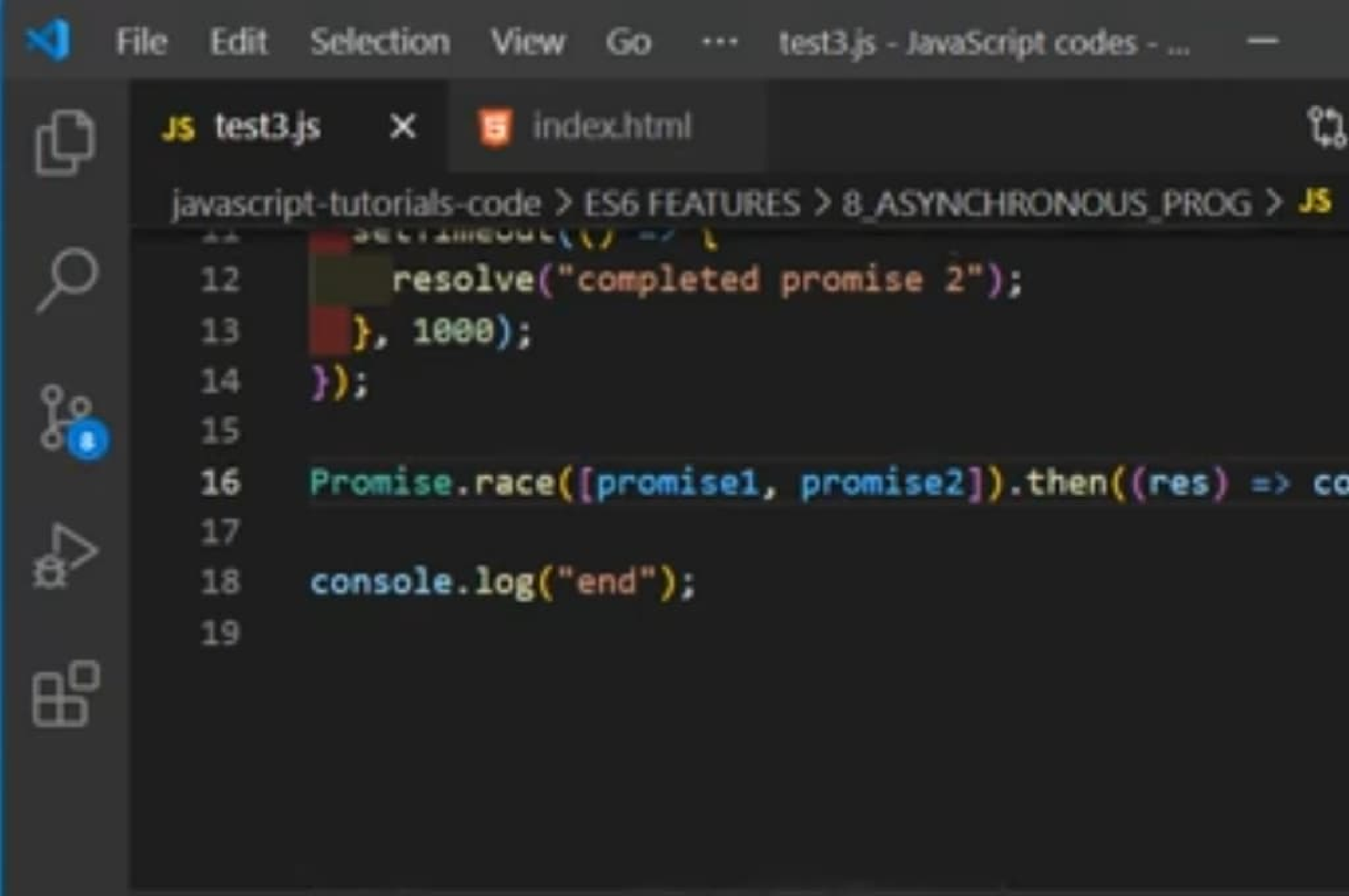
```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code>
```



