Java Intermediate Level Question: OOP, Collection, Concurrency, and File Handling

# Question:

You are tasked with building a simple multi-threaded Java application to manage a library system. The application should include the following features:

1. \*\*OOP Principles\*\*:  
- Define a `Book` class with the following properties: `title`, `author`, `isbn`, `publishedYear`, and `isCheckedOut` (boolean). The `Book` class should have methods for checking out and returning books.  
- Create a `Library` class that stores a collection of books. Implement methods for adding books, removing books, and listing all books. The `Library` class should also implement a method to search for books by their ISBN.  
- Implement a `User` class that represents a library user with properties such as `userId`, `name`, and a list of checked-out books.

2. \*\*Concurrency\*\*:  
- Implement multi-threading in the `Library` class to handle concurrent checkouts and returns. Use synchronization to ensure thread safety when modifying the list of books, particularly during checkout and return operations.

3. \*\*Collection Framework\*\*:  
- Use a `HashMap` to store books in the `Library` class, where the key is the ISBN of the book and the value is the `Book` object. This will allow efficient lookup and modification of books based on ISBN.  
- Implement a method that returns a list of all available books (books that are not checked out).

4. \*\*File Handling and Streams\*\*:  
- Implement functionality to read and write the book data to/from a file (CSV or JSON format). The file should store book details, including whether the book is checked out or available.  
- Use Java Streams (or `BufferedReader` and `BufferedWriter`) to read and write the book list efficiently. Ensure that the file handling is done asynchronously to prevent blocking the main thread.

Please provide the Java code to implement the program described above, focusing on the use of OOP, concurrency, collections, and file handling.