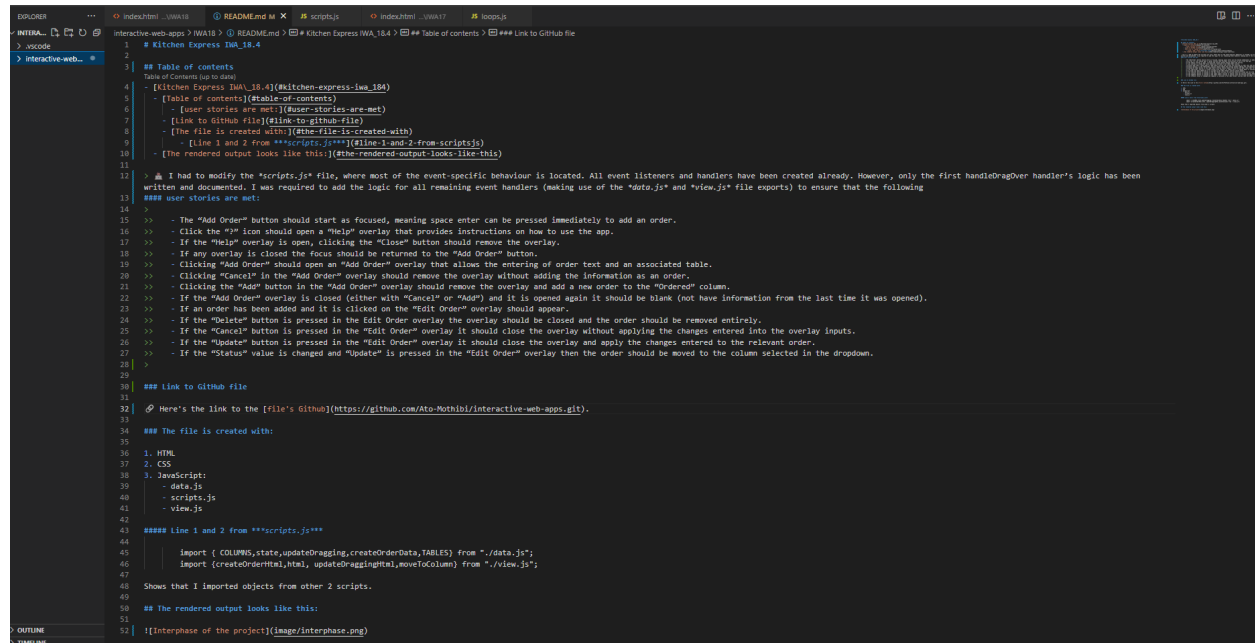


DWA_03.4 Knowledge Check_DWA3.1

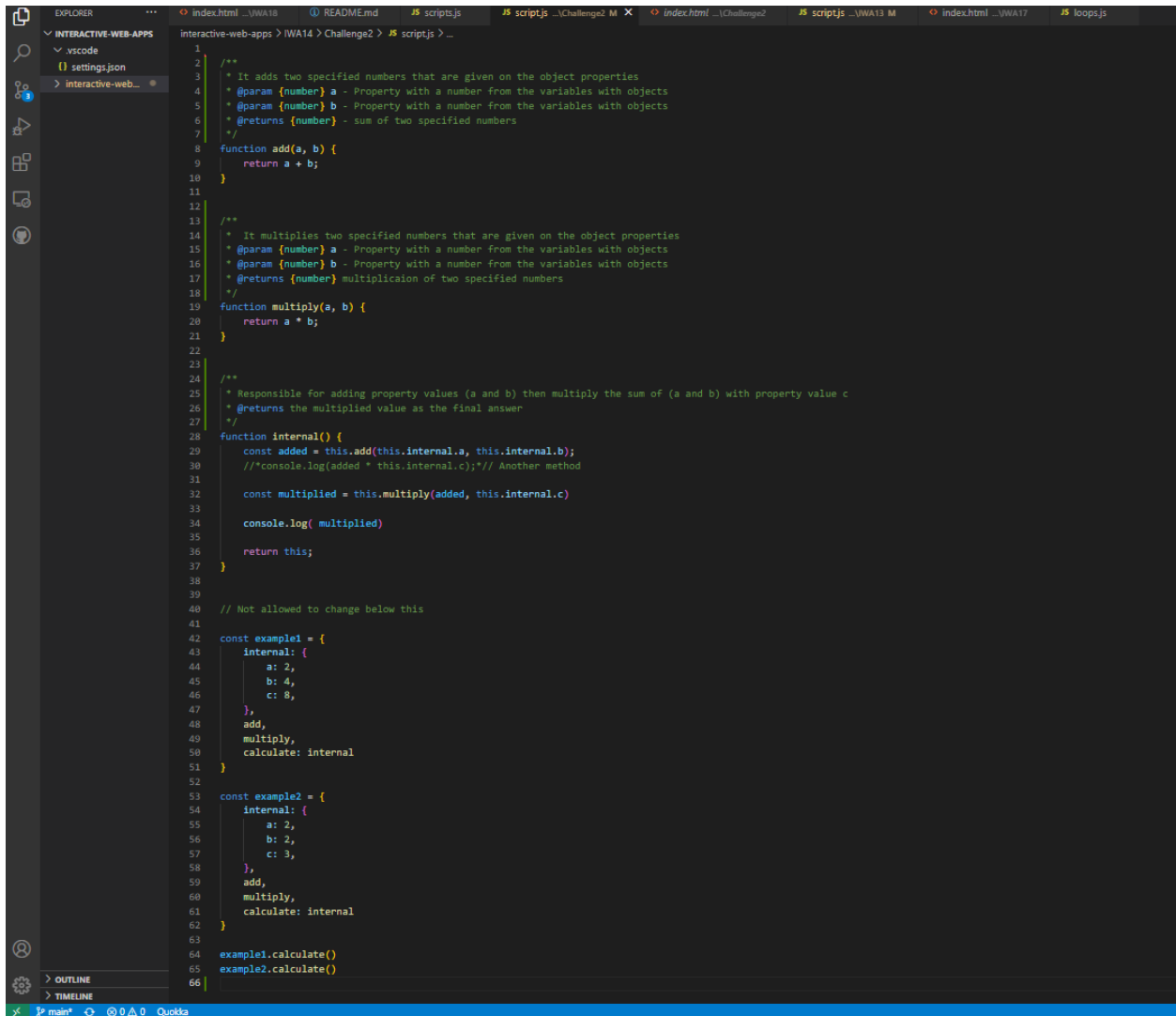
1. Please show how you applied a Markdown File to a piece of your code.



```
1 # Kitchen Express IWA_18.4
2
3 ## Table of contents
4 Table of Contents (up to date)
5 - [Kitchen Express IWA_18.4](kitchen-express-iwa_184)
6 - [Table of contents](#table-of-contents)
7 - [User stories are met](#user-stories-are-met)
8 - [Link to GitHub file](#link-to-github-file)
9 - [The file is created with](#the-file-is-created-with)
10 - [Line 1 and 2 from **scripts.js**](#line-1-and-2-from-scriptsjs)
11 - [The rendered output looks like this](#the-rendered-output-looks-like-this)
12
13 > I had to modify the *scripts.js* file, where most of the event-specific behaviour is located. All event listeners and handlers have been created already. However, only the first handleDragOver handler's logic has been
14 written and documented. I was required to add the logic for all remaining event handlers (making use of the *data.js* and *view.js* file exports) to ensure that the following
15 user stories are met:
16
17 > - The "Add Order" button should start as focused, meaning space enter can be pressed immediately to add an order.
18 > - Click the "p" icon should open a "Help" overlay that provides instructions on how to use the app.
19 > - If the "Help" overlay is open, clicking the "Close" button should remove the overlay.
20 > - If any overlay is closed the focus should be returned to the "Add Order" button.
21 > - Clicking "Add Order" should open an "Add Order" overlay that allows the entering of order text and an associated table.
22 > - Clicking "Cancel" in the "Add Order" overlay should remove the overlay and add a new order to the "Orders" column.
23 > - Clicking the "Add" button in the "Add Order" overlay should remove the overlay and add a new order to the "Orders" column.
24 > - If the "Add Order" overlay is closed (either with "Cancel" or "Add") and it is opened again it should be blank (not have information from the last time it was opened).
25 > - If an order has been added and it is clicked on the "Edit Order" overlay should appear.