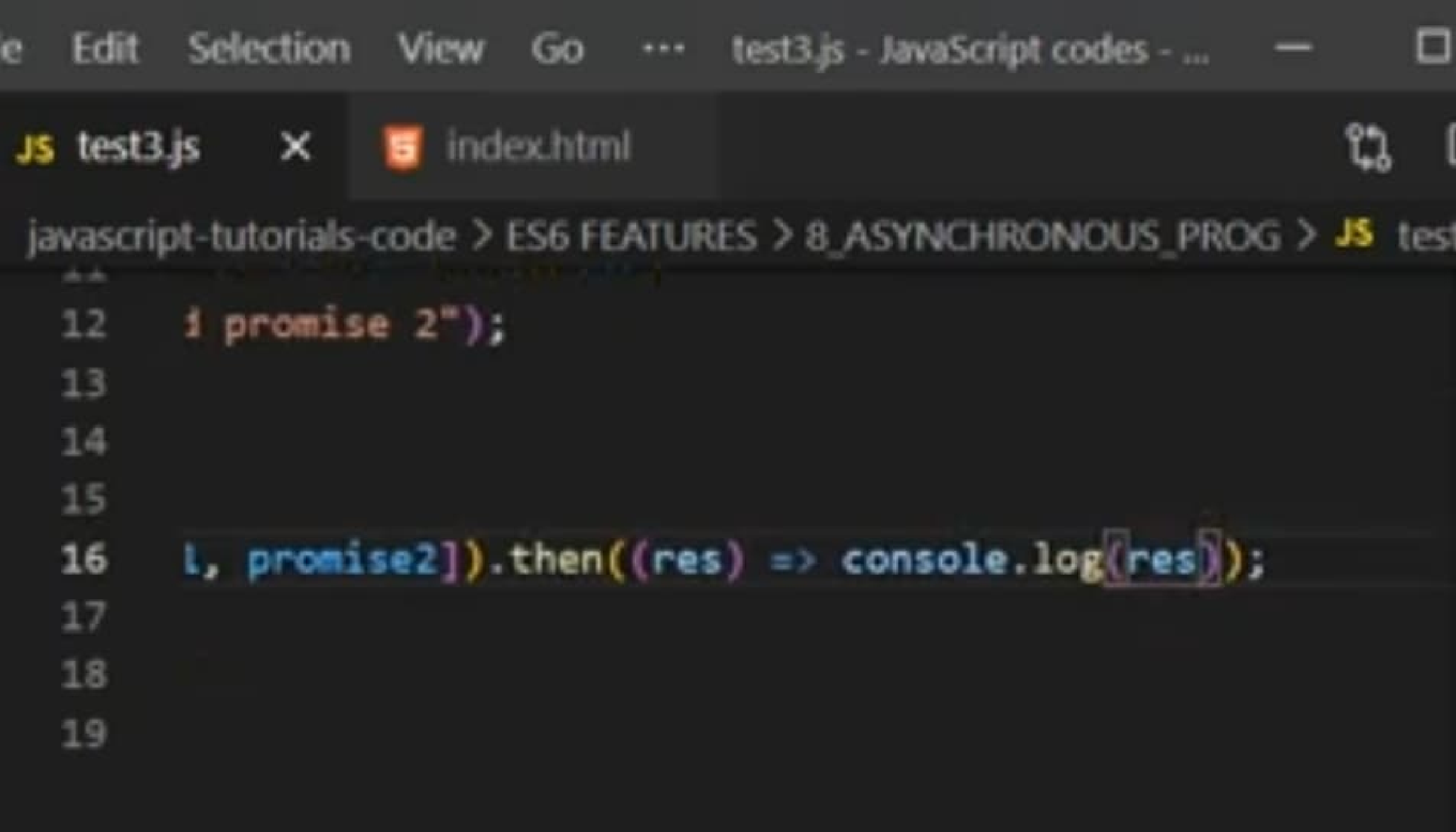
File
Edit
Selection
View
● test3.js - JavaScript codes -...
JS test3.js
index.html
> ES6 FEATURES > 8_ASYNCHRONOUS_PROG > JS test3.js > promise1 > <function>
const promise2: Promise<any>
TERMINAL
1: powershell
master*
0 0
javascript | test3.js
Port : 5500
Prettier: ✓
ENG
4:33 PM
12/29/2020

```
const promise1 = new Promise((resolve, reject) => {
    setTimeout(()=>{
        resolve("completed promise 1");
    }, 2000)
});

const promise2 = new Promise((resolve, reject) => {
  setTimeout(()=>{
    resolve("completed promise 2");
  },1000)
});

// console.log(promise1);
```

