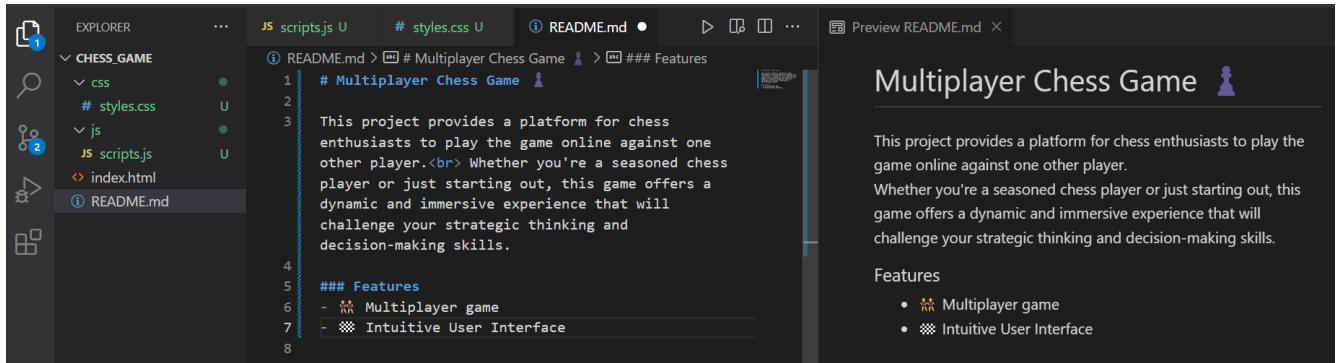


DWA_03.4 Knowledge Check_DWA3.1

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



The screenshot shows the VS Code interface with the Explorer sidebar on the left displaying a project structure for 'CHESS_GAME' containing files like 'scripts.js', 'styles.css', 'index.html', and 'README.md'. The main editor area shows the content of 'README.md' which describes a 'Multiplayer Chess Game' and lists features such as 'Multiplayer game' and 'Intuitive User Interface'. To the right, a preview window titled 'Preview README.md' shows the rendered markdown content with a chess knight icon.

```
scripts.js # styles.css U README.md ...
① README.md > # Multiplayer Chess Game 🕸️ > # Features
② # Multiplayer Chess Game 🕸️
1 This project provides a platform for chess
2 enthusiasts to play the game online against one
3 other player. Whether you're a seasoned chess
4 player or just starting out, this game offers a
5 dynamic and immersive experience that will
6 challenge your strategic thinking and
7 decision-making skills.
8
9 ## Features
10 - 🕸️ Multiplayer game
11 - 🌐 Intuitive User Interface
```

Multiplayer Chess Game 🕸️

This project provides a platform for chess enthusiasts to play the game online against one other player. Whether you're a seasoned chess player or just starting out, this game offers a dynamic and immersive experience that will challenge your strategic thinking and decision-making skills.

Features

- 🕸️ Multiplayer game
- 🌐 Intuitive User Interface

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

```
/** 
 * Moves the bishop chess piece diagonally (forwards and backwards)
 * across the board.
 * @param {Object} newCell - The chess board cell that the bishop piece
 * is moving to.
 */
export const moveBishop = (newCell) => {};
```

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

```
//@ts-check

/**
 * Adds two numbers together and returns a string that states the sum.
 * @param {number} num1 - The first number.
 * @param {number} [num2] - The second number.
 * @returns {string} - A string stating the sum.
 */
const add = (num1, num2 = 0) => {
  return `The sum is: ${num1 + num2}`;
};

add(1, "3")
```

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

How to create custom types. In this example I created a custom shopping list type

```
/**
 * Shopping list
 * @typedef {Object} ShoppingItems
 * @prop {Array} itemNames
 * @prop {number} amount
 */

/**
 * First trip shopping list
 * @type {ShoppingItems}
 */
const shoppingList1 = {
  itemNames: ["carrot", "banana"],
  amount: 3,
};
```