26 > - If the "Delete" button is pressed in the "Edit Order" overlay it should close the overlay and the order should be removed entirely.
27 > - If the "Cancel" button is pressed in the "Edit Order" overlay it should close the overlay without applying the changes entered into the overlay inputs.
28 > - If the "Update" button is pressed in the "Edit Order" overlay it should close the overlay and apply the changes entered to the relevant order.
29 > - If the "Status" value is changed and "Update" is pressed in the "Edit Order" overlay then the order should be moved to the column selected in the dropdown.
30
31 ## Link to GitHub file
32 > Here's the link to the [file's GitHub](https://github.com/Ata-Mothibi/interactive-web-apps.git).
33
34 ## The file is created with:
35
36 1. HTML
37 2. CSS
38 3. JavaScript:
39 - data.js
40 - scripts.js
41 - view.js
42
43 ##### Line 1 and 2 from **scripts.js**
44
45 import { COLUMNS, state, updateDragging, createOrderData, TABLES } from "../data.js";
46 import { createOrderForm, HTML, updateDraggingHTML, moveColumn } from "../view.js";
47
48 Shows that I imported objects from other 2 scripts.
49
50 ## The rendered output looks like this:
51
52 ![[Interphase of the project](image/interphase.png)]
```

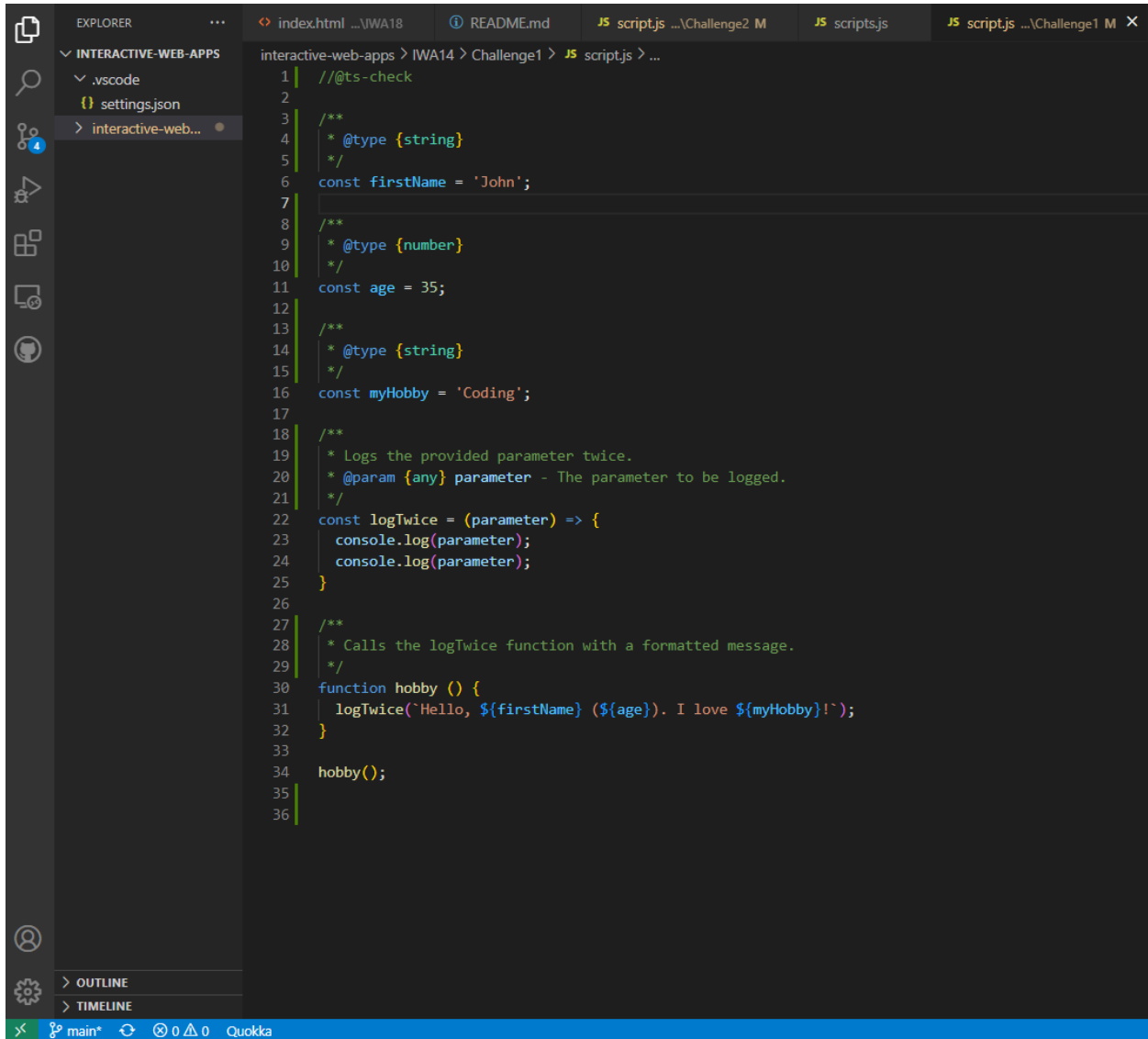
TB_Mothibi DWA_3.1