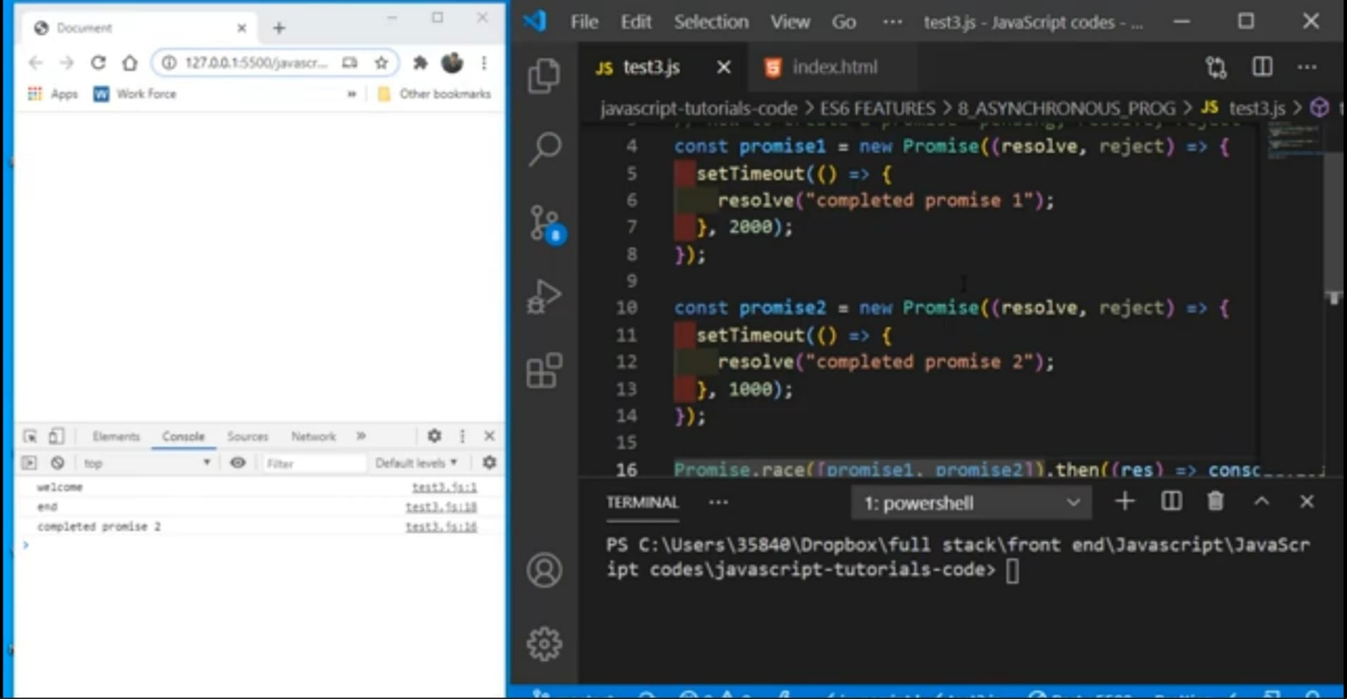
```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScript codes\javascript-tutorials-code> 
```


File
Edit
Selection
View
Go
···
test3.js - JavaScript codes - ...
JS test3.js
index.html
javascript-tutorials-code > ES6 FEATURES > 8_ASYNCHRONOUS_PROG > JS

```
    setTimeout(() => {
        resolve("completed promise 2");
    }, 1000);
});

Promise.race([promise1, promise2]).then((res) => co

console.log("end");

```


e Edit Selection View Go ···
test3.js - JavaScript codes - ...
JS test3.js ×
index.html
javascript-tutorials-code > ES6 FEATURES > 8_ASYNCHRONOUS_PROG > JS test

```
i promise 2");



i, promise2]).then((res) => console.log(res));
```

