```
// c/lms.c
// library management system
// guide in readme.md

// include libraries
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// define books.txt as books
#define books "books.txt"

// function for functionality

// error message
void error();

// function to get the capacity
int get_capacity();

// function to give the book an unique id
int get_last_id();

// main menu
void main_menu();

// add book

// add book
void add_book();

// checks if database is there
int checkDB();

// create database
void createDB();

// append book
void append_book(int lastid);

// display books
void display_books();

// clear
void clear();

// remove books

// remove book
void remove_book();
```

```c
// remove book by name
void remove_book_by_name();

// remove book by id
void remove_book_by_id();

// clear output
void clear();

// structures
struct book {
  int id;
  char name[200];
  char author[200];
};

// typedef of book
typedef struct book book;

int main() {
  printf("..........................Welcome To LIBRARY MANAGEMENT "
         "SYSTEM...........................\n");
  main_menu();
  return 0;
}

void main_menu() {
  printf("Guide: Enter the corresponding number to choose the menu only and "
         "press enter\n");
  int choice;

  while (1) {
    printf("   1.    Display Books\n     2.   Add Book\n     3.   Remove Book\n  "
           "  4.   clear\n    5.   Quit.\n");
    printf("Enter menu choice:  ");
    scanf("%d", &choice);

    switch (choice) {
    case 1: {
      display_books();
      break;
    }
    case 2: {
      add_book();
      break;
    }
    case 3: {
      remove_book();
      break;
    }
    case 4: {
      clear();
      break;
    }
    case 5: {
      return;
      break;
    }
    default: {
      error();
      break;
    }
    }
  }
}

void add_book() {
  while (1) {
    int lastid = get_last_id();
    if (checkDB() == 1) {
      append_book(lastid + 1);
      break;
    } else if (checkDB() == 0) {
      createDB();
      append_book(lastid + 1);
      break;
    } else {
      error();
    }
  }
}

void remove_book() {
  printf("Remove book by name or id (name,id):   ");
  char choice[4];
  fflush(stdin);
  scanf("%s", choice);
  while (1) {
    if (strcmp(choice, "name") == 0) {
      remove_book_by_name();
      break;
    } else if (strcmp(choice, "id") == 0) {
      remove_book_by_id();
      break;
    } else
      error();
  }
}

void clear() { system("clear"); }

void createDB() {
  FILE *file = fopen("books.txt", "w");

  printf("What is the capacity of your library (number of books - int): ");
  int capacity;
  scanf("%d", &capacity);

  fprintf(file, "%d\n", capacity);

  fclose(file);
}

void display_books() {
  if (checkDB()) {
    FILE *book = fopen("books.txt", "r");

    printf("\nLibrary Catalog:\n");
    printf("%-15s %-30s %-30s\n", "ID", "Book Name", "Author");
    printf("---------------------------------------------------------------
\n");
    int id;
    char name[200];
    char author[200];
    // Skip the first line
    fscanf(book, "%*[^\n]\n");
    // Read and print each subsequent line
    while (fscanf(book, "%d,%199[^,],%199[^\n]", &id, name, author) == 3) {
      printf("%-5d |%-30s|%-30s\n", id, name, author);
    }
    printf("---------------------------------------------------------------
\n");
    fclose(book);
  } else {
    printf("There is no database of books do you want to add it ?(1/0):
");
    int choice;
    scanf("%d", &choice);
    printf("%d\n", choice);
    if (choice == 1) {
      createDB();
    } else
      main_menu();
  }
}

int checkDB() {
  FILE *file = fopen("books.txt", "r");

  if (file != NULL) {
    fclose(file);
    return 1; // File exists
  } else {
    return 0; // File does not exist
  }
  return 69;
}

void append_book(int lastid) {
  FILE *file = fopen("books.txt", "a");
  book newbook;
  newbook.id = lastid;
  // input name of the book
  printf("Name of Book:   ");
  fflush(stdin);
  scanf(" %[^\n]%*c", newbook.name);

  // input author of the book
  printf("Author of the Book: ");
  fflush(stdin);
  scanf(" %[^\n]%*c", newbook.author);
  // write to file
  fprintf(file, "%d,%s,%s\n", newbook.id, newbook.name, newbook.author);

  fclose(file);
}

void remove_book_by_id() {
  int id;
  printf("Enter the ID of the book you want to remove: ");
  scanf("%d", &id);

  FILE *file = fopen("books.txt", "r");
  FILE *temp = fopen("temp.txt", "w");

  if (file == NULL || temp == NULL) {
    printf("Error opening file!");
    return;
  }

  int bookId;
  char name[200];
  char author[200];
  int capacity = get_capacity();
  printf("%d", capacity);

  if (id < 1 || id > capacity) {
    printf("Invalid book ID!\n");
    fclose(file);
    fclose(temp);
    return;
  }

  fscanf(file, "%d\n", &capacity);
  fprintf(temp, "%d\n", capacity);

  while (fscanf(file, "%d,%199[^,],%199[^\n]", &bookId, name, author) == 3)
{
    if (bookId != id) {
      fprintf(temp, "%d,%s,%s\n", bookId, name, author);
    }
  }
  fclose(file);
  fclose(temp);

  remove("books.txt");
  rename("temp.txt", "books.txt");

  printf("Book with id %d has been removed successfully!\n", id);
}

void remove_book_by_name() {
  FILE *file = fopen("books.txt", "r");
  FILE *temp = fopen("temp.txt", "w");

  if (file == NULL || temp == NULL) {
    printf("Error opening file!");
    return;
  }

  int bookId;
  char bookName[200];
  char author[200];
  char name[200];
  printf("Enter the name of the book you want to remove: ");
  fflush(stdin);
  scanf(" %[^\n]%*c", name);
  int capacity = get_capacity();

  // skip first line containing capacity
  fscanf(file, "%d\n", &capacity);
  fprintf(temp, "%d\n", capacity);

  while (fscanf(file, "%d,%199[^,],%199[^\n]", &bookId, bookName, author)
==
         3) {
    if (strcmp(bookName, name) != 0) {
      fprintf(temp, "%d,%s,%s\n", bookId, bookName, author);
    }
  }

  fclose(file);
  fclose(temp);

  remove("books.txt");
  rename("temp.txt", "books.txt");

  printf("Book with name %s has been removed successfully!\n", name);
}

void error() { printf("Invalid input\n"); }

int get_capacity() {
  FILE *file = fopen("books.txt", "r");
  int capacity = 0;
  fscanf(file, "%d", &capacity);
  return capacity;
}

int get_last_id() {
  FILE *boooks = fopen("books.txt", "r");
  book newbook;
  int lastid = 0;
  fscanf(boooks, "%*[^\n]\n");
  while (fscanf(boooks, "%d,%99[^,],%99[^\n]\n", &newbook.id, newbook.name,
                newbook.author) == 3) {
    lastid++;
  }
  fclose(boooks);
  return lastid;
}
```