2. Please show how you applied JSDoc Comments to a piece of your code.



```
1
2
3 /**
4  * It adds two specified numbers that are given on the object properties
5  * @param {number} a - Property with a number from the variables with objects
6  * @param {number} b - Property with a number from the variables with objects
7  * @returns {number} - sum of two specified numbers
8  */
9 function add(a, b) {
10   return a + b;
11 }
12
13
14 /**
15  * It multiplies two specified numbers that are given on the object properties
16  * @param {number} a - Property with a number from the variables with objects
17  * @param {number} b - Property with a number from the variables with objects
18  * @returns {number} multiplication of two specified numbers
19  */
20 function multiply(a, b) {
21   return a * b;
22 }
23
24
25 /**
26  * Responsible for adding property values (a and b) then multiply the sum of (a and b) with property value c
27  * @returns the multiplied value as the final answer
28  */
29 function internal() {
30   const added = this.add(this.internal.a, this.internal.b);
31   //console.log(added * this.internal.c); // Another method
32
33   const multiplied = this.multiply(added, this.internal.c)
34
35   console.log( multiplied)
36
37   return this;
38 }
39
40 // Not allowed to change below this
41
42 const example1 = {
43   internal: {
44     a: 2,
45     b: 4,
46     c: 8,
47   },
48   add,
49   multiply,
50   calculate: internal
51 }
52
53 const example2 = {
54   internal: {
55     a: 2,
56     b: 2,
57     c: 3,
58   },
59   add,
60   multiply,
61   calculate: internal
62 }
63
64 example1.calculate()
65 example2.calculate()
66
```

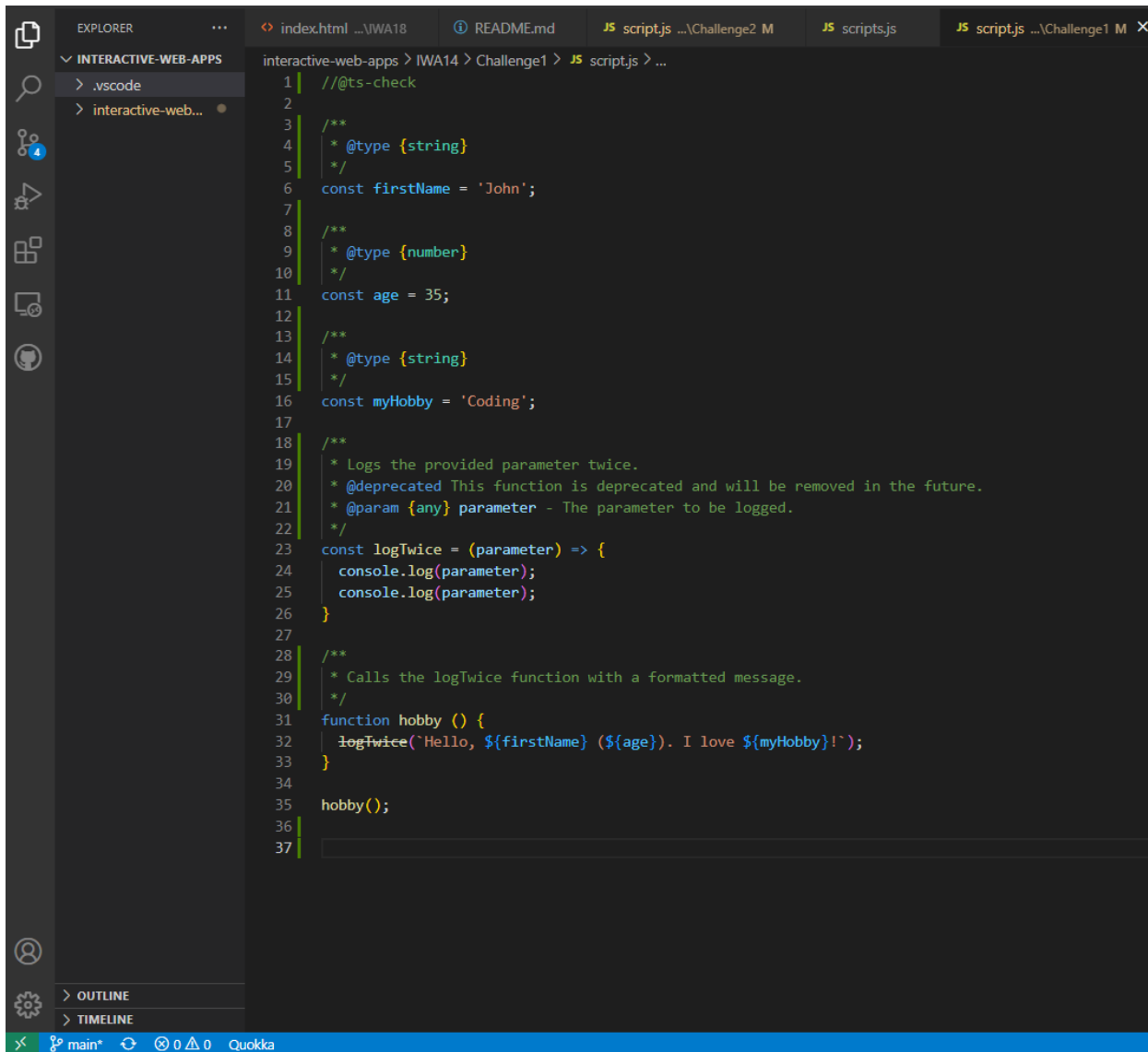
