```cpp
#pragma once

#ifndef STUDENT_TO_DO_REGION
  /// Include necessary header files
  /// Hint:  Include what you use, use what you include

  #include <cstddef>   // size_t
  #include <string>
  #include <vector>

  #include "Book.hpp"
#endif


// Singleton Design Pattern
class BookDatabase
{
  public:
    // Get a reference to the one and only instance of the database
    static BookDatabase & instance();

    // Locate and return a reference to a particular record
    Book * find( const std::string & isbn );                                   // Returns a pointer to the item in the database if
                                                                               // found, nullptr otherwise

    // Queries
    std::size_t size() const;                                                  // Returns the number of items in the database

  private:
    BookDatabase            ( const std::string & filename );

    BookDatabase            ( const BookDatabase & ) = delete;                  // intentionally prohibit making copies
    BookDatabase & operator=( const BookDatabase & ) = delete;                  // intentionally prohibit copy assignments

    #ifndef STUDENT_TO_DO_REGION
      /// Private implementation details

      // Private helper function
      Book * find( const std::string & isbn, std::vector<Book>::iterator currentBook );

      // We'll look at alternatives to std::vector later
      std::vector<Book> _data;
    #endif
};
```