Node Programming Question

Given the following Node project for a grocery shop, with the following `package.json` file:

```
{
  "name": "shop-app",
  "version": "1.0.0",
  "description": "SD540 Midterm exam",
  "scripts": {
    "shop": "ts-node shop.ts"
  },
  "author": "Asaad Saad",
  "devDependencies": {
    "@types/node": "^20.11.20",
    "ts-node": "^10.9.2"
  },
  "dependencies": {
    "dotenv": "^16.4.5"
  }
}
```

And the following `./data.json` file:

```
[
    { "name": "Tomato", "category": "vegetable", "price": 3, "amount": 20 },
    { "name": "Apple", "category": "fruit", "price": 2, "amount": 51 },
    { "name": "Banana", "category": "fruit", "price": 1, "amount": 48 },
    { "name": "Potato", "category": "vegetable", "price": 1, "amount": 17 },
    { "name": "Onion", "category": "vegetable", "price": 2, "amount": 3 }
]
```

And the following `./.env` file:

```
DATA_FILE_NAME="data.json"
```

And the following `./types.ts` file:

```
export interface Product {
    name: string;
    category: string;
    price: number;
    amount: number;
}
```

(1 point) Write the command to execute and run the project:

. . . . . .

Given the following `node:fs` API:

- `readFileSync(file: string): Buffer`
- `writeFileSync(file :string, data: string): undefined`

The following test is used to verify the code functionality:

```
// Creating instances of Shop and Cart
const shop = new Shop();
const cart = new Cart();

// Listen and print all messages from Shop and Cart
shop.on('message', console.log);
cart.on('message', console.log);

cart.add('Tomato', 4); // emits: "Added 4 Tomato to cart"
cart.add('Banana', 6); // emits: "Added 6 Banana to cart"
cart.add('Onion', 6); // emits: "Could not add 6 Onion to cart"

// Check out the cart
cart.checkout(); // emits: "Checked out successfully!"
```

(10 Points) Complete the code parts of `./shop.ts`:

1. **The `Shop` class constructor: You must read the data from `./data.json` file and assign it to the protected property `products`. Make sure you implement error handling for potential issues like file not found.**

2. **The `Cart` class, `add()` method: You must check if the product is available and if the amount is at or below the stock, if the requested amount is available, add the product to the cart and emit one of these two messages:**
   - `Added ${amount} ${product_name} to cart`
   - `Could not add ${amount} ${product_name} to cart`

3. **The `Shop` class, `checkout()` method: You must decrease the amount of all of the purchased products and then persist the products data back to the file system, and emit a confirmation message:**
   - `Checked out successfully!`

```
import events from 'node:events';
import { join } from 'node:path';
import { readFileSync, writeFileSync } from 'node:fs';

import 'dotenv/config';

import { Product } from './types';

class Shop extends events {

    protected products: Product[] = []; // all shop products
    #dataPath = join(__dirname, process.env.DATA_FILE_NAME as string);

    constructor() {
        super();
        // Write code here

    }
    checkout(cart: Product[]): void {
        // Write code here

    };
}

class Cart extends Shop {
    #cart: Product[] = []; // shopping cart products

    constructor() {
        super();
    }

    add(product_name: string, amount: number): void {
        // Write code here

    }

    checkout(): void {
        super.checkout(this.#cart);
        this.#cart = [];
    }

}
```