3. Please show how you applied the @ts-check annotation to a piece of your code.



```
1  //@ts-check
2
3  /**
4   * @type {string}
5   */
6  const firstName = 'John';
7
8  /**
9   * @type {number}
10  */
11  const age = 35;
12
13  /**
14   * @type {string}
15   */
16  const myHobby = 'Coding';
17
18  /**
19   * Logs the provided parameter twice.
20   * @param {any} parameter - The parameter to be logged.
21   */
22  const logTwice = (parameter) => {
23    console.log(parameter);
24    console.log(parameter);
25  }
26
27  /**
28   * Calls the logTwice function with a formatted message.
29   */
30  function hobby () {
31    logTwice(`Hello, ${firstName} (${age}). I love ${myHobby}!`);
32  }
33
34  hobby();
35
36
```

4. As a BONUS, please show how you applied any other concept covered in the 'Documentation' module.

Used the `@deprecated`: This helps to provide a clear indication to developers that the `logTwice` function should no longer be used and suggests finding an alternative solution.



```
1 | //@ts-check
2 |
3 | /**
4 |  * @type {string}
5 |  */
6 | const firstName = 'John';
7 |
8 | /**
9 |  * @type {number}
10 |  */
11 | const age = 35;
12 |
13 | /**
14 |  * @type {string}
15 |  */
16 | const myHobby = 'Coding';
17 |
18 | /**
19 |  * Logs the provided parameter twice.
20 |  * @deprecated This function is deprecated and will be removed in the future.
21 |  * @param {any} parameter - The parameter to be logged.
22 |  */
23 | const logTwice = (parameter) => {
24 |   console.log(parameter);
25 |   console.log(parameter);
26 | }
27 |
28 | /**
29 |  * Calls the logTwice function with a formatted message.
30 |  */
31 | function hobby () {
32 |   logTwice(`Hello, ${firstName} (${age}). I love ${myHobby}!`);
33 | }
34 |
35 | hobby();
36 |
37 |
```