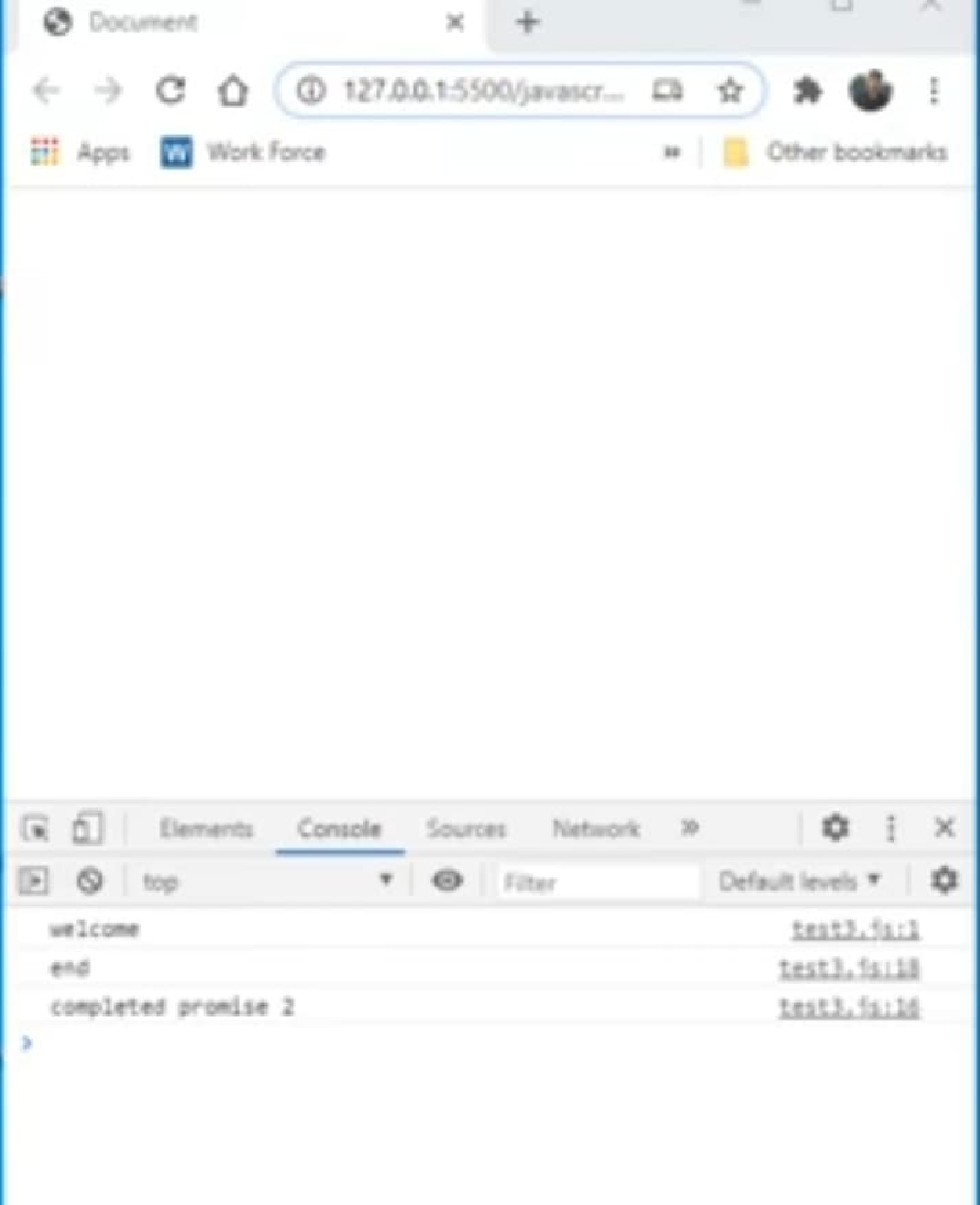


```
welcome                          test3.js:1
end                              test3.js:18
completed promise 2              test3.js:16
>
```

```javascript
const promise1 = new Promise((resolve, reject) => {
  setTimeout(() => {
    resolve("completed promise 1");
  }, 2000);
});

const promise2 = new Promise((resolve, reject) => {
  setTimeout(() => {
    resolve("completed promise 2");
  }, 1000);
});

Promise.race([promise1, promise2]).then((res) => conso
```

```
PS C:\Users\35840\Dropbox\full stack\front end\Javascript\JavaScr
ipt codes\javascript-tutorials-code> 
```

No connection



```
welcome                                                      test3.js:1
end                                                          test3.js:18
completed promise 2                                          test3.js:16
>
```